Reachability and Observability Analysis of Hybrid Systems

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

Reachability Analysis

Verification

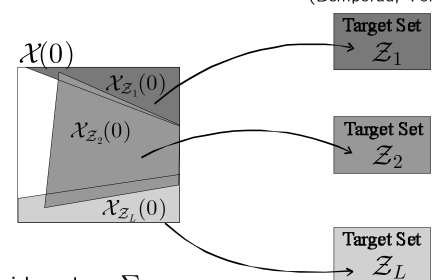
- GIVEN: an embedded system (continuous dynamical system + logic controller = hybrid system)
- CERTIFY that such combination behaves as desired
  - for ALL initial conditions within a given set
  - for ALL disturbances within a given class
- or PROVIDE a counterexample.

Simulation: provides a partial answer (not all possibilities can be tested!)

Reachability Analysis: provides the answer

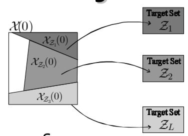
Reachability Analysis/Verification

(Bemporad, Torrisi, Morari, 2000)



- Given:
  - A hybrid system Σ
  - A set of initial conditions $\mathcal{X}(0)$
  - Target sets $\mathcal{Z}_1, \ \mathcal{Z}_2, \ \ldots, \ \mathcal{Z}_L$ (disjoint)
  - Time horizon $t \leq T_{\max}$
- Problem:
  - Is $\mathcal{Z}_i$ reachable from $\mathcal{X}(0)$ in t steps ?
  - If yes, from which subset $\mathcal{X}_{\mathcal{Z}_i}(0)$ of $\mathcal{X}(0)$?
  - Disturbance/input sequences driving $\mathcal{X}_{\mathcal{Z}_i}(0)$ to $\mathcal{Z}_i$.

Complexity of Reachability Analysis



- $T_{\max} < \infty$, discrete-time model $\longrightarrow$ Decidable !

BUT

- Mixed-integer linear feasibility test: for all $t \le T_{\max}$, for all $\mathcal{Z}_j$, $j = 1, \ldots, L$, solve:

$$\begin{array}{l} x(k+1) = A^{i(k)}x(t) + B^{i(k)}u(k) + f^{i(k)} \\ H^{i(k)}x(k) \le K^{i(k)} \\ i(k) \in \{0, \ldots, s-1\} \\ x(0) \in \mathcal{X}(0) \\ u(k) \in \mathcal{U}, \quad k = 0, \ldots, t-1 \\ x(t) \in \mathcal{Z}_j \end{array}$$

- NP-hard ! because of free integer variables $i(k)$ (worst case: s^T)

Verification Algorithm via MILP

- Simple solution: Solve $\forall T > 0$ the mixed-integer linear feasibility test

$$\left\{ \begin{array}{c} x(0) \in \mathcal{X}(0) \\ x(T) \in \mathcal{Z}_i \\ u(t) \in \mathcal{U}, 0 \le t \le T \\ x(t+1) = Ax(t) + B_1u(t) + B_2\delta(t) + B_3z(t) \\ E_2\delta(t) + E_3z(t) \le E_1u(t) + E_4x(t) + E_5 \end{array} \right.$$

with respect to $x(0), \{u(t), \delta(t), z(t)\}_{t=0}^{T}$

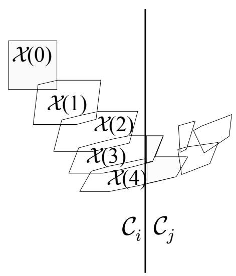
- Only practical for small problems ! because number of free integer variables $\delta(0), \delta(1), \ldots, \delta(T)$ grows with T

- **Efficient Solution**: Exploit the special structure of the problem.

(Bemporad, Torrisi, Morari, 2000)
(Torrisi, Bemporad, Giovanardi, 2003)

Reachability Analysis Algorithm

- Compute the polyhedral reach set $\mathcal{X}(t)$ (affine dynamics)
- Detect switching
- Describe new intersections $\mathcal{X}(t) \cap \mathcal{C}_j$
- Stopping criteria for a single exploration
- Organize the search



Reach Set Computation

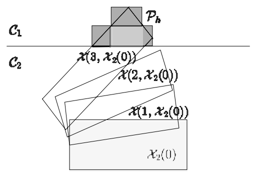
Reach set implicitly defined by linear inequalities

