Modeling & Control of Hybrid Systems

Chapter 7 — Symbolic methods [1]

Manuel Mazo Jr.

Utrecht, July 6th, 2020

[1] With (some) slides courtesy of Bart De Schutter, Alessandro Abate, and Majid Zamani

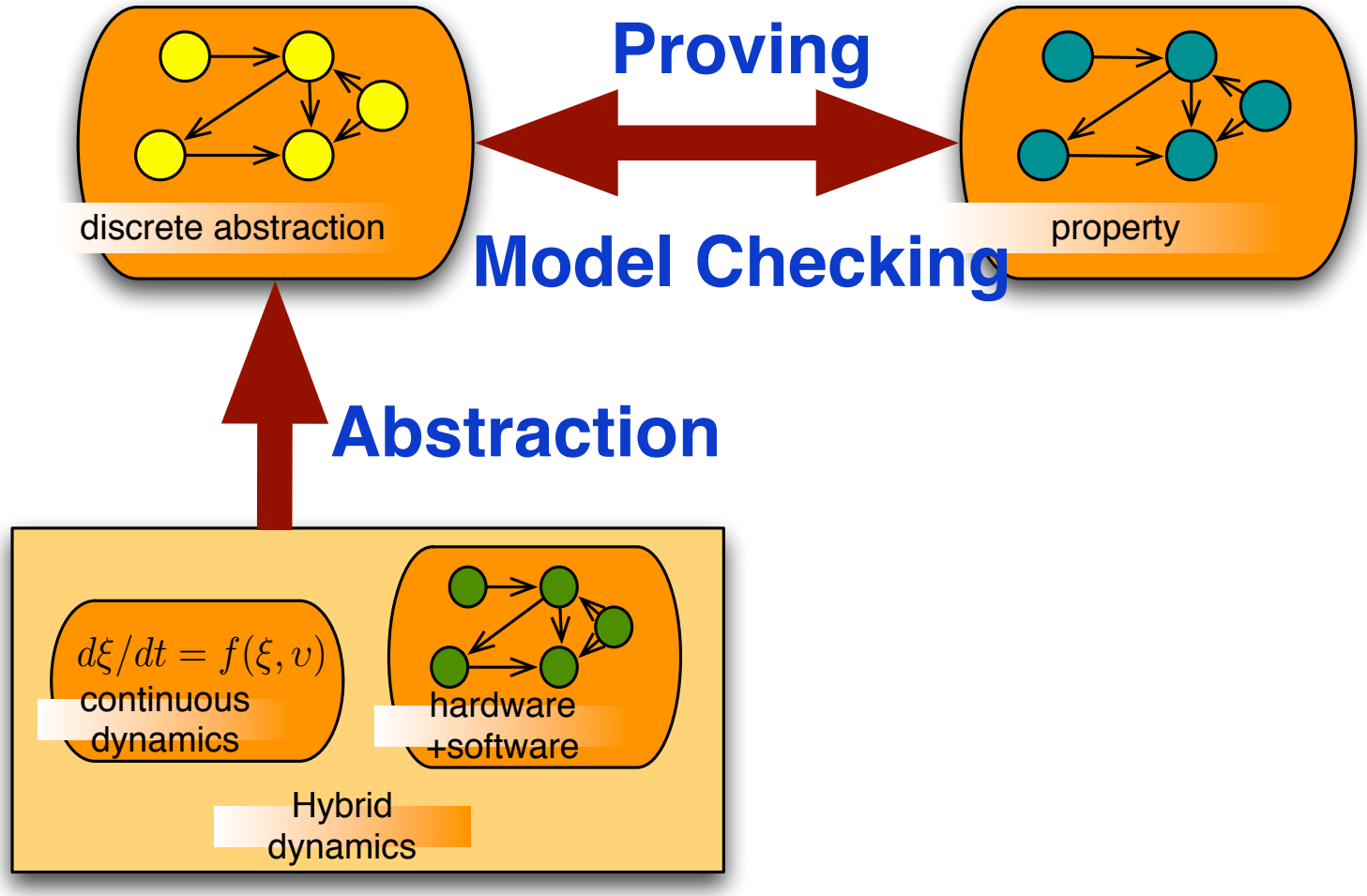
Outline

Outline

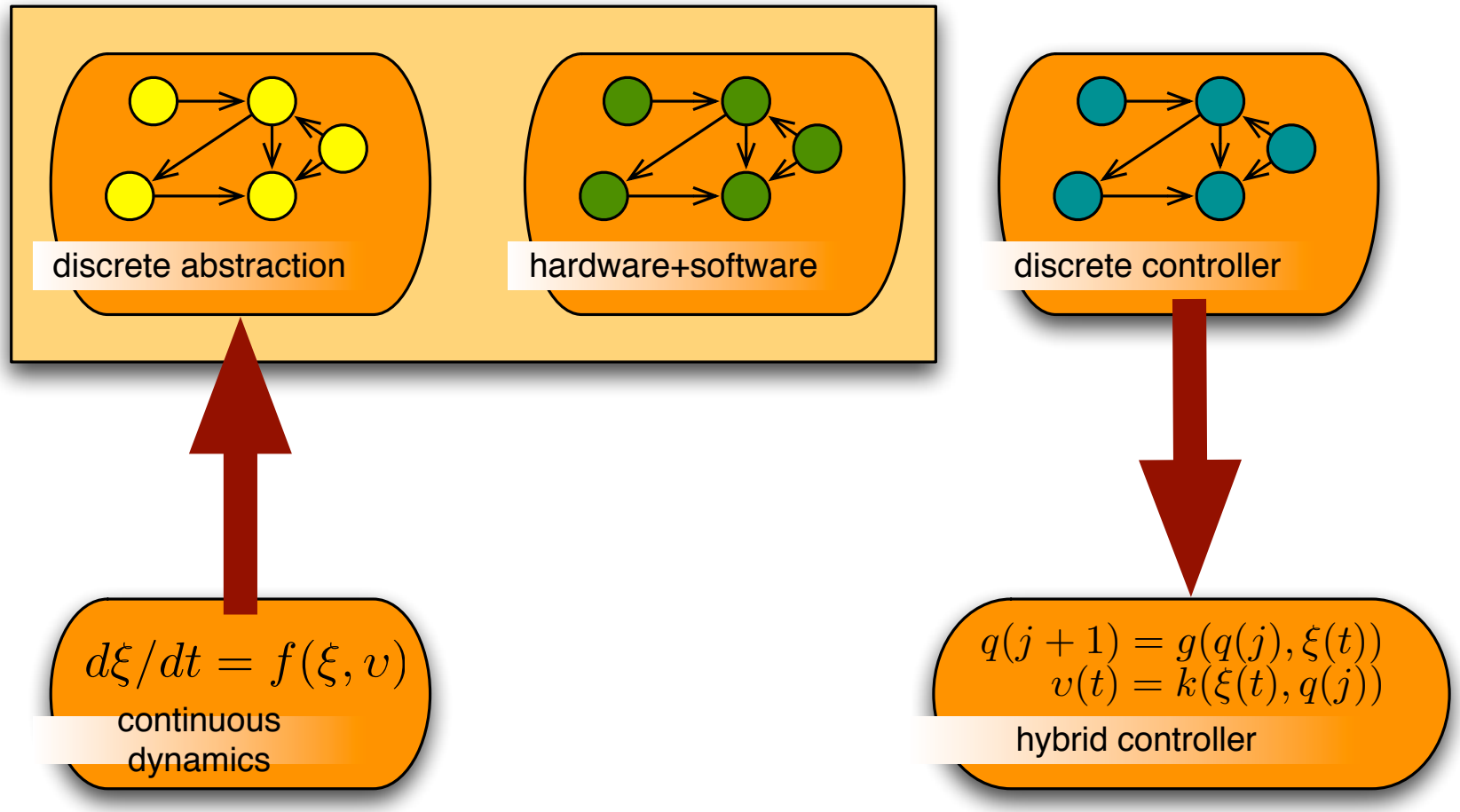
Introduction

- Model checking = process of automatically analyzing properties of systems by exploring their state space
- Finite state systems $\rightarrow$ properties can be investigated by systematically exploring states
  E.g., check whether particular set of states will be reached
- Not possible for hybrid systems since number of states is infinite
- However, for some hybrid systems one can find "equivalent" finite state system by partitioning state space into finite number of sets such that any two states in set exhibit similar behavior
  $\rightarrow$ analyze hybrid system by working with sets of partition
- Generation and analysis of finite partition can be carried out by computer

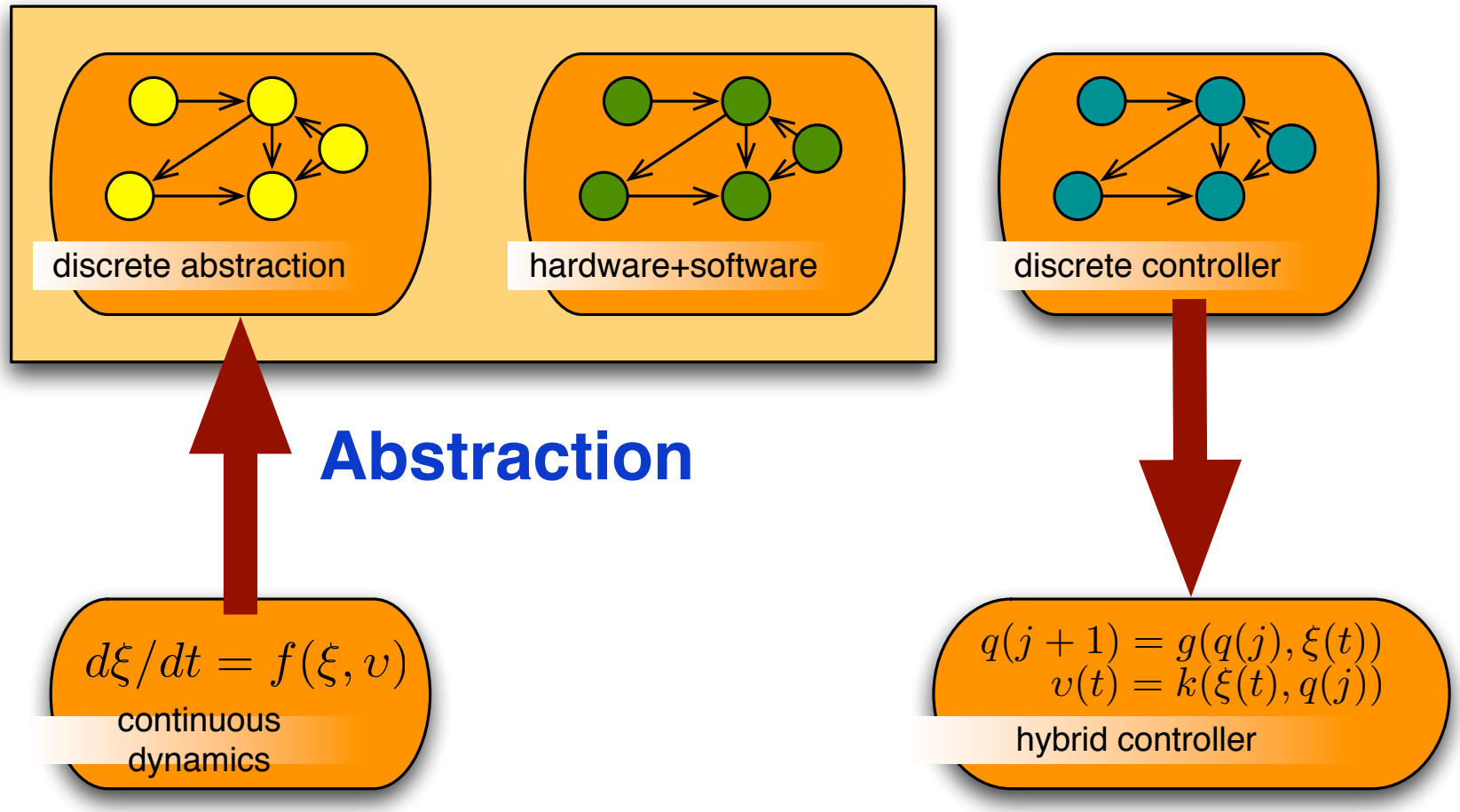
Verification of CPS



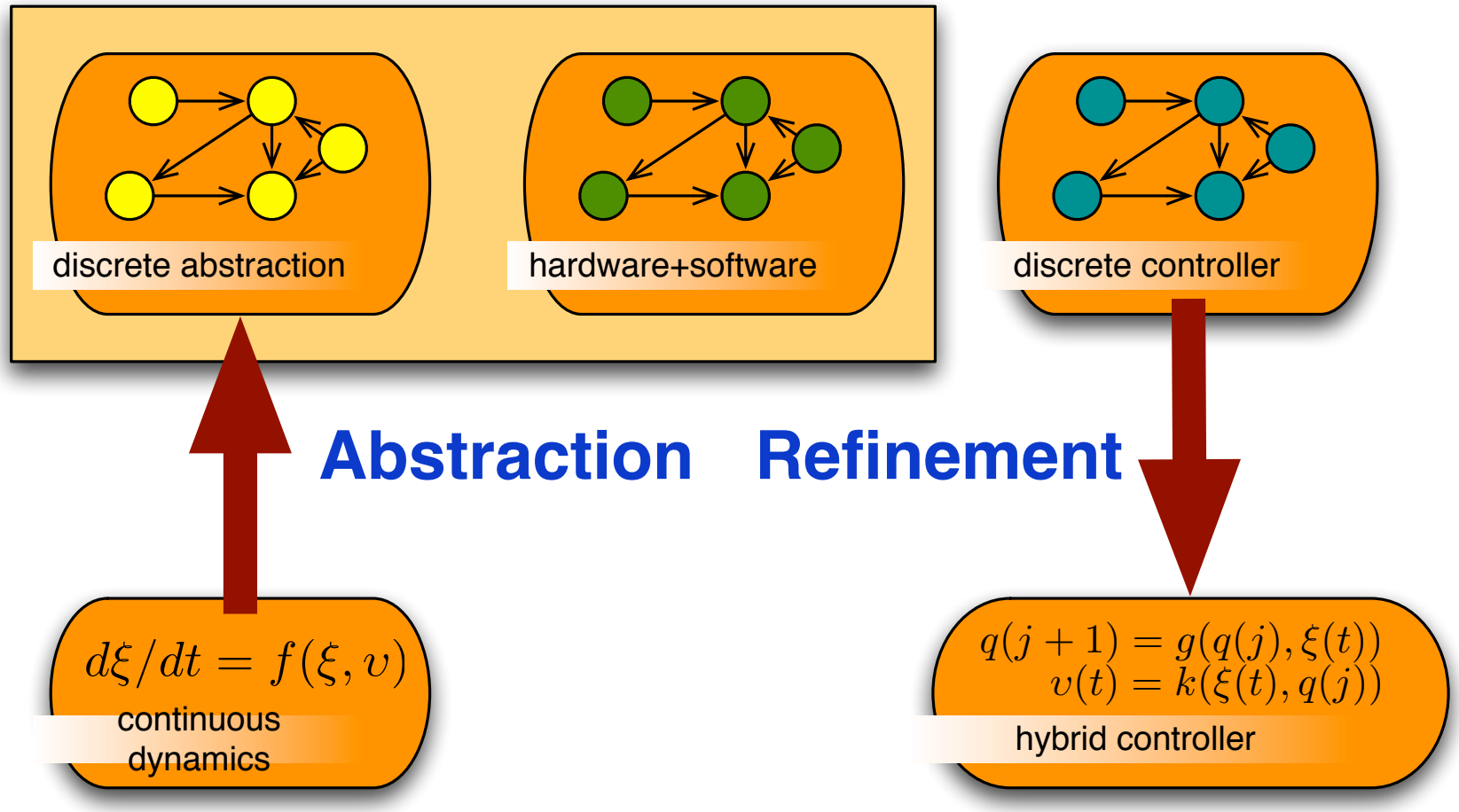
Correct-by-design synthesis of CPS



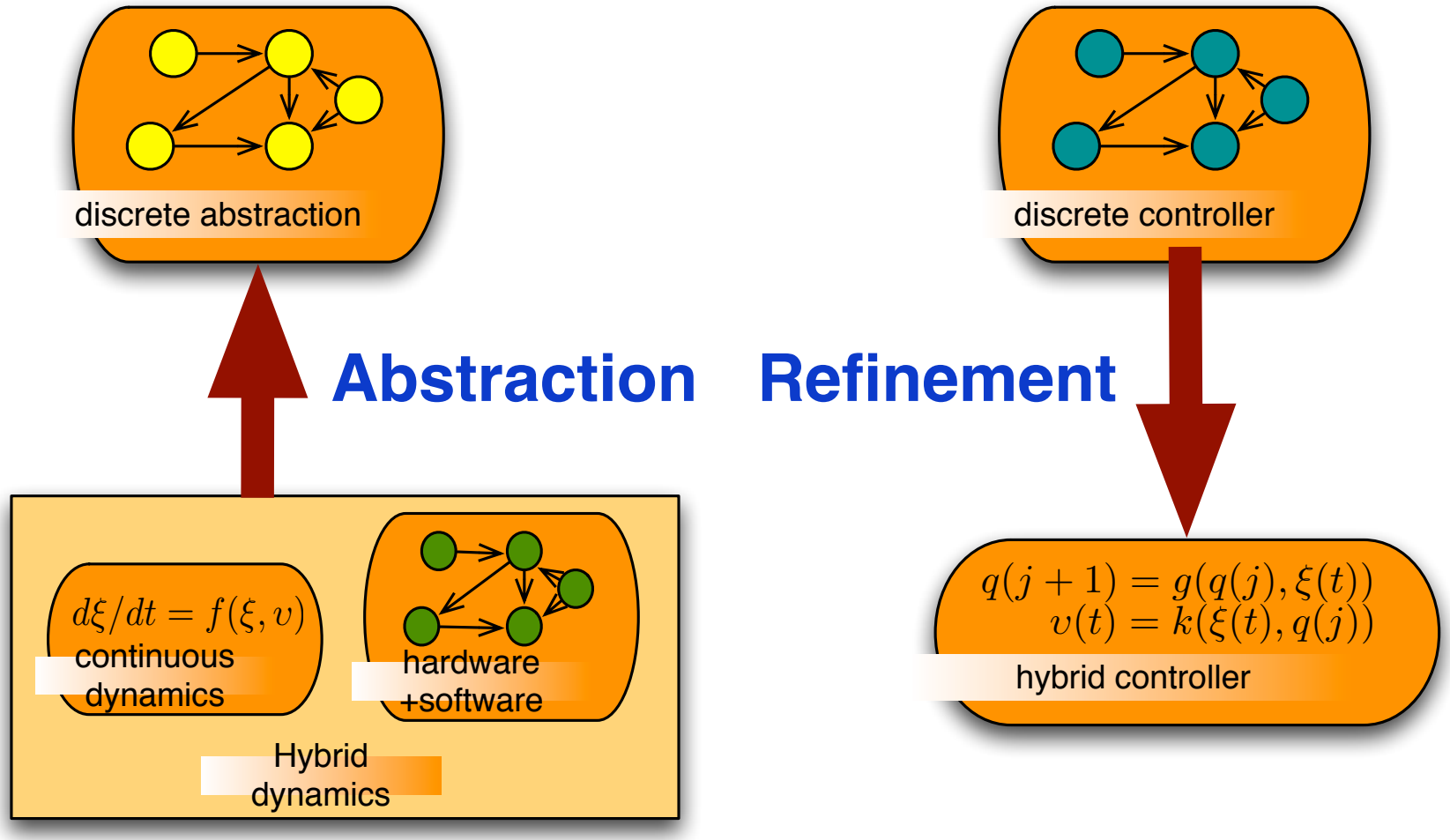
Correct-by-design synthesis of CPS



Correct-by-design synthesis of CPS



Correct-by-design synthesis of CPS



Outline

Generalized Transition Systems [PT09]

Definition (System)

A system S is a sextuple $(X, X_0, U, \longrightarrow, Y, H)$ consisting of:

- a set of states X;
- a set of initial states $X_0 \subseteq X$
- a set of inputs U;
- a transition relation $\longrightarrow \subseteq X \times U \times X$;
- a set of outputs Y (2^{AP});
- an output map $H : X \to Y$.
- (a set of final states $X_F \subseteq X$)

A system is said to be:

- *metric*, if the output set Y is equipped with a metric $\mathbf{d} : Y \times Y \to \mathbb{R}_0^+$;
- *countable*, if X is a countable set;
- *finite*, if X is a finite set.
- *blocking* if $\exists x \in X$ for which $\text{Post}(x) = \emptyset$

Notes

- Shorthand $x \to x'$ instead of $(x, x') \in \longrightarrow$

Definition

For $x \in X$, the set $Post(x)$ of *direct successors* of x is defined by

$$Post(x) = \{ x' \in X \mid x \to x' \}.$$

For $x \in X$, the set $Pre(x)$ of *direct predecessors* of x is defined by

$$Pre(x) = \{ x' \in X \mid x' \to x \}.$$

When we consider non-autonomous systems $|U| > 1$:
$\mathbf{Post}_{u_a}(x_a) = \left\{ x'_a \in X_a \mid x_a \xrightarrow[a]{u_a} x'_a \right\}$, $U_a(x_a) = \{ u \in U_a \mid |\mathbf{Post}_{u_a}(x_a)| \neq \varnothing \}$

Definition (Determinism)

A system S is *deterministic* if $\forall x \in X, u \in U(x)\ |(Post_u(x))| = 1$, where $|(\cdot)|$ denotes the cardinality of a set. Else S is non-deterministic. In the context of computer science often $|X_0| = 1$ is also required to talk about determinism.

