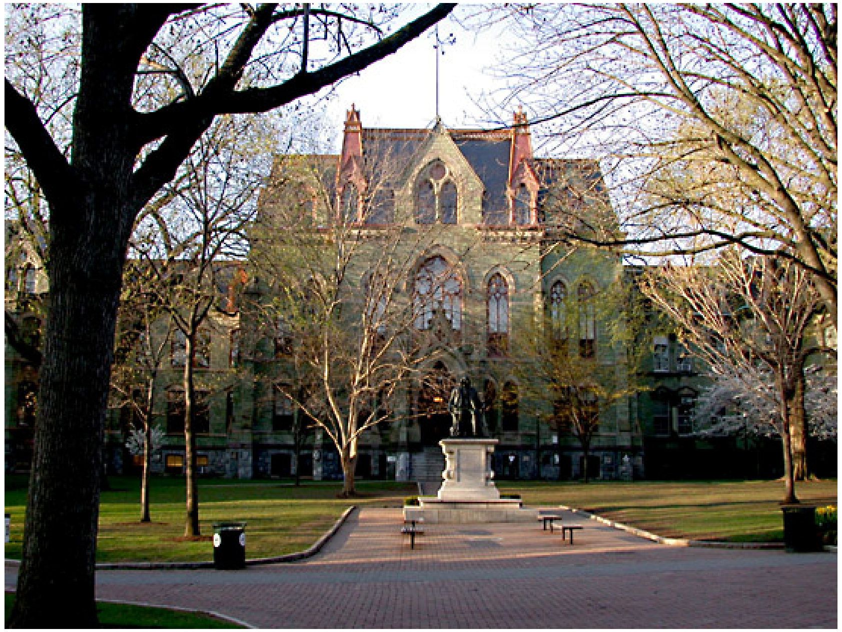
Modeling and Analysis of Distributed Control Networks

