Co-Optimization of Control and Actuator Selection for Cyber-Physical Systems

Chin-Yao Chang * **Sonia Martínez** * **Jorge Cortés** *

* *Department of Mechanical and Aerospace Engineering, University of California, San Diego, {chc433,soniamd,cortes}@ucsd.edu*

Abstract: This paper considers actuator selection problems which aim to maintain control performance of dynamical systems, and minimize operational costs or wear-and-tear of the actuators. The logical controls of actuators make the problem combinatorial, which make exhaustive search impractical. An actuator selection problem can be cast as a binary-integer programming with bilinear matrix inequalities (BIBMIs). In this paper, we first show that such non-convex optimization can be equivalently reformulated as an optimization problem with non-convexities restricted to binary decision variables. We next consider a continuous optimization which is equivalent to the BIBMIs, and leverage the continuous reformulation to derive a branch-and-bound method employing bound refinement. Numerical simulations demonstrate the effectiveness of the proposed approach.

Keywords: Bilinear matrix inequalities, cyber-physical systems, binary-integer programming

1. INTRODUCTION

Actuator (or sensor) selection is a decision problem that consists of choosing a subset of actuators (sensors) from the entire actuator (sensor) set in a cyber-physical system (CPS), while maintaining network performance such as controllability or observability. The activation of the entire actuator set leads in principle to a better control performance. However, wear-and-tear as well as the operating cost of actuators encourage the search of solutions that involve partial utilization. The balance of control performance and actuator selection poses a class of binary-integer programming problems with bilinear inequality constraints (BIBMIs). In this paper, we pursue a generic approach for BIBMIs in the context of control and actuator selection co-optimization for CPS.

Literature review

The identification of tractable formulations of network controllability and observability by means of suitable metrics is an excellent starting point for efficient actuator and sensor selection. Thus, many recent works are based on the characterization of the modularity properties of various control performance metrics (Summers, 2016; Jawaid and Smith, 2015). For network problems without modularity properties, mainstream approaches include greedy algorithms (Tzoumas et al., 2016; Zhang et al., 2017), and convex relaxation heuristics (Joshi and Boyd, 2009; Dhingra et al., 2014; Argha et al., 2018; Taylor et al., 2017). The literature above focuses more on classical performance metrics, such as Kalman filters, linear quadratic regulators, and Gramians. The work (Taha et al., 2017) casts the actuator selection problem as a BIBMIs. They further relax the bilinear constraints and focus on binary-integer semidefinite programming, and developed computational tractable approaches that provide upper and lower bounds for BIBMIs.

Mixed-integer programming (MIP) is a classical NP-hard optimization problem with a long list of heuristics that aim to find feasible solutions, cf. review papers (Johnson et al., 2000; Burer and Letchford, 2012). There are specialized algorithms for BIP (a branch of MIP) that are found effective. The work (Goemans and Williamson, 1995) showed that semidefinite programming (SDP) provides an expected solution at least .87856 times the optimal value. However, scalability problems still limit the practicality of SDP. Several recent works (Wu and Ghanem, 2016; Yuan and Ghanem, 2016) have reformulated the binary optimization to equivalent continuous optimizations including L_p-box and exact penalty methods. Simulation results show that those new reformulations are promising.

Statement of contributions

In this manuscript, we develop a systematic approach that finds a quality solution for integer programming with BIBMI in the context of control and actuator selection co-optimization of CPSs. Our contribution is two-fold. On the one hand, we show that to solve the integer programming with BIBMIs, it is sufficient to solve an optimization with the only non-convexity restricted to binary decision variables. This is done by first adding unbinding constraints to the original integer programming. We next show that with additional unbinding constraints, there is a bijection between a global optimal solution of the original problem and an optimum for the relaxed optimization without BMIs. This implies that it is sufficient to solve the relaxed optimization with the binary variables as the only non-convexity. On the other hand, we provide a continuous optimization which is equivalent to BIBMIs. We further leverage it to bound refinements for branch-and-bound (B&B) by analysis on the solutions of the reformulated continuous optimization. We also develop a special branching rule for B&B which prioritizes finding one sufficiently good solution instead of the global optimum. Simulations demonstrate that the developed approach is effective compared to the SDP-R method described in (Taha et al.,

2017). For reasons of space, all proofs are omitted and will appear elsewhere.

Notation

Throughout the paper, we use the following notation. We denote by $\mathbb{N}$ and $\mathbb{R}$ respectively the sets of positive integer and real numbers. We denote $|\mathcal{N}|$ as the cardinality of $\mathcal{N}$. The 2-norm and infinite norm of a complex vector $v \in \mathbb{C}^n$ are $\|v\|_2$ and $\|v\|$, respectively. We write the Frobenius norm of a matrix A as $\|A\|_F$. The block diagonal matrix for n number of matrices, $A_1, \cdots, A_n$, is written as $\text{blkdiag}(A_1, \cdots, A_n)$. A graph is a pair $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N} \subseteq \mathbb{N}$ is its set of nodes and $\mathcal{E} \subseteq \mathcal{N} \times \mathcal{N}$ is its set of edges. A *path* in a graph is a sequence of nodes such that any two consecutive nodes correspond to an edge of the graph. The *length* of a path is its number of edges.