Example: model of a traffic light

System description: A traffic light can be red, green, amber or black (not working). The traffic light might stop working at any time. After it has been repaired, it turns red. Initially, the light is red.

$$X = \{1, 2, 3, 4, 5\}$$

1 red
2 amber and red
3 green
4 amber
5 black

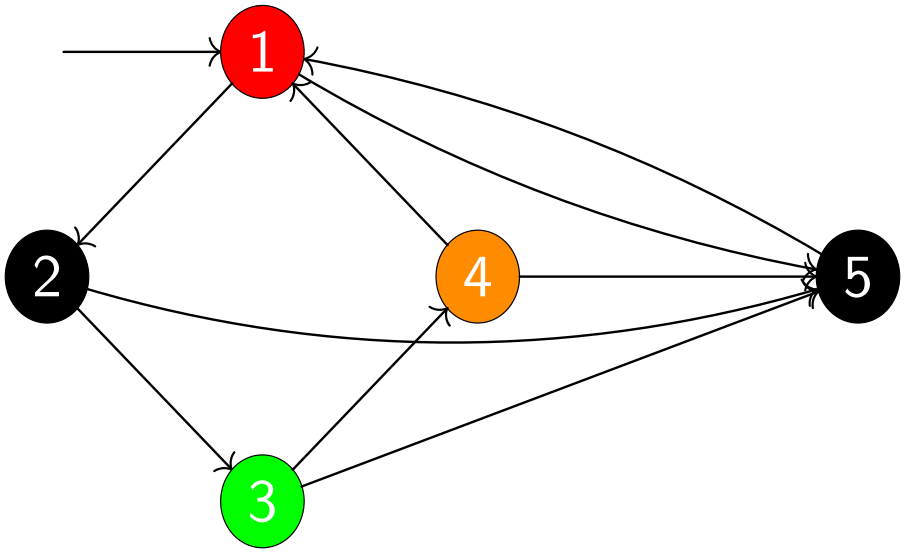
Example: model of a traffic light

System description: A traffic light can be red, green, amber or black (not working). The traffic light might stop working at any time. After it has been repaired, it turns red. Initially, the light is red.

$$
\begin{aligned}
X &= \{1,2,3,4,5\} \\
\rightarrow &= \{(1,2),(2,3),(3,4),(4,1),(1,5),(2,5),(3,5),(4,5),(5,1)\} \\
X_0 &= \{1\} \\
AP &= \{r,a,g\} \\
Y &= 2^{AP} \\
H &= \{1 \mapsto \{r\}, 2 \mapsto \{r,a\}, 3 \mapsto \{g\}, 4 \mapsto \{a\}, 5 \mapsto \emptyset\}
\end{aligned}
$$

Example: model of a traffic light

System description: A traffic light can be red, green, amber or black (not working). The traffic light might stop working at any time. After it has been repaired, it turns red. Initially, the light is red.



Control systems

Definition:

A control system Σ is a tuple $\Sigma = (\mathbb{R}^n, \mathsf{U}, \mathcal{U}, f)$ where:

- $\mathbb{R}^n$ is the state space;
- $\mathsf{U} \subseteq \mathbb{R}^m$ is the input set;
- $\mathcal{U}$ is a set of "nice" functions from $\mathbb{R}_0^+$ to U;
- $f : \mathbb{R}^n \times \mathsf{U} \to \mathbb{R}^n$ is a "nice" function;

A curve $\xi : \mathbb{R}_0^+ \to \mathbb{R}^n$ is a trajectory of Σ if there exists $\upsilon \in \mathcal{U}$ satisfying:

$$\dot{\xi} = f(\xi, \upsilon).$$

Control systems

Definition:

A control system Σ is a tuple $\Sigma = (\mathbb{R}^n, \mathsf{U}, \mathcal{U}, f)$ where:

- $\mathbb{R}^n$ is the state space;
- $\mathsf{U} \subseteq \mathbb{R}^m$ is the input set;
- $\mathcal{U}$ is a set of "nice" functions from $\mathbb{R}_0^+$ to U;
- $f : \mathbb{R}^n \times \mathsf{U} \to \mathbb{R}^n$ is a "nice" function;

A curve $\xi : \mathbb{R}_0^+ \to \mathbb{R}^n$ is a trajectory of Σ if there exists $\upsilon \in \mathcal{U}$ satisfying:

$$\dot{\xi} = f(\xi, \upsilon).$$

$\xi_{x\upsilon}(t)$ denotes the value of the trajectory of Σ at time t under the input υ from initial condition $x = \xi_{x\upsilon}(0)$.

Control systems as GTS

Given $\Sigma = (\mathbb{R}^n, \mathsf{U}, \mathcal{U}, f)$ and a sampling time $\tau \in \mathbb{R}^+$, the metric system $S_\tau(\Sigma) = (X, X_0, U, \longrightarrow, Y, H)$ associated with Σ is given by:

- $X = \mathbb{R}^n$;
- $X_0 = \mathbb{R}^n$;
- U: all the curves in $\mathcal{U}$ of duration τ;
- $x \xrightarrow{\upsilon} x'$ iff $\xi_{x\upsilon}(\tau) = x'$;
- $Y = \mathbb{R}^n$, equipped with the metric $\mathbf{d}(y, y') = \|y - y'\|$ for any $y, y' \in \mathbb{R}^n$;
- $H = 1_X$.

Control systems as GTS

Given $\Sigma = (\mathbb{R}^n, \mathsf{U}, \mathcal{U}, f)$ and a sampling time $\tau \in \mathbb{R}^+$, the metric system $S_\tau(\Sigma) = (X, X_0, U, \longrightarrow, Y, H)$ associated with Σ is given by:

- $X = \mathbb{R}^n$;
- $X_0 = \mathbb{R}^n$;
- U: all the curves in $\mathcal{U}$ of duration τ;
- $x \xrightarrow{\upsilon} x'$ iff $\xi_{x\upsilon}(\tau) = x'$;
- $Y = \mathbb{R}^n$, equipped with the metric $\mathbf{d}(y, y') = \|y - y'\|$ for any $y, y' \in \mathbb{R}^n$;
- $H = 1_X$.

$S_\tau(\Sigma)$ is an infinite system!

Control systems as GTS

Given $\Sigma = (\mathbb{R}^n, \mathsf{U}, \mathcal{U}, f)$ and a sampling time $\tau \in \mathbb{R}^+$, the metric system $S_\tau(\Sigma) = (X, X_0, U, \longrightarrow, Y, H)$ associated with Σ is given by:

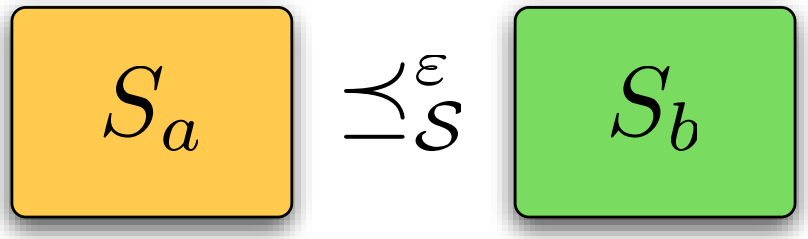
- $X = \mathbb{R}^n$;
- $X_0 = \mathbb{R}^n$;
- U: all the curves in $\mathcal{U}$ of duration τ;
- $x \xrightarrow{\upsilon} x'$ iff $\xi_{x\upsilon}(\tau) = x'$;
- $Y = \mathbb{R}^n$, equipped with the metric $\mathbf{d}(y, y') = \|y - y'\|$ for any $y, y' \in \mathbb{R}^n$;
- $H = 1_X$.

$S_\tau(\Sigma)$ is an infinite system!

Can we replace $S_\tau(\Sigma)$ with an equivalent and yet finite system?

Systems relations

Approximate Simulation Relation



All **behaviors** (outputs sequences) in S_a are *within* ε of behaviors in S_b (assuming a common metric for their output sets, and non-blocking systems).

Approximate (bi)simulation

Definition (Girard and Pappas 2007)

Consider metric systems $S_a = (X_a, X_{a0}, U_a, \xrightarrow[a]{}, Y_a, H_a)$ and $S_b = (X_b, X_{b0}, U_b, \xrightarrow[b]{}, Y_b, H_b)$ with the same output sets $Y_a = Y_b$ and metric $\mathbf{d}$. For $\varepsilon \in \mathbb{R}_0^+$, a relation $R \subseteq X_a \times X_b$ is an ε-approximate simulation relation from S_a to S_b ($S_a \preceq_S^{\epsilon} S_b$) if the following conditions are satisfied:

(i) $\forall x_{a0} \in X_{a0}$, $\exists x_{b0} \in X_{b0}$ with $(x_{a0}, x_{b0}) \in R$;

(ii) $\forall (x_a, x_b) \in R$, we have $\mathbf{d}(H_a(x_a), H_b(x_b)) \leq \varepsilon$;

(iii) $\forall (x_a, x_b) \in R$, $x_a \xrightarrow[a]{u_a} x_a'$ in S_a implies the existence of $x_b \xrightarrow[b]{u_b} x_b'$ in S_b satisfying $(x_a', x_b') \in R$

Reference:

PT09 (Chapters 4 and 9) Tabuada, Paulo. Verification and control of hybrid systems: a symbolic approach. Springer Science & Business Media, 2009. http://www.springer.com/gp/book/9781441902238

Approximate (bi)simulation

Definition (Girard and Pappas 2007)

Consider metric systems $S_a = (X_a, X_{a0}, U_a, \xrightarrow[a]{}, Y_a, H_a)$ and $S_b = (X_b, X_{b0}, U_b, \xrightarrow[b]{}, Y_b, H_b)$ with the same output sets $Y_a = Y_b$ and metric $\mathbf{d}$. For $\varepsilon \in \mathbb{R}_0^+$, a relation $R \subseteq X_a \times X_b$ is an ε-approximate simulation relation from S_a to S_b ($S_a \preceq_S^\epsilon S_b$) if the following conditions are satisfied:

(i) $\forall x_{a0} \in X_{a0}$, $\exists x_{b0} \in X_{b0}$ with $(x_{a0}, x_{b0}) \in R$;

(ii) $\forall (x_a, x_b) \in R$, we have $\mathbf{d}(H_a(x_a), H_b(x_b)) \leq \varepsilon$;

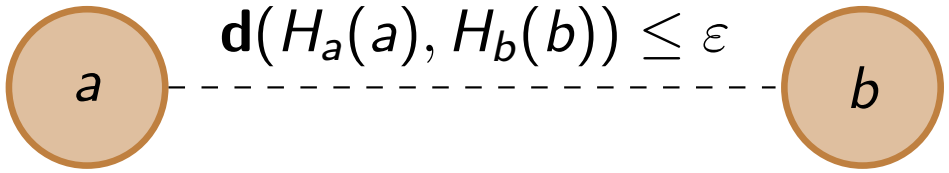
(iii) $\forall (x_a, x_b) \in R$, $x_a \xrightarrow[a]{u_a} x_a'$ in S_a implies the existence of $x_b \xrightarrow[b]{u_b} x_b'$ in S_b satisfying $(x_a', x_b') \in R$

A relation $R \subseteq X_a \times X_b$ is said to be an ε-approximate bisimulation relation between S_a and S_b ($S_a \cong_S^\epsilon S_b$) if R is an ε-approximate simulation relation from S_a to S_b and R^{-1} is an ε-approximate simulation relation from S_b to S_a.

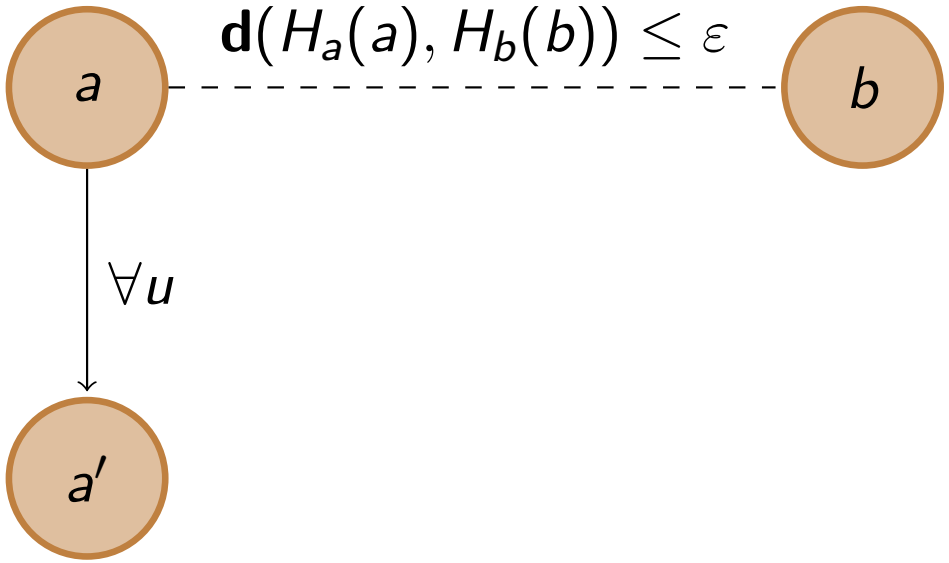
Reference:

PT09 (Chapters 4 and 9) Tabuada, Paulo. Verification and control of hybrid systems: a symbolic approach. Springer Science & Business Media, 2009. http://www.springer.com/gp/book/9781441902238