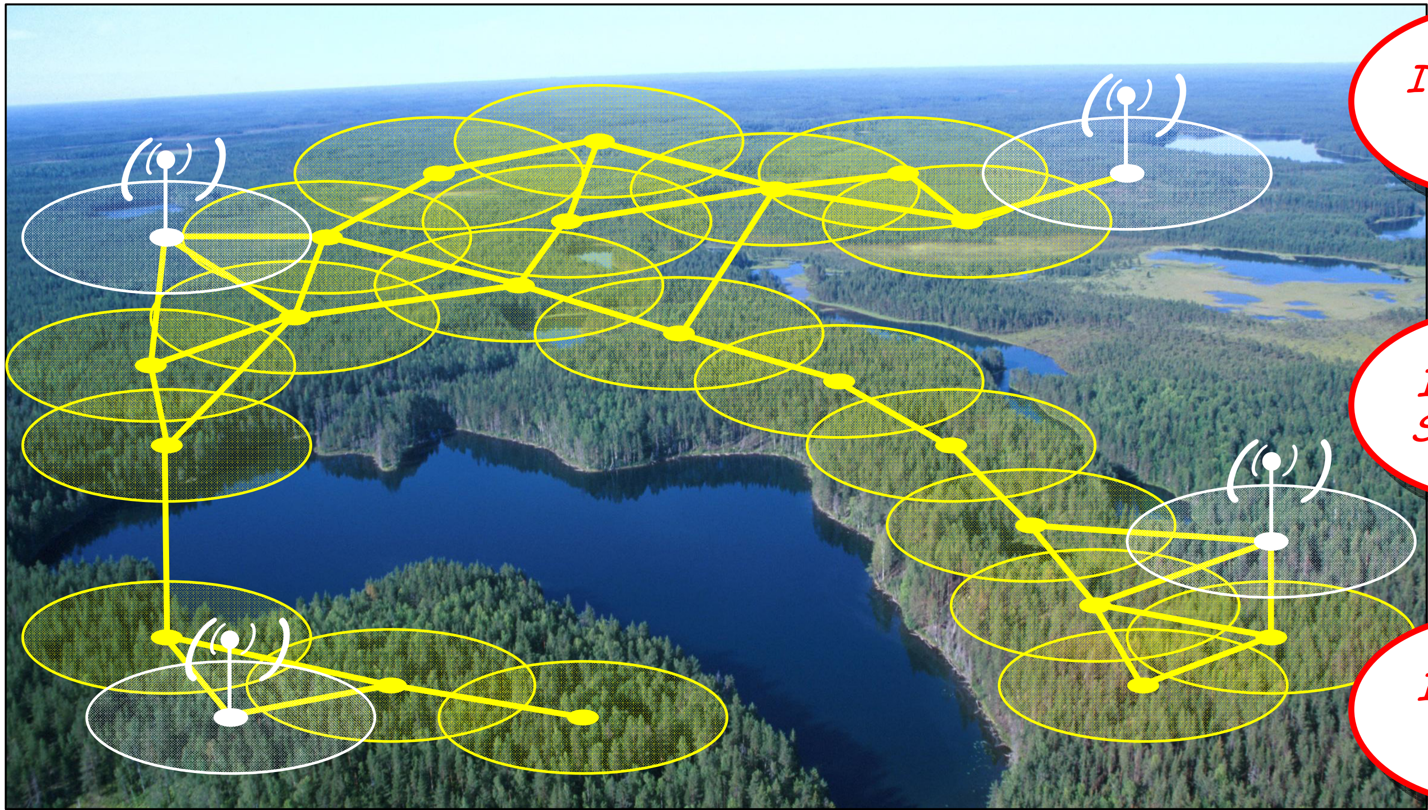


Rajeev Alur, Alessandro D'Innocenzo, Gera Weiss, **George J. Pappas**

PRECISE Center for Embedded Systems

University of Pennsylvania

Motivation



Impact of Delays

Impact of Scheduling

Impact of Routing

Challenge: Close the loop around wireless sensor networks

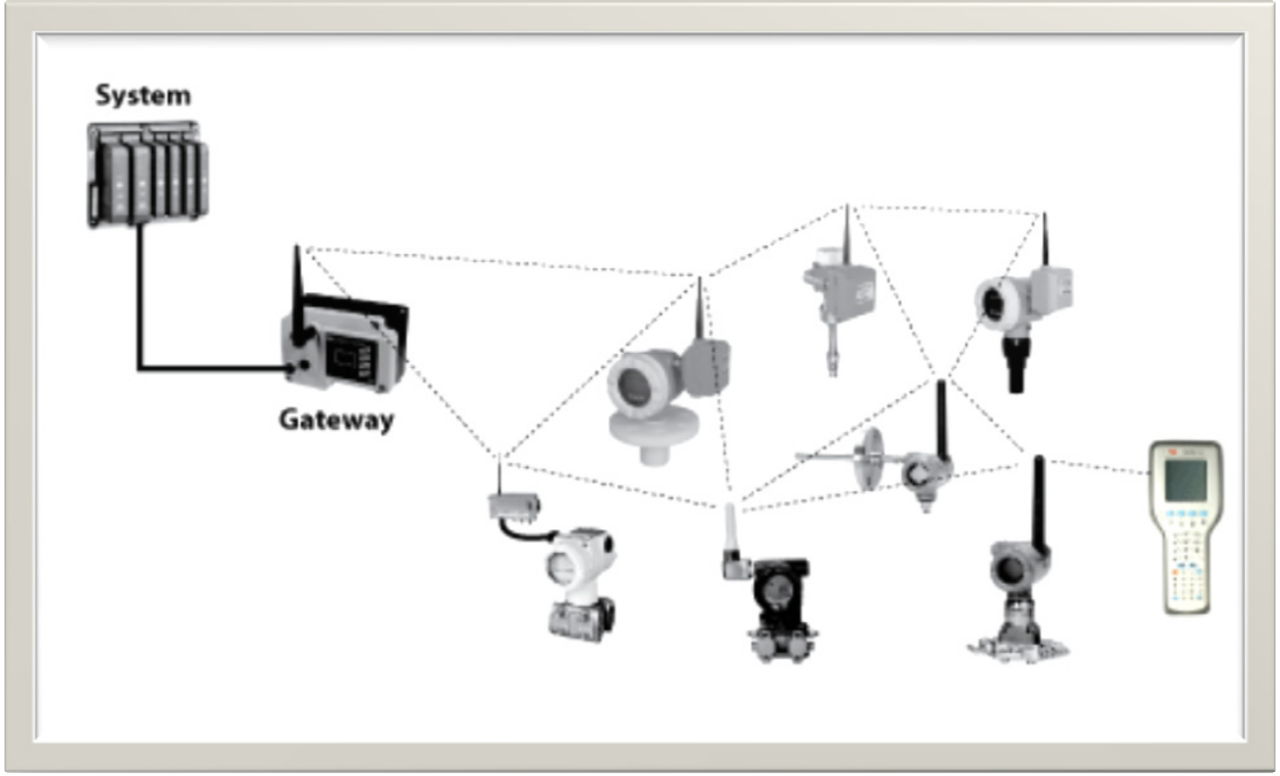
Challenges

- Understand the impact of
  - Time delays
  - Channel capacity
  - Packet losses
  - Scheduling
  - Network topology
  - Routing

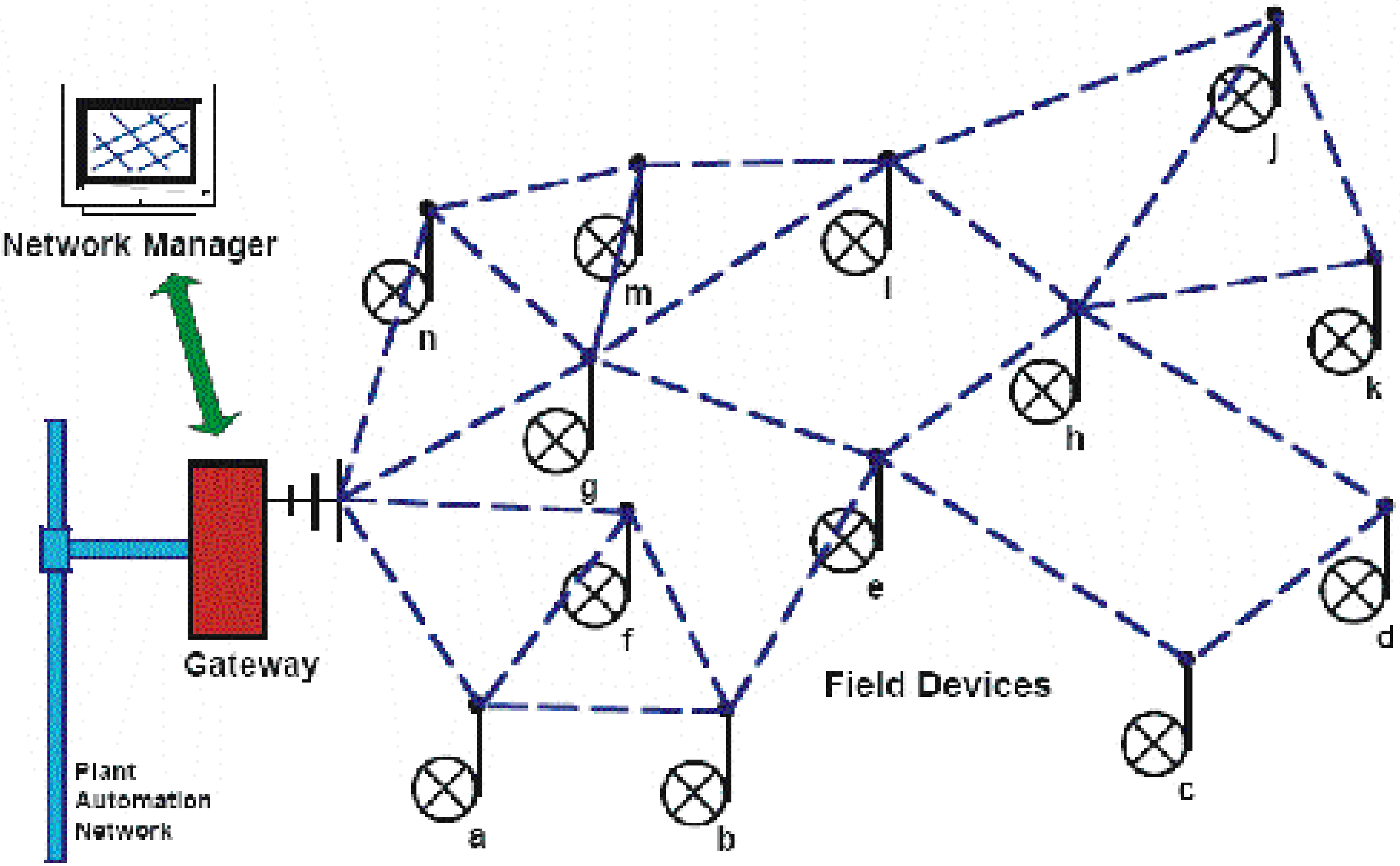
on controller performance, enabling analysis or co-design

