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

Challenge: Close the loop around wireless sensor networks

Impact of Delays
Impact of Scheduling
Impact of Routing
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 network or the addition control control loops
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 $D = (P_1, \ldots, P_n, C_1, \ldots, C_n)$, 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

\[ \Omega_{Plant} \]

Plant 1

Plant 2

Controller 1

Controller 2

\[ \Omega_{Con} \]

Communication schedule

Computation schedule
Semantics in each time slot

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

1a, 4a | 2a, 5a | 4a, Ca | 5a, Ce | 2b, 5b | 5b, Cd | Cb, 4b | 4b, 1b | Ce, 7a | 7a, 6a | 6a, 3a | ...
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.

\[
A_l(e, m) := \begin{pmatrix}
A_{\text{Plant}} & B_{\text{Plant}} \cdot O_{\text{Plant}} & 0 \\
I^T_{\text{Plant}} \cdot C_{\text{Plant}} & \text{Adj}(\langle V_R, e \rangle)^T & O^T_{\text{Con}} \cdot C_{\text{Con}(m)} \\
0 & B_{\text{Con}(m)} \cdot I_{\text{Con}} & A_{\text{Con}(m)}
\end{pmatrix}
\]
Algebraic representations of the graph are very useful.

Size of matrices depends on the network and hence on the routing.
Mathematical Tool

- **Control Loops**

\[
\text{Plant}[1] = \{A_p, B_p, C_p\}; \\
\text{Controller}[1] = \{A_c, B_c, C_c\}; \\
\text{Plant}[2] = \text{Plant}[1]; \\
\text{Controller}[2] = \text{Controller}[1]; \\
\text{controlLoops} = \{\{\text{Plant}[1], \text{Controller}[1]\}, \{\text{Plant}[2], \text{Controller}[2]\}\};
\]

- **Wireless Network**

\[
\text{networkTopology} := \{1 \rightarrow 4, 4 \rightarrow 1, 4 \rightarrow 5, 5 \rightarrow 4, 4 \rightarrow C, \\
C \rightarrow 4, 2 \rightarrow 5, 5 \rightarrow 2, 5 \rightarrow C, C \rightarrow 5, 3 \rightarrow 6, 6 \rightarrow 3, 6 \rightarrow 7, 7 \rightarrow 6, 7 \rightarrow C, C \rightarrow 7\}
\]

- **Routing**

\[
\text{routin}[y_{1,1}] = \{1, 4, C\}; \\
\text{routin}[y_{1,2}] = \{2, 5, C\}; \\
\text{routin}[u_{1,1}] = \{C, 4, 1\};
\]

\[
\text{routin}[y_{2,1}] = \{2, 5, C\}; \\
\text{routin}[y_{2,2}] = \{2, 5, C\}; \\
\text{routin}[u_{2,1}] = \{C, 7, 6, 3\};
\]

```
In[36]:= \text{SW} = \text{SwitchedSystem}_i[\text{controlLoops}, \text{networkTopology}, \text{routing}]; \\
\text{edges} = \{(1, y_{1,1}) \rightarrow (4, y_{1,1})\}; \\
\text{M} = \text{SW}[\text{edges}, \text{Active}, \\
\begin{bmatrix}
1 & 1/20 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1/20 & 0 & 0 & 0 & 0 & 0 & 0 & 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 & 1 & 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 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 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{bmatrix}
```
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

**Ideal Semantics**

\[
x(t + 1) = A_i x(t) + B_i u(t)
\]

\[
y(t) = C_i x(t)
\]

\[
\tilde{x}(t + 1) = \tilde{A}_i \tilde{x}(t) + \tilde{B}_i \tilde{u}(t)
\]

\[
\tilde{y}(t) = \tilde{C}_i \tilde{x}(t)
\]

**Implementation Semantics**

\[
u = y
\]

\[
\tilde{u} = y
\]

