Co-design of Implementation Platform and Control in NCS Using an Overrun Framework

Damoon Soudbakhsh[1], Linh T.X. Phan[2], Anuradha M. Annaswamy[1], Oleg Sokolsky[2]
[1] *Massachusetts Institute of Technology* [2] *University of Pennsylvania*

Objective

Efficient implementation of multiple control applications in a Network Control System (NCS)

Approach

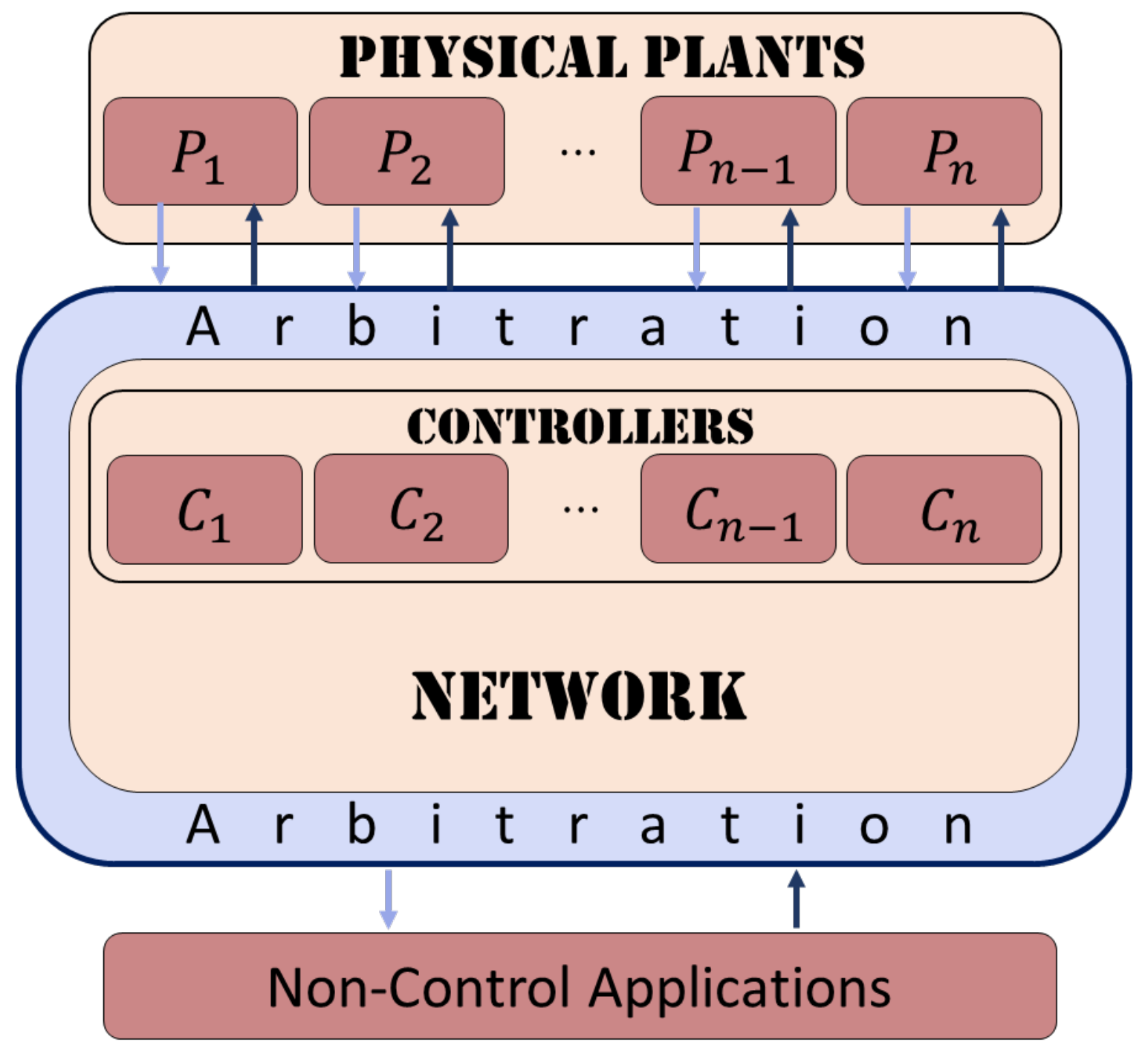
- Co-design of control and implementation platform that delivers high quality of control with efficient resource utilization

