



HAL
open science

Event-Driven Joint Mobile Actuators Scheduling and Control in Cyber-Physical Systems

Lei Mo, Pengcheng You, Xianghui Cao, Ye-Qiong Song, Angeliki Kritikakou

► **To cite this version:**

Lei Mo, Pengcheng You, Xianghui Cao, Ye-Qiong Song, Angeliki Kritikakou. Event-Driven Joint Mobile Actuators Scheduling and Control in Cyber-Physical Systems. *IEEE Transactions on Industrial Informatics*, 2019, pp.1-13. 10.1109/TII.2019.2906061 . hal-02080647

HAL Id: hal-02080647

<https://inria.hal.science/hal-02080647v1>

Submitted on 26 Mar 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Event-Driven Joint Mobile Actuators Scheduling and Control in Cyber-Physical Systems

Lei Mo^{*}, *Member, IEEE*, Pengcheng You[†], *Student Member, IEEE*, Xianghui Cao[‡], *Senior Member, IEEE*,
Yeqiong Song[§], and Angeliki Kritikakou^{*}

Abstract—In cyber-physical systems, mobile actuators can enhance system’s flexibility and scalability, but at the same time incurs complex couplings in the scheduling and controlling of the actuators. In this paper, we propose a novel event-driven method aiming at satisfying a required level of control accuracy and saving energy consumption of the actuators, while guaranteeing a bounded action delay. We formulate a joint-design problem of both actuator scheduling and output control. To solve this problem, we propose a two-step optimization method. In the first step, the problem of actuator scheduling and action time allocation is decomposed into two subproblems. They are solved iteratively by utilizing the solution of one in the other. The convergence of this iterative algorithm is proved. In the second step, an on-line method is proposed to estimate the error and adjust the outputs of the actuators accordingly. Through simulations and experiments, we demonstrate the effectiveness of the proposed method.

Keywords—Cyber-physical systems, mobile actuator, scheduling, control, actuation delay, joint design, experiments.

I. INTRODUCTION

Cyber-physical systems (CPS) bridge the cyber world with the physical world using sensors and actuators connected through a wireless communication network. Sensing information and controlling actions through the CPS remove the limitations of wired connections and fixed network structures, and, thus, the flexibility and the scalability of the system can be further enhanced. The CPS are characterized by the sensor-actuator coordination, the heterogeneous information communication, and the intelligent decision/actuation [1]. These characteristics enable the use of CPS to several domains such as smart building [2], [3], power grid [4], mobile charging [5]–[8], industrial and environment control [9]–[12].

To support the requirements of the applications through the CPS, the sensors and the actuators have to efficiently coordinate with each other. For example, in the mobile charging application [5] as shown in Fig. 1, sensors perform environment monitoring, while actuators (i.e., mobile chargers) are responsible for charging the sensors. Each sensor reports its energy level periodically to the Base Station (BS). When the energy of a sensor is lower than a predefined threshold, an event is triggered at the BS. Based on the energy information collected from the sensors and the charging abilities of the actuators, the BS coordinates the actuators on how to perform the energy charging tasks. The coordination includes both the scheduling and the control of the actuators. The scheduling problem refers to the relocation of the actuators [5], [11], [13]. On the other

hand, the control problem refers to the output adjustment of the actuators [4], [9], [12], which directly influence the physical variables under consideration. Since different actuator scheduling schemes lead to different actuator control decisions, to find the optimal solution, the actuator scheduling and control problems should be jointly addressed.

The design of the scheduling of the mobile actuators and the control of their outputs should meet: 1) control accuracy requirement [6], e.g., to ensure the sensors have enough energy to work until being charged again, the replenished energy of the sensors should reach to some certain levels; 2) real-time requirement [8], e.g., the tasks assigned to the actuators (relocating and controlling) should be finished within a pre-defined threshold so as to generate a correct result; and 3) energy efficiency requirement [14]. Although the actuators are more powerful than the sensors, they still have limited energy budgets: 1) the actuators cannot be easily recharged after the deployment, and 2) the relocating and the controlling tasks consume much more energy than the computation and the communication tasks. To meet the above system requirements, the efficient coordination of the actuators is crucial.

Existing work on actuator coordination can be classified according to whether 1) the system runs with multiple actuators, 2) the actuator is mobile, 3) the problem is multi-objective, and 4) the solution is optimal. Table I provides a summary of some representative papers from the literature. The extension from single-actuator (stationary actuators or single-objective) case to multi-actuator (mobile actuators or multi-objective) case is not straightforward, since additional variables and constraints need to be added into the problem formulation to satisfy the new requirements. When taking multi-objective such as the multi-actuator scheduling and control, the real-time performance and the energy efficiency into account, the optimization variables are coupled with each other nonlinearly. Heuristics [5], [14] and evolutionary approaches [8] are popular methods to solve complex optimization problems. However, the solution qualities of these methods are hard to guarantee. Finding an optimal solution is also very important. Only by doing so, we can find out how far the non-optimal solution is from the optimal one, and how to improve the non-optimal approaches based on the optimal solution.

Complementary to the state of the art, this paper jointly optimizes the multi-actuator scheduling and control under the control accuracy, the energy and the real-time constraints and provides an optimal solution with reduced computing time. Our main contributions are summarized as follows.

- 1) To jointly optimize the scheduling and the control of the actuators under multiple system requirements, we introduce an Actuator Scheduling and Action Time Allocation (ASATA) problem. The ASATA problem is a Mixed-Integer Program (MIP) problem. To reduce the computational complexity, the MIP-based ASATA

^{*}University of Rennes, INRIA, CNRS, IRISA, 35042 Rennes, France. E-mails: lei.mo@inria.fr, angeliki.kritikakou@irisa.fr.

[†]The Whiting School of Engineering, Johns Hopkins University, 21218 Baltimore, USA. E-mail: pcyou@jhu.edu.

[‡]School of Automation, Southeast University, 210096 Nanjing, China. E-mail: xhcao@seu.edu.cn.

[§]University of Lorraine, CNRS, INRIA, LORIA, 54000 Nancy, France. E-mail: ye-qiong.song@loria.fr.

References		With stationary actuators									With mobile actuators								
		[2]	[3]	[4]	[9]	[10]	[12]	[15]	[16]	[17]	[5]	[6]	[7]	[8]	[11]	[13]	[14]	[18]	This paper
Actuator	Single	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Multiple	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Objective	Control accuracy	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Task real-time	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Energy efficiency	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Solution	Non-optimal	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Optimal	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

TABLE I. CLASSIFICATION OF NODE COORDINATION METHODS

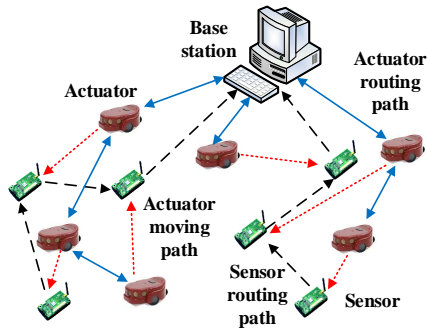


Fig. 1. An example of CPS with mobile actuators.

problem is relaxed to an MILP. Instead of periodically updating the actuator scheduling and action time decisions, the decisions are updated only when events occur.

- 2) To efficiently solve the MILP-based ASATA problem, we divide this problem into two correlated subproblems: an Integer Linear Program (ILP)-based subproblem, dedicated to the actuator scheduling (Master Problem – MP), and an LP-based subproblem, dedicated to the action time allocation (Slave Problem – SP). The correlated subproblems are solved in an iterative way, where the solution of the one subproblem is propagated to the other. We prove that through limited iterations between the MP and the SP, the algorithm converges to the global optimal solution.
- 3) Using the solution of the ASATA problem to schedule the actuators and adjust their action time, an error is introduced, since the actuators are mobile and the system states are dynamic. To enhance the control accuracy, as well as to eliminate the performance degradation caused by the problem relaxation from MIP to MILP, we propose an on-line method to estimate the introduced error. On this basis, we introduce an LP-based Actuator Output Adjustment (AOA) problem to adjust the outputs of the actuators.
- 4) We evaluate the performance of the proposed method by both simulations and experiments that are based on a physical testbed.

The remainder of this paper is organized as follows. Section II presents the related work. Section III introduces the system model and formulates the problem under study. Section IV presents the joint-design algorithm. Section V shows the simulation and experimental results. Section VI concludes this study.

II. RELATED WORK

With respect to the actuator control problem, the requirements on the control accuracy should be satisfied [2]. On this basis, the control quality can be further enhanced by reducing the action delay [12] and packet-loss rate [10] or resisting the disturbances [16] and inaccurate system parameters [9].

Some works consider the network delay, energy efficiency and control accuracy joint optimization [3] or the communication protocol and control accuracy joint optimization [4], [15], [17]. However, the aforementioned studies mainly focus on the control of the stationary actuators.

With respect to the actuator scheduling problem, studies exist for single-actuator and multi-actuator cases. For the single-actuator scheduling case, the location and the emergency of an event are estimated by the Maximum Likelihood Estimation (MLE) and the actuator is scheduled by the Markov Decision Processes (MDPs) to handle this event in [11]. The Traveling Salesman Problem (TSP) [13] and the Orienteering Problem (OP) [7] can be used to formulate the actuator scheduling problems. The basic idea of TSP (or OP) is utilizing a graph to model the system, where the vertexes are usually associated with the profits (e.g., the priorities of the events) while the edges are usually associated with the costs (e.g., the moving time or moving energy of the actuator). The aim of the problem is to maximize the profits (or minimize the costs), under the constraints that all the vertexes must be visited (or the moving time is limited). For the multi-actuator scheduling case, the TSP can be extended to Multiple Traveling Salesman Problem (m-TSP) [5]. In [18], to balance the workloads (i.e., travel cost and data collection cost) of the actuators, the sensors are divided into several groups and the actuators are scheduled to visit these groups in sequence. However, the actuator output control problem is not taken into account in these studies, since the actuators are mainly used to visit the event area [11] or collect data from the sensors [13], [18] or are required to arrive at the sensors before the energy of the sensors is lower than a predefined threshold [5], [7].