- Formal network abstractions enabling analysis

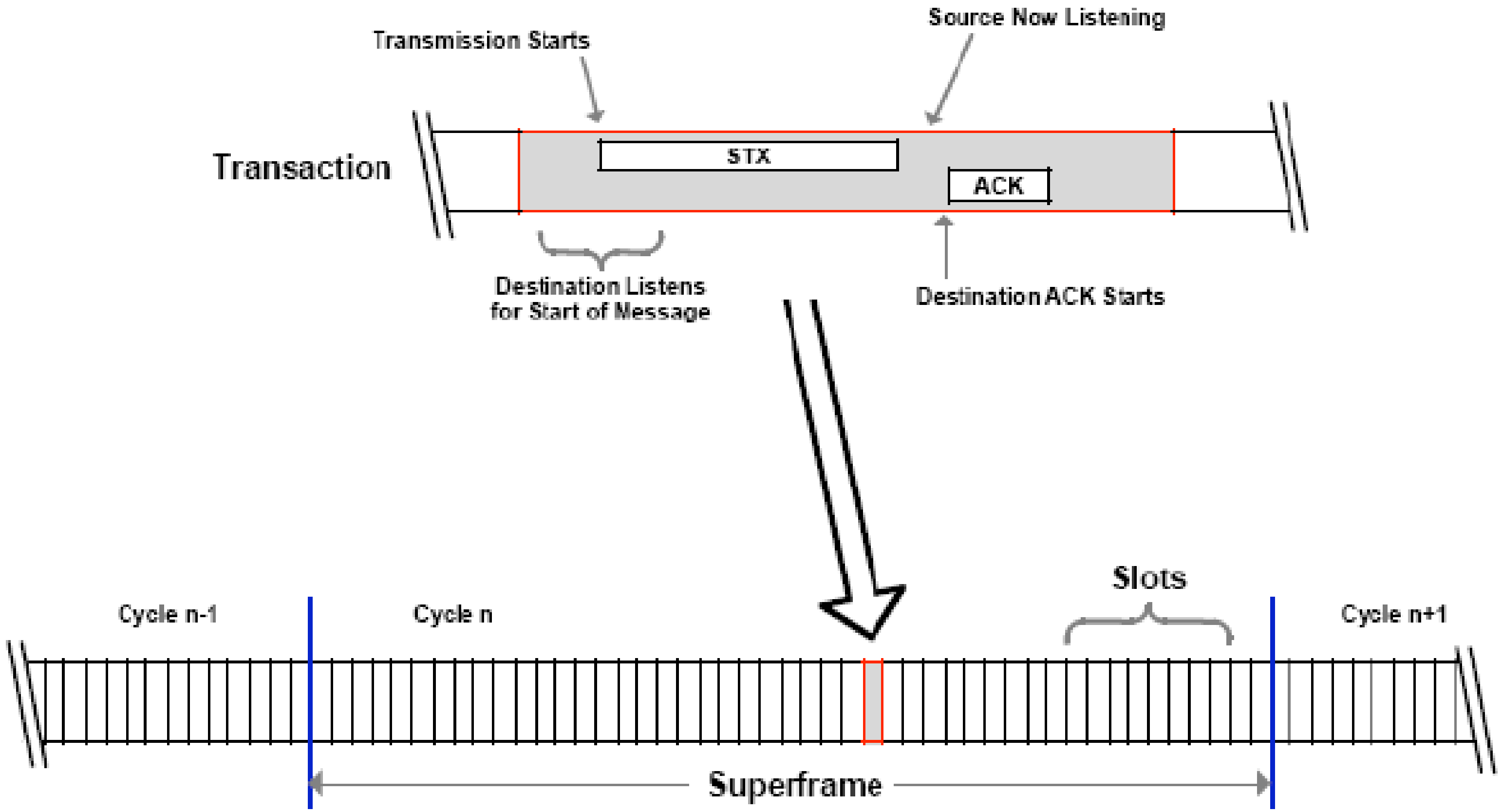
- Analysis should be compositional to changes in the netwok or the addition control control loops


System
Gateway

Wireless HART: a specification for control over wireless networks

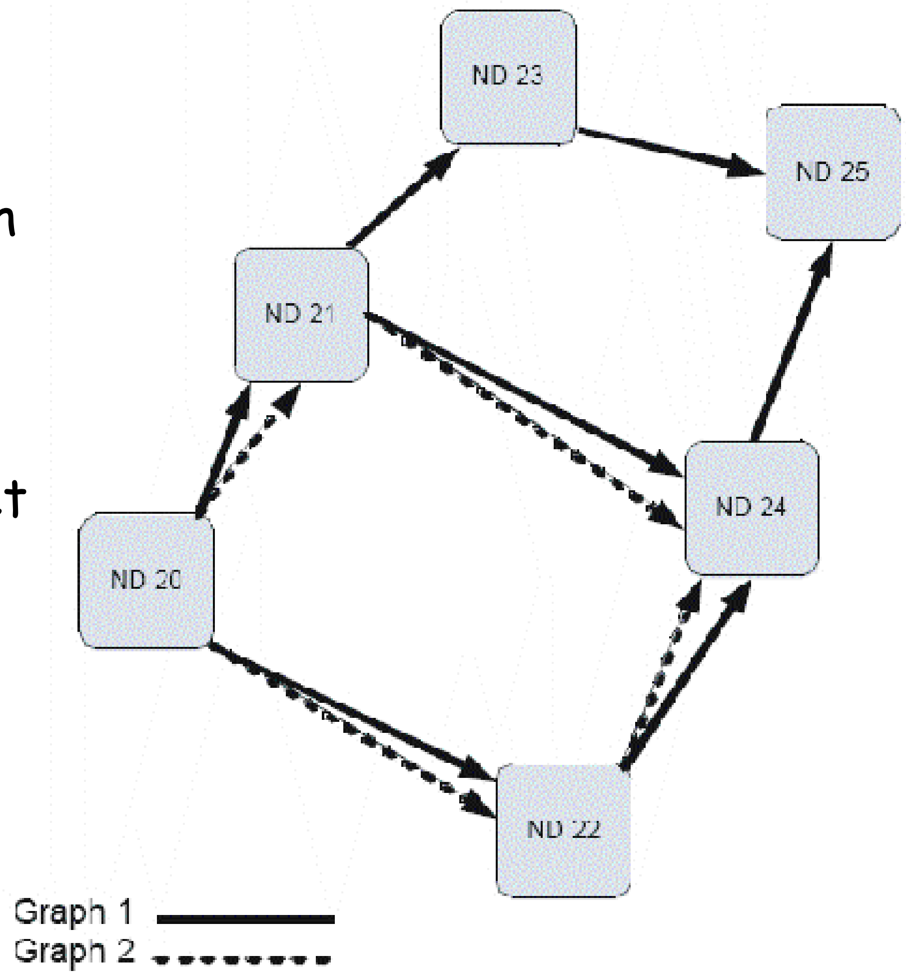


Wireless HART - MAC level (TDMA - FDMA)



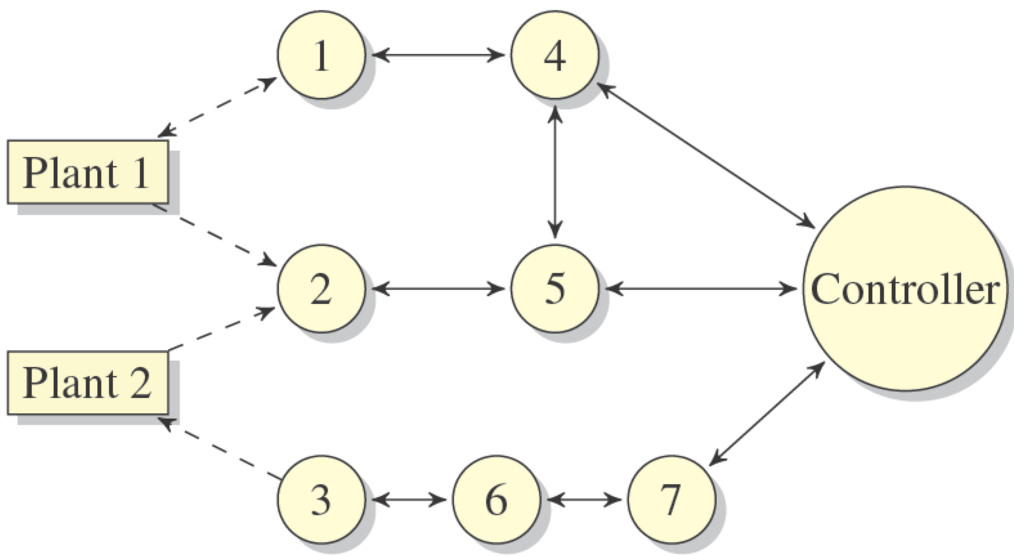
Wireless HART - Network level (Routing)

- Each pair of nodes (source,destination) is associated to an acyclic graph that defines the set of allowed routing
- Dynamic routing in a finite set
- Redundancy in the routing path