Rationale

- Several complex systems consist of multiple control and non-control applications mapped onto a NCS
- Resource sharing introduces delays
- Significant transparency and flexibility available in platform design
- Powerful analytical methods exist both for stability of switched delay-systems in control theory and for estimation of end-to-end delays in real-time calculus
- Co-design of control and platform can leverage these methods

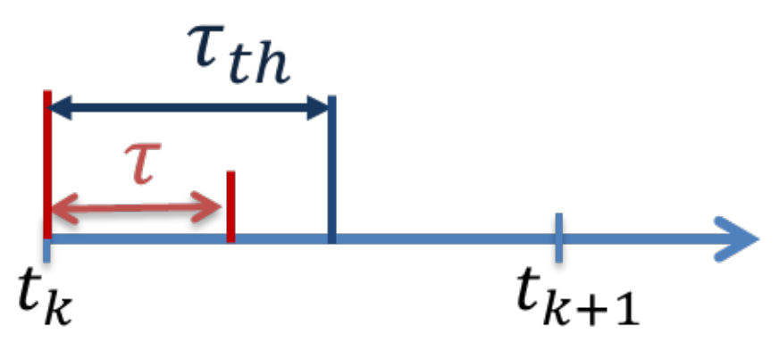
Problem Statement

- Control multiple applications with shared resources
- Co-design of controllers and implementation platform
- Approach – ANCS (Arbitrated Network Control Systems)
  - use of Arbitration
  - stability analysis of Switched Systems
  - automata models and verification-based analysis
- Estimate delays
- Define delay thresholds

