Cherry-Picking Control Nodes and Sensors in Dynamic Networks

Mixed-Integer Programming, Heuristics, Challenges

Ahmad F. Taha, with Sebastian Nugroho, Nikolaos Gatsis, Ram Krishnan

March 2, 2018

Electrical and Computer Engineering, UT San Antonio

Introduction, Motivation

Ubiquity of Sensors & Actuators in Cyber-Physical Systems

- Sensors and actuators* (SaA) are ubiquitous in CPSs
- By 2050, we'll have tens of billions of SaAs between smart grids, transportation networks, and water distribution networks
- How to activate/deactivate SaAs depending on the *cyber-physical state* of the infrastructure?
- How to drive a network from one state to another via real-time SaA selection?

*Actuators and control nodes have the same meaning—actuators is an *old* word.

SaA Selection Applications

Smart Power Grids

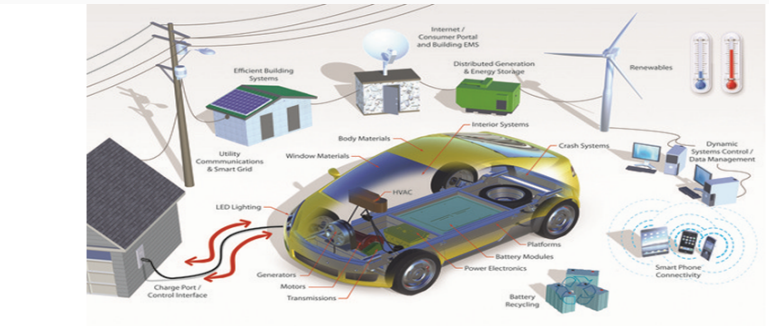
- Actuator selection: distributed energy resources
- Sensor selection: smart meter and PMU data

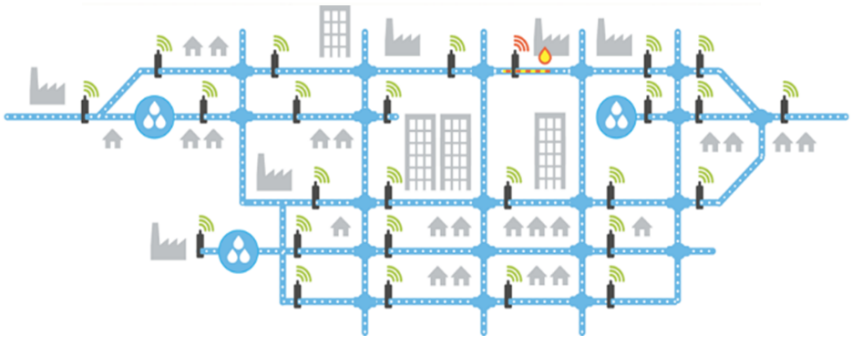
Water Distribution Networks

- Actuator selection: opening/closing valves, releasing contamination
- Sensor selection: managing smart mobile water sensors

Transportation Networks

- Traffic light control, EV charging





Genetic Regulatory Networks

- Choose genes to measure

SaA Selection Problem is Very Diverse

- Different perspectives for the sensor/actuator selection problem
- Planning problems (years–decades as time horizon) focused on SaA selection/placement
- Operation problems (minutes–days)
- **Real-time control** (msecs–hours)—most interests, lots of interest
- Renewed interest in the context of networks—not only control-theory
- **Research objective:** Focusing on the time-varying sensor/actuator selection in dynamic networks
    * Figure out a general framework to deal with this problem
- Most literature focuses on simple linear networks, ignores nonlinearity, disturbances, time-varying nature of networks

Uncertain CPS Model

Time-Varying Uncertain CPS Model

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}^j \boldsymbol{x}(t) + \boldsymbol{B}_u^j \boldsymbol{u}(t) + \boldsymbol{B}_w^j \boldsymbol{w}(t) + \boldsymbol{B}_f^j \boldsymbol{f}(\boldsymbol{x}(t)),$$
$$\boldsymbol{y}(t) = \boldsymbol{C}^j \boldsymbol{x}(t)(t) + \boldsymbol{D}_u^j \boldsymbol{u}(t) + \boldsymbol{D}_v^j \boldsymbol{v}(t)$$

- $\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{y}$ denote the state, control input, and measured output
- $\boldsymbol{w}$ and $\boldsymbol{v}$ are the unknown inputs such as:
  - disturbances, parametric uncertainty, data attacks, sensor faults
- CPS has a total of N subsystems, with a total of n_x states, n_u control inputs, n_y outputs
- j is the time-period
- Dynamics are faster than the change in the CPS topology across j

CPS Model with Sensor and Actuator (SaA) Selection

Time-Varying SaA Selection Model

$$\text{CPSDynamics}: \quad \dot{\boldsymbol{x}}(t) = \boldsymbol{A}^j \boldsymbol{x}(t) + \boldsymbol{B}_u^j \boldsymbol{\Pi}^j \boldsymbol{u}(t) + \boldsymbol{B}_w^j \boldsymbol{w}(t) + \boldsymbol{B}_f^j \boldsymbol{f}(\boldsymbol{x}(t))$$
$$\boldsymbol{y}(t) = \boldsymbol{\Gamma}^j \boldsymbol{C}^j \boldsymbol{x}(t) + \boldsymbol{D}_u^j \boldsymbol{u}(t) + \boldsymbol{D}_v^j \boldsymbol{v}(t)$$

