Short Course on Hybrid Systems

Alberto Bemporad

Dip. di Ingegneria dell'Informazione
Università degli Studi di Siena

Automatic Control Laboratory, ETH Zürich

bemporad@dii.unisi.it
http://www.dii.unisi.it/~bemporad

University of Pisa, December 11-12, 2001

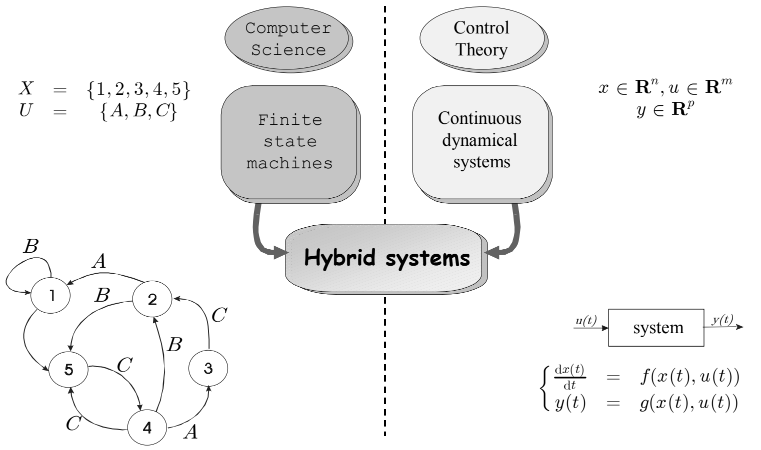
Course Summary

1. What's a "hybrid system" ?
2. Models for hybrid systems
3. Control of hybrid systems
4. Piecewise linear optimal control
5. Examples (synthesis of a cruise controller, …)
6. Safety analysis of hybrid systems

Hybrid Systems

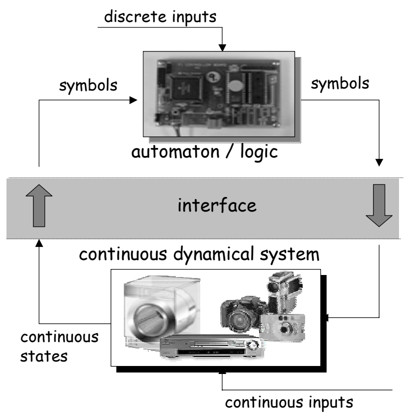
$$X = \{1,2,3,4,5\}$$
$$U = \{A,B,C\}$$

$$x \in \mathbf{R}^n, u \in \mathbf{R}^m$$
$$y \in \mathbf{R}^p$$



$$\begin{cases} \frac{dx(t)}{dt} = f(x(t),u(t)) \\ y(t) = g(x(t),u(t)) \end{cases}$$

Motivation: Embedded Systems

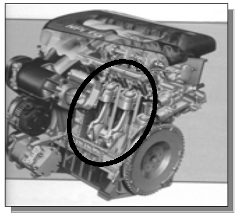


- Consumer electronics
- Home appliances
- Office automation
- Automobiles
- Industrial plants
- ...

Motivation: “Intrisically Hybrid”

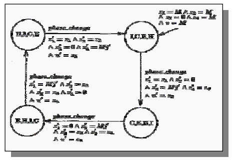
• Transmission

Discrete command (R,N,1,2,3,4,5) + Continuous dynamical variables (velocities, torques)



• Four-stroke engines

Automaton, dependent on power train motion



Key Requirements for Hybrid Models

• **Descriptive** enough to capture the behavior of the system

– continuous dynamics (physical laws)
– logic components (switches, automata, software code)
– interconnection between logic and dynamics

• **Simple** enough for solving *analysis* and *synthesis* problems

$$x' = Ax + Bu \qquad y = Cx + Du$$

linear systems

$$x' = f(t, x, u) \qquad y = g(t, x, u)$$

nonlinear systems

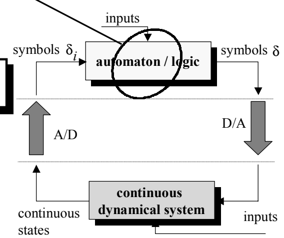
?