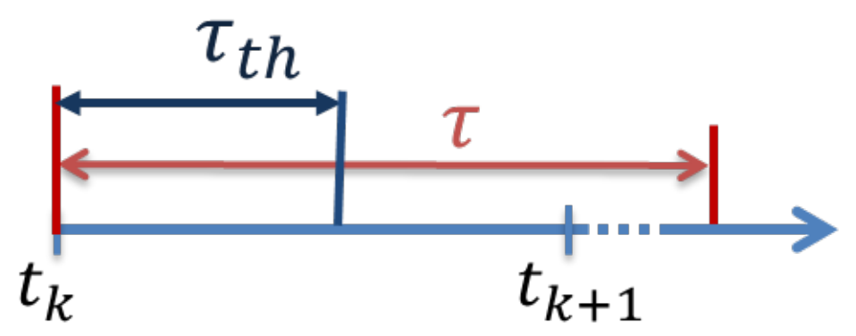


Shared Resources and Delay Threshold

- Nominal Case - $\tau \leq \tau_{th}$



- Overrun Case - $\tau > \tau_{th}$

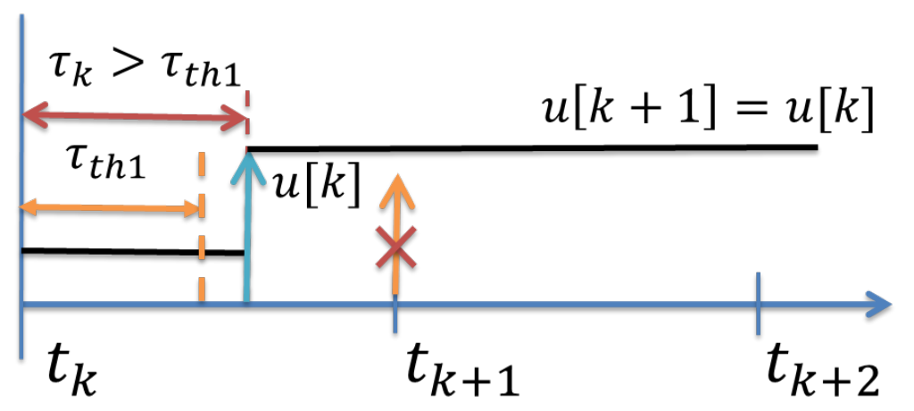


Overrun Framework

Overrun Strategies

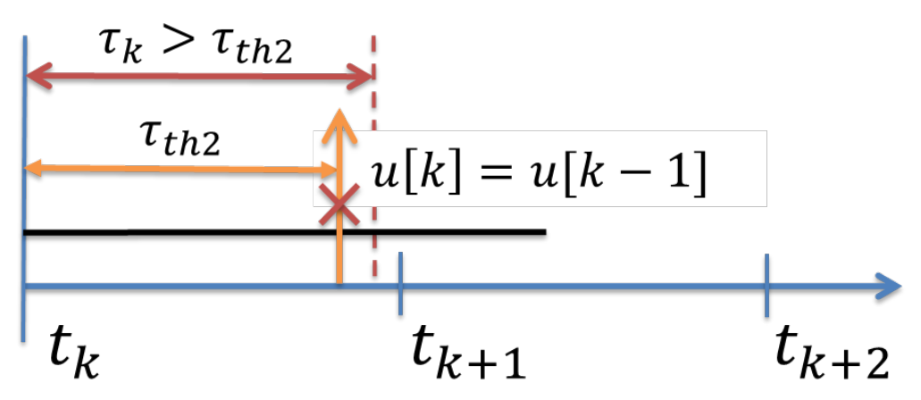
- Nominal Mode
  - stabilizing controller (ex. LQR)
- Skip Strategy

  If delay exceeds a small threshold τ_{th1}, skip next message
  - $u[k] = u_{LQR}[k]$, if $\tau_k \leq \tau_{th1}$ $X[k+1] = \Gamma_n X[k]$
  - $u[k+1] = u[k]$, if $\tau_k > \tau_{th1}$ $X[k+2] = \Gamma_s X[k]$



- Abort Strategy

  If delay exceeds a large threshold τ_{th2}, abort computations of current message
  - $u[k] = u_{LQR}[k]$, if $\tau_k \leq \tau_{th1}$ $X[k+1] = \Gamma_n X[k]$
  - $u[k] = u[k-1]$, if $\tau_k > \tau_{th2}$ $X[k+1] = \Gamma_a X[k]$



Stability with Skip and Abort Strategies

Theorem: The system

$$X[k+N] = \Gamma_a^{s_p}\Gamma_s^{j_p}\Gamma_n^{i_p}\cdots\Gamma_a^{r_1}\Gamma_s^{j_1}\Gamma_n^{i_1}X[k]$$

