Modeling & Control of Hybrid Systems

Chapter 2 — Modeling frameworks [1]

Manuel Mazo Jr.

Utrecht, June 26th, 2020

[1] Based on the original slides from Bart De Schutter

Outline

Recap and motivation

- Many modeling frameworks for hybrid systems
  $\Rightarrow$ trade-off: modeling power $\leftrightarrow$ decision power, tractability
- Hybrid automata:
  - very general, high modeling power, but low decision power
  - analysis and control $\rightarrow$ computationally hard
    (NP-hard, undecidable problems)

Simulation vs Verification vs Analysis

- Computer simulation and verification tools: Modelica, HyTech, KRONOS, Chi, 20-sim, UPPAAL, ...
  - + simulation models can represent plant with high degree of detail (high modeling power)
  - − computationally very demanding for large systems
  - − difficult to understand from simulation how behavior depends on model parameters
  - + formal verification can provide better insight, and provides stronger guarantees
  - − computationally are often more demanding than simulation
- In this chapter: special classes of hybrid systems for which *tractable* analysis and control design techniques are available (cf. next chapters)

Outline

Piecewise affine systems (PWA) systems

Definition (PWA system)

A discrete-time PWA system is given by the difference equations:

$$\begin{aligned} x(k+1) &= A_i x(k) + B_i u(k) + f_i \\ y(k) &= C_i x(k) + D_i u(k) + g_i \end{aligned} \quad \text{for} \begin{bmatrix} x(k) \\ u(k) \end{bmatrix} \in \Omega_i, \ \ i = 1, \dots, N$$

where $\Omega_1, \dots, \Omega_N$ are convex polyhedra (i.e., given by finite number of linear inequalities) in input/state space with non-overlapping interiors.

- PWA can be used as approximation of nonlinear model

$$\begin{aligned} x(k+1) &= \mathcal{N}_x(x(k), u(k)) \\ y(k) &= \mathcal{N}_y(x(k), u(k)) \end{aligned}$$

$\rightarrow$ "simplest" extension of linear systems that can still model non-linear & non-smooth processes with arbitrary accuracy
$+$ are capable of handling hybrid phenomena

PWA systems

Definition (PWA system)

A discrete-time PWA system is given by the difference equations:

$$\begin{aligned} x(k+1) &= A_i x(k) + B_i u(k) + f_i \\ y(k) &= C_i x(k) + D_i u(k) + g_i \end{aligned} \quad \text{for } \begin{bmatrix} x(k) \\ u(k) \end{bmatrix} \in \Omega_i, \;\; i = 1, \dots, N$$

where $\Omega_1, \dots, \Omega_N$ are convex polyhedra (i.e., given by finite number of linear inequalities) in input/state space with non-overlapping interiors.

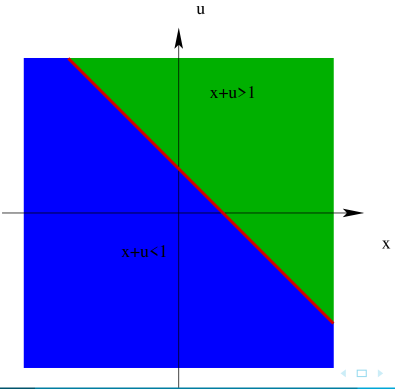
- Extensions to continuous-time and not necessarily polytopic partitionings.

Example of PWA model

Integrator with upper saturation:

$$x(k+1) = \begin{cases} x(k) + u(k) & \text{if } x(k) + u(k) \leqslant 1 \\ 1 & \text{if } x(k) + u(k) \geqslant 1 \end{cases}$$

$$y(k) = x(k)$$



Outline

Recap boolean logic

- Boolean operators:
  $\wedge$ (and), $\vee$ (or), $\sim$ (not), $\Rightarrow$ (implies), $\Leftrightarrow$ (iff), $\oplus$ (xor)

X_1	X_2	$X_1 \wedge X_2$	$X_1 \vee X_2$	$\sim X_1$	$X_1 \Rightarrow X_2$	$X_1 \Leftrightarrow X_2$	$X_1 \oplus X_2$
T	T	T	T	F	T	T	F
T	F	F	T	F	F	F	T
F	T	F	T	T	T	F	T
F	F	F	F	T	T	T	F

- Properties:
  - $X_1 \Rightarrow X_2$ is same as $\sim X_1 \vee X_2$
  - $X_1 \Rightarrow X_2$ is same as $\sim X_2 \Rightarrow \sim X_1$
  - $X_1 \Leftrightarrow X_2$ is same as $(X_1 \Rightarrow X_2) \wedge (X_2 \Rightarrow X_1)$