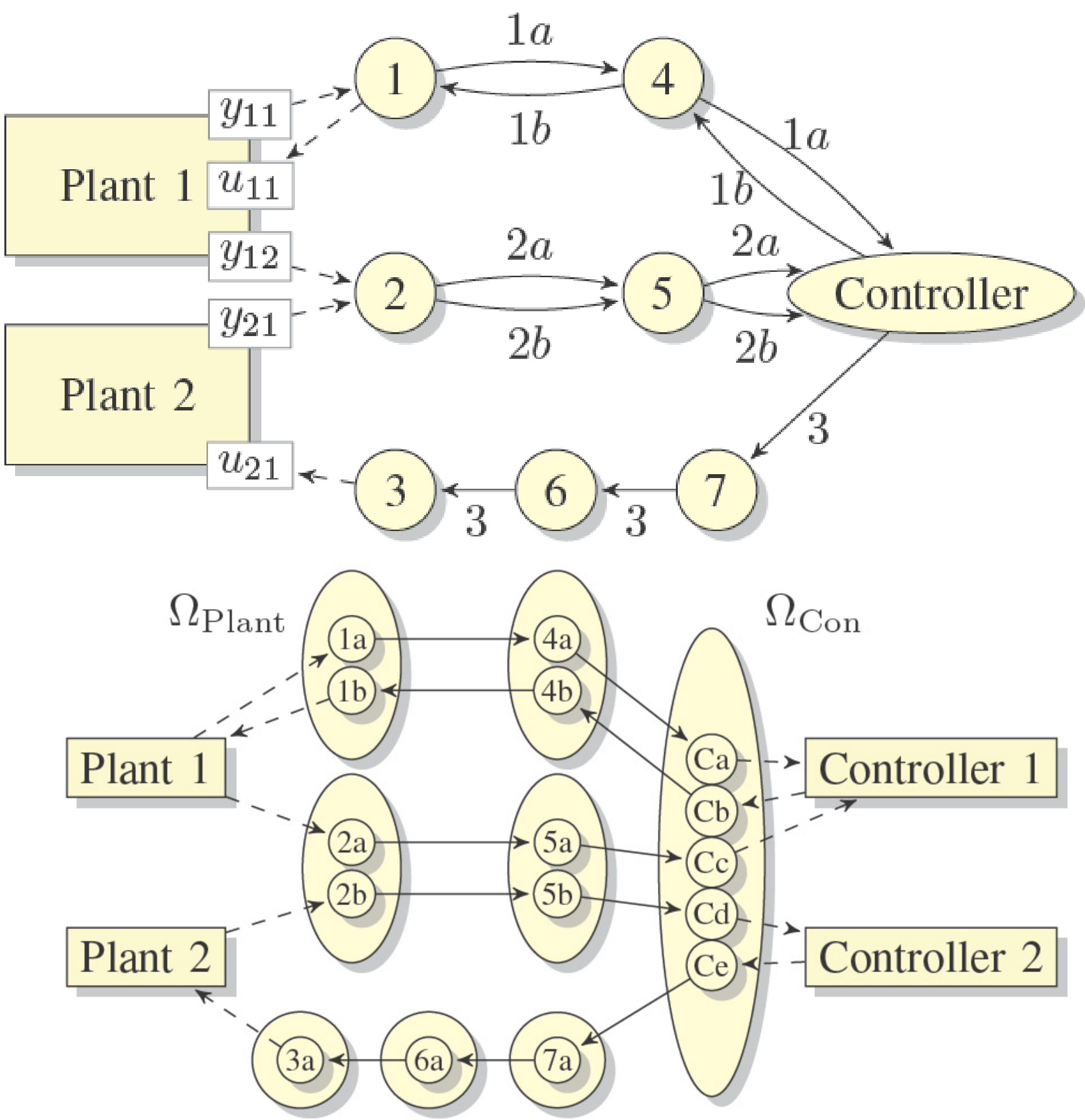


A formal model - syntax

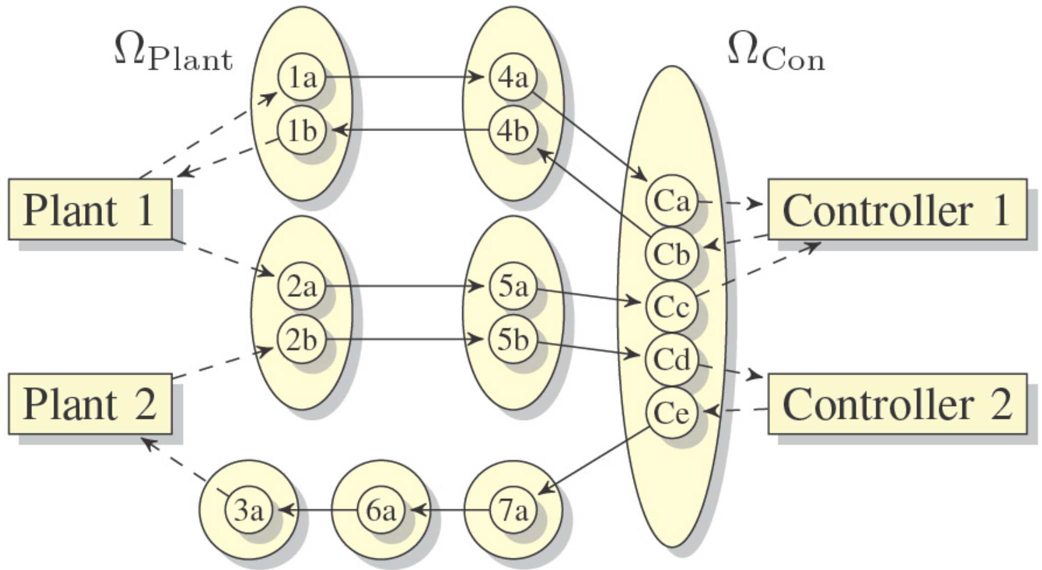
- Plants/Controllers D = (P_1, … P_n, C_1, … C_n), where P_i and C_i are LTI systems
- Graph G = (V,E) where V is the set of nodes and E is the radio connectivity graph
- Routing R : $I \cup O \rightarrow 2^{V^*} \setminus \{\emptyset\}$ associates to each pair sensor-controller or controller actuator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