- $\boldsymbol{\Gamma}^j$ and $\boldsymbol{\Pi}^j$ are binary variables for the SaA selection
  - $\boldsymbol{\Pi}^j = \text{blkdiag}(\pi_1^j \boldsymbol{I}_{n_{u_1}}, \ldots, \pi_N^j \boldsymbol{I}_{n_{u_N}})$ places vector $\boldsymbol{\pi}^j$ in a block diagonal matrix
- **Objective:** Find the optimal combination of $\boldsymbol{\Gamma}^j$ and $\boldsymbol{\Pi}^j$ such that the system obeys certain physical properties, dynamic metrics
  - Common Metrics: minimum energy, robustness, boundedness, asymptotic stability

High-Level Research Problem

Problem Formulation

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{T_f} \left\{ c_\pi(\pi_j) + c_\gamma(\gamma_j) + \mathrm{CtrlObj}_j + \mathrm{EstObj}_j \right\} \\ \text{subject to} \quad & \mathrm{CPSDynamics} \\ & \pi_j \in \mathcal{A} \subset \{0,1\}^N, \gamma_j \in \mathcal{S} \subset \{0,1\}^N \\ & \mathrm{ControlConstraints}(\pi_j) \\ & \mathrm{EstimationConstraints}(\gamma_j) \end{aligned}$$

- $\mathrm{CtrlObj}, \mathrm{EstObj}$: Quantify the needed estimation/control metrics
- $\mathcal{A}, \mathcal{S}$ represent logistic constraints on SaA selection
- Control and estimation constraints are often derived from Lyapunov-like inequalities, yielding SDPs
- Literature is rich with formulations for Est/Ctrl problems as SDPs

Problem Formulation

Some Background

- Dynamic system consisting of N nodes:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t)$$
$$\boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{x}(t)$$

- Physical state: $\boldsymbol{x}(t) \in \mathbb{R}^n$, control input: $\boldsymbol{u}(t) \in \mathbb{R}^m$, sensors data: $\boldsymbol{y}(t) \in \mathbb{R}^p$; $n > m, n > p$
- Dynamic network is **unstable**; $\mathrm{Re}[\lambda_i(\boldsymbol{A})] > 0$
- Control objective: stabilize the network using output measurements
- **Objective:** Design an *output feedback controller*

$$\boldsymbol{u}(t) = \boldsymbol{F}\boldsymbol{y}(t)$$

  such that closed-loop system is stable
- Closed loop dynamics:
  $\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{F}\boldsymbol{y}(t) = (\boldsymbol{A} + \boldsymbol{B}\boldsymbol{F}\boldsymbol{C})\boldsymbol{x}(t)$

Background

Dynamic network with output feedback control: $\dot{\boldsymbol{x}}(t) = (\boldsymbol{A} + \boldsymbol{B}\boldsymbol{F}\boldsymbol{C})\boldsymbol{x}(t)$

- Nonconvex feasibility problem:

$$\text{find} \qquad \boldsymbol{F}, \quad \text{subject to} \quad \text{eig}(\boldsymbol{A} + \boldsymbol{B}\boldsymbol{F}\boldsymbol{C}) < 0$$

- Above problem $\equiv$ to solving nonconvex bilinear matrix inequalities:

$$\textbf{BMI:} \quad \begin{aligned} &\text{find} && \boldsymbol{F}, \boldsymbol{P} \succ 0 \\ &\text{subject to} && \boldsymbol{A}^\top \boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} + \boldsymbol{C}^\top \boldsymbol{F}^\top \boldsymbol{B}^\top \boldsymbol{P} + \boldsymbol{P}\boldsymbol{B}\boldsymbol{F}\boldsymbol{C} \prec 0 \end{aligned}$$

- When is **BMI** solvable? It's an open problem—collaborations?
- But: for sure need **PBH test** to hold

Popov-Belevitch-Hautus (PBH) Tests

For all unstable eigenvalues λ_i of $\boldsymbol{A}$ ($\boldsymbol{w}_i, \boldsymbol{v}_i$ are left/right evectors of $\boldsymbol{A}$)

$$\text{rank} \begin{bmatrix} \boldsymbol{A} - \lambda_i \boldsymbol{I} & \boldsymbol{B} \end{bmatrix} = n, \quad \textbf{OR} \quad \boldsymbol{w}_i^\top \boldsymbol{B} \neq \boldsymbol{0}$$

$$\text{rank} \begin{bmatrix} \boldsymbol{A} - \lambda_i \boldsymbol{I} \\ \boldsymbol{C} \end{bmatrix} = n, \quad \textbf{OR} \quad \boldsymbol{C}\boldsymbol{v}_i \neq \boldsymbol{0}$$

