ESE601 - Hybrid Systems
Hybrid System Models

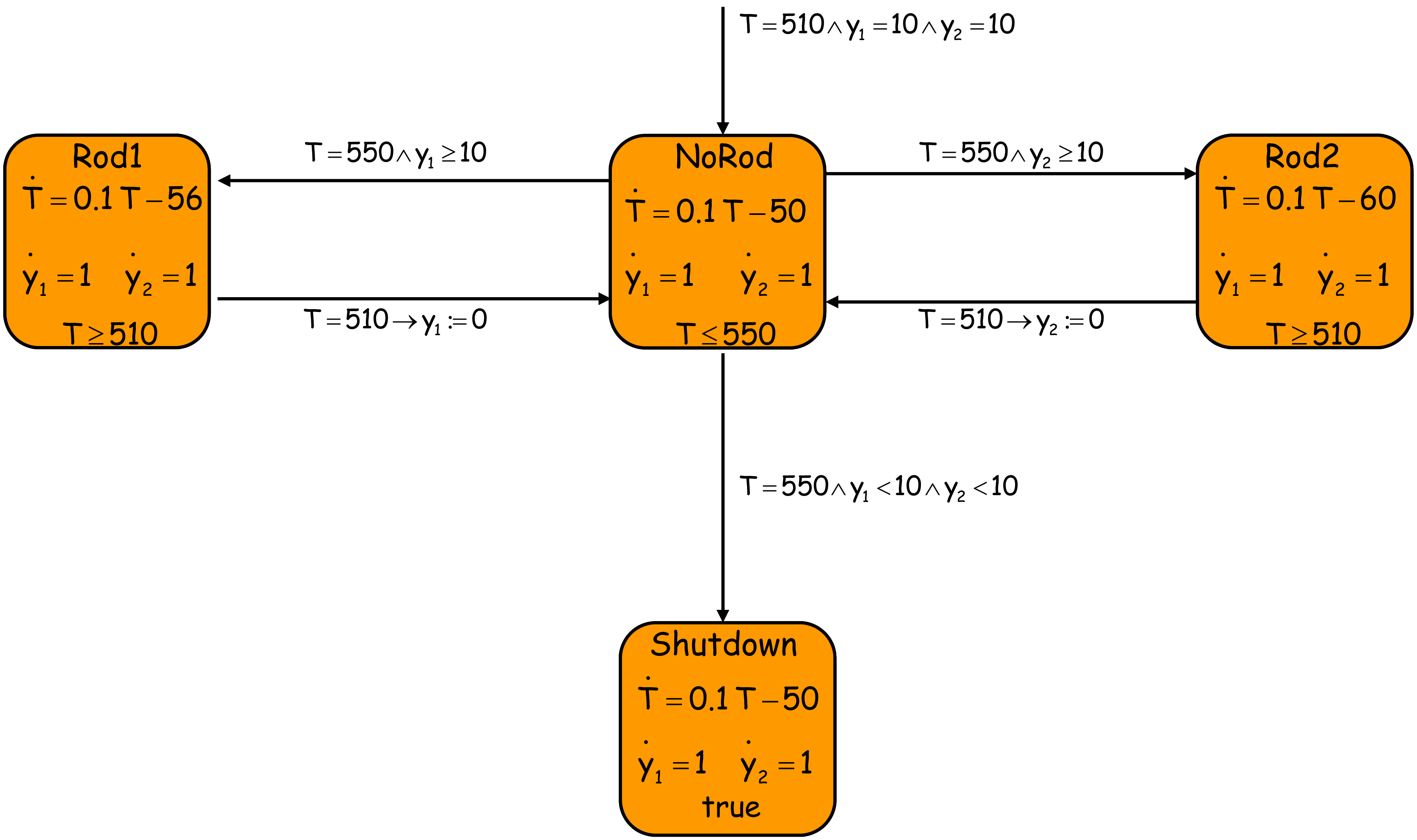
George J. Pappas

Department of Electrical and Systems Engineering

University of Pennsylvania

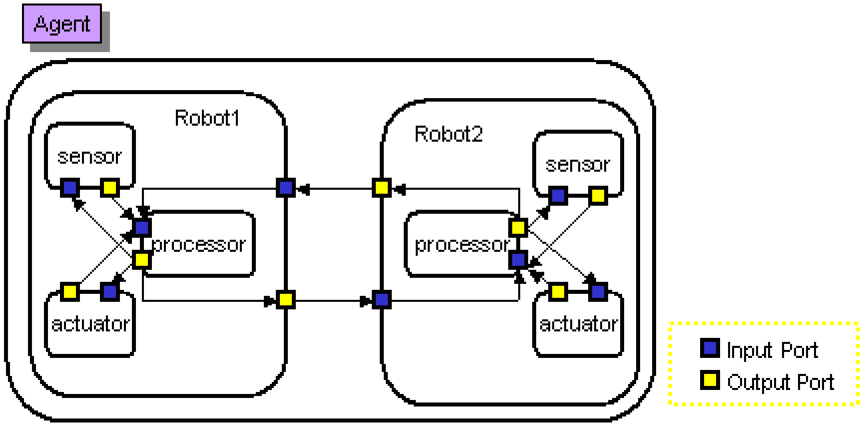
pappasg@seas.upenn.edu

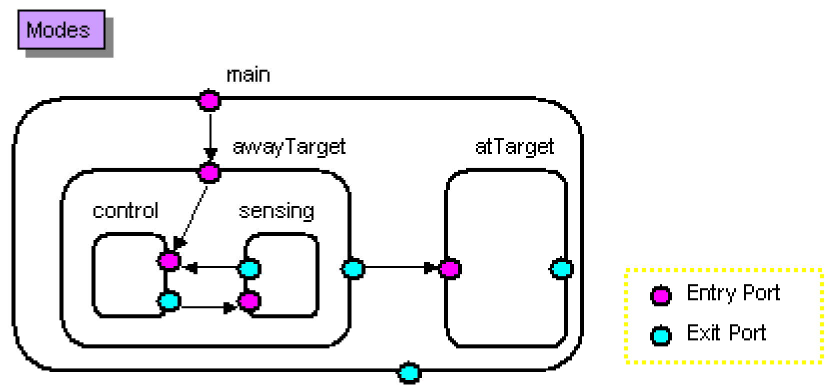
Hybrid automata



Penn University of Pennsylvania

CHARON

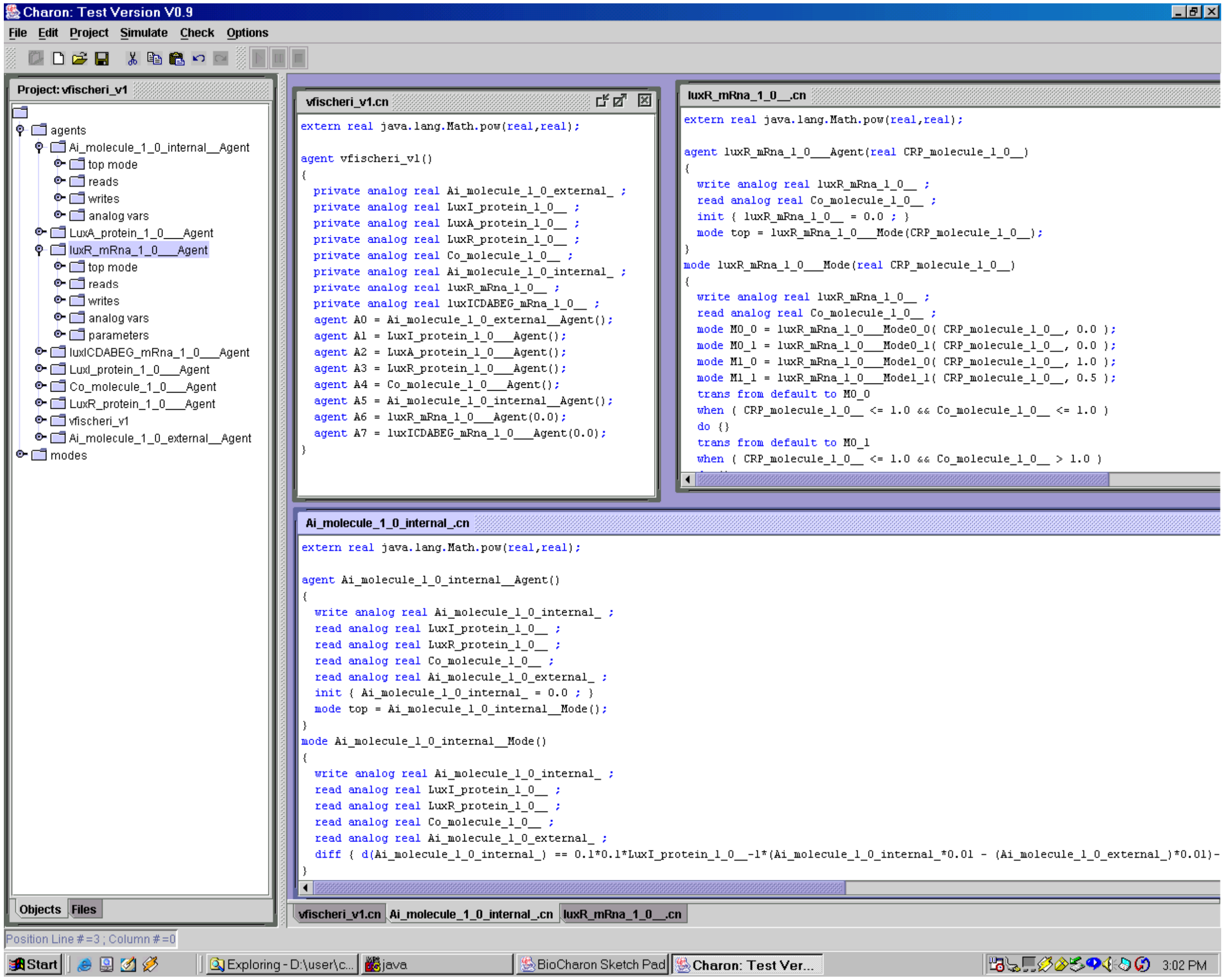


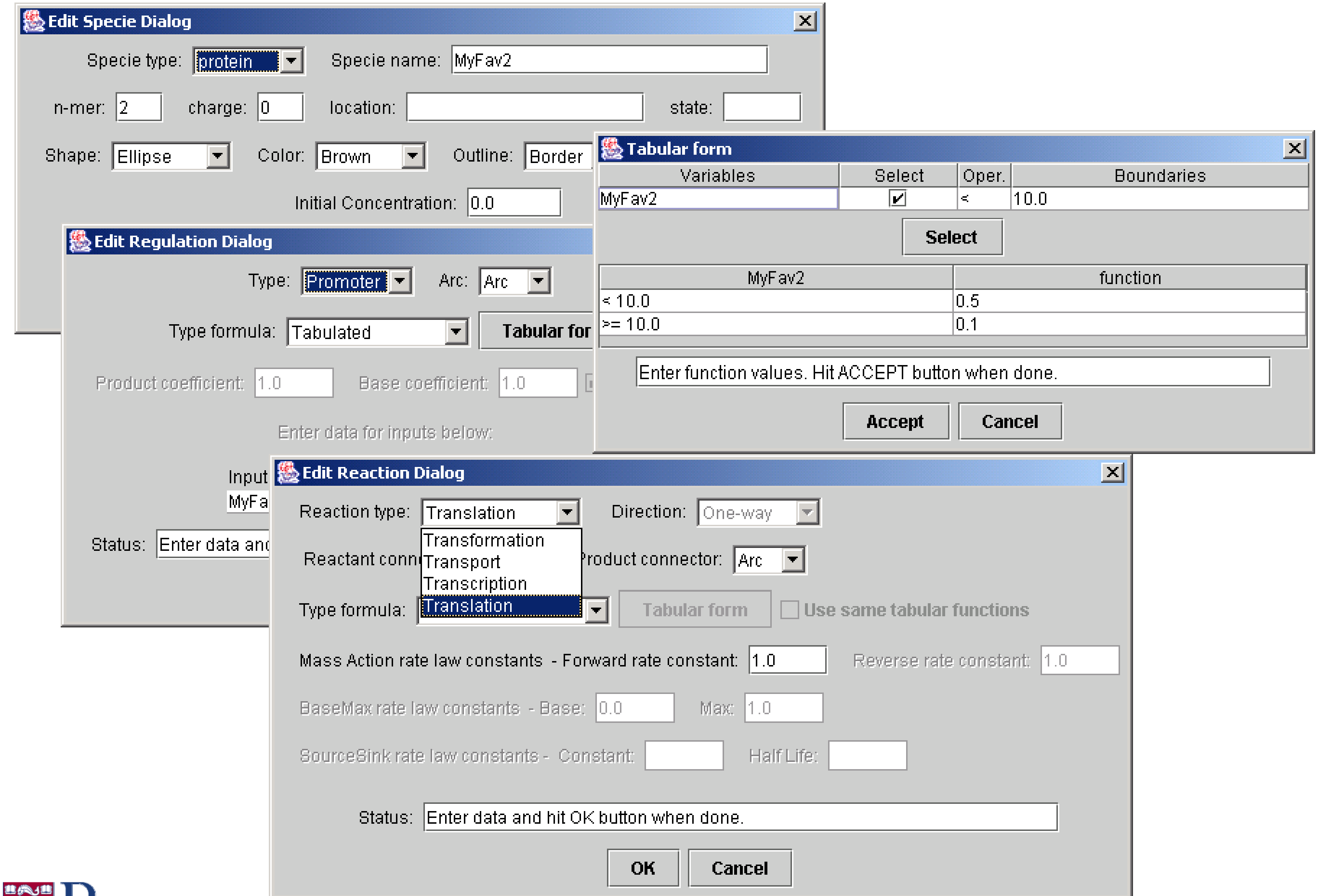


Hierarchical modeling and analysis of embedded systems
R. Alur, T. Dang, J. Esposito, Y. Hur, F. Ivancic, V. Kumar, I. Lee, P. Mishra, G. J. Pappas, and O. Sokolsky. Proceedings of the IEEE, 91(1):11-28, January 2003.