From Boolean logic to Integer programs I

- Associate with literal X_i logical variable $\delta_i \in \{0, 1\}$:
  $\delta_i = 1$ iff $X_i = \mathrm{T}$, $\delta_i = 0$ iff $X_i = \mathrm{F}$
  $\rightarrow$ compound statement can be transformed into
  linear integer program
- Examples:
  - $X_1 \wedge X_2$ equivalent to $\delta_1 = \delta_2 = 1$
  - $X_1 \vee X_2$ equivalent to $\delta_1 + \delta_2 \geq 1$
  - $\sim X_1$ equivalent to $\delta_1 = 0$
  - $X_1 \Rightarrow X_2$ equivalent to $\delta_1 - \delta_2 \leq 0$
  - $X_1 \Leftrightarrow X_2$ equivalent to $\delta_1 - \delta_2 = 0$
  - $X_1 \oplus X_2$ equivalent to $\delta_1 + \delta_2 = 1$
- For $f : \mathbb{R}^n \rightarrow \mathbb{R}$ and $x \in \mathcal{X}$ with $\mathcal{X}$ bounded, define

$$M \stackrel{\text{def}}{=} \max_{x \in \mathcal{X}} f(x) \qquad m \stackrel{\text{def}}{=} \min_{x \in \mathcal{X}} f(x)$$

From Boolean logic to Integer programs II

- Equivalences:
  * $[f(x) \leq 0] \wedge [\delta = 1]$ true iff $f(x) - \delta \leq -1 + m(1 - \delta)$
  * $[f(x) \leq 0] \vee [\delta = 1]$ true iff $f(x) \leq M\delta$
  * $\sim[f(x) \leq 0]$ true iff $f(x) \geq \varepsilon$ (with ε machine precision)
  * $[f(x) \leq 0] \Rightarrow [\delta = 1]$ true iff $f(x) \geq \varepsilon + (m - \varepsilon)\delta$
  * $[f(x) \leq 0] \Leftrightarrow [\delta = 1]$ true iff $\begin{cases} f(x) \leq M(1-\delta) \\ f(x) \geq \varepsilon + (m - \varepsilon)\delta \end{cases}$

- Product $\delta_1\delta_2$ can be replaced by auxiliary variable $\delta_3 = \delta_1\delta_2$:

$$\delta_3 = \delta_1\,\delta_2 \quad \text{is equivalent to} \quad \begin{cases} -\delta_1 + \delta_3 \leq 0 \\ -\delta_2 + \delta_3 \leq 0 \\ \delta_1 + \delta_2 - \delta_3 \leq 1 \end{cases}$$

From Boolean logic to Integer programs III

- $\delta f(x)$ can be replaced by auxiliary real variable $y = \delta f(x)$:

$$y = \delta f(x) \quad \text{is equivalent to} \quad \begin{cases} y \leq M\delta \\ y \geq m\delta \\ y \leq f(x) - m(1-\delta) \\ y \geq f(x) - M(1-\delta) \end{cases}$$

MLD systems

Definition (MLD)

A Mixed Logical Dynamical system is a system satisfying the following linear difference equation and linear inequalities:

$$\begin{aligned} x(k+1) &= Ax(k) + B_1 u(k) + B_2 \delta(k) + B_3 z(k) \\ y(k) &= Cx(k) + D_1 u(k) + D_2 \delta(k) + D_3 z(k) \\ E_1 x(k) &+ E_2 u(k) + E_3 \delta(k) + E_4 z(k) \leq g_5, \end{aligned}$$

where the state (and analogously u and y) is partitioned as $x(k) = [\, x_{\mathrm{r}}{}^{\mathrm{T}}(k)\; x_{\mathrm{b}}{}^{\mathrm{T}}(k) \,]^{\mathrm{T}}$ with $x_{\mathrm{r}}(k)$ real-valued and $x_{\mathrm{b}}(k)$ boolean; $z(k)$ and $\delta(k)$ are real-valued and boolean auxiliary variables resp.

- **Applications:** PWA systems, systems with discrete inputs, qualitative inputs, bilinear systems, finite state machines.

Reference: A. Bemporad and M. Morari, "Control of systems integrating logic, dynamics, and constraints," *Automatica*, vol. 35, no. 3, pp. 407–427, March 1999.

Example

- Consider the PWA system:

$$x(k+1) = \begin{cases} 0.8\,x(k) + u(k) & \text{if } x(k) \geq 0 \\ -0.8\,x(k) + u(k) & \text{if } x(k) < 0 \end{cases}$$

where $x(k) \in [-10, 10]$ and $u(k) \in [-1, 1]$

- Associate binary variable $\delta(k)$ to condition $x(k) \geq 0$ such that $[\delta(k) = 1] \Leftrightarrow [x(k) \geq 0]$ or

$$-m\delta(k) \leq x(k) - m$$
$$-(M+\varepsilon)\delta(k) \leq -x(k) - \varepsilon$$

where $M = -m = 10$, and ε is machine precision

- PWA system can be rewritten as

$$x(k+1) = 1.6\,\delta(k)x(k) - 0.8\,x(k) + u(k)$$

Example cont.

