Hybrid Models for Analysis and Control Design

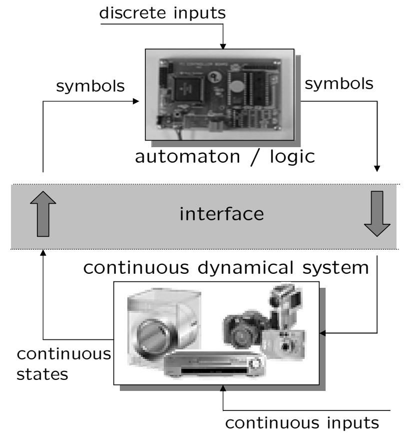
Alberto Bemporad

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

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

Università degli Studi di Siena
Facoltà di Ingegneria

Motivation: Embedded Systems



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

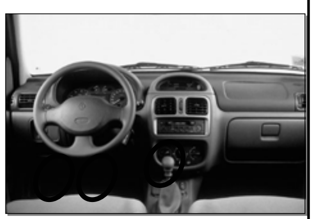
Motivating Problem #1

GOAL:

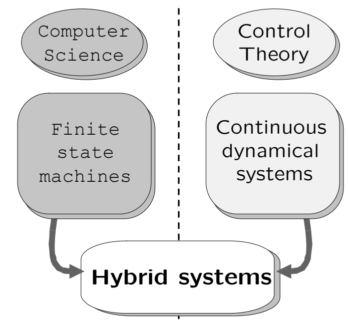
command gear ratio, gas pedal, and brakes to track a desired speed and minimize consumptions

CHALLENGES:

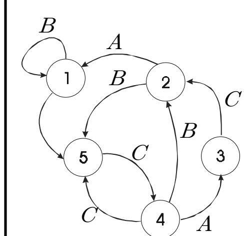
- continuous **and** discrete inputs
- dynamics depends on gear
- nonlinear torque/speed maps



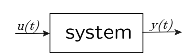
Hybrid Systems



$\mathrm{x} \in \{1,2,3,4,5\}$
$\mathrm{u} \in \{A,B,C\}$



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



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

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

$$\begin{cases} x' = Ax + Bu \\ y = Cx + Du \end{cases}$$

linear systems

$$\begin{cases} x' = f(x,u,t) \\ y = g(x,u,t) \end{cases}$$

nonlinear systems

?

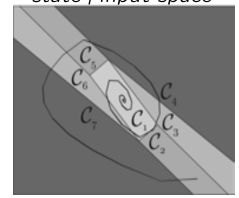
linear hybrid systems

Piecewise Affine Systems

(Sontag 1981)

$$\begin{aligned} x'(k) &= A_{i(k)}x(k) + B_{i(k)}u(k) + f_{i(k)} \\ y(k) &= C_{i(k)}x(k) + D_{i(k)}u(k) + g_{i(k)} \\ i(k) \ \text{s.t.} \ & H_{i(k)}x(k) + J_{i(k)}u(k) \leq K_{i(k)} \end{aligned}$$

state+input space



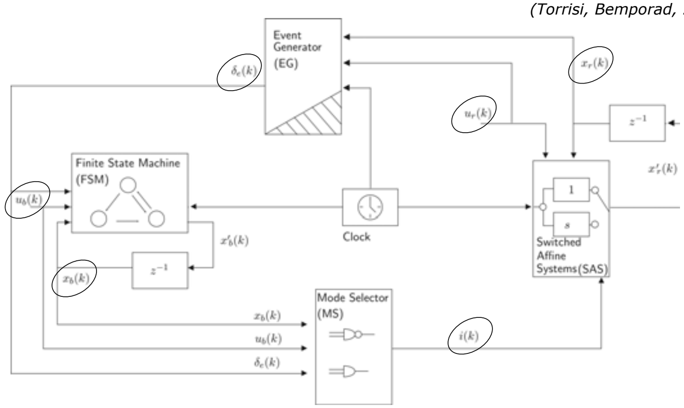
$x \in \mathcal{X} \subseteq \mathbb{R}^n$, $u \in \mathcal{U} \subseteq \mathbb{R}^m$, $y \in \mathcal{Y} \subseteq \mathbb{R}^p$
$i(k) \in \{1, \ldots, s\}$

- Approximates nonlinear dynamics arbitrarily well

- Suitable for stability analysis, reachability analysis (verification), controller synthesis, ...