linear hybrid systems

Mixed Integer Programming and Hybrid Systems

$X_1 \vee X_2$	$\delta_1 + \delta_2 \geq 1$
$X_1 \wedge X_2$	$\delta_1 + \delta_2 \geq 2$
$\neg X_1$	$\delta_1 \leq 0$
$X_1 \Rightarrow X_2$	$\delta_1 - \delta_2 \leq 0$
$X_1 \Leftrightarrow X_2$	$\delta_1 - \delta_2 = 0$

any logic proposition	$A\delta \leq B$, $\delta \in \{0,1\}^n$



Transformation into Linear Integer Inequalities

0. Given a Boolean statement $F(X_1, X_2, \ldots, X_n)$

1. Convert to Conjunctive Normal Form (CNF):

$$\bigwedge_{j=1}^{m} \left(\bigvee_{i \in P_j} X_i \bigvee_{i \in N_j} \bar{X}_i \right)$$

2. Transform into inequalities:

$$\begin{aligned} 1 &\leq \sum_{i \in P_1} \delta_i + \sum_{i \in N_1} (1 - \delta_i) \\ &\vdots \\ 1 &\leq \sum_{i \in P_m} \delta_i + \sum_{i \in N_m} (1 - \delta_i) \end{aligned}$$

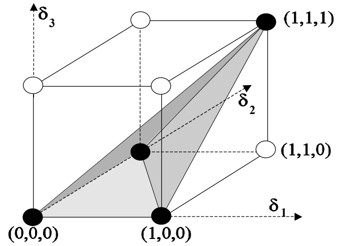
$$A\delta \leq B, \quad \delta \in \{0, 1\}$$

➡ Every logic proposition can be translated into linear integer inequalities

Truth Tables ➡ Linear Integer Inequalities

Example: logic "AND"

δ_1	δ_2	δ_3
0	0	0
0	1	0
1	0	0
1	1	1



Key idea: White points cannot be in the hull of black points

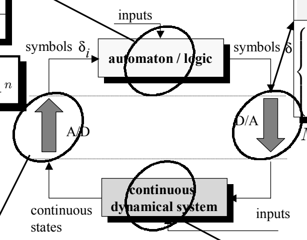
$$\mathrm{conv}\left(\begin{bmatrix}0\\0\\0\end{bmatrix},\begin{bmatrix}0\\1\\0\end{bmatrix},\begin{bmatrix}1\\0\\0\end{bmatrix},\begin{bmatrix}1\\1\\1\end{bmatrix}\right)=\left\{\delta:\begin{array}{r}-\delta_1+\delta_3\leq 0\\-\delta_2+\delta_3\leq 0\\\delta_1+\delta_2-\delta_3\leq 1\end{array}\right\}$$

Algorithms to compute convex hull: `cdd, lrs, qhull, chD, Hull, Porto`

Mixed Integer Programming and Hybrid Systems

$X_1 \vee X_2$	$\delta_1+\delta_2\geq 1$
$X_1 \wedge X_2$	$\delta_1+\delta_2\geq 2$
$\neg X_1$	$\delta_1\leq 0$
$X_1 \Rightarrow X_2$	$\delta_1-\delta_2\leq 0$
$X_1 \Leftrightarrow X_2$	$\delta_1-\delta_2=0$

any logic proposition	$A\delta \leq B$, $\delta \in \{0,1\}^n$



IF X THEN $z = a_1'x + b_1$
ELSE $z = a_2'x + b_2$

$$\begin{cases}(-M_2-M_1)\delta+z\leq a_2'x+b_2\\(-M_1-M_2)\delta-z\leq -a_2'x-b_2\\(-M_1-M_2)(1-\delta)+z\leq a_1'x+b_1\\(-M_2-M_1)(1-\delta)-z\leq -a_1'x-b_1\end{cases}$$

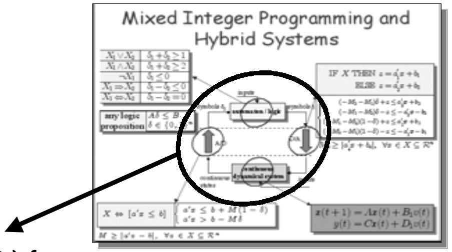
$M_i \geq |a_i'x+b_i|,\ \forall x \in X \subseteq \mathcal{R}^n$

$$X \Leftrightarrow [a'x \leq b] \qquad \begin{cases}a'x \leq b + M(1-\delta)\\a'x > b - M\delta\end{cases}$$