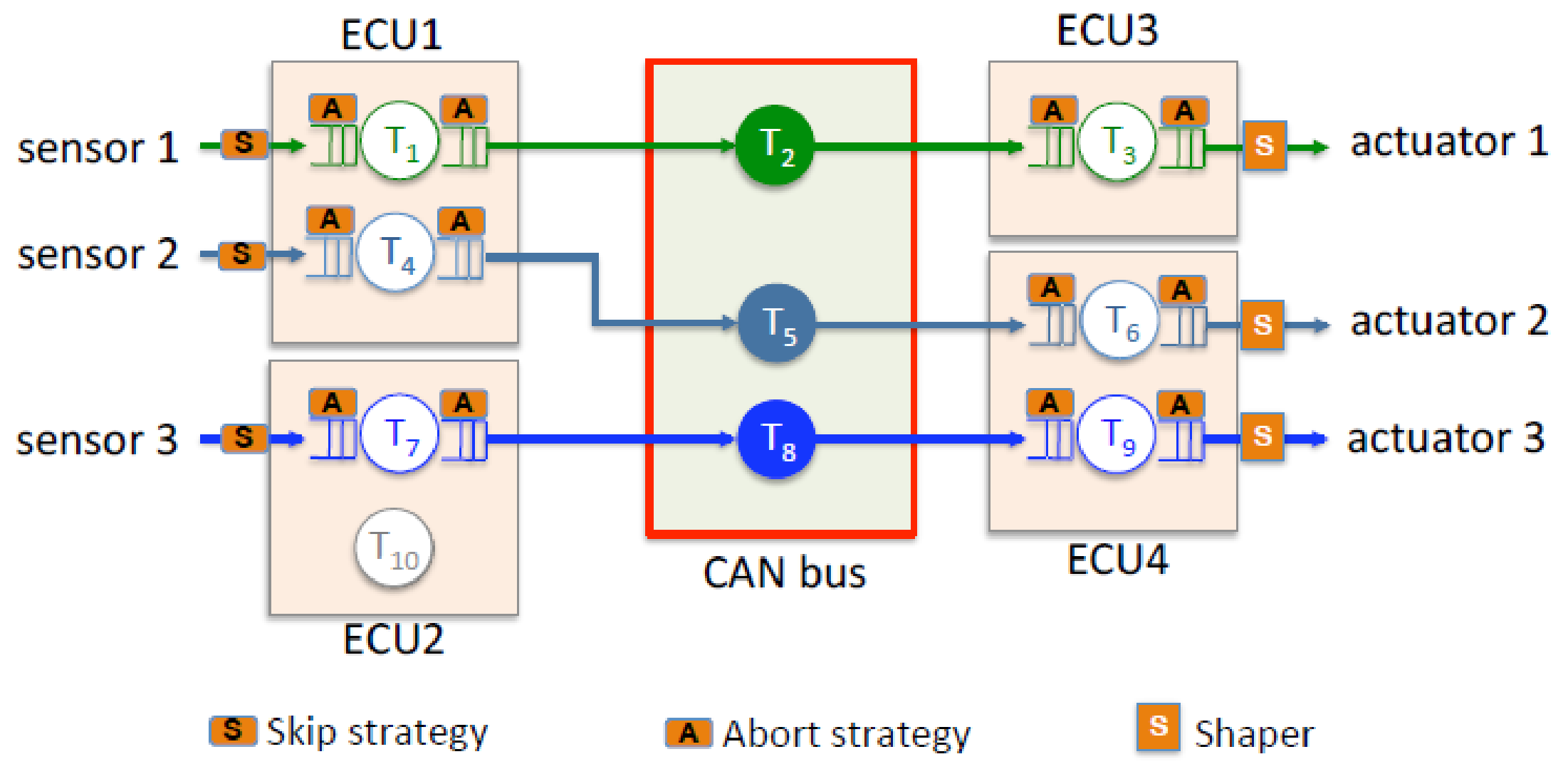
is stable if

$$\gamma \overset{\text{def}}{=} \gamma_n^{1-2r_{skip}-r_{abort}}\gamma_s^{r_{skip}}\gamma_a^{r_{abort}} \leq 1,$$

where γ_n, γ_s, and γ_a are the combined decay rates of the system in nominal, skip, and abort modes, and r_{skip} and r_{abort} are maximum skip and abort rates, respectively.