2. PROBLEM FORMULATION

We consider a CPS composed of a team of N agents whose collective dynamics are described by

$$\begin{aligned} \dot{x} &= Ax + B_u \Pi u + B_w w, \\ y &= Cx + Dw, \end{aligned} \tag{1}$$

where $x \in \mathbb{R}^{n_x}$, $u \in \mathbb{R}^{n_u}$, $w \in \mathbb{R}^{n_w}$, $y_i \in \mathbb{R}^{n_y}$ are, respectively, the state variable, control input, disturbance, and output measurement. All A, B_u, B_w, C, and D are matrices with the appropriate dimensions. The matrix Π is block-diagonal, of the form

$$\Pi = \begin{bmatrix} I_{n_{u_1}}\pi_1 & 0 & \cdots & 0 \\ 0 & I_{n_{u_2}}\pi_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & I_{n_{u_N}}\pi_N \end{bmatrix}, \tag{2}$$

where n_{u_i} is the dimension of the control associated with agent i, and π_i denotes whether the actuator of agent i is activated ($\pi_i = 1$) or not ($\pi_i = 0$). We write $\pi \in \{0,1\}^N$ as the collection of π_i for all i. The network state $x \in \mathbb{R}^{n_x}$ is a concatenation of the states of the agents, $x_i \in \mathbb{R}^{n_{x_i}}$ for all $i = 1, \cdots, N$. Similarly, u and y are, respectively, the concatenation of the controls and measurements of all agents. The mapping from the input to state vector can thus be written in the form $B_u = \text{blkdiag}(B_{u_1}, \cdots, B_{u_N})$.

There are two major objectives in controller design for CPS. One is a generic control objective, most commonly stabilization under certain assumptions on the disturbance or control effort; another is the minimization of the number of selected actuators. Designing a controller that simultaneously accounts for both objectives naturally results in an BIBMIs. In the following, we consider the robust L_∞-control of (1) by minimizing the number of activated actuators, which results into the following optimization problem (Taha et al., 2017; Pancake et al., 2000):

$$\textbf{(P1)} \quad \min_{S\succeq 0, \zeta>0, \pi, Z_i, i=1,\cdots,N} (\eta+1)\zeta + \alpha_\pi^\top \pi$$

$$\begin{bmatrix} AS + SA^\top - B_u\Pi Z - Z^\top \Pi B_u^\top + \alpha S & B_w \\ B_w^\top & -\alpha\eta I \end{bmatrix} \preceq 0, \tag{3}$$

$$\begin{bmatrix} -S & 0 & SC^\top \\ 0 & -I & D^\top \\ CS & D & -\zeta I \end{bmatrix} \preceq 0, \tag{4}$$

$$Z = [Z_1^\top, \cdots, Z_N^\top]^\top, \ \ \|Z_i\|_F \le \overline{z}_i, \ \ \forall i = 1, \cdots, N, \tag{5}$$

$$H\pi \le h, \ \ \pi \in \{0,1\}^N,$$

where the i^{th} component of α_π is the cost of activating actuator i, H and h denote the constraints for activating the actuators, α and η for all i, are predetermined constants which characterize control performance. The matrix inequalities (3) and (4) ensure that system (1) is L_∞-stable via the quadratic Lyapunov function defined by S. The control associated with the optimal solution of **(P1)** is $u = -ZS^{-1}x$. The cost function is composed of two objectives including the number of activated actuators, and the L_∞-control performance index ζ. The coefficient for the objective function $(\eta+1)\zeta$ poses a desired L_∞ control performance. Details for the formulation of **(P1)** are available at (Pancake et al., 2000). Notice that for any solution $(S^\star, \Pi, Z^\star)$ such that constraint (3) and $B_u\Pi Z + Z^\top \Pi B_u^\top \succeq 0$, one can always choose some $Z^{\star 2} = aZ^\star$ with scalar $a > 1$ so that $(S^\star, \Pi, Z^{\star 2})$ also satisfies (3). To this end, constraint (5) is not binding as long as $\overline{z}$ is chosen large enough. In fact, constraint (5) is not included in (Taha et al., 2017). We add constraint (5) because it is very useful in approximating BMIs. We will revisit this constraint in the numerical studies and show that it does not change the optimal value as long as $\overline{z}_i$ is not too small for every i.

Two reasons explain the non-convexity of **(P1)**: one is the discrete decision variables and the other is the BMI terms, ΠZ and $Z^T\Pi$, embedded in (3). We deal with these problems in inverse order: first, we describe a problem reformulation that incorporates the BMI constraints effectively using several linear matrix inequalities, and then we address the combinatorial problem by B&B with new insights from continuous reformulations.