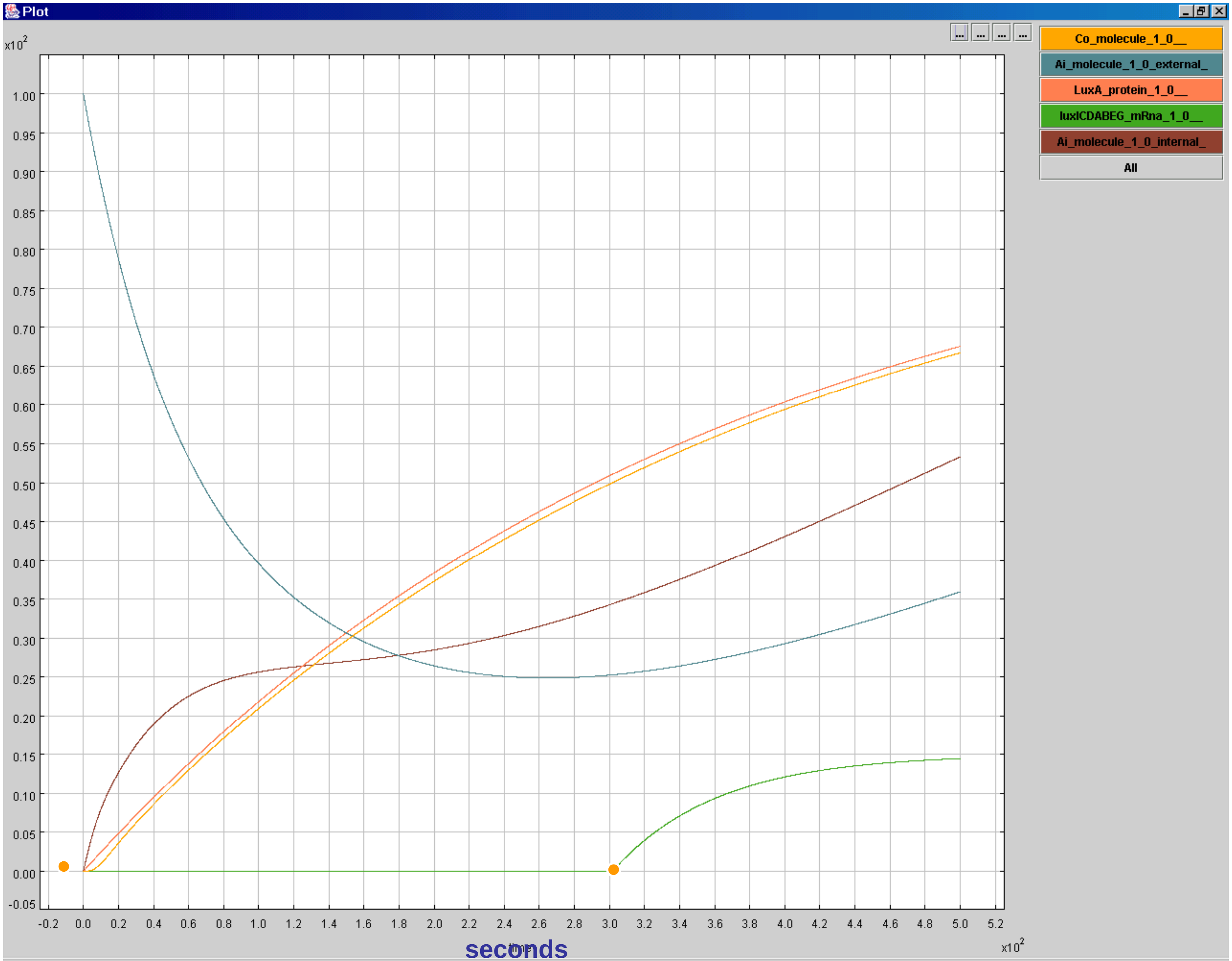
http://rtg.cis.upenn.edu/mobies/charon/index.html

CHARON


Charon: Test Version V0.9
File Edit Project Simulate Check Options
Project: vfischeri_v1
vfischeri_v1.cn
luxR_mRna_1_0__.cn
Ai_molecule_1_0_internal_.cn
extern real java.lang.Math.pow(real,real);
agent vfischeri_v1()
agent luxR_mRna_1_0___Agent(real CRP_molecule_1_0__)
mode luxR_mRna_1_0___Mode(real CRP_molecule_1_0__)
agent Ai_molecule_1_0_internal__Agent()
mode Ai_molecule_1_0_internal__Mode()
Objects Files


Edit Specie Dialog
Specie type: protein
Specie name: MyFav2
n-mer: 2
charge: 0
location:
state:
Shape: Ellipse
Color: Brown
Outline: Border
Initial Concentration: 0.0
Edit Regulation Dialog
Type: Promoter
Arc: Arc
Type formula: Tabulated
Tabular for
Product coefficient: 1.0
Base coefficient: 1.0
Enter data for inputs below:
Input
MyFa
Status: Enter data and
Tabular form
Variables
Select
Oper.
Boundaries
MyFav2
<
10.0
Select
MyFav2
function
< 10.0
0.5
>= 10.0
0.1
Enter function values. Hit ACCEPT button when done.
Accept
Cancel
Edit Reaction Dialog
Reaction type: Translation
Direction: One-way
Transformation
Transport
Transcription
Translation
Reactant conn
Product connector: Arc
Type formula:
Tabular form
Use same tabular functions
Mass Action rate law constants - Forward rate constant: 1.0
Reverse rate constant: 1.0
BaseMax rate law constants - Base: 0.0
Max: 1.0
SourceSink rate law constants - Constant:
Half Life:
Status: Enter data and hit OK button when done.
OK
Cancel

Concentration in nM

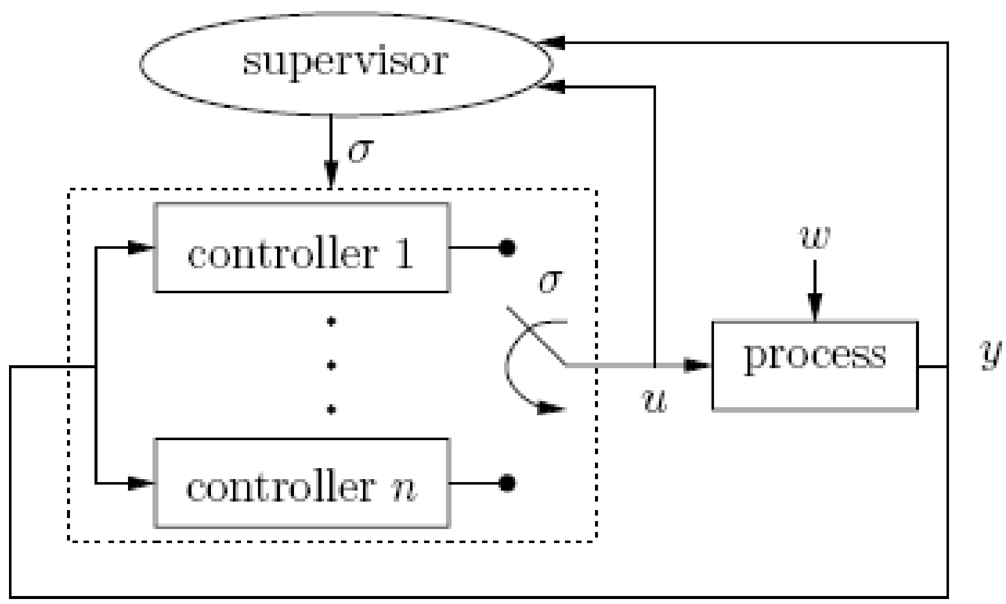


A zoo of hybrid systems

Hybrid Automata
Hybrid Input-Output Automata
Hybrid Petri Nets
Simulink/Stateflow MATLAB models

Supervisory control systems
Switched systems
Nonsmooth systems
Piece-wise affine systems (PWA)
Mixed Logical Dynamical
Linear Complementarity models

Supervisory Control Systems



Switched Control Systems

parameterized family of vector fields $\equiv$ $f_p : \mathbb{R}^n \to \mathbb{R}^n$ $\qquad p \in Q$

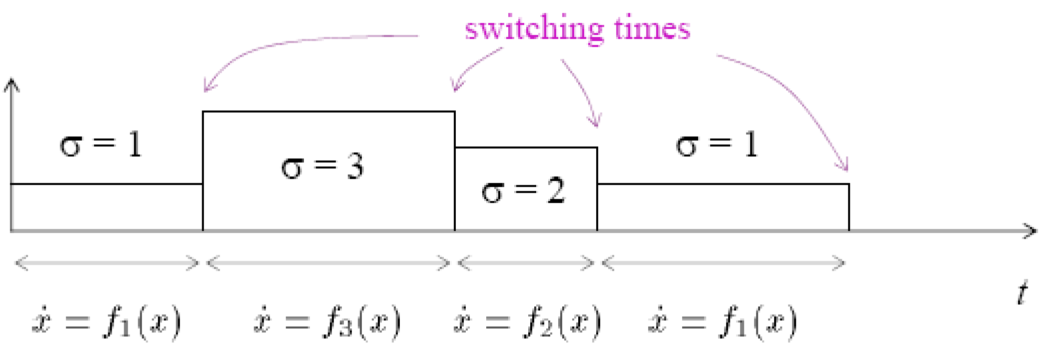
switching signal $\equiv$ piecewise constant signal $\sigma : [0,\infty) \to Q$ $\qquad$ parameter set

$\mathcal{S} \equiv$ set of admissible pairs (σ, x) with σ a switching signal and x a signal in $\mathbb{R}^n$

E.g., $\mathcal{S} := \{(\sigma, x) : N_\sigma(\tau, t) \leq 1 + \sup_{s\in(\tau,t)} \|x(s)\|\,(t-\tau),\ \forall\, t > \tau \geq 0 \}$

$$\dot{x} = f_\sigma(x) \qquad (\sigma, x) \in \mathcal{S}$$

for each x only some σ may be admissible



A ***solution*** to the switched system is a pair $(\sigma, x) \in \mathcal{S}$ for which x is a solution to

$$\dot{x} = f_{\sigma(t)}(x)$$

time-varying ODE

Penn
University of Pennsylvania

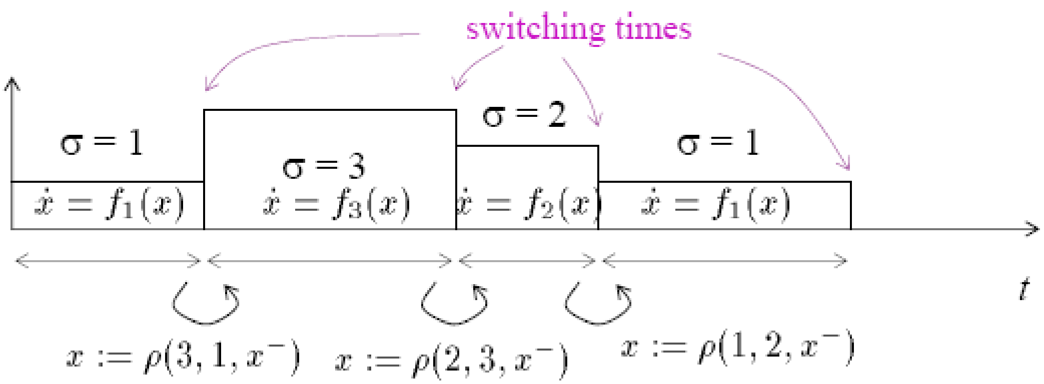
Switched Control Systems with resets

parameterized family of vector fields $\equiv$ $f_p : \mathbb{R}^n \to \mathbb{R}^n$ $\quad p \in Q$

switching signal $\equiv$ piecewise constant signal $\sigma : [0,\infty) \to Q$ $\quad$ parameter set

$\mathcal{S} \equiv$ set of admissible pairs (σ, x) with σ a switching signal and x a signal in $\mathbb{R}^n$

$$\dot{x} = f_\sigma(x) \qquad x = \rho(\sigma, \sigma^-, x^-) \qquad (\sigma, x) \in \mathcal{S}$$



A ***solution*** to the switched system is a pair $(\sigma, x) \in \mathcal{S}$ for which

1. on every open interval on which σ is constant, x is a solution to

$$\dot{x} = f_{\sigma(t)}(x)$$

time-varying ODE

2. at every switching time t, $x(t) = \rho\big(\sigma(t), \sigma^-(t), x^-(t)\big)$

Switched Control Systems

Time-varying system $\equiv$ for each initial condition $x(0)$ there is only one solution

$$\dot{x} = f_{\sigma(t)}(x) \qquad \text{(all } f_p \text{ locally Lipschitz)}$$

Switched system $\equiv$ for each $x(0)$ there may be several solutions, one for each admissible σ

$$\dot{x} = f_\sigma(x) \qquad x = \rho(\sigma, \sigma^-, x^-) \qquad (\sigma, x) \in \mathcal{S}$$

the notions of stability, convergence, etc.
must address "uniformity" over all solutions

Switched Control Systems

$$\dot{x} = \sigma x$$

$\mathcal{S} \equiv$ set of piecewise constant switching signals taking values in $Q := \{-1, +1\}$
unstable

$\mathcal{S} \equiv$ set of piecewise constant switching signals taking values in $Q := \{-1, 0\}$
stable but not asympt.

$\mathcal{S} \equiv$ set of piecewise constant switching signals taking values in $Q := \{-1, 0\}$
with infinitely many switches
stable but not asympt.