Error

Plant 1

Plant 2

Controller
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:

\[(x(t), y(t), u(t), z(t)) = [\mathcal{M}](x(0))\]
\[(\tilde{x}(t), \tilde{y}(t), \tilde{u}(t), \tilde{z}(t)) = [\mathcal{M}](\rho, \tau, \delta)(x(0))\]

\[e_{\mathcal{M}}(\rho, \tau, \delta, x(0)) = \int_{0}^{+\infty} \|y(t) - \tilde{y}(t)\|_2^2 dt\]

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:

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

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:

\[ e_M(\rho, \tau, \delta, x(0)) = x(0)^T H^T \hat{O} H x(0) \]

which requires the solution of the Lyapunov equations

\[
\begin{align*}
\dot{\hat{O}} &= \hat{G}_0^T \hat{Q} \hat{G}_0 + \hat{E}_0^T \hat{O} \hat{E}_0 \\
\dot{\hat{O}} &= \hat{E}_0^T \hat{O} \hat{E} + \hat{G}_0^T \hat{Q} \hat{G}.
\end{align*}
\]

for implementation dependent matrices
Example - Models

LTI plant

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

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

The PID controller

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

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

\[
K_D = \begin{bmatrix}
-0.2 & 0 \\
0 & -20
\end{bmatrix}
\]
Example - Implementation Errors

Ideal Controller

Implementation 1

\[ u_1 = (B_1 B_2)^\omega, \quad \dot{u}_1(B_1) = 1, \quad \dot{t}_1 = 0.001 \text{sec} \]
Euler & Backward Difference
\[ e_M(u_1; \dot{u}_1; \dot{t}_1; x(0)) = 10.0058 \]

Implementation 2

\[ u_2 = (B_1 B_2)^\omega, \quad \dot{u}_2(B_1) = 1, \quad \dot{t}_2 = 0.00075 \text{sec} \]
Trapezoid & Backward Difference
\[ e_M(u_2; \dot{u}_2; \dot{t}_2; x(0)) = 1.9263 \]

Implementation 3

\[ u_3 = (B_1 B_2)^\omega, \quad \dot{u}_3(B_1) = 1, \quad \dot{t}_3 = 0.001 \text{sec} \]
Euler & Backward Difference
\[ e_M(u_3; \dot{u}_3; \dot{t}_3; x(0)) = 0.5241 \]
Example – More Results

<table>
<thead>
<tr>
<th>$(\rho, \tau, \delta)$</th>
<th>$\delta$</th>
<th>Integration</th>
<th>Differentiation</th>
<th>$\rho$</th>
<th>$\epsilon_M$ (PI)</th>
<th>$\epsilon_M$ (PID)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(\rho_1, \tau_1, \delta_1)$</td>
<td>0.001</td>
<td>Euler</td>
<td>Backward Diff.</td>
<td>$(B_1B_1B_2)\omega$</td>
<td>$\infty$</td>
<td>10.0058</td>
</tr>
<tr>
<td>$(\rho_2, \tau_2, \delta_2)$</td>
<td>0.001</td>
<td>Trapezoid</td>
<td>Backward Diff.</td>
<td>$(B_1B_2B_2)\omega$</td>
<td>5.3074</td>
<td>7.6896</td>
</tr>
<tr>
<td>$(\rho_3, \tau_3, \delta_3)$</td>
<td>0.001</td>
<td>Euler</td>
<td>Backward Diff.</td>
<td>$(B_1B_2B_1)\omega$</td>
<td>8.8155</td>
<td>0.5241</td>
</tr>
<tr>
<td>$(\rho_4, \tau_4, \delta_4)$</td>
<td>0.001</td>
<td>Euler</td>
<td>Backward Diff.</td>
<td>$(B_1B_2B_1B_1)\omega$</td>
<td>1.1461</td>
<td>$\infty$</td>
</tr>
<tr>
<td>$(\rho_5, \tau_5, \delta_5)$</td>
<td>0.001</td>
<td>Euler</td>
<td>Backward Diff.</td>
<td>$(B_1B_2B_1B_1B_1)\omega$</td>
<td>$\infty$</td>
<td>0.63357</td>
</tr>
<tr>
<td>$(\rho_6, \tau_6, \delta_6)$</td>
<td>0.001</td>
<td>Trapezoid</td>
<td>Backward Diff.</td>
<td>$(B_1B_1B_2B_2B_1B_1B_1B_1B_1)\omega$</td>
<td>0.61896</td>
<td>1.6412</td>
</tr>
<tr>
<td>$(\rho_7, \tau_7, \delta_7)$</td>
<td>0.00075</td>
<td>Trapezoid</td>
<td>Backward Diff.</td>
<td>$(B_1B_1B_2)\omega$</td>
<td>0.75237</td>
<td>1.9263</td>
</tr>
<tr>
<td>$(\rho_8, \tau_8, \delta_8)$</td>
<td>0.0005</td>
<td>Trapezoid</td>
<td>Backward Diff.</td>
<td>$(B_1B_1B_2B_1B_1B_1B_1B_1B_1)\omega$</td>
<td>0.19384</td>
<td>0.51015</td>
</tr>
</tbody>
</table>