3. APPROXIMATION OF THE BMI CONSTRAINTS

In this section, we show how to effectively deal with the nonconvexity arising from the bilinear matrix inequalities. We start by defining the optimization problem **(P2)** by relaxing the binary variables $\pi \in \{0,1\}^N$ of **(P1)** into $\pi \in [0,1]^N$. Notice that **(P2)** remains non-convex due to the bilinear terms involving the products of the variables π_i and the matrices Z_i, for each $i = 1, \ldots, N$. To deal with this, we next remove the bilinear terms by introducing the new variables $G_i = \pi_i Z_i$ in **(P2)**. This results into the following convex problem,

$$\textbf{(P3)} \quad \min_{S\succeq 0, \zeta>0, G_i, i=1,\cdots,N} (\eta+1)\zeta + \alpha_\pi^\top \pi$$

$$\begin{bmatrix} AS + SA^\top - B_uG - G^\top B_u^\top + \alpha S & B_w \\ B_w^\top & -\alpha\eta I \end{bmatrix} \preceq 0,$$

$$\begin{bmatrix} -S & 0 & SC^\top \\ 0 & -I & D^\top \\ CS & D & -\zeta I \end{bmatrix} \preceq 0, \quad G = [G_1^\top, \cdots, G_N^\top]^\top,$$

$$\|G_i\|_F \le \overline{z}_i, \ \ \forall i = 1, \cdots, N,$$

$$\pi_i = \|G_i\|_F / \overline{z}_i, \ \ H\pi \le h.$$

The decision variables π and Z in **(P2)** are removed in **(P3)**, with the constraints for π and Z being imposed on G instead. Note that the equalities $\pi_i = \|G_i\|_F/\overline{z}_i$ should really be inequalities $\pi_i \ge \|G_i\|_F/\overline{z}_i$. The following result paves the way to show later that the equality is tight.

Proposition 3.1. ***(Optimal solution of (P1)).*** There exists a global optimizer of **(P1)**, denoted $(S^{\star_1}, \zeta^{\star_1}, \pi^{\star_1}, Z^{\star_1})$,

with the property that, for each $i \in \{1,\dots,N\}$, $\|Z_i^{\star_1}\|_F = \overline{z}_i$ if $\pi_i^{\star_1} = 1$.

The following result states a similar property for the optimization **(P2)**.

Corollary 3.2. ***(Optimal solution of (P2)).*** There exists a global optimizer of **(P2)**, denoted $(S^{\star_2}, \zeta^{\star_2}, \pi^{\star_2}, Z^{\star_2})$, with the property that, for each $i \in \{1,\dots,N\}$,

$$\|Z_i^{\star_2}\|_F = \overline{z}_i \quad \text{if } \pi_i^{\star_2} > 0. \tag{6}$$

Using Corollary 3.2, we can conclude that **(P2)** and **(P3)** are equivalent.

Lemma 3.3. ***(Equivalence between (P2) and (P3)).*** The optimizations **(P2)** and **(P3)** are equivalent.

One can prove Lemma 3.3 by showing that a global optimal solution of **(P2)** from can be derived from **(P3)** and vice versa. Lemma 3.3 shows that the BMIs embedded in **(P2)** can be replaced by linear matrix inequalities without affecting the optimal solution. Therefore, the non-convexity in the original optimization problem **(P1)** can be effectively treated once the integer constraints are relaxed. Consequently, our focus next is on how to effectively deal with the latter beyond the coarse $[0,1]^N$-relaxation.

4. BINARY-INTEGER PROGRAMMING

In this section, we first use the results in the previous section to formulate a continuous optimization problem without binary variables, while being equivalent to **(P1)**. The reformulation is beneficial to address the non-convexity of **(P1)** originated from the binary variables. We next revisit B&B with both novel branching and bound-refinement methods from the insights of the continuous reformulation.

*4.1 Equivalent formulation of **(P1)***

We consider a reformulation of **(P3)** by adding a non-convex constraint, shown in the following

$$\textbf{(P4)} \qquad \min_{(x,\sigma)\in\mathcal{X},\sigma\in[-1,1]^N} f(x,\sigma),$$
$$\text{s.t. } N - \sigma^\top\sigma = 0. \tag{7}$$

where x collects all the decision variables in **(P3)**, $\sigma_i = 2\frac{\|G_i\|_F}{\overline{z}_i} - 1$ for all $i = 1,\cdots,N$, $\mathcal{X}$ denotes all the constraints in **(P3)**, and f is the objective function of **(P3)** written with variables x and σ. Constraint (7) forces σ to take the vertex points of the hypercube $[-1,1]^N$. Adding constraint (7) makes **(P4)** equivalent to **(P1)**.

Lemma 4.1. ***(Equivalence between (P1) and (P4)).*** Optimizations **(P1)** and **(P4)** are equivalent.

To this end, we have reformulated **(P1)** into **(P4)** with the only non-convexity lying in the concave quadratic equality constraint. We further consider the following optimization which penalizes **(P3)** with the concave constraint in **(P4)**

$$\textbf{(P5-}\mu\textbf{)} \qquad \min_{(x,\sigma)\in\mathcal{X},\sigma\in[-1,1]^N} f(x,\sigma) + \mu(N - \sigma^\top\sigma),$$

where $\mu > 0$. Proposition 4.2 shows that the optimal solution of **(P5-μ)** satisfies $\sigma^{\star_5} \in \{-1,1\}^N$ if μ is large enough. Let $L < \infty$ be the smallest constant such that

$$\|f(x,\sigma_a) - f(x,\sigma_b)\| \le L\|\sigma_a - \sigma_b\|, \tag{8}$$
$$\forall (x,\sigma_a) \in \mathcal{X},\ (x,\sigma_b) \in \mathcal{X},\ \sigma_a,\sigma_b \in [-1,1]^N.$$

The inequality above is also known as uniformly Lipschitz in x with respect to σ. We have $L < \infty$ in (8) because f is linear to σ. The gradient of f is given as

$$\nabla f(x,\sigma) = [\nabla_x f(x,\sigma)^\top,\ \ \nabla_\sigma f(x,\sigma)^\top]^\top,$$

then (8) implies that

$$\|\nabla_\sigma f(x,\sigma)\| \le L. \tag{9}$$

We now state Proposition 4.2 which shows the properties of the optimal solution of **(P5-μ)**.