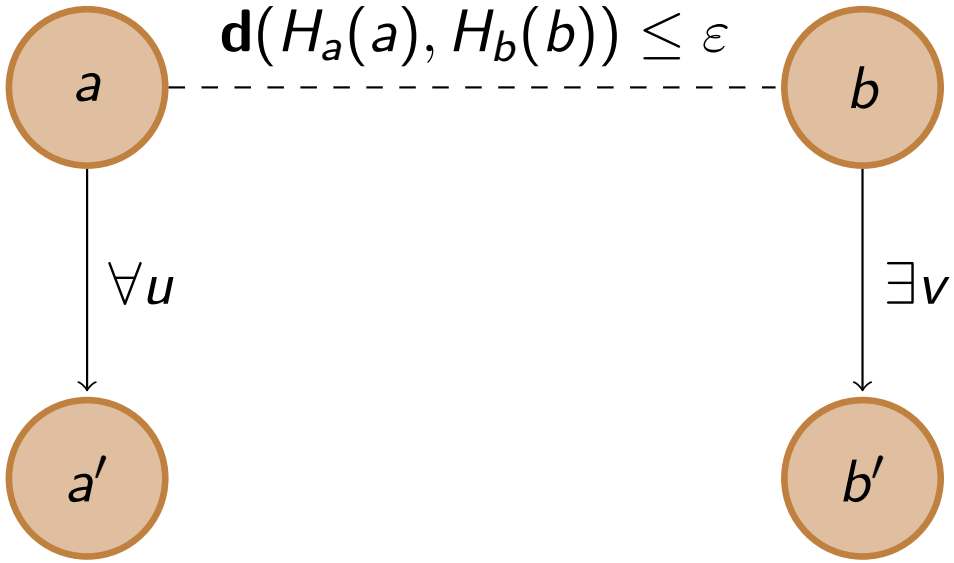
Approximate simulation



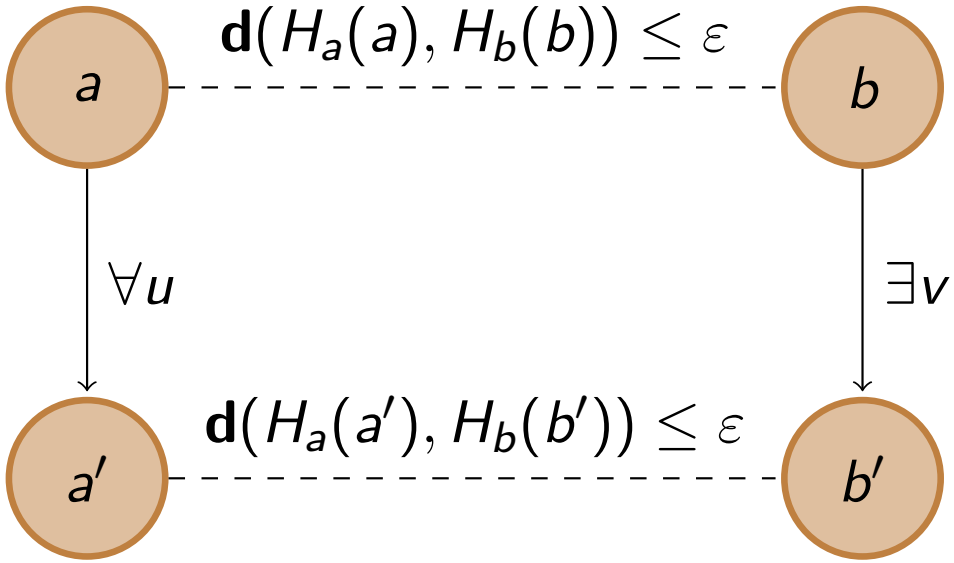
Approximate simulation



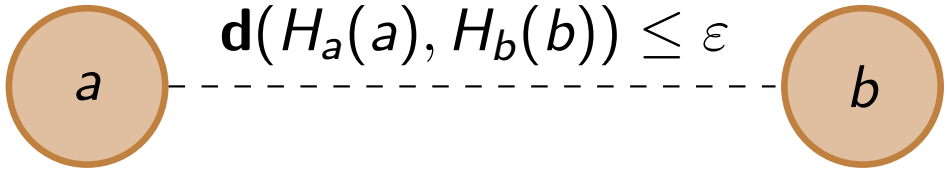
Approximate simulation



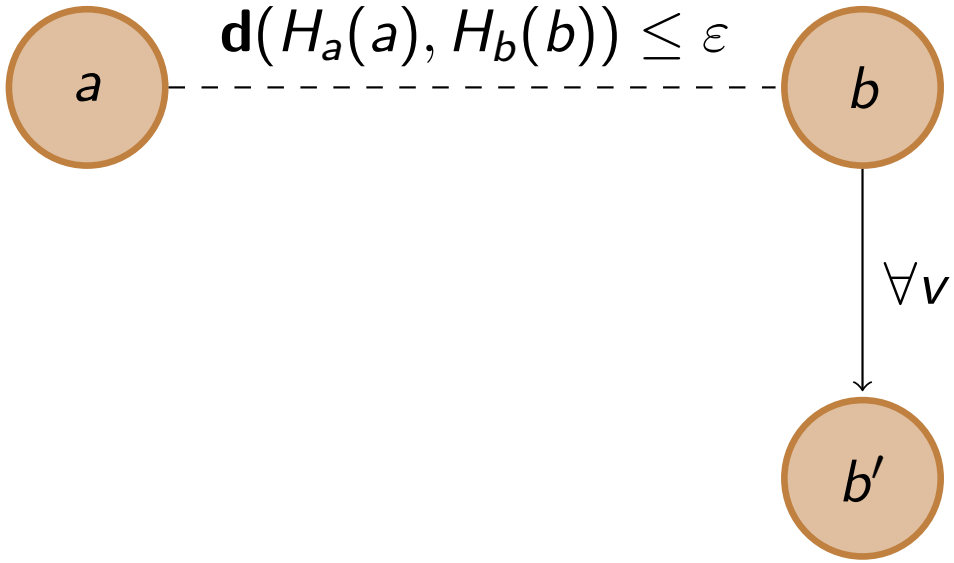
Approximate simulation



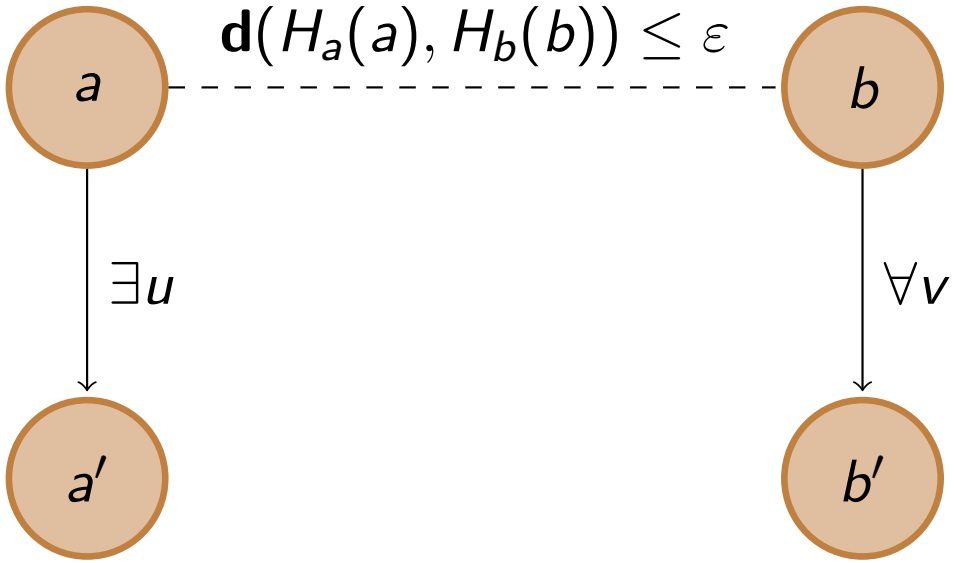
Approximate bisimulation



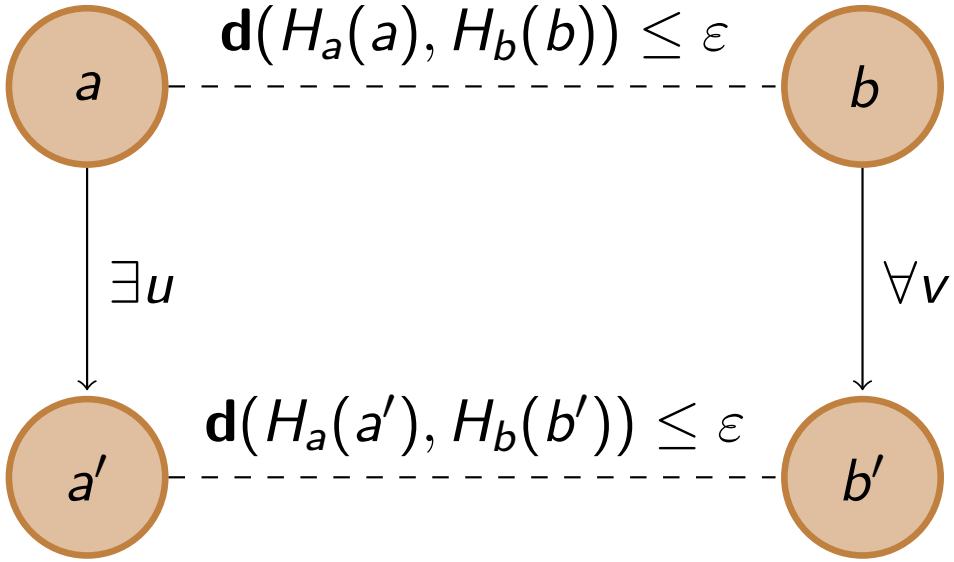
Approximate bisimulation



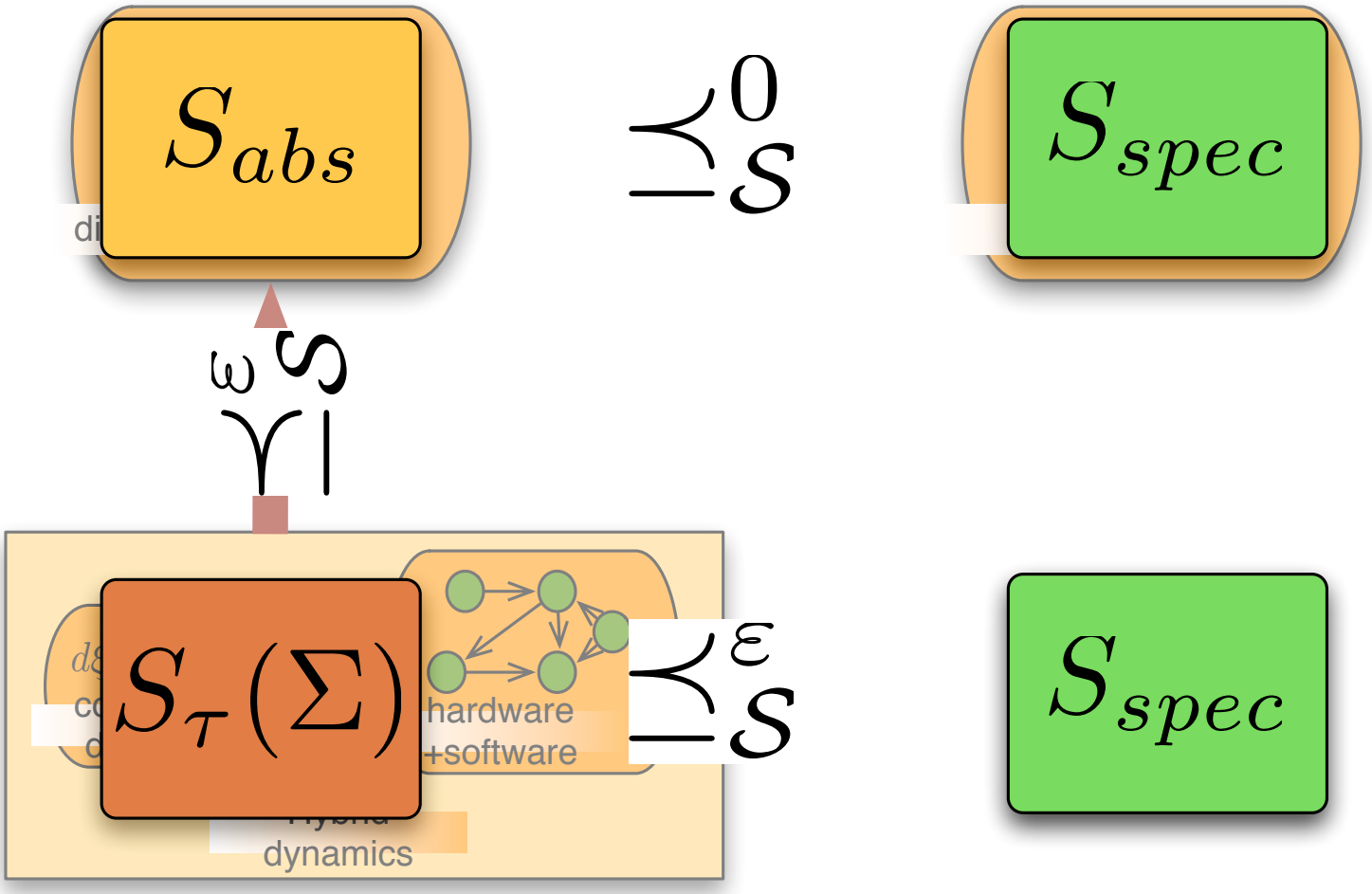
Approximate bisimulation



Approximate bisimulation



Verification overview

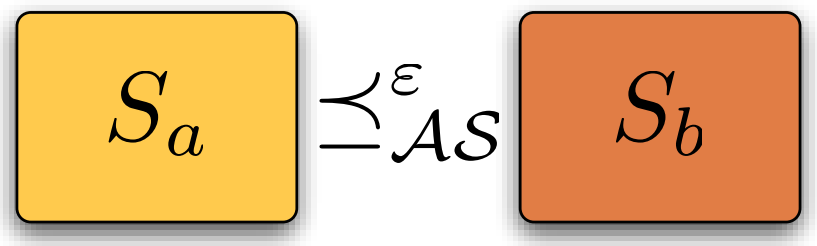


Σ is a (hybrid) autonomous system,

$S_\tau(\Sigma)$ is the system representing the exact time discretization of Σ with step τ.

Controller refinement

Approximate Alternating Simulation Relation



Controllers for S_a can be refined into controllers for S_b.

Alternating approximate (bi)simulation

Definition (Pola and Tabuada 2009)

Consider metric systems $S_a = (X_a, X_{a0}, U_a, \xrightarrow[a]{}, Y_a, H_a)$ and $S_b = (X_b, X_{b0}, U_b, \xrightarrow[b]{}, Y_b, H_b)$ with the same output sets $Y_a = Y_b$ and metric $\mathbf{d}$. For $\varepsilon \in \mathbb{R}_0^+$, a relation $R \subseteq X_a \times X_b$ is an alternating ε-approximate simulation relation from S_a to S_b ($S_a \preceq^{\epsilon}_{\mathcal{AS}} S_b$) if the following conditions are satisfied:

(i) $\forall x_{a0} \in X_{a0}$, $\exists x_{b0} \in X_{b0}$ with $(x_{a0}, x_{b0}) \in R$;

(ii) $\forall (x_a, x_b) \in R$, we have $\mathbf{d}(H_a(x_a), H_b(x_b)) \leq \varepsilon$;

(iii) $\forall u_a \in U_a(x_a)$ $\exists u_b \in U_b(x_b)$ such that $\forall x_b' \in \mathbf{Post}_{u_b}(x_b)$ $\exists x_a' \in \mathbf{Post}_{u_a}(x_a)$ satisfying $(x_a', x_b') \in R$.

Reference:

PT09 (Chapters 4 and 9) Tabuada, Paulo. Verification and control of hybrid systems: a symbolic approach. Springer Science & Business Media, 2009. http://www.springer.com/gp/book/9781441902238

Alternating approximate (bi)simulation

Definition (Pola and Tabuada 2009)

Consider metric systems $S_a = (X_a, X_{a0}, U_a, \xrightarrow[a]{}, Y_a, H_a)$ and $S_b = (X_b, X_{b0}, U_b, \xrightarrow[b]{}, Y_b, H_b)$ with the same output sets $Y_a = Y_b$ and metric $\mathbf{d}$. For $\varepsilon \in \mathbb{R}_0^+$, a relation $R \subseteq X_a \times X_b$ is an alternating ε-approximate simulation relation from S_a to S_b ($S_a \preceq^{\epsilon}_{\mathcal{AS}} S_b$) if the following conditions are satisfied:

(i) $\forall x_{a0} \in X_{a0}$, $\exists x_{b0} \in X_{b0}$ with $(x_{a0}, x_{b0}) \in R$;

(ii) $\forall (x_a, x_b) \in R$, we have $\mathbf{d}(H_a(x_a), H_b(x_b)) \leq \varepsilon$;

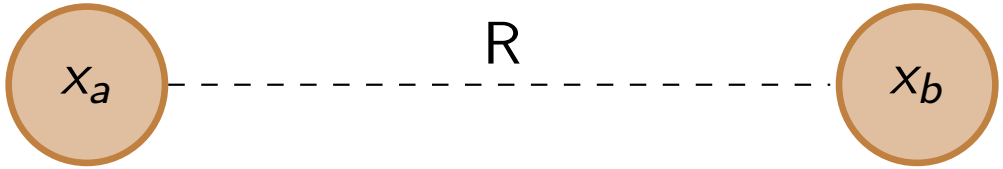
(iii) $\forall u_a \in U_a(x_a)$ $\exists u_b \in U_b(x_b)$ such that $\forall x'_b \in \mathbf{Post}_{u_b}(x_b)$ $\exists x'_a \in \mathbf{Post}_{u_a}(x_a)$ satisfying $(x'_a, x'_b) \in R$.

A relation $R \subseteq X_a \times X_b$ is said to be an alternating ε-approximate bisimulation relation between S_a and S_b ($S_a \cong^{\epsilon}_{\mathcal{AS}} S_b$) if R is an alternating ε-approximate simulation relation from S_a to S_b and R^{-1} is an alternating ε-approximate simulation relation from S_b to S_a.

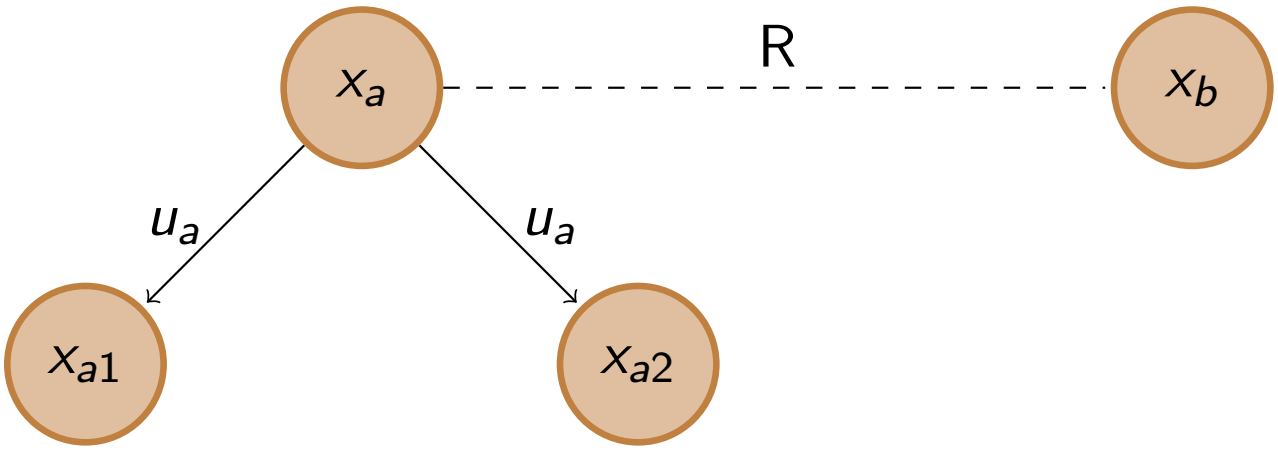
Reference:

PT09 (Chapters 4 and 9) Tabuada, Paulo. Verification and control of hybrid systems: a symbolic approach. Springer Science & Business Media, 2009. http://www.springer.com/gp/book/9781441902238

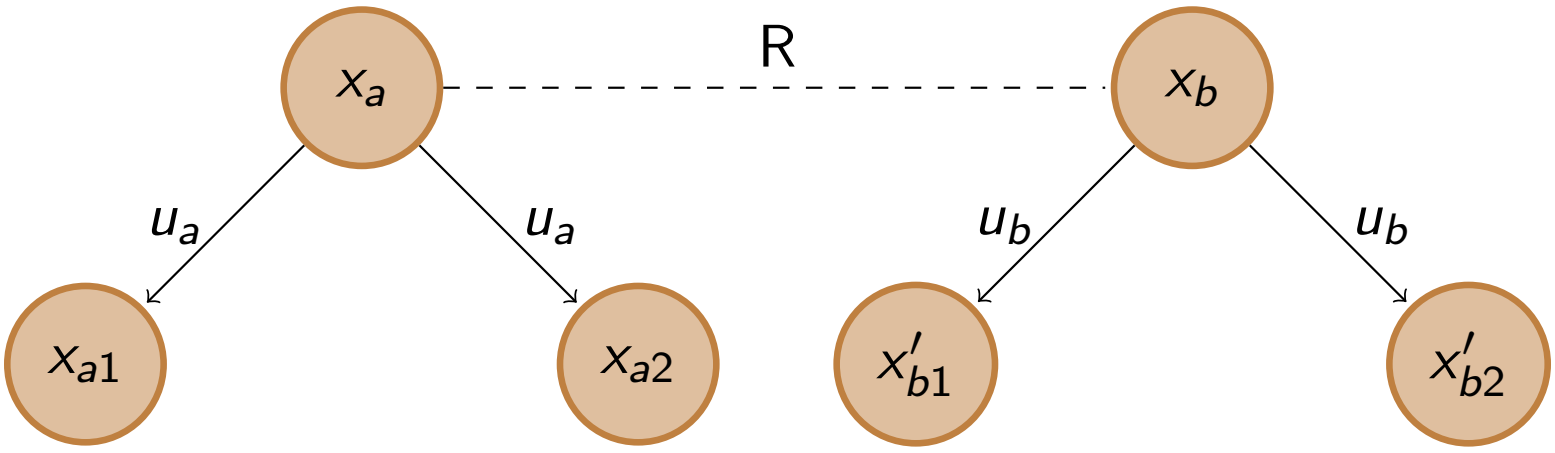
Alternating approximate simulation



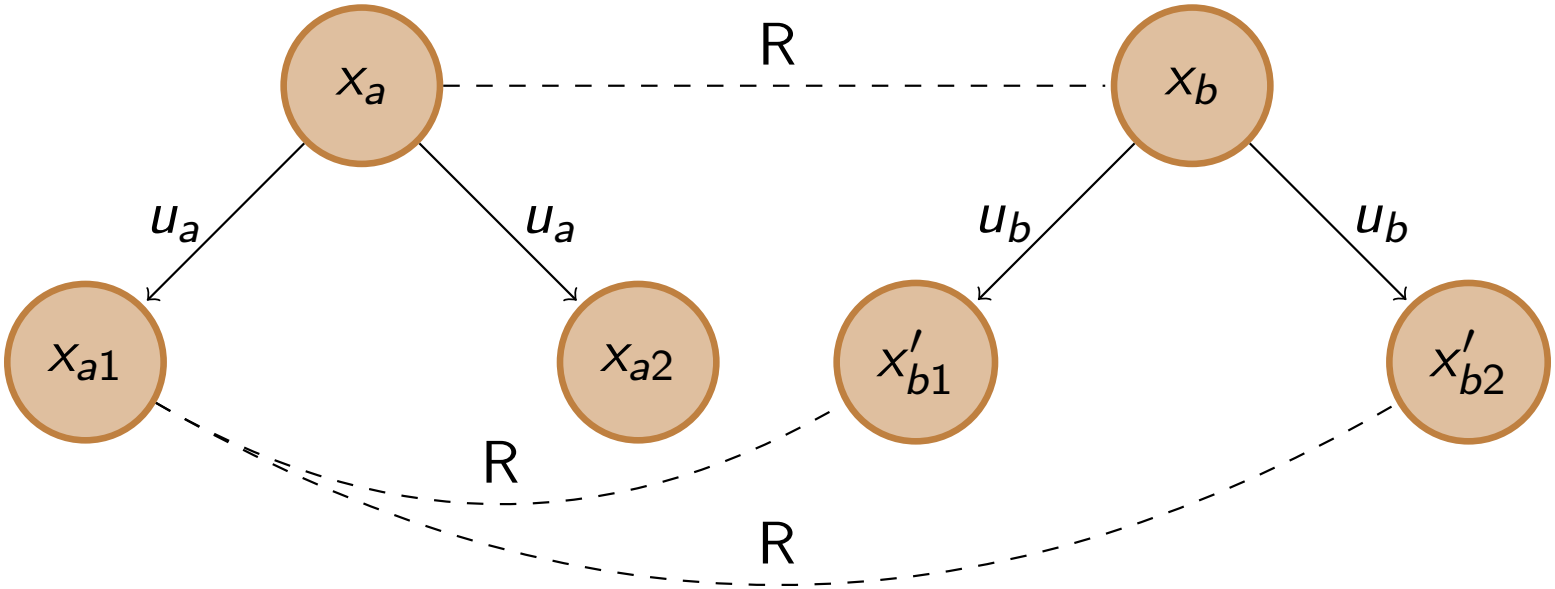
Alternating approximate simulation



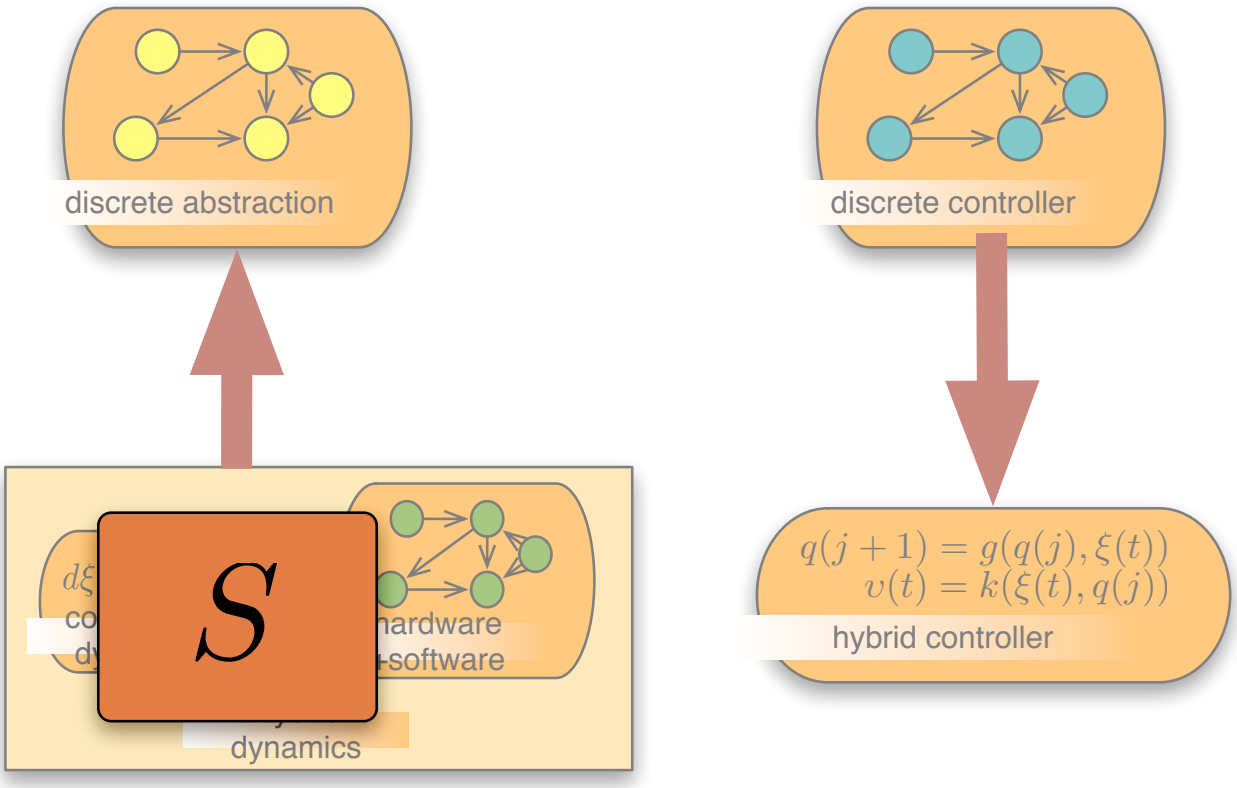
Alternating approximate simulation



Alternating approximate simulation



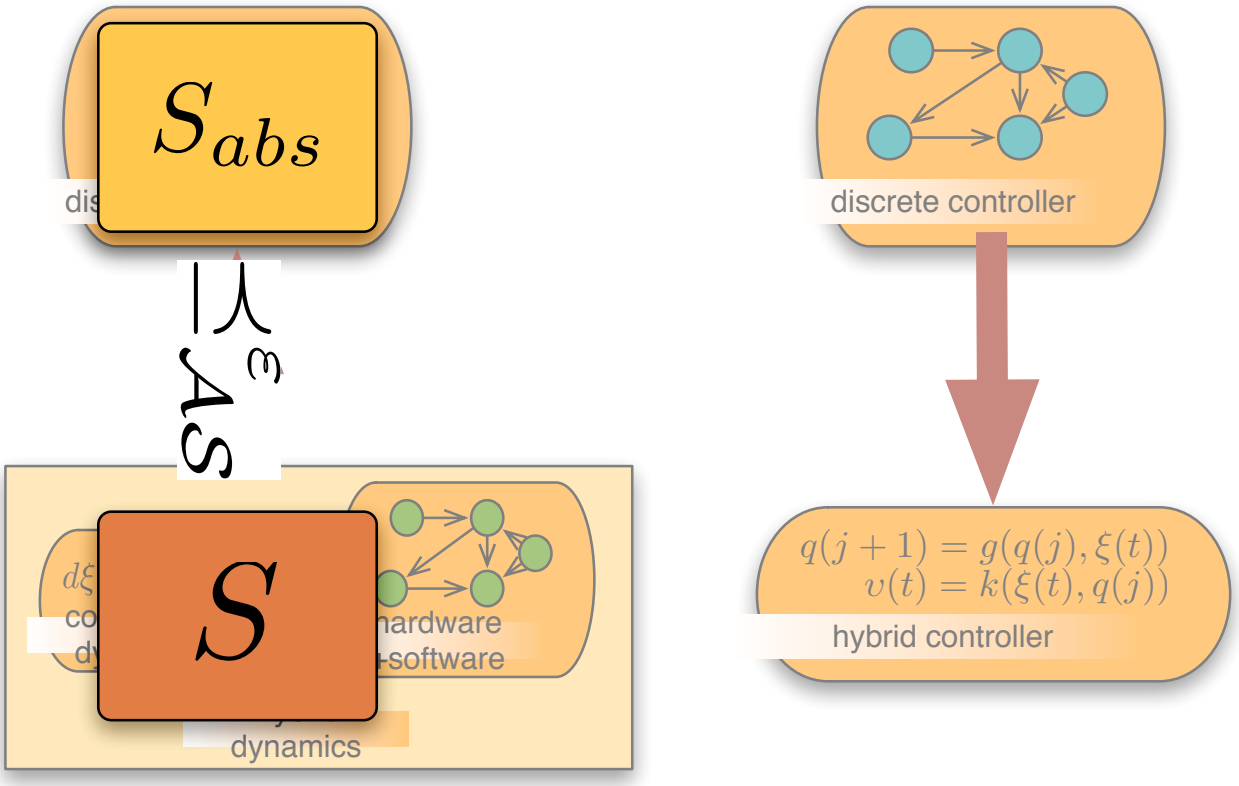
Controller synthesis overview



Σ is a (hybrid) control system

$S_\tau(\Sigma)$ is the system representing the exact time discretization of Σ with step τ.