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

\[ \Rightarrow \quad \text{Scheduling has great affect on the overall performance} \]

- 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](http://www.seas.upenn.edu/~nghiem/publications/2006/emsoft06_code.zip)
### Example – Is Faster Better?

<table>
<thead>
<tr>
<th>$(\rho, \tau, \delta)$</th>
<th>$\delta$</th>
<th>Integration</th>
<th>Differentiation</th>
<th>$\rho$</th>
<th>$e_M$ (PI)</th>
<th>$e_M$ (PID)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(\rho_1, \tau_1, \delta_1)$</td>
<td>0.001</td>
<td>Euler</td>
<td>Backward Diff.</td>
<td>$\omega$</td>
<td>$\infty$</td>
<td>10.0058</td>
</tr>
<tr>
<td>$(\rho_2, \tau_2, \delta_2)$</td>
<td>0.001</td>
<td>Trapezoid</td>
<td>Backward Diff.</td>
<td>$\omega$</td>
<td>5.3074</td>
<td>7.6896</td>
</tr>
<tr>
<td>$(\rho_3, \tau_3, \delta_3)$</td>
<td>0.001</td>
<td>Euler</td>
<td>Backward Diff.</td>
<td>$\omega$</td>
<td>8.8155</td>
<td>0.5241</td>
</tr>
<tr>
<td>$(\rho_4, \tau_4, \delta_4)$</td>
<td>0.001</td>
<td>Euler</td>
<td>Backward Diff.</td>
<td>$\omega$</td>
<td>1.1461</td>
<td>$\infty$</td>
</tr>
<tr>
<td>$(\rho_5, \tau_5, \delta_5)$</td>
<td>0.001</td>
<td>Euler</td>
<td>Backward Diff.</td>
<td>$\omega$</td>
<td>$\infty$</td>
<td>0.63357</td>
</tr>
<tr>
<td>$(\rho_6, \tau_6, \delta_6)$</td>
<td>0.001</td>
<td>Trapezoid</td>
<td>Backward Diff.</td>
<td>$\omega$</td>
<td>0.61896</td>
<td>1.6412</td>
</tr>
<tr>
<td>$(\rho_7, \tau_7, \delta_7)$</td>
<td>0.00075</td>
<td>Trapezoid</td>
<td>Backward Diff.</td>
<td>$\omega$</td>
<td>0.75237</td>
<td>1.9263</td>
</tr>
<tr>
<td>$(\rho_8, \tau_8, \delta_8)$</td>
<td>0.0005</td>
<td>Trapezoid</td>
<td>Backward Diff.</td>
<td>$\omega$</td>
<td>0.19384</td>
<td>0.51015</td>
</tr>
</tbody>
</table>

- 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
Non determinism in routing

Given a communication schedule \( \eta(t) \), 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
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

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

Conclusions
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

Future work
Exploit matrix structure of switched linear system
Apply tool to practical applications (ABB, Honeywell)
Consider sensor, network uncertainty
Control the network!