Hybrid Toolbox for Matlab

Alberto Bemporad

http://www.dii.unisi.it/hybrid

Hybrid Control Problem

hybrid process

on-line decision maker

desired behavior

operation constraints

continuous inputs

binary inputs

continuous states

binary states

COHES Group
Control and Optimization of Hybrid and Embedded Systems

Dept. of Information Engineering
University of Siena, Italy (founded in 1240)
Hybrid Verification Problem

- Need for a hybrid model of the process reproducing the behavior of the process (simulation)
- A model suitable for controller synthesis and verification
- A model for which computational tools can be applied
Hybrid Model: Discrete Hybrid Automaton

Event Generator

Switched Affine System

Finite State Machine

Mode Selector

\[ x_c(k + 1) = A_{i(k)}x_c(k) + B_{i(k)}u_c(k) + f_{i(k)} \]

\[ x_c \in \mathbb{R}^{n_c}, \quad u_c \in \mathbb{R}^{m_c} \]
Event variables are generated by linear threshold conditions over continuous states, continuous inputs, and time:

\[ [\delta_e^i(k) = 1] \leftrightarrow [H^i x_c(k) + K^i u_c(k) \leq W^i] \]

\( x_c \in \mathbb{R}^{nc}, \ u_c \in \mathbb{R}^{mc}, \ \delta_e \in \{0, 1\}^{ne} \)

Example: \([\delta = 1] \leftrightarrow [x_c(k) \geq 0]\)

The active mode \(i(k)\) is selected by a Boolean function of the current binary states, binary inputs, and event variables:

\[ i(k) = f_M(x_\ell(k), u_\ell(k), \delta_e(k)) \quad x_\ell \in \{0, 1\}^{n_\ell}, \ u_\ell \in \{0, 1\}^{m_\ell}, \ \delta_e \in \{0, 1\}^{ne} \]

Example:

\[ i(k) = \left[ \begin{array}{c} -u_\ell(k) \lor x_\ell(k) \\ u_\ell(k) \land x_\ell(k) \end{array} \right] \quad \begin{array}{c|c|c} u_\ell/x_\ell \end{array} \]

\[ \begin{array}{c|c|c} 0 & i = \delta & i = \delta \\ 1 & i = \delta & i = \delta \end{array} \]

the system has 3 modes
The binary state of the finite state machine evolves according to a Boolean state update function:

$$x_\ell(k+1) = f_B(x_\ell(k), u_\ell(k), \delta_e(k)) \quad x_\ell \in \{0,1\}^{n_\ell}, \ u_\ell \in \{0,1\}^{m_\ell}, \ \delta_e \in \{0,1\}^{n_e}$$

Example: $$x_\ell(k+1) = -\delta_e(k) \lor (x_\ell(k) \land u_\ell(k))$$

### Computational Hybrid Models

**Discrete Hybrid Automaton**

**Piecewise Affine (PWA) Systems**

$$x(k+1) = A_i(k)x(k) + B_i(k)u(k) + f_i(k)$$

$$y(k) = C_i(k)x(k) + D_i(k)u(k) + g_i(k)$$

$$i(k) \text{ s.t. } H_i(k)x(k) + J_i(k)u(k) \leq K_i(k)$$

**Mixed Logical Dynamical (MLD) Systems**

$$x(t+1) = Ax(t) + B_1u(t) + B_2\delta(t) + B_3z(t) + B_5$$

$$y(t) = Cx(t) + D_1u(t) + D_2\delta(t) + D_3z(t) + D_5$$

$$E_2\delta(t) + E_3z(t) \leq E_4x(t) + E_1u(t) + E_5$$

The translation from DHA to MLD/PWA is done automatically (using symbolic/mathematical programming tools).
Example: Room Temperature

Hybrid Dynamics

- #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 #1 is cold
- Otherwise, heater and air conditioning are off

\[ \dot{T}_1 = -\alpha_1(T_1 - T_{\text{amb}}) + k_1(u_{\text{hot}} - u_{\text{cold}}) \]  
\[ \dot{T}_2 = -\alpha_2(T_2 - T_{\text{amb}}) + k_2(u_{\text{hot}} - u_{\text{cold}}) \]  

(body temperature dynamics of #1)