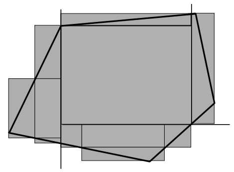
$$\left\{ \begin{array}{l} x \in \mathcal{X}(0) \\ K_i^x \left(A_i^k x + \sum_{k=0}^{t-1} A_i^j [B_i u(t-1-k) + f_i] \right) + \\ \quad + K_i^u u(t) \le H, \; k = 1, \ldots, t \\ u_{\min} \le u(k) \le u_{\max}, \; k = 0, \ldots, t-1 \end{array} \right.$$

where $\mathcal{X}_i = \{(x,u) : \; K_i^x x + K_i^u u \le H\}$ is the current region

- Simple to compute
- Number of constraints grows linearly with time
- Explicit form also possible via projection methods (e.g. CDD by K. Fukuda)

Approximation of Intersections





✓ Simple to compute via Linear Programming (LP)

✓ Can approximate with arbitrary precision

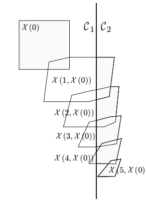
✓ Trade off between conservativeness and complexity

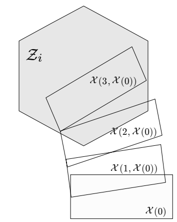
✓ Both inner and outer approximations in one shot

✓ Approximate computation of projections

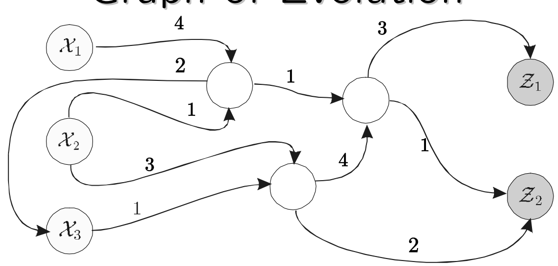
Stopping Criteria

- Reach set has left the current region
- Reach set is all inside a target $\mathcal{Z}_i$
- $t > T_{\max}$ (to guarantee termination)



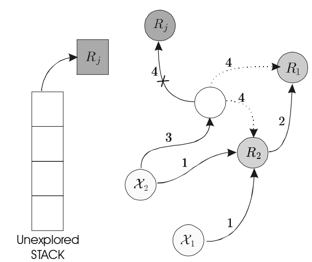


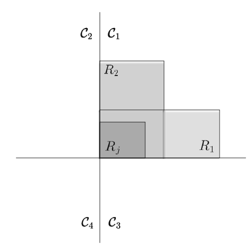
Graph of Evolution



- The **graph** is initialized with all the $\boxed{\text{initial}}$ subsets $\mathcal{X}_i \triangleq \mathcal{X}(0) \cap \mathcal{C}_i$ and all the $\boxed{\text{target}}$ sets
- A **node** is added for each initial region of a new exploration
- If a set can reach another set, an **oriented arc** is drawn
- The time needed to reach the set is a **weight** associated with the arc

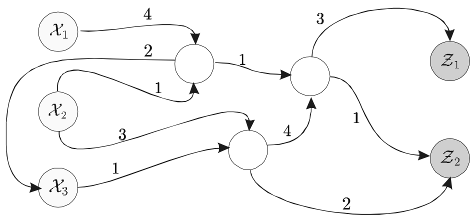
Removing Nodes





- Before starting a new exploration, if the initial set associated with the node is included in another set, the node is removed and the arcs are redirected

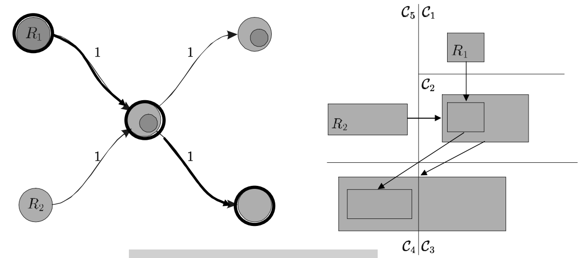
Switching Sequences



- All switching admissible switching sequences $\{i(0), i(1), \ldots, i(T)\}$ of the system are paths in the graph
- The converse is not true in general

  (the graph is only a simulation)