$M \geq |a'x - b|,\ \forall x \in X \subseteq \mathcal{R}^n$

$$\begin{aligned}x(t+1) &= Ax(t)+B_1v(t)\\y(t) &= Cx(t)+D_1v(t)\end{aligned}$$

Mixed Logical Dynamical Systems



Mixed Logical Dynamical (MLD) form

$$\begin{aligned}x(t+1) &= Ax(t)+B_1u(t)+B_2\delta(t)+B_3z(t)\\y(t) &= Cx(t)+D_1u(t)+D_2\delta(t)+D_3z(t)\\E_2\delta(t)+E_3z(t) &\leq E_4x(t)+E_1u(t)+E_5\end{aligned}$$

(Bemporad, Morari, *Automatica*, 1999)

$x,y,u=\begin{bmatrix}\star_c\\\star_\ell\end{bmatrix}, \star_c\in\mathbf{R}^{n_c}, \star_\ell\in\{0,1\}^{n_\ell},\ z\in\mathbf{R}^{r_c},\ \delta\in\{0,1\}^{r_\ell}$

Well-posedness

Are state and output trajectories defined ? Uniquely defined ? Persistently defined ?

- MLD well-posedness :

$$\begin{aligned}x(t+1) &= Ax(t)+B_1u(t)+B_2\delta(t)+B_3z(t)\\y(t) &= Cx(t)+D_1u(t)+D_2\delta(t)+D_3z(t)\\E_2\delta(t)+E_3z(t) &\leq E_1u(t)+E_4x(t)+E_5\end{aligned}$$

$$\begin{aligned}\delta(t) &= F(x(t),u(t))\\z(t) &= G(x(t),u(t))\end{aligned}$$

$$\begin{aligned}&\{x(t),u(t)\}\rightarrow\{x(t+1)\}\\&\{x(t),u(t)\}\rightarrow\{y(t)\}\end{aligned}$$

are single valued

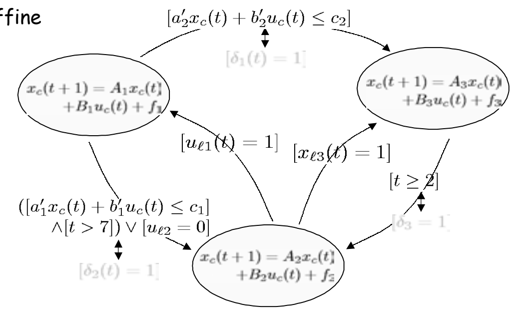
Definition 1 *Let $\Omega \subseteq \mathbb{R}^n \times \mathbb{R}^m$ be a set of input+state pairs. A hybrid MLD system is called* well-posed *on Ω, if for all pairs $(x(t),u(t)) \in \Omega$ there exists a solution $x(t+1), y(t), \delta(t), z(t)$ and moreover, $x(t+1), y(t)$ are uniquely determined.*

Numerical test based on mixed-integer programming available

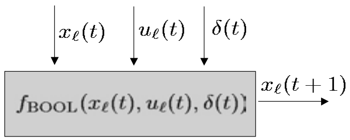
(Bemporad, Morari, *Automatica,* 1999)

MLD Hybrid Dynamics

- Continuous dynamics is affine and change according to:
  - Exogenous logic inputs
  - Threshold conditions
  - Time conditions
  - Any logic combination of the former

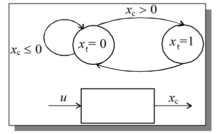


- Discrete dynamics change according to a Boolean fnc of
  - Previous logic states
  - Exogenous logic inputs
  - Threshold conditions



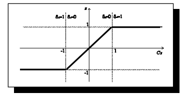
Examples

- Automata driven by dynamical systems

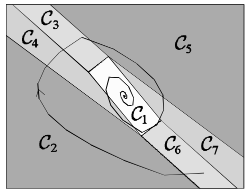


- Discrete inputs

- Nonlinearities (by piecewise linear approx.)