Discrete Hybrid Automata

(Torrisi, Bemporad, 2003)

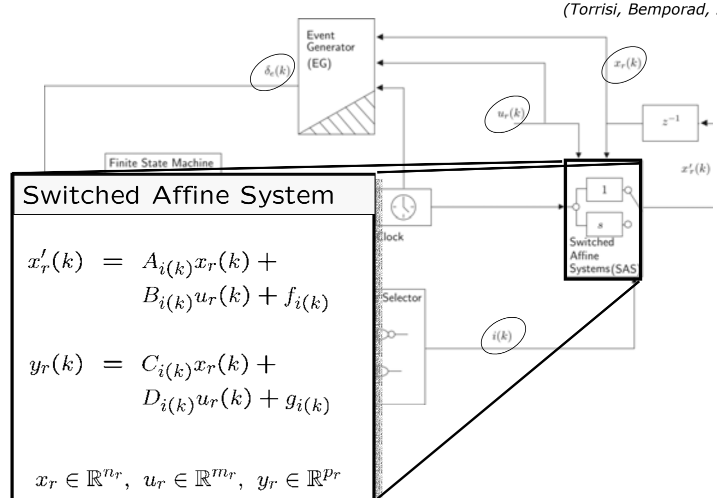


$x_r \in \mathbb{R}^{n_r}$ = continuous states
$x_b \in \{0,1\}^{n_b}$ = binary states
$i(k) \in \{1, \ldots, s\}$ = mode

$u_r \in \mathbb{R}^{m_r}$ = continuous inputs
$u_b \in \{0,1\}^{m_b}$ = binary inputs
$\delta_e \in \{0,1\}^{n_e}$ = event conditions

Discrete Hybrid Automata

(Torrisi, Bemporad, 2003)



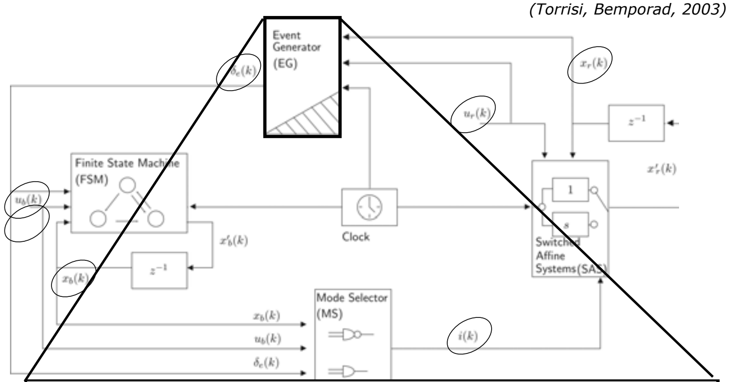
Switched Affine System

$$\begin{aligned} x_r'(k) &= A_{i(k)}x_r(k) + \\ & \quad B_{i(k)}u_r(k) + f_{i(k)} \\ y_r(k) &= C_{i(k)}x_r(k) + \\ & \quad D_{i(k)}u_r(k) + g_{i(k)} \end{aligned}$$

$x_r \in \mathbb{R}^{n_r}$, $u_r \in \mathbb{R}^{m_r}$, $y_r \in \mathbb{R}^{p_r}$

Discrete Hybrid Automata

(Torrisi, Bemporad, 2003)



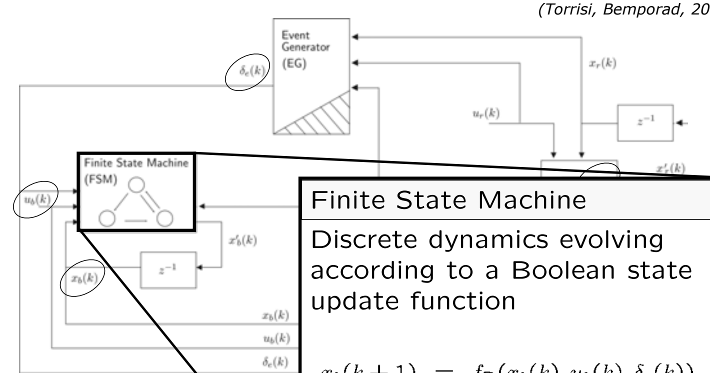
Event Generator

$$[\delta_e^i(k)=1] \leftrightarrow [H^i x_r(k)+K^i u_r(k)+J^i k \le W^i],\ \ \delta_e^i \in \{0,1\}$$

$$x_r \in \mathbb{R}^{n_r},\ \ u_r \in \mathbb{R}^{m_r},\ \ \delta_e^i \in \{0,1\}$$

Discrete Hybrid Automata

(Torrisi, Bemporad, 2003)