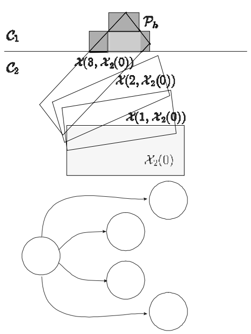
Path Refinement



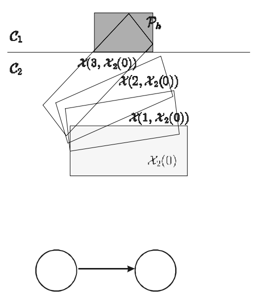
A fake path from C_1 to C_3 has appeared

- ✗ Conservativeness introduced by
  - hyper-rectangular **approximation**
  - **redirection** of arcs (for set inclusion)
- ✓ Each path of the graph can be validated by a Linear Program (LP)

Conservativeness vs Graph Complexity



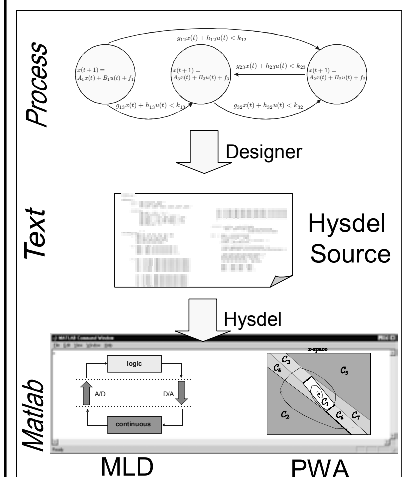
- ◦ Less conservative
- ◦ Graph more complicate



- ◦ Graph less complicate
- ◦ More conservative

LP determines if a switching sequence is feasible

Verification Algorithm: Features



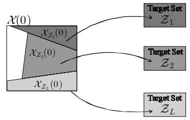
- Discrete-time, affine dynamics

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

- Logic-, threshold-, and time-based switching
- Inputs (e.g.: disturbances, references)
- Finite-time reachability analysis
- Logic-, threshold-, and time-based verification queries

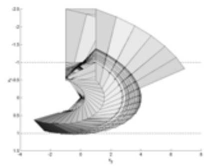
Software: shortly on the web

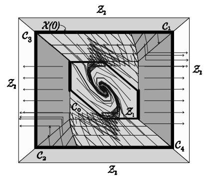
Applications

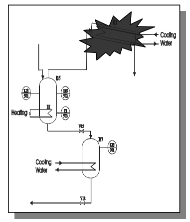


- Safety ($\mathcal{Z}_1, \ \mathcal{Z}_2, \ \ldots, \ \mathcal{Z}_L$ = unsafe sets)
- Stability ($\mathcal{Z}_1$ = invariant set around the origin)
- Optimal control ($u(0), \ldots, u(T_{\max})$ = optimal strategy,

  $\mathcal{Z}_1$ = reference set) (Bemporad, Giovanardi, Torrisi, CDC 2000)
- (practical) Liveness ($\mathcal{Z}_1$ = set to be reached within a finite time)
- Robust Simulation

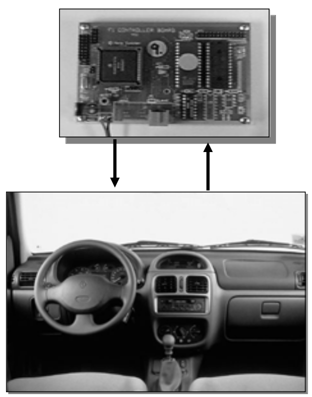






Verification Example: Cruise Control System

Cruise Control System



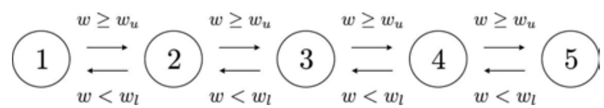
GOAL:

Verify if a given switching controller satisfies certain specifications

(Torrisi, Bemporad, 2001)

Cruise Control System

Gear selector:

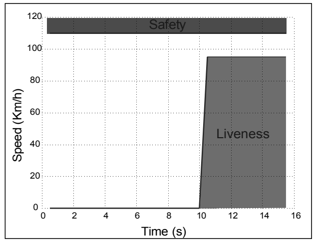


Speed controller:

$$
\begin{aligned}
e(t+1) &= e(t) + T_s(v_r(t) - v(t)) \quad (+ \text{ saturation}) \\
u_e(t) &= \begin{cases} k_e(v_e(t) - v(t)) + i_e e(t) & \text{if } v(t) < v_r + 1 \\ 0 & \text{otherwise} \end{cases} \\
u_b(t) &= \begin{cases} k_b(v_e(t) - v(t)) & \text{if } v(t) \geq v_r + 1 \\ 0 & \text{otherwise} \end{cases}
\end{aligned}
$$

Verification Problem

Question: Will the cruise control reach the desired speed reference within 10 s without exceeding the speed limit?