- Piecewise affine (PWA) systems



state+input space

- Polyhedral partition of state+input space

$$\mathcal{X}_i = \left\{ \begin{bmatrix} x \\ u \end{bmatrix} : \; H_i \begin{bmatrix} x \\ u \end{bmatrix} \le K_i \right\}, \quad i = 1, \ldots, s$$

- Linear affine dynamics in each region

$$x(t+1) = A_{i(t)} x(t) + B_{i(t)} u(t) + f_{i(t)}$$
$$\text{if} \quad \begin{bmatrix} x(t) \\ u(t) \end{bmatrix} \in \mathcal{X}_{i(t)}$$

A Simple Example

- System: $x(t+1) = \begin{cases} 0.8x(t) + u(t) & \text{if } x(t) \ge 0 \\ -0.8x(t) + u(t) & \text{if } x(t) < 0 \end{cases}$

$$-10 \le x(t) \le 10, -1 \le u(t) \le 1$$

- Associate $[\delta(t) = 1] \leftrightarrow [x(t) \ge 0]$ and transform

$\longrightarrow$ $\quad -m\delta(t) \le x(t) - m$ $\quad$ $M = -m = 10$

$\quad -(M + \epsilon)\delta(t) \le -x(t) - \epsilon$ $\quad$ $\epsilon > 0$ small

- Then $x(t+1) = 1.6\delta(t)x(t) - 0.8x(t) + u(t)$

$z(t) = \delta(t)x(t) \longrightarrow$

$$\begin{aligned} z(t) &\le M\delta(t) \\ z(t) &\ge m\delta(t) \\ z(t) &\le x(t) - m(1 - \delta(t)) \\ z(t) &\ge x(t) - M(1 - \delta(t)) \end{aligned}$$

- Rewrite as linear equation

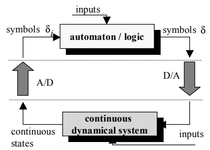
$\longrightarrow$ $\quad x(t+1) = 1.6z(t) - 0.8x(t) + u(t)$

HYSDEL

(HYbrid Systems DEscription Language)

- Describe *hybrid systems:*

- Automata
- Logic
- Lin. Dynamics
- Interfaces
- Constraints



(Torrisi, Bemporad, Mignone, 2000)

- Automatically generate MLD models in Matlab

- MLD model is not unique in terms of the number of auxiliary variables $\longrightarrow$ optimize model (minimize # binary variables !)

http://control.ethz.ch/~hybrid/hysdel

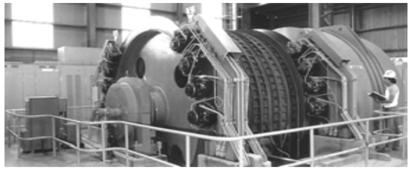
Example 1: AD section

$$[s = \mathsf{T}] \leftrightarrow [h \geq h_{\max}]$$

```
SYSTEM tank {
   INTERFACE {
      STATE {
         REAL h; }
      INPUT {
         REAL Q; }
      PARAMETER {
         REAL hmax = 0.3;
         REAL k    = 1;
         REAL e    = 1e-6; }
   } /* end interface */

   IMPLEMENTATION {
      AUX {
         BOOL s; }
      AD {
         s = hmax - h <= 0 [hmax, -hmax, e]; }
      CONTINUOUS {
         h = h + k * Q; }
   } /* end implementation */
} /* end system */
```

Example 2: DA section



Nonlinear amplification unit

$$u_{comp} = \begin{cases} u & (u < u_t) \\ 2.3u - 1.3u_t & (u \geq u_t) \end{cases}$$

```
SYSTEM motor {
   INTERFACE {
      STATE {
         REAL ucomp; }
      INPUT {
         REAL u; }
      PARAMETER {
         REAL ut   = 1;
         REAL umax = 10;
         REAL e    = 1e-6;}
   } /* end interface */
```

```
IMPLEMENTATION {
      AUX {
         REAL unl;
         BOOL th; }
      AD {
         th = ut - u <= 0 [ut, ut-umax, e]; }
      DA {
         unl = { IF th THEN 2.3*u - 1.3*ut
                     [umax-1.3*ut, -1.3*ut,  0]
                     ELSE u
                     [          0,       0,  0]}; }
      CONTINUOUS {
         ucomp = unl; }
   } /* end implementation */
} /* end system */
```