BMI Posed as an LMI

- Don't even try solve the BMI if PBH test is not satisfied
- Luckily, we can solve linear matrix inequalities (LMI):

$$\begin{aligned} \textbf{LMI:} \quad & \text{find} && \boldsymbol{M}, \boldsymbol{N}, \boldsymbol{P} \\ & \text{subject to} && \boldsymbol{A}^\top \boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} + \boldsymbol{C}^\top \boldsymbol{N}^\top \boldsymbol{B}^\top + \boldsymbol{B}\boldsymbol{N}\boldsymbol{C} \prec 0 \\ & && \boldsymbol{B}\boldsymbol{M} = \boldsymbol{P}\boldsymbol{B}, \boldsymbol{P} \succ 0 \end{aligned}$$

then compute $\boldsymbol{F} = \boldsymbol{M}^{-1}\boldsymbol{N}$

- This guarantees that $\boldsymbol{A} + \boldsymbol{B}\boldsymbol{F}\boldsymbol{C}$ is stable
- Caveat: **LMI** only sufficient 😏 but still good enough
- Other approach: successive convex approximations for BMIs

SaA Selection in Linear Dynamic Networks

- Now, let's consider the SaA selection with output feedback control
- Binary variables: $\gamma_i \in \{0,1\}$, $\pi_i \in \{0,1\}$; network dynamics:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{\Pi}\boldsymbol{u}(t)$$
$$\boldsymbol{y}(t) = \boldsymbol{\Gamma}\boldsymbol{C}\boldsymbol{x}(t),$$

- Selecting minimal # of SaA to stabilize dynamic networks:

$$\textbf{Monster:} \quad \min \quad \sum_{k=1}^{N} \pi_k + \gamma_k$$
$$\text{s.t.} \quad \boldsymbol{A}^\top \boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} + \boldsymbol{C}^\top \boldsymbol{\Gamma} \boldsymbol{N}^\top \boldsymbol{\Pi} \boldsymbol{B}^\top + \boldsymbol{B}\boldsymbol{\Pi}\boldsymbol{N}\boldsymbol{\Gamma}\boldsymbol{C} \prec 0$$
$$\boldsymbol{B}\boldsymbol{\Pi}\boldsymbol{M} = \boldsymbol{P}\boldsymbol{B}\boldsymbol{\Pi}, \boldsymbol{P} \succ 0$$
$$\boldsymbol{\Phi} \begin{bmatrix} \boldsymbol{\pi} \\ \boldsymbol{\gamma} \end{bmatrix} \leq \boldsymbol{\phi} \longleftarrow \text{LogisticConstraints}$$
$$\boldsymbol{\pi} \in \{0,1\}^N, \ \boldsymbol{\gamma} \in \{0,1\}^N$$

- **Monster**: mixed-integer nonlinear matrix inequalities (MI-NMI)

Solving Monster, Inc—Method 1

Dealing with Monster

- Nonconvex terms in **Monster**:

$$\boldsymbol{B\Pi N\Gamma C}, \quad \boldsymbol{B\Pi M} = \boldsymbol{PB\Pi}$$

- Best thing we can do is to transform MI-NMI to MI-SDP
- Since $\boldsymbol{\Pi}$ and $\boldsymbol{\Gamma}$ are diagonal matrices with binary values, we have:

$$(\boldsymbol{\Pi N\Gamma})_{ij} = \begin{cases} \boldsymbol{N}_{ij}, & \text{if } \pi_i \wedge \gamma_j = 1 \\ 0, & \text{if } \pi_i \wedge \gamma_j = 0 \end{cases}$$

- Big-M method comes handy here: for large L_1, above rule implies if $\{\pi_i, \gamma_j\} = \{1, 1\}$, then $\boldsymbol{N}_{ij} = \boldsymbol{\Theta}_{ij}$; $\boldsymbol{\Theta}_{ij} = 0$ otherwise
- Hence, we can write

$$|\boldsymbol{\Theta}_{ij}| \leq L_1 \pi_i$$
$$|\boldsymbol{\Theta}_{ij}| \leq L_1 \gamma_j$$
$$|\boldsymbol{\Theta}_{ij} - \boldsymbol{N}_{ij}| \leq L_1 (2 - \pi_i - \gamma_j),$$

Dealing with Monster—2

- We still have to deal with: $\boldsymbol{B\Pi M = PB\Pi}$
- Using similar ideas (but a bit more complicated derivation), we can prove that:

$$\boldsymbol{B\Pi M = PB\Pi}$$

is equivalent to:

$$\begin{bmatrix} |\boldsymbol{M}_{ij}| \\ |\boldsymbol{\Omega}_{ij}| \\ |\boldsymbol{M}_{ij} - \boldsymbol{\Omega}_{ij}| \end{bmatrix} \leq L_2 \begin{bmatrix} 1 & 0 & 1 & -1 \\ 1 & 1 & 0 & -1 \\ 2 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ \pi_i \\ \pi_j \\ |\pi_i - \pi_j| \end{bmatrix}$$

$$|\boldsymbol{M}_{ij} - \boldsymbol{\Omega}_{ij}| \leq L_2(1 - \pi_i),\ \boldsymbol{\Omega} = (\boldsymbol{B}^\top \boldsymbol{B})^{-1} \boldsymbol{B}^\top \boldsymbol{P} \boldsymbol{B}$$

Approach 1: Being Clever with Optimization

Theorem—For large L_1 & L_2 **Monster** $\equiv$ **Beast**[†], **Beast** is MI-SDP