In [14], considering the energy consumed by the actuators to move and perform action, the actuator scheduling and control problem is formulated by Mixed-Integer Non-Linear Program (MINLP). In [6], [8], the actuator is scheduled to visit the sensors in sequence and changes their energy to some certain levels. However, the dynamics of the system states is not taken into account in [6], [8], [14]. During the movement of the actuators, the system states will change, and, thus, the actuator control decision should be updated accordingly. For the dynamic system, it will introduce a nonlinear coupling among the optimization variables (i.e., the actuator scheduling and control decisions). The mobile actuator scheduling and/or control problems are usually NP-hard. The common solutions include: 1) heuristics, e.g., greedy and approximation algorithms [5], [7], [13], [14], [18], 2) evolutionary approaches, e.g. genetic algorithm [8], and 3) problem relaxation, e.g., under specific conditions or assumptions, the Non-Linear Program (NLP) problem can be transformed to an MILP problem [6]. Although the heuristics are able to find the feasible solution in a short time, they do not provide the bounds on solution quality, and are sensitive to changes in the problem structures.

III. SYSTEM MODEL AND PROBLEM FORMULATION

In this section, we first present the system model with stationary sensors and mobile actuators. And then, we formulate the joint-design problem of multiple actuators scheduling and control. The main notations are summarized in Table II.

n_s	number of sensors
n_a	number of actuators
n_p	number of system states
n_l	number of working points
\mathbb{L}_i	the i^{th} working point
\mathbb{A}_j	the j^{th} actuator
\mathbb{S}_l	the l^{th} sensor
$\mathbf{x}(k)$	system states at the k^{th} step
$\mathbf{u}(k)$	actuators' outputs at the k^{th} step
$\mathbf{z}(k)$	sensors' measurements at the k^{th} step
Δ_s	system's sampling period
τ	action delay
$[\underline{\mathbf{x}}_{th}, \bar{\mathbf{x}}_{th}]$	event trigger threshold
$[\underline{\mathbf{x}}, \bar{\mathbf{x}}]$	user's requirement on \mathbf{x}
$[\underline{\mathbf{u}}, \bar{\mathbf{u}}]$	bounds of actuators' outputs
$\mathbf{x}(k k)$	a posteriori estimate of $\mathbf{x}(k)$
$\mathbf{x}(k+1 k)$	a priori estimate of $\mathbf{x}(k)$
$d_{ij}(k)$	distance between \mathbb{L}_i and \mathbb{A}_j at the k^{th} step
v	mean velocity of \mathbb{A}_j
u_j	output amplitude of \mathbb{A}_j during the action time before output adjustment
t_j^m	maximum action time of \mathbb{A}_j
k_d	energy consumed by actuator to move a unit distance
k_u	coefficient of actuator's action energy
s_{ij}	$= \begin{cases} 1 & \text{if } \mathbb{A}_j \text{ is scheduled to } \mathbb{L}_i \\ 0 & \text{otherwise} \end{cases}$
t_{ij}	action time of \mathbb{A}_j at \mathbb{L}_i
$\Delta u_j(k)$	output adjustment of \mathbb{A}_j at the k^{th} step

TABLE II. MAIN NOTATIONS

A. System Model

We consider n_s static sensors $\{\mathbb{S}_1, \dots, \mathbb{S}_{n_s}\}$ and n_a mobile actuators $\{\mathbb{A}_1, \dots, \mathbb{A}_{n_a}\}$ are randomly deployed in a Region Of Interest (ROI) to monitor n_p system states $\{x_1, \dots, x_{n_p}\}$ and take necessary actions to deal with the events in that area, respectively. The system states represent the physical variables under consideration. We assume that 1) the velocity of the actuator is constant and no obstacle exists in the ROI, similar to the previously published works [6], [13], [14], 2) there are n_l working points $\{\mathbb{L}_1, \dots, \mathbb{L}_{n_l}\}$ that the actuators can stay and perform actions, and 3) the actuator \mathbb{A}_j starts performing actions only when it arrives at the designed working point.

Since the information exchange among the sensors and the actuators is carried out by discrete data packets, we consider a linear discrete-time model for the dynamic physical system:

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}(k)\mathbf{u}(k) + \boldsymbol{\omega}(k), \quad (1)$$

where $\mathbf{x}(k) = [x_1(k), \dots, x_{n_p}(k)]'$. \mathbf{A} and $\mathbf{B}(k)$ are the system matrix and the input matrix with appropriate dimensions, respectively. Note that the input matrix $\mathbf{B}(k)$ may vary in different time steps depending on the scheduling of actuators. This is because the actuators may be scheduled to

move to different working points, and only those active ones that arrive at the designated working points start performing control actions. The system noises $\boldsymbol{\omega}(k) \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$, and the outputs of the actuators are bounded by $\underline{\mathbf{u}} \preceq \mathbf{u}(k) \preceq \bar{\mathbf{u}}$. $\mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$ represents the random variables \mathbf{y} follow a Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\sigma}$. $\mathbf{p} \preceq \mathbf{q}$ represents $p_i \leq q_i$ ($1 \leq i \leq m$), where $\mathbf{p} = [p_1 \dots, p_m]'$ and $\mathbf{q} = [q_1 \dots, q_m]'$.

The sensors measure the system states $\mathbf{x}(k)$ as follows:

$$\mathbf{z}(k) = \mathbf{C}\mathbf{x}(k) + \boldsymbol{\nu}(k), \quad (2)$$

where $\mathbf{z}(k) = [z_1(k), \dots, z_{n_s}(k)]'$ are the measurements of the sensors, \mathbf{C} is the measurement matrix, and the measurement noises $\boldsymbol{\nu}(k) \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$.

For the system described by equations (1) and (2), Kalman Filter (KF) is an optimal estimator since it provides a minimum variance unbiased estimate of system states $\mathbf{x}(k)$. Therefore, we run KF at each step k to estimate the system states $\mathbf{x}(k)$ from the noisy measurements $\mathbf{z}(k)$:

$$\begin{aligned} \mathbf{x}(k+1|k) &= \mathbf{A}\mathbf{x}(k|k) + \mathbf{B}(k)\mathbf{u}(k), \\ \mathbf{P}(k+1|k) &= \mathbf{A}\mathbf{P}(k|k)\mathbf{A}' + \mathbf{Q}, \\ \mathbf{G}(k) &= \mathbf{P}(k+1|k)\mathbf{C}'(\mathbf{C}\mathbf{P}(k|k-1)\mathbf{C}' + \mathbf{R})^{-1}, \\ \mathbf{x}(k|k) &= \mathbf{x}(k|k-1) + \mathbf{G}(k)(\mathbf{z}(k) - \mathbf{C}\mathbf{x}(k|k-1)), \\ \mathbf{P}(k|k) &= \mathbf{P}(k|k-1) - \mathbf{G}(k)\mathbf{C}\mathbf{P}(k|k-1), \end{aligned}$$

where $\mathbf{G}(k)$ is the Kalman gain, $\mathbf{x}(k|k)$ and $\mathbf{x}(k+1|k)$ are the a posteriori and the a priori estimates of the system states $\mathbf{x}(k)$, respectively. $\mathbf{P}(k|k)$ and $\mathbf{P}(k+1|k)$ are the error covariance matrices with respect to the estimates $\mathbf{x}(k|k)$ and $\mathbf{x}(k+1|k)$, respectively. The communication delay is not considered in this paper.

Definition 3.1 (Event): If the state estimate $\mathbf{x}(k|k)$ is out of a predefined threshold $[\underline{\mathbf{x}}_{th}, \bar{\mathbf{x}}_{th}]$, while the state estimate $\mathbf{x}(k-1|k-1)$ was within that threshold, an event occurs at the k^{th} step.

Definition 3.2 (Action delay): For an event, the action delay τ is the number of sampling periods after the event occurrence during which the user's requirement on the system states, i.e., $[\underline{\mathbf{x}}, \bar{\mathbf{x}}]$, remains unsatisfied.

Definition 3.3 (Action time): The action time of actuator \mathbb{A}_j is the number of sampling periods during which the \mathbb{A}_j 's output is not equal to zero, i.e., $u_j(k) \neq 0$.

The sensors $\{\mathbb{S}_1, \dots, \mathbb{S}_{n_s}\}$ transmit their measurements $\mathbf{z}(k)$ to the BS at each step k to estimate the system states $\mathbf{x}(k)$. Without losing generality, we assume that an event occurs at the k^{th} step. Therefore, the BS schedules the actuators to the designed working points, and adjusts their outputs $\mathbf{u}(k)$ to control the system states $\mathbf{x}(k+\tau|k+\tau-1)$ to meet user's requirement $[\underline{\mathbf{x}}, \bar{\mathbf{x}}]$. Based on different applications, the ranges $[\underline{\mathbf{x}}_{th}, \bar{\mathbf{x}}_{th}]$ and $[\underline{\mathbf{x}}, \bar{\mathbf{x}}]$ may be different.