- $x(k+1) = 1.6\,\delta(k)x(k) - 0.8\,x(k) + u(k)$
- Define new variable $z(k) = \delta(k)\,x(k)$ or

$$
\begin{aligned}
z(k) &\leq M\delta(k) \\
z(k) &\geq m\delta(k) \\
z(k) &\leq x(k) - m(1-\delta(k)) \\
z(k) &\geq x(k) - M(1-\delta(k))
\end{aligned}
$$

- PWA system now becomes

$$x(k+1) = 1.6z(k) - 0.8x(k) + u(k)$$

subject to linear constraints above $\rightarrow$ MLD

Outline

Linear Complementarity (LC) systems

Definition (LC systems)

An LC system is a system of the form:

$$\begin{aligned} x(k+1) &= Ax(k) + B_1 u(k) + B_2 w(k) \\ y(k) &= Cx(k) + D_1 u(k) + D_2 w(k) \\ v(k) &= E_1 x(k) + E_2 u(k) + E_3 w(k) + e_4 \\ 0 \leq v(k) &\perp w(k) \geq 0 \end{aligned}$$

where $v(k)$, $w(k) \in \mathbb{R}_{\geq 0}$ are named the "complementarity variables".

- Applications: constrained mechanical systems, electrical networks with ideal diodes, dynamical systems with PWA relations, variable-structure systems, projected dynamical systems
- Examples: two-carts system, boost converter (*continuous*-time LC systems)

Outline

Extended Linear Complementarity (ELC) systems

Definition (ELC systems)

An ELC system is a system of the form:

$$x(k+1) = Ax(k) + B_1u(k) + B_2d(k) \tag{5.1}$$

$$y(k) = Cx(k) + D_1u(k) + D_2d(k) \tag{5.2}$$

$$E_1x(k) + E_2u(k) + E_3d(k) \leqslant e_4 \tag{5.3}$$

$$\sum_{i=1}^{p} \prod_{j\in\phi_i} \left(e_4 - E_1x(k) - E_2u(k) - E_3d(k)\right)_j = 0 \tag{5.4}$$

where $d(k) \in \mathbb{R}^r$ is an auxiliary variable.

Condition (5.4) is equivalent to

$$\prod_{j\in\phi_i} \left(e_4 - E_1x(k) - E_2u(k) - E_3d(k)\right)_j = 0 \quad \text{for each } i \in \{1,\dots,p\}$$

$\rightarrow$ p groups of linear inequalities, in each group at least one must hold as equality.

Outline

Max-Min-Plus-Scaling (MMPS) systems

- Max-min-plus-scaling expression:

$$f := x_i | \alpha | \max(f_k, f_l) | \min(f_k, f_l) | f_k + f_l | \beta f_k$$

  with α, $\beta \in \mathbb{R}$ and f_k, f_l again MMPS expressions.

- Example: $5x_1 - 3x_2 + 7 + \max(\min(2x_1, -8x_2), x_2 - 3x_3)$
- MMPS systems:

$$\begin{aligned} x(k+1) &= \mathcal{M}_x(x(k), u(k), d(k)) \\ y(k) &= \mathcal{M}_y(x(k), u(k), d(k)) \\ \mathcal{M}_\mathrm{c}(x(k), u(k), d(k)) &\leqslant c \end{aligned}$$

  with $\mathcal{M}_x$, $\mathcal{M}_y$, $\mathcal{M}_\mathrm{c}$ MMPS expressions

- $d(k)$: real-valued auxiliary variables

Applications

- discrete-event systems (also max-plus)
- traffic-signal controlled intersection
- communications capacity analysis (network calculus)
- railway networks
- manufacturing systems
- systems with soft & hard synchronization constraints
- logistic systems

Example of MMPS system

- Integrator with upper saturation:

$$
x(k+1) = \begin{cases} x(k) + u(k) & \text{if } x(k) + u(k) \leqslant 1 \\ 1 & \text{if } x(k) + u(k) \geqslant 1 \end{cases}
$$
$$
y(k) = x(k)
$$

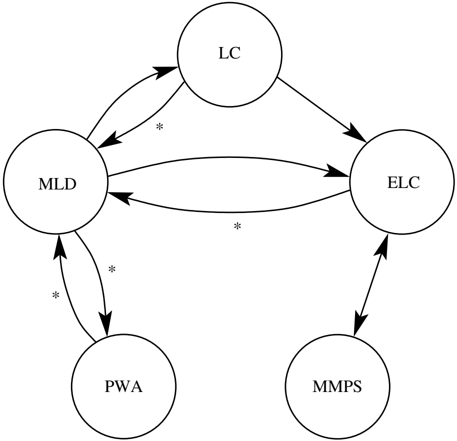
can be recast as

$$
x(k+1) = \min(x(k) + u(k), 1)
$$
$$
y(k) = x(k)
$$

Outline

Equivalences map

Equivalence between model classes $\mathcal{A}$ and $\mathcal{B}$:
for each model $\in \mathcal{A}$ there exists model $\in \mathcal{B}$ with same input/output behavior (+ vice versa) (*under some conditions).