Beast: $\min \sum_{k=1}^{N} \pi_k + \gamma_k$

$$\text{s.t. } \boldsymbol{A}^\top \boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} + \boldsymbol{C}^\top \boldsymbol{\Theta} \boldsymbol{B}^\top + \boldsymbol{B}\boldsymbol{\Theta}\boldsymbol{C} \prec 0$$

$$\begin{bmatrix} |\boldsymbol{\Theta}_{ij}| \\ |\boldsymbol{\Theta}_{ij}| \\ |\boldsymbol{\Theta}_{ij} - \boldsymbol{N}_{ij}| \end{bmatrix} \leq L_1 \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 2 & -1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ \pi_i \\ \gamma_j \end{bmatrix}$$

$$\begin{bmatrix} |\boldsymbol{M}_{ij}| \\ |\boldsymbol{\Omega}_{ij}| \\ |\boldsymbol{M}_{ij} - \boldsymbol{\Omega}_{ij}| \end{bmatrix} \leq L_2 \begin{bmatrix} 1 & 0 & 1 & -1 \\ 1 & 1 & 0 & -1 \\ 2 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ \pi_i \\ \pi_j \\ |\pi_i - \pi_j| \end{bmatrix}$$

$$|\boldsymbol{M}_{ij} - \boldsymbol{\Omega}_{ij}| \leq L_2(1 - \pi_i), \ \boldsymbol{\Omega} = (\boldsymbol{B}^\top \boldsymbol{B})^{-1} \boldsymbol{B}^\top \boldsymbol{P} \boldsymbol{B}$$

$$\boldsymbol{\Phi} \begin{bmatrix} \boldsymbol{\pi} \\ \boldsymbol{\gamma} \end{bmatrix} \leq \boldsymbol{\phi}, \ \boldsymbol{\pi} \in \{0,1\}^N, \ \boldsymbol{\gamma} \in \{0,1\}^N$$

[†] I suppose that Monsters are more difficult to deal with than Beasts. *I suppose.*

Solving Beast

- Well, now we have MI-SDP which is much easier to deal with than MI-NMIs
- But MI-SDPs are still messy—I mean, even SDPs are messy
- MI-SDPs can be solved using branch-and-bound algorithms
- We struggled with **Beast**
  * Very few solvers in the market that are easy to interface with
  * Yalmip's MI-SDP solver is the only high-level one[‡]
  * Yalmip's BnB can take ages for even smaller networks with $N < 20$
- Also, any MI-SDP solver will scale so poorly with number of nodes
- Maybe cutting plane methods will perform better
- Bottom line: problem is still very hard

[‡]Is it? Other solvers require bringing the SDP to a minimalist form.

Solving Monster—Method 2

Method 2 to Solve Monster

- We developed another method to solve Monster
- Idea is based, in a way, on binary search algorithms
- First, recall the main complexity:

LMI: $$\boldsymbol{A}^\top \boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} + \boldsymbol{C}^\top \boldsymbol{\Gamma} \boldsymbol{N}^\top \boldsymbol{\Pi} \boldsymbol{B}^\top + \boldsymbol{B}\boldsymbol{\Pi}\boldsymbol{N}\boldsymbol{\Gamma}\boldsymbol{C} \prec 0, \quad \boldsymbol{B}\boldsymbol{\Pi}\boldsymbol{M} = \boldsymbol{P}\boldsymbol{B}\boldsymbol{\Pi}$$

- **Main idea**: if a fixed binary combination of $\{\boldsymbol{\Pi}, \boldsymbol{\Gamma}\}$ is feasible for the **LMI** above, most likely it's sub-optimal—discard *similar combinations*
- If a binary combination $\{\boldsymbol{\Pi}, \boldsymbol{\Gamma}\}$ is infeasible for the **LMI**, discard many *similar* combinations

What Combinations to Discard?

LMI: find $\boldsymbol{M}, \boldsymbol{N}, \boldsymbol{P} \succ 0$

subject to $\boldsymbol{A}^\top \boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} + \boldsymbol{C}^\top \boldsymbol{\Gamma} \boldsymbol{N}^\top \boldsymbol{\Pi} \boldsymbol{B}^\top + \boldsymbol{B}\boldsymbol{\Pi}\boldsymbol{N}\boldsymbol{\Gamma}\boldsymbol{C} \prec 0$

$\boldsymbol{B}\boldsymbol{\Pi}\boldsymbol{M} = \boldsymbol{P}\boldsymbol{B}\boldsymbol{\Pi}, \boldsymbol{P} \succ 0$

$\boldsymbol{\Pi}, \boldsymbol{\Gamma}$ are not variables now

- Alright then, but why and how can you discard combinations?
- **Lemma**—If a fixed combination $\{\boldsymbol{\Pi}, \boldsymbol{\Gamma}\}$ yields infeasible LMI, then deactivating one or more SaA from $\{\boldsymbol{\Pi}, \boldsymbol{\Gamma}\}$ also yields an infeasible LMI
- Similarly, if a combination is feasible, then activating one or more SaA yields also feasible, but now sub-optimal solution to **Monster**
- Given this, one can discard many combinations