$\mathcal{S} \equiv$ set of piecewise constant switching signals taking values in $Q := \{-1, 0\}$
with infinitely many switches and interval between consecutive
discontinuities bounded below by 1 asympt. stable

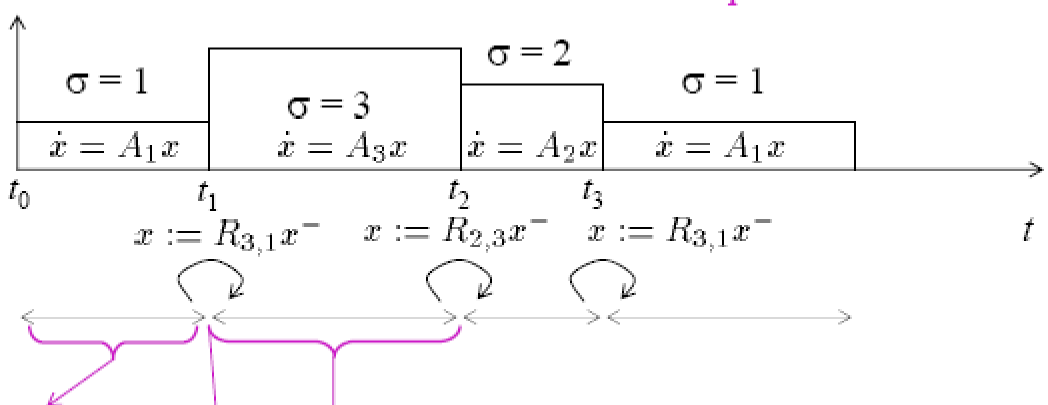
$\mathcal{S} \equiv$ set of piecewise constant switching signals taking values in $Q := \{-1, 0\}$
with infinitely many switches and interval between consecutive
discontinuities below by 1 and above by 2 uniformly asympt. stable

Penn
University of Pennsylvania

Switched Linear Systems

$$\dot{x} = A_\sigma x \qquad x = R_{\sigma,\sigma^-} x^- \qquad (\sigma, x) \in \mathcal{S} \qquad A_q, R_{q,q'} \in \mathbb{R}^{n\times n} \quad q,q' \in Q$$

vector fields and reset maps linear on x



$$x(t) = e^{A_1(t-t_0)} x(t_0)$$

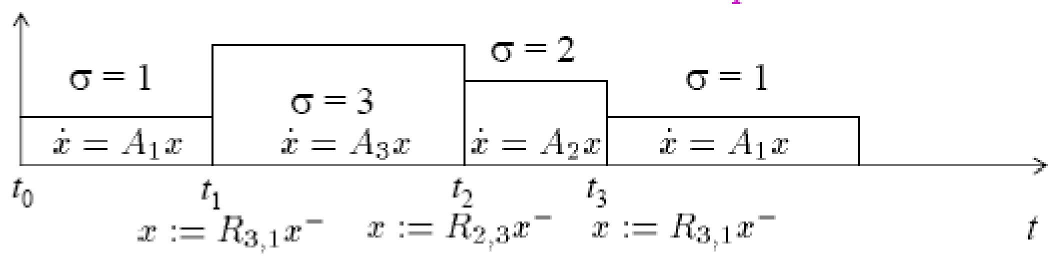
$$x(t_1) = R_{3,1} x^-(t_1) = R_{3,1} e^{A_1(t_1-t_0)} x(t_0)$$

$$x(t) = e^{A_3(t-t_1)} x(t_1) = e^{A_3(t-t_1)} R_{3,1} e^{A_1(t_1-t_0)} x(t_0)$$

Switched Linear Systems

$\dot{x} = A_\sigma x \qquad x = R_{\sigma,\sigma^-} x^- \qquad (\sigma, x) \in \mathcal{S} \qquad A_q, R_{q,q'} \in \mathbb{R}^{n\times n} \quad q, q' \in Q$

vector fields and reset maps linear on x



$$x(t) = \Phi_\sigma(t, \tau) x(\tau)$$

state-transition matrix for the switched system (σ-dependent)

$$\Phi_\sigma(t,\tau) := e^{A_{\sigma(t_k)}(t-t_k)} R_{\sigma(t_k),\sigma(t_{k-1})} e^{A_{\sigma(t_{k-1})}(t_k - t_{k-1})} \cdots$$
$$\cdots R_{\sigma(t_2),\sigma(t_1)} e^{A_{\sigma(\tau)}(t_1-\tau)} \qquad t \geq \tau$$

$t_1, t_2, t_3, \ldots, t_k \equiv$ switching times of σ in the interval $[t, \tau)$

Switched Linear Systems

$$\dot{x} = A_\sigma x \qquad x = R_{\sigma,\sigma^-} x^- \qquad (\sigma, x) \in \mathcal{S} \qquad A_q, R_{q,q'} \in \mathbb{R}^{n\times n} \quad q,q' \in Q$$

$$x(t) = \Phi_\sigma(t,\tau)x(\tau)$$

state-transition matrix (σ-dependent)

$$\Phi_\sigma(t,\tau) := e^{A_{\sigma(t_k)}(t-t_k)} R_{\sigma(t_k),\sigma(t_{k-1})} e^{A_{\sigma(t_{k-1})}(t_k - t_{k-1})} \cdots$$

$$\cdots R_{\sigma(t_1),\sigma(\tau)} e^{A_{\sigma(\tau)}(t_1-\tau)} \qquad t \geq \tau$$

$t_1, t_2, t_3, \ldots, t_k \equiv$ switching times of σ in the interval $[t, \tau)$

Analogous to what happens for (unswitched) linear systems:

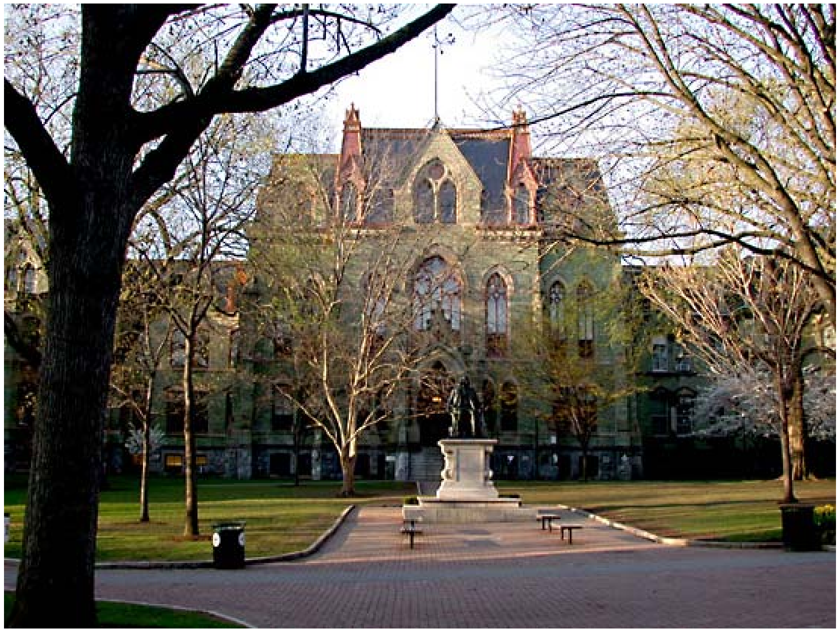
1. $\Phi_\sigma(\tau,\tau) = I \quad \forall\, \tau$
2. $\Phi_\sigma(t,s)\,\Phi_\sigma(s,\tau) = \Phi_\sigma(t,\tau) \quad \forall\, t \geq s \geq \tau$ (semi-group property)
3. if t is not a switching time, $\Phi_\sigma(t,\tau)$ is differentiable at t and
$$\frac{\mathrm{d}}{\mathrm{d}t}\Phi_\sigma(t,\tau) = A_{\sigma(t)}\Phi_\sigma(t,\tau)$$
4. if t is a switching time,
$$\Phi_\sigma(t,\tau) = R_{\sigma(t),\sigma^-(t)}\Phi_\sigma^-(t,\tau)$$

for a given σ, Φ_σ is a "solution" to the switched system with resets

Two major switching types

Time-triggered:	Switching depends on time only Switching and dynamics are decoupled Switching times are known a priori Switched systems more appropriate
Event-triggered:	Switching also depends on state Switching and dynamics are coupled Switching times not known a priori Hybrid automata more appropriate Guards/resets etc model coupling

Time-Triggered Implementations of Dynamic Controllers



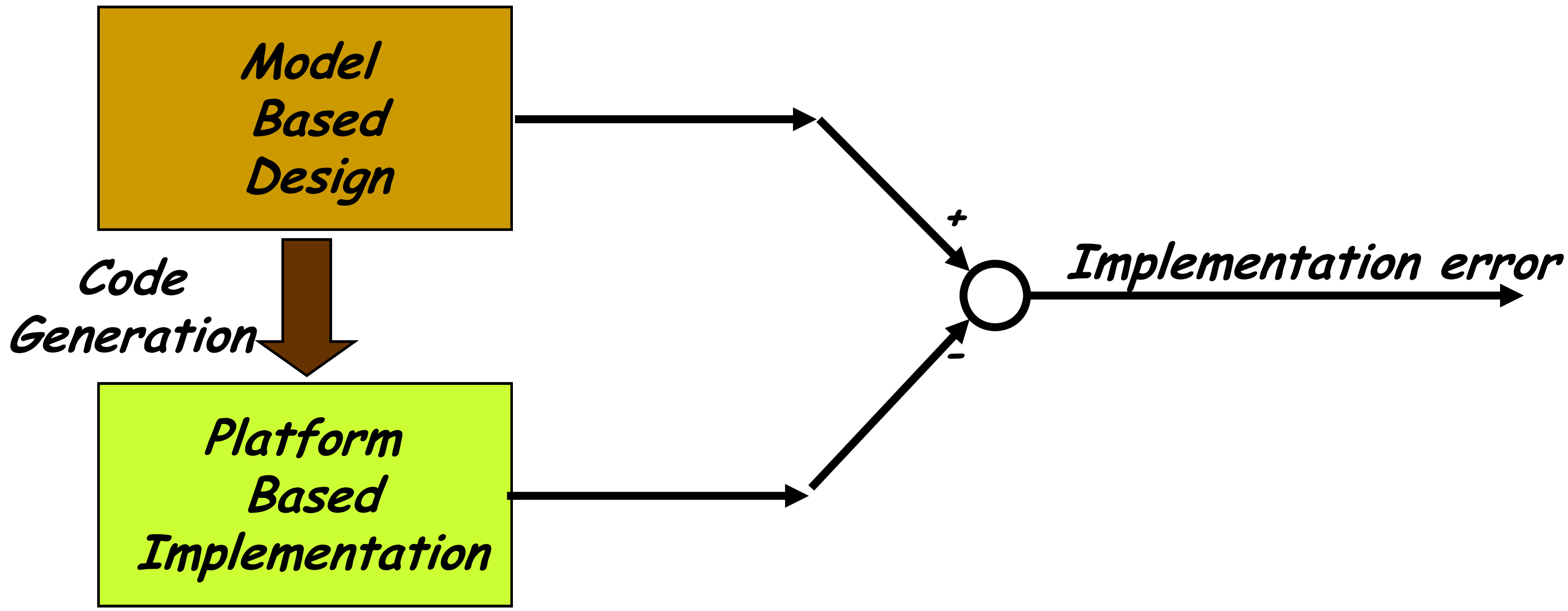
Truong Nghiem, **George J. Pappas**, Antoine Girard* and Rajeev Alur

*Universite Joseph Fourier, Grenoble, France

University of Pennsylvania, Philadelphia, U.S.A.

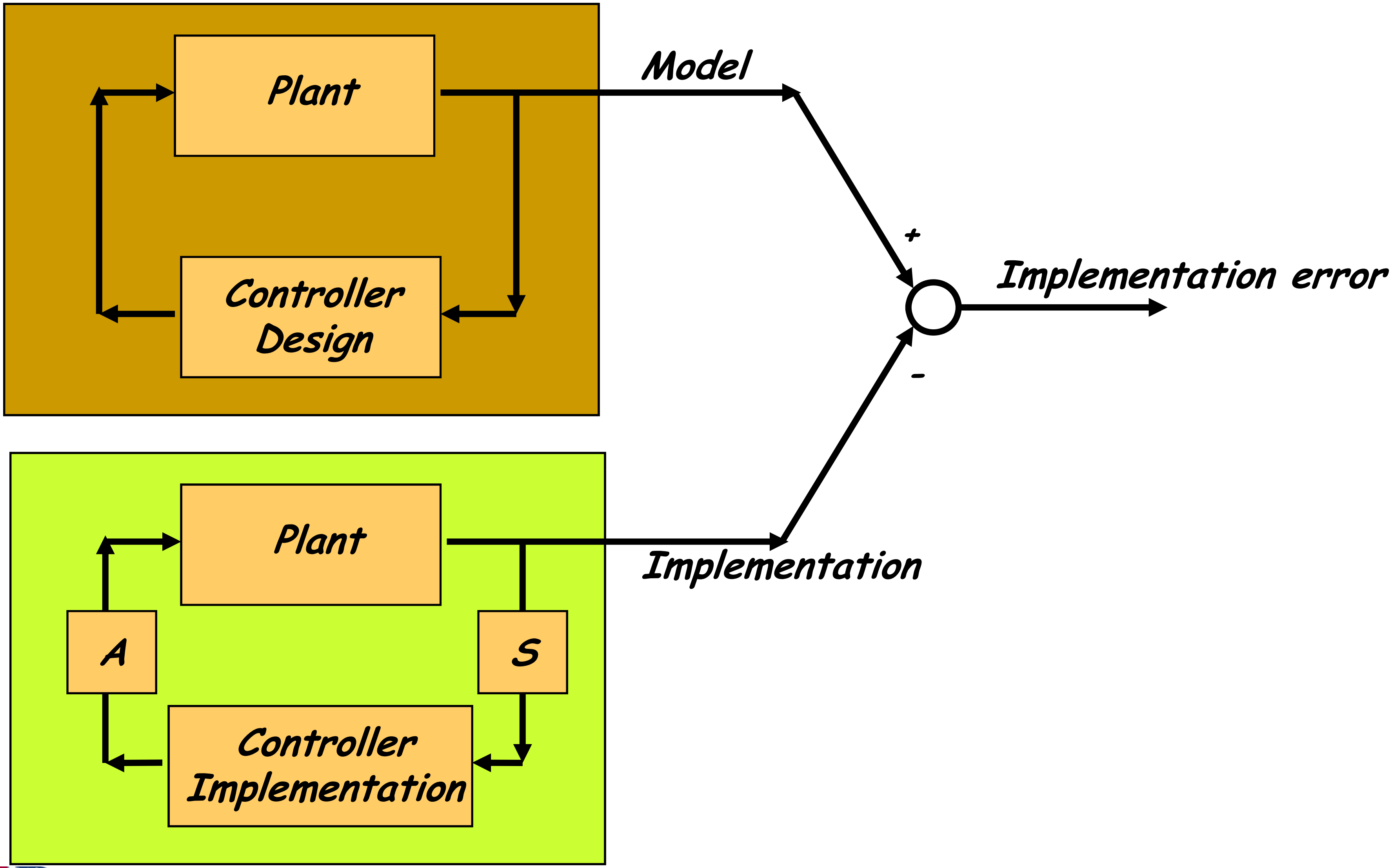
Motivation

Context : Model-based design, platform-based implementation
Problem : Relationship between model and implementation properties