Proposition 4.2. ***(Boundary points are local optima).*** If $\mu \ge L$, then every $(x_c,\sigma_c) \in \mathcal{X}$ and $\sigma_c \in \{-1,1\}^N$ such that

$$\nabla_x f(x_c,\sigma_c)^\top (x - x_c) \ge 0,\ \ \nabla_x^2 f(x_c,\sigma_c) \succeq 0, \tag{10}$$

is a local minimum of **(P5-μ)**. Moreover, the global optimal solution of **(P5-μ)** satisfies $\sigma^{\star_5} \in \{-1,1\}^N$.

We describe the sketch proof briefly here. For the local optimality, we first use (10) to show that first-order local optimality condition always holds. Uniform Lipschitz property (8) is then used to show that every $(x_c,\sigma_c) \in \mathcal{X}$ and $\sigma_c \in \{-1,1\}^N$ can only be a local minimum. We also apply (8) to show $\sigma^{\star_5} \in \{-1,1\}^N$ by contradiction.

4.2 Branch-and-Bound Method Revisited

In the last section, we have shown that the penalization method can enforce integer σ. In this section, we leverage a convexified **(P5-μ)** into the B&B method to find better upper and lower bounds of each branch, which may lead to a faster convergence.

Branch-and-bound (Schrijver, 1998; Karlof, 2005) B&B is an algorithm that systematically enumerates candidate solutions. The enumeration starts with a root (initial) node which includes the entire solution set. The algorithm next branches the solution set into two disjoint sets, and computes their respective upper and lower bounds. A branch is "pruned" if it has a bigger lower bound than the upper bound of any other branch. The procedure repeats until the optimal solution is found. This branching process generates a binary tree with nodes being "branches" and edges connecting the branches. The B&B algorithm usually finds the optimal solution with much less number of iterations than a brute-force search for MIP.

Branching Method There are many branching strategies for B&B, such as most infeasible branching, pseudo cost branching, and strong branching, cf., (Martin, 2001). In this work, we adapt a rounding branching strategy which is inspired by the fact that the rounding method usually helps deriving practically useful solutions (Lenstra et al., 1990). The rounding branching strategy rounds the optimal switching in **(P3)**, $\sigma^{\star_3}$, to a closest point σ^{p_3} such that $\sigma^{p_3} \in \{-1,1\}^N$. It then computes

$$d_\sigma := \left\lceil (\sigma^{\star_3})^\top \sigma^{p_3} \right\rceil. \tag{11}$$

The definition of d_σ requires the following considerations. First of all, $(\sigma^{\star_3})^\top\sigma^{p_3}$ is the projection of $\sigma^{\star_3}$ on a line segment $l_\sigma = [-\sigma^{p_3}, \sigma^{p_3}]$. Second, if we evenly divide l_σ into N number of line segments, then d_σ represents the segment that the projection is in. The definition is very useful in separating the integer points based on the

distance to σ^{p_3}, which gives rise to the following disjoint sets

$$\mathcal{X}_u := \{\sigma \mid \sigma^\top \sigma^{p_3} \geq d_\sigma,\ \ \sigma \in [-1,1]^N\}, \tag{12a}$$

$$\mathcal{X}_d := \{\sigma \mid \sigma^\top \sigma^{p_3} \leq d_\sigma - 1,\ \ \sigma \in [-1,1]^N\}. \tag{12b}$$

All the points in $\mathcal{X}_u$ are closer to the rounded solution, σ^{p_3}, compared to the points in $\mathcal{X}_d$. We branch **(P3)** by $\mathcal{X}_u$ and $\mathcal{X}_d$

$$\textbf{(P3)-}\mathcal{X}_u : \textbf{(P3)} \text{ with constraint } \mathcal{X}_u,$$
$$\textbf{(P3)-}\mathcal{X}_d : \textbf{(P3)} \text{ with constraint } \mathcal{X}_d\,.$$

The intuition behind this branching approach is that if the rounded solution is close to the global optimum, then the optimum is likely to be in $\mathcal{X}_u$.

The rounding branching method may get stuck at a branch for which no new branches can be derived by (12). To illustrate such situation, consider a path $\mathcal{P}_i$ from the root node of the enumeration tree to node i, and $\mathcal{X}_{\mathcal{P}_i} := \cap_{k=1}^{n_{p_i}} \mathcal{X}_{ik}$, where $n_{p_i} = |\mathcal{P}_i|$ is the length of $\mathcal{P}_i$ and $\mathcal{X}_{ik}$ denotes the constraint added by either (12a) or (12b) at node (branch) k in $\mathcal{P}_i$. Denote $\mathcal{X}_{u_i}$ and $\mathcal{X}_{d_i}$ as the candidate disjoint sets for the next branches from the end node of $\mathcal{P}_i$ (or node i). If one of the following holds