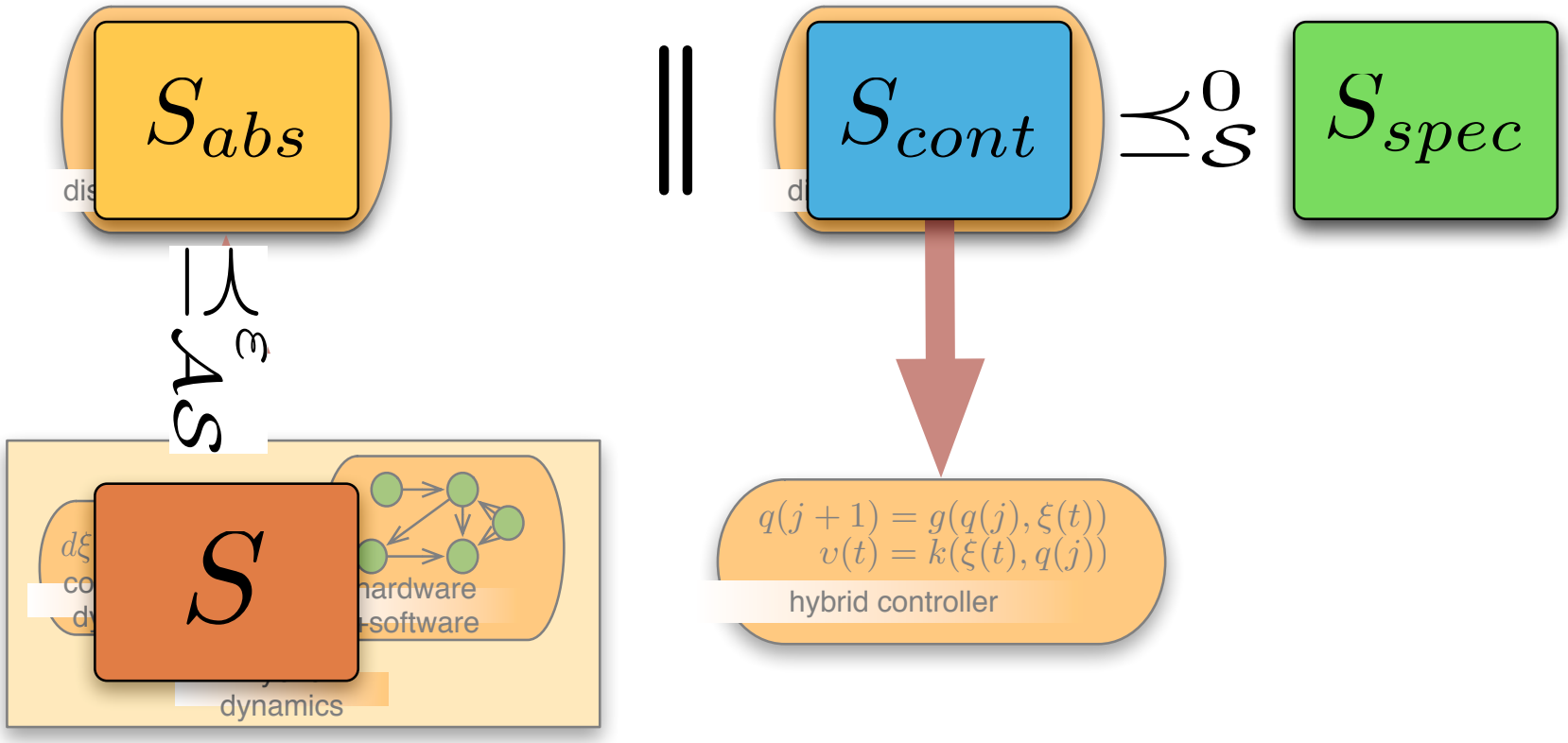
Controller synthesis overview



Σ is a (hybrid) control system

$S_\tau(\Sigma)$ is the system representing the exact time discretization of Σ with step τ.

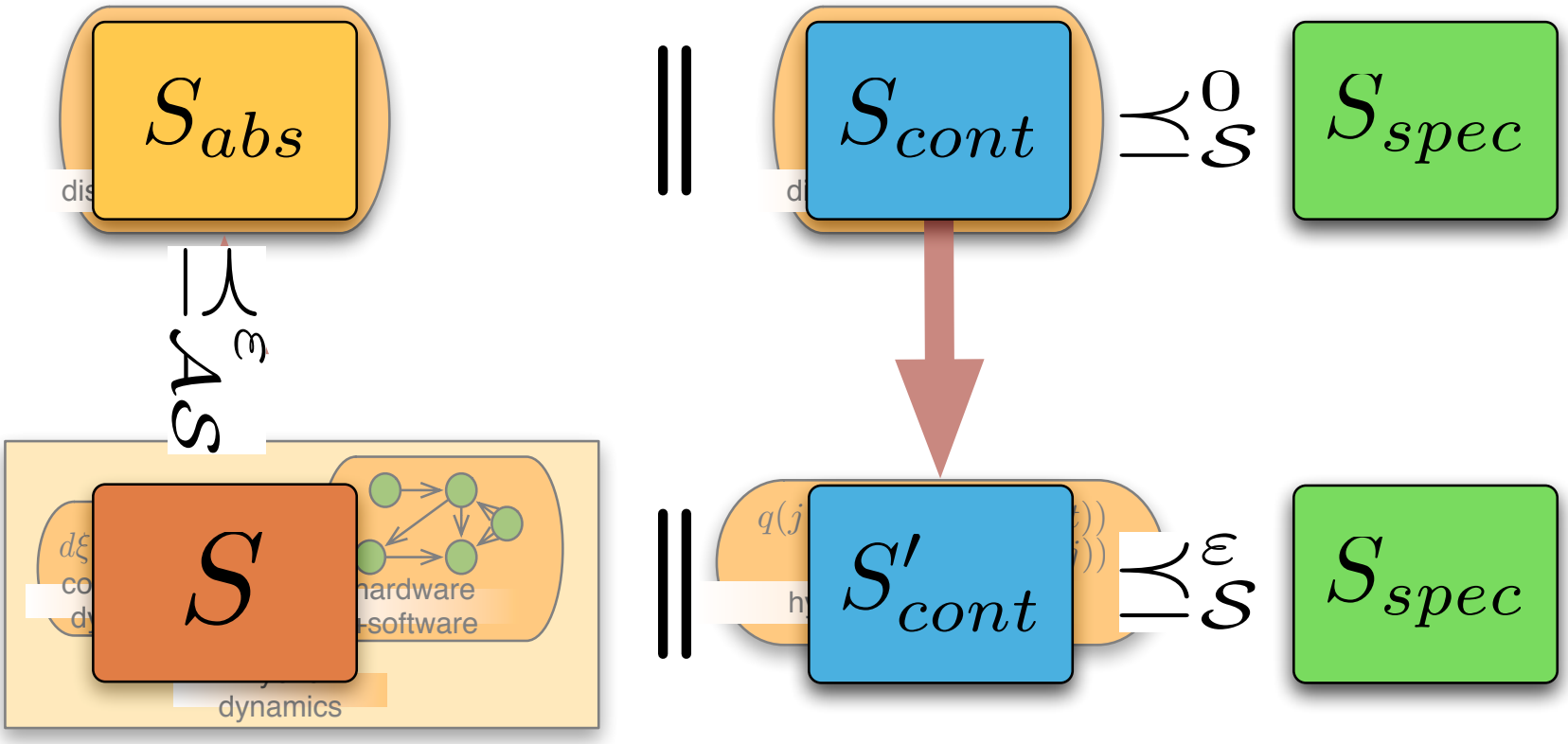
Controller synthesis overview



Σ is a (hybrid) control system

$S_\tau(\Sigma)$ is the system representing the exact time discretization of Σ with step τ.

Controller synthesis overview



Σ is a (hybrid) control system

$S_\tau(\Sigma)$ is the system representing the exact time discretization of Σ with step τ.

Alternating Simulation vs Alternating Bisimulation

- **Simulation:** If a controller is found for the abstraction, then a controller can be constructed for the concrete system.

$$\exists C_s \Rightarrow \exists C_c$$

- **Bisimulation:** Additionally, if no controller is found for the abstraction, then no controller can be constructed for the concrete system.

$$\nexists C_s \Rightarrow \nexists C_c$$

Or putting together the two logical statements:

$$\exists C_s \Leftrightarrow \exists C_c$$

Verification/Controller design: Fixed points

Consider some system S_a, the following two problems can be solved through a fixed point iteration:

$$S_c \preceq_S^0 S_{spec}$$

- **Safety (in W):** Controller: $S_c = (X_c, X_{c0}, U_a, \underset{c}{\longrightarrow})$
  - $X_c = Z_c = \lim_{i\to\infty} F_W^i(X_a)$
  - $X_{c0} = Z_c \cap X_{a0}$
  - $x_c \xrightarrow[c]{u_a} x_c'$ if $\emptyset \neq \mathrm{Post}_{u_a}(x_c) \subseteq Z_c$

  $F_W(Z) = \{x_a \in Z \,|\, x_a \in W \text{ and } \exists u_a \in U_a(x_a) : \emptyset \neq \mathrm{Post}_{u_a}(x_a) \subseteq Z\}$
- **Reachability (to W):** Controller: $S_c = (X_c, X_{c0}, U_a, \underset{c}{\longrightarrow})$
  - $X_c = Z_c = \lim_{i\to\infty} G_W^i(\emptyset)$
  - $X_{c0} = Z_c \cap X_{a0}$
  - $x_c \xrightarrow[c]{u_a} x_c'$ if $\exists k \in \mathbb{N}$ such that $x_c \notin G_W^k(\emptyset)$ and $\emptyset \neq \mathrm{Post}_{u_a}(x_c) \subseteq G_W^k(\emptyset)$

  $G_W(Z) = \{x_a \in X_a \,|\, x_a \in W \text{ or } \exists u_a \in U_a(x_a) : \emptyset \neq \mathrm{Post}_{u_a}(x_a) \subseteq Z\}$

Reference:

PT09 (Chapter 6) Tabuada, Paulo. Verification and control of hybrid systems: a symbolic approach. Springer Science &

Outline

Quotient system

Definition

Let $S = (X, X_0, U, \longrightarrow, Y, H)$ be a system and let Q be an equivalence relation on X such that $(x, x') \in Q$ implies that $H(x) = H(x')$. The quotient of S by Q, denoted by $S_{/Q}$, is the system $S_{/Q} = \left(X_{/Q}, X_{/Q0}, U_{/Q}, \xrightarrow[/Q]{}, Y_{/Q}, H_{/Q}\right)$ consisting of:

- $X_{/Q} = X/Q$;
- $X_{/Q0} = \{x_{/Q} \in X_{/Q} \mid x_{/Q} \cap X_0 \neq \varnothing\}$;
- $U_{/Q} = U$;
- $x_{/Q} \xrightarrow[/Q]{u} x'_{/Q}$ if there exists $x \xrightarrow{u} x'$ in S with $x \in x_{/Q}$ and $x' \in x'_{/Q}$;
- $Y_{/Q} = Y$;
- $H_{/Q}(x_{/Q}) = H(x)$ for some $x \in x_{/Q}$.

Quotient system

Definition

Let $S = (X, X_0, U, \longrightarrow, Y, H)$ be a system and let Q be an equivalence relation on X such that $(x, x') \in Q$ implies that $H(x) = H(x')$. The quotient of S by Q, denoted by $S_{/Q}$, is the system $S_{/Q} = \left(X_{/Q}, X_{/Q0}, U_{/Q}, \xrightarrow[/Q]{}, Y_{/Q}, H_{/Q}\right)$ consisting of:

- $X_{/Q} = X/Q$;
- $X_{/Q0} = \{x_{/Q} \in X_{/Q} \mid x_{/Q} \cap X_0 \neq \varnothing\}$;
- $U_{/Q} = U$;
- $x_{/Q} \xrightarrow[/Q]{u} x'_{/Q}$ if there exists $x \xrightarrow{u} x'$ in S with $x \in x_{/Q}$ and $x' \in x'_{/Q}$;
- $Y_{/Q} = Y$;
- $H_{/Q}(x_{/Q}) = H(x)$ for some $x \in x_{/Q}$.

When the equivalence relation Q has finitely many equivalence classes, $S_{/Q}$ is guaranteed to be finite.

Bisimulation Algorithm

Theorem

IF *algorithm 1 terminates, then* $S_{/\mathcal{P}'} \cong^0_{\mathcal{AS}} S$.

Input: Equivalence Relation $\mathcal{P}$ and system S
Output: $\mathcal{P}'$
$\mathcal{P}' := \mathcal{P}$;
while $\exists P, P' \in \mathcal{P}'$ *such that* $\varnothing \neq P' \cap \mathrm{Pre}(P) \neq P'$ **do**
  $\widehat{\mathcal{P}}' := \varnothing$;
  forall $P, P' \in \mathcal{P}'$ **do**
    $P_a := P' \cap \mathrm{Pre}(P)$;
    $P_b := P' \backslash \mathrm{Pre}(P)$;
    $\widehat{\mathcal{P}}' := \widehat{\mathcal{P}}' \cup \{P_a, P_b\}$;
  end
  $\mathcal{P}' := \widehat{\mathcal{P}}'$;
end

Algorithm 1: Computation of a bisimulation relation, respecting partition $\mathcal{P}$, between S and S.

$$\forall W \subseteq X,\ Pre(W) = \left\{ x \in X \mid x \xrightarrow{u} x' \text{ for some } u \in U \text{ and } x' \in W \right\}.$$

Reference:

PT09 (Chapters 4.2 and 5.2) Tabuada, Paulo. Verification and control of hybrid systems: a symbolic approach. Springer Science & Business Media, 2009. http://www.springer.com/gp/book/9781441902238

Constructing Similar Models

Theorem

Algorithm 2 results in $S_{/\mathcal{P}'} \preceq^0_{\mathcal{AS}} S \preceq^0_{\mathcal{S}} S_{/\mathcal{P}'}$.

Input: Equivalence Relation $\mathcal{P}$, system S and $k \in \mathbb{N}_0$
Output: $\mathcal{P}'$
$\mathcal{P}' := \mathcal{P}$;
$i := 1$;
while $i \leq k$ **do**
  $i := i + 1$;
  $\widehat{\mathcal{P}}' := \varnothing$;
  forall $P, P' \in \mathcal{P}'$ **do**
    $P_a := P' \cap \mathrm{Pre}(P)$;
    $P_b := P' \backslash \mathrm{Pre}(P)$;
    $\widehat{\mathcal{P}}' := \widehat{\mathcal{P}}' \cup \{P_a, P_b\}$;
  end
  $\mathcal{P}' := \widehat{\mathcal{P}}'$;
end

Algorithm 2: Computation of a simulation relation, respecting partition $\mathcal{P}$, between S and S.

Reference:

PT09 (Chapters 4.2 and 5.2) Tabuada, Paulo. Verification and control of hybrid systems: a symbolic approach. Springer Science & Business Media, 2009. http://www.springer.com/gp/book/9781441902238

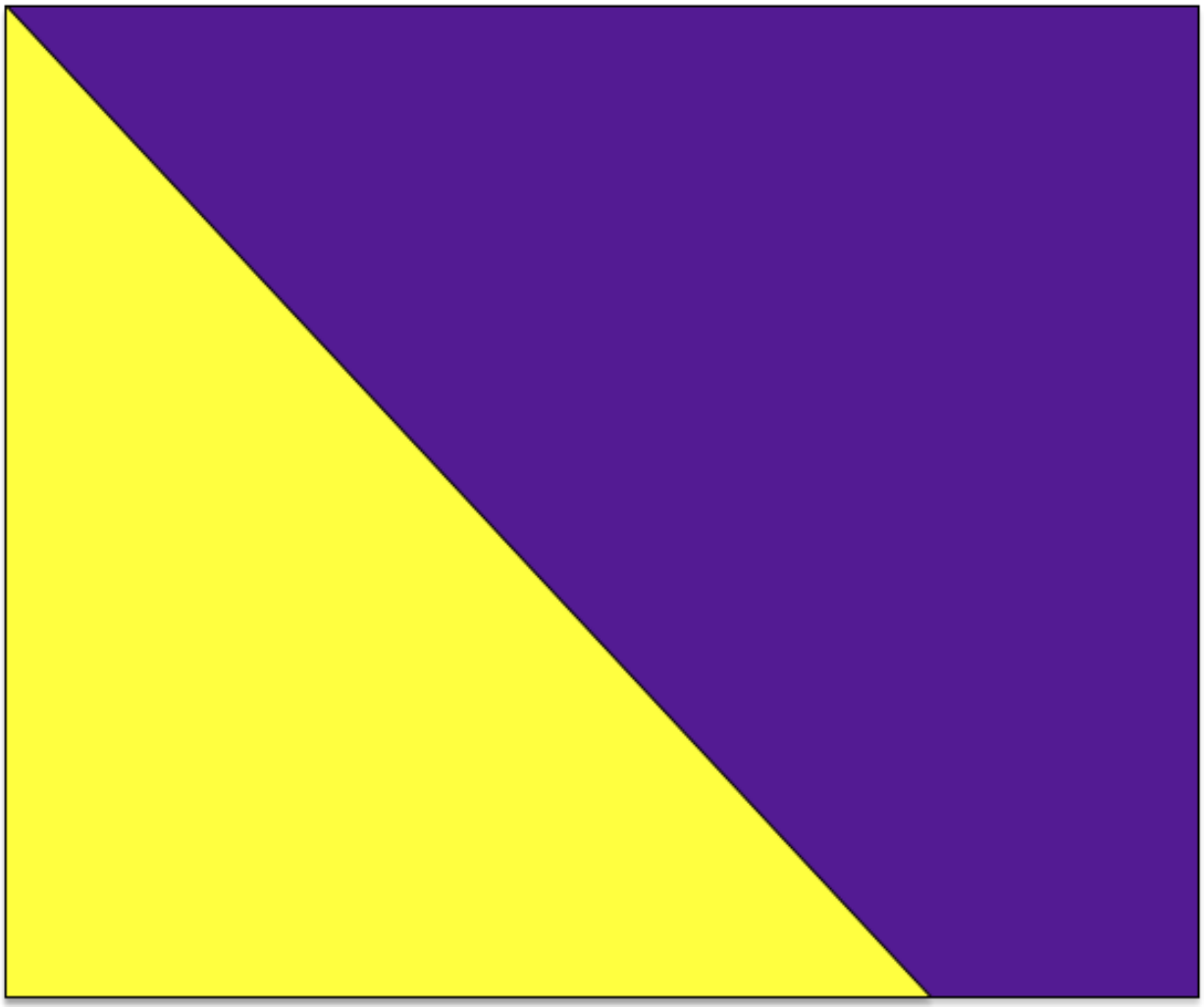
Exact symbolic models

A natural question one may ask is:

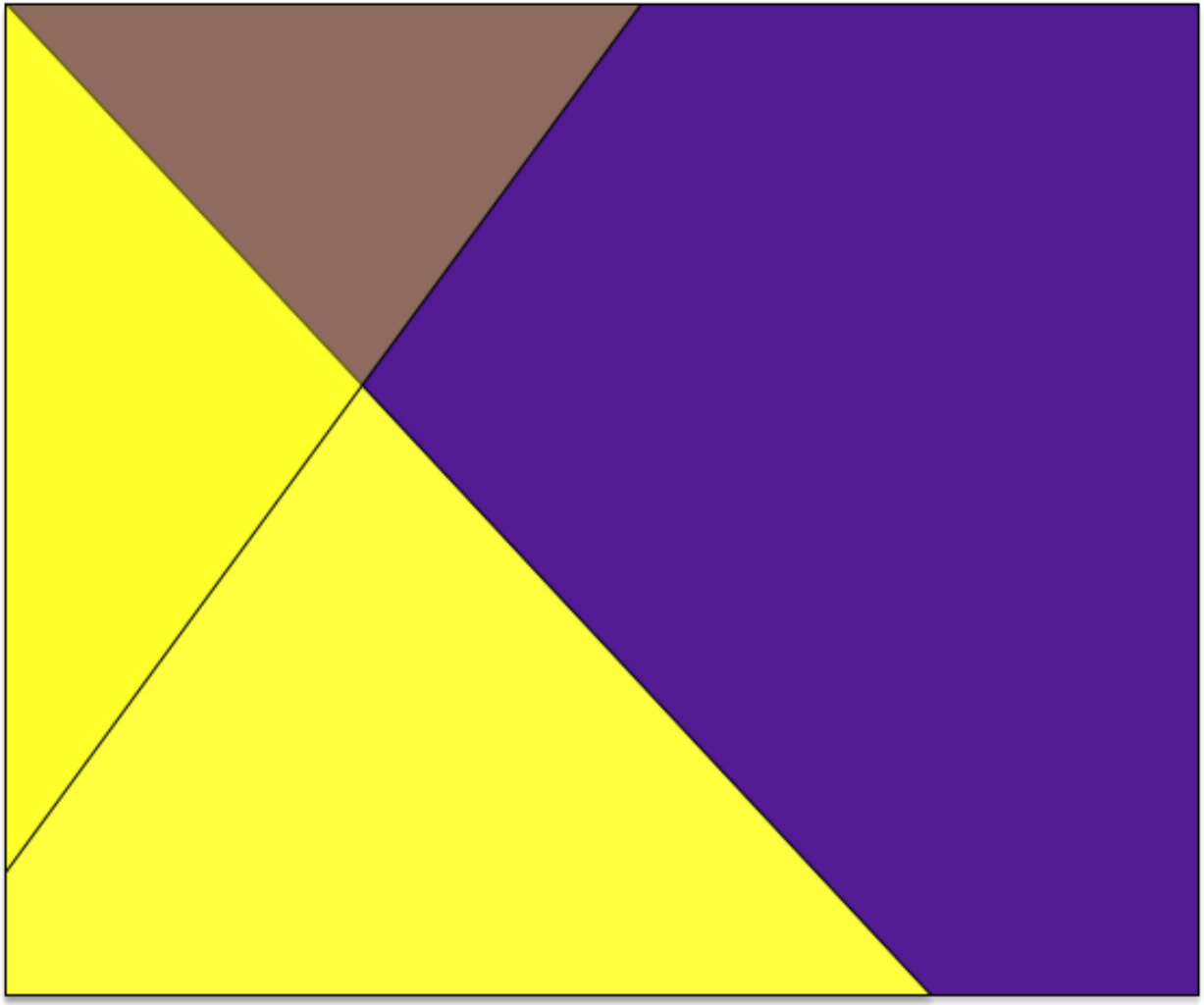
When do there exist exact symbolic abstractions of infinite systems?