Example 3: LOGIC section

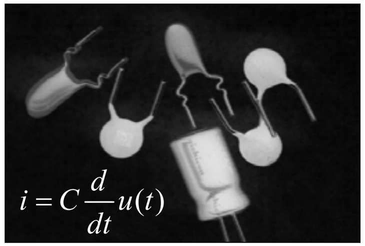
$$u_{brake} = u_{alarm} \wedge (\neg s_{tunnel} \vee \neg s_{fire})$$

$$s_{fire} \rightarrow u_{alarm}$$

```
SYSTEM train {
   INTERFACE {
      STATE {
         BOOL brake; }
      INPUT {
         BOOL alarm, tunnel, fire; }
   } /* end interface */

   IMPLEMENTATION {
      AUX {
         BOOL decision; }
      LOGIC {
         decision =
            alarm & (~tunnel | ~fire); }
      AUTOMATA {
         brake = decision; }
      MUST {
         fire -> alarm; }
   } /* end implementation */
} /* end system */
```

Example 4: CONTINUOUS section



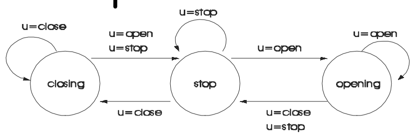
Apply forward difference rule:

$$u(k+1) = u(k) + \frac{T}{C} i(k)$$

```
SYSTEM capacitorD {
   INTERFACE {
      STATE {
         REAL u; }
      PARAMETER {
         REAL R = 1e4;
         REAL C = 1e-4;
         REAL T = 1e-1; }
   } /* end interface */

   IMPLEMENTATION {
      CONTINUOUS {
        u = u - T/C/R*i;  }
   } /* end implementation */
} /* end system */
```

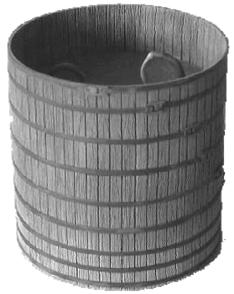
Example 5: AUTOMATA section



```
SYSTEM outflow {
   INTERFACE {
      STATE {
         BOOL closing, stop, opening; }
      INPUT {
         BOOL uclose, uopen, ustop; }
   } /* end of interface */

   IMPLEMENTATION {
      AUTOMATA {
         closing = (uclose & closing) | (uclose & stop);
         stop    = ustop | (uopen & closing) | (uclose & opening);
         opening = (uopen & stop) | (uopen & opening);  }
      MUST {
         ~(uclose & uopen);
         ~(uclose & ustop);
         ~(uopen  & ustop); }
   } /* end implementation */
} /* end system */
```

Example 6: MUST section

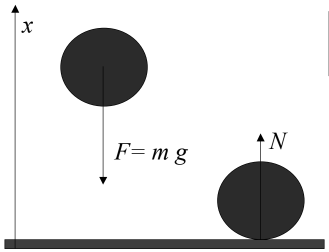


$0 \le h \le h_{\max}$

```
SYSTEM watertank {
   INTERFACE {
      STATE {
         REAL h; }
      INPUT {
         REAL Q; }
      PARAMETER {
         REAL hmax = 0.3;
         REAL k    = 1; }
   } /* end interface */

   IMPLEMENTATION {
      CONTINUOUS {
         h = h + k*Q; }
      MUST {
          h - hmax <= 0;
         -h        <= 0; }
   } /* end implementation */
} /* end system */
```