Finite State Machine

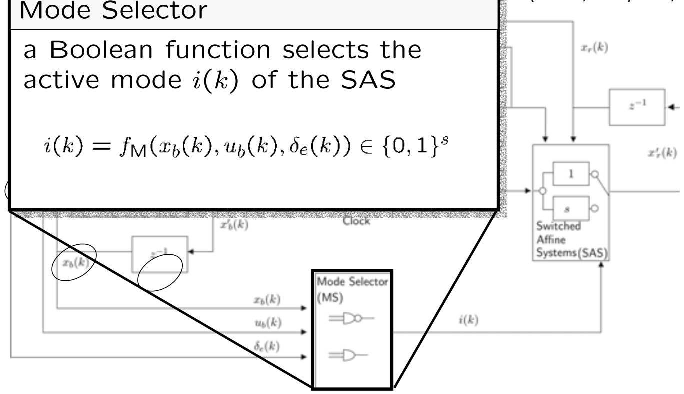
Discrete dynamics evolving according to a Boolean state update function

$$\begin{aligned} x_b(k+1) &= f_{\mathsf{B}}(x_b(k),u_b(k),\delta_e(k)) \\ y_b(k) &= g_{\mathsf{B}}(x_b(k),u_b(k),\delta_e(k)) \end{aligned}$$

$$x_b \in \{0,1\}^{n_b},\ \ y_b \in \{0,1\}^{p_b}$$

Discrete Hybrid Automata

(Torrisi, Bemporad, 2003)



Mode Selector

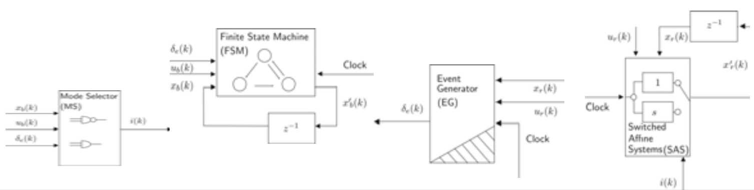
a Boolean function selects the active mode $i(k)$ of the SAS

$$i(k) = f_{\mathsf{M}}(x_b(k),u_b(k),\delta_e(k)) \in \{0,1\}^s$$

Logic and Inequalities

Glover 1975, Williams 1977

$X_1 \vee X_2$	$\delta_1+\delta_2 \ge 1$
Any logic statement $f(X)=$TRUE	$A\delta \le B$
$\bigwedge_{j=1}^{m}\left(\vee_{i\in P_j} X_i \vee_{i\in N_j} \neg X_i\right)$ (CNF), $N_j, P_j \subseteq \{1,\ldots,n\}$	$\begin{cases} 1 \le \sum_{i\in P_1}\delta_i + \sum_{i\in N_1}(1-\delta_i) \\ \vdots \\ 1 \le \sum_{i\in P_m}\delta_i + \sum_{i\in N_m}(1-\delta_i)\end{cases}$
$[\delta_e^i(k)=1] \leftrightarrow [H^i x_r(k) \le W^i]$	$H^i x_r(k) - W^i \le M^i(1-\delta_e^i)$; $H^i x_r(k) - W^i > m^i \delta_e^i$
IF δ THEN $z=a_1^T x + b_1^T u + f_1$ ELSE $z=a_2^T x+b_2^T u+f_2$	$(m_2-M_1)\delta + z \le a_2 x + b_2 u + f_2$; $(m_1-M_2)\delta - z \le -a_2 x - b_2 u - f_2$; $(m_1-M_2)(1-\delta)+z \le a_1 x + b_1 u + f_1$; $(m_2-M_1)(1-\delta)-z \le -a_1 x - b_1 u - f_1$



Logic $\rightarrow$ Inequalities: Symbolic Approach

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:

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

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

➡ Every logic proposition can be translated into linear integer inequalities

Logic $\rightarrow$ Inequalities: Geometric Approach

Geometric approach: The polytope $P \triangleq \{\delta : A\delta \leq B\}$ is the convex hull of the rows of the truth table T

T:

x_1	x_2	$\ldots$	x_{n-1}	$x_n = F(x_1, \ldots, x_{n-1})$
0	0		0	1
0	0		1	0
$\vdots$	$\vdots$		$\vdots$	
1	1		1	1

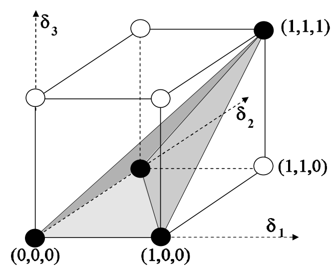
$\Longrightarrow A\delta \leq B, \delta \in \{0,1\}^n$

➡ Every logic proposition can be translated into linear integer inequalities

Logic $\rightarrow$ Inequalities: Geometric Approach

Example: logic "AND"