- In general they do not exist, not even for simple linear systems
- Some particular cases when they do exist:
  - Order Minimal Hybrid Systems
  - Linear Discrete Time Systems with adapted sets
  - Multi-affine control systems with rectangular sets
  - Timed Automata

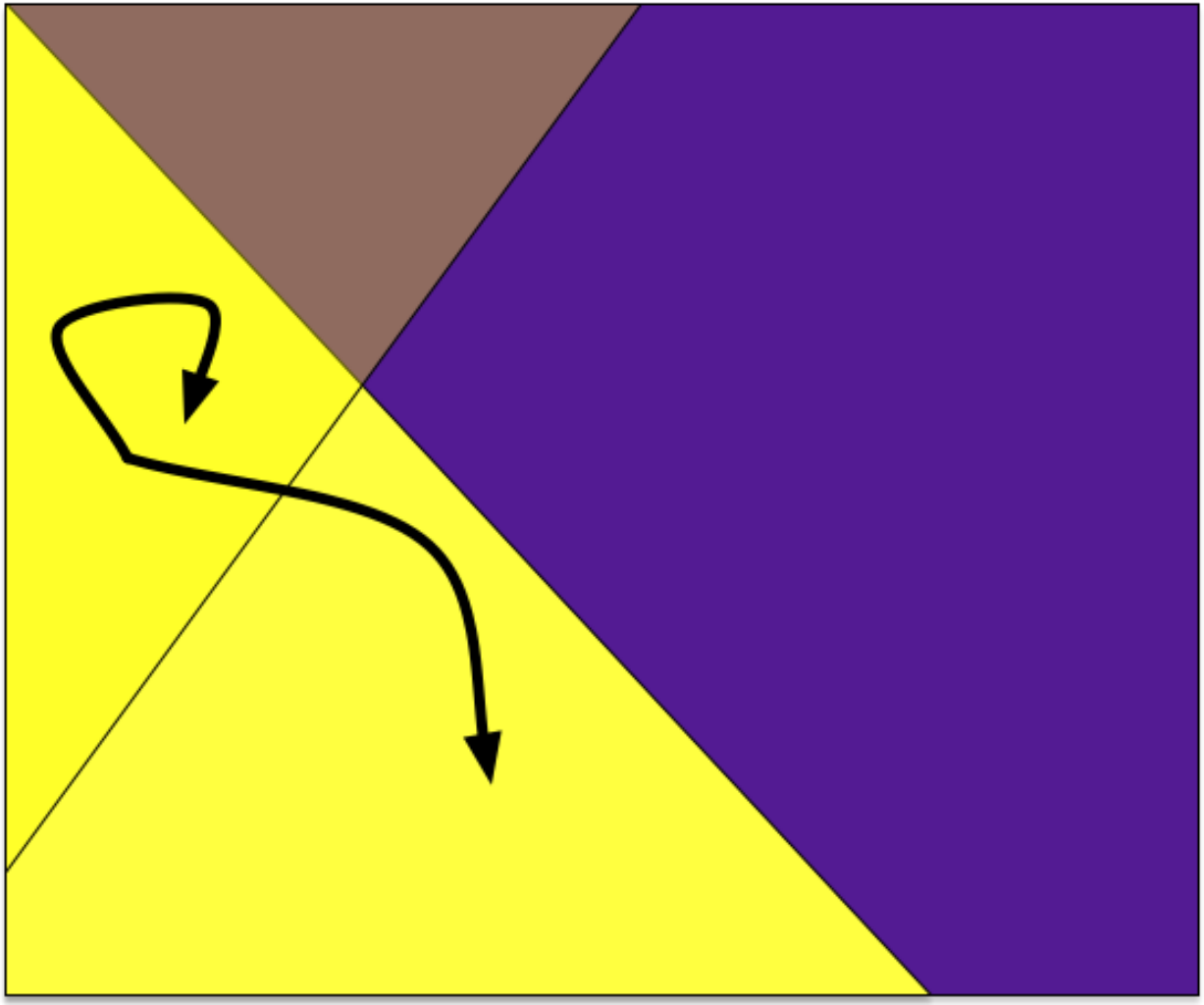
Construction of exact symbolic models



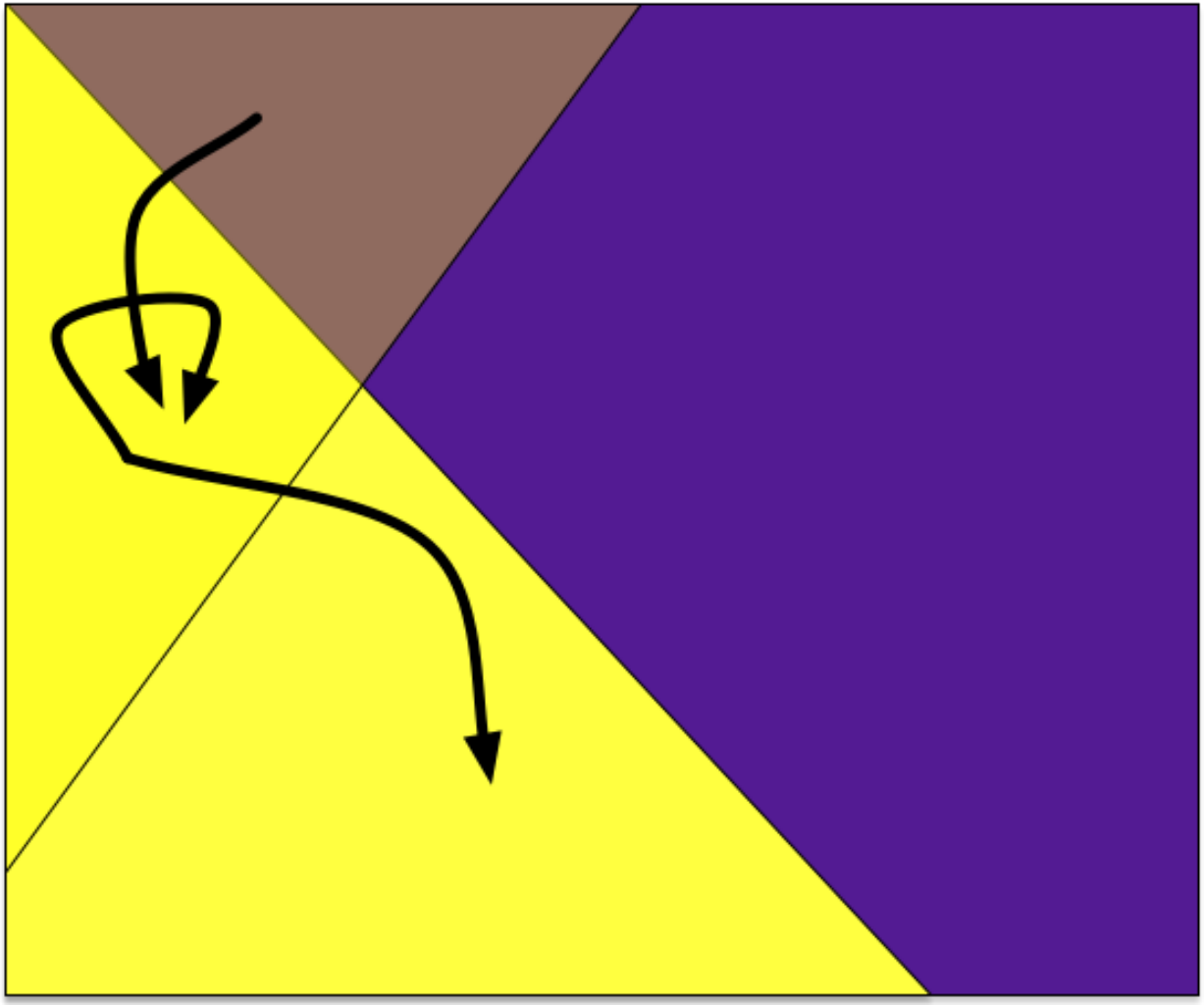
Construction of exact symbolic models



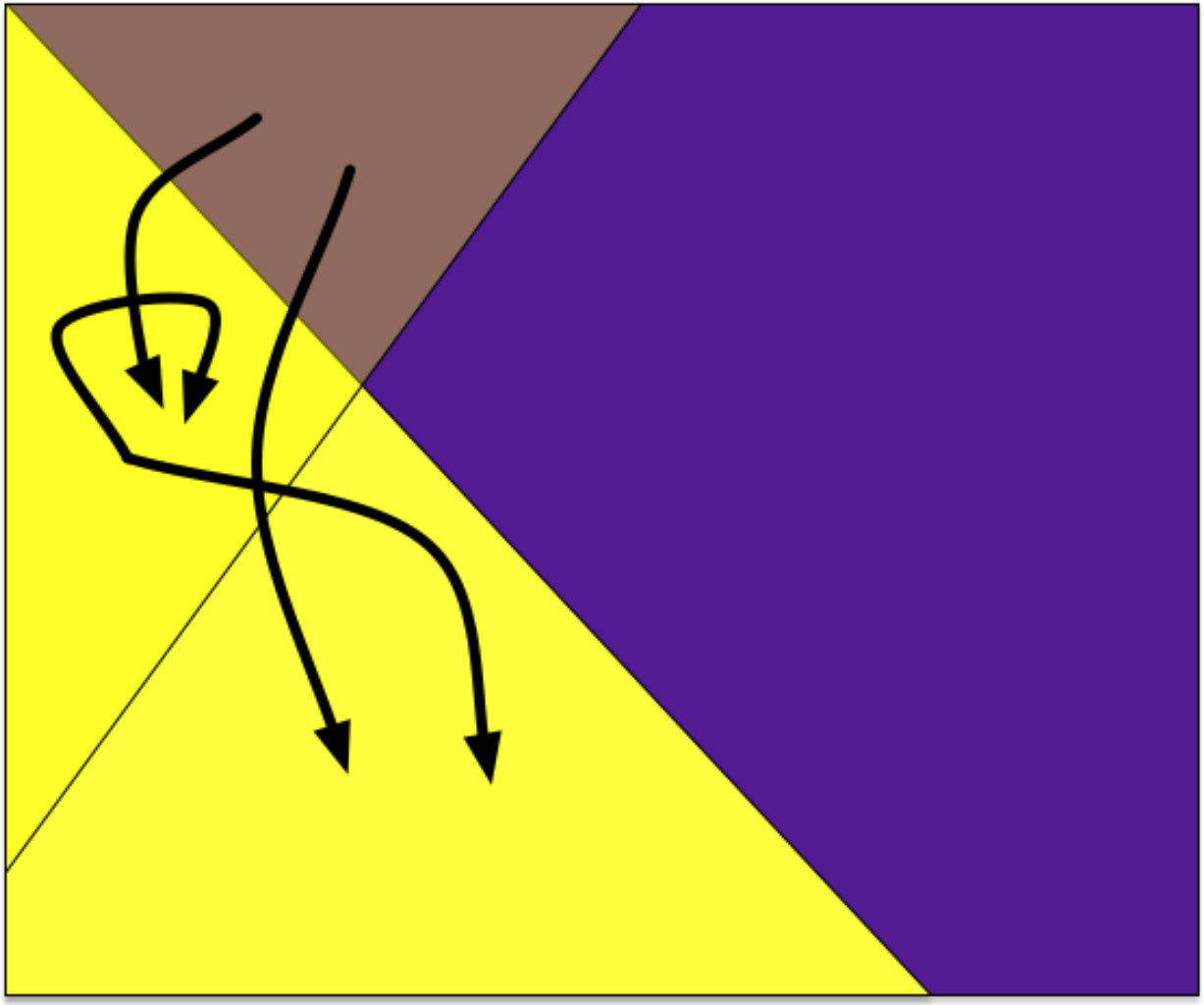
Construction of exact symbolic models



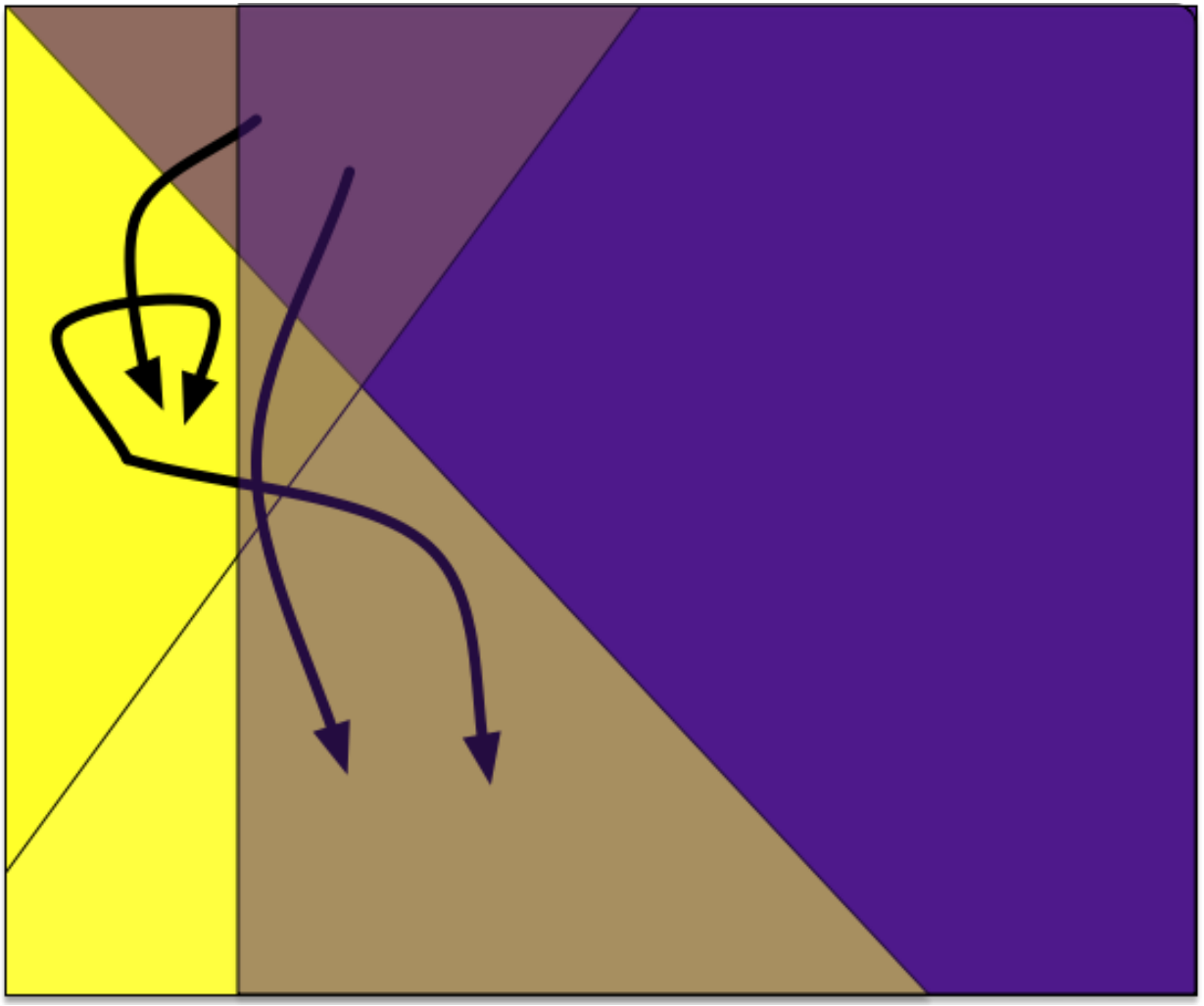
Construction of exact symbolic models



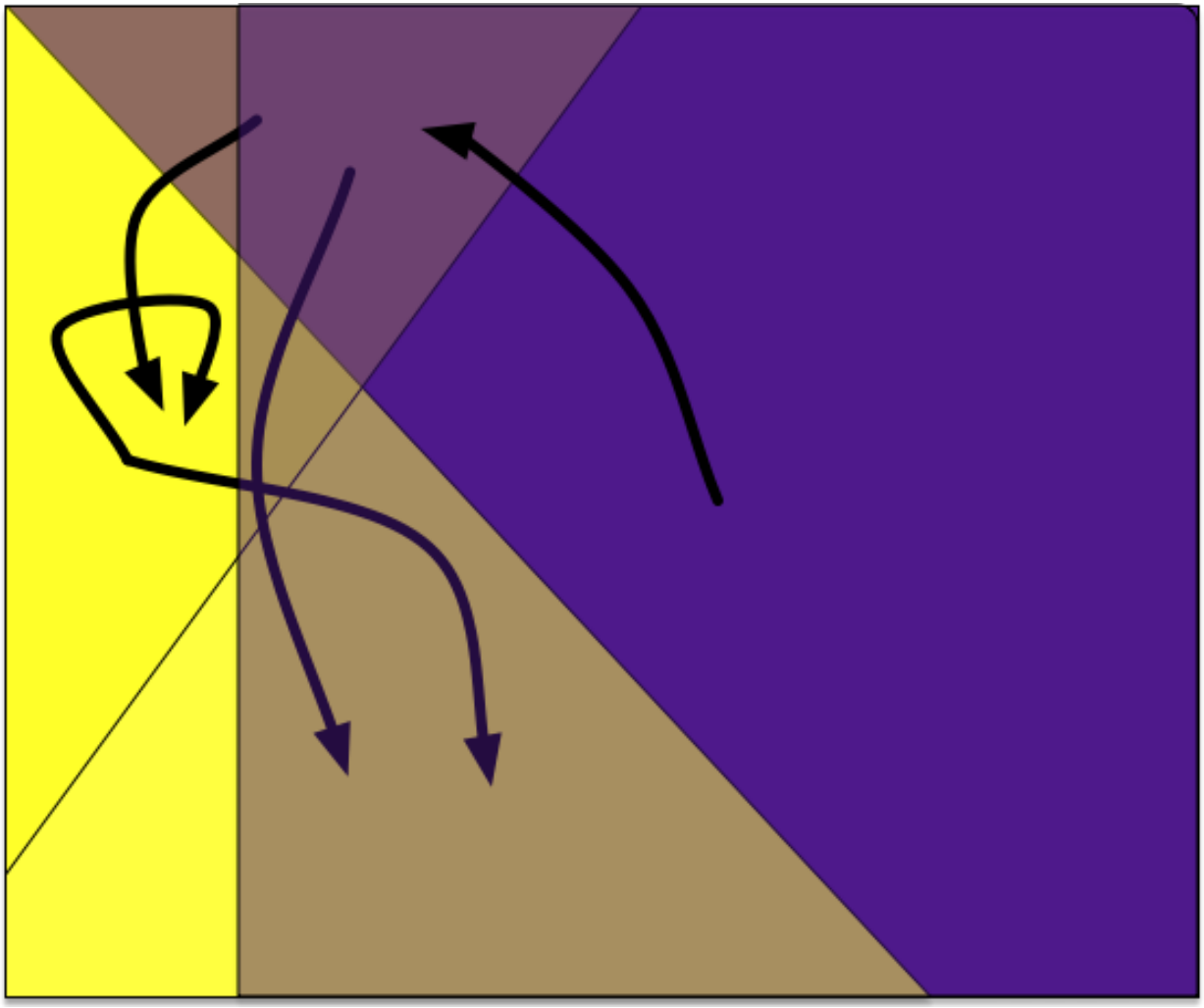
Construction of exact symbolic models



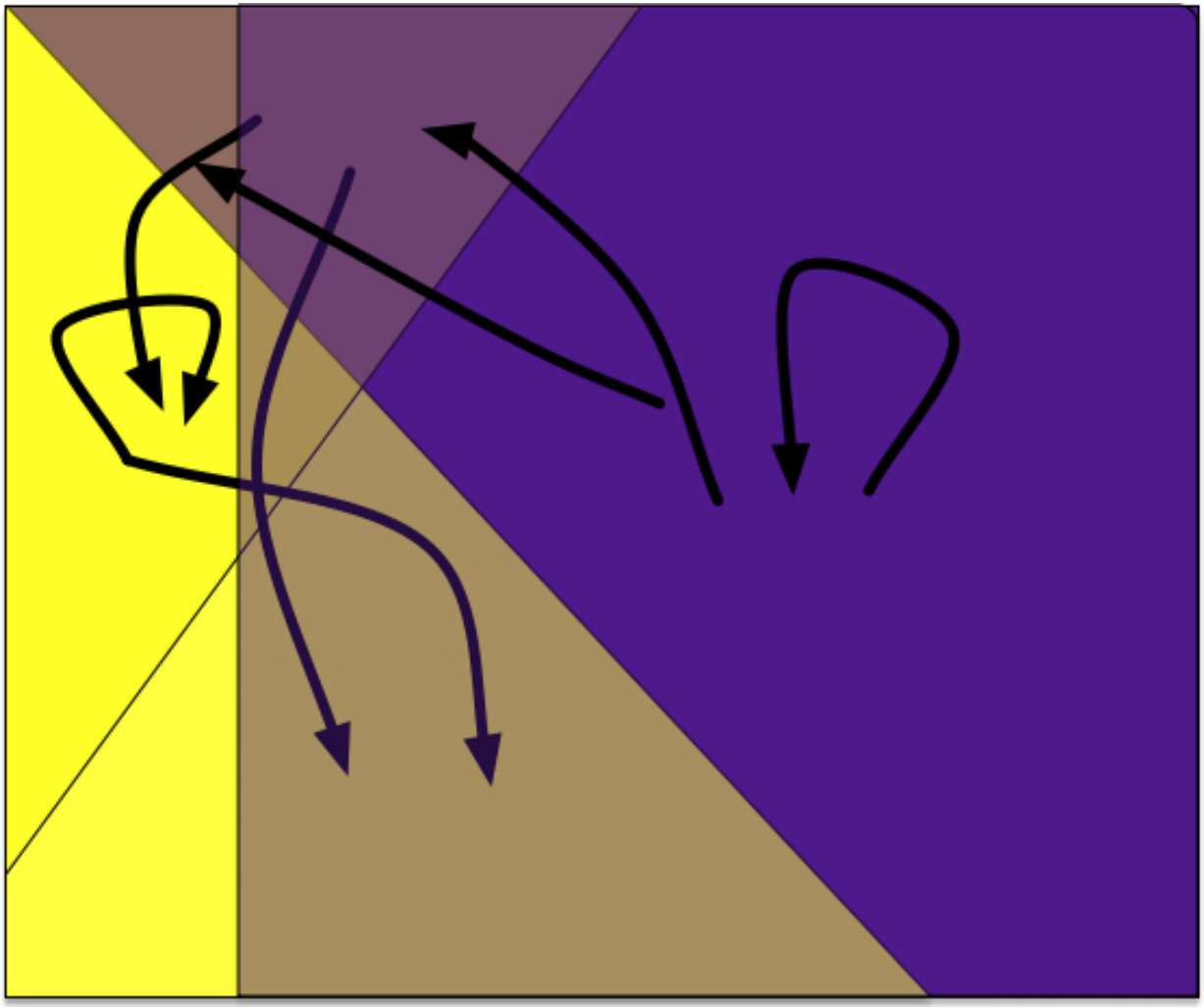
Construction of exact symbolic models



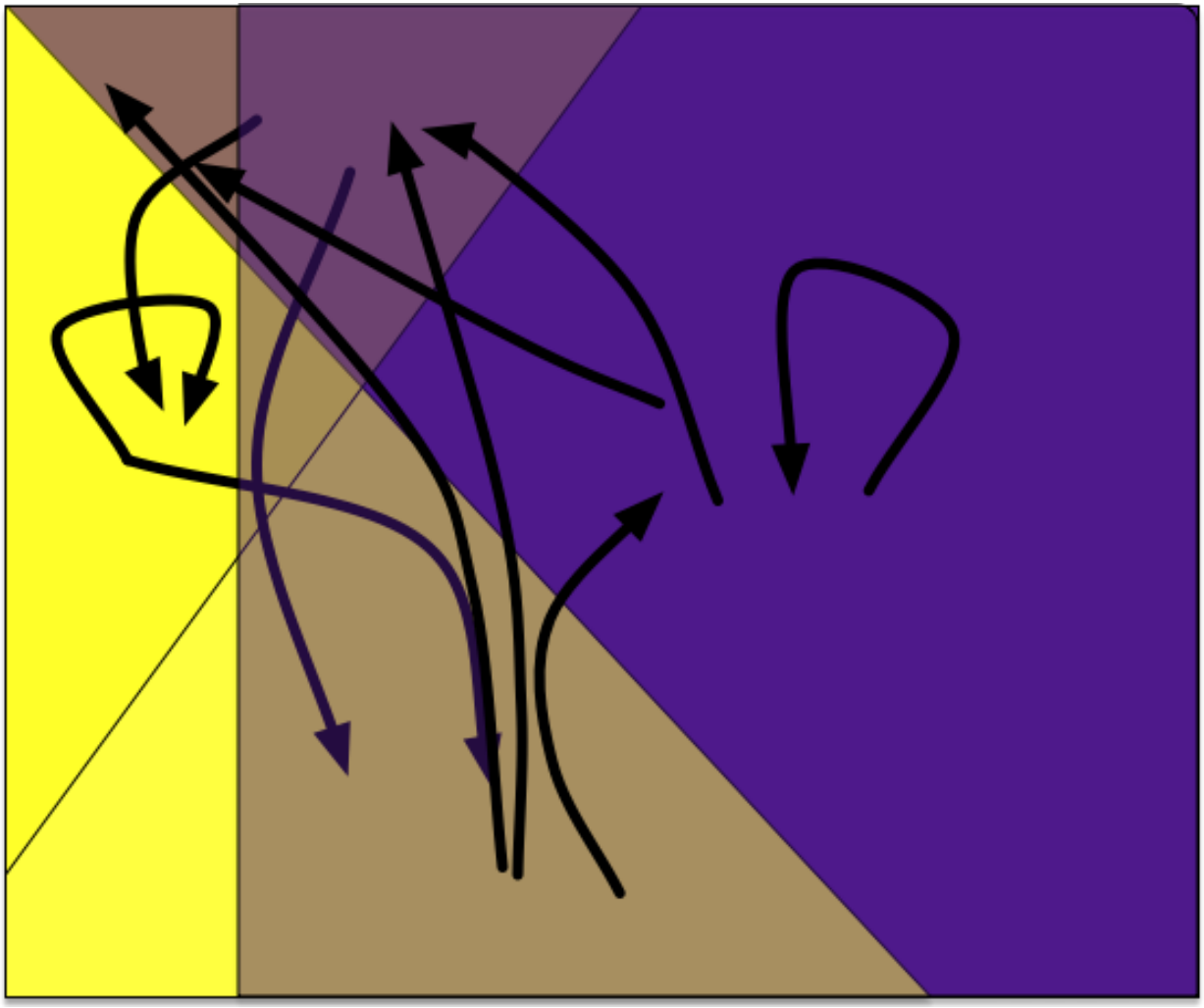
Construction of exact symbolic models



Construction of exact symbolic models



Construction of exact symbolic models

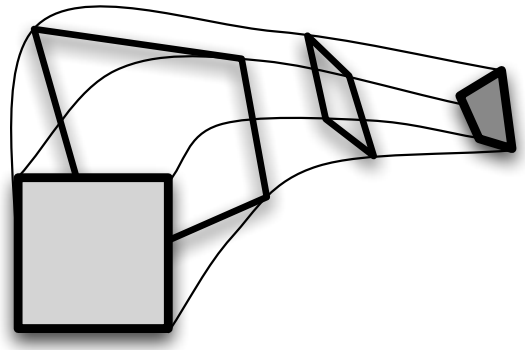


Incremental stability notion

Definition (Angeli 2002)

A control system Σ is incrementally input-to-state stable (δ-ISS) if it is forward complete and there exist a $\mathcal{KL}$ function β and a $\mathcal{K}_\infty$ function γ such that for any $t \in \mathbb{R}_0^+$, any $x, x' \in \mathbb{R}^n$, and any υ, $\upsilon' \in \mathcal{U}$ the following condition is satisfied:

$$\|\xi_{x\upsilon}(t) - \xi_{x'\upsilon'}(t)\| \leq \beta\left(\left\|x - x'\right\|, t\right) + \gamma\left(\left\|\upsilon - \upsilon'\right\|_\infty\right).$$



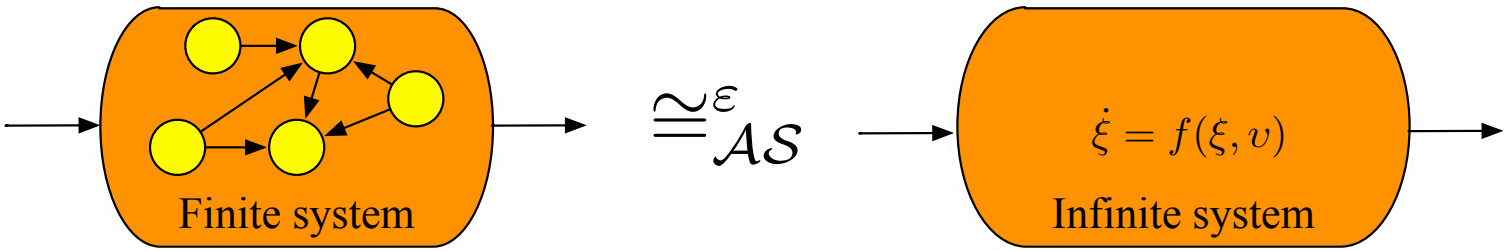
Reference:

Ang02 Angeli, David. "A Lyapunov approach to incremental stability properties." IEEE Transactions on Automatic Control 47.3 (2002): 410-421.

Existence of bisimilar symbolic models