Platform Architecture

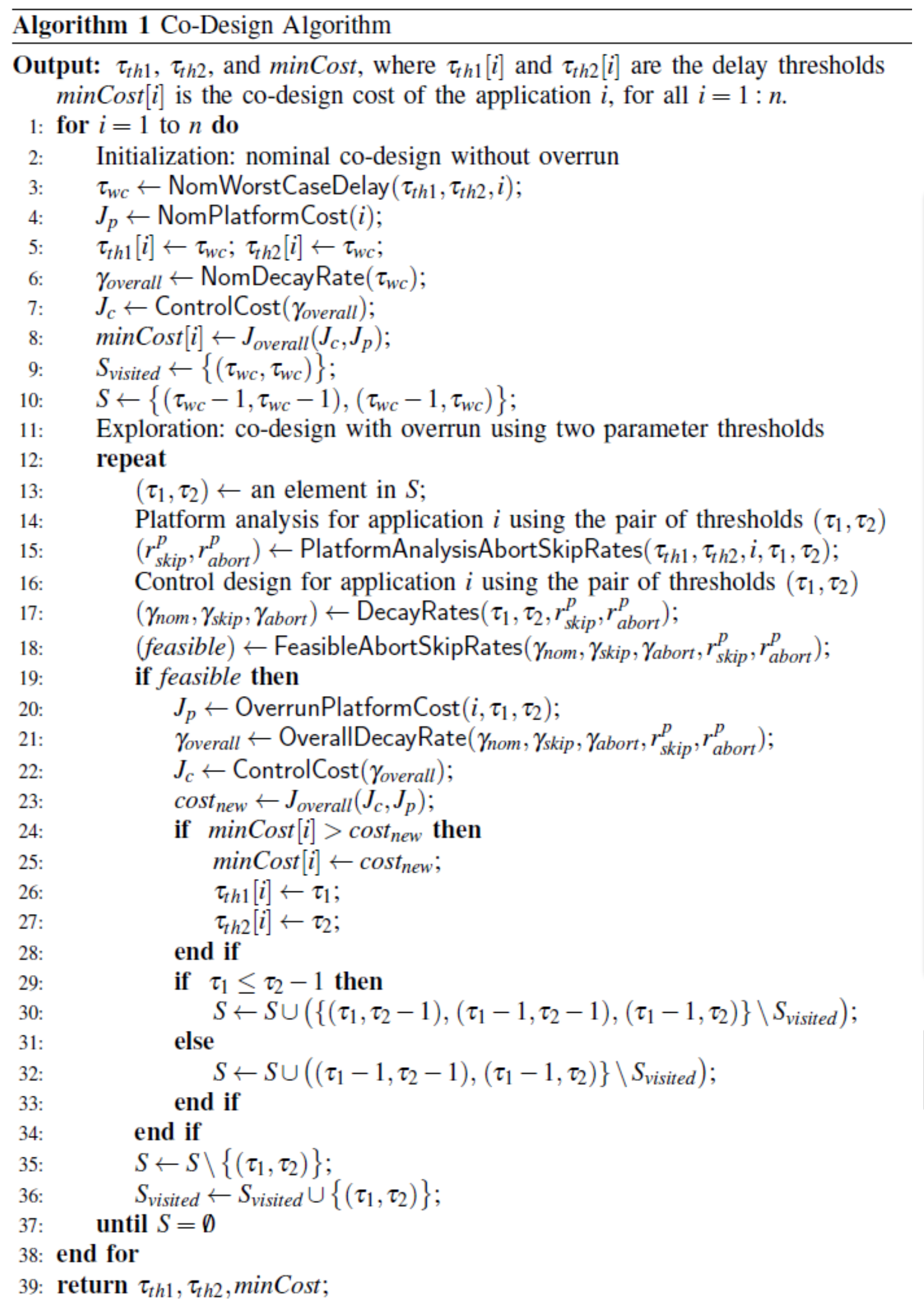
- Sequence of processing elements connected by buffers
  - PEs: tasks on an ECU or messages on the network



Platform Design

- Introduce buffer control into the platform
  - proactively drop delayed samples
- Develop automata models and verification-based analysis for the system under overrun design
- Results in
  - reduced resource consumption
  - reduced delays of remaining samples

