Motivation: Embedded Systems

- Consumer electronics
- Home appliances
- Office automation
- Automobiles
- Industrial plants
- ...
Key Requirements for Hybrid Models

- **Descriptive** enough to capture the behavior of the system
  - continuous dynamics (physical laws)
  - logic/discrete components (switches, automata, ...)
  - interconnections between logic and dynamics

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

\[
\begin{align*}
\begin{cases}
    x' &= Ax + Bu \\
y &= Cx + Du
\end{cases} & \quad \text{linear systems} \\
\begin{cases}
    x' &= f(x, u, t) \\
y &= g(x, u, t)
\end{cases} & \quad \text{nonlinear systems}
\end{align*}
\]

 Generally speaking, the goodness of a model depends on the task the model is build for!

Piecewise Affine Systems

(Sontag 1981)

\[
x' = A_i(k)x(k) + B_i(k)u(k) + f_i(k) \\
y(k) = C_i(k)x(k) + D_i(k)u(k) + g_i(k) \\
i(k) \text{ s.t. } H_i(k)x(k) + J_i(k)u(k) \leq K_i(k)
\]

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

- Approximates nonlinear dynamics arbitrarily well
- Suitable for stability analysis, reachability analysis (verification), controller synthesis, ...

Discrete Hybrid Automata

(Torrisi, Bemporad, 2003)

\[
x_{r}(k) \in \mathbb{R}^{n_{r}} = \text{continuous states} \\
x_{b}(k) \in \{0,1\}^{n_{b}} = \text{binary states} \\
i(k) \in \{1, \ldots, s\} = \text{mode}
\]

\[
u_{r}(k) \in \mathbb{R}^{m_{r}} = \text{continuous inputs} \\
u_{b}(k) \in \{0,1\}^{m_{b}} = \text{binary inputs} \\
\delta(k) \in \{0,1\}^{n_{e}} = \text{event conditions}
\]
Discrete Hybrid Automata

Mode Selector

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

$$i(k) = f_M(x_i(k), u_i(k), \delta_e(k)) \in \{0, 1\}^n$$

Event Generator

$$x_i(k+1) = f_B(x_i(k), u_i(k), \delta_e(k))$$

$$y_i(k) = g_B(x_i(k), u_i(k), \delta_e(k))$$

$x_i \in \{0, 1\}^{r_i}$, $y_i \in \{0, 1\}^{p_i}$

Finite State Machine

Discrete dynamics evolving according to a Boolean state update function

Logic and Inequalities

Glover 1975, Williams 1977

$$X_1 \lor X_2$$

Any logic statement $f(X) = \text{TRUE}$

$$A \emptyset B$$

$$\delta_1 + \delta_2 \geq 1$$

$$1 \leq \sum_{i \in P_1} \delta_i + \sum_{i \in P_2} (1 - \delta_i)$$

$$1 \leq \sum_{i \in P_1} \delta_i + \sum_{i \in P_2} (1 - \delta_i)$$

$$[\delta_i(k) = 1] \Rightarrow [H^i x_r(k) \leq W^i]$$

$$H^{r_i}(x_r(k) - W^i) \leq M^{r_i}(1 - \delta_i)$$

$$H^{r_i}(x_r(k) - W^i) > M^{r_i} \delta_i$$

IF $\delta$ THEN $z = a_1^i x + b_1^i u + f_1$

ELSE $z = a_2^i x + b_2^i u + f_2$
Mixed Logical Dynamical Systems

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

(Bemporad, Morari 1999)

Continuous and binary variables

\[
x \in \mathbb{R}^n \times \{0,1\}^{n_b}, \ u \in \mathbb{R}^m \times \{0,1\}^{n_b} \\
y \in \mathbb{R}^p \times \{0,1\}^{n_b}, \ \delta \in \{0,1\}^{n_r}, \ z \in \mathbb{R}^r
\]

HYSDEL

(HYbrid Systems DEscription Language)

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

Describe hybrid systems:

- Automatically generate MLD models in Matlab

Download:

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

MLD and PWA Systems

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

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

- 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 \( \delta \) variables
- Most of such combinations lead to empty regions
- Efficient algorithms are available for converting MLD models into PWA models avoiding such an enumeration:
  - T. Geyer, F.D. Torrisi and M. Morari, "Efficient Mode Enumeration of Compositional Hybrid Models", HSCC’03

Identification of Hybrid Systems

Estimate from data both the parameters of the affine submodels and the partition of the PWA map