Binary Search Algorithm to Solve Monster

- $\boldsymbol{\mathcal{S}}_p$: database of all *candidate* binary combinations of $\{\boldsymbol{\Pi}, \boldsymbol{\Gamma}\}$ satisfying logistic constraints at iteration p
- $\mathcal{S}_p$: optimal combination at iteration p
- # of active SaA: $\mathcal{H}(\mathcal{S}) \triangleq \sum_{k=1}^{N} \pi_k + \gamma_k$

Algorithm 1 Binary Search to Solve **Monster**

input: $\boldsymbol{\mathcal{S}}_p$
while $\boldsymbol{\mathcal{S}}_p \neq \emptyset$ **do**
  compute: $\sigma \leftarrow |\boldsymbol{\mathcal{S}}_p|$, $q \leftarrow \lceil \sigma/2 \rceil$, $\mathcal{S}_q \in \boldsymbol{\mathcal{S}}_p$
  if LMI is feasible **then**
    $\mathcal{S}^* \leftarrow \mathcal{S}_q$, $\boldsymbol{\mathcal{S}}_p \leftarrow \boldsymbol{\mathcal{S}}_p \setminus \{\mathcal{S} \in \boldsymbol{\mathcal{S}}_p \,|\, \mathcal{H}(\mathcal{S}) \geq \mathcal{H}(\mathcal{S}_q)\}$
  else
    $\boldsymbol{\mathcal{S}}_p \leftarrow \boldsymbol{\mathcal{S}}_p \setminus \{\mathcal{S} \in \boldsymbol{\mathcal{S}}_p \,|\, \mathcal{S}_q \vee \mathcal{S} = \mathcal{S}_q\}$
  end if
  $p \leftarrow p + 1$
end while
output: $\mathcal{S}^*$

Example

- Dynamic network of 2 nodes, 1 input, 1 output for each node
- Logistic constraint: $1 \leq \sum_{k=1}^{N} \pi_k + \gamma_k < 4$; $\boldsymbol{\mathcal{S}}$ can be constructed as:

$$\boldsymbol{\mathcal{S}} = \Big\{(1,0,0,0),(0,1,0,0),(0,0,1,0),(0,0,0,1),$$
$$(1,1,0,0),(1,0,1,0),(1,0,0,1),(0,1,1,0),$$
$$(0,1,0,1),(0,0,1,1),(1,1,1,0),(1,1,0,1),(1,0,1,1),(0,1,1,1)\Big\}.$$

- Let $(1,0,0,1)$ be the starting combination; assume **LMI** is infeasible for this combination
- Discard $(1,0,0,0)$ and $(0,0,0,1)$; updated candidate set:

$$\boldsymbol{\mathcal{S}}_2 = \Big\{(0,1,0,0),(0,0,1,0),(1,1,0,0),(1,0,1,0),(0,1,1,0),(0,1,0,1),$$
$$(0,0,1,1),(1,1,1,0),(1,1,0,1),(1,0,1,1),(0,1,1,1)\Big\}.$$

- New candidate: $(0,1,0,1)$; assume **LMI** is feasible now, updated set:

$$\boldsymbol{\mathcal{S}}_3 = \Big\{(0,1,0,0),(0,0,1,0)\Big\}.$$

Main Result

- **Theorem**—Algorithm 1 returns an optimal solution to **Monster**
- We need Lemma 1 to prove the theorem
- Would this perform better than MI-SDPs and BnB?
- You still have to solve **LMI** at each iteration
- **Alternative**: Instead of solving **LMI** at each iteration, use PBH test
- Check if every combination satisfies the test; discard combinations
- Challenge: computing evectors/evalues for large-scale matrices

Solving Monster—Method 3

We Want Something Faster

- The previous algorithm requires an offline database of all feasible binary combinations
- For large-scale networks, the database might require TBs of storage
- Alternative: learn in a smart way the binary combinations we should not test
- Then, apply Algorithm 1 without requiring the database
- Things are tricky here, because this heuristic won't return optimal solution

Definitions for the Heuristic

- $\mathcal{W}$: set comprising all binary combinations that do not satisfy the logistic constraint
- All elements in $\mathcal{W}$ do not need to be known
- $\underline{w}_p, \bar{w}_p$: required min./max. # of activated SaA such that any candidate $\mathcal{S}_p$ must satisfy $\underline{w}_p \leq \mathcal{H}(\mathcal{S}_p) \leq \bar{w}_p$
- In contrast with Algorithm 1, the heuristic constructs and updates a infeasible binary SaA set
- $\mathcal{Z}$: *Forbidden Set*
- Since $\mathcal{W} \subseteq \mathcal{Z}$, initialize $\mathcal{Z}$ by $\mathcal{W}$
- At each iteration of the heuristic, randomly generate $\mathcal{S} \notin \mathcal{Z}$ while updating $\underline{w}_p, \bar{w}_p$ and $\mathcal{Z}$—call this procedure $\mathrm{GenRandComb}$
- This procedure is computationally cheap