Why different models?

- Each subclass has own advantages:
  - stability criteria for PWA
  - control and verification techniques for MLD
  - control techniques for MMPS
  - conditions of existence and uniqueness of solutions for LC

  → transfer techniques from one class to other

- It depends on the application which class is best suited

MLD $\rightarrow$ LC

Proposition

Every MLD system can be written as an LC system

- $\delta_i(k) \in \{0,1\}$ is equivalent to $0 \leq \delta_i(k) \perp 1 - \delta_i(k) \geq 0$
  $\rightarrow$ introduce auxiliary variable $p(k) = [1 \ 1 \ \dots \ 1]^{\mathrm{T}} - \delta(k)$ with

$$0 \leq \delta(k) \perp p(k) \geq 0$$

- For constraint $E_1 x(k) + E_2 u(k) + E_3 \delta(k) + E_4 z(k) \leq g_5$, introduce auxiliary variables
  $q(k) = g_5 - E_1 x(k) - E_2 u(k) - E_3 \delta(k) - E_4 z(k) \geq 0$ and $r(k) = 0$ with

$$0 \leq q(k) \perp r(k) \geq 0$$

MLD $\rightarrow$ LC

- For LC: all variables ≥ 0
  $\rightarrow$ split real-valued variable $z(k)$ in "positive" and "negative part":
  $z(k) = z^+(k) - z^-(k)$ with $z^+(k) = \max(0, z(k))$,
  $z^-(k) = \max(0, -z(k))$
  or $0 \leq z^+(k) \perp z^-(k) \geq 0$
- Results in LC system:

$$x(k+1) = Ax(k) + B_1 u(k) + [B_2 \;\; 0 \;\; B_3 \;\; -B_3]w(k)$$

$$y(k) = Cx(k) + D_1 u(k) + [D_2 \;\; 0 \;\; D_3 \;\; -D_3]w(k)$$

$$\underbrace{\begin{pmatrix} p(k) \\ q(k) \\ s(k) \\ t(k) \end{pmatrix}}_{=:v(k)} = \begin{pmatrix} e \\ g_5 - E_1 x(k) - E_2 u(k) \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} -I & 0 & 0 & 0 \\ -E_3 & 0 & -E_4 & E_4 \\ 0 & 0 & 0 & I \\ 0 & 0 & I & 0 \end{pmatrix} \underbrace{\begin{pmatrix} \delta(k) \\ r(k) \\ z^+(k) \\ z^-(k) \end{pmatrix}}_{=:w(k)}$$

$$0 \leq v(k) \perp w(k) \geq 0$$

LC $\rightarrow$ MLD

Proposition

Every LC system can be written as an MLD provided that $w(k)$ and $v(k)$ are bounded

- LC complementarity condition $0 \leq v(k) \perp w(k) \geq 0$ implies that for each i we have $v_i(k) = 0,\ w_i(k) \geq 0$ or $v_i(k) \geq 0,\ w_i(k) = 0$
- Introduce boolean vector $\delta(k)$ such that

$$
\begin{aligned}
v_i(k) = 0,\ w_i(k) \geq 0 &\leftrightarrow \delta_i(k) = 1 \\
v_i(k) \geq 0,\ w_i(k) = 0 &\leftrightarrow \delta_i(k) = 0
\end{aligned}
$$

- Can be achieved by introducing constraints

$$
\begin{aligned}
&w(k) \leq M_w \delta(k) \\
&v(k) \leq M_v([1\ 1\ \ldots\ 1]^{\mathrm{T}} - \delta(k)) \\
&w(k), v(k) \geq 0
\end{aligned}
$$

with M_w, M_v diagonal matrices containing upper bounds on $w(k), v(k)$

LC → MLD

- Note: Upper bounds usually known in practice due to physical reasons/insight.
- Finally results in MLD model

$$x(k+1) = Ax(k) + B_1 u(k) + B_2 z(k)$$
$$y(k) = Cx(k) + D_1 u(k) + D_2 z(k)$$
$$\begin{bmatrix} 0 \\ E_1 \\ 0 \\ -E_1 \end{bmatrix} x(k) + \begin{bmatrix} 0 \\ E_2 \\ 0 \\ -E_2 \end{bmatrix} u(k) + \begin{bmatrix} -M_w \\ M_v \\ 0 \\ 0 \end{bmatrix} \delta(k) + \begin{bmatrix} I \\ E_3 \\ -I \\ -E_3 \end{bmatrix} z(k) \leq \begin{bmatrix} 0 \\ M_v e - e_4 \\ 0 \\ e_4 \end{bmatrix}$$