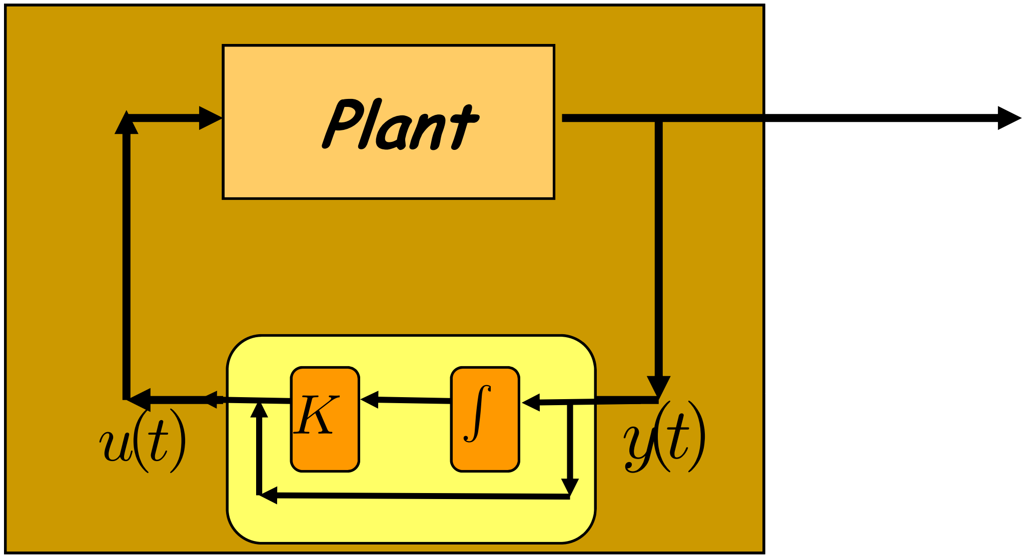
Goal : Formalize and quantify the implementation error
Focus : Feedback control designs over time-triggered platforms

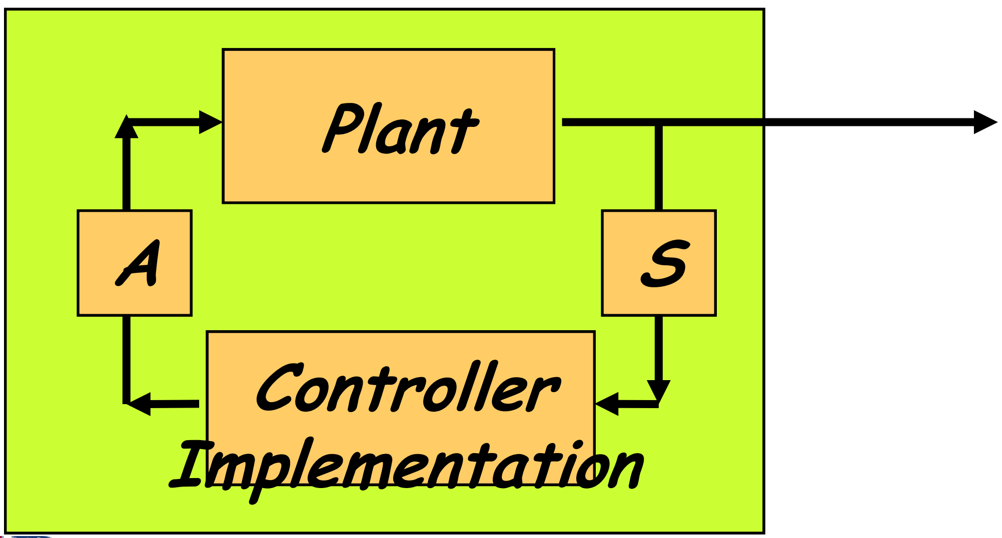
Embedded Control Design



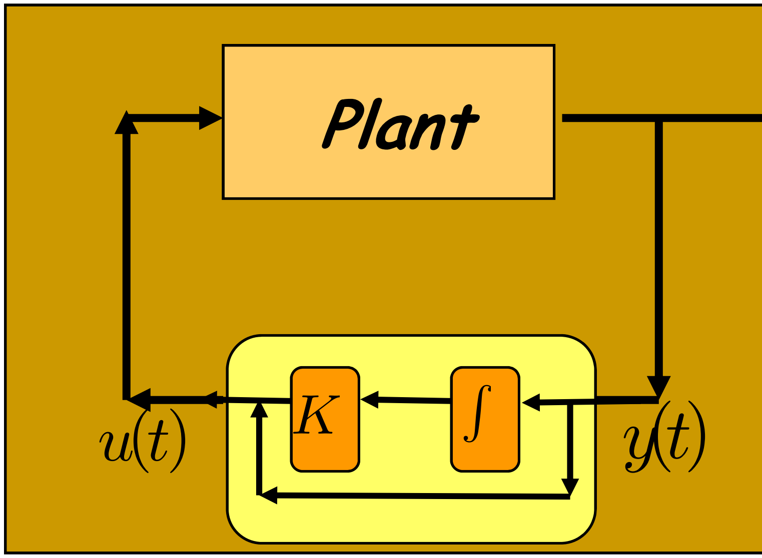
Paul Caspi and Oded Maler. From Control Loops to Real-Time Programs, 2005

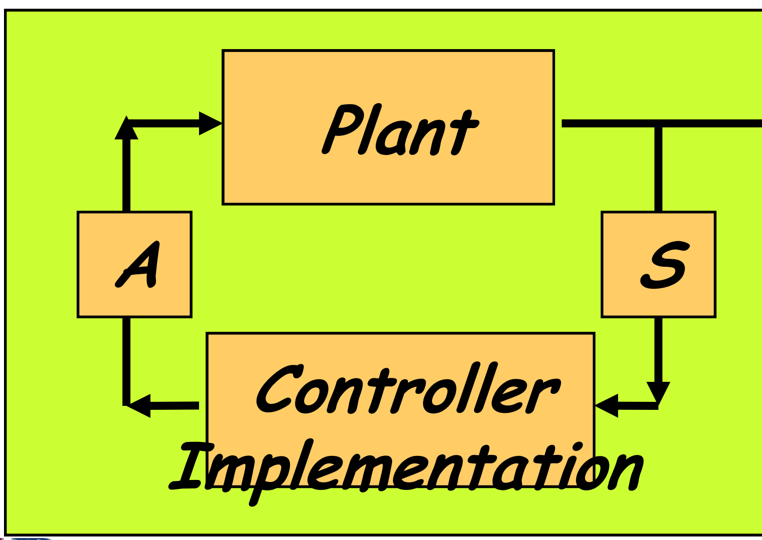
Embedded Control Design





Embedded Control Design

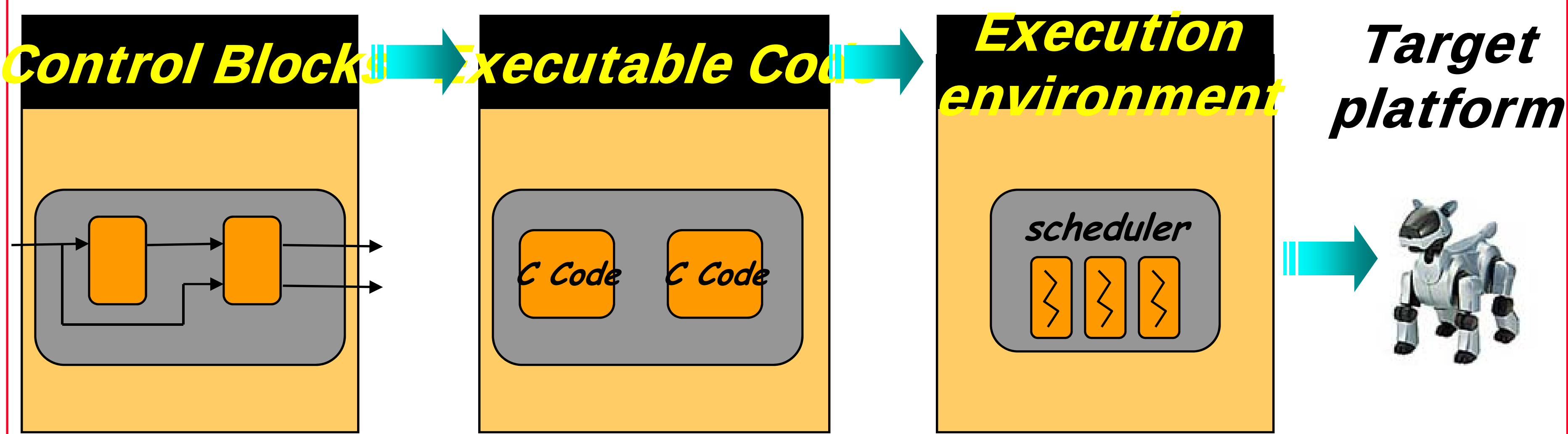





```
float updatePI(pi_block* ppi, float v);

/* Integrator Block I */
void integrator() {
    double in1, in2;          /* Two inputs */
    double curTime;           /* Current time */
    double deltaT;

    curTime = getTime();

    in1 = Input(1);           /* Read input 1 */
    in2 = Input(2);           /* Read input 2 */

    deltaT = curTime - prevTime;
    prevTime = curTime;

    x1 += deltaT*0.5*(in1 + prevIn1);
    x2 += deltaT*0.5*(in2 + prevIn2);

    prevIn1 = in1;
    prevIn2 = in2;
}

/* Propotional Block P code */
void block1() {
    double in1, out1;     /* Input & output */

    in1 = Input(1);       /* Read input 1 */

    /* Compute the output */
    out1 = 116.0*in1 + 480.0*x1;

    Output(1, out1);      /* Write output 1 */
}
```

Typical Controller Implementation

Control design is expressed using control blocks



Control designer specifies periods for control tasks

Real-time scheduling determines WCET and schedules

Real-time Scheduling

Advantages

- Offers separation of concerns between control and scheduling
- Abstracts real-time tasks with periods and deadlines

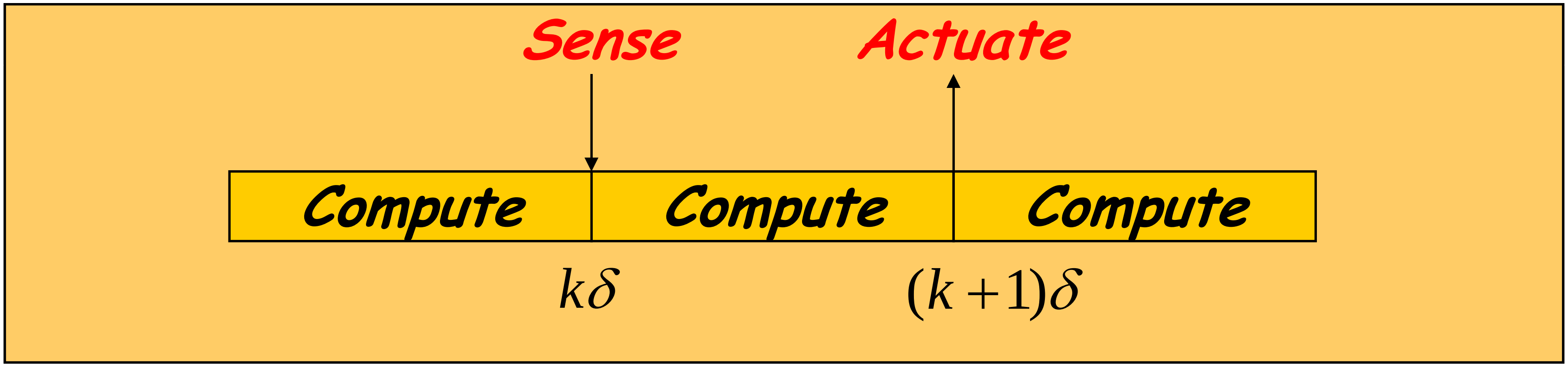
Challenges

- Real-time scheduler only guarantees that the control blocks will get a chance to complete execution once during its period
- No guarantees regarding when a control block actually reads its inputs, writes its outputs, and the order in which the various control blocks execute.
- Difficult to predict ordering impedes implementation (and therefore) error modeling, quantification, and analysis.