B. Preliminaries

Equation (1) shows that the system states $\mathbf{x}(k+\tau)$ are determined by the input matrices $\{\mathbf{B}(k), \dots, \mathbf{B}(k+\tau-1)\}$ and the outputs of the actuators $\{\mathbf{u}(k), \dots, \mathbf{u}(k+\tau-1)\}$. Due to the product of the variables $\mathbf{B}(l)\mathbf{u}(l)$ ($k \leq l \leq k+\tau-1$), it's difficult to solve $\mathbf{B}(l)$ and $\mathbf{u}(l)$ directly. Alternatively, we fix the output amplitude of the actuators as $\mathbf{u} = [u_1, \dots, u_{n_a}]'$ during the action time, and introduce binary matrix $\mathbf{S} = [s_{ij}]_{n_l \times n_a}$ ($s_{ij} \in \{0, 1\}$) and integer matrix $\mathbf{T} = [t_{ij}]_{n_l \times n_a}$ ($t_{ij} \in \mathbb{Z}^+$) to schedule the actuators and adjust their action time during

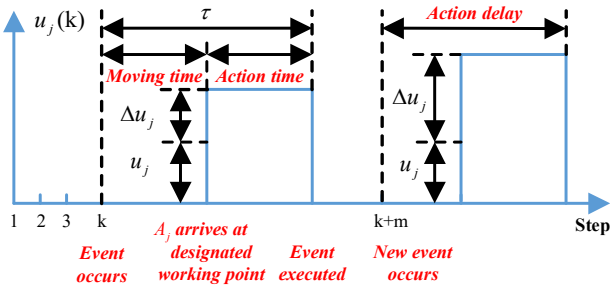


Fig. 2. The control sequence of the actuator \mathbb{A}_j .

the steps $[k, k + \tau - 1]$. This problem is called Actuator Scheduling and Action Time Allocation (ASATA) problem, which is detailed in Section III-C. The benefit of introducing variables \mathbf{S} and \mathbf{T} to the problem formulation is that they are coupled with each other linearly, which makes the problem easier to solve. On the other hand, to enhance the control accuracy, based on the solution of the ASATA problem, we introduce a continuous vector $\Delta \mathbf{u}(l) = [\Delta u_1(l), \dots, \Delta u_{n_a}(l)]'$ ($\Delta u_j(l) \in \mathbb{R}$) to adjust the outputs of the actuators at the l^{th} step ($k \leq l \leq k + \tau - 1$). This problem is called Actuator Output Adjustment (AOA) problem, which is detailed in Section III-D. If the values of variables \mathbf{S} , \mathbf{T} and $\Delta \mathbf{u}(l)$ are determined, the input matrix $\mathbf{B}(k)$ and the outputs of the actuators $\mathbf{u}(k)$ during the steps $[k, k + \tau - 1]$ are calculated as follows:

- 1) Following the scheduling decision \mathbf{S} to move the actuators, the location of the actuator \mathbb{A}_j at the l^{th} step ($k \leq l \leq k + \tau - 1$) is known. We assume that at the l^{th} step the relocation of the actuators $\{\mathbb{A}_1, \dots, \mathbb{A}_j\}$ has been completed while the actuators $\{\mathbb{A}_{j+1}, \dots, \mathbb{A}_{n_a}\}$ are still moving. With the given sampling period Δ_s , we obtain an $n_p \times j$ matrix $\tilde{\mathbf{B}}$ by using the methods in [2], [10], and, thus, the input matrix $\mathbf{B}(l) = [\tilde{\mathbf{B}}, \mathbf{0}_{n_p \times (n_a - j)}]$. This matrix remains constant until a new actuator (e.g., \mathbb{A}_{j+1}) arrives at its designed working point.
- 2) Denote $d_{ij}^s(k) = \lceil \frac{d_{ij}(k)}{v \Delta_s} \rceil$ as the number of steps required for the actuator to move $d_{ij}(k)$ distance, where $\lceil y \rceil \triangleq \min\{n \in \mathbb{Z} | y \leq n\}$. According to the scheduling decision \mathbf{S} and the action time decision \mathbf{T} , the actuator \mathbb{A}_j takes $\sum_{i=1}^{n_i} s_{ij} d_{ij}^s(k)$ steps to move and $\sum_{i=1}^{n_i} t_{ij}$ steps to perform the actions. Therefore, the output of the actuator \mathbb{A}_j at the l^{th} step is

$$u_j(l) = \begin{cases} u_j, & \text{if } k + \sum_{i=1}^{n_i} s_{ij} d_{ij}^s(k) \leq l \leq k \\ & + \sum_{i=1}^{n_i} s_{ij} d_{ij}^s(k) + \sum_{i=1}^{n_i} t_{ij}, \\ 0, & \text{otherwise.} \end{cases}$$

- 3) Based on the control decisions $\mathbf{u}(l)$ and $\Delta \mathbf{u}(l)$, the outputs of the actuators at the l^{th} step are given by $\mathbf{u}(l) + \Delta \mathbf{u}(l)$, as shown in Fig. 2.

From the above statement, we can see that the ASATA problem and the AOA problem are solved in sequence. In the following sections, we explain how to formulate the actuator scheduling and control joint-design problem through the new variables \mathbf{S} , \mathbf{T} and $\Delta \mathbf{u}(l)$.

C. Actuator Scheduling and Action Time Allocation Problem

The problem consists of an objective function that minimizes the moving and the action energy consumption of the actuators subject to the control accuracy and the action real-time

constraints. Under these constraints, we determine 1) which working point should the actuator be assigned to (scheduling); and 2) when the actuators start and end the actions (action time allocation). Let $\mathcal{I} \triangleq \{1, \dots, n_i\}$, $\mathcal{J} \triangleq \{1, \dots, n_a\}$ and $\mathcal{M} \triangleq \{1, \dots, n_p\}$. Since each actuator moves toward at most one working point, the scheduling variable s_{ij} should satisfy the inequality:

$$\sum_{i=1}^{n_i} s_{ij} \leq 1, \quad \forall j \in \mathcal{J}. \quad (3)$$

The maximum action time of the actuator \mathbb{A}_j is limited by its residual energy [6]. Hence, the action time variable t_{ij} is bounded by

$$0 \leq t_{ij} \leq t_j^m s_{ij}, \quad \forall i \in \mathcal{I}, \quad \forall j \in \mathcal{J}. \quad (4)$$

To finish the relocating and the controlling tasks within the action delay τ , we have

$$s_{ij} d_{ij}^s(k) + t_{ij} \leq \tau, \quad \forall i \in \mathcal{I}, \quad \forall j \in \mathcal{J}. \quad (5)$$

Note that the system states are hard to precisely estimate when the scheduling and the action time decisions \mathbf{S} and \mathbf{T} are unknown. In the ASATA problem, we let the system matrix $\mathbf{A} = \mathbf{I}$. To meet the control requirement $[\underline{x}, \bar{x}]$, we have

$$\underline{x}_m \leq x_m(k|k) + \sum_{i=1}^{n_i} \sum_{j=1}^{n_a} u_j t_{ij} \tilde{b}_{mi} \leq \bar{x}_m, \quad \forall m \in \mathcal{M}. \quad (6)$$

\tilde{b}_{mi} is the $(m, i)^{\text{th}}$ element of input matrix $\tilde{\mathbf{B}}$ and it evaluates the influence of the actuators exerts on the system state x_m , where these actuators are placed to the working point \mathbb{L}_i . Note that matrices $\tilde{\mathbf{B}}$ and \mathbf{B} are different. First, the dimension of $\tilde{\mathbf{B}}$ is $n_p \times n_i$ while the dimension of \mathbf{B} is $n_p \times n_a$. Second, $\tilde{\mathbf{B}}$ is fixed and given in advance while \mathbf{B} is time-varying and determined by the scheduling decision. In Section III-D, we explain the necessity to introduce constraint (6) and how to extend it to the case $\mathbf{A} \neq \mathbf{I}$ through the AOA problem.

Our objective is to minimize the moving and the action energy consumption of the actuators. Specifically, the moving and action energy are assumed to be proportional to the moving distance and the action time, respectively [6]. Therefore, the objective function is $\Phi(\mathbf{S}, \mathbf{T}) = \sum_{i=1}^{n_i} \sum_{j=1}^{n_a} (k_d s_{ij} d_{ij}(k) + k_u |u_j| t_{ij})$. Summarizing the objective and the aforementioned constraints, the ASATA problem is formulated as

$$\mathbf{P1} : \min_{\mathbf{S}, \mathbf{T}} \Phi(\mathbf{S}, \mathbf{T}) \quad (7)$$

$$\text{s.t.} \begin{cases} (3), (4), (5), (6), \\ s_{ij} \in \{0, 1\}, t_{ij} \in \mathbb{Z}^+, 0 \leq t_{ij} \leq t_j^m, \quad \forall i \in \mathcal{I}, \quad \forall j \in \mathcal{J}. \end{cases}$$

Since $s_{ij} \in \{0, 1\}$ and $t_{ij} \in \mathbb{Z}^+$ ($\forall i \in \mathcal{I}, \forall j \in \mathcal{J}$), the P1 is an MIP problem. To reduce the computational complexity, we relax the P1 to the following MILP problem:

$$\mathbf{P2} : \min_{\mathbf{S}, \hat{\mathbf{T}}} \sum_{i=1}^{n_i} \sum_{j=1}^{n_a} (k_d s_{ij} d_{ij} + k_u |u_j| \hat{t}_{ij}) \quad (8)$$

$$\text{s.t.} \begin{cases} (3), \\ \hat{t}_{ij} \leq t_j^m s_{ij}, \quad \forall i \in \mathcal{I}, \quad \forall j \in \mathcal{J}, \\ s_{ij} d_{ij}^s(k) + \hat{t}_{ij} \leq \tau, \quad \forall i \in \mathcal{I}, \quad \forall j \in \mathcal{J}, \\ \underline{x}_m \leq x_m(k|k) + \sum_{i=1}^{n_i} \sum_{j=1}^{n_a} u_j \hat{t}_{ij} \tilde{b}_{mi} \leq \bar{x}_m, \quad \forall m \in \mathcal{M}, \\ s_{ij} \in \{0, 1\}, \hat{t}_{ij} \in \mathbb{R}^+, 0 \leq \hat{t}_{ij} \leq t_j^m, \quad \forall i \in \mathcal{I}, \quad \forall j \in \mathcal{J}. \end{cases}$$

Finding an optimal scheduling decision is the most important step to solve P2. Since if the value of binary variables \mathbf{S} is

determined, P2 reduces to an LP problem, which has a simpler structure and is more easy to solve. In contrast, for the P1, even if the value of \mathbf{S} is given, we still need to solve an ILP problem. Solving an ILP problem is more complex than solving an LP problem, especially when the problem size is large [19].

Equations (1) and (2) show that system is discrete, and, thus, each actuator maintains the output step-wise. We define $\tilde{\mathbf{T}} = [\tilde{t}_{ij}]_{n_i \times n_a}$ as the real action time matrix, where $\tilde{t}_{ij} = \lfloor t_{ij} \rfloor$ and $\lfloor y \rfloor \triangleq \max\{n \in \mathbb{Z} | n \leq y\}$. Hence, the actuator \mathbb{A}_j takes $\sum_{i=1}^{n_i} \tilde{t}_{ij}$ steps to perform actions. Since $\tilde{t}_{ij} \leq t_{ij} \leq t_{ij} + 1$ ($\forall i \in \mathcal{I}, \forall j \in \mathcal{J}$), the real action time decision $\tilde{\mathbf{T}}$ does not violate the constraints (4) and (5) except the constraint (6). In Section III-D, we explain how to mitigate this influence through the AOA problem.