δ_1	δ_2	δ_3
0	0	0
0	1	0
1	0	0
1	1	1

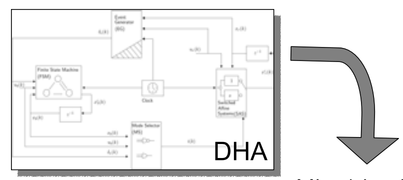


Key idea:
White points cannot be in the hull of black points

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

Convex hull algorithms: `cdd, lrs, qhull, chD, Hull, Porto`

Mixed Logical Dynamical Systems



Mixed Logical Dynamical (MLD) Systems

$$\begin{array}{rcl} x(t+1) & = & Ax(t) + B_1 u(t) + B_2 \delta(t) + B_3 z(t) + B_5 \\ y(t) & = & Cx(t) + D_1 u(t) + D_2 \delta(t) + D_3 z(t) + D_5 \\ E_2 \delta(t) + E_3 z(t) & \leq & E_4 x(t) + E_1 u(t) + E_5 \end{array}$$

(Bemporad, Morari 1999)

Continuous and binary variables $\quad x \in \mathbb{R}^{n_r} \times \{0,1\}^{n_b},\ u \in \mathbb{R}^{m_r} \times \{0,1\}^{m_b}$
$\quad y \in \mathbb{R}^{p_r} \times \{0,1\}^{p_b},\ \delta \in \{0,1\}^{r_b},\ z \in \mathbb{R}^{r_r}$

- Computationally oriented (Mixed Integer Programming)
- Suitable for optimal controller synthesis, verification, ...

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

$\longrightarrow$

$$\begin{aligned} -m\delta(t) &\le x(t) - m \\ -(M+\epsilon)\delta(t) &\le -x(t) - \epsilon \end{aligned} \qquad \begin{aligned} &M = -m = 10 \\ &\epsilon > 0 \quad \text{small} \end{aligned}$$

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

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

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

- Rewrite as a linear equation

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

HYSDEL

(HYbrid Systems DEscription Language)

- Describe *hybrid systems:*

– Automata
– Logic
– Lin. Dynamics
– Interfaces
– Constraints

(Torrisi, Bemporad, 2003)

- Automatically generate MLD models in Matlab

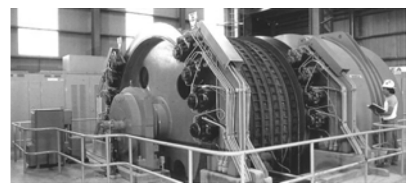
Download: **http://www.dii.unisi.it/~bemporad/tools.html**
http://control.ethz.ch/~hybrid/hysdel

Example 1: AD section

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

```
SYSTEM tank {
   INTERFACE {
      STATE {
         REAL h  [-0.3,0.3]; }
      INPUT {
         REAL Q  [-10,10]; }
      PARAMETER {
         REAL k    = 1;
         REAL e    = 1e-6; }
   } /* end interface */

   IMPLEMENTATION {
      AUX {
         BOOL s; }

      AD {
         s = hmax - h <= 0; }

      CONTINUOUS {
         h = h + k * Q; }
   } /* end implementation */
} /* end system */
```

Example 2: DA section



Nonlinear amplification unit

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

```
SYSTEM motor {
   INTERFACE {
      STATE {
         REAL ucomp; }
      INPUT {
         REAL u [0,10]; }
      PARAMETER {
         REAL ut    = 1;
         REAL e     = 1e-6;}
   } /* end interface */

IMPLEMENTATION {
      AUX {
         REAL unl;
         BOOL th; }
      AD {
         th = ut - u <= 0; }

      DA {
         unl = { IF th THEN 2.3*u - 1.3*ut
                   ELSE u };  }

      CONTINUOUS {
         ucomp = unl; }
   } /* end implementation */
} /* end system */
```

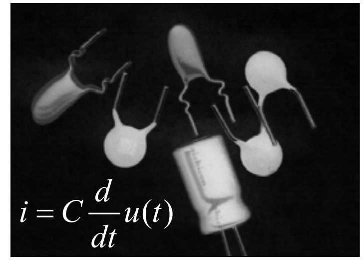
Example 3: LOGIC section

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

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

```
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

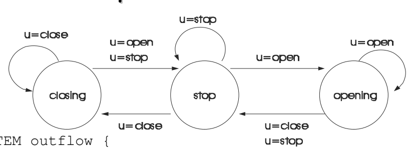


Apply forward difference rule:

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