Time-triggered Platforms

Offers opportunities for a more predictable mapping of control models to real-time code

Instead of mapping control blocks to periodic-tasks, the compiler can allocate control blocks to precise time-slots



Advantage : Precise implementation semantics

Programming for Real-time Control

Synchronous reactive programming

Esterel, Lustre, Simulink-to-Lustre compilers

Fixed-logical execution time

Giotto

Time determinism

- Sensor readings, computation, actuation time are exactly known
- Leads to predictability (at expense of performance)

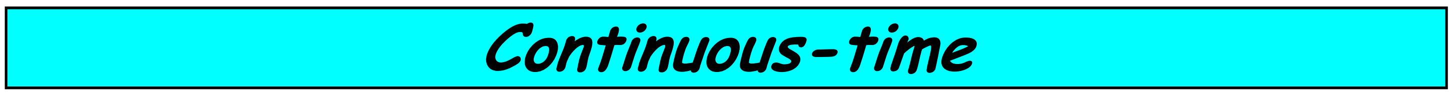
A mapping of all the control blocks to the time slots

- Can precisely define the trajectories of the implementation

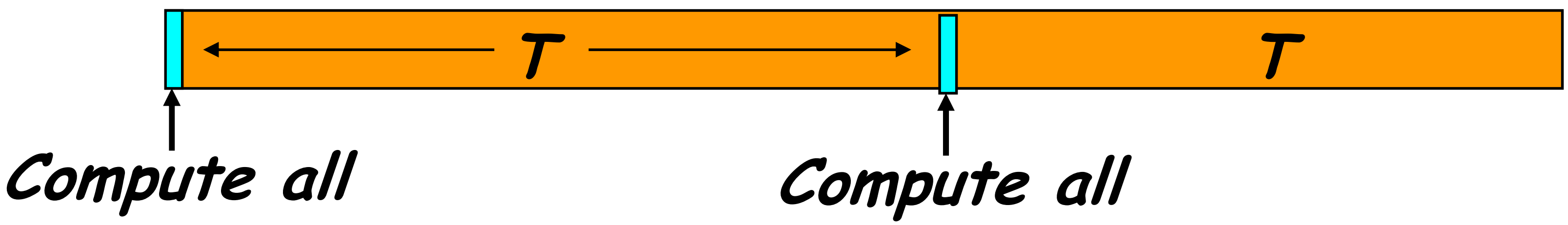
Penn
University of Pennsylvania

Digital Control

Continuous-time control : Control tasks execute and communicate instantaneously at every time point



Discrete-time control : Control tasks execute instantaneously at fixed (periodic) discrete points



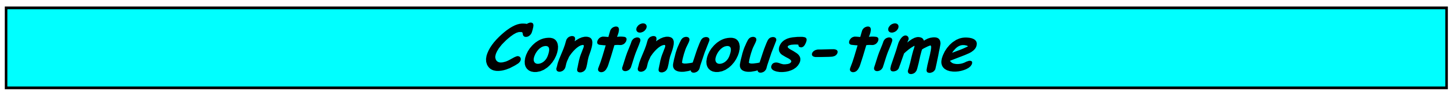
Two main approaches:

Given T, discretize model, design discrete-time controller

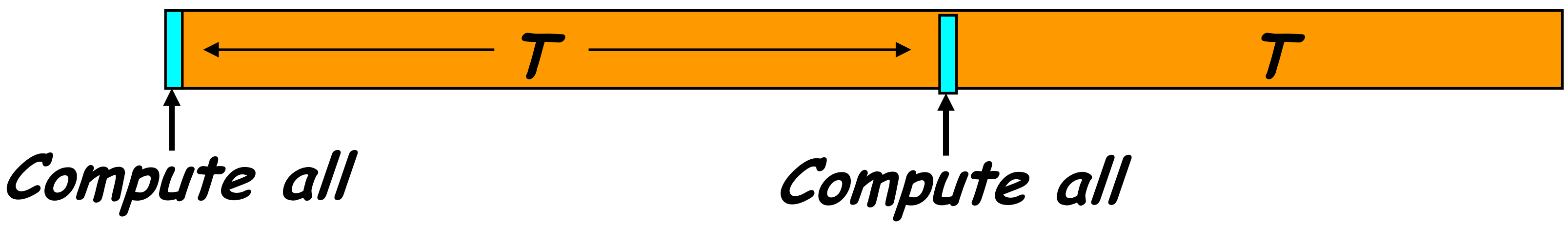
Design continuous controller, then discretize controller using T.

Digital Control Assumptions

Continuous-time control : Control tasks execute and communicate instantaneously at every time point



Discrete-time control : Control tasks execute instantaneously at fixed (periodic) discrete points



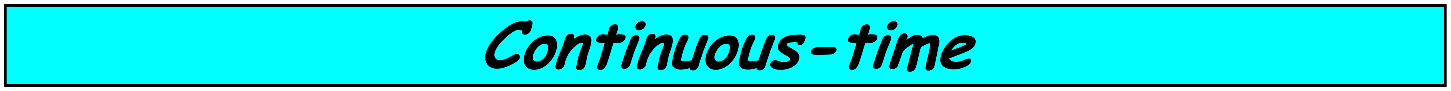
Assumptions:

All computation happens simultaneously
Emphasis on errors due to sampling period T

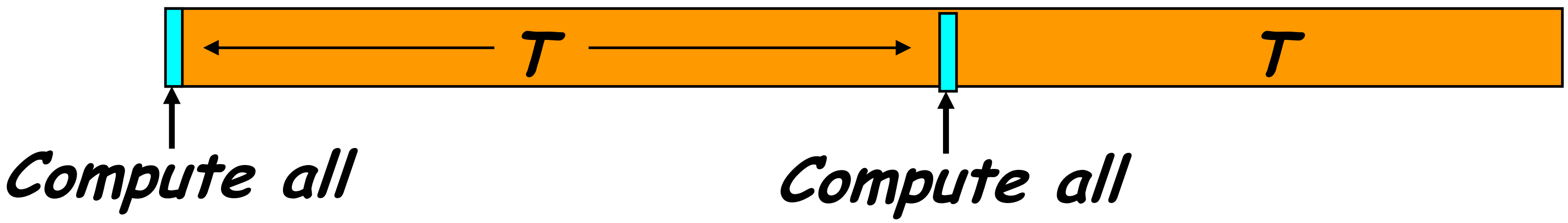
Computational model does not capture modern platforms
Effects of control task scheduling are not modeled nor analyzed

Rethinking Digital Control

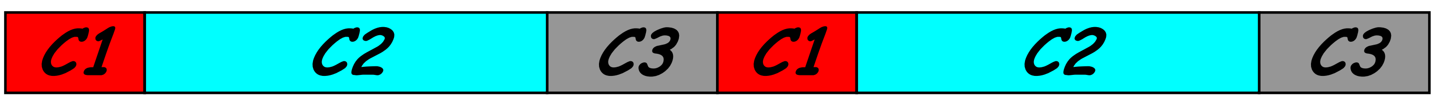
Continuous-time control : Control tasks execute and communicate instantaneously at every time point



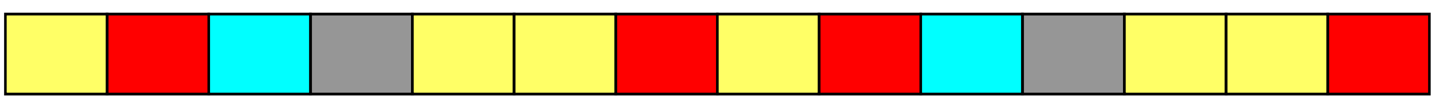
Discrete-time control : Control tasks execute instantaneously at fixed (periodic) discrete points



Periodic computations take time



Schedule on TTP: Fixed sized slots



Control and Implementation

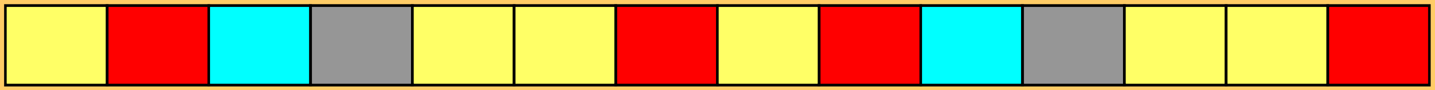
Continuous-time control : Control tasks execute and communicate instantaneously at every time point

Continuous-time

Given continuous controller, TT platform, and schedule, quantify implementation error

Given TT platform, schedule, design continuous controller. (Control-scheduling co-design).

Schedule on TTP : Fixed sized slots



Separation of Concerns

Control design in continuous-time

- Many benefits: composable, powerful design tools
- Portable to many (or evolving) platforms
- Provides interface to system/software engineer to implement
- Should not worry about platform details

Software implementation

- Should not worry about control methods or details
- Focus on fault tolerant implementation, code, scheduling
- Make sure the implementation follows continuous time design

Feedback Control Model - Syntax

Plant variable $X = \{x_1, x_2, \dots, x_n\}$

Plant output variables $Y = \{y_1, \dots, y_p\}$

Control variables $U = \{u_1, \dots, u_m\}$

Internal variables $Z = \{z_1, \dots, z_q\}$

Plant model $\mathcal{M}_P$

$$f : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n$$

$$h : \mathbb{R}^n \rightarrow \mathbb{R}^p$$

Controller model $\mathcal{M}_C$

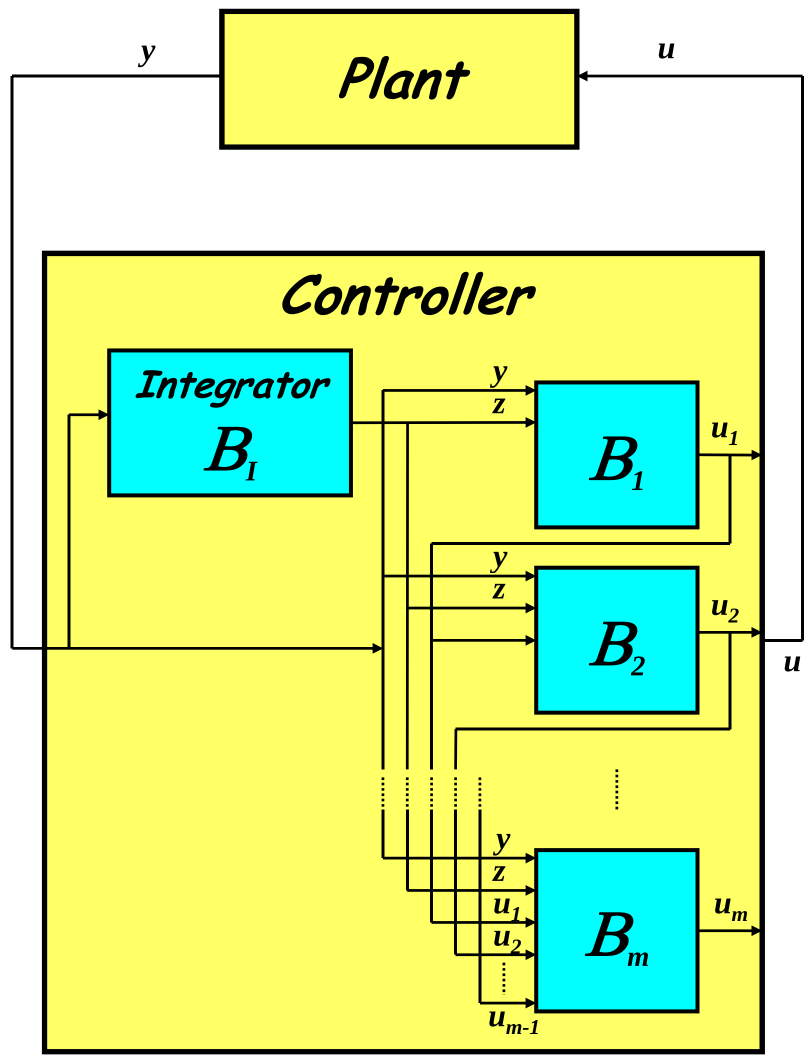
Control blc $\mathcal{M}_C = (\mathcal{B}_I, \mathcal{B}_1, \dots, \mathcal{B}_m)$

Functions for feedback control law

$$g : \mathbb{R}^q \times \mathbb{R}^p \rightarrow \mathbb{R}^q$$

$$k_j : \mathbb{R}^p \times \mathbb{R}^p \times \mathbb{R}^q \times \mathbb{R}^{j-1} \rightarrow \mathbb{R}$$

Acyclic dependence of control variables



Model-level (ideal) Semantics

Consider trajectories of all variables, for $t \geq 0$,

$$x(t) = (x_1(t), \ldots, x_n(t)) \qquad u(t) = (u_1(t), \ldots, u_m(t))$$

$$y(t) = (y_1(t), \ldots, y_p(t)) \qquad z(t) = (z_1(t), \ldots, z_q(t))$$

Given feedback control model $\mathcal{M} = \langle \mathcal{M}_P, \mathcal{M}_C \rangle$ and initial state $x(0)$, the continuous-time semantics is the **unique** trajectory satisfying

$$M_P : \begin{cases} \dot{x}(t) & = & f(x(t), u(t)) \\ y(t) & = & h(x(t)) \\ x(0) & \in & \mathbb{R}^n \end{cases}$$

$$M_C : \begin{cases} \dot{z}(t) & = & g(z(t), y(t)) \\ u_1(t) & = & k_1(y(t), \dot{y}(t), z(t)) \\ u_j(t) & = & k_j(y(t), \dot{y}(t), z(t), u_1(t), \ldots, u_{j-1}(t)) \\ & & \qquad 2 \leq j \leq m \\ z(0) & = & 0 \end{cases}$$

We denote this trajectory as $(x(t), y(t), u(t)) = [\![\mathcal{M}]\!]_C(x(0))$

The model-level semantics is *implementation independent.*

Penn
University of Pennsylvania

Implementation Modeling

Ideal (model) semantics assumes

Control blocks compute simultaneously

Control blocks compute instantaneously

Time-triggered platform model (ρ, τ, δ)