D. Actuator Output Adjustment Problem

When $\mathbf{A} \neq \mathbf{I}$, the constraint (6) should be constructed by using the predicted system states. For example, at the k^{th} step, by solving the P2, the actuator \mathbb{A}_1 is scheduled to the working point \mathbb{L}_1 to control the system state x_1 , and the moving time takes k_1 steps. When the actuator \mathbb{A}_1 arrives at the designed working point \mathbb{L}_1 and starts acting, the state of x_1 has been changed from $x_1(k)$ to $x_1(k+k_1)$. If the scheduling decision \mathbf{S} and the action time decision $\tilde{\mathbf{T}}$ are given, we obtain the input matrices $\{\mathbf{B}(k), \dots, \mathbf{B}(k+\tau-1)\}$ and the outputs of the actuators $\{\mathbf{u}(k), \dots, \mathbf{u}(k+\tau-1)\}$. Therefore, the system states at the l^{th} step ($k+1 \leq l \leq k+\tau$) are estimated by

$$\mathbf{x}(l|k) = \mathbf{A}^{l-k} \mathbf{x}(k|k) + \sum_{i=k}^{l-1} \mathbf{A}^{l-1-i} \mathbf{B}(i) \mathbf{u}(i).$$

Note that $\mathbf{x}(l|k)$ is a KF-based priori estimate. The posteriori estimates $\{\mathbf{x}(k+1|k+1), \dots, \mathbf{x}(k+\tau|k+\tau)\}$ are unavailable since the future measurements $\{\mathbf{z}(k+1), \dots, \mathbf{z}(k+\tau)\}$ are unknown at the current step k . In contrast, if the scheduling decision \mathbf{S} and the action time decision $\tilde{\mathbf{T}}$ are unknown, the input matrix $\mathbf{B}(l)$, the outputs of the actuators $\mathbf{u}(l)$, and the priori estimate $\mathbf{x}(l|k)$ ($k+1 \leq l \leq k+\tau$) are hard to derive. This is because different \mathbf{S} and $\tilde{\mathbf{T}}$ decisions lead to different values of $\mathbf{B}(l)$, $\mathbf{u}(l)$ and $\mathbf{x}(l|k)$. In addition, it's difficult to formulate the functions of $\mathbf{B}(l)$, $\mathbf{u}(l)$ and $\mathbf{x}(l|k)$ by using the variables \mathbf{S} and $\tilde{\mathbf{T}}$, i.e., it's hard to construct the constraint (6) through the state estimate $\mathbf{x}(l|k)$.

Since the posteriori estimate $\mathbf{x}(k|k)$ is known at the current step k , we can construct the constraint (6) through $\mathbf{x}(k|k)$, while satisfying the real-time and the control accuracy requirements through another way, e.g., adjusting the outputs of the actuators during the steps $[k, k+\tau-1]$. Therefore, we introduce an AOA problem. The basic idea is similar to the closed-loop control: we 1) fix \mathbf{u} to determine the decisions \mathbf{S} and $\tilde{\mathbf{T}}$, 2) estimate the error under the given decisions \mathbf{S} and $\tilde{\mathbf{T}}$, and 3) adjust the outputs of the actuators $\{\mathbf{u}(k), \dots, \mathbf{u}(k+\tau-1)\}$ based on the estimated error. The details are as follow:

- 1) At the k^{th} step, we obtain the scheduling decision \mathbf{S} and the action time decision $\tilde{\mathbf{T}}$ by solving the P2, and further, we derive the input matrix $\mathbf{B}(l)$ and the outputs of the actuators $\mathbf{u}(l)$ at the l^{th} step ($k+1 \leq l \leq k+\tau$). We define $\Delta \mathbf{u} = [\Delta \mathbf{u}(k), \dots, \Delta \mathbf{u}(k+\tau-1)]$ as the output adjustment decision during the steps $[k, k+\tau-1]$. To drive the priori estimate $\mathbf{x}(k+\tau|k)$ toward the user's requirement $[\underline{\mathbf{x}}, \bar{\mathbf{x}}]$, we let the output adjustment

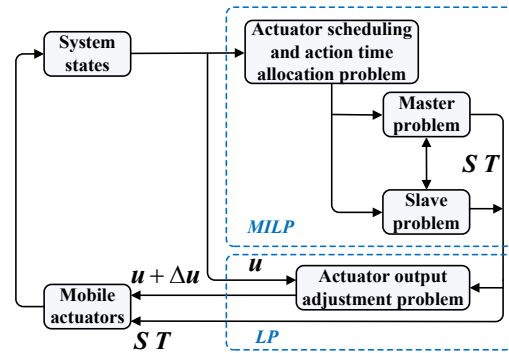


Fig. 3. Joint multi-actuator scheduling and control algorithm.

variables $\Delta \mathbf{u}$ to satisfy the following constraint:

$$\underline{\mathbf{x}} \preceq \mathbf{x}(k+\tau|k) = \mathbf{A}^\tau \mathbf{x}(k|k) + \sum_{l=k}^{k+\tau-1} \mathbf{A}^{k+\tau-1-l} \mathbf{B}(l) (\mathbf{u}(l) + \Delta \mathbf{u}(l)) \preceq \bar{\mathbf{x}}. \quad (9)$$

- 2) Note that 1) the output adjustment variables $\Delta \mathbf{u}(l)$ are bounded by $\underline{\mathbf{u}} \preceq \mathbf{u}(l) + \Delta \mathbf{u}(l) \preceq \bar{\mathbf{u}}$ ($k+1 \leq l \leq k+\tau$), and 2) the output amplitude vector \mathbf{u} is fixed and given. The AOA problem is formulated as

$$\begin{aligned} \mathbf{P3} : & \min_{\Delta \mathbf{u}} \sum_{l=k}^{k+\tau-1} \sum_{j=1}^{n_a} |\Delta u_j(l)| \\ \text{s.t.} & \begin{cases} (9), \\ \underline{\mathbf{u}} \preceq \mathbf{u}(l) + \Delta \mathbf{u}(l) \preceq \bar{\mathbf{u}}, k \leq l \leq k+\tau-1. \end{cases} \end{aligned} \quad (10)$$

Equation (9) shows that the real-time and the control accuracy constraints are actually determined by the P3. The constraints (5) and (6) can be removed from the P2. However, the P2 with these constraints can provide more accurate scheduling and action time decisions, i.e., the solution of the P3 is more easy to find out (the constraints of the P3 are more easy to satisfy) if the constraints (5) and (6) are included in the P2. Therefore, the P1 can be replaced by the P2 due to the introduction of the P3.

IV. ALGORITHM DESIGN AND ANALYSIS

In this section, we propose a joint actuator scheduling and control algorithm to solve the P2 and the P3. As shown in Fig. 3, the proposed algorithm contains two steps. In the first step, based on Benders decomposition [20], we divide the P2 into two subproblems, and then find the global optimal solution by iterating the subproblems. In the second step, based on the solution of the first step, we solve the P3 in an on-line manner and adjust the outputs of the actuators accordingly.

A. Decomposition-based ASATA Algorithm

As we mentioned before, finding an optimal scheduling decision \mathbf{S} is the most important step to solve the P2. Based on this idea, we divide the P2 into two subproblems with less variables and constraints: a *Master Problem* (MP) and a *Slave Problem* (SP). Instead of considering all the variables and the constraints simultaneously, the P2 is solved by iterating the MP and the SP. By doing so, the computing time can be further reduced. The MP accounts for all the binary variables and the associated portion of the objective function and the constraints of the P2. It also includes the information regarding the SP via a set of constraints called *Benders cuts*. The SP includes all the continuous variables and the associated constraints of the

P2. Solving the SP provides some information regarding the SP portion of the P2, and this information is included in the MP through Benders cut.

For simplicity and generality, we remove the step index k from the equations. Based on the structure of the P2, the MP and the SP are formulated as

$$\begin{aligned} \text{MP} : \Phi_L(l) &= \min_{\mathbf{S}, \hat{\Phi}} \hat{\Phi} \\ \text{s.t.} \quad &\begin{cases} (3), \\ C_1 : \hat{\Phi} \geq \Gamma(\mathbf{S}, \boldsymbol{\alpha}(\varsigma), \boldsymbol{\beta}(\varsigma), \boldsymbol{\gamma}(\varsigma), \boldsymbol{\psi}(\varsigma)), \forall \varsigma \in \mathcal{A}, \\ C_2 : 0 \geq \Lambda(\mathbf{S}, \hat{\boldsymbol{\alpha}}(\vartheta), \hat{\boldsymbol{\beta}}(\vartheta), \hat{\boldsymbol{\gamma}}(\vartheta), \hat{\boldsymbol{\psi}}(\vartheta)), \forall \vartheta \in \mathcal{B}. \end{cases} \end{aligned} \quad (11)$$

where

$$\begin{aligned} &\Gamma(\mathbf{S}, \boldsymbol{\alpha}(\varsigma), \boldsymbol{\beta}(\varsigma), \boldsymbol{\gamma}(\varsigma), \boldsymbol{\psi}(\varsigma)) \\ &= \sum_{i=1}^{n_l} \sum_{j=1}^{n_a} [k_d s_{ij} d_{ij} - t_j^m s_{ij} \alpha_{ij}(\varsigma) + (s_{ij} d_{ij}^s - \tau) \beta_{ij}(\varsigma)] \\ &\quad + \sum_{m=1}^{n_p} [(x_m - \bar{x}_m) \psi_m(\varsigma) - (x_m - \underline{x}_m) \gamma_m(\varsigma)], \\ &\Lambda(\mathbf{S}, \hat{\boldsymbol{\alpha}}(\vartheta), \hat{\boldsymbol{\beta}}(\vartheta), \hat{\boldsymbol{\gamma}}(\vartheta), \hat{\boldsymbol{\psi}}(\vartheta)) \\ &= \Gamma(\mathbf{S}, \hat{\boldsymbol{\alpha}}(\vartheta), \hat{\boldsymbol{\beta}}(\vartheta), \hat{\boldsymbol{\gamma}}(\vartheta), \hat{\boldsymbol{\psi}}(\vartheta)) - \sum_{i=1}^{n_l} \sum_{j=1}^{n_a} k_d s_{ij} d_{ij}. \end{aligned}$$