Safety $\mathcal{Z}_1 = \{v : v > v_r + r_{\text{toll}}\}$

Liveness $\mathcal{Z}_2 = \{v, t : v < v_r - 2r_{\text{toll}}, t > 10/T_s\}$

$r_{\text{toll}} = 5$ km/h

Hysdel Model (HYbrid Systems DEscription Language)

```
SYSTEM car {
INTERFACE {
    STATE { REAL speed, err, vr; BOOL gear1, gear2, gear3, gear4, gear5; }
    PARAMETER {…}}
IMPLEMENTATION {
    AUX {
        REAL Fe1, Fe2, Fe3, Fe4, Fe5, w1, w2, w3, w4, w5, DCe1, DCe2, DCe3, DCe4,zut, zub, ierr, torque, F_brake;
        BOOL dPWL1, dPWL2, dPWL3, dPWL4, sd, su, verr, sat_torque, sat_F_brake, no_sat;}
    LOGIC { no_sat = ~(sat_torque | sat_F_brake) & verr; }
    AD {dPWL1 = wPWL1 - (w1 + w2 + w3 + w4 + w5) <= 0; dPWL2 = wPWL2 - (w1 + w2 + w3 + w4 + w5) <= 0;
        dPWL3 = wPWL3 - (w1 + w2 + w3 + w4 + w5) <= 0; dPWL4 = wPWL4 - (w1 + w2 + w3 + w4 + w5) <= 0;

        sd = (w1 + w2 + w3 + w4 + w5) - w1 <= 0; su = wu - (w1 + w2 + w3 + w4 + w5) <= 0; verr = speed - vr - 2 <= 0;
        sat_torque = - zut + (DCe1 + DCe2 + DCe3 + DCe4) + 1 <= 0; sat_F_brake = - zub + max_brake_force <= 0;          }
    DA {Fe1 = {IF gear1 THEN torque / speed_factor * Rgear1}; Fe2 = {IF gear2 THEN torque / speed_factor * Rgear2};
        Fe3 = {IF gear3 THEN torque / speed_factor * Rgear3}; Fe4 = {IF gear4 THEN torque / speed_factor * Rgear4};
        Fe5 = {IF gear5 THEN torque / speed_factor * Rgear5};

        w1 = {IF gear1 THEN speed / speed_factor * Rgear1}; w2 = {IF gear2 THEN speed / speed_factor * Rgear2};
        w3 = {IF gear3 THEN speed / speed_factor * Rgear3}; w4 = {IF gear4 THEN speed / speed_factor * Rgear4};
        w5 = {IF gear5 THEN speed / speed_factor * Rgear5};

        DCe1 = {IF dPWL1 THEN (aPWL2) + (bPWL2) * (w1 + w2 + w3 + w4 + w5) ELSE (aPWL1) + (bPWL1) * (w1 + w2 + w3 + w4 + w5)};
        DCe2 = {IF dPWL2 THEN (aPWL3 - aPWL2) + (bPWL3 - bPWL2) * (w1 + w2 + w3 + w4 + w5)};
        DCe3 = {IF dPWL3 THEN (aPWL4 - aPWL3) + (bPWL4 - bPWL3) * (w1 + w2 + w3 + w4 + w5)};
        DCe4 = {IF dPWL4 THEN (aPWL5 - aPWL4) + (bPWL5 - bPWL4) * (w1 + w2 + w3 + w4 + w5)};

        zut = {IF verr THEN kt * (vr - speed) + it * err}; zub = {IF ~verr THEN - kb * (vr - speed) - ib * err};
        torque ={IF sat_torque THEN (DCe1 + DCe2 + DCe3 + DCe4) + 1 ELSE zut}; F_brake = {IF sat_F_brake THEN max_brake_force ELSE zub};
        ierr = {IF no_sat THEN err + Ts * (vr - speed)};}
    CONTINUOUS {
        speed = speed + Ts / mass * (Fe1 + Fe2 + Fe3 + Fe4 + Fe5 - F_brake - beta_friction * speed); err = ierr; vr = vr;}
    AUTOMATA {
        gear1 =                (gear2 & sd) | (gear1 &       ~su); gear2 = (gear1 & su) | (gear3 & sd) | (gear2 & ~sd & ~su);
        gear3 = (gear2 & su) | (gear4 & sd) | (gear3 & ~sd & ~su); gear4 = (gear3 & su) | (gear5 & sd) | (gear4 & ~sd & ~su);
        gear5 = (gear4 & su)                | (gear5 & ~sd );}
    MUST {
        -w1 <= -wemin; w1 <= wemax; -w2 <= -wemin; w2 <= wemax; -w3 <= -wemin; w3 <= wemax; -w4 <= -wemin; w4 <= wemax; -w5 <= -wemin;
        w5 <= wemax; -F_brake <= 0; F_brake <= max_brake_force; torque - (DCe1 + DCe2 + DCe3 + DCe4) - 1 <= 0;
        -((REAL gear1) + (REAL gear2) + (REAL gear3) + (REAL gear4) + (REAL gear5)) <= -0.9999;
        (REAL gear1) + (REAL gear2) + (REAL gear3) + (REAL gear4) + (REAL gear5)  <= 1.0001;
        dPWL4 -> dPWL3; dPWL4 -> dPWL2; dPWL4 -> dPWL1; dPWL3 -> dPWL2; dPWL3 -> dPWL1; dPWL2 -> dPWL1;}}
```