1. Dispatch sequence ρ (models ordering)

   Periodic string over

   $$\{\mathcal{B}_0, \mathcal{B}_I, \mathcal{B}_1, \ldots, \mathcal{B}_m\}$$

   Examples: $(\mathcal{B}_I \mathcal{B}_1 \mathcal{B}_2 \mathcal{B}_3)^\omega$

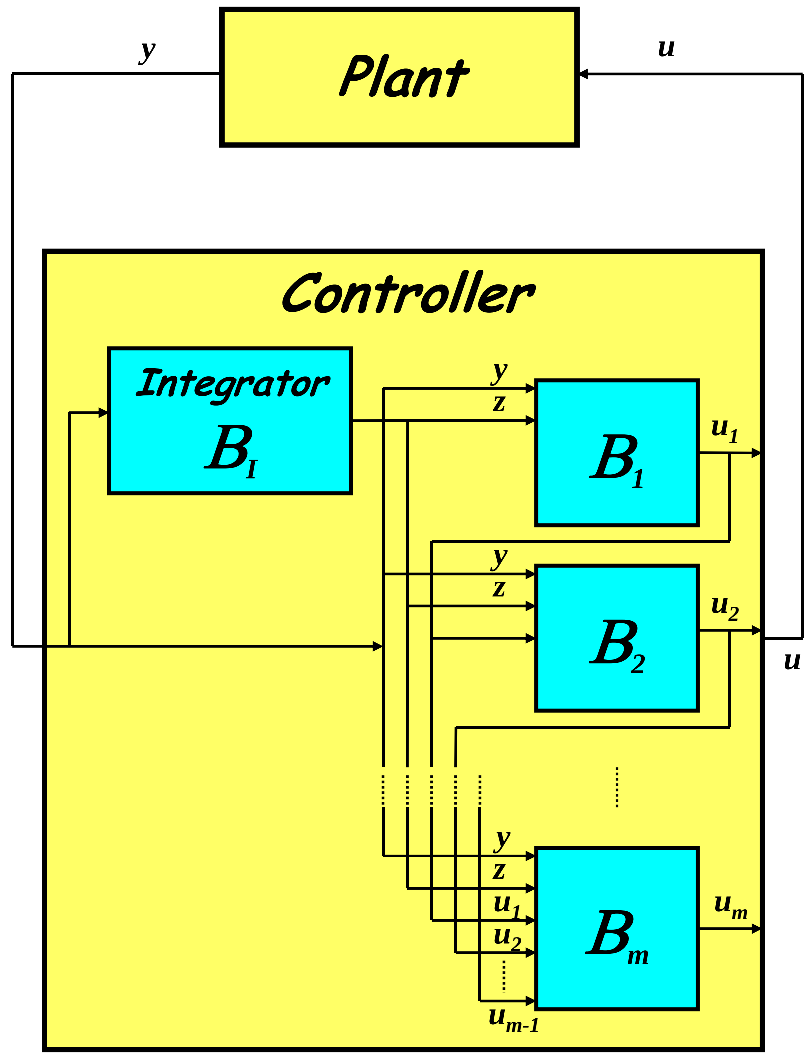
   $(\mathcal{B}_I \mathcal{B}_1 \mathcal{B}_I \mathcal{B}_2 \mathcal{B}_I \mathcal{B}_1 \mathcal{B}_I \mathcal{B}_3 \mathcal{B}_0)^\omega$

2. Timing function τ (models timing)

   $$\tau : \{\mathcal{B}_I, \mathcal{B}_1, \ldots, \mathcal{B}_m\} \rightarrow \mathbb{Z}^+$$

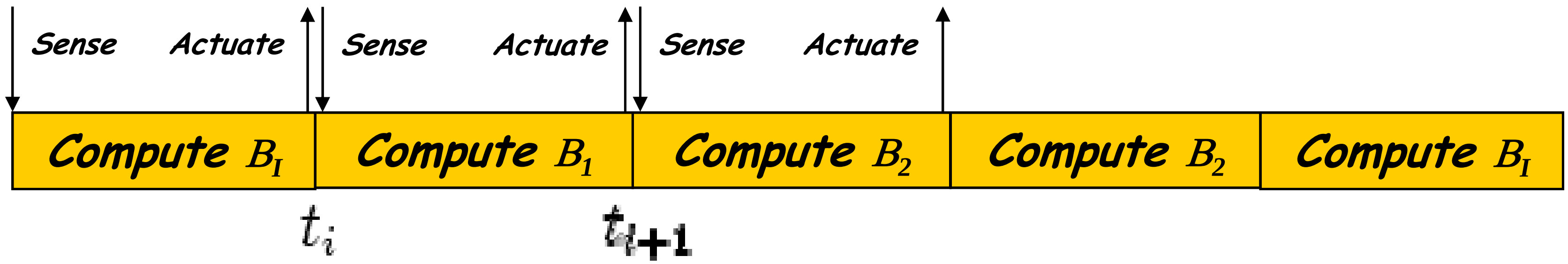
   $$\tau(\mathcal{B}_0) = 1$$

3. Duration of time slot δ



Implementation Semantics - Timing

Consider the dispatch sequence $(\mathcal{B}_I \mathcal{B}_1 \mathcal{B}_2 \mathcal{B}_2)^\omega$



Timing

$$t_0 = 0$$

$$t_i = \sum_{k=0}^{i-1} \tau(\rho(k))\delta \quad \text{for } \ i \geq 1$$

Integration

$$\Delta_I(0) = 0$$

$$\Delta_I(i+1) = \begin{cases} \Delta_I(i) + \tau(\rho(i))\delta & \text{if } \rho(i) \neq \mathcal{B}_I \\ \tau(\mathcal{B}_I)\delta & \text{if } \rho(i) = \mathcal{B}_I \end{cases}$$

Differentiation

$$\Delta_D(0) = 0$$

$$\Delta_D(i+1) = \begin{cases} \Delta_D(i) + \tau(\rho(i))\delta & \text{if } \rho(i) \in \{\mathcal{B}_0, \mathcal{B}_I\} \\ \tau(\mathcal{B}_j)\delta & \text{if } \rho(i) \notin \{\mathcal{B}_0, \mathcal{B}_I\} \end{cases}$$

Implementation Error

Given model and implementation semantics, the implementation error is defined as :

$$\begin{aligned}
(x(t), y(t), u(t), z(t)) &= [\![\mathcal{M}]\!](x(0)) \\
(\tilde{x}(t), \tilde{y}(t), \tilde{u}(t), \tilde{z}(t)) &= [\![\mathcal{M}]\!]_{(\rho,\tau,\delta)}(x(0)) \\
e_{\mathcal{M}}(\rho, \tau, \delta, x(0)) &= \int_0^{+\infty} \|y(t) - \tilde{y}(t)\|_2^2 dt
\end{aligned}$$

Note that error is measured using the infinite horizon L_2 norm.

Partial order on implementations based on errors

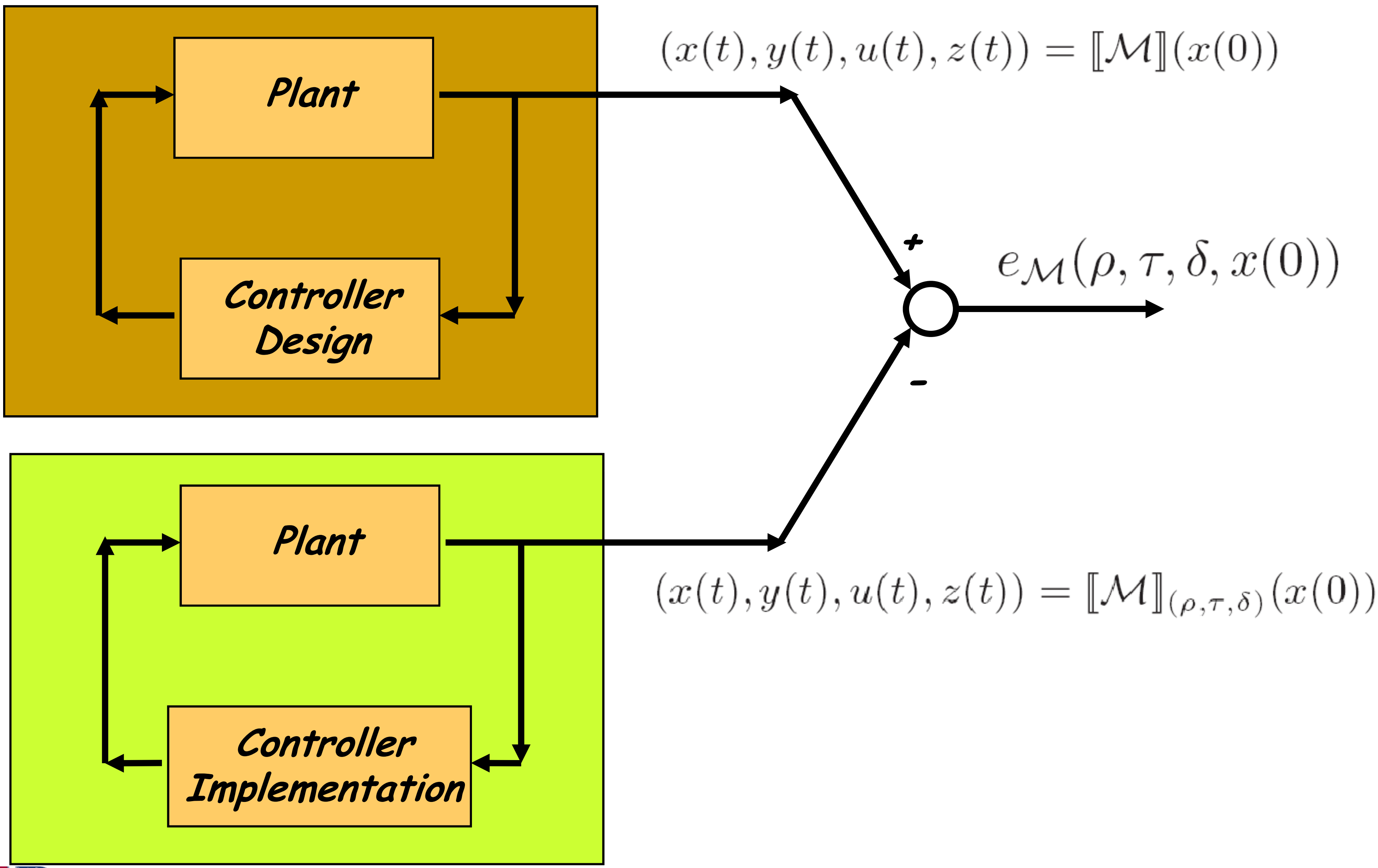
$$(\rho_1, \tau_1, \delta_1) \preceq_M (\rho_2, \tau_2.\delta_2)$$

iff

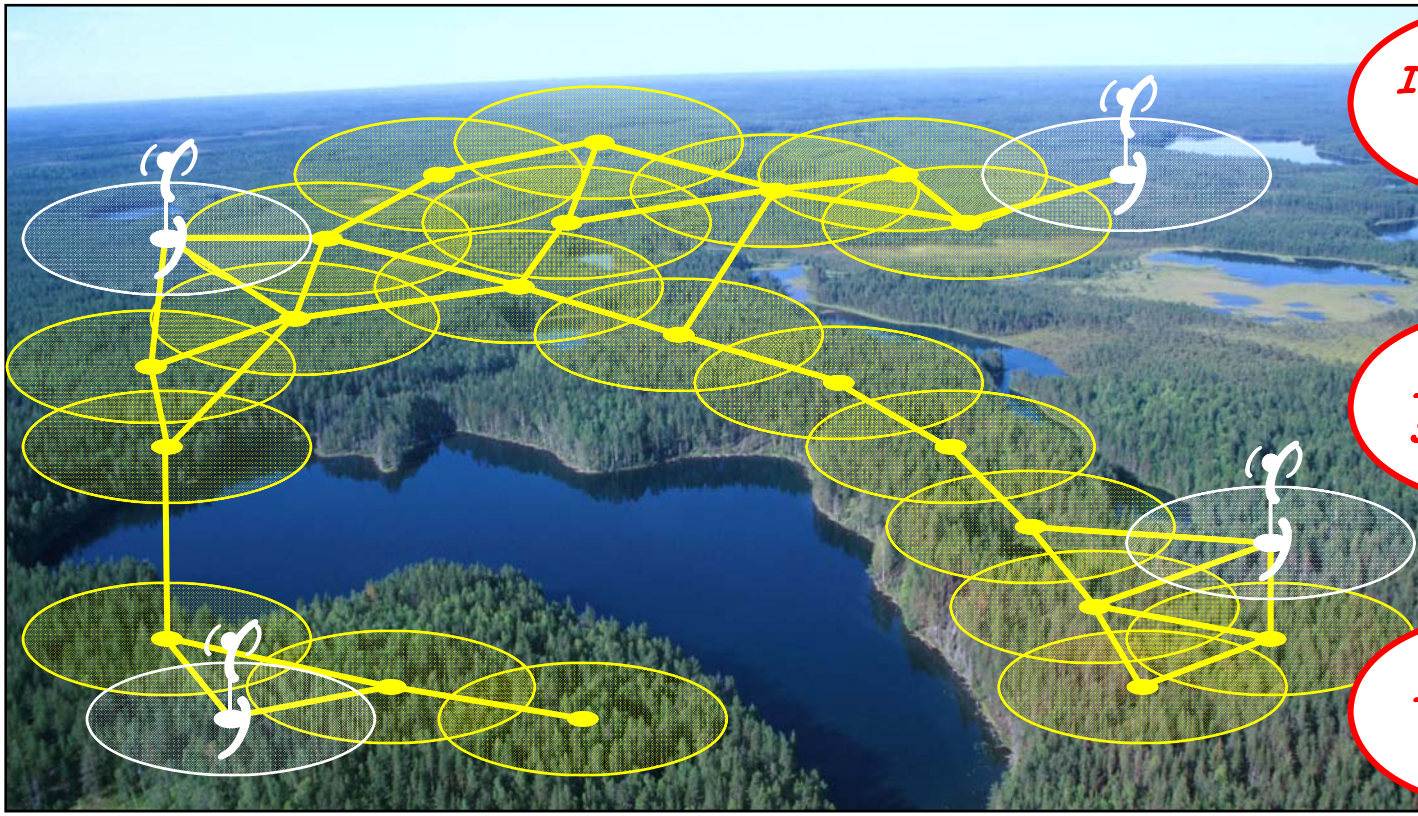
$$\forall x(0) \quad e_M(\rho_1, \tau_1, \delta_1, x(0)) \leq e_M(\rho_2, \tau_2, \delta_2, x(0)) \quad \textbf{\textit{(Global)}}$$

$$\forall\, x(0) \in I \quad e_M(\rho_1, \tau_1, \delta_1, x(0)) \leq e_M(\rho_2, \tau_2, \delta_2, x(0)) \quad \textbf{\textit{(Local)}}$$