**Example** Let the data be generated by the PWARX system

\[
y_k = \begin{cases}
-0.4 & 1 \ 1.5 \varphi_k + \varepsilon_k \\
\text{if} & 4 \ -1 \ 10 \varphi_k < 0
\end{cases}
\]

\[
y_k = \begin{cases}
0.5 & -1 \ -0.5 \varphi_k + \varepsilon_k \\
\text{if} & -4 \ 1 \ -10 \varphi_k \leq 0
\end{cases}
\]

\[
y_k = \begin{cases}
-0.3 & 0.5 \ -1.7 \varphi_k + \varepsilon_k \\
\text{if} & -5 \ -1 \ 6 \varphi_k < 0
\end{cases}
\]

with \( \varphi_k = [y_{k-1} \ u_{k-1} \ 1]^T \), \( |u_k| \leq 5 \) and \( |\varepsilon_k| \leq 0.1 \)
Approaches to PWA Identification

- Mixed-integer linear or quadratic programming
  J. Roll, A. Bemporad and L. Ljung, "Identification of hybrid systems via mixed-integer programming", Automatica, 2004

- Bounded error & partition of infeasible set of inequalities

- K-means clustering in a feature space

- Other approaches:
  - Polynomial factorization (algebraic approach) (R. Vidal, S. Soatto, S. Sastry, 2003)
  - Hyperplane clustering in data space (E. Münz, V. Krebs, IFAC 2002)

Why are we interested in getting MLD and PWA models?

Major Advantages of MLD/PWA Models

Many problems of analysis:
- Stability
- Safety
- Reachability
- Observability
- Well-posedness

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

Controller Synthesis for Hybrid Systems

...can be solved using mathematical programming

(However, all these problems are NP-hard !)
Mixed Integer Quadratic Program (MIQP)

\[ \min_{\xi} J(\xi, x(0)) = \sum_{t=0}^{T-1} y(t) Q y(t) + u(t) R u(t) \]

subject to \[
\begin{align*}
x(t+1) &= A x(t) + B_1 u(t) + B_2 \delta(t) + B_3 z(t) + B_5 \\
y(t) &= C x(t) + D_1 u(t) + D_2 \delta(t) + D_3 z(t) + D_5 \\
E_2 \delta(t) + E_3 z(t) &\leq E_4 x(t) + E_5 u(t) + E_6 
\end{align*}
\]

\[ \xi = [u(0), \ldots, u(T-1), \delta(0), \ldots, \delta(T-1), z(0), \ldots z(T-1)]' \]

Mixed Integer Linear Program (MILP)

\[ \min_{[\xi]} J(\xi, x(0)) = \sum_{t=0}^{T-1} \| Q y(t) \|_\infty + \| R u(t) \|_\infty \]

subject to MLD model

- Introduce slack variables:
  \[ \begin{align*}
  \epsilon_i^x &\geq \| Q y(t+k) \|_\infty, i = 1, \ldots, p, k = 1, \ldots, T-1 \\
  \epsilon_i^u &\geq -\| Q y(t+k) \|_\infty, i = 1, \ldots, p, k = 1, \ldots, T-1 \\
  \epsilon_i^x &\geq \| R u(t+k) \|_\infty, i = 1, \ldots, m, k = 0, \ldots, T-1 \\
  \epsilon_i^u &\geq -\| R u(t+k) \|_\infty, i = 1, \ldots, m, k = 0, \ldots, T-1 
  \end{align*} \]

Set \[ \xi \triangleq [\epsilon_1^x, \ldots, \epsilon_p^x, \epsilon_1^u, \ldots, \epsilon_m^u, U, \delta, z] \]

MPC for Hybrid Systems

Model Predictive (MPC) Control

- At time \( t \) solve with respect to \( U \triangleq \{u(t), u(t+1), \ldots, u(t+T-1)\} \) the finite-horizon open-loop, optimal control problem:

\[
\begin{align*}
\min_{u(t), \ldots, u(t+T-1)} & \sum_{k=0}^{T-1} \| y(t+k|t) - r(t) \| + \rho \| u(t+k) \| \\
& + \sigma(\| \delta(t+k) - \delta_r \| + \| z(t+k) - z_r \| + \| x(t+k|t) - x_r \|) \\
\text{subject to} & \ MLD \text{ model} \\
& x(t+1) = A x(t) + B_1 u(t) + B_2 \delta(t) + B_3 z(t) + B_5 \\
& y(t) = C x(t) + D_1 u(t) + D_2 \delta(t) + D_3 z(t) + D_5 \\
& E_2 \delta(t) + E_3 z(t) \leq E_4 x(t) + E_5 u(t) + E_6 \\
& x(t) = x(0) \\
& x(t+T|t) = x_r 
\end{align*}
\]