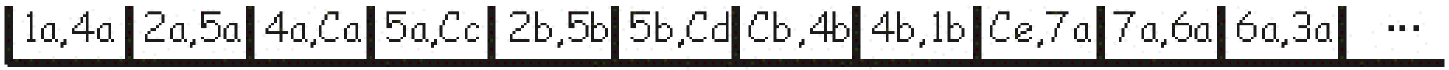


From radio connectivity graph to memory slots graph

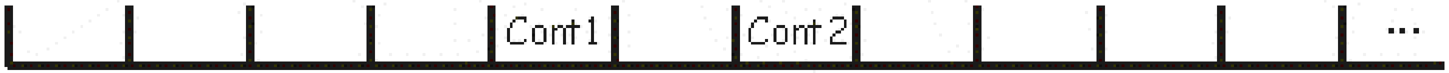


Communication and computation schedule



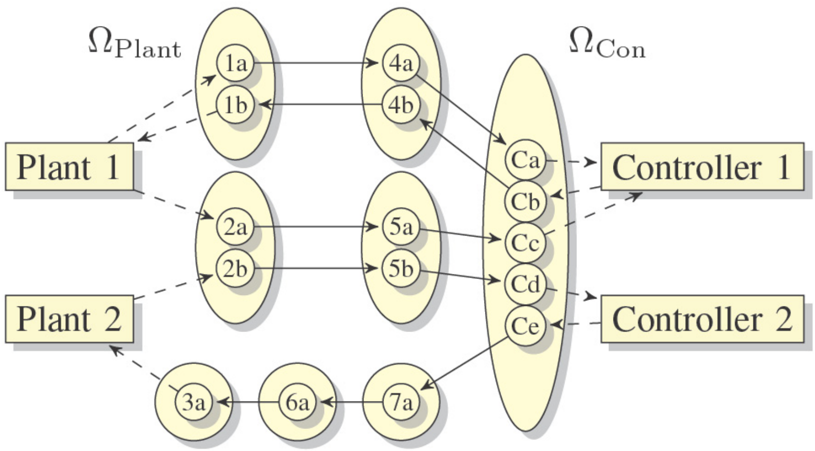


Communication schedule

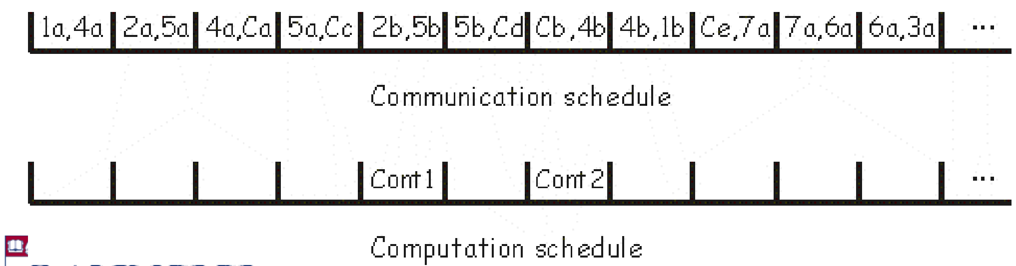


Computation schedule

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

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

Remarks

Algebraic representations of the graph are very useful

Size of matrices depends on the network and hence on the routing

Mathematical Tool

- **Control Loops**

```
Plant[1] = {A_p, B_p, C_p};

Controller[1] = {A_c, B_c, C_c};

Plant[2] = Plant[1];
Controller[2] = Controller[1];

controlLoops = {{Plant[1], Controller[1]}, {Plant[2], Controller[2]} };
```

- **Wireless Network**

```
In[35]:= networkTopology := {1 → 4, 4 → 1, 4 → 5, 5 → 4, 4 → C,
           C → 4, 2 → 5, 5 → 2, 5 → C, C → 5, 3 → 6, 6 → 3, 6 → 7, 7 → 6, 7 → C, C → 7}
```

- **Routing**

```
In[36]:= routing[y_{1,1}] = {{1, 4, C}};
         routing[y_{1,2}] = {{2, 5, C}};
         routing[u_{1,1}] = {{C, 4, 1}};

         routing[y_{2,1}] = {{2, 5, C}};
         routing[y_{2,2}] = {{2, 5, C}};
         routing[u_{2,1}] = {{C, 7, 6, 3}};
```

```
SW = SwitchedSystem_1[controlLoops, networkTopology, routing];

edges = {{1, y_{1,1}} → {4, y_{1,1}}};
M1 = SW[edges, Active],
```

Out[334]=

$$\begin{pmatrix} 1 & \frac{1}{20} & \frac{1}{800} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & \frac{1}{20} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & K_1 & K_2 & K_3 \end{pmatrix}$$

Analysis

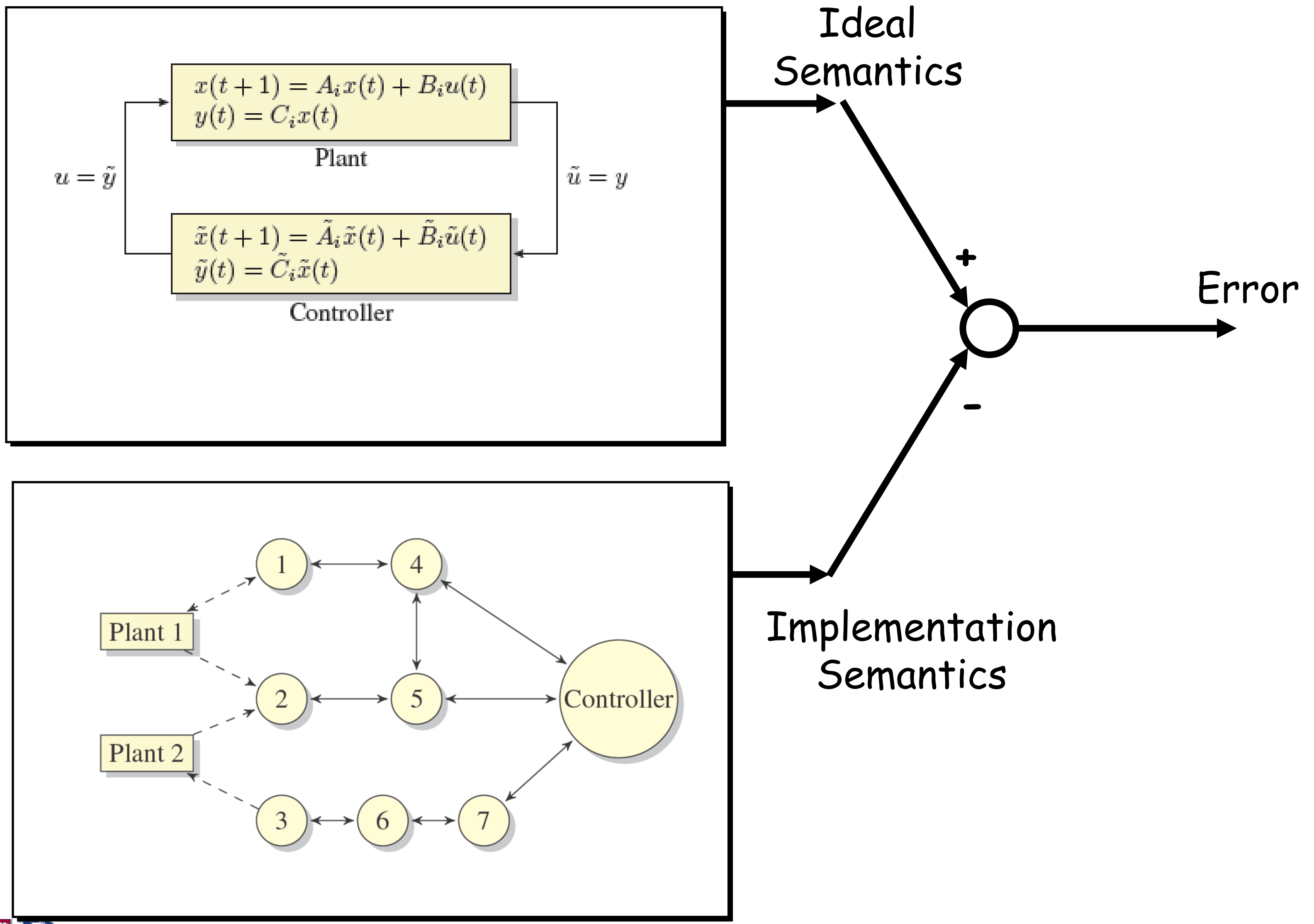
Periodic deterministic scheduling (Wireless HART single-hop)

- Theory of periodic time varying linear systems is relevant
- Schedule is a fixed string in the alphabet of edges/controllers
- Nghiem,Pappas,Girard,Alur - EMSOFT06