C_1 and C_2 are the sets of the Feasibility Constraints (FCs) and the infeasibility constraints (ICs), respectively. \mathcal{A} and \mathcal{B} are the sets of iterations that the Dual Slave Problem (DSP) (13) has the bounded and the unbounded solutions, respectively. $(\boldsymbol{\alpha}(\varsigma), \boldsymbol{\beta}(\varsigma), \boldsymbol{\gamma}(\varsigma), \boldsymbol{\psi}(\varsigma))$ is the solution of the DSP at the ς^{th} iteration. $(\hat{\boldsymbol{\alpha}}(\vartheta), \hat{\boldsymbol{\beta}}(\vartheta), \hat{\boldsymbol{\gamma}}(\vartheta), \hat{\boldsymbol{\psi}}(\vartheta))$ is the solution of the Dual Feasibility Check Problem (DFCP) (20) at the ϑ^{th} iteration. Note that the objective function of P2 contains the binary variables \mathbf{S} as well as the continuous variables $\hat{\mathbf{T}}$, while MP only considers the binary variables \mathbf{S} . We introduce an auxiliary variable $\hat{\Phi}$ into the MP as the objective function, where $\hat{\Phi}$ has the same physical meaning as Φ .

$$\begin{aligned} \text{SP} : \Phi_U(l) &= \min_{\hat{\mathbf{T}} \geq 0} \Phi(\mathbf{S}(l), \hat{\mathbf{T}}) \\ \text{s.t.} \quad &\begin{cases} \hat{t}_{ij} \leq t_j^m s_{ij}(l), \forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \\ s_{ij}(l) d_{ij}^s + \hat{t}_{ij} \leq \tau \Delta_s, \forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \\ \underline{x}_m \leq x_m + \sum_{i=1}^{n_l} \sum_{j=1}^{n_a} u_j \hat{t}_{ij} \tilde{b}_{mi} \leq \bar{x}_m, \forall m \in \mathcal{M}, \end{cases} \end{aligned} \quad (12)$$

where $(\mathbf{S}(l), \hat{\Phi}(l))$ is the solution of the MP at the l^{th} iteration. Comparing the SP with the P2, we observe that their formulations are the same with the exception that the binary variables \mathbf{S} in the SP are fixed.

For the MP, since the constraints regarding the action time variables $\hat{\mathbf{T}}$ are relaxed, solving this problem yields a lower bound $\Phi_L(l)$. On the other hand, since the scheduling decision $\mathbf{S}(l)$ may be just a feasible solution (not optimal yet), solving the SP yields an upper bound $\Phi_U(l)$ (the proofs are provided in Appendix A). Denote $(\mathbf{S}^*, \hat{\mathbf{T}}^*)$ as the optimal solution of the P2, and $\Phi^* = \Phi(\mathbf{S}^*, \hat{\mathbf{T}}^*)$. Hence, we have $\Phi_L(l) \leq \Phi^* \leq \Phi_U(l)$. At each iteration l , a new FC or IC (Benders cuts) is generated and added into the MP to reduce the gap between the bounds $\Phi_L(l)$ and $\Phi_U(l)$ (the proof is provided in Appendix B). The iteration process is summarized as follows.

1) *Step 1: Initialization:* Initialize the iteration counter $l = 0$, the solution $\mathbf{S}(0)$ of the MP, the lower bound $\Phi_L(0) = -\infty$, and the upper bound $\Phi_U(0) = +\infty$. The sets of feasibility and infeasibility constraints, i.e., C_1 and C_2 , are set to null. The

initial solution $\mathbf{S}(0)$ is given arbitrarily, as long as it satisfies the constraint (3).

2) *Step 2: Solving Slave Problem:* In this paper, rather than solving the SP directly, we solve its dual problem. This is because the SP is an LP problem, the optimal objective function values of the SP and its dual problem are equivalent due to the strong duality [21]. In addition, the feasibility constraint (14) and the infeasibility constraint (15) can be constructed through the solution of the DSP. To develop the dual of the SP, the positive Lagrange multipliers $\boldsymbol{\alpha} = [\alpha_{ij}]$, $\boldsymbol{\beta} = [\beta_{ij}]$, $\boldsymbol{\gamma} = [\gamma_m]$ and $\boldsymbol{\psi} = [\psi_m]$ ($\forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \forall m \in \mathcal{M}$) are introduced to the SP. Hence, the Lagrangian is

$$\begin{aligned} \mathcal{L}_1(\mathbf{S}(l), \hat{\mathbf{T}}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\psi}) &= \sum_{i=1}^{n_l} \sum_{j=1}^{n_a} [k_d s_{ij}(l) d_{ij} \\ &\quad + k_u |u_j| \hat{t}_{ij} + (\hat{t}_{ij} - t_j^m s_{ij}(l)) \alpha_{ij} + (s_{ij}(l) d_{ij}^s + \hat{t}_{ij} - \tau) \beta_{ij}] \\ &\quad - \sum_{m=1}^{n_p} \left(\sum_{i=1}^{n_l} \sum_{j=1}^{n_a} u_j \hat{t}_{ij} \tilde{b}_{mi} + x_m - \underline{x}_m \right) \gamma_m \\ &\quad + \sum_{m=1}^{n_p} \left(\sum_{i=1}^{n_l} \sum_{j=1}^{n_a} u_j \hat{t}_{ij} \tilde{b}_{mi} + x_m - \bar{x}_m \right) \psi_m. \end{aligned}$$

The dual function [21] is defined as the minimum value of Lagrangian $\mathcal{L}_1(\mathbf{S}(l), \hat{\mathbf{T}}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\psi})$ with respect to the variables \hat{t}_{ij} , i.e.,

$$\begin{aligned} \mathcal{D}(\mathbf{S}(l), \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\psi}) &= \min_{\hat{\mathbf{T}} \geq 0} \left\{ \sum_{i=1}^{n_l} \sum_{j=1}^{n_a} [k_u |u_j| + \alpha_{ij} + \right. \\ &\quad \left. \beta_{ij} + \sum_{m=1}^{n_p} u_j \tilde{b}_{mi} (\psi_m - \gamma_m)] \hat{t}_{ij} + \Gamma(\mathbf{S}(l), \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\psi}) \right\}. \end{aligned}$$

Since the variables \hat{t}_{ij} are positive, they are finite only when $k_u |u_j| + \alpha_{ij} + \beta_{ij} + \sum_{m=1}^{n_p} u_j \tilde{b}_{mi} (\psi_m - \gamma_m) \geq 0$ ($\forall i \in \mathcal{I}, \forall j \in \mathcal{J}$). Therefore, the DSP is formulated as

$$\begin{aligned} \text{DSP} : \max_{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\psi} \geq 0} \Gamma(\mathbf{S}(l), \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\psi}) \\ \text{s.t.} \quad k_d |u_j| + \alpha_{ij} + \beta_{ij} + \sum_{m=1}^{n_p} u_j \tilde{b}_{mi} (\psi_m - \gamma_m) \geq 0, \\ \forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \end{aligned} \quad (13)$$

3) *Step 3: Solving Master Problem:* Based on the solution of the DSP, two different types of new constraints are generated and added into the MP at the next iteration.

- 1) If the DSP is *infeasible*, the SP has an unbounded solution. Hence, the P2 has no feasible solution.
- 2) If the DSP has a *bounded solution* $(\boldsymbol{\alpha}(l), \boldsymbol{\beta}(l), \boldsymbol{\gamma}(l), \boldsymbol{\psi}(l))$, $\mathcal{A} \leftarrow \{l\} \cup \mathcal{A}$. The upper bound is updated by $\Phi_U(l) = \min\{\Phi_U(l-1), \Gamma(\mathbf{S}(l), \boldsymbol{\alpha}(l), \boldsymbol{\beta}(l), \boldsymbol{\gamma}(l), \boldsymbol{\psi}(l))\}$. Due to the strong duality, the SP is feasible. Denote $\hat{\mathbf{T}}(l)$ as the solution of the SP at the l^{th} iteration. Note that $\hat{\Phi}(l) < \Phi(\mathbf{S}(l), \hat{\mathbf{T}}(l)) = \Gamma(\mathbf{S}(l), \boldsymbol{\alpha}(l), \boldsymbol{\beta}(l), \boldsymbol{\gamma}(l), \boldsymbol{\psi}(l))$. To avoid selecting the non-optimal solution $\mathbf{S}(l)$ again, a new FC:

$$\hat{\Phi} \geq \Gamma(\mathbf{S}, \boldsymbol{\alpha}(l), \boldsymbol{\beta}(l), \boldsymbol{\gamma}(l), \boldsymbol{\psi}(l)), \quad (14)$$

- is generated and added into C_1 at the $(l+1)^{th}$ iteration.
- 3) If the DSP has an *unbounded solution*, i.e., $\Gamma(\mathbf{S}(l), \boldsymbol{\alpha}(l), \boldsymbol{\beta}(l), \boldsymbol{\gamma}(l), \boldsymbol{\psi}(l)) = +\infty$, due to the strong duality, the SP has no feasible solution under the given $\mathbf{S}(l)$, and $\mathcal{B} \leftarrow \{l\} \cup \mathcal{B}$. To exclude the infeasible solution $\mathbf{S}(l)$, we construct a Feasibility Check Problem (FCP) (19) and solve its dual problem. Based on the solution of the DFPC (20), i.e., $(\hat{\boldsymbol{\alpha}}(l), \hat{\boldsymbol{\beta}}(l), \hat{\boldsymbol{\gamma}}(l), \hat{\boldsymbol{\psi}}(l))$, a new IC:

$$0 \geq \Lambda(\mathbf{S}, \hat{\boldsymbol{\alpha}}(l), \hat{\boldsymbol{\beta}}(l), \hat{\boldsymbol{\gamma}}(l), \hat{\boldsymbol{\psi}}(l)), \quad (15)$$

is generated and added into C_2 at the $(l+1)^{th}$ iteration.