Theorem (Girard, Pola, Tabuada 2008)

Let Σ be a control system and let $\varepsilon \in \mathbb{R}^+$ be any desired precision. If Σ is δ-ISS, then the restriction of $S_\tau(\Sigma)$ to any compact subset X of $\mathbb{R}^n$ is ε-approximate bisimilar to a finite system.



Idea of the proof: show that accumulation of successive "rounding errors" is compensated by the incremental stability of the system.

Reference:

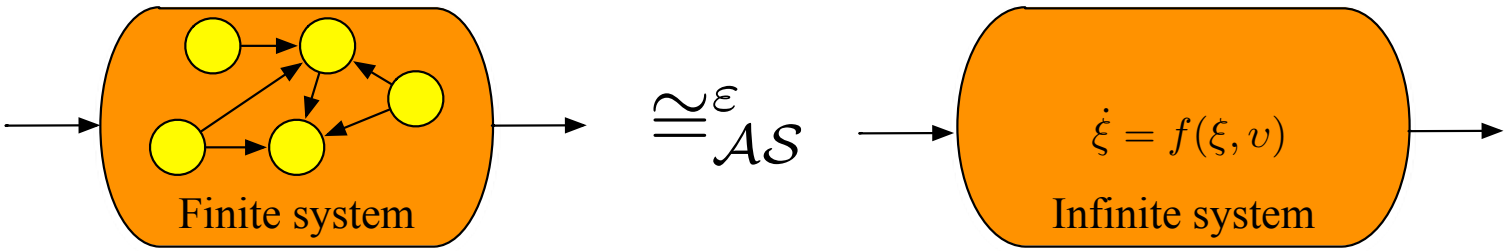
Gir08 Pola, Giordano, Antoine Girard, and Paulo Tabuada. "Approximately bisimilar symbolic models for nonlinear control systems." Automatica 44.10 (2008): 2508-2516.

PT09 (Chapters 10 and 11) Tabuada, Paulo. Verification and control of hybrid systems: a symbolic approach. Springer Science & Business Media, 2009. http://www.springer.com/gp/book/9781441902238

Existence of bisimilar symbolic models

Theorem (Girard, Pola, Tabuada 2008)

Let Σ be a control system and let $\varepsilon \in \mathbb{R}^+$ be any desired precision. If Σ is δ-ISS, then the restriction of $S_\tau(\Sigma)$ to any compact subset X of $\mathbb{R}^n$ is ε-approximate bisimilar to a finite system.



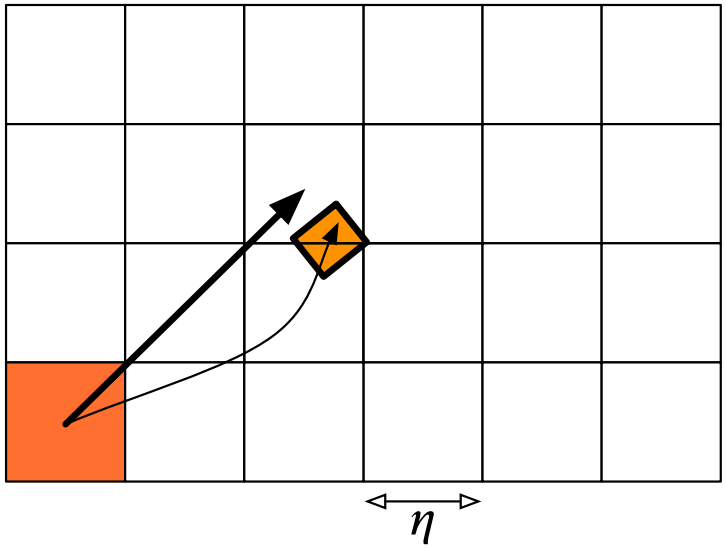
Idea of the proof: show that accumulation of successive "rounding errors" is compensated by the incremental stability of the system.

Reference:

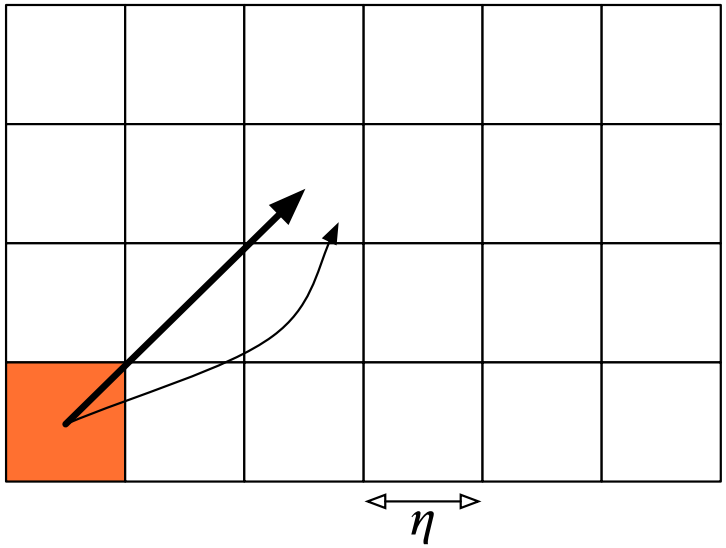
Gir08 Pola, Giordano, Antoine Girard, and Paulo Tabuada. "Approximately bisimilar symbolic models for nonlinear control systems." Automatica 44.10 (2008): 2508-2516.

PT09 (Chapters 10 and 11) Tabuada, Paulo. Verification and control of hybrid systems: a symbolic approach. Springer Science & Business Media, 2009. http://www.springer.com/gp/book/9781441902238