```
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



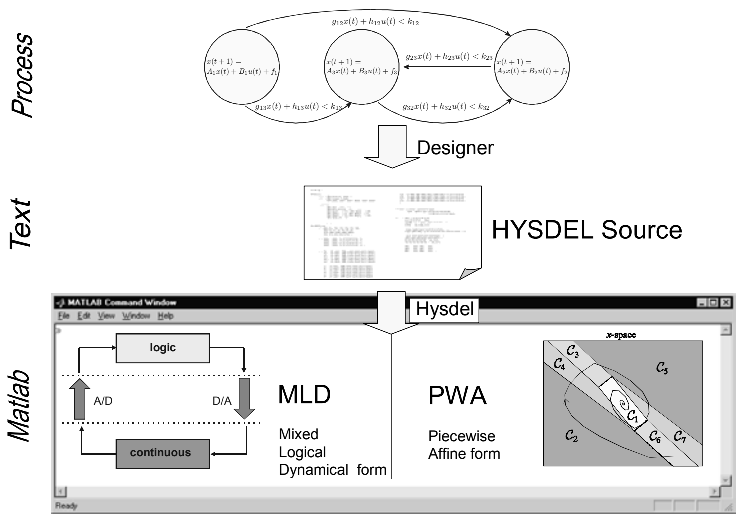
```
SYSTEM outflow {
   INTERFACE {
      STATE {
         BOOL closing, stop, opening; }
      INPUT {
         BOOL uclose, uopen, ustop; }
   } /* end of interface */

   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

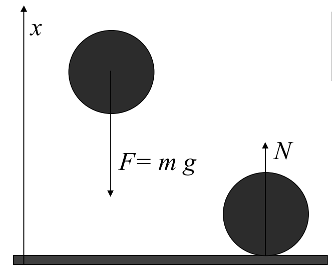
$$0 \le h \le h_{\max}$$

```
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 */
```

Modeling Flow



Example: Bouncing Ball



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

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

How to model this system in MLD form?

HYSDEL - Bouncing Ball

```
SYSTEM bouncing_ball {
INTERFACE {
/* Description of variables and constants */
        STATE { REAL height [-10,10];
                REAL velocity [-100,100];   }

        PARAMETER {
                REAL g=9.8;
                REAL dissipation=.4;  /* 0=elastic, 1=completely anelastic */
                REAL Ts=.05; }
}
IMPLEMENTATION {
        AUX {   REAL z1;
                REAL z2;
                BOOL negative;   }