(body temperature dynamics of #2)

go to demo /demos/hybrid/heatcool.m

HYSDEL Model

```matlab
SYSTEM heatcool {
    INTERFACE {
        STATE { REAL T1 [-10,50];
        REAL T2 [-10,50];
    }
    INPUT { REAL Tamb [-10,50];
    }
    PARAMETER {
        REAL alpha; alpha2, k1, k2;
        REAL Thot1, Tcold1, Thot2, Tcold2, Th, Uc;
    }
    IMPLEMENTATION {
        A1 { REAL uhot, ucold;
            REAL hot1, hot2, cold1, cold2;
        }
        A2 { hot1 = T1<Thot1;
            hot2 = T2<Thot2;
            cold1 = T1<Tcold1;
            cold2 = T2<Tcold2;
        }
        A3 { uhot = (IF cold1 | (cold1 & ~hot1) THEN Uc ELSE 0);
            ucold = (IF hot1 | (hot1 & ~cold1) THEN Uc ELSE 0);
        }
        CONTINUOUS { T1 = T1+T1*(-alpha1*(T1-Tamb)+k1*(uhot-ucold));
            T2 = T2+T2*(-alpha2*(T2-Tamb)+k2*(uhot-ucold));
        }
    }
}
```

get the MLD model in Matlab

simulate the MLD model

>>S=mld('heatcoolmodel',Ts)

>>[XX,TT]=sim(S,x0,U);
Hybrid PWA Model

- **PWA model**

\[
\begin{align*}
    x(k+1) &= A_i(k)x(k) + B_i(k)u(k) + f_i(k) \\
    y(k) &= C_i(k)x(k) + D_i(k)u(k) + g_i(k) \\
    i(k) \text{ s.t. } H_i(k)x(k) + J_i(k)u(k) &\leq K_i(k)
\end{align*}
\]

- 2 continuous states: 
  (temperatures $T_1, T_2$)

- 1 continuous input: 
  (room temperature $T_{amb}$)

- 5 polyhedral regions 
  (*partition does not depend on input*)

>> P = pwa(S);

Simulation in Simulink
Verification of DHA/MLD/PWA

Verification Algorithm

**QUERY:** Is the target set $X_f$ reachable after $N$ steps from some initial state $x_0 \in X_0$ for some input profile $u \in U$?

**Computation:** Solve the mixed-integer linear program (MILP)

$$\begin{align*}
\text{min} & \quad 0 \\
\text{s.t.} & \quad x(k+1) = A x(k) + B_1 u(k) + B_2 \delta(k) + B_3 z(k) \\
& \quad E_2 \delta(k) + E_3 z(k) \leq E_1 u(k) + E_4 x(k) + E_5 \\
& \quad S_u u(k) \leq T_u \quad (u(k) \in U) \\
& \quad k = 0, 1, \ldots, N - 1 \\
& \quad S_0 x(0) \leq T_0 \quad (x(0) \in X_0) \\
& \quad S_f x(N) \leq T_f \quad (x(N) \in X_f)
\end{align*}$$

with respect to $u(0), \delta(0), z(0), \ldots, u(N-1), \delta(N-1), z(N-1), x(0)$

**Alternative solutions:**

- Exploit the special structure of the problem and use polyhedral computation. (Torrisi, 2003)

- Use abstractions (LPs) + SAT solvers (Giorgetti, Pappas, 2005)
**Verification Example**

- MLD model: room temperature system
- $X_f = \{\begin{bmatrix} T_1 \\ T_2 \end{bmatrix} : 10 \leq T_1, T_2 \leq 15\}$ (set of unsafe states)
- $X_0 = \{\begin{bmatrix} T_1 \\ T_2 \end{bmatrix} : 35 \leq T_1, T_2 \leq 40\}$ (set of initial states)
- $U = \{T_{\text{amb}} : 10 \leq T_{\text{amb}} \leq 30\}$ (set of possible inputs)
- $N=10$ (time horizon)

\[
\text{MATLAB:}
\> [\text{flag, } x_0, U] = \text{reach}(S, N, X_f, X_0, u_{\text{min}}, u_{\text{max}});
\]
Controller Synthesis

Control Strategy: MPC

- 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:

\[
\min_{u(t), \ldots, u(t+T-1)} \sum_{k=0}^{T-1} \| R(y(t+k|t) - r(t+k)) \|_p + \| Qu(t+k) \|_p
\]

subject to

\[
\begin{align*}
&M:\text{MLD or PWA model} \\
x(t|t) = x(t)
\end{align*}
\]

\[p = 1, 2, \infty \quad \|v\|_2 = v'v \quad \|v\|_\infty = \max |v_i| \quad \|v\|_1 = \sum v_i\]