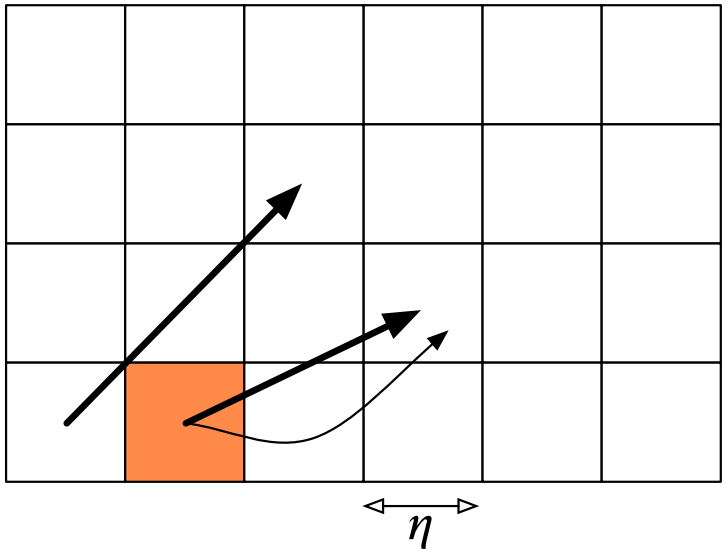
δ-ISS construction



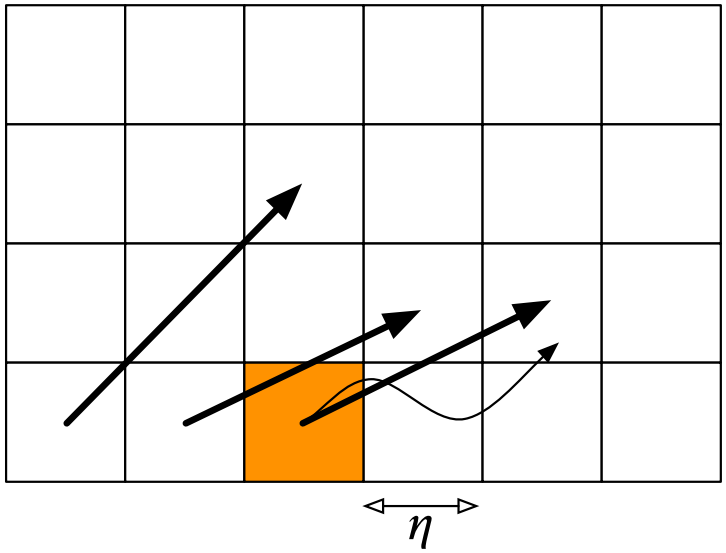
δ-ISS construction



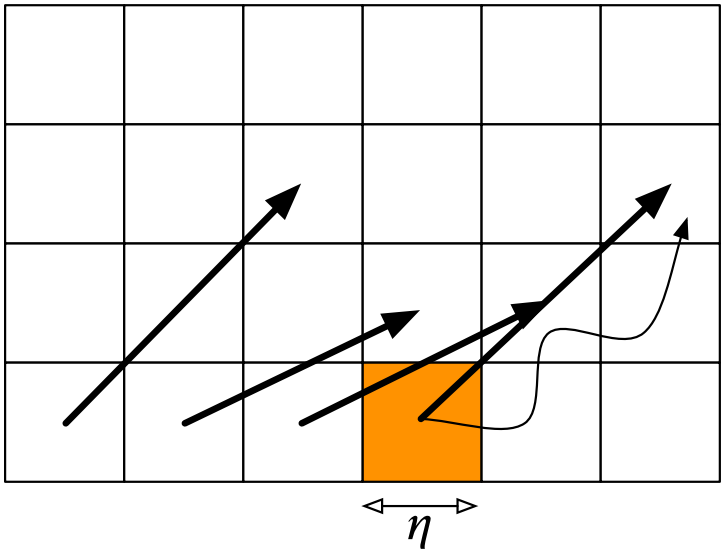
δ-ISS construction



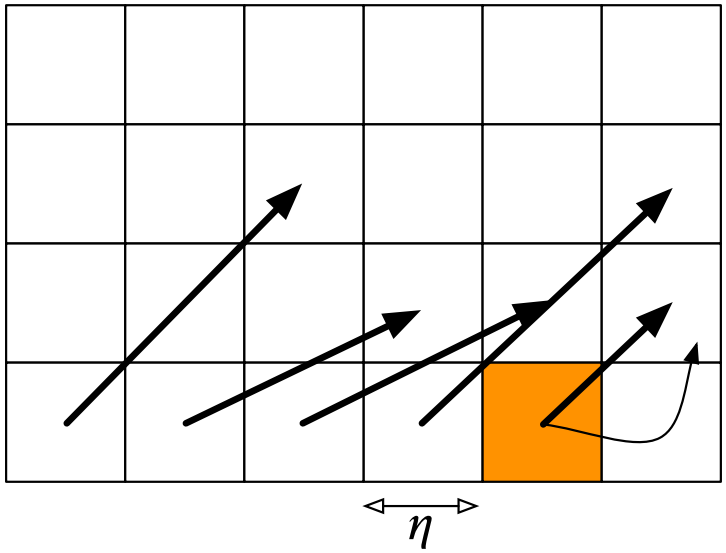
δ-ISS construction



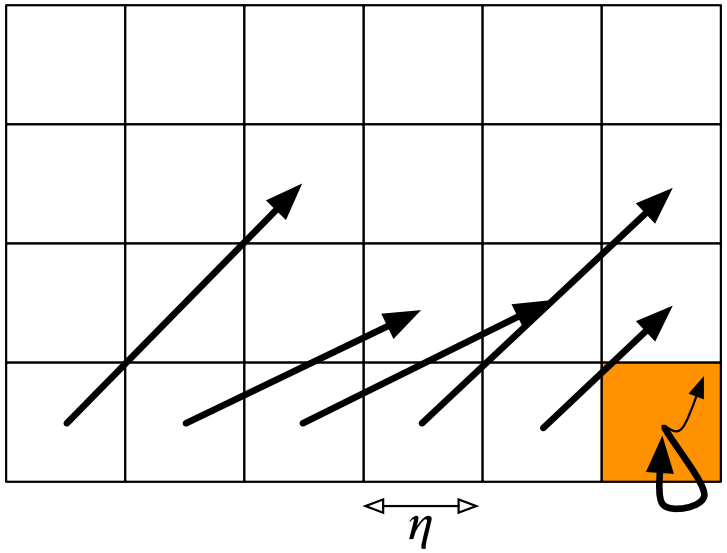
δ-ISS construction



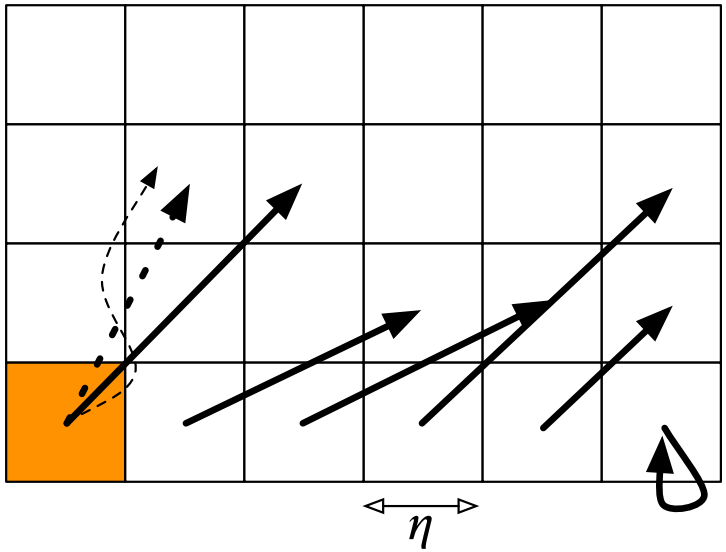
δ-ISS construction



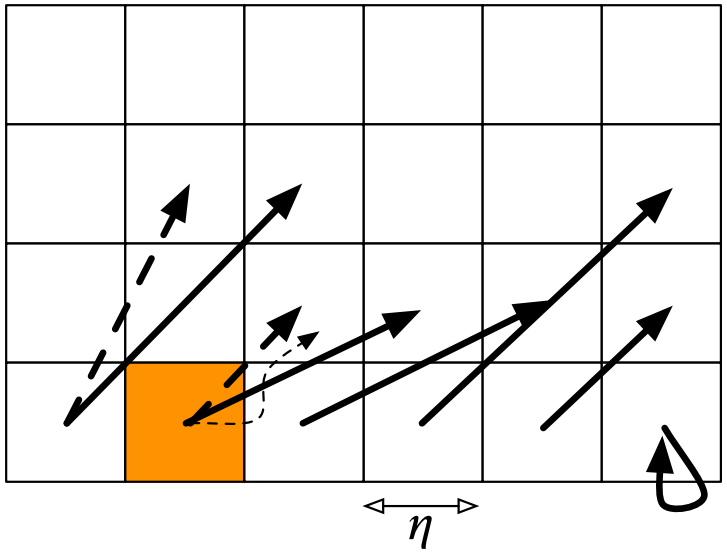
δ-ISS construction



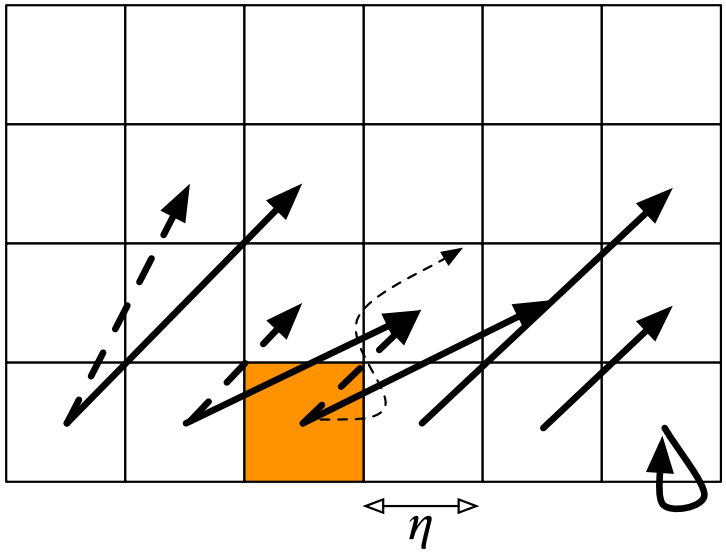
δ-ISS construction



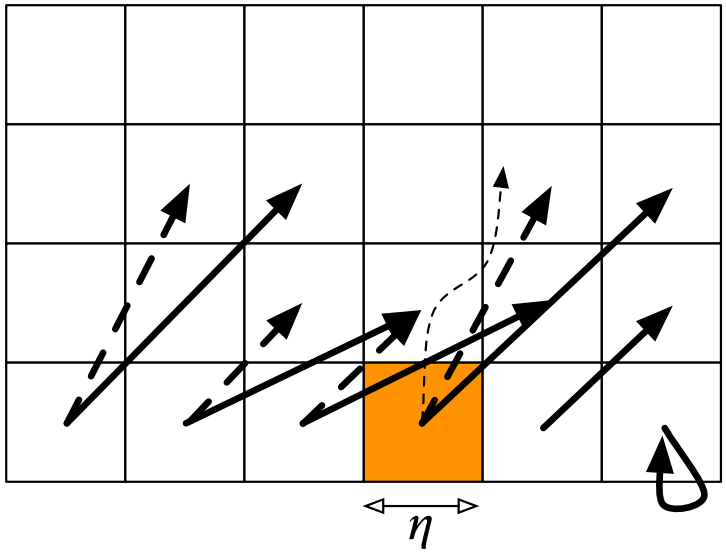
δ-ISS construction



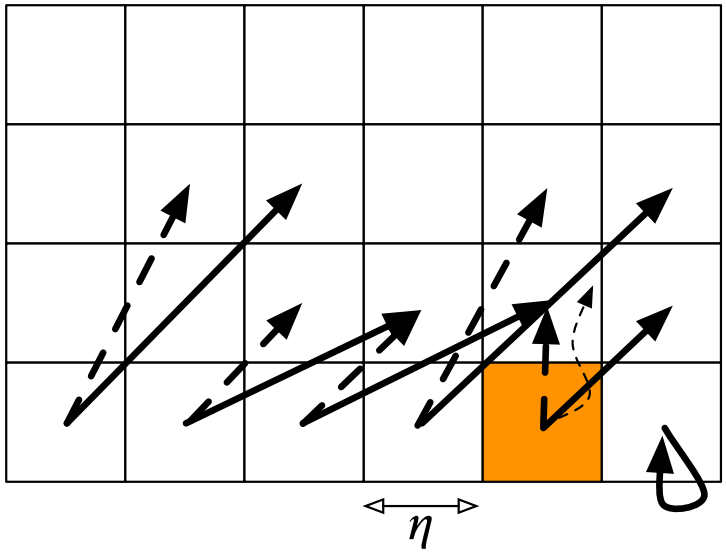
δ-ISS construction



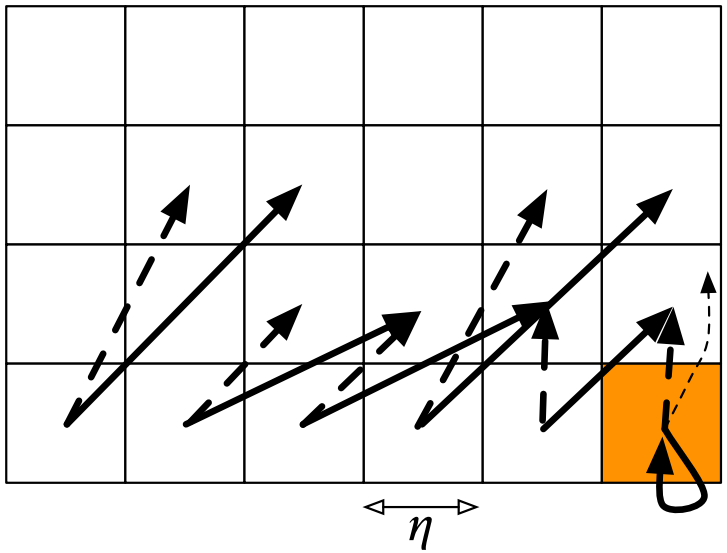
δ-ISS construction



δ-ISS construction



δ-ISS construction



Existence of symbolic models (without stability assumptions)

If Σ is not δ-ISS, are we still able to construct a finite system for Σ?

Reference:

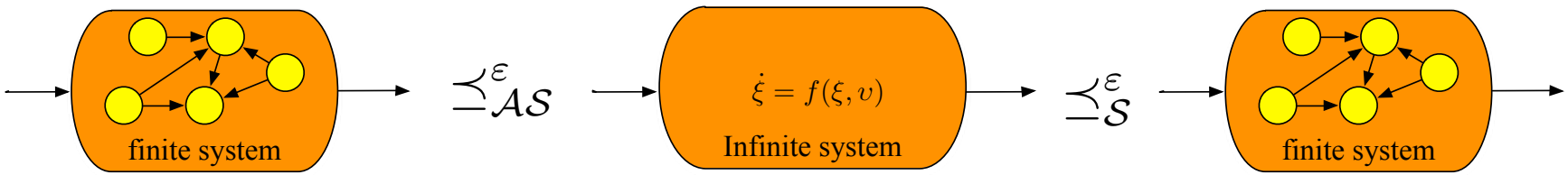
Zam12 Zamani, Majid, Pola, Giordano, Mazo Jr., Manuel, and Tabuada, Paulo. "Symbolic models for nonlinear control systems without stability assumptions." Automatic Control, IEEE Transactions on 57.7 (2012): 1804-1809.

Existence of symbolic models (without stability assumptions)

If Σ is not δ-ISS, are we still able to construct a finite system for Σ?

Theorem (Zamani, Pola, Mazo, Tabuada 2012)

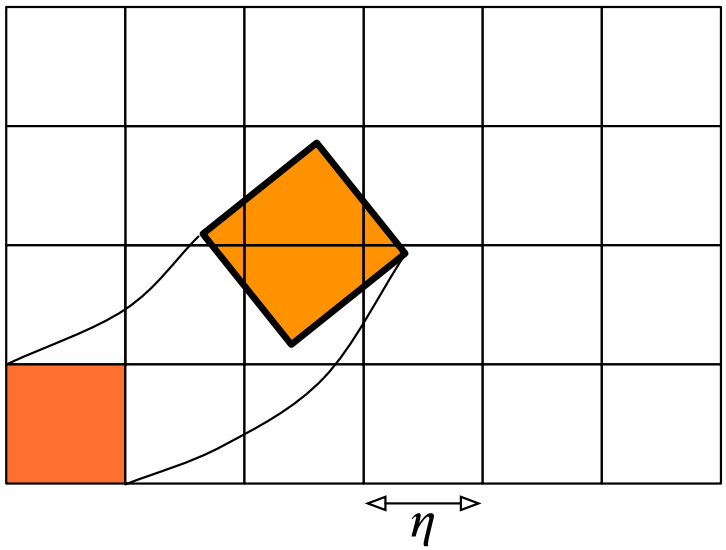
Let Σ be a control system and let $\varepsilon \in \mathbb{R}^+$ be any desired precision. The restriction of $S_\tau(\Sigma)$ to any compact subset X of $\mathbb{R}^n$ alternatingly ε-approximately simulates a finite system and is ε-approximately simulated by the finite system.



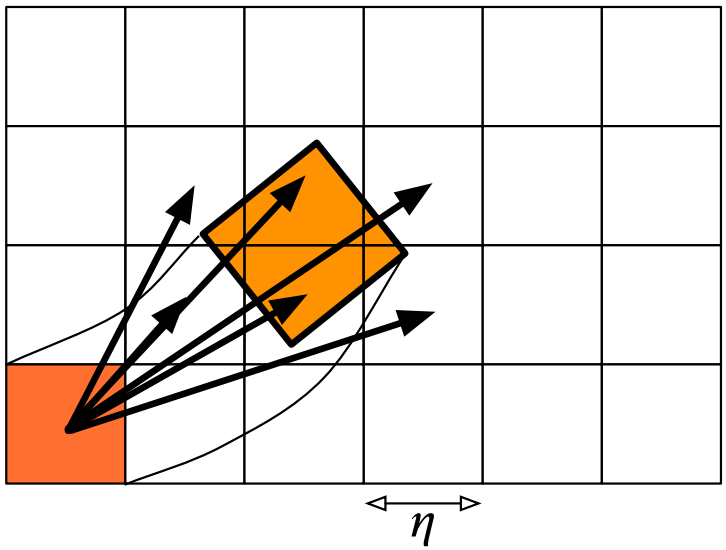
Reference:

Zam12 Zamani, Majid, Pola, Giordano, Mazo Jr., Manuel, and Tabuada, Paulo. "Symbolic models for nonlinear control systems without stability assumptions." Automatic Control, IEEE Transactions on 57.7 (2012): 1804-1809.

Unstable case construction



Unstable case construction



Transition system of hybrid automaton

- Hybrid automaton can be transformed into transition system by abstracting away time
- Consider hybrid automaton $H = (Q, X, \mathrm{Init}, f, \mathrm{Inv}, E, G, R)$ and "final" set of states $F \subseteq Q \times S$ (no inputs U)
- Define
  - $\mathrm{X} = Q \times X$, i.e., $\mathrm{x} = (q, x)$
  - $\mathrm{X}_0 = \mathrm{Init}$
  - $\mathrm{X}_F = F$
  - $Y = Q$

Transition system of hybrid automaton (cont.)

- transition relation $\delta : \mathrm{X} \to \mathrm{X}$, $\delta(\mathrm{x}) = \{\mathrm{x}' | \mathrm{x}' \in \mathrm{Post}(\mathrm{x})\}$, consists of two parts:
  - discrete transition relation δ_e for each edge $e = (q, q') \in E$:

$$\delta_e(\hat{q}, \hat{x}) = \begin{cases} \{q'\} \times R(e, \hat{x}) & \text{if } \hat{q} = q \text{ and } \hat{x} \in G(e) \\ \emptyset & \text{if } \hat{q} \neq q \text{ or } \hat{x} \notin G(e) \end{cases}$$

  - continuous transition relation δ_C:

$$\delta_C(\hat{q}, \hat{x}) = \{(\hat{q}', \hat{x}') \mid \hat{q}' = \hat{q} \text{ and } \exists t_f \geq 0,\ x(t_f) = \hat{x}' \wedge \forall t \in [0, t_f], x(t) \in \mathrm{Inv}(\hat{q})\}$$

    where $x(\cdot)$ is solution of

$$\dot{x} = f(\hat{q}, x) \text{ with } x(0) = \hat{x}$$

- Overall transition relation is then

$$\delta(\mathrm{x}) = \delta_C(\mathrm{x}) \cup \bigcup_{e \in E} \delta_e(\mathrm{x})$$

Transition system of hybrid automaton (cont.)

- Time has been abstracted away:
  we do not care how long it takes to get from s to s', we only care whether it is possible to get there eventually

$\rightarrow$ transition system captures sequence of events that hybrid system may experience, but *not* timing of these events

Timed-Automata

Definition (Timed Automaton)

A timed automaton TA is a sextuple $(L, \ell_0, \text{Act}, C, E, \text{Inv})$ where

- L is the set of finitely many locations (or nodes);
- $\ell_0 \in L$ is the initial location;
- Act is the set of finitely many actions;
- C is the set of finitely many real-valued clocks;
- $E \subseteq L \times \mathcal{B}(C) \times \text{Act} \times 2^C \times L$ is the set of edges;
- $\text{Inv} : L \to \mathcal{B}(C)$ assigns invariants to locations.

where $\mathcal{B}(C)$ to denote the set of clock constraints.

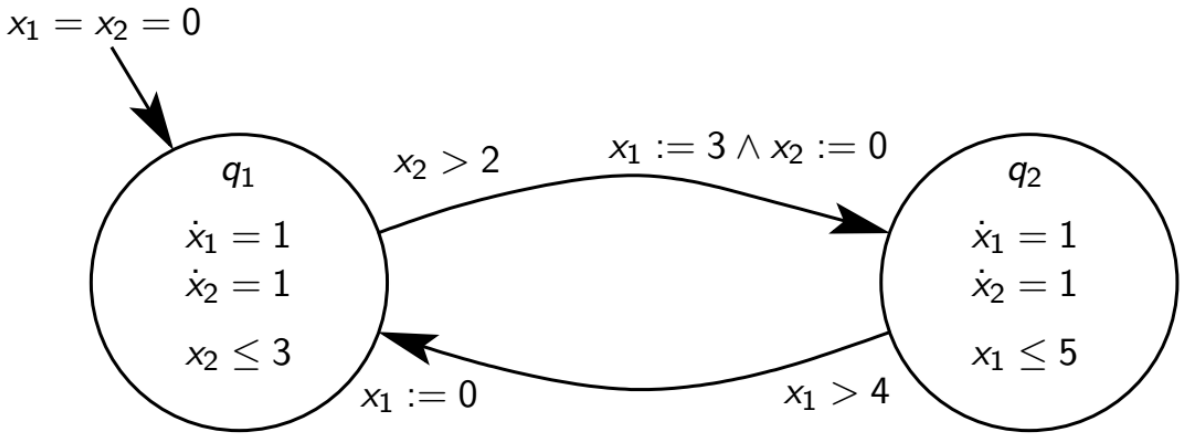
Reference:

Al94 Alur, Rajeev, and David L. Dill. "A theory of timed automata." Theoretical computer science 126.2 (1994): 183-235.

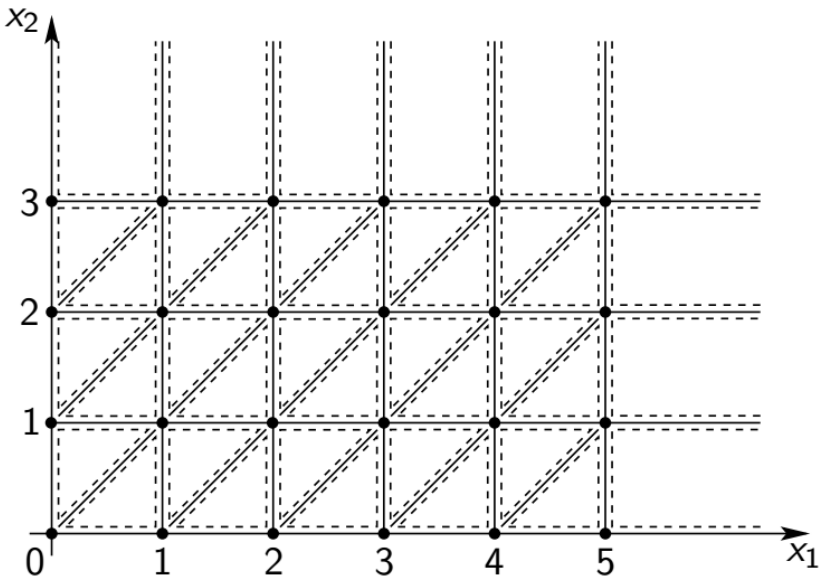
Timed automata (cont.)

- Given any TA whose definition involves rational and/or negative constants, we can define an equivalent TA only involving non-negative integers (by "scaling" and "shifting")
- Transition System corresponding to TA always has a finite bisimulation
- Standard bisimulation for timed automata is region graph

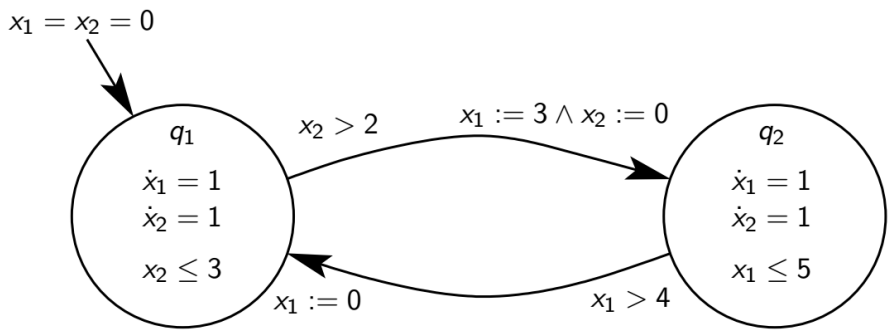
Region graph



$\Leftrightarrow$

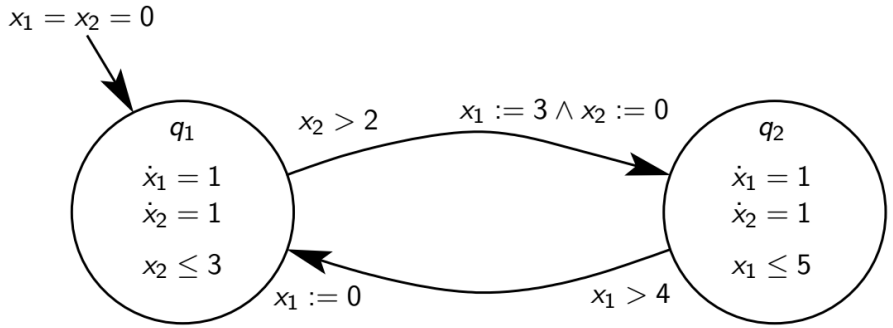


Construction of region graph



- Assume w.l.o.g. that all constants are non-negative integers
- Let C_i be largest constant with which x_i is compared in initial sets, guards, invariants and resets. In example: $C_1 = 5$ and $C_2 = 3$
- Knowing these bounds C_i, x_i *could* be compared with any integer $M \in \{0, 1 \ldots, C_i\}$ in guards, resets or initial condition set.
- Hence, discrete transitions of timed automaton may be able to "distinguish" states with $x_i < M$ from states with $x_i = M$ and from states with $x_i > M$

Construction of region graph (cont.)