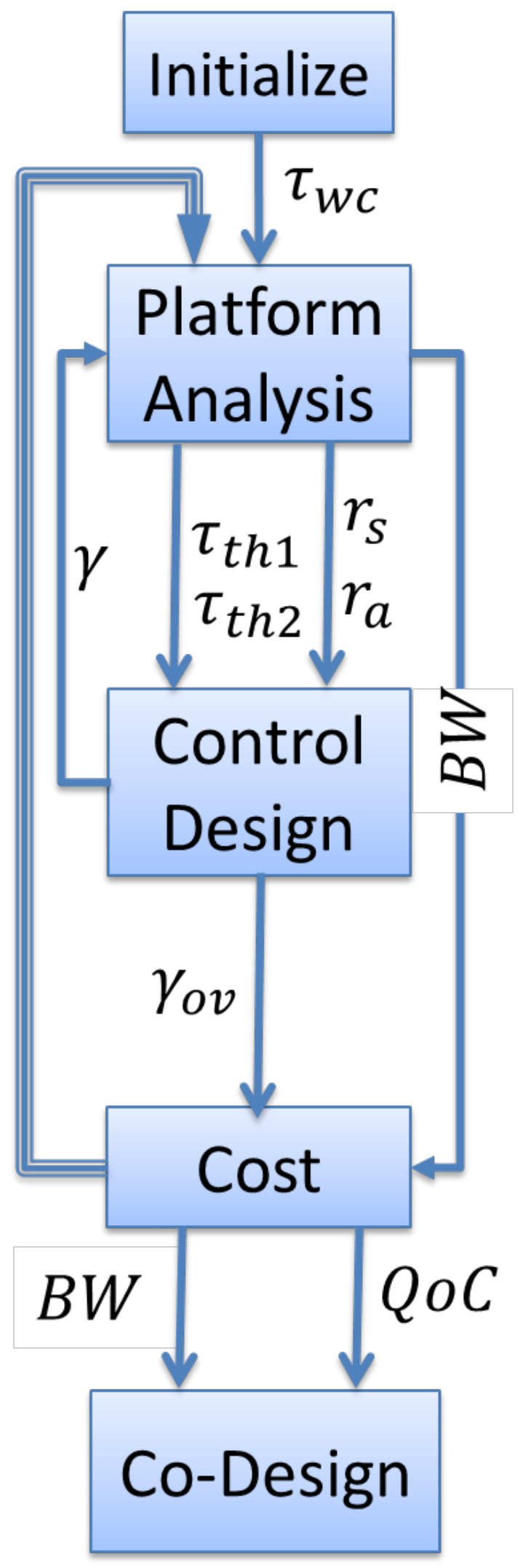
Co-Design Algorithm



Algorithm 1 Co-Design Algorithm

Output: τ_{th1}, τ_{th2}, and *minCost*, where $\tau_{th1}[i]$ and $\tau_{th2}[i]$ are the delay thresholds *minCost*$[i]$ is the co-design cost of the application i, for all $i = 1:n$.

```
for $i = 1$ to $n$ do
    Initialization: nominal co-design without overrun
    $\tau_{wc} \leftarrow \mathrm{NomWorstCaseDelay}(\tau_{th1}, \tau_{th2}, i)$;
    $J_p \leftarrow \mathrm{NomPlatformCost}(i)$;
    $\tau_{th1}[i] \leftarrow \tau_{wc}$; $\tau_{th2}[i] \leftarrow \tau_{wc}$;
    $\gamma_{overall} \leftarrow \mathrm{NomDecayRate}(\tau_{wc})$;
    $J_c \leftarrow \mathrm{ControlCost}(\gamma_{overall})$;
    $minCost[i] \leftarrow J_{overall}(J_c, J_p)$;
    $S_{visited} \leftarrow \{(\tau_{wc}, \tau_{wc})\}$;
    $S \leftarrow \{(\tau_{wc}-1, \tau_{wc}-1), (\tau_{wc}-1, \tau_{wc})\}$;
    Exploration: co-design with overrun using two parameter thresholds
    repeat
        $(\tau_1, \tau_2) \leftarrow$ an element in $S$;
        Platform analysis for application $i$ using the pair of thresholds $(\tau_1, \tau_2)$
        $(r^p_{skip}, r^p_{abort}) \leftarrow \mathrm{PlatformAnalysisAbortSkipRates}(\tau_{th1}, \tau_{th2}, i, \tau_1, \tau_2)$;
        Control design for application $i$ using the pair of thresholds $(\tau_1, \tau_2)$
        $(\gamma_{nom}, \gamma_{skip}, \gamma_{abort}) \leftarrow \mathrm{DecayRates}(\tau_1, \tau_2, r^p_{skip}, r^p_{abort})$;
        $(feasible) \leftarrow \mathrm{FeasibleAbortSkipRates}(\gamma_{nom}, \gamma_{skip}, \gamma_{abort}, r^p_{skip}, r^p_{abort})$;
        if $feasible$ then
            $J_p \leftarrow \mathrm{OverrunPlatformCost}(i, \tau_1, \tau_2)$;
            $\gamma_{overall} \leftarrow \mathrm{OverallDecayRate}(\gamma_{nom}, \gamma_{skip}, \gamma_{abort}, r^p_{skip}, r^p_{abort})$;
            $J_c \leftarrow \mathrm{ControlCost}(\gamma_{overall})$;
            $cost_{new} \leftarrow J_{overall}(J_c, J_p)$;
            if $minCost[i] > cost_{new}$ then
                $minCost[i] \leftarrow cost_{new}$;
                $\tau_{th1}[i] \leftarrow \tau_1$;
                $\tau_{th2}[i] \leftarrow \tau_2$;
            end if
            if $\tau_1 \leq \tau_2 - 1$ then
                $S \leftarrow S \cup (\{(\tau_1, \tau_2-1), (\tau_1-1, \tau_2-1), (\tau_1-1, \tau_2)\} \setminus S_{visited})$;