When the MP is solved, the iteration counter l increases, and step 2 to step 3 is repeated. The iteration stops when $|\Phi_U(l) - \Phi_L(l)| \leq \varepsilon$, where ε is a small positive value. Note that the SP and the P2 have the same objective functions, as well as the fact that the SP and the DSP are equivalent due to the strong duality. From inequation (14), we can see that the auxiliary variable $\hat{\Phi}$ has the same physical meaning as Φ .

Lemma 4.1: The lower bound $\Phi_L(l)$ (upper bound $\Phi_U(l)$) on the optimal objective function value Φ^* can be derived from the solution of the MP (SP) at the l^{th} iteration.

Proof: See Appendix A for the proof. ■

Theorem 4.1: At each iteration with FC (14) or IC (15) added into the MP, the solution converges to the global optimal value within a finite number of iterations.

Proof: See Appendix B for the proof. ■

B. Estimation-based AOA Algorithm

Since the objective function of the P3 contains the absolute value $|\Delta u_j(l)|$, this formulation is not a standard LP. To transform the P3 into a standard LP problem, we introduce two new variables $\Delta \tilde{u}_j(l)$ and $\Delta \bar{u}_j(l)$ to replace the original variable $\Delta u_j(l)$, where $\Delta \tilde{u}_j(l) = \frac{|\Delta u_j(l)| + \Delta u_j(l)}{2}$ and $\Delta \bar{u}_j(l) = \frac{|\Delta u_j(l)| - \Delta u_j(l)}{2}$. With the new output adjustment variables $\Delta \tilde{\mathbf{u}} = [\Delta \tilde{\mathbf{u}}(k), \dots, \Delta \tilde{\mathbf{u}}(k + \tau - 1)]$ and $\Delta \bar{\mathbf{u}} = [\Delta \bar{\mathbf{u}}(k), \dots, \Delta \bar{\mathbf{u}}(k + \tau - 1)]$, the P3 is rewritten as

$$\begin{aligned} \text{P4: } \min_{\Delta \tilde{\mathbf{u}}, \Delta \bar{\mathbf{u}}} & \sum_{l=k}^{k+\tau-1} \sum_{j=1}^{n_a} (\Delta \tilde{u}_j(l) + \Delta \bar{u}_j(l)) \quad (16) \\ \text{s.t. } & \begin{cases} \mathbf{x} \preceq \mathbf{x}(k + \tau|k) = \mathbf{A}^\tau \mathbf{x}(k|k) + \\ \sum_{l=k}^{k+\tau-1} \mathbf{A}^{k+\tau-1-l} \mathbf{B}(l)(\mathbf{u}(l) + \Delta \tilde{\mathbf{u}}(l) - \Delta \bar{\mathbf{u}}(l)) \preceq \bar{\mathbf{x}}, \\ \mathbf{u} \preceq \mathbf{u}(l) + \Delta \tilde{\mathbf{u}}(l) - \Delta \bar{\mathbf{u}}(l) \preceq \bar{\mathbf{u}}, \quad k \leq l \leq k + \tau - 1. \end{cases} \end{aligned}$$

Following the solution of the P4 (i.e., $\Delta \mathbf{u} = \Delta \tilde{\mathbf{u}} + \Delta \bar{\mathbf{u}}$) to adjust the actuators' outputs during the steps $[k, k + \tau - 1]$, we have $\mathbf{x} \preceq \mathbf{x}(k + \tau|k) \preceq \bar{\mathbf{x}}$. However, $\mathbf{x} \preceq \mathbf{x}(k + \tau|k) \preceq \bar{\mathbf{x}}$ cannot guarantee that $\mathbf{x} \preceq \mathbf{x}(k + \tau|k + \tau - 1) \preceq \bar{\mathbf{x}}$. The state estimate $\mathbf{x}(k + \tau|k + \tau - 1)$ may be out of the range $[\underline{\mathbf{x}}, \bar{\mathbf{x}}]$ due to the system noises $\omega(k)$. To achieve $\mathbf{x} \preceq \mathbf{x}(k + \tau|k + \tau - 1) \preceq \bar{\mathbf{x}}$, the only way is based on $\mathbf{x}(k + \tau - 1|k + \tau - 1) \preceq \bar{\mathbf{x}}$, the only way is based on $\mathbf{x}(k + \tau - 1|k + \tau - 1)$ to adjust $\mathbf{u}(k + \tau - 1)$. Therefore, we propose an on-line output adjustment method. The details are as follow:

- 1) At the l^{th} step, we construct the P4 based on the output adjustment variables $\Delta \tilde{\mathbf{u}} = [\Delta \tilde{\mathbf{u}}(l), \dots, \Delta \tilde{\mathbf{u}}(k + \tau - 1)]$ and $\Delta \bar{\mathbf{u}} = [\Delta \bar{\mathbf{u}}(l), \dots, \Delta \bar{\mathbf{u}}(k + \tau - 1)]$, and the current step state estimate $\mathbf{x}(l|l)$, where the control accuracy constraint is given by $\mathbf{x} \preceq \mathbf{x}(k + \tau|l) \preceq \bar{\mathbf{x}}$.
- 2) We solve the P4 through LP and follow its solution, i.e., $\mathbf{u}(l) + \Delta \mathbf{u}(l) = \mathbf{u}(l) + \Delta \tilde{\mathbf{u}}(l) + \Delta \bar{\mathbf{u}}(l)$, to adjust the outputs of the actuators at the l^{th} step.

The above two steps are repeated until $l = k + \tau - 1$. For the on-line output adjustment method, we need to solve P4 $\tau - 1$ times during the steps $[k, k + \tau - 1]$. However, with step number increasing, the dimension of variables $\Delta \mathbf{u}$ reduces.

V. PERFORMANCE EVALUATION

A. Simulation Setup

We consider a Wireless Rechargeable Sensor Network (WRSN) as the simulation and experimental case study. Eight

sensors and eight mobile actuators are randomly deployed in a 20 m \times 20 m ROI to perform the environment sensing and the sensor energy charging tasks ($n_s = n_a = 8$). The mobile robots and the sensors (e.g., Mica2) that equipped with Powercast chargers and receivers (see Fig. 4) are served as the mobile chargers (actuators) and the rechargeable sensors, respectively. Therefore, the system states \mathbf{x} represent the residual energy of the sensors ($n_p = 8$). The system parameters are summarized in Table III. The sensor model is adopted from [6], where r_i is the energy consumption rate of the sensor \mathbb{S}_i . f_{ij} (f_{ib}) is the flow rate from \mathbb{S}_i to \mathbb{S}_j (from \mathbb{S}_i to BS), ρ and η_{ij} (or η_{ib}) are the rate of energy consumption for receiving a unit of data rate, and transmitting a unit of data rate from \mathbb{S}_i to \mathbb{S}_j (or BS), respectively. λ_1 and λ_2 are the distance-independent and the distance-dependent constant terms, respectively. d_{ij} is the distance between \mathbb{S}_i and \mathbb{S}_j . θ is the path loss index. Here, we consider r_i is invariant with time. For a regular AA battery, its nominal cell voltage and the quantity of electricity is 1.2 V/2.5 Ah. Since two AA batteries provide an average voltage 2.4 V for the Mica2 and the operating limit is 2.1 V [22], we have $\underline{x}_{th,i} = 2.1 \times 2.5 \times 3600 = 18900$ J and $\bar{x}_{th,i} = 2.4 \times 2.5 \times 3600 = 21600$ J [6].

Sensor \mathbb{S}_i characteristics				
$r_i = \rho \sum_{k=1, k \neq i}^{n_s} f_{ki} + \sum_{j=1, j \neq i}^{n_s} \eta_{ij} f_{ij} + \eta_{ib} f_{ib}$				
$\eta_{ij} = \lambda_1 + \lambda_2 (d_{ij})^\theta$		$\rho = 50$ nJ/b	$f_{ij} \in [1, 10]$ kb/s	
$\lambda_1 = 50$ nJ/b		$\lambda_2 = 0.0013$ pJ/(b.m ⁴)		$\theta = 4$
Actuator \mathbb{A}_j characteristics				
$b_{ij} = 6$ %	$u_j = 3$ W	$\underline{u}_j = 0$ W	$\bar{u}_j = 5$ W	$v = 0.5$ m/s
System state characteristics				
$n_a = 8$	$n_s = 8$	$\Delta_s = 1$ s	$\tau = 1000$ s	
$\underline{x}_{th,i} = 18900$ J		$\bar{x}_i = 21580$ J		$n_p = 8$
$\bar{x}_{th,i} = 21600$ J		$\bar{x}_i = 21600$ J		$n_l = 8$

TABLE III. SYSTEM PARAMETERS

As the experiments shown in [23], when a charger is placed 10 cm away from the receiver, the charging efficiency reduces to 1.5 %. To enhance the charging efficiency, we consider each sensor has only one working point, which is very close to this sensor ($n_l = 8$). Hence, the charging efficiency increases to 6 % [23]. Since $n_p = n_a$, all the sensors can be charged in one round. If $n_p > n_a$, based on the lifetime of sensors, we can divide the charging process into several rounds, and then in each round, we have $n_p \leq n_a$. Note that different sensor and actuator parameters lead to different values in the parameters for the P2 and the P4. However, the structures of the problems under different values of parameters are the same, and, thus, the proposed methods are still applicative. The simulations are performed on a PC with dual-core 2.6 GHz Intel i5 processor and 12 GB RAM, and the algorithms are implemented in Matlab 2013a.

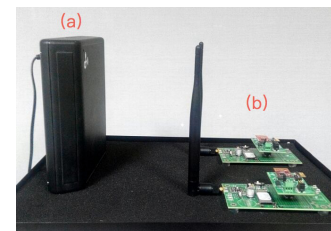


Fig. 4. (a) Powercast charger; (b) Powercast receivers.

B. System Performance

The initial system states $\mathbf{x}(1)$ are assumed to be in the range [18700, 18900] J. Hence, an event occur at step $k = 1$. We use the root mean square error between lower bound $\underline{\mathbf{x}}$ and posteriori estimate $\mathbf{x}(k|k)$, i.e., $J(k) = \sqrt{\frac{(\underline{\mathbf{x}} - \mathbf{x}(k|k))'(\underline{\mathbf{x}} - \mathbf{x}(k|k))}{n_p}}$, to evaluate the control error. Following the solutions of the P2 and the P4, i.e., the matrices \mathbf{S} , $\tilde{\mathbf{T}}$ and $\Delta\mathbf{U}$, to schedule the actuators and adjust their outputs, the dynamic change of control error $J(k)$ is shown in Fig. 5(a). At the beginning, the actuators take several steps to relocate. When the actuators arrive at the designed working points and start charging the sensors, the control error $J(k)$ converges to δ gradually, where δ is a small positive value. After the energy charging process is completed, the control error $J(k)$ will gradually increase. When a new event occurs, the actuator scheduling and control decisions should be updated again.