Algorithm 2

1. Let p denote the iteration index and $q = \lceil (\underline{w} + \bar{w})/2 \rceil$ denote the desired number of activated SaA for the candidate $\mathcal{S}$ such that $\mathcal{H}(\mathcal{S}) = q$
2. $\mathcal{S}_p^{(q)}$: candidate at iteration p with q activated SaA
3. $\mathrm{GenRandComb}(\mathcal{S}_p^{(q)}) \notin \mathcal{Z}$
4. Check if this combination yields a feasible **LMI**: if it does, discard all suboptimal combinations by updating $\mathcal{Z}$; otherwise also update $\mathcal{Z}$ and choose a different starting point $\underline{w}_p, \bar{w}_p$
5. Run this algorithm with thresholds and max. iterations

Numerical Tests

Numerical Tests: Series Mass-Spring Systems

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t), \quad \boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{x}(t), \quad \boldsymbol{u}(t) = \boldsymbol{K}\boldsymbol{y}(t)$$

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{O}_{N\times N} & \boldsymbol{I}_N \\ \boldsymbol{T} & \boldsymbol{O}_{N\times N} \end{bmatrix}, \boldsymbol{B} = \begin{bmatrix} \boldsymbol{O}_{N\times N} \\ \boldsymbol{I}_{N\times N} \end{bmatrix}, \boldsymbol{C} = \boldsymbol{I}_{2N}$$

- States are position & velocity of each mass
- We choose the following linear constraint on the number of actuators:

$$\sum_{i=1}^{N} \pi_i \geq \mathrm{floor}(N/4)$$

- Objective: minimal total # of activated SaAs given that $\boldsymbol{A} + \boldsymbol{B}\boldsymbol{\Pi}\boldsymbol{F}\boldsymbol{\Gamma}\boldsymbol{C}$, the closed loop system, is stable

Numerical Tests

Recall that we want to solve **Moster**. The methods we test are:

- **Beast**—a mixed-integer SDP: `MI-SDP`
- Binary search algorithm (Algorithm 1): `BSA-SDP`
- Algorithm 1 with PBH test: `BSA-PBH`
- Heuristic with SDP: `HEU-SDP`
- Heuristic with PBH test: `HEU-PBH`

Scenario	$\mathrm{Max}(\mathrm{Re}(\Lambda(\boldsymbol{A}+\boldsymbol{B\Pi}^*\boldsymbol{F\Gamma}^*\boldsymbol{C})))$	$\sum_{k=1}^{N}\pi_k+\gamma_k$	$\Delta t(s)$	Iterations	$\boldsymbol{\gamma}^*$ and $\boldsymbol{\pi}^*$
MI-SDP	-4.64×10^{-4}	4	166.74	—	$\boldsymbol{\gamma}^*=\{0,1,0,0,0,0,0,0,0,0,0,1\}$ $\boldsymbol{\pi}^*=\{0,1,1,0,0,0,0,0,0,0,0,0\}$
BSA-SDP	-1.94×10^{-3}	2	33.66	41	$\boldsymbol{\gamma}^*=\{0,0,0,0,0,0,0,1,0,0,0,0\}$ $\boldsymbol{\pi}^*=\{0,0,0,0,0,0,0,1,0,0,0,0\}$
BSA-PBH	-1.97×10^{-3}	5	21.83	10	$\boldsymbol{\gamma}^*=\{0,1,0,0,0,0,0,0,0,0,1,0\}$ $\boldsymbol{\pi}^*=\{0,1,0,0,0,1,0,0,0,1,0,0\}$
HEU-SDP	-1.81×10^{-3}	2	7.98	23	$\boldsymbol{\gamma}^*=\{0,0,0,1,0,0,0,0,0,0,0,0\}$ $\boldsymbol{\pi}^*=\{0,0,0,1,0,0,0,0,0,0,0,0\}$
HEU-PBH	-3.19×10^{-3}	7	2.06	5	$\boldsymbol{\gamma}^*=\{1,0,0,1,0,0,0,0,0,1,0,0\}$ $\boldsymbol{\pi}^*=\{1,1,0,0,0,0,0,0,1,1,0,0\}$

Extensions, Interesting Problems

Extensions to Time-Varying Selection

- Suppose now that the network topology is changing:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}^{\boldsymbol{j}}\boldsymbol{x}(t) + \boldsymbol{B}^{\boldsymbol{j}}\boldsymbol{\Pi}^{\boldsymbol{j}}\boldsymbol{u}(t)$$
$$\boldsymbol{y}(t) = \boldsymbol{\Gamma}^{\boldsymbol{j}}\boldsymbol{C}^{\boldsymbol{j}}\boldsymbol{x}(t),$$

- Common problem in dynamic networks
- Minimal # of SaAs to stabilize dynamic networks for all time-periods j:

$$\begin{aligned}
\min \quad & \sum_{k,j} \pi_k^j + \gamma_k^j \\
\text{s.t.} \quad & \boldsymbol{A}^{j\top}\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A}^{j} + \boldsymbol{C}^{j\top}\boldsymbol{\Gamma}^{j}\boldsymbol{N}^{j\top}\boldsymbol{\Pi}^{j}\boldsymbol{B}^{j\top} + \boldsymbol{B}^{j}\boldsymbol{\Pi}^{j}\boldsymbol{N}^{j}\boldsymbol{\Gamma}^{j}\boldsymbol{C}^{j} \prec 0 \\
& \boldsymbol{B}^{j}\boldsymbol{\Pi}^{j}\boldsymbol{M}^{j} = \boldsymbol{P}\boldsymbol{B}^{j}\boldsymbol{\Pi}^{j}, \boldsymbol{P} \succ 0 \\
& \boldsymbol{\Phi}^{j}\begin{bmatrix}\boldsymbol{\pi}^{j} \\ \boldsymbol{\gamma}^{j}\end{bmatrix} \leq \boldsymbol{\phi}^{j} \\
& \boldsymbol{\pi}^{j} \in \{0,1\}^{N},\ \boldsymbol{\gamma}^{j} \in \{0,1\}^{N}
\end{aligned}$$