$$\begin{cases} \mathcal{X}_{\mathcal{P}_i} \subseteq \mathcal{X}_{u_i} \cap \mathcal{X}_{\mathcal{P}_i}, & \mathcal{X}_{d_i} \cap \mathcal{X}_{\mathcal{P}_i} = \emptyset, \\ \mathcal{X}_{\mathcal{P}_i} \subseteq \mathcal{X}_{d_i} \cap \mathcal{X}_{\mathcal{P}_i}, & \mathcal{X}_{u_i} \cap \mathcal{X}_{\mathcal{P}_i} = \emptyset, \end{cases} \tag{13}$$

then the rounding branching method is "stuck" or "locked" because the search space can not be further reduced. The situation may occur when there are multiple nodes in $\mathcal{P}_i$ that give the same σ^{p_3}, and eventually make $\mathcal{X}_{\mathcal{P}_i} \subseteq \{\sigma \mid \sigma^\top \sigma^{p_3} = d_\sigma\}$ for some $d_\sigma \in [-N, N]$. Any further branching by σ^{p_3} becomes meaningless along the path $\mathcal{P}_i$. To address the issue, we adapt a simple strategy that unlocks the branching process by choosing $\sigma_i^{p_3}$ and d_{σ_i} such that the resulting $\mathcal{X}_{u_i}$ and $\mathcal{X}_{d_i}$ do not satisfy (13). This can be achieved by choosing new $\sigma_i^{p_3}$ and d_{σ_i} by

$$\{\sigma_i^{p_3}, d_{\sigma_i}\} \notin \cup_{k=1}^{n_{p_i}} \{\sigma_{ik}^{p_3}, d_{\sigma_{ik}}\}, \tag{14}$$

where $d_{\sigma_{ik}}$ is the d_σ associated with node k of $\mathcal{P}_i$. We summarize the rounding branching method (for one round of branching) in Algorithm 1.

Algorithm 1 Rounding branching method

1: **Given** the optimal solution of **(P3)**-$\mathcal{X}_{\mathcal{P}_i}$
2: **Compute** $\mathcal{X}_{u_i}$ and $\mathcal{X}_{d_i}$ by (11) and (12).
3: **If** (13) holds
4: choose $\{\sigma_i^{p_3}, d_{\sigma_i}\}$ by (14)
5: update $\mathcal{X}_{u_i}$ and $\mathcal{X}_{d_i}$ by $\{\sigma_i^{p_3}, d_{\sigma_i}\}$ and (12)
6: **end**

Refining the Upper and Lower Bounds An important aspect of B&B are the computations of the upper and lower bounds for each branch. We continue the discussion on **(P5-μ)** in Section 4.1 for a bound refinement method.

The refining procedure starts with the optimal solution of the convexified problem associated with the end node (branch) of a path, say **(P3)**-$\mathcal{X}_{\mathcal{P}_i}$. Let $(x^{\star_3}, \sigma^{\star_3})$ be the optimal solution of **(P3)**-$\mathcal{X}_{\mathcal{P}_i}$, then $f(x^{\star_3}, \sigma^{\star_3})$ is clearly a lower bound for branch **(P3)**-$\mathcal{X}_{\mathcal{P}_i}$. Based on $(x^{\star_3}, \sigma^{\star_3})$, we consider penalized **(P3)**-$\mathcal{X}_{\mathcal{P}_i}$ given as

$$\begin{aligned} \textbf{(P3-}\mu\textbf{)-}\mathcal{X}_{\mathcal{P}_i} \quad & \min_{x,\sigma} f(x, \sigma + \sigma^{\star_3}) - 2\mu\sigma^\top \sigma^{\star_3}, \\ & \text{s.t. } \sigma + \sigma^{\star_3} \in [-1,1]^N \cap \mathcal{X}_{\mathcal{P}_i}, \\ & (x, \sigma + \sigma^{\star_3}) \in \mathcal{X}\,. \end{aligned}$$

Let **(P5-μ)**-$\mathcal{X}_{\mathcal{P}_i}$ be an optimization which is the same as **(P3-μ)**-$\mathcal{X}_{\mathcal{P}_i}$ except that the term $\mu(N - \|\sigma\|_2^2 - \|\sigma^{\star_3}\|_2^2)$ is added to the objective function. Accordingly, the objective function of **(P5-μ)**-$\mathcal{X}_{\mathcal{P}_i}$ is $\mu(N - \|\sigma + \sigma^{\star_3}\|_2^2)$. Notice that **(P3-$\mu$)**-$\mathcal{X}_{\mathcal{P}_i}$ can be viewed as a convexified optimization of **(P5-μ)**-$\mathcal{X}_{\mathcal{P}_i}$ by removing the concave components. This correlation will be helpful for bound refinements. Figure 1 illustrates how the optimizations are related. The bound refinement requires solving **(P3-μ)**-$\mathcal{X}_{\mathcal{P}_i}$ with the optimal solution given by (x_μ, σ_μ), and finding $\Delta\sigma_\mu$ such that

$$\begin{aligned} \Delta\sigma_\mu = \operatorname{argmin}_\sigma\ & \sigma^\top \sigma^{\star_3}, \\ \text{s.t. } & \sigma^{\star_3} + \sigma \in [-1,1]^N \cap \mathcal{X}_{\mathcal{P}_i}\,. \end{aligned} \tag{15}$$

Solving (15) is a simple linear programming with ignorable complexity compared to **(P3-μ)**-$\mathcal{X}_{\mathcal{P}_i}$. Proposition 4.3 shows how the bounds are refined by (x_μ, σ_μ) and $\Delta\sigma_\mu$.