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

- Theory of switched/hybrid linear system applies
- Schedule is an automaton over edges/controllers
- Weiss – EMSOFT08 – Session 5

Approach



$x(t+1) = A_i x(t) + B_i u(t)$
$y(t) = C_i x(t)$
Plant
$u = \tilde{y}$
$\tilde{u} = y$
$\tilde{x}(t+1) = \tilde{A}_i \tilde{x}(t) + \tilde{B}_i \tilde{u}(t)$
$\tilde{y}(t) = \tilde{C}_i \tilde{x}(t)$
Controller
Ideal Semantics
+
Error
-
1
4
Plant 1
2
5
Controller
Plant 2
3
6
7
Implementation Semantics

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 tolerance, routing, scheduling
- Make sure the implementation follows continuous time design

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

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

Approximation Error

(EMOSFT06) The implementation error is exactly equal to :

$$e_{\mathcal{M}}(\rho, \tau, \delta, x(0)) = x(0)^T H^T \hat{O} H x(0)$$

which requires the solution of the Lyapunov equations

$$\hat{O} = \hat{G}_0^T \hat{Q} \hat{G}_0 + \hat{E}_0^T O \hat{E}_0$$
$$O = \hat{E}^T O \hat{E} + \hat{G}^T \hat{Q} \hat{G}.$$

for implementation dependent matrices

Penn
University of Pennsylvania

Example - Models

LTI plant

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \\ \dot{x}_4(t) \end{bmatrix} = \begin{bmatrix} -1020 & -156.3 & 0 & 0 \\ 128 & 0 & 0 & 0 \\ 0 & 0 & -10.2 & -2.002 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \end{bmatrix} + \begin{bmatrix} 8 & 0 \\ 0 & 0 \\ 0 & 0.5 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix}$$

$$\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 4.8828 & 0 & 0 \\ 0 & 0 & 0 & 0.4 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \end{bmatrix}$$

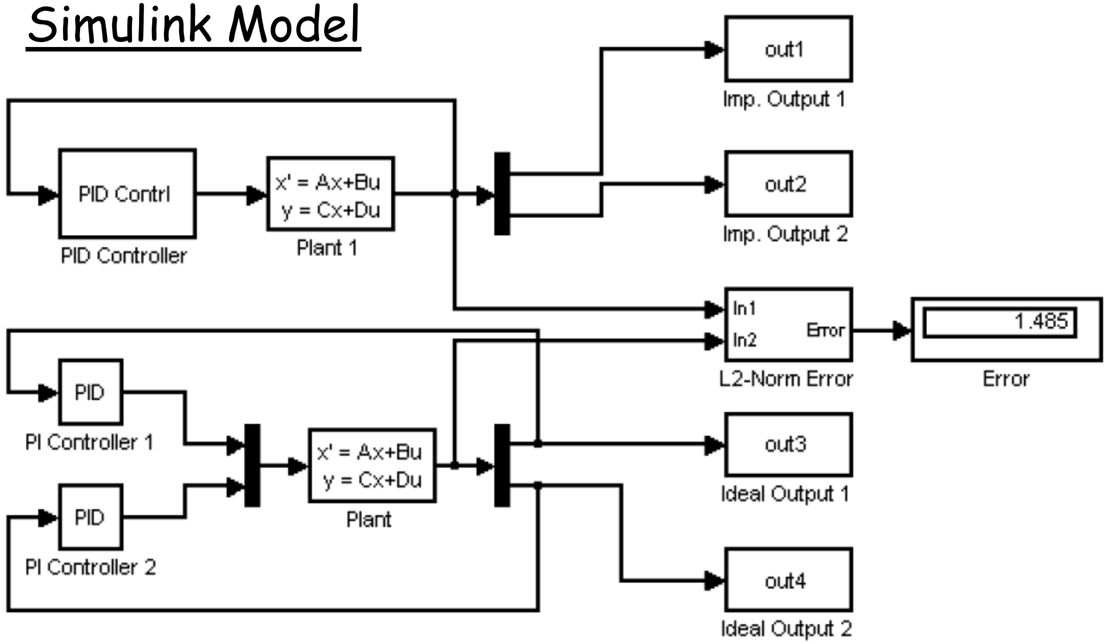
The PID controller

$$K_P = \begin{bmatrix} -116 & 0 \\ 0 & -250 \end{bmatrix}$$

$$K_I = \begin{bmatrix} -480 & 0 \\ 0 & -30 \end{bmatrix}$$

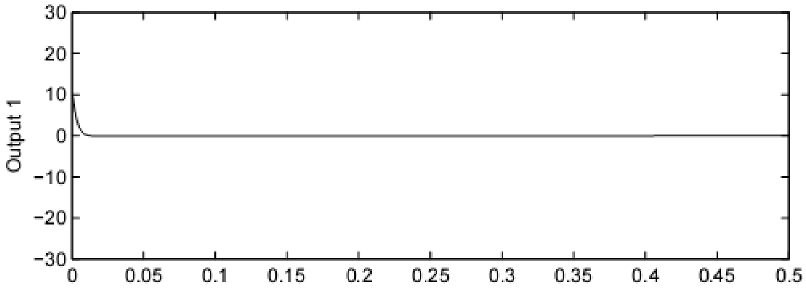
$$K_D = \begin{bmatrix} -0.2 & 0 \\ 0 & -20 \end{bmatrix}$$

Simulink Model

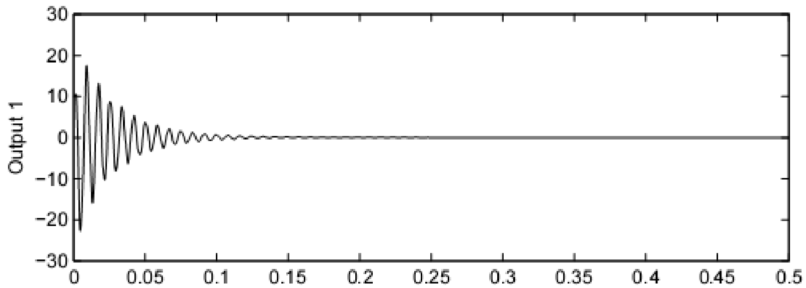


Penn
UNIVERSITY of PENNSYLVANIA

Example - Implementation Errors



Ideal Controller

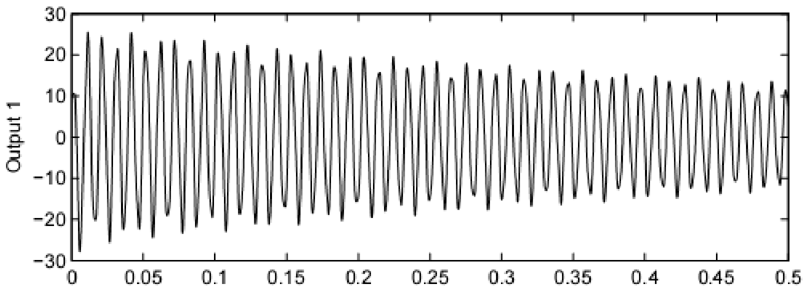


Implementation 2

$\acute{u}_2 = (\mathcal{B}_I\mathcal{B}_1\mathcal{B}_2)^{\omega}$ $\quad \ddot{u}_2(B_j) = 1$ $\quad \hat{\imath}_2 = 0{:}00075\,sec$