Example of SDP Formulations of Control Problems

Linear Quadratic Regulator (LQR) Control

$$\begin{aligned} \min \quad & J = \mathbb{E} \int_{t_0}^{\infty} \boldsymbol{x}(\tau)\boldsymbol{Q}\boldsymbol{x}(\tau) + \boldsymbol{u}(\tau)\boldsymbol{R}\boldsymbol{u}(\tau) d\tau \\ \text{s.t.} \quad & \dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}_u\boldsymbol{u}(t) + \boldsymbol{w}(t) \\ & \boldsymbol{w}(t) \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{W}) \end{aligned}$$

is equivalent to:

$$\begin{aligned} \min_{\boldsymbol{S},\boldsymbol{Y}} \quad & \textbf{trace}(\boldsymbol{W}\boldsymbol{S}^{-1}) \\ \text{s.t.} \quad & \begin{bmatrix} \boldsymbol{A}\boldsymbol{S} + \boldsymbol{S}\boldsymbol{A}^\top + \boldsymbol{B}_u\boldsymbol{Y} + \boldsymbol{Y}^\top\boldsymbol{B}_u^\top & \boldsymbol{S} & \boldsymbol{Y} \\ \boldsymbol{S} & -\boldsymbol{Q}^{-1} & 0 \\ \boldsymbol{Y}^\top & 0 & -\boldsymbol{R}^{-1} \end{bmatrix} \preceq \boldsymbol{O} \end{aligned}$$

Solve the SDP, then compute the optimal state-feedback controller that minimizes J^*

$$\boxed{\boldsymbol{u}(t) = -\boldsymbol{R}^{-1}\boldsymbol{B}_u^\top\boldsymbol{S}^{-1}\boldsymbol{x}(t).}$$

Example of SDP Formulations of Control Problems

Linear Quadratic Regulator (LQR) Control with Actuator Selection

$$\begin{aligned} \min \quad & J = \mathbb{E} \int_{t_0}^{\infty} \boldsymbol{x}(\tau)\boldsymbol{Q}\boldsymbol{x}(\tau) + \boldsymbol{u}(\tau)\boldsymbol{R}\boldsymbol{u}(\tau) d\tau \\ \text{s.t.} \quad & \dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}_u \boldsymbol{\Pi} \boldsymbol{u}(t) + \boldsymbol{w}(t) \\ & \boldsymbol{w}(t) \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{W}), \boldsymbol{\Pi} \in \{0,1\} \end{aligned}$$

is equivalent to:

$$\begin{aligned} \min_{\boldsymbol{S},\boldsymbol{Y},\boldsymbol{\Pi}} \quad & \textbf{trace}(\boldsymbol{W}\boldsymbol{S}^{-1}) + \sum_k \boldsymbol{\Pi}_k \\ \text{s.t.} \quad & \begin{bmatrix} \boldsymbol{A}\boldsymbol{S} + \boldsymbol{S}\boldsymbol{A}^\top + \boldsymbol{B}_u\boldsymbol{\Pi}\boldsymbol{Y} + \boldsymbol{Y}^\top\boldsymbol{\Pi}\boldsymbol{B}_u^\top & \boldsymbol{S} & \boldsymbol{Y} \\ \boldsymbol{S} & -\boldsymbol{Q}^{-1} & 0 \\ \boldsymbol{Y}^\top & 0 & -\boldsymbol{R}^{-1} \end{bmatrix} \preceq \boldsymbol{O} \end{aligned}$$

Solve the MI-BMI, compute optimal state-feedback controller:

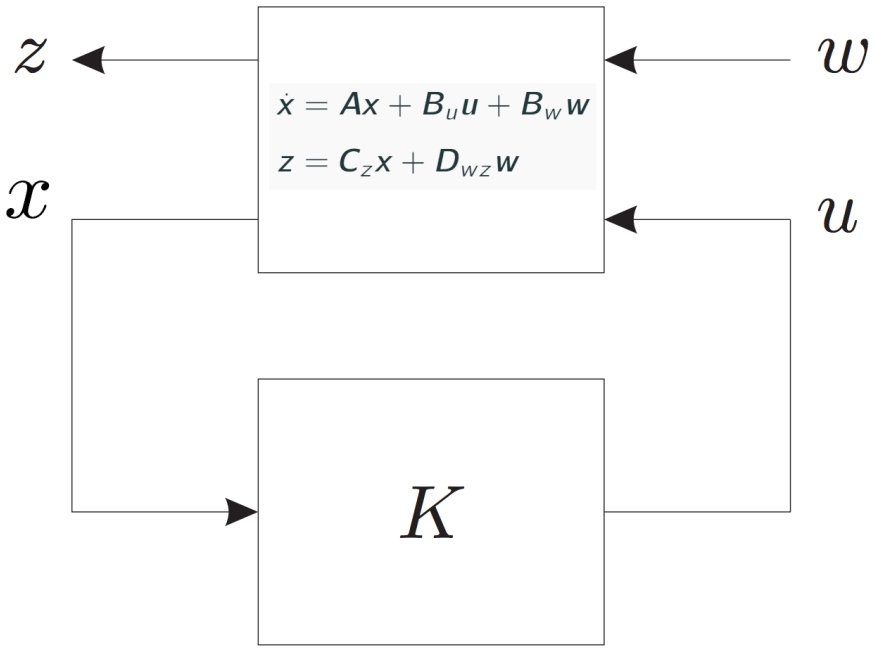
$$\boxed{\boldsymbol{u}(t) = -\boldsymbol{R}^{-1}\boldsymbol{\Pi}\boldsymbol{B}_u^\top \boldsymbol{S}^{-1}\boldsymbol{x}(t).}$$