- Apply only $u(t) = u^*(t)$ (discard the remaining optimal inputs);
- Repeat the whole optimization at time $t+1$
Hybrid MPC - Example

```matlab
>> refs.x=2;   % just weight state #2
>> Q.x=1;
>> Q.rho=Inf;  % hard constraints
>> Q.norm=2;   % quadratic costs
>> N=2;        % optimization horizon
>> limits.xmin=[25;-Inf];

>> C=hybcon(S,Q,N,limits,refs);

>> C
Hybrid controller based on MLD model S <heatcoolmodel.hys>
2 state measurement(s)
0 output reference(s)
0 input reference(s)
1 state reference(s)
0 reference(s) on auxiliary continuous z-variables
20 optimization variable(s) (8 continuous, 12 binary)
46 mixed-integer linear inequalities
sampling time = 0.5, MILP solver = 'glpk'
Type "struct(C)" for more details.

>> [XX,UU,DD,ZZ,TT]=sim(C,S,r,x0,Tstop);
```
On-Line vs Off-Line Optimization

\[
\begin{align*}
\min_U J(U, x(t)) & = \sum_{k=0}^{T-1} \|Rx(t+k|t)\|_p + \|Qu(t+k)\|_p \\
\text{subject to} & \begin{cases} 
\text{MLD model} \\
x(t|t) = x(t)
\end{cases}
\end{align*}
\]

• On-line optimization: given \(x(t)\) solve the problem at each time step \(t\).

Mixed-Integer Linear/Quadratic Program (MILP/MIQP)

- Good for large sampling times (e.g., 1 h) / expensive hardware ...
- ... but not for fast sampling (e.g., 10 ms) / cheap hardware!

• Off-line optimization: solve the MILP/MIQP for all \(x(t)\)

\[
\begin{align*}
\min_{\xi} J(\xi, x(t)) & = \begin{cases}
f'\xi \\
\xi' H \xi + f' \xi
\end{cases} \\
\text{s.t. } G\xi \leq W + Fx(t)
\end{align*}
\]

multi-parametric programming

Explicit Hybrid MPC

\[
\begin{align*}
\min_U J(U, x, r) & = \sum_{k=0}^{T-1} \|R(y(k) - r)\|_p + \|Qu(k)\|_p \\
\text{subject to} & \begin{cases} 
\text{PWA model} \\
x(0) = x
\end{cases}
\end{align*}
\]

• Solution \(u(x, r)\) found via a combination of
  - Dynamic programming or enumeration of feasible mode sequences, multi-parametric linear or quadratic programming, and polyhedral computation.

  (Borrelli, Baotic, Bemporad, Morari, 2003)
  (Mayne, ECC 2001)
  (Alessio, Bemporad, 2005)

• The MPC controller is piecewise affine in \(x, r\)

\[
\begin{align*}
u(x, r) & = \begin{cases}
F_1 x + E_1 r + g_1 & \text{if } H_1 \left[ \frac{x}{r} \right] \leq K_1 \\
& \vdots \\
F_M x + E_M r + g_M & \text{if } H_M \left[ \frac{x}{r} \right] \leq K_M
\end{cases}
\end{align*}
\]

Note: in the quadratic case the partition may not be fully polyhedral
Explicit Hybrid MPC - Example

```matlab
>> E = expcon(C, range, options);
```

```
>> E

Explicit controller (based on hybrid controller C)
  3 parameter(s)
  1 input(s)
  12 partition(s)

The controller is for hybrid systems (tracking)
This is a state-feedback controller.
Type "struct(E)" for more details.
```

\[
\min \sum_{k=1}^{2} x_2(k)^2
\]

s.t. \( x_1(k) \geq 25 \) for \( k = 1, 2 \)

PWA model

Section in the \((T_1, T_2)\)-space for \( T_{ref} = 30 \)

Explicit Hybrid MPC - Example

Generated C-code

utils/expcon.h
Hybrid Control Design Flow

- **physical system**
- **control objectives & constraints**
- **hybrid model**
- **simulation and validation**
- **optimal control setup (on-line MIP)**
- **simulation and validation**
- **PWA control law**
- **implementation (C code) and experiments**
- **too complex?**
- **yes or no?**

Conclusions

- **Discrete hybrid automata** are simple yet versatile models of hybrid systems, and lead immediately computationally-useful models

- **Optimization-based control** handles performance specs and constraints in a natural and direct way. Quite complex hybrid systems can be controlled using on-line mixed-integer programming

- **Piecewise affine MPC controllers** can be synthesized, off-line, and implemented as look-up tables of linear gains

\[
u(x, r) = \begin{cases} 
F_1x + E_1r + g_1 & \text{if } H_1 \left[ \frac{x}{r} \right] \leq K_1 \\
\vdots & \vdots \\
F_Mx + E_Mr + g_M & \text{if } H_M \left[ \frac{x}{r} \right] \leq K_M 
\end{cases}
\]

- **Hybrid Toolbox for Matlab** available to assist controller design: modeling, simulation, verification, MPC, code generation