- Add sets to candidate bisimulation:

  for $x_1 : x_1 \in (0,1), x_1 \in (1,2), x_1 \in (2,3), x_1 \in (3,4), x_1 \in (4,5), x_1 \in (5,\infty)$

  $x_1 = 0, x_1 = 1, x_1 = 2, x_1 = 3, x_1 = 4, x_1 = 5$

  for $x_2 : x_2 \in (0,1), x_2 \in (1,2), x_2 \in (2,3), x_2 \in (3,\infty)$

  $x_2 = 0, x_2 = 1, x_2 = 2, x_2 = 3$

- Products of all sets:

  $\{x \in \mathbb{R}^2 \mid x_1 \in (0,1) \wedge x_2 \in (0,1)\}$ $\{x \in \mathbb{R}^2 \mid x_1 \in (0,1) \wedge x_2 = 1\}$

  $\{x \in \mathbb{R}^2 \mid x_1 = 1 \wedge x_2 \in (0,1)\}$ $\{x \in \mathbb{R}^2 \mid x_1 = 1 \wedge x_2 = 1\}$

  $\{x \in \mathbb{R}^2 \mid x_1 \in (1,2) \wedge x_2 \in (3,\infty)\}$, etc.

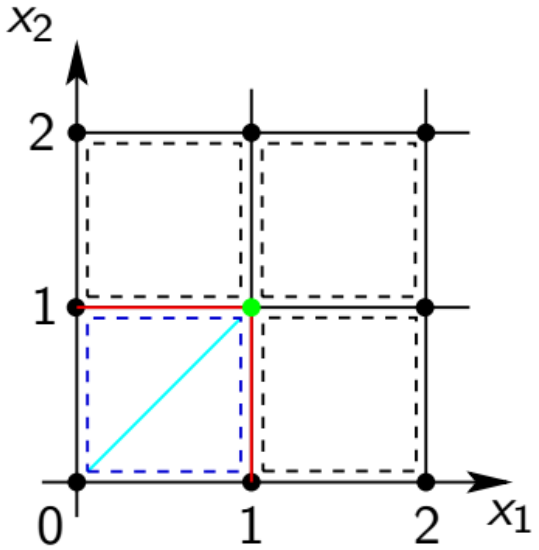
  define all sets in $\mathbb{R}^2$ that discrete dynamics can distinguish

Construction of region graph (cont.)

- Since $\dot{x}_1 = \dot{x}_2 = 1$, continuous states move diagonally up along 45° lines

→ by allowing time to flow timed automaton may distinguish points below diagonal of each square, points above diagonal, and points on the diagonal

- E.g., points above diagonal of square



$$\{x \in \mathbb{R}^2 \mid x_1 \in (0,1) \wedge x_2 \in (0,1)\}$$

will leave square through line $\{x \in \mathbb{R}^2 \mid x_1 \in (0,1) \wedge x_2 = 1\}$
Points below diagonal leave square through line

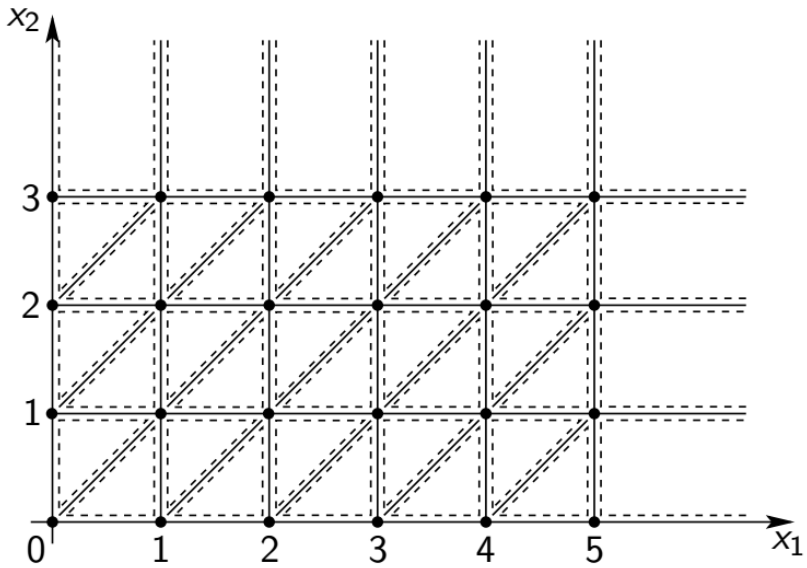
$$\{x \in \mathbb{R}^2 \mid x_1 = 1 \wedge x_2 \in (0,1)\}$$

Points on diagonal leave square through point $(1,1)$

Construction of region graph (cont.)

- Split each open square in three: two open triangles and open diagonal line segment

$\rightarrow$ is enough to generate bisimulation:



Theorem

The region graph is finite bisimulation of timed automaton

- Disadvantage: total number of regions in the region graph grows very quickly (exponentially) as n increases

Outline

Path fragment, path, trace

- nomenclature: *path*, *run*, *execution*, *trajectory* are synonyms

Definition

Consider the transition system $< S, \rightarrow, I, AP, L >$.

A *finite path fragment* is a finite state sequence $s_0 s_1 \dots s_n$ for some $n \geq 0$ such that $s_i \in Post(s_{i-1})$ for all $0 < i \leq n$.

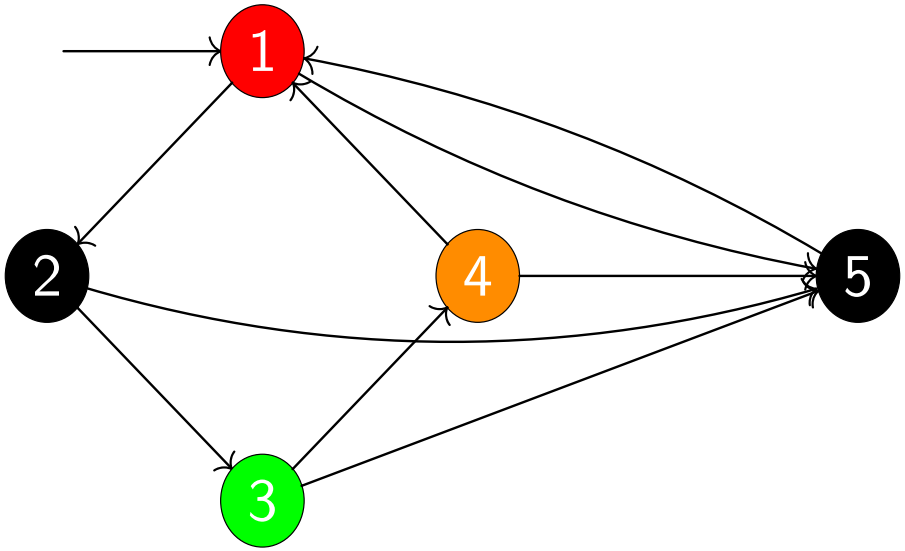
An *infinite path fragment* is an infinite state sequence $s_0 s_1 \dots$ such that $s_i \in Post(s_{i-1})$ for all $i > 0$.

A path fragment is *initial* if $s_0 \in I$.

A *path* is an initial infinite path fragment $\rightarrow Paths(TS)$

A trace is the "output" of a path: $L(s_0)L(s_1)\dots$

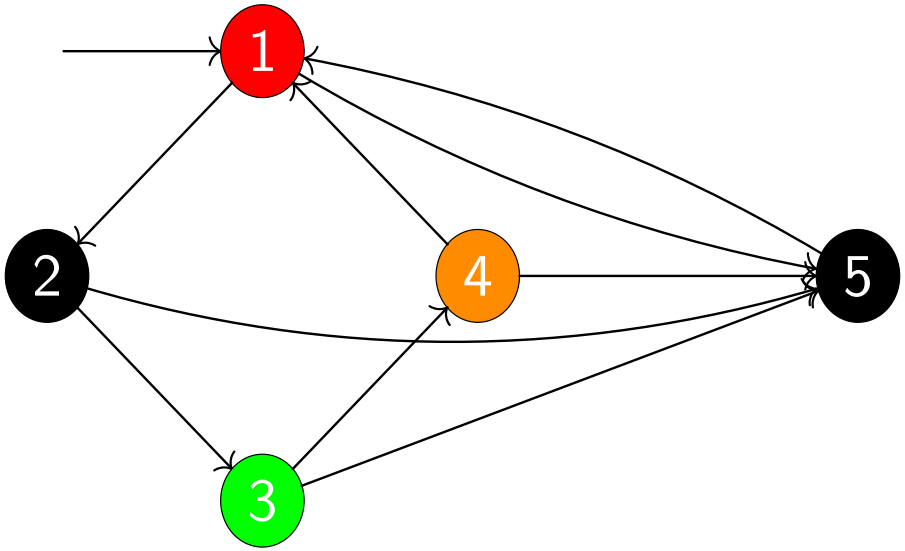
Path



Give an example of a finite initial path fragment:

Give an example of an initial infinite path fragment (a path):

Path



Give an example of a finite initial path fragment: 12341

Give an example of an initial infinite path fragment (a path): $(15)^{\omega}$

Some notational conventions

- let $\pi = s_0 s_1 \ldots$ be an infinite path fragment (notions below apply to finite path as well)
- for $j \geq 0$, the jth state of π, s_j is denoted by $\pi[j]$ (initial state is indexed by 0)
- for $j \geq 0$, the jth prefix of π, $s_0 s_1 \ldots s_j$ is denoted by $\pi[..j]$

$$\overbrace{s_0 s_1 \ldots s_{j-1} s_j}^{\pi[..j]} s_{j+1} \ldots$$

- for $j \geq 0$, the jth suffix of π, $s_j s_{j+1} \ldots$ is denoted by $\pi[j..]$

$$s_0 s_1 \ldots s_{j-1} \overbrace{s_j s_{j+1} \ldots}^{\pi[j..]}$$

- the set of infinite path fragments π with $\pi[0] = s$ is denoted by $Paths(s)$

Modal logics

- based on propositional and predicate logic
- used to reason about objects with modalities (expressed via modal operators)
- in particular, modal operators qualify temporal expressions
- in this lecture we shall focus on LTL, other temporal logics have been proposed:
  1. LTL: linear temporal logic
  2. CTL: computational tree logic
  3. MTL: metric temporal logic ("robust" LTL)
  4. STL: signal temporal logic (continuous time TL)
  5. CTL*,...

Syntax of LTL

- $\varphi ::= \text{true} \mid a \mid \varphi \wedge \varphi \mid \neg\varphi \mid \bigcirc\varphi \mid \varphi \,\mathsf{U}\, \varphi, \quad a \in AP$

- alternative expression of more formulae

$$\begin{aligned} \varphi_1 \vee \varphi_2 &= \neg(\neg\varphi_1 \wedge \neg\varphi_2) \\ \varphi_1 \Rightarrow \varphi_2 &= \neg\varphi_1 \vee \varphi_2 \end{aligned}$$

  and of two temporal modalities

$$\begin{aligned} \Diamond\varphi &= \text{true}\,\mathsf{U}\,\varphi \\ \Box\varphi &= \neg\Diamond\neg\varphi \end{aligned}$$

Alternative syntax in the literature

- you may encounter the following notations:

$$\begin{array}{lcl} \mathsf{X}\varphi & : & \bigcirc\varphi \\ \mathsf{F}\varphi & : & \Diamond\varphi \\ \mathsf{G}\varphi & : & \Box\varphi \end{array}$$

(notation on left-hand side from [CGP99], on right-hand side from [BK08])

Semantics of LTL

$$TS \models \varphi \text{ iff } \forall s \in I : s \models \varphi$$

where

$$s \models \varphi \text{ iff } \forall \pi \in Paths(s) : \pi \models \varphi$$

Semantics of LTL

$$TS \models \varphi \text{ iff } \forall s \in I : s \models \varphi$$

where

$$s \models \varphi \text{ iff } \forall \pi \in Paths(s) : \pi \models \varphi$$

and where (cf. LTL syntax)

$$\begin{aligned}
\pi &\models \text{true} \\
\pi &\models a \quad \text{iff} \quad a \in L(\pi[0]) \\
\pi &\models \varphi \wedge \psi \quad \text{iff} \quad \pi \models \varphi \wedge \pi \models \psi \\
\pi &\models \neg\varphi \quad \text{iff} \quad \pi \not\models \varphi \\
\pi &\models \bigcirc\varphi \quad \text{iff} \quad \pi[1..] \models \varphi \\
\pi &\models \varphi \,\mathsf{U}\, \psi \quad \text{iff} \quad \exists i \geq 0 : \pi[i..] \models \psi \wedge \forall 0 \leq j < i : \pi[j..] \models \varphi
\end{aligned}$$

LTL properties for the traffic light model

- how to express the property

  "the light is infinitely often red"

  by an LTL formula?
- $\square\Diamond \text{red}$

LTL properties for the traffic light model

- how to express the property

  "the light is infinitely often red"

  by an LTL formula?
- $\square\Diamond \text{red}$

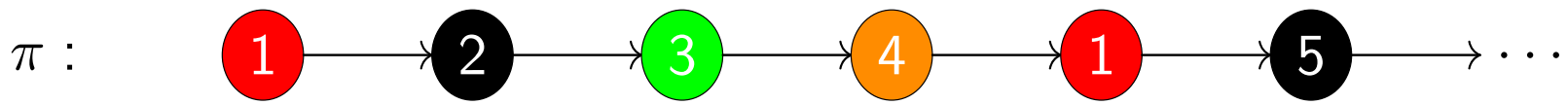
- how to express the property

  "once green, the light cannot become red immediately"

  by an LTL formula?
- $\square(\text{green} \Rightarrow \neg\bigcirc \text{red})$

Verification of LTL specs is over linear-time paths

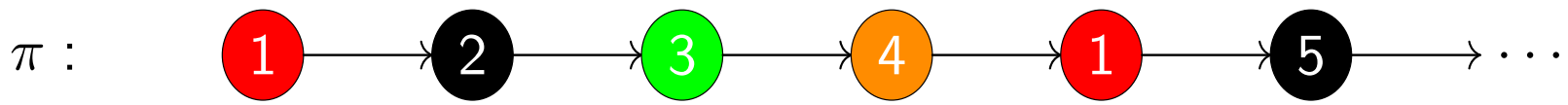
- back to the traffic light model, consider the following path:



- question: $\pi \models$ red?

Verification of LTL specs is over linear-time paths

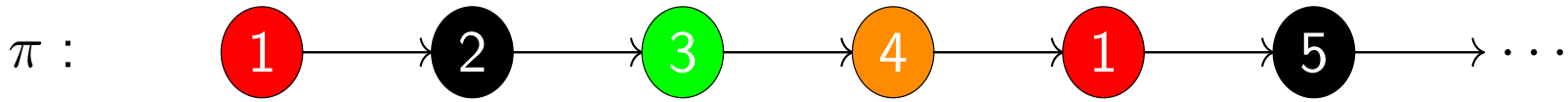
- back to the traffic light model, consider the following path:



- question: $\pi \models$ red?
- answer: yes

Verification of LTL specs

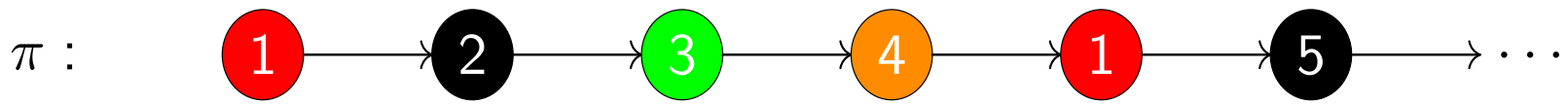
- back to the traffic light model, consider the following path:



- question: $\pi \models \bigcirc\bigcirc \text{red}$?

Verification of LTL specs

- back to the traffic light model, consider the following path:



- question: $\pi \models \bigcirc\bigcirc \text{red}$?
- answer: no

Classes of LTL specifications

- *invariance* properties: $\Box\varphi$, where $\varphi ::= \text{true} \mid a \mid \varphi \wedge \varphi \mid \neg\varphi$

Classes of LTL specifications

- *invariance* properties: $\Box\varphi$, where $\varphi ::= \text{true} \mid a \mid \varphi \wedge \varphi \mid \neg\varphi$
- example: $\Box\neg\text{red}$

Classes of LTL specifications

- *safety properties*: "nothing bad ever happens"

Classes of LTL specifications

- *safety properties*: "nothing bad ever happens"
- **example:** "a red light is immediately preceded by amber"
- **question:** how can we express this property in LTL?

Classes of LTL specifications

- *safety properties*: "nothing bad ever happens"
- **example:** "a red light is immediately preceded by amber"
- **question:** how can we express this property in LTL?
- **answer:** $\Box(\bigcirc \text{red} \Rightarrow \text{amber})$

Classes of LTL specifications

Two major classes:

- *liveness properties*: "something good eventually happens"

Classes of LTL specifications

Two major classes:

- *liveness properties*: "something good eventually happens"
- **e.g.**: "the light is infinitely often red"
- **question:** how do we express this property in LTL?

Classes of LTL specifications

Two major classes:

- *liveness properties*: "something good eventually happens"
- **e.g.**: "the light is infinitely often red"
- **question:** how do we express this property in LTL?
- **answer:** $\square\lozenge$red

LTL and Büchi Automata

Definition (Büchi Automaton)

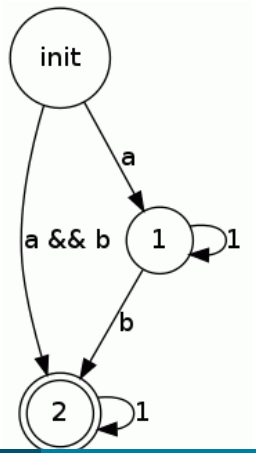
A Büchi Automaton is an automaton in which the accepting condition is given by infinitely often visiting one state in the set of *final* states.

Theorem (Büchi Automata for LTL)

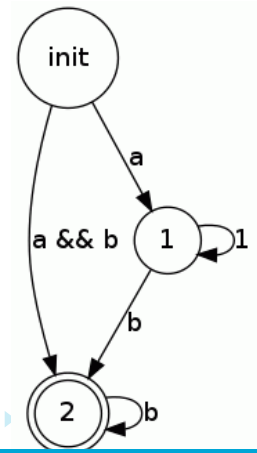
Every LTL formula accepts an equivalent Büchi Automaton representation, i.e. one automaton accepting the same language as the LTL formula.

`http://www.lsv.fr/~gastin/ltl2ba/index.php`

$a \wedge \Diamond b$



$a \wedge \Diamond\Box b$



Comments on LTL model checking

Model Checking: Does *model* satisfy the *formalized property*?
Answer: Yes/No(+ counter-example)

- automated and computer aided technique(s)
- **guarantees are provided with respect to models!**
- many advanced techniques for scaling-up model checking:
  - on-the-fly procedures
  - symbolic procedures with BDD,
  - BMC,
  - use of counter-examples,
  - Craig interpolation and induction,
  - *abstractions and refinements*
  - compositional verification
- SPIN is an LTL model checker widely used

Outline

Tools: Verification

There is a large number of tools that enable different levels of verification of Hybrid Systems:

- **Reachability computation:**
  - **MATISSE** - Approximate (bi)simulations linear systems, reachability/safety, zonotopes.
  - **Ariadne, d/dt, SpaceEX, Breach, HSolver** - Precise computation of reachability/safety for hybrid systems, including non-linear ODEs.
  - **MPT, Hybrid Toolbox, HYSDEL** - MPC tools, convex optimization, reachability/safety.
  - **(ProHVer) PHVer** - (Probabilistic) reachability/safety computation for linear hybrid automata, compositional reasoning.

Tools: Verification

There is a large number of tools that enable different levels of verification of Hybrid Systems:

- **Complex specifications:**
  - **KeYmaera** - Differential Dynamic Logic (dL) verification. Allows non-linearities in ODE's, guards, etc.
  - **HybridSAL** - LTL model checking, relies on SAT and SMT solvers. Allows non-linearities in ODE's, guards, etc.
  - **S-TaLiRo** - Metric Temporal Logic verification. Temporal logic robustness.
  - **FAUST**^{2} - Probabilistic CTL verification of discrete-time Markov Processes.
  - **Checkmate** - Always CTL verification for linear hybrid automata.
- **Special types of Hybrid Systems:**
  - **UPPAAL** - Timed Automata
  - **VeriSiMPL** - Max-Plus Linear Systems.
  - **SimHPN** - Hybrid Petri Nets.

Tools: Synthesis

- **PESSOA** - Non-linear dynamics, safety and reachability. (Not maintained)
- **SCOTS** - Non-linear dynamics, safety and reachability. (Faster implementation of PESSOA)
- **CoSyMa** - Incrementally stable switched dynamics, safety or reachability.
- **LTLMoP** - Single integrator dynamics, generalize reactive properties.
- **TuLiP** - Affine dynamics, LTL properties, receding horizon.
- **LTLCon** - Affine Dynamics, LTL properties.
- **Con-Pas2** -Piecewise Affine Systems, LTL properties.
- **UPPAAL - TiGA/CORA/Stratego** - Synthesis versions of UPPAAL, for timed-automata.

A thorough list of tools for synthesis and verification can be found here:
http://hybrid-systems.ieeecss.org/tc-hybrid/tools-hybrid-systems

Bibliography

- [PT09] Tabuada, Paulo. Verification and control of hybrid systems: a symbolic approach. Springer Science & Business Media, 2009. http://www.springer.com/gp/book/9781441902238
- Girard, Antoine, and George J. Pappas. "Approximation metrics for discrete and continuous systems." Automatic Control, IEEE Transactions on 52.5 (2007): 782-798.
- Pola, Giordano, and Paulo Tabuada. "Symbolic models for nonlinear control systems: Alternating approximate bisimulations." SIAM Journal on Control and Optimization 48.2 (2009): 719-733.
- Pola, Giordano, Antoine Girard, and Paulo Tabuada. "Approximately bisimilar symbolic models for nonlinear control systems." Automatica 44.10 (2008): 2508-2516.
- Angeli, David. "A Lyapunov approach to incremental stability properties." IEEE Transactions on Automatic Control 47.3 (2002): 410-421.
- [Zam12] Zamani, Majid, Pola, Giordano, Mazo Jr., Manuel, and Tabuada, Paulo. "Symbolic models for nonlinear control systems without stability assumptions." Automatic Control, IEEE Transactions on 57.7 (2012): 1804-1809.