- Apply only \( u(t) = u'(t) \) (discard the remaining optimal inputs)
- Repeat the whole optimization at time \( t+1 \)

Closed-Loop Stability

**Theorem 1** Let \( (x_r, u_r, \delta_r, z_r) \) be the equilibrium values corresponding to the set point \( r \), and assume \( x(0) \) is such that the MPC problem is feasible at time \( t = 0 \). Then \( \forall Q, R > 0, \forall \sigma > 0 \), the MPC controller stabilizes the MLD system

\[
\begin{align*}
l \lim_{t \to \infty} y(t) &= r \\
l \lim_{t \to \infty} u(t) &= u_r \\
l \lim_{t \to \infty} x(t) &= x_r, l \lim_{t \to \infty} \delta(t) &= \delta_r, l \lim_{t \to \infty} z(t) &= z_r, \\
\end{align*}
\]

and all constraints are fulfilled.

(Bemporad, Morari 1999)

Proof: Easily follows from standard Lyapunov arguments
Switching System:

\[ x(t+1) = 0.5 \begin{bmatrix} \cos \alpha(t) & -\sin \alpha(t) \\ \sin \alpha(t) & \cos \alpha(t) \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) \]

\[ y(t) = \begin{bmatrix} 0 & 1 \end{bmatrix} x(t) \]

\[ \alpha(t) = \begin{cases} \pi & \text{if } 0.5 x(t) \geq 0 \\ \frac{\pi}{2} & \text{if } 0.5 x(t) < 0 \end{cases} \]

Constraint: \(-1 \leq u(t) \leq 1\)

Mixed-Integer Programming Solvers

- Mixed-Integer Programming is \(NP\)-hard

**BUT**

- General purpose Branch & Bound/Branch & Cut solvers available for MILP and MIQP (CPLEX, Xpress-MP, BARON, GLPK, ...)

More solvers and benchmarks: [http://plato.la.asu.edu/bench.html](http://plato.la.asu.edu/bench.html)

- No need to reach global optimum (see proof of the theorem), although performance deteriorates

Main drawbacks:

- Loss of the original discrete structure (Boolean formulas)
- On-line combinatorial optimization
Mixed-Integer Programming Solvers

- Mixed-Integer Programming is NP-hard

BUT

- General purpose Branch & Bound/Branch & Cut solvers available for MILP and MIQP (CPLEX, Xpress-MP, BARON, GLPK, ...)
  
More solvers and benchmarks: [http://plato.la.asu.edu/bench.html](http://plato.la.asu.edu/bench.html)

- No need to reach global optimum (see proof of the theorem), although performance deteriorates

Main drawbacks:
- Loss of the original discrete structure (Boolean formulas)
- On-line combinatorial optimization

Limitations of MIP Approaches

1) The discrete/logic part of the hybrid system must be converted into linear MI inequalities (e.g.: by HYSDEL)

   \[ \text{LOSS of the original discrete structure} \]
   \[ \text{Introduction of auxiliary binary variables} \]

2) The efficiency of the MIP solver relies upon the tightness of the continuous LP/QP relaxations.

   \[ \text{Poor performance (=many LP/QPs) if relaxations are not tight} \]

   \[ \text{We need better solution techniques} \]

A “Hybrid” Solver

Combine MIP and Constraint Programming (CP) to overcome the previous difficulties

Why CP?
- More flexible modeling than MIP for logic constraints;
- Structure is kept and exploited to direct the search.

Why MIP?
- Specialized techniques for highly structured problems (e.g. LP problems);
- A wide range of tight relaxations are available

Why a combined approach?
Performance increase already shown in other application domains

(Harjunkoski, Jain, Grossmann, 2000)

Logic-Based Branch&Bound

The optimal control problem can be recast as

\[ \text{min} \quad f(z) \]
\[ \text{s.t.} \quad Gz \leq \bar{d}, \quad Gz = \bar{d} \]
\[ G'z + D\mu \leq E', \quad G'z + D\mu = E' \]
\[ g(\nu, \mu) \]

\[ z \in \mathbb{R}^n, \quad \mu \in \{0,1\}^n, \quad \nu \in \{0,1\}^n \]

This problem can be solved by a Logic-based Branch&Bound algorithm

Logic-based B&B “ingredients”

- A relaxed MILP problem obtained by (1a),(1b) and (1c)
  
  LP solver

- A CP feasibility problem obtained by (1d)
  
  CP solver
**Hybrid Solvers**

Using a combined approach of Constraint Logic Programming (CLP) and LP solvers.  
(Bemporad, Giorgetti, CDC’2003)

A new combined approach using SAT and Convex solvers.  
(Bemporad, Giorgetti, HSCC’04)

...SAT better solves satisfiability problems of logic formulas than CLP.