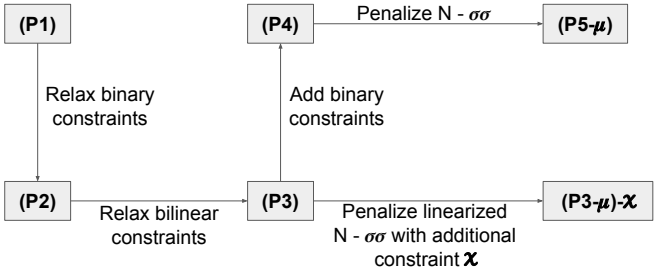

Fig. 1. Summary for how the optimization formulations are related.

Proposition 4.3. ***(Lower and Upper Bounds).*** If $\mu \geq L$ and $\sigma^{\star_3} + \sigma_\mu \notin \{-1,1\}^N$, then

(a) $f(x_\mu, \sigma_\mu + \sigma^{\star_3}) \geq f(x^{\star_3}, \sigma^{\star_3})$,

(b) **Lower bound:** $\forall \sigma \in \mathcal{X}_{\mathcal{P}_i} \cap \{-1,1\}^N$ and $(x,\sigma) \in \mathcal{X}$,
$$f(x_\mu, \sigma_\mu + \sigma^{\star_3}) - 2\mu(\sigma_\mu - \Delta\sigma_\mu)^\top \sigma^{\star_3} \leq f(x,\sigma), \tag{16}$$

(c) **Upper bound:** $\exists \sigma \in \mathcal{X}_{\mathcal{P}_i} \cap \{-1,1\}^N$ and $(x,\sigma) \in \mathcal{X}$ such that
$$f(x_\mu, \sigma_\mu + \sigma^{\star_3}) + \mu(N - \|\sigma_\mu + \sigma^{\star_3}\|_2^2) \geq f(x,\sigma). \tag{17}$$

Notice that though we have identified the LHS of (16) is a lower bound, it is not necessary bigger than $f(x^{\star_3}, \sigma^{\star_3})$ (depending on the value of $-2\mu(\sigma_\mu - \Delta\sigma_\mu)^\top \sigma^{\star_3}$) A proper way to choose the lower bound is then naturally

$$\max\{f(x^{\star_3}, \sigma^{\star_3}),\ \ f(x_\mu, \sigma_\mu) - 2\mu(\sigma_\mu - \Delta\sigma_\mu)^\top \sigma^{\star_3}\}.$$

One the other hand, (17) does not provide a candidate integer solution for the branch. We therefore regard (17) as a way to refine an upper bound obtained by some standard ways, such as polynomial-time complexity algorithms that find a local optimum of **(P4)**.

Although B&B can avoid checking all the combinations in σ, it can still take long to find the global optimal solution. In the interest of reducing the computational complexity, we instead pursue finding a quality solution efficiently rather than the global optimum. We consider that a sufficiently good solution has been found if there is a branch i which satisfies

$$\mathrm{UB}_i - \mathrm{LB}_i \leq \epsilon, \quad \epsilon > 0, \tag{18}$$

where UB_i and LB_i respectively refer to the upper and lower bounds for branch i. To facilitate getting a branch that satisfies (18), we prioritize branching on a node that

has the smallest $\mathrm{UB}_i - \mathrm{LB}_i$. Let Ł be the set of leaf nodes in the binary enumeration tree. Algorithm 2 summarizes the B&B method that pursues a quality solution efficiently instead of the global optimum. For convenience of presentation, we will call the B&B with the rounding branching and bound-refinement (16)-(16) as Early Termination B&B (ETB&B) though it is essentially a special variant of B&B. The update of Ł replaces the branched node by its

Algorithm 2 ETB&B

1: **Initialize:**
   $\epsilon > 0$, $i = 0$, Ł $= \{0\}$, $M \in \mathbb{N}$, count $= 0$
2: **Solve (P3)** (for LB_i) and find a UB_i
3: **Refine** LB_i and UB_i by (16) and (17)
4: **while** $(\mathrm{UB}_i - \mathrm{LB}_i \geq \epsilon, \forall i \in$ Ł$)$ or $(\text{count} \leq M)$ **do**
5: count $\leftarrow$ count $+1$
6: **Compute** $i = \operatorname{argmin}_{k \in \text{Ł}} \mathrm{UB}_k - \mathrm{LB}_k$
7: **Run** Algorithm 1 for node i
8: **Solve (P3)-$\mathcal{X}_{u_i}$** and the associated UB and LB
9: **Solve (P3)-$\mathcal{X}_{d_i}$** and the associated UB and LB
10: **Update** Ł, UB_k and LB_k for all $k \in$ Ł
11: **end while**
12: **Return** the incumbent solution

children nodes. Notice that we do not restrict the way of computing the upper bound in Algorithm 2. In addition, we impose another termination criteria (count $> M$) in Algorithm 2 to restrict the total number of branches.