        AD {    negative = height <= 0; }

        DA {    z1 = {  IF negative THEN height-Ts*velocity
                        ELSE height+Ts*velocity-Ts*Ts*g};
                z2 = {  IF negative THEN -(1-dissipation)*velocity
                        ELSE velocity-Ts*g};  }
        CONTINUOUS {
                height = z1;
                velocity=z2;}
}}
```

System Theory for Hybrid Systems

- Analysis
  - Well-posedness
  - Realization & Transformation
  - Reachability (=Verification)
  - Observability
  - Stability

- Synthesis
  - Control
  - State estimation
  - Identification
  - Modeling language

Well-posedness

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

- MLD well-posedness :

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

$$\delta(t) = F(x(t), u(t))$$
$$z(t) = G(x(t), u(t))$$

$$\{x(t), u(t)\} \to \{x(t+1)\}$$
$$\{x(t), u(t)\} \to \{y(t)\}$$

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)

Realization and Transformation (State-Space Hybrid Models)

Other 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
Generalization of LC systems (De Schutter, De Moor, 2000)

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

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

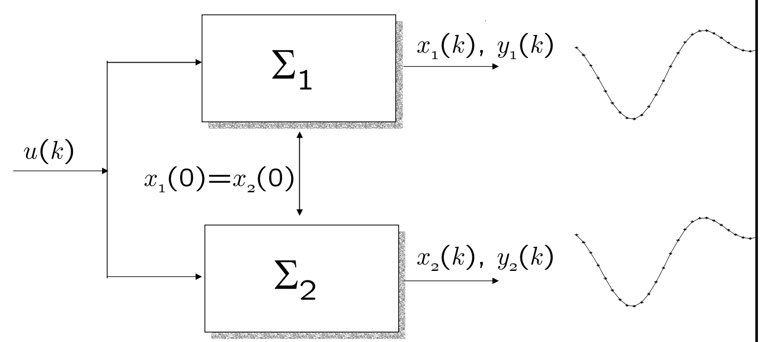
MMPS function: defined by the grammar
$M := x_i | \alpha | \max(M_1, M_2) | \min(M_1, M_2) | M_1 + M_2 | \beta M_1$

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

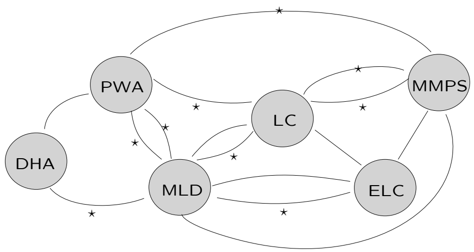
Used for modeling discrete-event systems (t=event counter)

Equivalences of Hybrid Models

Definition 1 *Two hybrid systems Σ_1, Σ_2 are* equivalent *if for all initial conditions $x_1(0) = x_2(0)$ and input $\{u_1(k)\}_{k\in\mathbb{Z}_+} = \{u_2(k)\}_{k\in\mathbb{Z}_+}$ then $x_1(k) = x_2(k)$ and $y_1(k) = y_2(k)$, for all $k \in \mathbb{Z}_+$.*



Equivalence Results



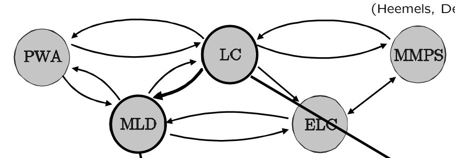
Theorem 1 *All the above six classes of discrete-time hybrid models are equivalent (possibly under some additional assumptions, such as boundedness of input and state variables)*

(Heemels, De Schutter, Bemporad, *Automatica,* 2001)
(Torrisi, Bemporad, *IEEE CST,* 2003)
(Bemporad and Morari, *Automatica*, 1999)
(Bemporad, Ferrari-T.,Morari, *IEEETAC*, 2000)

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

MLD and LC Systems

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



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 $\delta_i(t) \in \{0,1\}$

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

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

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

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

MLD and PWA Systems

Theorem 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

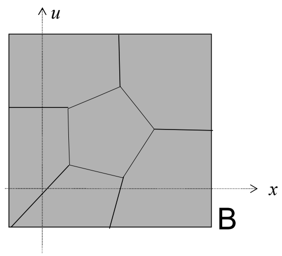
Efficient MLD to PWA Conversion

- Proof is constructive: given an MLD system it returns its equivalent PWA form
- Drawback: it needs the enumeration of all possible combinations of binary states, binary inputs, and δ variables
- Most of such combinations lead to empty regions
- Efficient algorithms are available for converting MLD models into PWA models avoiding such an enumeration:
  - A. Bemporad, "*A Recursive Algorithm for Converting Mixed Logical Dynamical Systems into an Equivalent Piecewise Affine Form*", IEEE Trans. Autom. Contr., October 2003.
  - T. Geyer, F.D. Torrisi and M. Morari, "*Efficient Mode Enumeration of Compositional Hybrid Models*", HSCC'03

MLD to PWA Conversion Algorithm

(Bemporad, 2002)

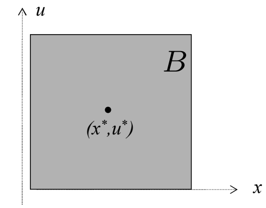
GOAL: split a given set B of states+inputs into polyhedral cells and find the affine dynamics in each cell, therefore determining the PWA system which is equivalent to the given MLD system