LC $\rightarrow$ ELC

Proposition

Every LC system can be written as ELC system

- $v(k) \perp w(k)$ is equivalent to $\sum_i v_i(k) w_i(k) = 0$

PWA and MLD systems

Proposition

Well-posed PWA system can be rewritten as MLD system assuming that set of feasible states and inputs is bounded

- Cf. examples.

Proposition

Completely well-posed MLD can be rewritten as PWA

- If $\delta(k) \in \{0, 1\}^s \to 2^s$ possible combinations
- For each combination MLD constraint

$$E_1 x(k) + E_2 u(k) + E_3 \delta(k) + E_4 z(k) \leqslant g_5$$

  defines polyhedral region in $x/u/z$ space
- For each combination, $z(k)$ is linear function of $u(k)$ and $x(k)$ due to well-posedness + linearity of all constraints
- Results in linear state space model for each polyhedral region

MMPS $\equiv$ ELC (MMPS $\subseteq$ ELC)

Proposition

The classes of MMPS and ELC systems coincide.

MMPS $\subseteq$ ELC

- Basic constructors for MMPS expressions fit ELC framework:
  - Expressions of form $f = x_i$, $f = \alpha$, $f = f_k + f_l$, $f = \beta f_k$ result in linear equations
  - $f = \max(f_k, f_l) = -\min(-f_k, -f_l)$ can be rewritten as

$$f - f_k \geqslant 0, \quad f - f_l \geqslant 0, \quad (f - f_k)(f - f_l) = 0$$

$\rightarrow$ is ELC expression

- Two or more ELC systems can be combined into one large ELC

MMPS $\equiv$ ELC (ELC $\subseteq$ MMPS)

ELC $\subseteq$ MMPS

- Linear equations are MMPS expressions (albeit without max or min)
- Complementarity condition can be rewritten as

$$\forall i, \exists j \in \phi_i \text{ such that } \underbrace{\left(e_4 - E_1 x(k) - E_2 u(k) - E_3 d(k)\right)_j}_{\geq 0} = 0$$

So

$$\min_{j \in \phi_i} \left(e_4 - E_1 x(k) - E_2 u(k) - E_3 d(k)\right)_j = 0 \quad \text{for each } i$$

MLD $\rightarrow$ ELC

Proposition

Every MLD system can be rewritten as an ELC system.

- Condition $\delta_i(k) \in \{0, 1\}$ is equivalent to ELC conditions

$$
\begin{aligned}
-\delta_i(k) &\leqslant 0 \\
\delta_i(k) &\leqslant 1 \\
\delta_i(k)(1 - \delta_i(k)) &= 0
\end{aligned}
$$

- Note: condition $\delta_i(k) \in \{0, 1\}$ also equivalent to MMPS constraints

$$\max(-\delta_i(k), \delta_i(k) - 1) = 0$$

or

$$\min(\delta_i(k), 1 - \delta_i(k)) = 0$$

ELC $\to$ MLD

Proposition

Every ELC system can be written as an MLD system, provided that $e_4 - E_1x(k) - E_2u(k) - E_3d(k)$ *is bounded.*

- Introduce conditions:

$$(e_4)_j - (E_1x(k) + E_2u(k) + E_3d(k))_j \leq M_j\delta_j(k) \quad \text{for each } j \in \phi_i$$

$$\sum_{j\in\phi_i} \delta_j(k) \leq \#\phi_i - 1$$

  with $\delta_j(k) \in \{0,1\}$ auxiliary variables,
  and M_j upper bound for $(e_4 - E_1x(k) - E_2u(k) - E_3d(k))_j$

- By last condition at least one $\delta_h(k)$ is zero for some $h \in \phi_i$
  $\to$ 1st inequality and ELC inequality $(e_4)_j - (E_1x(k) + E_2u(k) + E_3d(k))_j \geq 0$ degenerate to equality condition for $j = h$

- Hence, (nonlinear) ELC complementarity condition can be replaced by above (linear) equations $\to$ MLD system