...SAT solvers are widely used in the Bounded Model Checking, Reachability analysis, and more.

We need a suitable reformulation of the optimal control problem to take advantage of SAT solvers.

---

**Numerical Example**

In a room:

- $u_{\text{hot}}$, $u_{\text{cold}}$, $u_e$: thermal powers
- $T_1$, $T_2$, $T_{\text{amb}}$: temperatures

Discrete-time continuous dynamics

$$\frac{T_i(k + 1) - T_i(k)}{T_s} = -\alpha_i(T_i(k) - T_{\text{amb}}) + k_i(u_{\text{hot}}(k) - u_{\text{cold}}(k)) + c u_e(k)$$

($T_s$: sampling time)

$\alpha_i, k_i, c$: suitable constants

---

**Switching Logic (Automaton)**

Heater:

Logic variables

$[\phi_{\text{on}}(k) = 1] \iff [T_i(k) \leq T_{\text{on}}]$  
$[\phi_{\text{off}}(k) = 1] \iff [T_i(k) < T_{\text{off}}]$  

AC System

Logic variables

$[\gamma_{\text{hi}}(k) = 1] \iff [T_i(k) \geq T_{\text{hi}}]$  
$[\gamma_{\text{lo}}(k) = 1] \iff [T_i(k) \geq T_{\text{lo}}]$  

where

$T_{\text{on}} \leq T_{\text{off}} \leq T_{\text{hi}} \leq T_{\text{lo}}$

are constant thresholds.

---

**Optimal Control Problem**

Rules for governing heater & A/C:

$$u_{\text{hot}} = \begin{cases} u_H & \text{if } h_3 = 1 \text{ (HEAT)} \\ 0 & \text{otherwise} \end{cases}$$

$$u_{\text{cold}} = \begin{cases} u_C & \text{if } h_4 = 1 \text{ (REFRIGERATE)} \\ 0 & \text{otherwise} \end{cases}$$

$u_H, u_C$: constants

Minimize

$$\sum_{k=0}^{T} |T_i(k) - T_{\text{amb}}|$$

subject to the overall hybrid dynamics and the following additional design constraints:

### Continuous Constraints on temperatures

- $-10 \leq T_1(k) \leq 50$
- $-10 \leq T_2(k) \leq 50.$

### Continuous constraint on exogenous input

$$-10 \leq u_e(k) \leq 10$$
Numerical Results

From initial condition: $T_1(0) = 5\,^\circ \mathrm{C}$, $T_2(0) = 2\,^\circ \mathrm{C}$ and for $T_{\text{amb}} = 25\,^\circ \mathrm{C}$

Optimal Control solution

<table>
<thead>
<tr>
<th>T</th>
<th>Int.-vars (s)</th>
<th>MILP (s)</th>
<th>LPs (s)</th>
<th>SATs &amp; B&amp;B</th>
<th>&quot;cuts&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>75</td>
<td>0.04</td>
<td>69</td>
<td>0.04</td>
<td>15</td>
</tr>
<tr>
<td>10</td>
<td>150</td>
<td>0.22</td>
<td>119</td>
<td>0.38</td>
<td>15</td>
</tr>
<tr>
<td>15</td>
<td>225</td>
<td>0.61</td>
<td>152</td>
<td>0.66</td>
<td>17</td>
</tr>
<tr>
<td>20</td>
<td>300</td>
<td>1.452</td>
<td>248</td>
<td>1.011</td>
<td>17</td>
</tr>
<tr>
<td>25</td>
<td>375</td>
<td>2.594</td>
<td>301</td>
<td>1.512</td>
<td>10</td>
</tr>
<tr>
<td>30</td>
<td>450</td>
<td>4.307</td>
<td>363</td>
<td>2.093</td>
<td>20</td>
</tr>
<tr>
<td>35</td>
<td>525</td>
<td>5.729</td>
<td>367</td>
<td>2.844</td>
<td>20</td>
</tr>
<tr>
<td>40</td>
<td>600</td>
<td>12.065</td>
<td>486</td>
<td>3.505</td>
<td>27</td>
</tr>
<tr>
<td>45</td>
<td>675</td>
<td>13.479</td>
<td>534</td>
<td>4.367</td>
<td>31</td>
</tr>
<tr>
<td>50</td>
<td>750</td>
<td>19.108</td>
<td>607</td>
<td>5.368</td>
<td>43</td>
</tr>
</tbody>
</table>