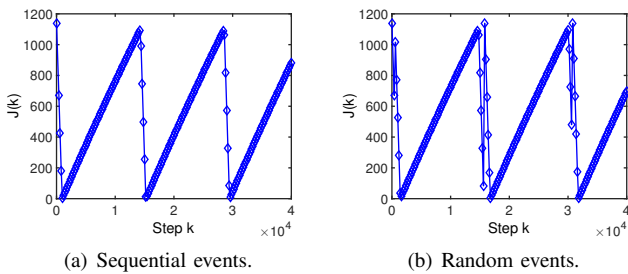
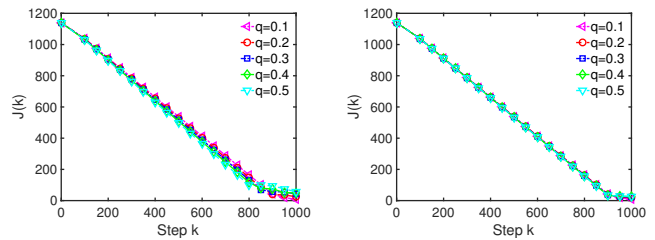


Fig. 5. System performance to deal with sequential and random events.

Fig. 5(b) evaluates the system performance (control error) to deal with random events, where the sensors $\{\mathcal{S}_1, \mathcal{S}_3\}$, $\{\mathcal{S}_4, \mathcal{S}_8\}$, $\{\mathcal{S}_2, \mathcal{S}_7\}$ and $\{\mathcal{S}_5, \mathcal{S}_6\}$ require charging at steps $k = 1, 500, 15800, 30800$, respectively. From Fig. 5(b), we observe that the new events occur when the actuators are still handling the previous events. In this case, the scheduling decision \mathbf{S} and the action time decision $\tilde{\mathbf{T}}$ should be updated again, even if the previous tasks haven't finished yet. Note that there is no need to update the decisions \mathbf{S} and $\tilde{\mathbf{T}}$ at each step k . This is because solving an MILP problem at each step k is time consuming and changing scheduling decision \mathbf{S} at each step k will cause the actuators to move in a zigzag way. Since the decision update process is event-driven rather than time-driven, we can handle random event efficiently, while avoiding high computational complexity.

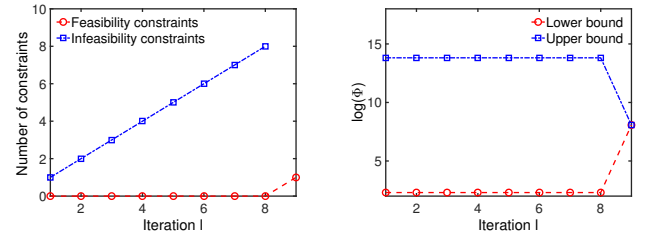
Fig. 6 compares the changes of control error $J(k)$ with the on-line and the off-line output adjustment methods. The off-line method solves the P4 only once at step $k = 1$. We set $\mathbf{Q} = q \cdot \mathbf{I}_{n_p \times n_p}$ and change the value of q from 0.1 to 0.5 with a step of 0.1. With the off-line method, since $\mathbf{x}(k + \tau|k)$ is a multi-step prediction, the control error $J(k + \tau)$ increases with the value of q . In contrast, with the on-line method, the influence of q exerts on the control error $J(k + \tau)$ is limited. This is because the output adjustment matrix $\Delta\mathbf{U}$ needs to be updated at each step k and the update process is based on a posteriori estimate $\mathbf{x}(k|k)$ of current step. Although the on-line method needs to solve the P4 multiple times, the P4 can be solved very fast using polynomial-time algorithms as it is an LP problem. Using the on-line method, we can get a better robustness to against the system noises $\omega(k)$.

Fig. 7 shows the convergence of the decomposition-based ASATA algorithm. With the IC (14) and the FC (15) added into the MP during the iterations $l = 1 \sim 8$, and $l = 9$, respectively, the upper bound $\Phi_U(l)$ and the lower bound $\Phi_L(l)$ quickly



(a) With off-line adjustment method. (b) With on-line adjustment method.
Fig. 6. Control errors with on-line and off-line output adjustment methods.

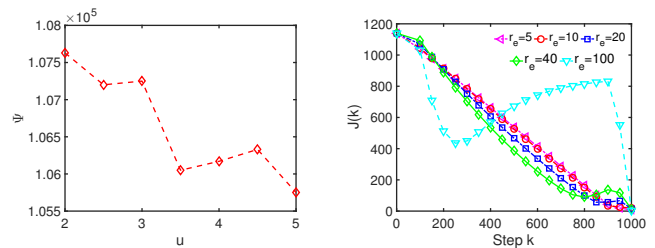
converge to the optimal value Φ^* .



(a) The number of feasibility and infeasibility constraints. (b) The gap between the lower and the upper bounds.
Fig. 7. Convergence of decomposition-based ASATA algorithm.

C. Scalability and Robustness Evaluation

Denote Ψ as the total energy consumption of the actuators during the steps [1, 40000]. Fig. 8(a) evaluates the values of Ψ under different actuator output amplitudes \mathbf{u} . We set $n_a = n_p = 8$ and $u_i = u_j$ ($i \neq j, 1 \leq i, j \leq n_a$) and change the value of u_j ($1 \leq j \leq n_a$) from 2 to 5 with a step of 0.5. Fig. 8(a) shows that the differences are small. This is because when the output amplitude \mathbf{u} increases, the actuator action time $\tilde{\mathbf{T}}$ will reduce, while Φ represents the total energy consumption. In addition, the elements of output adjustment matrix $\Delta\mathbf{U}$ are usually very small, since the aim of the P4 is to minimize them. However, if the value of u_j ($1 \leq j \leq n_a$) is too small, e.g., $u_j = 1$, the P2 or the P4 may be infeasible.



(a) Actuator energy consumption with \mathbf{u} varying. (b) Control error with r_e varying.
Fig. 8. System performance with system parameter varying.

Denote $r_e = \frac{\sum_{i=1}^{n_p} r_i}{n_p}$ as the average energy consumption rate of the sensors. Fig. 8(b) shows the values of control error $J(k)$ under different r_e . We set $n_a = n_p = 8$ and $u_j = 3$ ($1 \leq j \leq n_a$) and change the value of r_e between [5, 100]. Note that 1) r_e only influences the system matrix \mathbf{A} , and 2) we assume that $\mathbf{A} = \mathbf{I}$ in the P2. The solution of the P2, i.e., the matrices \mathbf{S} and $\tilde{\mathbf{T}}$, will not change with r_e . To compensate the error introduced by the constraint (6), based on the real system matrix \mathbf{A} ($\mathbf{A} \neq \mathbf{I}$), we construct the P4 through the estimate of system states $\mathbf{x}(k + \tau)$. Hence, r_e will influence

the solution of the P4, i.e., the output adjustment matrix ΔU . Fig. 8(b) shows that the larger r_e is, the larger error overshoot is. This is because before the actuator relocation is completed, a large r_e leads to a fast change of the residual energy $x(k)$. Fig. 8(b) also shows that at the beginning, under the different average energy consumption rates, the control error $J(k)$ could be either increase or decrease. However, when the actuator relocation is completed, the control error $J(k)$ converges to δ within the action delay τ .

We define the ASATA convergence iteration as the number of iterations to achieve $|\Phi_U(l) - \Phi_L(l)| \leq \varepsilon$. Table IV shows that the ASATA convergence iteration almost linearly increases with the value of n_p , where we set $n_a = n_p$ and $u_j = 3$ ($1 \leq j \leq n_a$) and change the value of n_p from 5 to 50 with a step of 5. The parameters in the simulations with $n_p = i$ ($i = 10, \dots, 50$) and $n_p = i - 5$ are correlated, i.e., the sensors and the actuators in the simulation with $n_p = i$ are extended based on the sensors and the actuators in the simulation with $n_p = i - 5$. Higher dimension of the system states usually involves more variables and constraints into the problem. Hence, more iterations are required to search for the optimal solution.

n_p	5	10	15	20	25	30	35	40	45	50
Iterations	4	9	14	19	27	30	32	35	41	49

TABLE IV. ASATA CONVERGENCE ITERATION WITH n_p VARYING.

Since the actuator scheduling and control problems are jointly addressed in the P2 and the input matrix is determined by the scheduling decision, the input matrix is unknown until the P2 is solved. This is in contrast to the traditional system stability analysis that the system model is usually given in advance. Hence, we evaluate the influence of the sampling period Δ_s through the experiments. We define the error convergence speed as the number of steps to achieve $J(k) \leq \delta$. Table V shows the changes of error convergence speed under different Δ_s . We set $n_a = n_p = 8$, $u_j = 3$ ($1 \leq j \leq n_a$) and change the value of Δ_s between $[1, 120]$ s. If the sampling period is too large, e.g., $\Delta_s = 100$ s, control error $J(k)$ is hard to converge since the P2 or the P4 is infeasible.

Δ_s (s)	1	20	40	60	80	100	120
Steps	1000	1000	1000	1000	1000	$+\infty$	$+\infty$

TABLE V. ERROR CONVERGENCE SPEED WITH Δ_s VARYING.

D. Comparison with Existing Algorithms

Fig. 9(a) compares the performance (the objective function value and computation time) of the proposed decomposition-based method with 1) optimal approaches: Branch and Bound (B&B) method [24] and Branch and Cut (B&C) method [25], which are known to provide the optimal solution for the MILP problems such as P2, and 2) evolutionary approach: Genetic Algorithm (GA) [8], [26]. We set $n_a = n_p$ and $u_j = 3$ ($1 \leq j \leq n_a$) and change the value of n_p from 5 to 50 with a step of 5. Fig. 9(a) shows that the solutions given by the proposed method, the B&B and the B&C are the same, since the proposed method is able to find the global optimal solution. In addition, the proposed method achieves lower value for the objective function than GA, since the P2 is a minimization problem and there is no guarantee of convergence to a global

optimum for GA [19]. The convergence of GA is sensitive to the choice of the genetic operators, the mutation probability and the selection criteria, while fine-tuning of these parameters is often required.