Closed-loop Implementation Error



Control over sensor networks

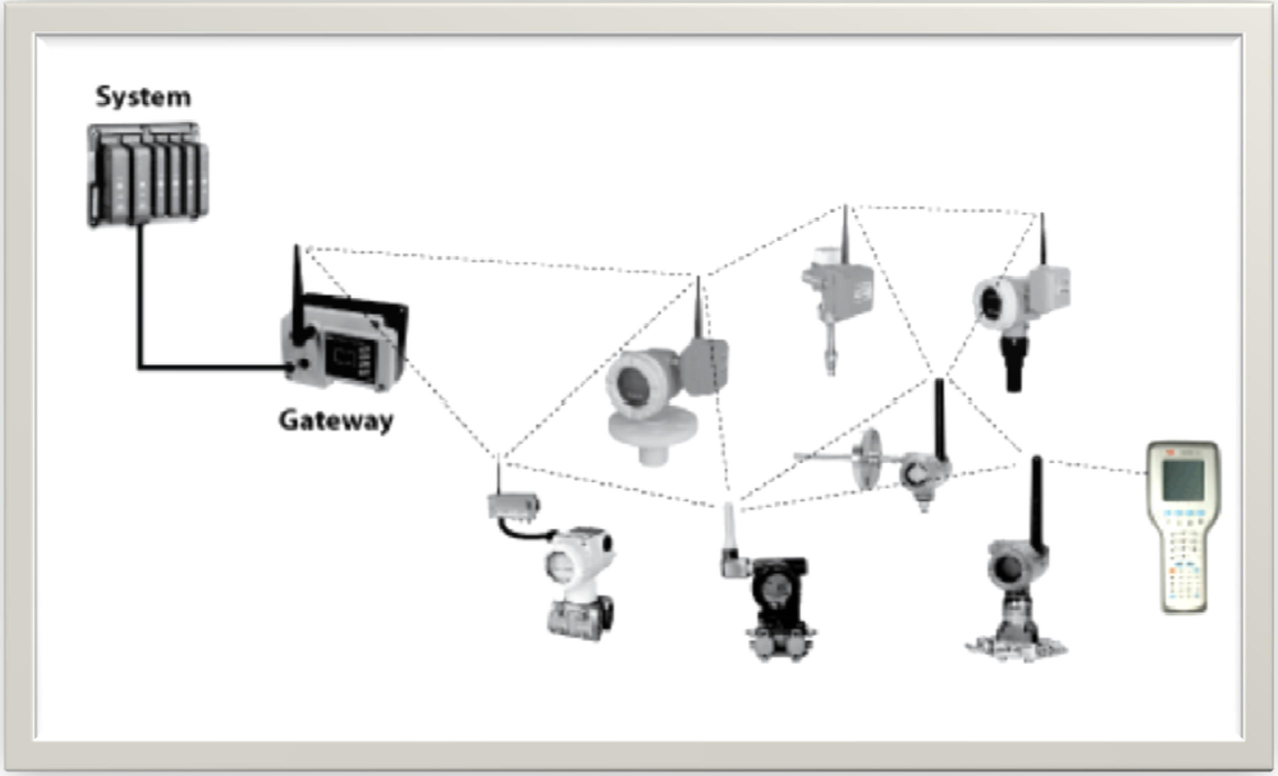


Impact of Delays

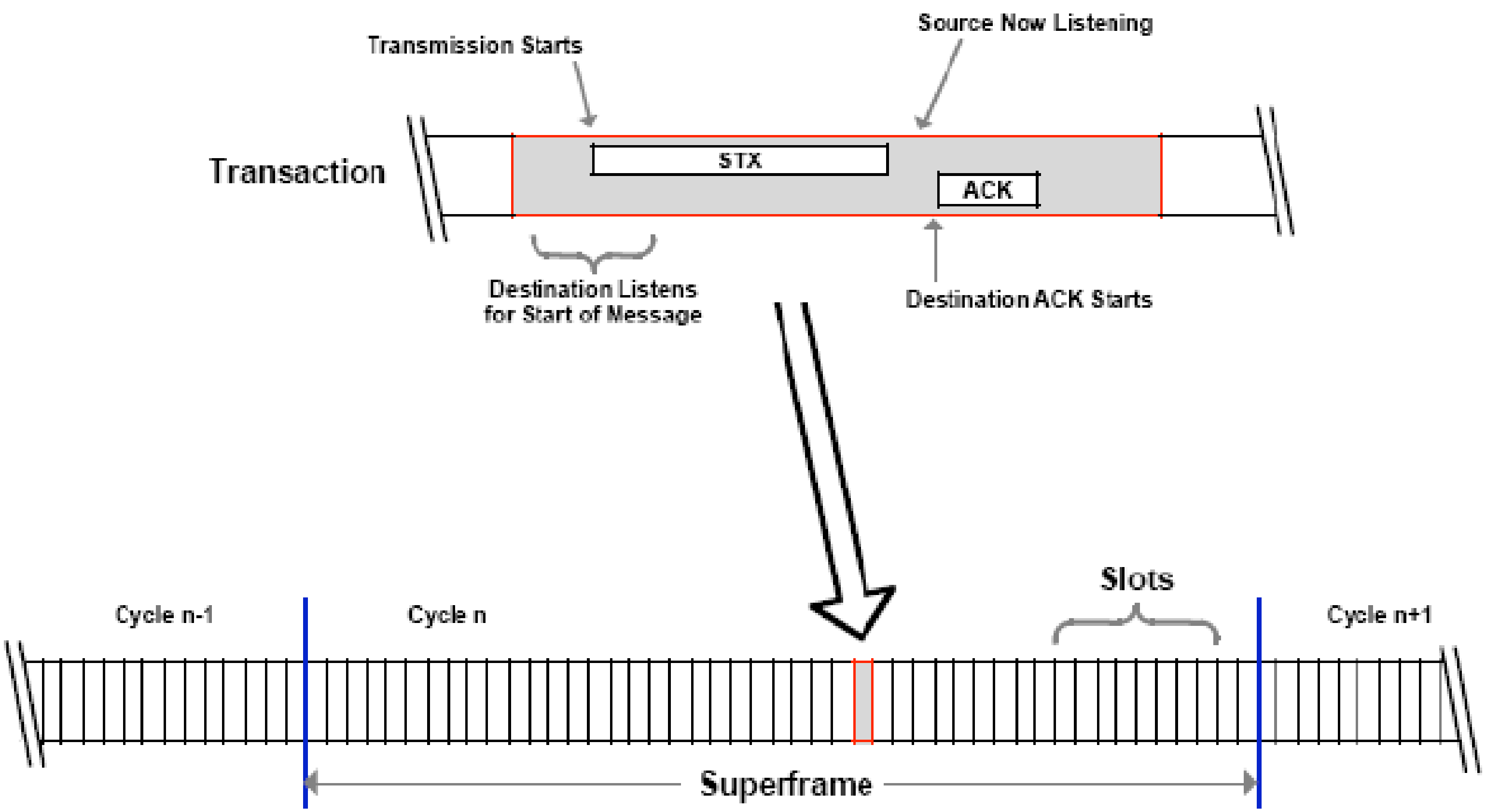
Impact of Scheduling

Impact of Routing

Challenge: Close the loop around wireless sensor networks

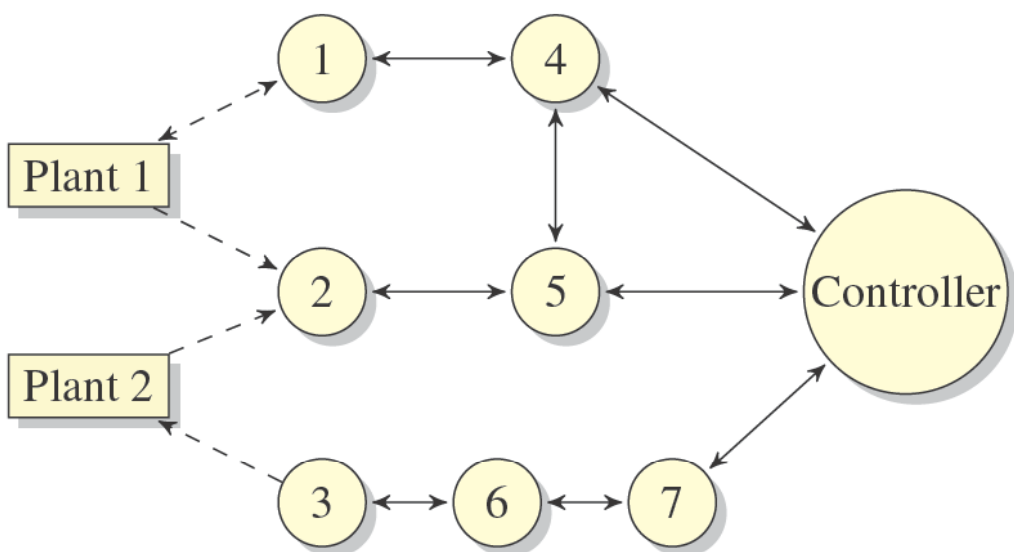

System
Gateway

Wireless HART - MAC level (TDMA - FDMA)



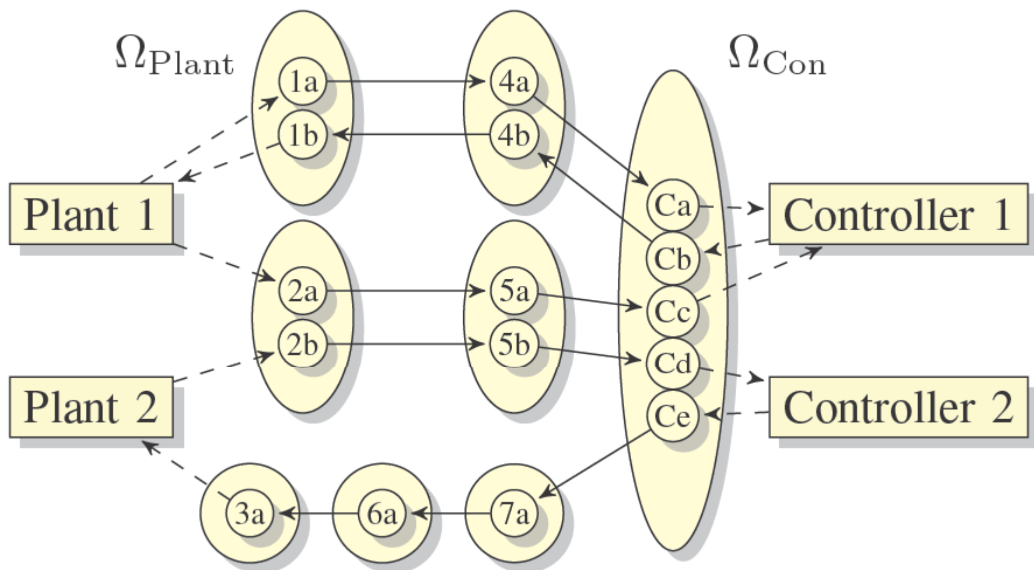
A formal model - syntax

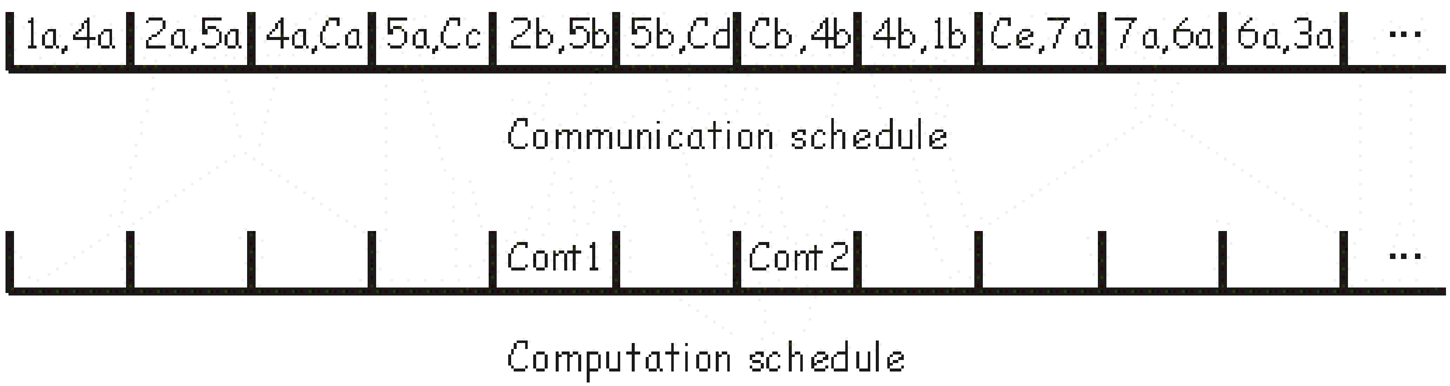
- **Plants/Controllers** D = (P_1, … P_n, C_1, … C_n), are discrete-time LTI systems/controllers
- **Graph** G = (V,E) where V is the set of nodes and E is the radio connectivity graph
- **Routing** R : $I \cup O \to 2^{V^*} \backslash \{\varnothing\}$ associates to each pair sensor-controller or controller actuator a set of allowed routing paths