Hybrid Model

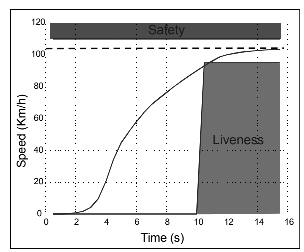
- MLD model

$$
\begin{aligned}
x'(k) &= Ax(k) + B_1u(k) + B_2\delta(k) + B_3z(k) + B_5 \\
y(k) &= Cx(k) + D_1u(k) + D_2\delta(k) + D_3z(k) + D_5 \\
E_2\delta(k) + E_3z(k) &\leq E_1u(k) + E_4x(k) + E_5
\end{aligned}
$$

- 3 continuous states: v, v_r, e (speed, reference and tracking error)
- 5 binary states: g_1, g_2, g_3, g_4, g_5 (gears)
- 19 auxiliary continuous vars: (5 traction force, 5 engine speed, 5 reset/saturation, 4 PWL max engine torque)
- 15 auxiliary binary vars: (4 PWL max torque breakpoints, 4 saturations 5 logic updates, 2 gear switching conditions)
- 173 mixed-integer inequalities

Verification Results

- For all $v_r \in [30, 70]$ km/h the controller satisfies both liveness & safety properties
- CPU time: ~2.5h (Matlab 5.3, PC650MHz)



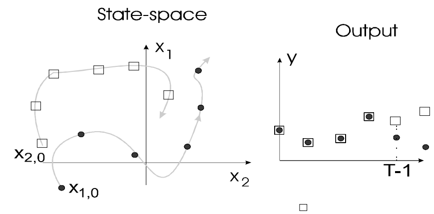
- For $v_r \in [30, 120]$ km/h the verification algorithm finds the first counterexample after ~7m

Observability Analysis and State Estimation/Fault Detection

Observability of Hybrid Systems

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

Motivation: *can we estimate states from a certain set of output measurements ?*



Complexity of Observability

Consider the PWA system:

$$\begin{aligned} x(t+1) &= A_i x(t) + B_i u(t) + f_i \\ y(t) &= C_i x(t) + g_i \end{aligned} \quad \text{for} \quad \begin{bmatrix} x(t) \\ u(t) \end{bmatrix} \in \mathcal{X}_i$$

i-th component $(A_i, B_i, C_i, f_i, g_i)$

Possible conjectures:

1. PWA systems with observable components are observable
2. PWA systems with unobservable components are unobservable

All these conjectures are false !

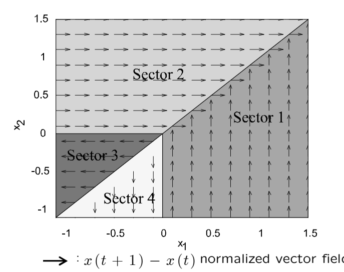
Observability is undecidable
(Sontag, 1996)

Example

An observable PWL system with unobservable components.