Note: the cells Ω_i are embedded in R^{n+m} by treating integer states and integer inputs as continuous vars:
$x_b, u_b \in [0,1] \rightarrow x_b, u_b \in [-1/2, 1/2) \cup [1/2, 3/2]$

MLD to PWA Conversion - Start

- Let (x^*, u^*) be a given point in B (e.g.: (x^*, u^*) is the Chebychev center of B, computable via LP)
- Problem: (x^*, u^*) may not be 0/1 valued



- Find (x_1, u_1) which is closest to (x^*, u^*), is integer feasible, and satisfies the MLD constraints:

$$
\begin{array}{rcll}
(x_1, u_1, \delta_1, z_1) & = & \arg\min_{x,u,\delta,z} & \left\| \left[\begin{smallmatrix} x \\ u \end{smallmatrix}\right] - \left[\begin{smallmatrix} x^* \\ u^* \end{smallmatrix}\right] \right\|_\infty \\
 & & \text{subj. to} & E_2\delta + E_2 z \leq E_1 u + E_4 x + E_5 \\
 & & & \left[\begin{smallmatrix} x \\ u \end{smallmatrix}\right] \in \mathcal{B} \\
 & & & x_\ell \in \{0,1\}^{n_\ell},\ u_\ell \in \{0,1\}^{m_\ell} \\
 & & & \delta \in \{0,1\}^{r_\ell},\ z \in \mathbb{R}^{r_c}
\end{array}
$$

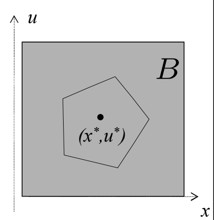
Mixed Integer Linear Program (MILP)

MLD to PWA Conversion

Now fix $\delta = \delta_1$. To find the polyhedral cell Ω_i and dynamics (A_i, B_i, f_i):

1. From MLD constraints, compute

$$
\begin{array}{rcl}
z(k) & = & K_{4i} x_c(k) + K_{1i} u_c(k) + K_{5i}, \\
 & & \forall x(k), u(k) : \begin{bmatrix} x_\ell(k) \\ u_\ell(k) \\ F(x(k), u(k)) \end{bmatrix} = \begin{bmatrix} x_{\ell 1} \\ u_{\ell 1} \\ \delta_1 \end{bmatrix}
\end{array}
$$



2. Substitute z(k) in the MLD dynamics

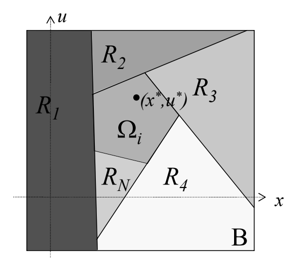
$$
\begin{array}{rcl}
x_c(k+1) & = & (A_{cc} + B_{3c}K_{4i})x_c(k) + (B_{1cc} + B_{3c}K_{1i})u_c(k) + (B_{3c}K_{5i} + B_{2c}\delta_1 + A_{c\ell}x_{\ell 1} + B_{1c\ell}u_{\ell 1}) \\
x_\ell(k+1) & = & [\text{similar}] \\
y_c(k) & = & (C_{cc} + D_{3c}K_{4i})x_c(k) + (D_{1cc} + D_{3c}K_{4i})u_c(k) + (D_{3c}K_{5i} + D_{2c}\delta_1 + C_{c\ell}x_{\ell 1} + D_{1c\ell}u_{\ell 1}) \\
y_\ell(k) & = & [\text{similar}]
\end{array}
$$

3. Find polyhedral cell Ω_i

$$
\begin{array}{rl}
\Omega_i = \{\left[\begin{smallmatrix} x_c \\ u_c \end{smallmatrix}\right] : & (E_3K_{4i} - E_{4c})x_c + (E_3K_{1i} - E_{1c})u_c \leq \\
 & (E_{1\ell}u_{\ell i} - E_2\delta_i - E_3K_{5i} + E_{4\ell}x_{\ell i} + E_5)\} \times \{x_{\ell i}\} \times \{u_{\ell i}\}
\end{array}
$$

MLD to PWA Conversion - Partition

Now partition the rest of the space $B \backslash \Omega_i$



$$\Omega_i = \{\left[\begin{smallmatrix} x \\ u \end{smallmatrix}\right] : A\left[\begin{smallmatrix} x \\ u \end{smallmatrix}\right] \leq B\}$$

$$R_i = \{\left[\begin{smallmatrix} x \\ u \end{smallmatrix}\right] : A^i\left[\begin{smallmatrix} x \\ u \end{smallmatrix}\right] > B^i, A^j\left[\begin{smallmatrix} x \\ u \end{smallmatrix}\right] > B^j,\ \forall j < i\}$$

Theorem: $\{\Omega_i, R_1, \ldots, R_N\}$ is a partition of B

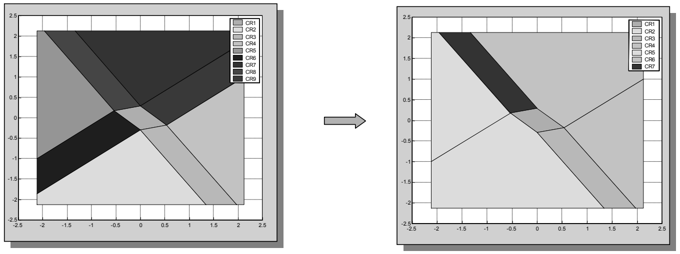
Proceed iteratively: for each region R_i repeat the whole procedure with $B \cap R_i$

Note: similar to multiparametric Quadratic Programming algorithm

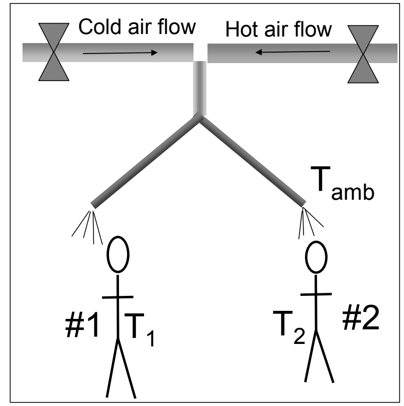
(Bemporad et al., 2002)

MLD to PWA Conversion - Union

Regions where the affine dynamics is the same can be joined (when their union is convex). *(Bemporad, Fukuda, Torrisi, 2001)*



Example: Heat and Cool



Rules of the game:

- #1 turns the heater (air conditioning) on whenever he is cold (hot)
- If #2 is cold he turns the heater on, unless #1 is hot
- If #2 is hot he turns the air conditioning on, unless #2 is cold
- Otherwise, heater and air conditioning are off

- $\dot{T}_1 = -\alpha_1(T_1 - T_{\text{amb}}) + k_1(u_{\text{hot}} - u_{\text{cold}})$ (body temperature dynamics of #1)
- $\dot{T}_2 = -\alpha_2(T_2 - T_{\text{amb}}) + k_2(u_{\text{hot}} - u_{\text{cold}})$ (body temperature dynamics of #2)

Hybrid HYSDEL Model