Trapezoid & Backward Difference

$e_M(\acute{u}_2; \ddot{u}_2; \hat{\imath}_2; x(0)) = 1{:}9263$

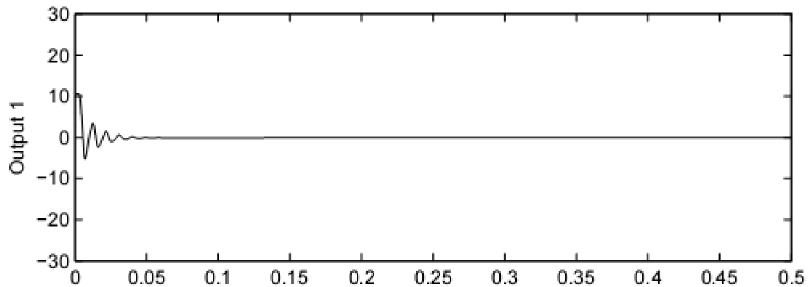


Implementation 1

$\acute{u}_1 = (\mathcal{B}_I\mathcal{B}_1\mathcal{B}_2)^{\omega}$ $\quad \ddot{u}_1(B_j) = 1$ $\quad \hat{\imath}_1 = 0{:}001\,sec$

Euler & Backward Difference

$e_M(\acute{u}_1; \ddot{u}_1; \hat{\imath}_1; x(0)) = 10{:}0058$



Implementation 3

$\acute{u}_3 = (\mathcal{B}_I\mathcal{B}_2\mathcal{B}_1)^{\omega}$ $\quad \ddot{u}_3(B_j) = 1$ $\quad \hat{\imath}_3 = 0{:}001\,sec$

Euler & Backward Difference

$e_M(\acute{u}_3; \ddot{u}_3; \hat{\imath}_3; x(0)) = 0{:}5241$

Penn
University of Pennsylvania

Example – More Results

(ρ, τ, δ)	δ	Integration	Differentiation	ρ	$e_{\mathcal{M}}$ (PI)	$e_{\mathcal{M}}$ (PID)
$(\rho_1, \tau_1, \delta_1)$	0.001	Euler	Backward Diff.	$(\mathcal{B}_I \mathcal{B}_1 \mathcal{B}_2)^\omega$	∞	10.0058
$(\rho_2, \tau_2, \delta_2)$	0.001	Trapezoid	Backward Diff.	$(\mathcal{B}_I \mathcal{B}_1 \mathcal{B}_2)^\omega$	5.3074	7.6896
$(\rho_3, \tau_3, \delta_3)$	0.001	Euler	Backward Diff.	$(\mathcal{B}_I \mathcal{B}_2 \mathcal{B}_1)^\omega$	8.8155	0.5241
$(\rho_4, \tau_4, \delta_4)$	0.001	Euler	Backward Diff.	$(\mathcal{B}_I \mathcal{B}_2 \mathcal{B}_1 \mathcal{B}_1)^\omega$	1.1461	∞
$(\rho_5, \tau_5, \delta_5)$	0.001	Euler	Backward Diff.	$(\mathcal{B}_I \mathcal{B}_2 \mathcal{B}_1 \mathcal{B}_1 \mathcal{B}_1 \mathcal{B}_1)^\omega$	∞	0.63357
$(\rho_6, \tau_6, \delta_6)$	0.001	Trapezoid	Backward Diff.	$(\mathcal{B}_I \mathcal{B}_1 \mathcal{B}_2 \mathcal{B}_1 \mathcal{B}_1 \mathcal{B}_1 \mathcal{B}_1 \mathcal{B}_1 \mathcal{B}_1 \mathcal{B}_1)^\omega$	0.61896	1.6412
$(\rho_7, \tau_7, \delta_7)$	0.00075	Trapezoid	Backward Diff.	$(\mathcal{B}_I \mathcal{B}_1 \mathcal{B}_2)^\omega$	0.75237	1.9263
$(\rho_8, \tau_8, \delta_8)$	0.0005	Trapezoid	Backward Diff.	$(\mathcal{B}_I \mathcal{B}_1 \mathcal{B}_2 \mathcal{B}_1 \mathcal{B}_1 \mathcal{B}_1 \mathcal{B}_1 \mathcal{B}_1 \mathcal{B}_1 \mathcal{B}_1)^\omega$	0.19384	0.51015

- (Poor) Implementation can destabilize the plant
- Good scheduling can improve the quality of the implementation greatly (compare implementations 1 and 3, 4 and 5).

  ⇨ Scheduling has great affect on the overall performance

- Integration and differentiation algorithms can affect the performance (compare implementations 1 and 2).

Source code: www.seas.upenn.edu/~nghiem/publications/2006/emsoft06_code.zip

Penn
UNIVERSITY of PENNSYLVANIA

Example – Is Faster Better?

(ρ,τ,δ)	δ	Integration	Differentiation	ρ	$e_{\mathcal{M}}$ (PI)	$e_{\mathcal{M}}$ (PID)
(ρ_1,τ_1,δ_1)	0.001	Euler	Backward Diff.	$(\mathcal{B}_I\mathcal{B}_1\mathcal{B}_2)^\omega$	∞	10.0058
(ρ_2,τ_2,δ_2)	0.001	Trapezoid	Backward Diff.	$(\mathcal{B}_I\mathcal{B}_1\mathcal{B}_2)^\omega$	5.3074	7.6896
(ρ_3,τ_3,δ_3)	0.001	Euler	Backward Diff.	$(\mathcal{B}_I\mathcal{B}_2\mathcal{B}_1)^\omega$	8.8155	0.5241
(ρ_4,τ_4,δ_4)	0.001	Euler	Backward Diff.	$(\mathcal{B}_I\mathcal{B}_2\mathcal{B}_1\mathcal{B}_1)^\omega$	1.1461	∞
(ρ_5,τ_5,δ_5)	0.001	Euler	Backward Diff.	$(\mathcal{B}_I\mathcal{B}_2\mathcal{B}_1\mathcal{B}_1\mathcal{B}_1\mathcal{B}_1)^\omega$	∞	0.63357
(ρ_6,τ_6,δ_6)	0.001	Trapezoid	Backward Diff.	$(\mathcal{B}_I\mathcal{B}_1\mathcal{B}_2\mathcal{B}_1\mathcal{B}_1\mathcal{B}_1\mathcal{B}_1\mathcal{B}_1\mathcal{B}_1\mathcal{B}_1)^\omega$	0.61896	1.6412
(ρ_7,τ_7,δ_7)	0.00075	Trapezoid	Backward Diff.	$(\mathcal{B}_I\mathcal{B}_1\mathcal{B}_2)^\omega$	0.75237	1.9263
(ρ_8,τ_8,δ_8)	0.0005	Trapezoid	Backward Diff.	$(\mathcal{B}_I\mathcal{B}_1\mathcal{B}_2\mathcal{B}_1\mathcal{B}_1\mathcal{B}_1\mathcal{B}_1\mathcal{B}_1\mathcal{B}_1\mathcal{B}_1)^\omega$	0.19384	0.51015

- For fixed schedule, faster is better (compare implementations 6 and 8)
- Across schedules, faster is not necessarily better (compare implementations 6 and 7)