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}(t+1) = \begin{cases} \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}(t) \;\; if \;\; x_1(t) > x_2(t) \\ \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}(t) \;\; if \;\; x_1(t) \le x_2(t) \end{cases}$$

$$y(t) = \begin{cases} x_1(t) & \text{if } x_1(t) > x_2(t) \\ x_2(t) & \text{if } x_1(t) \le x_2(t) \end{cases}$$



$\mathcal{X}(0) \subset$ Sector 1 $\cup$ Sector 2 is observable
$\mathcal{X}(0) \subset$ Sector 3 $\cup$ Sector 4 is unobservable

Observability is not a global property in general !

Practical Observability

For any pair ($x_1(0)$,$x_2(0)$) of initial states in $\mathcal{X}(0)$, require that

$$\sum_{t=0}^{T-1} \|y_1(t) - y_2(t)\|_\infty \geq w\|x_1(0) - x_2(0)\|_1$$

whatever the input signal $u(t)$ is (within a given input set $\mathcal{U}$).

1. $w > 0$ is a sensitivity indicator ⇒ Require $w \geqslant w_{min}$
2. T is an observability index ⇒ Require $T \leq T_{max}$

Equivalently:

$$\min_{\substack{x_1(0), x_2(0) \in \mathcal{X}(0) \\ u(t) \in \mathcal{U}, t=0,\ldots,T-1}} \sum_{t=0}^{T-1} \|y_1(t) - y_2(t)\|_\infty - w\|x_1(0) - x_2(0)\|_1 \geq 0$$

Practical observability is a decidable property

Observability Algorithm #1

Goal: Compute, for $T \leq T_{max}$

$$J^* \triangleq \sum_{t=0}^{T-1} \|y_1(t) - y_2(t)\|_\infty \geq w\|x_1(0) - x_2(0)\|_1$$

w.r.t. $x_1(0)$,$x_2(0)$∈$\mathcal{X}(0)$ and u(t)∈$\mathcal{U}$, and subj. to the MLD equations + constraints.

The cost function is not convex !

Idea: The 1-norm is a PWL function and it can be represented via mixed integer linear inequalities

The ∞-norm can be represented via linear inequalities

J^* can be computed by solving a Mixed Integer Linear Program

Only suitable for relatively small T
(because the number of free integer variables grows linearly with T)

Observability Algorithm #2

Alternative approach:

Use a reachability analysis algorithm to verify that $J^* \geq 0$ for all initial conditions

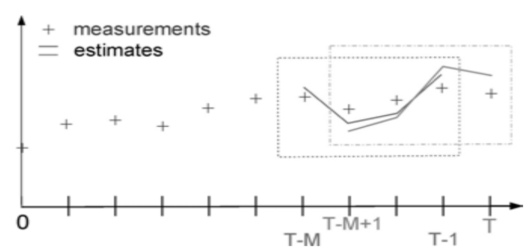
(reachability analysis is not propagated from sets where $J^* \geq 0$)

Computationally very efficient also for large T
(complexity depends on number of possible switches over the horizon T)

State Estimation / Fault Detection

- Problem: given past output measurements and inputs, estimate the current states of the hybrid systems (including discrete states and 0/1 faults)
- Idea: Use Moving Horizon Estimation ideas on the MLD model (modified w/ disturbances). This is the (almost) dual of MPC

(Rao, Rawlings, Lee, *Automatica* 2001)



MLD Systems w/ Disturbances

• Mixed logic dynamic fault (MLDF) form: (Bemporad, Mignone, Morari, ACC 99)

$$\begin{aligned} x(t+1) &= Ax(t) + B_1u(t) + B_2\delta(t) + B_3z(t) + B_6\phi(t) + \xi(t) \\ y(t) &= Cx(t) + D_1u(t) + D_2\delta(t) + D_3z(t) + D_6\phi(t) + \zeta(t) \\ E_2\delta(t) + E_3z(t) &\le E_1u(t) + E_4x(t) + E_5 \quad + E_6\phi(t) \end{aligned}$$