```
/* Heat and Cool example
   (C) 2002 by A. Bemporad, Las Vegas, December 9, 2002 */

SYSTEM heatcool {

INTERFACE {
    STATE { REAL T1 [-10,50];
            REAL T2 [-10,50];
          }
    INPUT { REAL Tamb [-10,50];
          }
    PARAMETER {
        REAL Ts = .5; /* sampling time, second
        REAL alpha1 = 1;
        REAL alpha2 = 0.5;
        REAL k1 = .8;
        REAL k2 = .4;
        REAL Thot1 = 30;
        REAL Tcold1 = 15;
        REAL Thot2 = 35;
        REAL Tcold2 = 10;
        REAL Uc = 2; /* cold water flow */
        REAL Uh = 2; /* hot water flow */
    }
}

IMPLEMENTATION {
        AUX { REAL uhot, ucold;
              BOOL hot1, hot2, cold1, cold2;
        }
        AD  { hot1 = T1>=Thot1;
              hot2 = T2>=Thot2;
              cold1 = T1<=Tcold1;
              cold2 = T2<=Tcold2;
        }
        DA  { uhot = {IF cold1 | (cold2 & ~hot1) THEN Uh ELSE 0};
              ucold = {IF hot1 | (hot2 & ~cold1) THEN Uc ELSE 0};
        }
        CONTINUOUS { T1 = T1+Ts*(-alpha1*(T1-Tamb)+k1*(uhot-ucold));
                     T2 = T2+Ts*(-alpha2*(T2-Tamb)+k2*(uhot-ucold));
        }
      }
    }
```

Hybrid MLD Model

- MLD model

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

- 2 continuous states: (temperatures T_1, T_2)
- 1 continuous input: (room temperature T_{amb})
- 2 auxiliary continuous vars: (water flows u_{hot}, u_{cold})
- 6 auxiliary binary vars: (4 thresholds + 2 for OR condition)
- 20 mixed-integer inequalities

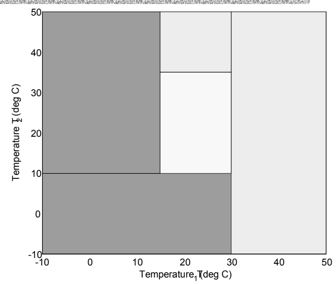
Possible combination of integer variables: $2^6 = 64$

Hybrid PWA Model

- PWA model

$$\begin{aligned} x(k+1) &= A_{i(k)}x(k) + B_{i(k)}u(k) + f_{i(k)} \\ y(k) &= C_{i(k)}x(k) + D_{i(k)}u(k) + g_{i(k)} \\ \\ i(k) \text{ s.t. } & H_{i(k)}x(k) + J_{i(k)}u(k) \leq K_{i(k)} \end{aligned}$$

- 2 continuous states: (temperatures T_1,T_2)
- 1 continuous input: (room temperature T_{amb})
- 5 polyhedral regions *(partition does not depend on input)*



Computation time: 0.72 s in Matlab

$u_{\text{hot}} = 0$, $u_{\text{cold}} = 0$ | $u_{\text{hot}} = 0$, $u_{\text{cold}} = \bar{U}_C$ | $u_{\text{hot}} = \bar{U}_H$, $u_{\text{cold}} = 0$

Why are we interested in getting MLD and PWA models ?

Major Advantage of MLD Framework

Many problems of analysis:
- Stability
- Safety
- Controllability
- Observability

Many problems of synthesis:
- Controller design
- Filter design / Fault detection & state estimation

can be expressed as (mixed integer) mathematical programming problems for which many algorithms and software tools exist.

(However, all these problems are NP-hard !)

Hybrid Models

Each model is most advantageous depending on task:

Task	*Model*
Modeling	DHA
Simulation	DHA
Control	MLD,PWA,MMPS
Stability	PWA
Verification	PWA
Identification	PWA
Fault Detection	MLD
Estimation	MLD