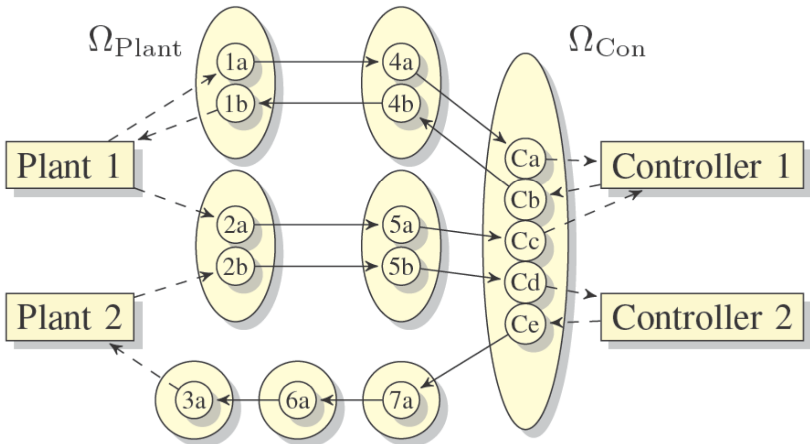
Penn
University of Pennsylvania

Communication and computation schedule

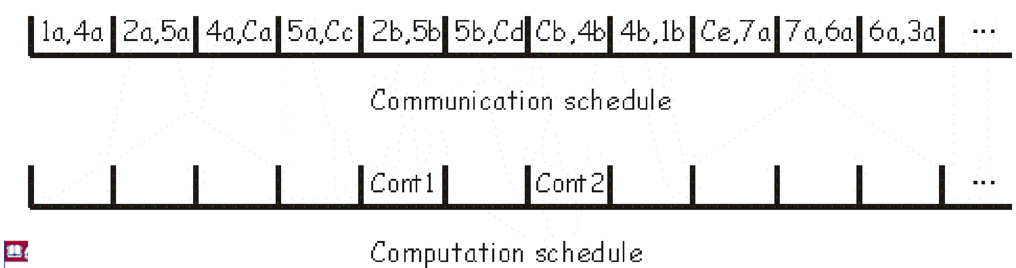




Semantics in each time slot



$$A_l(e,m) := \begin{pmatrix} A_{\text{Plant}} & B_{\text{Plant}} \cdot O_{\text{Plant}} & 0 \\ I_{\text{Plant}}^T \cdot C_{\text{Plant}} & \text{Adj}(\langle V_{\mathfrak{R}}, e\rangle)^{\text{T}} & O_{\text{Con}}^T \cdot C_{\text{Con}}(m) \\ 0 & B_{\text{Con}}(m) \cdot I_{\text{Con}} & A_{\text{Con}}(m) \end{pmatrix}$$



A formal model - Semantics

Given communication/computation schedules, the closed loop control system is a switched linear system:

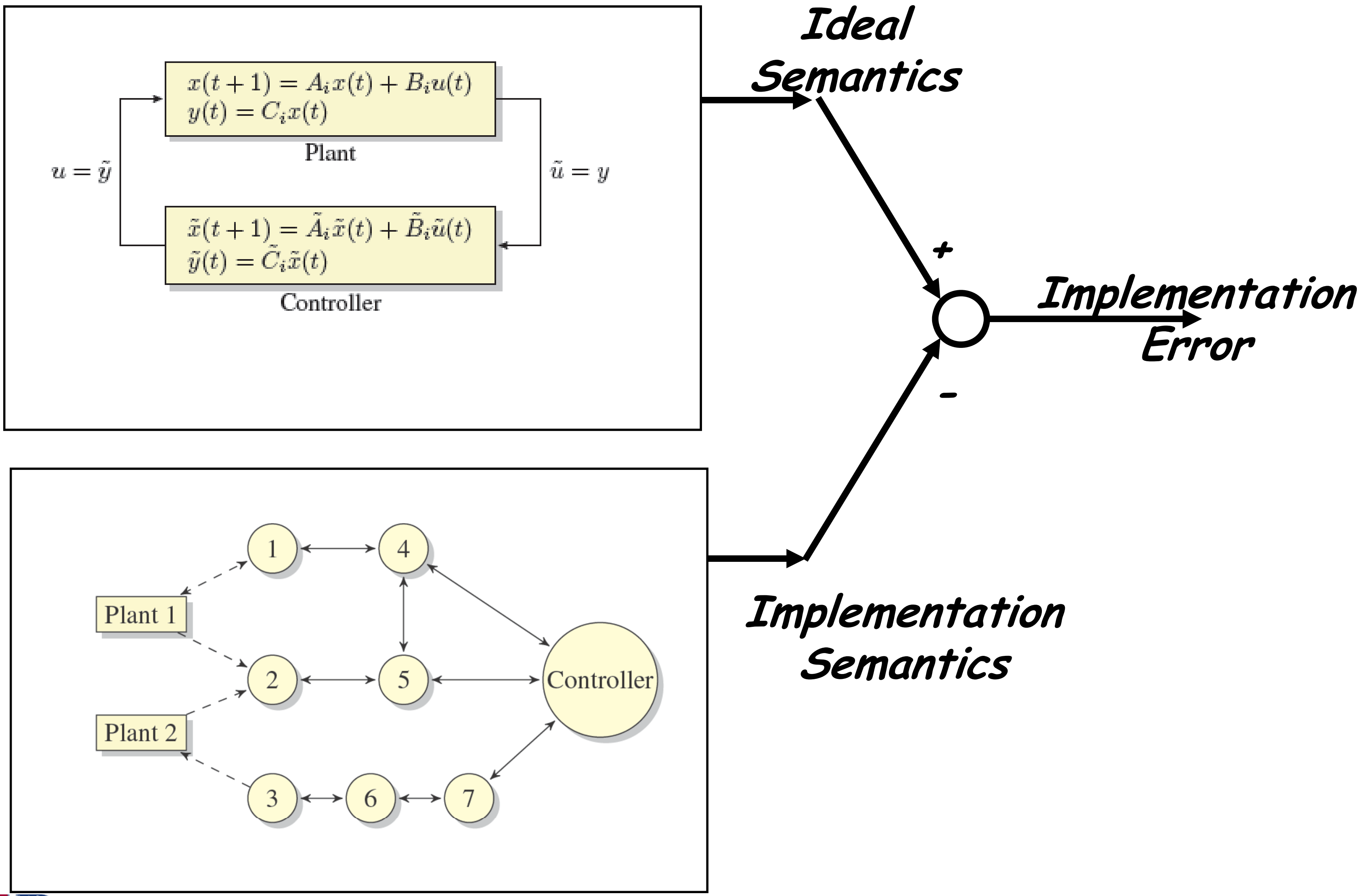
$$x(t+1) = A_c(\eta(t), \mu_c(t))x(t)$$

where $x = (x_p, x_v, x_c)$ and x_p, x_c model the states of the plant and of the controller, and x_v models the measured and control data flow in the nodes of the network

$$A_l(e,m) := \begin{pmatrix} A_{\text{Plant}} & B_{\text{Plant}} \cdot O_{\text{Plant}} & 0 \\ I^T_{\text{Plant}} \cdot C_{\text{Plant}} & \text{Adj}(\langle V_{\mathcal{R}}, e\rangle)^{\text{T}} & O^T_{\text{Con}} \cdot C_{\text{Con}}(m) \\ 0 & B_{\text{Con}}(m) \cdot I_{\text{Con}} & A_{\text{Con}}(m) \end{pmatrix}$$

Penn
University of Pennsylvania

Analysis Approach



Approximation Error

Given model and implementation semantics, the implementation error is defined as :

$$
\begin{aligned}
(x(t), y(t), u(t), z(t)) &= [\![\mathcal{M}]\!](x(0)) \\
(\tilde{x}(t), \tilde{y}(t), \tilde{u}(t), \tilde{z}(t)) &= [\![\mathcal{M}]\!]_{(\rho,\tau,\delta)}(x(0)) \\
e_{\mathcal{M}}(\rho, \tau, \delta, x(0)) &= \int_0^{+\infty} \|y(t) - \tilde{y}(t)\|_2^2 dt
\end{aligned}
$$

Note that error is measured using the L_2 norm.

Partial order on implementations based on errors

Penn
University of Pennsylvania

Analysis

Periodic deterministic scheduling (Wireless HART single-hop)

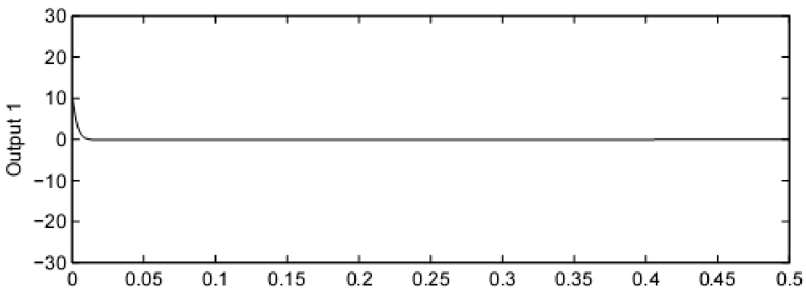
- Theory of periodic time varying linear systems is relevant
- Schedule is a fixed string in the alphabet of edges/controllers
- Nghiem,Pappas,Girard,Alur – EMSOFT 2006, ACM TECS 2008

Periodic non-deterministic scheduling (Wireless HART multi-hop)

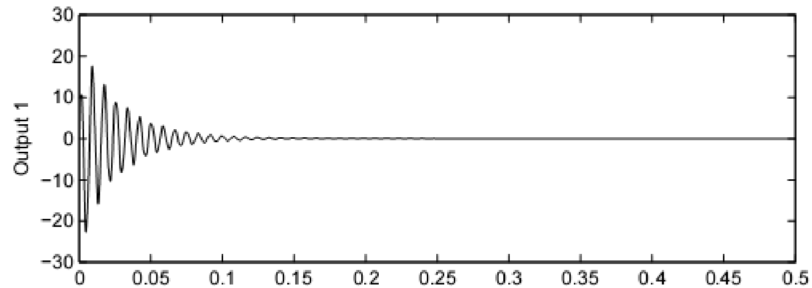
- Theory of switched/hybrid linear system applies
- Schedule is an automaton over edges/controllers
- Alur, Weiss – HSCC 2008

Penn
UNIVERSITY of PENNSYLVANIA

Example - Implementation Errors



Ideal Controller

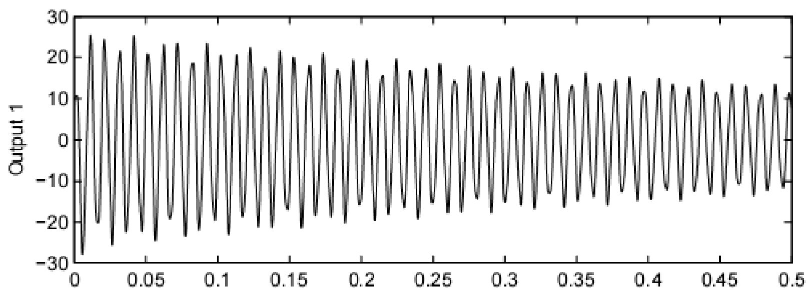


Implementation 2

$\rho_2 = (\mathcal{B}_I \mathcal{B}_1 \mathcal{B}_2)^\omega \quad \tau_2(B_j) = 1 \quad \delta_2 = 0.00075 sec$

Trapezoid & Backward Difference

$e_M(\rho_2, \tau_2, \delta_2, x(0)) = 1.9263$

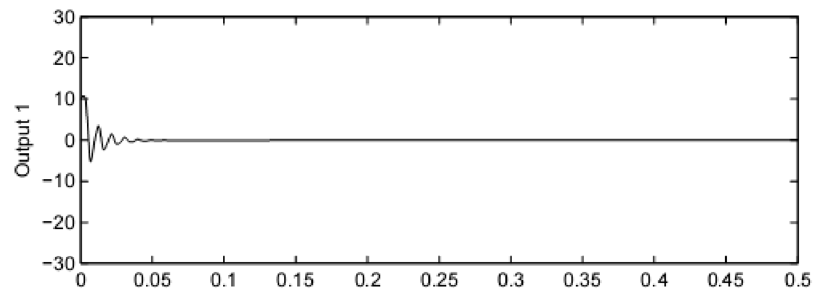


Implementation 1

$\rho_1 = (\mathcal{B}_I \mathcal{B}_1 \mathcal{B}_2)^\omega \quad \tau_1(B_j) = 1 \quad \delta_1 = 0.001 sec$

Euler & Backward Difference

$e_M(\rho_1, \tau_1, \delta_1, x(0)) = 10.0058$



Implementation 3

$\rho_3 = (\mathcal{B}_I \mathcal{B}_2 \mathcal{B}_1)^\omega \quad \tau_3(B_j) = 1 \quad \delta_3 = 0.001 sec$

Euler & Backward Difference

$e_M(\rho_3, \tau_3, \delta_3, x(0)) = 0.5241$

Penn
UNIVERSITY of PENNSYLVANIA

A zoo of hybrid systems

Hybrid Automata
Hybrid Input-Output Automata
Hybrid Petri Nets
Simulink/Stateflow models

Supervisory control systems
Switched systems
Nonsmooth systems
Piece-wise affine systems (PWA)
Mixed Logical Dynamical
Linear Complementarity models