Example: Bouncing Ball



$$\ddot{x} = -g$$
$$x \le 0 \Rightarrow \dot{x}(t^+) = -(1-\alpha)\dot{x}(t^-)$$

$$\alpha \in [0,1]$$

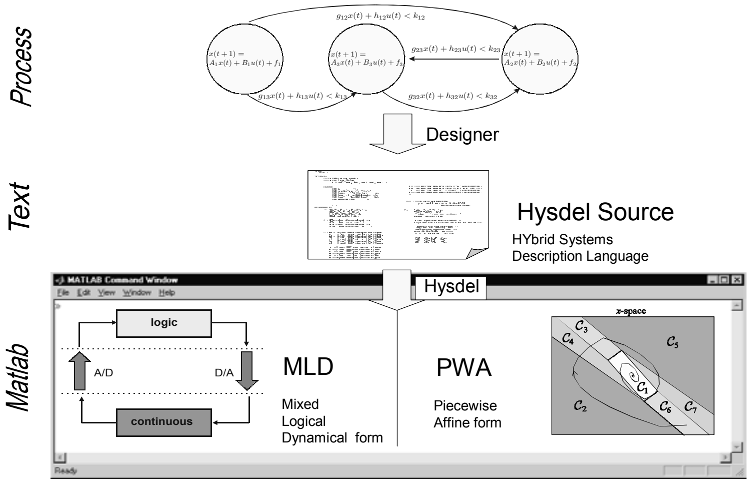
How to model this system in MLD form?

HYSDEL - Bouncing Ball

```
SYSTEM ball {
INTERFACE {
/* Description of variables and constants */
        STATE { REAL height;
                REAL velocity;
        }
        PARAMETER {
                REAL g=9.8;
                REAL dissipation=.4;  /* 0=elastic, 1=completely anelastic */
                REAL hmax=10;
                REAL hmin=-10;
                REAL vmax=100;
                REAL vmin=-100;
                REAL e=1e-6;
                REAL Ts=.05;
        }
}
IMPLEMENTATION {
        AUX {   REAL z1;
                REAL z2;
                BOOL negative;
        }
        AD {
                negative = height <= 0 [hmax,hmin,e];
        }
        DA {    z1 = {  IF negative THEN height-Ts*velocity [hmax,hmin,e]
                        ELSE height+Ts*velocity-Ts*Ts*g [hmax,hmin,e]   };
                z2 = {  IF negative THEN -(1-dissipation)*velocity [vmax,vmin,e]
                        ELSE velocity-Ts*g [vmax,vmin,e]                };
        }
        CONTINUOUS {
                height = z1;
                velocity=z2;
        }

}
}
```

Modeling Flow



System Theory for Hybrid Systems

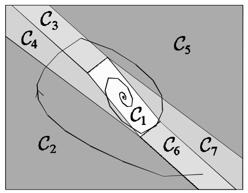
- Analysis
  - Realization & Transformation
  - Well-posedness
  - Stability
  - Reachability (=Verification)
  - Observability
- Synthesis
  - Control
  - State estimation
  - Identification
  - Modeling language

Realization and Transformation (State-Space Hybrid Models)

Existing Hybrid Models

- Piecewise affine (PWA) systems (Sontag, 1981, 1996)

state+input space



- Polyhedral partition of state+input space

$$\mathcal{X}_i = \{ \begin{bmatrix} x \\ u \end{bmatrix} : \; H_i \begin{bmatrix} x \\ u \end{bmatrix} \leq K_i \}, \;\; i = 1, \ldots, s$$

- Affine dynamics in each region

$$x(t+1) = A_{i(t)} x(t) + B_{i(t)} u(t) + f_{i(t)} \quad \text{if} \begin{bmatrix} x(t) \\ u(t) \end{bmatrix} \in \mathcal{X}_{i(t)}$$

- Can approximate nonlinear dynamics arbitrarily well

Existing Hybrid Models

- Linear complementarity (LC) systems (Heemels, 1999)

$$\begin{aligned} x(t+1) &= Ax(t) + B_1u(t) + B_2w(t) \\ y(t) &= Cx(t) + D_1u(t) + D_2w(t) \\ v(t) &= E_1x(t) + E_2u(t) + E_3w(t) + e_4 \\ 0 &\le v(t) \perp w(t) \ge 0 \end{aligned}$$

Ex: mechanical systems
circuits with diodes etc.