5. NUMERICAL STUDIES

Our simulations compare the proposed method against the work in (Taha et al., 2017). The comparison is two-fold. One is on the relaxation of the bilinear constraints in **(P1)**. The other is on the derivation of a feasible solution. We will also compare the proposed ETB&B against B&B (most infeasible branching). All the simulations are done on a desktop with 3.5GHz CPU and 16GB RAM, using MATLAB and its CVX toolbox (Grant and Boyd, 2014) to solve the convex optimization problems.

We consider the test cases in (Taha et al., 2017) which has a random network with following structure

$$\dot{x}_i = -\begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} x_i + \sum_{i \neq k} e^{a(i,k)} x_k + \begin{bmatrix} 0 \\ 1 \end{bmatrix} (\pi_i u_i + w_i), \quad (19)$$

where $a(i,k)$ is randomly generated with values in $[-\frac{N}{5}, \frac{N}{5}]$, and N is the number of nodes. We assume that C and D in (1) are respectively an identity matrix and a zero matrix. The constraint for the actuator selection is rather simple with $H = \mathbb{1}^\top$ and $h = N/5$ in **(P1)**. The constraint makes sure that the number of activated actuators is no less than $N/5$. More detailed discussions about test system (19) are in (Taha et al., 2017).

We first compare the relaxation on the bilinear constraints by SDP-R (Taha et al., 2017) and **(P3)**. Both methods give almost exactly the same lower bound for **(P1)**. The main difference is on the computational time, shown in Table 1. It is unsurprising to see improvements of the

Table 1. Computational time.

	$N = 10$	$N = 15$	$N = 20$
SDP-R	10.51	40.70	154.50
(P3)	3.74	8.82	29.42

computational time by solving **(P3)** instead of SDP-R, because SDP-R introduces N number of additional linear matrix inequalities in the approximation.

We next implement the ETB&B method to find a candidate solution for **(P1)** associated with (19). We choose $M = 10$ in Algorithm 2, and choose the incumbent solution when it terminates. Each branch i finds the lower bound by solving **(P3)-$\mathcal{X}_{\mathcal{P}_i}$**. The upper bound is simply a feasible solution that is closest to the rounded solution of **(P3)-$\mathcal{X}_{\mathcal{P}_i}$**. The lower and upper bounds are then refined by (16) and (17). Table 2 compares the solutions derived from the ETB&B and the directed rounding method in (Taha et al., 2017). The computational time of ETB&B grows linearly

Table 2. Solutions of rounding method and ETB&B.

	$N = 10$	$N = 15$	$N = 20$
Rounded soln.	6.67	9.87	8.76
ETB&B soln.	4.42	3.64	7.40
LB of ETB&B	3.29	3.5	5.50

with respect to the number of branches. Namely, if the computational time the rounding method is T, then the one for the ETB&B is MT. This could be dramatically reduced by sophisticated coding such as parallel computing. Figure 2 illustrates that quality of the solution in general improves when M increases.

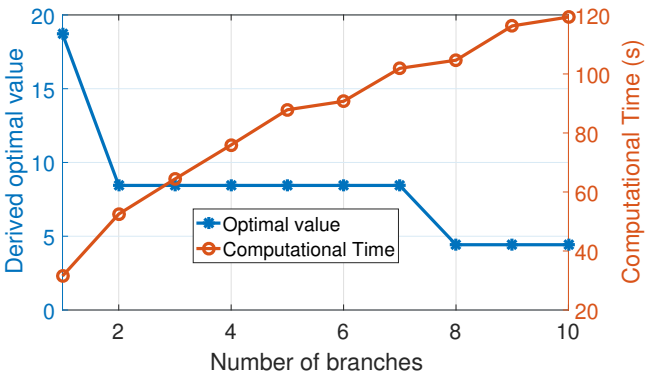

Fig. 2. Quality of the solution and the computational time.

The next part of the numerical analysis is on the bound refinement by (16) and (17). We compare in Figure 3 about the final solutions of ETB&B with and without the bound refinement for one case with $N = 10$. Figure 3 also includes numerical result for standard B&B. We further show in Figure 4 regarding to the evolution of $\mathrm{UB} - \mathrm{LB}$ for completeness. Note that although Figure 3 suggests that the rounding branching method performs better than the standard most infeasible branching method, we observe in a number cases that the rounding branching method do worse. However, the bound refinement (16) and (17) consistently lead to better convergence properties regardless on which branching method is implemented.

In all the numerical examples above, we choose $\overline{z}_i = 10$ for all $i = 1, \cdots, N$, but observe that the retrieved $\|Z_i\|_F$ does not necessary equal $\overline{z}_i$. Though as analyzed in Section 3, there exists an optimal solution with $\|Z_i\|_F = \overline{z}_i$, the solver does not return it. This confirms that choosing $\overline{z}_i = 10$ makes constraint (5) unbinding. The rigorous selection of $\overline{z}$ is an interesting direction for future research.

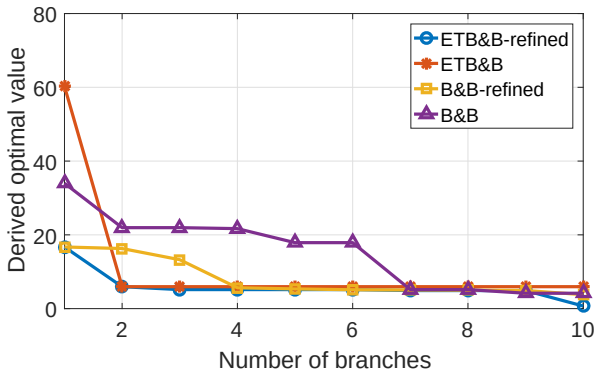

Fig. 3. Optimal values obtained through refined and non-refined B&B methods.