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 - EMSOFT06

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

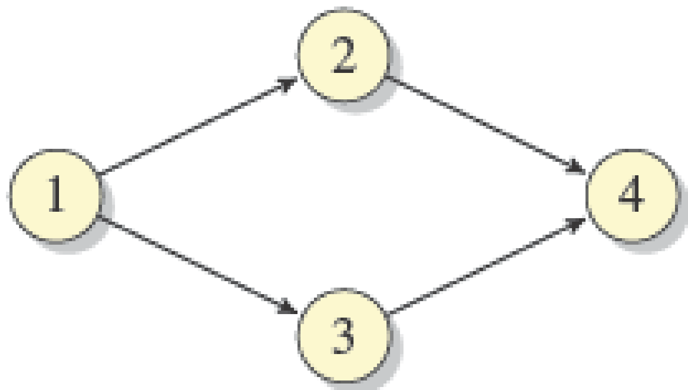
- Theory of switched/hybrid linear system applies
- Schedule is an automaton over edges/controllers
- Weiss – EMSOFT08 – Session 5

Penn
University of Pennsylvania

Non determinism in routing

Given a communication schedule η(t), the effective schedule that acts on the network depends on the status of nodes and channel:

- Set of allowed routing paths is centralized
- Routing decisions are decentralized



(a) Routing graph of node 4

$1 \to 2$	$1 \to 3$	$2 \to 4$	$3 \to 4$	$\cdots$

(b) Superframe schedule

$1 \to 2$		$2 \to 4$		$\cdots$

(c) Effective schedule, case 1

	$1 \to 3$		$3 \to 4$	$\cdots$

(d) Effective schedule, case 2

Penn
UNIVERSITY of PENNSYLVANIA

Key Challenges

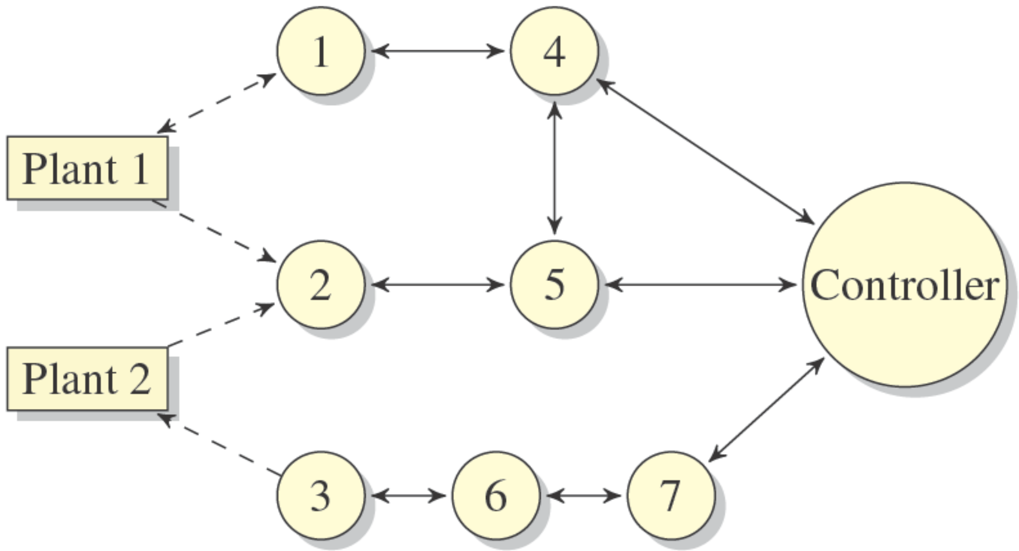
Periodic non-deterministic scheduling (Wireless HART multi-hop)

- Verification : given a schedule, compute the language of effective schedules and verify stability
- Design: compute the set of schedules that satisfy control specifications (exponential convergence rate)

Aperiodic scheduling

- Verification : given a schedule, verify whether the system is stable
- Design: compute a regular language of scheduling that satisfy control specifications (exponential convergence rate)

Compositional analysis

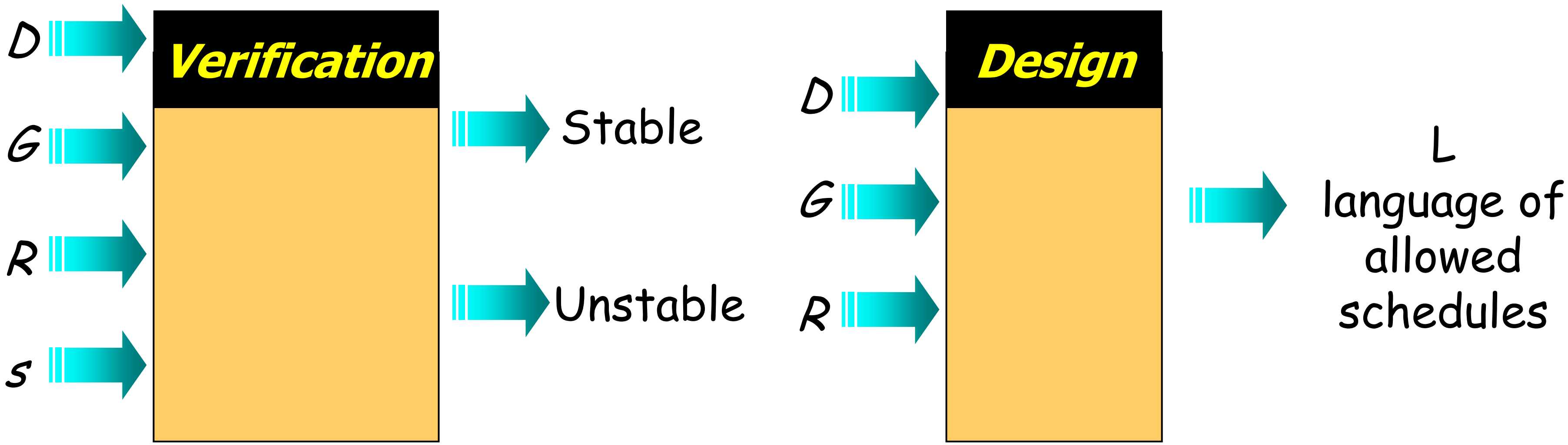


$$
\begin{aligned}
A_1 &= \text{SwitchedSystem}_1[\text{controlLoops}, \text{netTopology}, \text{routing}];\\
\text{lang}_1 &= \text{ExpStabLang}[A_1, 5, 1];\\
\\
A_2 &= \text{SwitchedSystem}_2[\text{controlLoops}, \text{netTopology}, \text{routing}];\\
\text{lang}_2 &= \text{ExpStabLang}[A_2, 5, 1];\\
\\
\text{inter} &= \text{LangIntersection}[\text{lang}_1, \text{lang}_2];\\
\text{schedule} &= \text{ExtractShortestPeriodicSchedule}[\text{inter}];
\end{aligned}
$$

Penn
UNIVERSITY of PENNSYLVANIA

A tool

- Plants/Controllers $D = (P_1, \ldots P_n, C_1, \ldots C_n)$,
- Radio connectivity Graph $G = (V,E)$
- Routing $R : I \cup O \rightarrow 2^{V^*} \setminus \{\emptyset\}$
- Schedule $s = (\eta, \mu)$



The End

<u>Conclusions</u>

WirelesHART protocol allows implementation error analysis

Semantics of switched linear systems

Given periodic schedules, implementation error can be computed exactly

For nondeterministic routing, scheduling languages can be obtained

Compositional admission control policies are possible

<u>Future work</u>

Exploit matrix structure of switched linear system

Apply tool to practical applications (ABB, Honeywell)

Consider sensor, network uncertainty

Control the network!