Other Control Metrics: Robustness to Unknown Inputs

$\mathcal{L}_\infty$ Control as an SDP

We want to minimize $\frac{||\boldsymbol{z}(t)||_\infty}{||\boldsymbol{w}(t)||_\infty}$; solve this SDP

$$\begin{aligned} &\min \ \zeta \\ &\text{s.t.} \begin{bmatrix} \boldsymbol{A}\boldsymbol{S} + \boldsymbol{S}\boldsymbol{A}^\top + 2\alpha\boldsymbol{S} & & \\ -\boldsymbol{B}_u\boldsymbol{Z} - \boldsymbol{Z}^\top\boldsymbol{B}_u^\top & & \boldsymbol{B}_w \\ \boldsymbol{B}_w^\top & & -2\alpha\boldsymbol{I} \end{bmatrix} \preceq \boldsymbol{O} \begin{bmatrix} -\boldsymbol{S} & \boldsymbol{O} & \boldsymbol{S}\boldsymbol{C}_z^\top \\ \boldsymbol{O} & -\boldsymbol{I} & \boldsymbol{D}_{wz}^\top \\ \boldsymbol{C}_z\boldsymbol{S} & \boldsymbol{D}_{wz} & -\zeta\boldsymbol{I} \end{bmatrix} \preceq \boldsymbol{O} \end{aligned}$$



- Obtain $\boldsymbol{K} = \boldsymbol{Z}\boldsymbol{S}^{-1}$; use feedback control $\boldsymbol{u}(t) = \boldsymbol{K}\boldsymbol{x}(t)$
- This minimizes impact of $\boldsymbol{w}(t)$ on $\boldsymbol{z}(t)$
- ζ is the control index guaranteeing that $\|\boldsymbol{z}(t)\| \leq \sqrt{\zeta}\|\boldsymbol{w}(t)\| \ \forall\, t$

$\mathcal{L}_\infty$ Control with Actuator Selection

- For this system:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}_u\boldsymbol{\Pi}\boldsymbol{u}(t) + \boldsymbol{B}_w\boldsymbol{w}(t)$$
$$\boldsymbol{z}(t) = \boldsymbol{C}_z\boldsymbol{x}(t) + \boldsymbol{D}_{wz}\boldsymbol{w}(t)$$

the co-design problem of $\mathcal{L}_\infty$ controller and actuator selection is nonconvex with *mixed-integer bilinear matrix inequalities* (MIBMI):

$$f^* = \min_{\boldsymbol{S},\boldsymbol{Z},\zeta,\boldsymbol{\Pi}} \quad \zeta + \boldsymbol{\alpha}_\pi^\top \boldsymbol{\pi}$$
$$\text{s.t.}$$
$$\begin{bmatrix} \boldsymbol{A}\boldsymbol{S} + \boldsymbol{S}\boldsymbol{A}^\top + 2\alpha\boldsymbol{S} & \\ -\boldsymbol{B}_u\boldsymbol{\Pi}\boldsymbol{Z} - \boldsymbol{Z}^\top\boldsymbol{\Pi}\boldsymbol{B}_u^\top & \boldsymbol{B}_w \\ \boldsymbol{B}_w^\top & -2\alpha\boldsymbol{I} \end{bmatrix} \preceq \boldsymbol{O} \begin{bmatrix} -\boldsymbol{S} & \boldsymbol{O} & \boldsymbol{S}\boldsymbol{C}_z^\top \\ \boldsymbol{O} & -\boldsymbol{I} & \boldsymbol{D}_{wz}^\top \\ \boldsymbol{C}_z\boldsymbol{S} & \boldsymbol{D}_{wz} & -\zeta\boldsymbol{I} \end{bmatrix} \preceq \boldsymbol{O}$$
$$\boldsymbol{\Pi} \in \mathcal{A} \subset \{0,1\}^N$$

- $\boldsymbol{\alpha}_\pi$: actuators weights; $\mathcal{A}$: actuator logistic constraints
- Relaxing integrality constraints to box yields a lower bound L^*

Discussion on Applications

- Applications in smart power grids
- Water distribution systems
- Contamination control in drink water networks
- Transportation networks
- Final remarks