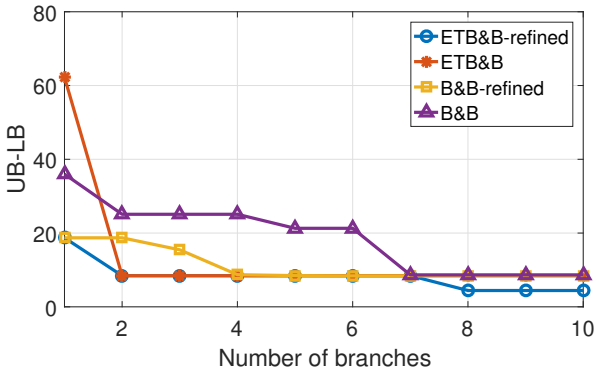

Fig. 4. Difference between UB and LB in various B&B.

6. CONCLUSIONS

This paper considers a co-optimization problem for control performance and actuator selection. This poses a general binary programming with BIBMIs. Our first contribution shows that it is sufficient to solve a binary-integer programming problem without BMIs. We next propose the bound refinement approach for B&B based on a continuous reformulation of the binary-integer programming problem. The bound refinement enhances the convergence of the B&B. Our future work is on improving the time complexity, including scalability of the SDP and a more sophisticated implementation of ETB&B.

ACKNOWLEDGMENTS

This research was supported by the ARPA-e Network Optimized Distributed Energy Systems (NODES) program, Cooperative Agreement DE-AR0000695.

REFERENCES

Argha, A., Su, S.W., Savkin, A., and Celler, B. (2018). A framework for optimal actuator/sensor selection in a control system. *International Journal of Control*. To appear.

Burer, S. and Letchford, A.N. (2012). Non-convex mixed-integer nonlinear programming: A survey. *Surveys in Operations Research and Management Science*, 17(2), 97–106.

Dhingra, N.K., Jovanović, M., and Luo, Z.Q. (2014). An ADMM algorithm for optimal sensor and actuator selection. In *IEEE Conf. on Decision and Control*, 4039–4044.

Goemans, M.X. and Williamson, D.P. (1995). Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6), 1115–1145.

Grant, M. and Boyd, S. (2014). CVX: Matlab software for disciplined convex programming, version 2.1. Available at `http://cvxr.com/cvx`.

Jawaid, S.T. and Smith, S.L. (2015). Submodularity and greedy algorithms in sensor scheduling for linear dynamical systems. *Automatica*, 61, 282–288.

Johnson, E.L., Nemhauser, G.L., and Savelsbergh, M.W. (2000). Progress in linear programming-based algorithms for integer programming: An exposition. *Informs journal on computing*, 12(1), 2–23.

Joshi, S. and Boyd, S. (2009). Sensor selection via convex optimization. *IEEE Transactions on Signal Processing*, 57(2), 451–462.

Karlof, J.K. (2005). *Integer programming: theory and practice*. CRC Press.

Lenstra, J.K., Shmoys, D.B., and Tardos, E. (1990). Approximation algorithms for scheduling unrelated parallel machines. *Mathematical programming*, 46(1-3), 259–271.

Martin, A. (2001). General mixed integer programming: Computational issues for branch-and-cut algorithms. In *Computational combinatorial optimization*, 1–25.

Pancake, T., Corless, M., and Brockman, M. (2000). Analysis and control of polytopic uncertain/nonlinear systems in the presence of bounded disturbance inputs. In *American Control Conference*, 159–163. Chicago, IL.

Schrijver, A. (1998). *Theory of linear and integer programming*. John Wiley & Sons.

Summers, T.H. (2016). Actuator placement in networks using optimal control performance metrics. In *IEEE Conf. on Decision and Control*, 2703–2708. Las Vegas, NV.

Taha, A.F., Gatsis, N., Summers, T.H., and Nugroho, S. (2017). Time-varying sensor and actuator selection for uncertain cyber-physical systems. *arXiv preprint arXiv:1708.07912*.

Taylor, J.A., Luangsomboon, N., and Fooladivanda, D. (2017). Allocating sensors and actuators via optimal estimation and control. *IEEE Transactions on Control Systems Technology*, 25(3), 1060–1067.

Tzoumas, V., Rahimian, M.A., Pappas, G.J., and Jadbabaie, A. (2016). Minimal actuator placement with bounds on control effort. *IEEE Transactions on Control of Network Systems*, 3(1), 67–78.

Wu, B. and Ghanem, B. (2016). l_p-box ADMM: A versatile framework for integer programming. *arXiv preprint arXiv:1604.07666*.

Yuan, G. and Ghanem, B. (2016). Binary optimization via mathematical programming with equilibrium constraints. *arXiv preprint arXiv:1608.04425*.

Zhang, H., Ayoub, R., and Sundaram, S. (2017). Sensor selection for Kalman filtering of linear dynamical systems: Complexity, limitations and greedy algorithms. *Automatica*, 78, 202–210.