- Extended linear complementarity (ELC) systems (De Schutter, De Moor, 2000)

Generalization of LC systems

- Min-max-plus-scaling (MMPS) systems (De Schutter, Van Den Boom, 2000)

$$\begin{aligned} x(t+1) &= M_x(x(t), u(t), d(t)) \\ y(t) &= M_y(x(t), u(t), d(t)) \\ 0 &\ge M_c(x(t), u(t), d(t)) \end{aligned}$$

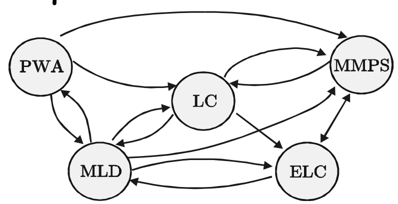
MMPS function: defined by the grammar

$M := x_i | \alpha | \max(M_1, M_2) | \min(M_1, M_2) | M_1 + M_2 | \beta M_1$

Example: $x(t+1) = 2\max(x(t), 0) + \min(-\frac{1}{2}u(t), 1)$

Used for modeling discrete-event systems (t=event counter)

Equivalence Results



Theorem 1 *All the above five classes of discrete-time hybrid models are equivalent (possibly under additional assumptions, like boundedness of input and state variables)*

(Heemels, De Schutter, Bemporad, *Automatica,* 2001 + CDC2001)

Theoretical properties and analysis/synthesis tools can be transferred from one class to another !

MLD and PWA Systems

Theorem 1 *MLD systems and PWA systems are equivalent.*

(Bemporad, Ferrari-Trecate, Morari, *IEEE TAC,2000*)

- MLD:

$$\begin{aligned} x(t+1) &= Ax(t) + B_1u(t) + B_2\delta(t) + B_3z(t) \\ y(t) &= Cx(t) + D_1u(t) + D_2\delta(t) + D_3z(t) \\ E_2\delta(t) + E_3z(t) &\le E_1u(t) + E_4x(t) + E_5 \end{aligned}$$

- By well-posedness hypothesis on $z(t), \delta(t)$ + linearity of MLD constraints

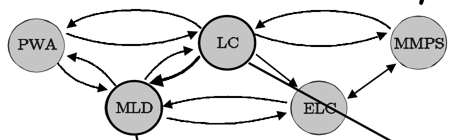
$$z = K_4^i x + K_1^i u + K_5^i \qquad \forall (x,u) : F(x,u) = \delta^i$$

→ PWA form

$$\begin{aligned} x(t+1) &= A^i x(t) + B^i u(t) + f^i \\ y(t) &= C^i x(t) + D^i u(t) + g^i \end{aligned} \qquad F^i x(t) + G^i u(t) \le h^i$$

- Confirms (Sontag, 1996): PWL systems and hybrid systems are equivalent

MLD and LC Systems



Theorem 1 *Every LC system can be written as an MLD system, provided that the variables $w(k)$ and $v(k)$ are (componentwise) bounded.*

Proof:

$$\begin{aligned} x(t+1) &= Ax(t) + B_1u(t) + B_2\delta(t) + B_3z(t) \\ y(t) &= Cx(t) + D_1u(t) + D_2\delta(t) + D_3z(t) \\ E_2\delta(t) + E_3z(t) &\le E_4x(t) + E_1u(t) + E_5 \end{aligned}$$

$$\begin{aligned} x(t+1) &= Ax(t) + B_1u(t) + B_2w(t) \\ y(t) &= Cx(t) + D_1u(t) + D_2w(t) \\ v(t) &= E_1x(t) + E_2u(t) + E_3w(t) + e_4 \\ 0 &\le v(t) \perp w(t) \ge 0 \end{aligned}$$

For each complementarity pair $v_i(t), w_i(t)$ introduce a binary variable $\delta_i(t) \in \{0,1\}$

$[\delta_i(t) = 1] \to [v_i(t) = 0,\ w_i(t) \ge 0]$

$[\delta_i(t) = 0] \to [v_i(t) \ge 0,\ w_i(t) = 0]$

⇒

$$\begin{aligned} w_i(t) &\le M\delta_i(t) \\ v_i(t) &\le M(1-\delta_i(t)) \\ w_i(t) &\ge 0 \\ v_i(t) &\ge 0 \end{aligned}$$

Set $z_i(t) = w_i(t)$ and substitute $v(t) = E_1x(t) + E_2u(t) + E_3w(t) + e_4$ ■