Pentium IV 1.8GHz
LP solver: CPLEX 8.1.1
SAT solver: zCHAFF 2003.07.22
MILP solver: CPLEX 8.1.1

Mixed-Integer Programming Solvers

- Mixed-Integer Programming is \emph{NP}-hard

\begin{itemize}
  \item General purpose Branch & Bound/Branch & Cut solvers available for MILP and MIQP (CPLEX, Xpress-MP, BARON, GLPK, ...)
  \item More solvers and benchmarks: \url{http://plato.la.asu.edu/bench.html}
  \item No need to reach global optimum (see proof of the theorem), although performance deteriorates
\end{itemize}

Main drawbacks:
- Loss of the original discrete structure (Boolean formulas)
- On-line combinatorial optimization

On-Line vs. Off-Line Optimization

- On-line optimization: given $x(t)$, solve the problem at each time step $t$
  - Mixed-Integer Linear Program (MILP)
- Off-line optimization: solve the MILP for all $x(t)$
  - Multi-parametric Mixed Integer Linear Program (mp-MILP)

Example of Multiparametric Solution

Multiparametric LP ($\xi \in \mathbb{R}^2$)

\[
\begin{align*}
\min_{\xi} & \quad -3\xi_1 - 8\xi_2 \\
\text{s.t.} & \quad \xi_1 + \xi_2 \leq 13 + x_1 \\
& \quad 5\xi_1 - 4\xi_2 \leq 20 \\
& \quad -8\xi_1 + 22\xi_2 \leq 121 + x_2 \\
& \quad -4\xi_1 - \xi_2 \leq -8 \\
& \quad -\xi_2 \leq 0 \\
& \quad -\xi_2 \leq 0
\end{align*}
\]

\[
\xi(x) = \begin{cases}
0.00 & \text{if } x \leq 1.00 \\
0.00 & \text{if } x \leq 1.00 \\
0.00 & \text{if } x \leq 1.00 \\
0.00 & \text{if } x \leq 1.00
\end{cases}
\]

\begin{cases}
0.00 & \text{if } x \leq 1.00 \\
0.00 & \text{if } x \leq 1.00 \\
0.00 & \text{if } x \leq 1.00 \\
0.00 & \text{if } x \leq 1.00
\end{cases}
\]

\begin{cases}
0.00 & \text{if } x \leq 1.00 \\
0.00 & \text{if } x \leq 1.00 \\
0.00 & \text{if } x \leq 1.00 \\
0.00 & \text{if } x \leq 1.00
\end{cases}
\]
Theorem: The multiparametric solution \( \xi^* \) is piecewise affine.

\[
\begin{align*}
\min_{\xi = \{\xi_c, \xi_d\}} & \quad f'\xi_c + d'\xi_d \\
\text{s.t.} & \quad G\xi_c + E\xi_d \leq W + Fx \\
& \quad \xi_c \in \mathbb{R}^n \\
& \quad \xi_d \in \{0, 1\}^m
\end{align*}
\]

- mp-MILP can be solved (by alternating MILPs and mp-LPs) 
  (Dua, Pistikopoulos, 1999)

- **Theorem:** The multiparametric solution \( \xi^*(x) \) is piecewise affine

- **Corollary:** The MPC controller is piecewise affine in \( x \)

\[
u(x) = \begin{cases} 
F_1 x + G_1 & \text{if } H_1 x \leq K_1 \\
\vdots & \vdots \\
F_N x + G_N & \text{if } H_N x \leq K_N 
\end{cases}
\]

More Efficient Approaches

(Borrelli, Bao, Bemporad, Morari, 2003)
(Mayne, ECC 2001)

- Explicit solutions to finite-time optimal control problems for PWA systems can be obtained using a combination of
  - Multiparametric linear (1-norm, \( \infty \)-norm), or
    - or quadratic (squared 2-norm) programming
  - Dynamic programming or enumeration of feasible mode sequences