Example

- Consider

$$x(k+1) = \begin{cases} 0.8x(k) + u(k) & \text{if } x(k) \geq 0 \\ -0.8x(k) + u(k) & \text{if } x(k) < 0 \end{cases}$$

  with $m \leq x(k) \leq M$

- MLD:

$$\begin{aligned} x(k+1) &= -0.8x(k) + u(k) + 1.6z(k) \\ -m\delta(k) &\leq x(k) - m & x(k) &\leq (M+\varepsilon)\delta(k) - \varepsilon \\ z(k) &\leq M\delta(k) & z(k) &\geq m\delta(k) \\ z(k) &\leq x(k) - m(1-\delta(k)) & z(k) &\geq x(k) - M(1-\delta(k)) \end{aligned}$$

  with $\delta(k) \in \{0, 1\}$

- MMPS:

$$x(k+1) = -0.8x(k) + 1.6\max(0, x(k)) + u(k)$$

Example

- Consider

$$x(k+1) = \begin{cases} 0.8x(k) + u(k) & \text{if } x(k) \geq 0 \\ -0.8x(k) + u(k) & \text{if } x(k) < 0 \end{cases}$$

- LC:

$$\begin{aligned} x(k+1) &= -0.8x(k) + u(k) + 1.6z(k) \\ 0 \leq w(k) &= -x(k) + z(k) \perp z(k) \geq 0 \end{aligned}$$

- ELC:

$$x(k+1) = -0.8x(k) + u(k) + 1.6d(k)$$
$$-d(k) \leq 0, \qquad x(k) - d(k) \leq 0, \qquad \big(x(k) - d(k)\big)\big(-d(k)\big) = 0$$

Outline

Timed Automata (TA)

- Timed automata involve simple continuous dynamics:
  - all differential equations of form $\dot{x} = 1$
  - all invariants, guards, etc. involve comparison of real-valued states with constants (e.g., $x = 1$, $x < 2$, $x \geq 0$, etc.)
- Timed automata are limited for modeling physical systems
- However, very well suited for encoding timing constraints such as "event A must take place at least 2 seconds after event B and not more than 5 seconds before event C"
- Applications: schedulling, multimedia, Internet, audio protocol verification

Rectangular sets

- A subset of $\mathbb{R}^n$ set is called rectangular if it can be written as finite boolean combination of constraints of form

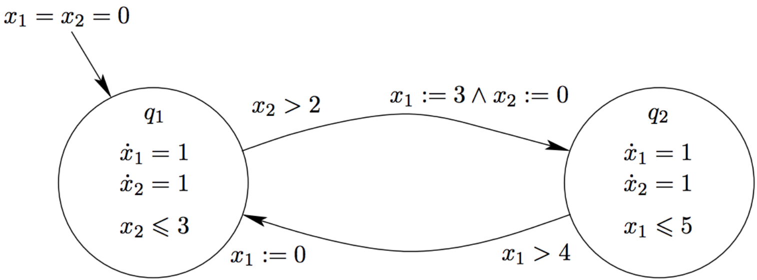
$$x_i \leq a, \quad x_i < b, \quad x_i = c, \quad x_i \geq d, \quad x_i > e$$

- Rectangular sets are "rectangles" or "boxes" in $\mathbb{R}^n$ whose sides are aligned with the axes, or unions of such rectangles/boxes
- Examples:
  - $\{(x_1, x_2) \mid (x_1 \geq 0) \wedge (x_1 \leq 2) \wedge (x_2 \geq 1) \wedge (x_2 \leq 2)\}$
  - $\{(x_1, x_2) \mid ((x_1 \geq 0) \wedge (x_2 = 0)) \vee ((x_1 = 0) \wedge (x_2 \geq 0))\}$
  - empty set (e.g., $\emptyset = \{(x_1, x_2) \mid (x_1 > 1) \wedge (x_1 \leq 0))\}$
- However, set $\{(x_1, x_2) \mid x_1 = 2x_2\}$ is not rectangular

Timed automaton

- A *timed automaton* is hybrid automaton with following characteristics:
  - automaton involves differential equations of form $\dot{x}_i = 1$
    continuous variables governed by this differential equation are called "clocks" or "timers"
  - sets involved in definition of initial states, guards, and invariants are rectangular sets
  - reset maps involve either rectangular set, or may leave certain states unchanged

Example of TA



Timed automaton

Definition (Timed Automaton)

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

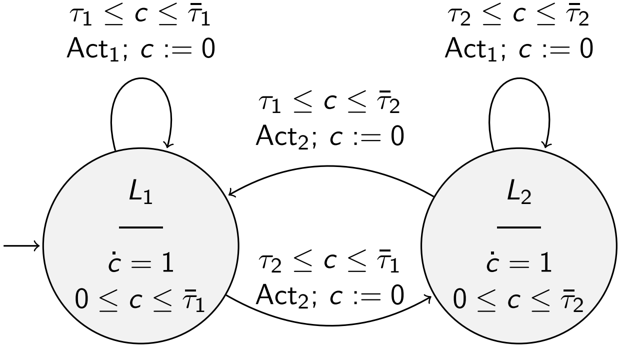
- L is the set of finitely many locations (or nodes);
- $\ell_0 \in L$ is the initial location;
- Act is the set of finitely many labelled actions;
- C is the set of finitely many real-valued clocks;
- $E \subseteq L \times \mathcal{B}(C) \times \text{Act} \times 2^C \times L$ is the set of edges;
-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

where $\mathcal{B}(C)$ denotes the set of clock constraints: $c \sim a$, $a \in \mathbb{Q}, \sim \in \{<, >, \leq, \geq\}$.