            else
                $S \leftarrow S \cup ((\tau_1-1, \tau_2-1), (\tau_1-1, \tau_2)\} \setminus S_{visited})$;
            end if
        end if
        $S \leftarrow S \setminus \{(\tau_1, \tau_2)\}$;
        $S_{visited} \leftarrow S_{visited} \cup \{(\tau_1, \tau_2)\}$;
    until $S = \emptyset$
end for
return $\tau_{th1}, \tau_{th2}, minCost$;
```



Case Study

Goal:

- Simultaneous control of three applications
- Three applications on an ECU

Results

- Co-design computes small (τ_{th1}) and large delay thresholds (τ_{th2}) (see table)
- Desired control performance achieved for all applications
- Co-design ensures resource savings (ex. reduced bandwidth)

Results

Baseline Design versus Co-Design

	Baseline Design	Co-Design	
	Worst-case delay	Small (τ_{th1})	Small (τ_{th2})
App1 –Delay (ms)	10	7	10
App2 –Delay (ms)	13	8	12
App2 –Delay (ms)	18	17	18
Bandwidth (BW)	**1.7**	**1.0**	

Summary & Conclusions

Proposed Co-design of ANCS

Includes an Overrun Framework
- much less conservative than worst-case delay designs
- optimal control performance
- optimal resource utilization

Two-parameter model that allows
- small and large delays

Switches between strategies that
- uses, skips, or aborts a given message

A co-design algorithm proposed
- to optimize small and large delays;
- ensures
  - desired control performance
  - optimal resource utilization

An automotive case study proposed for validation.

Acknowledgement

This work was supported in part by the NSF Grant No. ECCS-1135815 via the CPS initiative.

References

1. Soudbakhsh D., Phan L.X, Annaswamy A.M., and Sokolsky O., “Co-Design of Arbitrated Network Control Systems with Overrun Strategies,” IEEE Transactions on Control of Network Systems (submitted).
2. Annaswamy A.M., Soudbakhsh D., Schneider R., Goswami D., Chakraborty S., “Arbitrated Network Control Systems: A co-design of control and platform for cyber-physical systems,” Control of Cyber-Physical Systems, Lecture Notes in Control and Information Sciences, Vol. 449, Ed: D.C. Tarraf, Springer Verlag, 2013.
3. Soudbakhsh D., Phan L.X, Sokolsky O., Lee I., and Annaswamy A., “Co-design of control and platform with dropped signals,” The 4th ACM/IEEE International Conference on Cyber-Physical Systems [ICCPS’13], Philadelphia, PA, 2013.