• Faults: $\phi \in \{0,1\}^{n_f}$ = unknown binary disturbances

• Disturbances: $\xi \in \mathbf{R}^n$, $\zeta \in \mathbf{R}^p$

• Goal: obtain estimates $\hat{\phi}(t), \hat{x}(t)$ at each time t

Hybrid Moving Horizon Estimation

(Bemporad, Mignone, Morari, ACC 99)

• At time t, solve the optimization problem

$$\begin{array}{ll} \min & \sum_{k=1}^{T} \|\hat{y}(t-k|t) - y(t-k)\|^2 + \dots \\ \text{s.t.} & \text{MLDF dynamics} \end{array}$$

with respect to $\hat{x}(t-T|t)$, $\delta(t-T)$, $\dots\delta(t-1)$,
$z(t-T)$, $\dots z(t-1)$, $\phi(t-T)$, $\dots \phi(t-1)$
$\xi(t-T)$, $\dots\xi(t-1)$, $\zeta(t-T)$, $\dots\zeta(t-1)$.

• Compute the estimate $\hat{x}(t)$, $\phi(t)$

• Set $t \leftarrow t+1$ and repeat

Hybrid Moving Horizon Estimation

• **Complexity**: at each time step we must solve an MIQP
with respect to $\hat{x}(t-T|t)$, $\delta(t-T)$, $\dots\delta(t-1)$,
$z(t-T)$, $\dots z(t-1)$, $\phi(t-T)$, $\dots \phi(t-1)$
$\xi(t-T)$, $\dots\xi(t-1)$, $\zeta(t-T)$, $\dots\zeta(t-1)$.

• **Choice of T**: related to observability properties

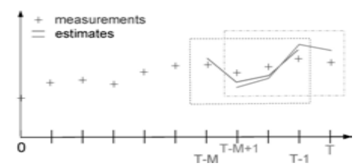
• **Convergence**: can be proved for state estimation problems using proper quadratic penalties on $\hat{x}(t-T|t)$

(Ferrari-T., Mignone, Morari, 2002)

State Estimation / Fault Detection

• Problem: given past output measurements and inputs, estimate the current state/faults

• Solution: Use Moving Horizon Estimation for MLD systems (dual of MPC)



Augment the MLD model with:

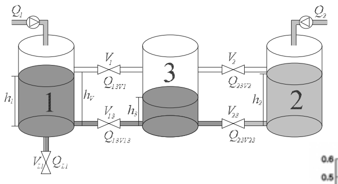
• Input disturbances $\xi \in \mathbf{R}^n$
• Output disturbances $\zeta \in \mathbf{R}^p$

At each time t solve the optimization: $\boxed{\min \sum_{k=1}^{T} \|\hat{y}(t-k|t) - y(t-k)\|^2 + \dots}$ and get estimates $\hat{x}(t)$

➡ MHE optimization = MIQP (Bemporad, Mignone, Morari, ACC 1999)
➡ Convergence can be guaranteed (Ferrari-T., Mignone, Morari, 2002)

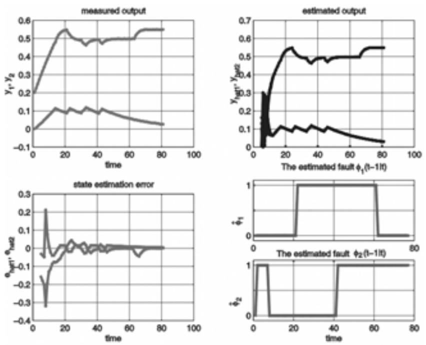
Fault detection: augment MLD with unknown **binary** distubances $\phi \in \{0,1\}^p$

Example: Three Tank System

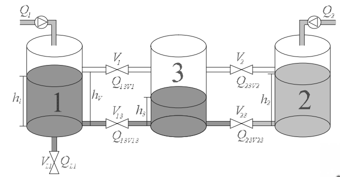


COSY Benchmark problem, ESF

- ϕ_1 : leak in tank 1
  for $20s \le t \le 60s$
- ϕ_2 : valve V_1 blocked
  for $t \ge 40s$



Example: Three Tank System



COSY Benchmark problem, ESF

- ϕ_1 : leak in tank 1
  for $20s \le t \le 60s$
- ϕ_2 : valve V_1 blocked
  for $t \ge 40s$
- Add logic constraint

  $[h_1 \le h_v] \Rightarrow \phi_2 = 0$