Reference:

Al04 Alur, Rajeev, and David L. Dill. "A theory of timed automata." Theoretical computer science 126.2 (1994): 183-235.

Example of labelled TA



Notes on TA

- Formally Timed Automata contain a set of *accepting states* in order to delimit an *accepted language*.
- The (language) emptiness problem for timed automata is decidable. It is PSPACE-complete.
- Safety and reachability control are decidable and are EXPTIME-complete.
- A key construction to prove the last two decidability results and synthesize winning strategies is the construction of a finite abstraction: the *Region Abstraction*.
- Extensions to include prices and solve games (*Timed Priced Game Automata*) for which approaches exist to synthesise (almost) optimal strategies.

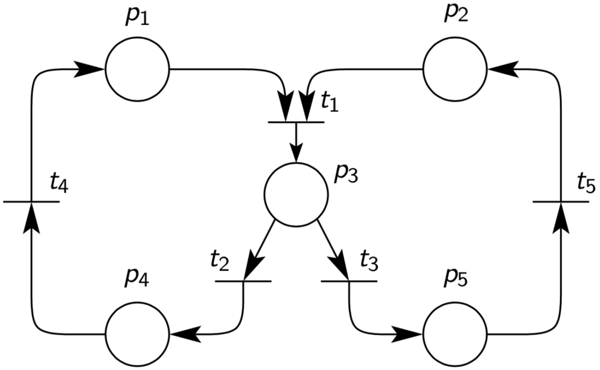
Reference: For a nice overview of the field of TA verification and synthesis look at:

Bou18 Bouyer, Patricia, et al. "Model checking real-time systems." Handbook of Model Checking. Springer, Cham, 2018. 1001-1046. z

Outline

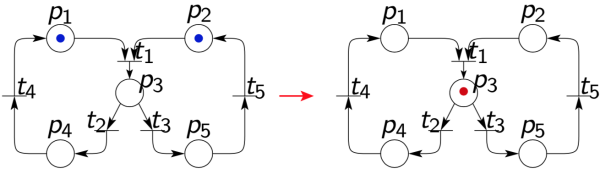
Petri nets

- Graphical representation: bipartite directed graph
  - places (circles) $\rightarrow$ activities
  - transitions (bars) $\rightarrow$ events, actions



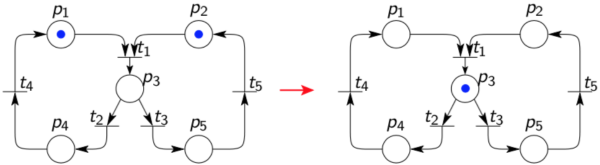
Petri nets execution

- marking $\rightarrow$ tokens are assigned to places
- execution of Petri net:
  - transition enabled if all input places (${}^\bullet t$) contain at least 1 token
  - enabled transition can fire:
    - one token is removed from each input place (${}^\bullet t$)
    - one token is deposited in each output place ($t^\bullet$)