Note: in the 2-norm case, the partition may not be polyhedral

Hybrid Control - Example

Switching System:

\[
x(t+1) = 0.8 \begin{bmatrix} \cos \omega(t) & -\sin \omega(t) \\ \sin \omega(t) & \cos \omega(t) \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) \\
y(t) = [0 \ 1] x(t) \\
a(t) = \begin{cases} 
\pi & \text{if } |1 0| x(t) \geq 0 \\
-\pi & \text{if } |1 0| x(t) < 0 
\end{cases}
\]

Constraint: \(-1 \leq u(t) \leq 1\)

Open loop:

Closed loop:
Explicit PWA Controller

\[
\begin{align*}
\dot{x}(t) &= A_i x(t) + B_i u(t) + w(t), \\
\text{if } 0.0008 - 1.000 &\leq x \\ a &\leq [0.0008 - 0.9998] \\
-1 &\leq [0.0008 - 0.9998] \\
[0.0008 - 1.000] &\leq x \\
0 &\leq [0.0008 - 0.9998] \\
[-1.000 - 0.000] &\leq x \\
\end{align*}
\]

\[u(x) = \begin{cases}
\begin{array}{c}
\text{Region } 1 \\
\text{Region } 2 \\
\text{Region } 3
\end{array}
\end{cases}
\]

\(\text{PWA law } \equiv \text{ MPC law}\)

(CPU time: 1.44 s, Pentium III 800)

Optimal Control of Continuous-Time Switched Affine Systems

(with C. Seatzu, A. Giua, D. Corona)

Hybrid MPC - Example

Closed loop:

Optimal Control of Continuous-Time Switched Affine Systems

(with C. Seatzu, A. Giua, D. Corona)
We focus on hybrid systems of the form:

\[ \dot{x}(t) = A_{i(t)}x(t) + f_{i(t)}, \quad i(t) \in S \]

\text{switched affine systems}

\[ i(t) \in S \] is the control variable, and \( S \equiv \{1, 2, \ldots, s\} \) is a finite set of integers, each one associated with an affine dynamics.

Example: A feedback control \( u(t) = F_i x(t) \) is already assigned for each mode, only the mode can be selected.

\textbf{Optimal Control Problem}

\[ V^*_N \triangleq \min_{I,T} \left\{ F(I,T) \triangleq \int_0^\infty x'(t)Q_{i(t)}x(t)dt \right\} \]

\[ \text{s.t.} \quad \dot{x}(t) = A_{i(t)}x(t) + f_{i(t)} \]

\[ x(0) = x_0 \]

\[ i(t) = i_k \text{ for } \tau_{k-1} \leq t < \tau_k \]

\[ i_k \in S, \quad k = 1, \ldots, N + 1 \]

\[ \tau_0 = 0, \quad \tau_{N+1} = +\infty \]

\[ \tau_k \in \mathbb{R}_{>0} \forall k = 1, \ldots, N \]

\( N \) = the maximum allowed number of switches (fixed a priori)

\( Q_i > 0 \) (weight for the \( i \)-th mode)

For at least one mode \( i \), \( A_i \) is as. stable and \( f_i = 0 \)

Costs associated with switches can be included

Decision Variables:

- Finite sequence of switching times \( T \equiv \{\tau_1, \ldots, \tau_N\} \)
- Finite sequence of switching modes \( I \equiv \{i_1, \ldots, i_{N+1}\} \)

\textbf{Event-based Hybrid Models}

\textbf{Continuous Petri-nets w/ controllable transitions} can be modeled as MLD systems, where the index \( k \) represent the event counter, rather than discrete-time

\textbf{Optimal control problems} can be solved via MILP, where times between transitions are continuous variables (no sampling!)

- Time optimal control
- Maximum steady-state throughput
- Other criteria ...

\textbf{Event-based Hybrid Models}

\textbf{Example:} small manufacturing system

Objective: maximize \( m[p_1] + m[p_4] \) in steady state, as soon as possible

The obtained control is: \( u[t_1](0) = 2, \ u[t_4](0) = 2 \) with \( q(0) = 0.2 \)

reaching \( m(1) = (5 \ 2.4 \ 0 \ 1.6 \ 2.4) \), \( u[t_1](1) = 0, \ u[t_4](1) = 2 \) with \( q(1) = 0.8 \)

reaching \( m(2) = (5 \ 0 \ 0 \ 4 \ 0) \) that is a steady state marking with the inputs \( u[t_1](2) = 0, \ u[t_4](2) = 0.5 \).