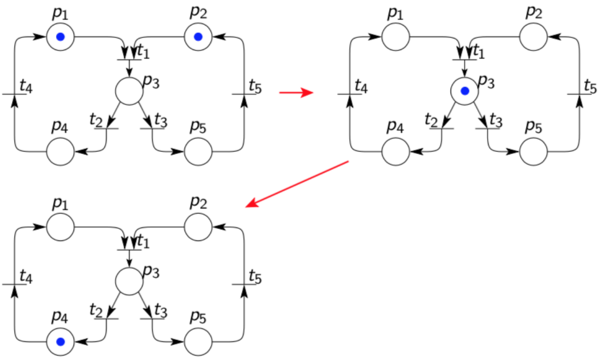


- synchronization & choice

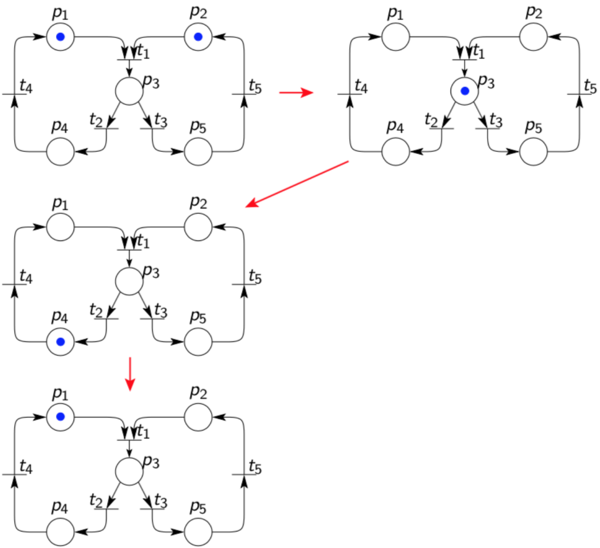
Petri Nets execution



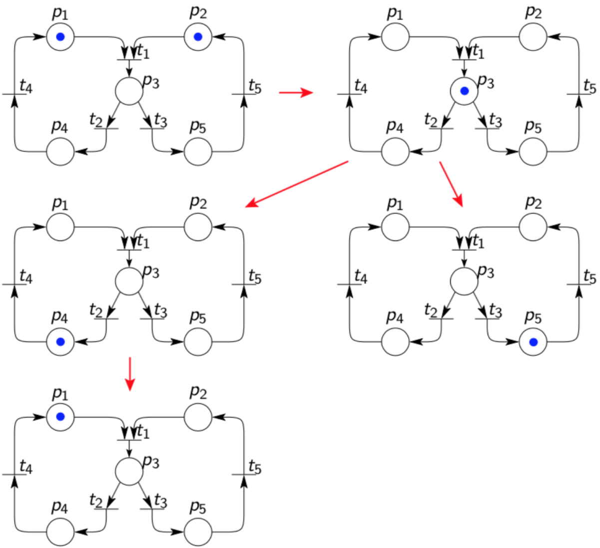
Petri Nets execution



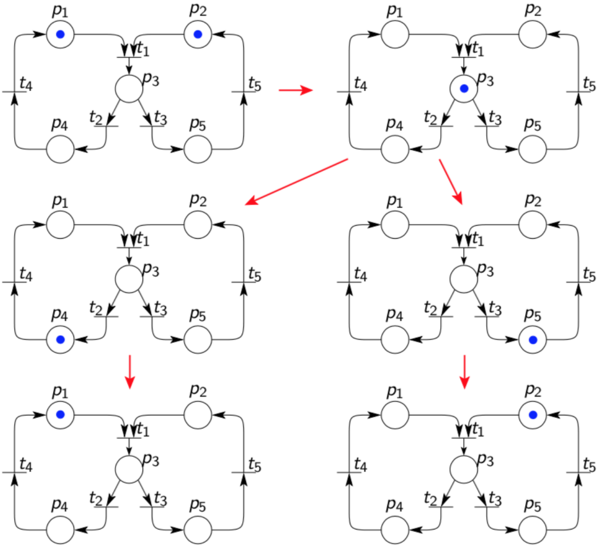
Petri Nets execution



Petri Nets execution



Petri Nets execution



Timed Petri Nets

- Untimed Petri net describes order in which events can occur, but no timing
- Timed Petri $\rightarrow$ timing, transition should be executed within certain time interval after it becomes enabled
  - discrete state variables (markings, $m_\theta(p)$)
  - continuous state variables (arrival times, $M_\theta(p)$)
- $M_\theta(p) := \{\theta_1, \ldots, \theta_{m_\theta(p)}\}$ are the arrival times $\theta_1 \leq \theta_2 \leq \ldots \leq \theta_{m_\theta(p)}$ of the $m_\theta(p)$ tokens in place p.
- For each transition t we define the interval $[L(t), U(t)]$

Timed Petri Nets execution

- Transition t becomes enabled at

$$\max_{p \in {}^\bullet t} \min M_\theta(p)$$

- Then transition t may fire at some time

$$\theta \in [\max_{p \in {}^\bullet t} \min M_\theta(p) + L(t), \max_{p \in {}^\bullet t} \min M_\theta(p) + U(t)]$$

  provided t is enabled during whole interval
- If enabling condition is still valid at final time of firing interval, then transition is forced to fire
- Many techniques for untimed Petri nets can be extended to timed Petri nets
- However, many problems are undecidable or NP-hard

Outline

Summary

- Trade-off: modeling power $\leftrightarrow$ decision power
  $\rightarrow$ focus on tractable classes of hybrid systems
- Piecewise affine systems (PWA)
- Mixed Logical Dynamical systems (MLD)
- Linear Complementarity systems (LC)
- Extended Linear Complementarity systems (ELC)
- Max-Min-Plus-Scaling systems (MMPS)
- Equivalence of MLD, LC, ELC, PWA, and MMPS systems
- Timed automata
- Timed Petri nets