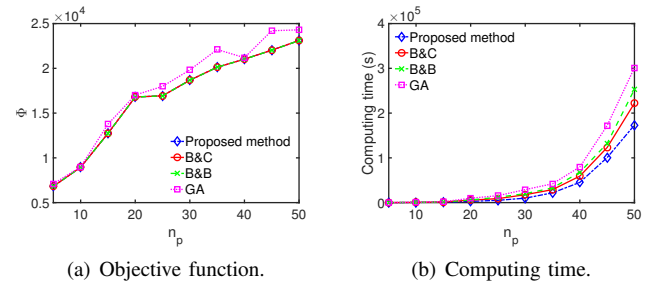


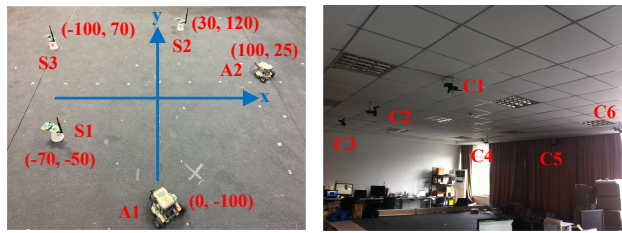
Fig. 9. Comparison between the proposed method and other methods.

Fig. 9(b) shows that when the value of n_p is increased, the computing time of all the four methods grows, since more variables and constraints are involved into the problem, and, thus, the problem size is enlarged. However, the proposed method takes a shorter computing time than the others. Compared with the proposed method, GA is more complex since it needs to generate new populations in each iteration by applying several procedures, such as selection, reproduction, mutation and crossover. B&C, which combines the benefits of B&B and Gomory cutting schemes, can better explore the trade-off between optimality, efficiency and stability. Usually, B&C has a faster convergence speed than B&B [25]. For an optimization problem, the computing time increases significantly with the number of variables and constraints. Hence, solving smaller problems (i.e., MP and SP) iteratively is more efficient than solving a single large problem. This result is in line with the comparison in [27], where the decomposition-based method is faster than B&B and B&C (computing time) when solving large problem instances.

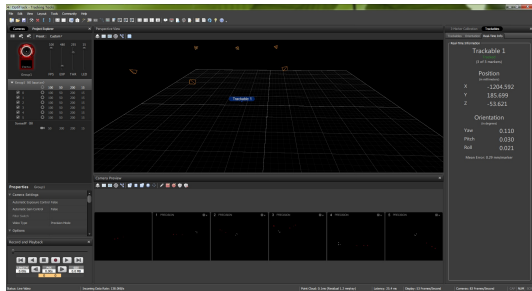
E. Experiment

In this section, we evaluate the performance of the proposed method through the experiments, where three sensors (Mica2 equipped with Powercast receiver) are randomly deployed in a $3 \text{ m} \times 3 \text{ m}$ ROI and two LEGO NXT wheeled robots are employed to serve as the actuators (mobile chargers). The initial locations of the sensors and the robots are shown in Fig. 10(a). The constrained environment (e.g., collision/obstacle avoidance) is not considered in this paper. We use the OptiTrack system [28] to detect and track the movements of the robots. This system contains six cameras (with 0.3 MP resolution at 100 FPS), which are installed on the ceiling (Fig. 10(b)). The cameras are connected to the BS (i.e., PC) via a USB hub and the realtime positions of the mobile targets are shown through a Graphical User Interface (GUI) on PC (Fig. 10(c)). The parameters of the sensors and the actuators are listed in Table III. The BS collects the realtime positions of the robots obtained from the OptiTrack, constructs the P2 and the P4 and solves the problems in Matlab to obtain the actuator scheduling and control decisions and simulates the dynamic energy change of the sensors. At the beginning of the experiment, the BS simulates the dynamic change of the system states $x(k)$ without the system inputs $u(k)$. If an event is detected, the BS updates the scheduling and the control decisions and the mobile robots are relocated accordingly. When the mobile robots arrive at the designed working points and start performing actions, the BS

calculates the dynamic change of system states $x(k)$ under the influences of system inputs $u(k)$.



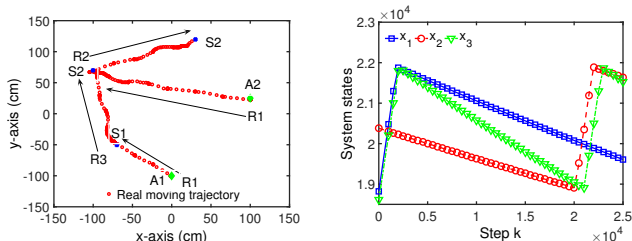
(a) LEGO NXT robots and Mica2 nodes with Powercast receivers. (b) OptiTrack tracking system.



(c) GUI.

Fig. 10. Overview of the testbed.

When the start and the goal points of each robot are determined, the BS performs the path tracking control through the quadratic curve method [29]. The aim of the path tracking control is to control the system states of a mobile robot $(p_x, p_y, p_\theta)'$ by using two parameters (φ_r, φ_l) , where (p_x, p_y) and p_θ are the position and the posture angle of a robot, respectively. φ_r and φ_l are the angular velocities of the right and the left wheels, respectively. The quadratic curve method contains two steps. In the first step, a quadratic curve that links the reference position (p_x^*, p_y^*) and the robot position (p_x, p_y) is calculated. In the second step, by using the error vector $(e_x, e_y, e_\theta)'$, the control inputs to the robot (φ_r, φ_l) are determined so that the robot can move along with the calculated quadratic curve. The control command (φ_r, φ_l) is sent from the BS to the robot via Bluetooth with a period of 0.2 s. Since this command only contains the angular velocities of the left and the right wheels, the packet size of one command for a robot is usually smaller than 100 Bytes. The control of the robot is carried-out in a closed-loop manner. Based on the received control command, a classic PID algorithm [30] is implemented in the robot to adjust the velocities of two driving wheels. Although there exists some errors between the designed and real goal points, these errors can be further reduced by adjusting the parameters of the algorithms, e.g., the period for sending commands.



(a) Real robot moving trajectory

(b) Dynamic sensor energy

Fig. 11. Experimental results.

The real robot moving trajectory (from OptiTrack) and the

simulated sensor energy (from Matlab) are shown in Fig. 11(a) and Fig 11(b), respectively. Since the velocities of the robots are not constant, the mean velocity in the P2 is set to $v = 6$ cm/s, which is an empirical value. The charging process contains three rounds: \mathcal{R}_1 , \mathcal{R}_2 and \mathcal{R}_3 . Table VI summarizes the experimental details, where I_r , L_i , L_f , T_m and T_c are the scheduled robot, the initial and final positions of robots, the moving and charging time of robots, respectively. The experimental results show that for the robots, depends on the different initial orientations, their movements are not always a straight line. In addition, the moving time of the robots is much shorter than the charging time of the sensors, and, thus, the approximation of the mean velocity in the P2 is acceptable. The experiments based on the testbed provides some demonstrative results of our proposed method for the multi-actuator coordination. Based on the different applications, e.g., fire monitoring and extinguishing [11], boundary and zone control [31], or multi-vehicle formation control [32], the LEGO robots can be replaced by the appropriate types of robots. In these applications, the temperature or the gas concentration at some POIs or the distances between the leader and the followers can be viewed as the system states. Therefore, the aim of the problem is to schedule the mobile actuators and adjust their outputs to control the system states in order to meet the system requirements, such as control accuracy, real-time performance and energy-efficiency.

	I_r	L_i	L_f	T_m (s)	T_c (s)
\mathcal{R}_1	A ₁	A ₁	S ₁	3.5	1859.8
\mathcal{R}_1	A ₂	A ₂	S ₃	6.3	2115.4
\mathcal{R}_2	A ₂	S ₃	S ₂	4.9	1792.3
\mathcal{R}_3	A ₁	S ₁	S ₃	4.8	1928.9

TABLE VI. ROBOT SCHEDULING AND CHARGING TIME.

F. Discussion

In the simulations and the experiments, we consider WRSN as an illustrative example. However, the proposed method is not limited to WRSN. Similar joint-design problems can be found in many other areas, such as the task mapping problem [33] in CPS devices. The CPS devices could be sensors, actuators or BS, depending on the different applications, and they have different computing abilities. An application (e.g., image processing or target tracking) usually consists of a set of tasks. These tasks could be executed on different single-core devices (e.g., sensors or actuators) and/or on different cores of a multi-core device (e.g., BS). For example, in the target tracking application, instead of collecting and sending all data to BS, a part of the processing is done on-site with sensors to reduce the network traffic. Therefore, only a small part of pre-processed data needs to be sent. This model of computation is known as ‘‘Fog/Edge-computing’’.

Generally, the task mapping problem includes the task allocation subproblem with the aim to decide task-to-device and/or task-to-core allocation and the task scheduling subproblem with the aim to determine the start and the end time of each task. Since different task allocation schemes lead to different task scheduling decisions, the correlated subproblems should be optimized simultaneously to find the optimal solution. Usually, the task mapping is performed under a set of real-time and energy constraints, since the tasks need to be executed before

a deadline to generate a correct result and some devices (e.g., sensors and actuators) have limited energy budgets. The energy consumption of the CPS devices is determined by the task mapping decision, which is a dynamic process. In this context, the approach presented in this paper can be easily extended to formulate the task mapping problem. Some existing works, e.g., [34], [35], have already considered the task mapping problem with MILP structure. Such kind of problem can be optimally solved by the proposed decomposition-based method.

VI. CONCLUSION

In this paper, we solved the actuator scheduling and control joint-design problem in CPS. To enhance the system real-time and energy-efficiency performances, as well as to reduce the error introduced by the actuator movement and dynamic system states, we formulated a joint-design problem that consists of an ASATA problem and an AOA problem. On this basis, we proposed a decomposition-based method and an estimation-based method to solve these problems efficiently. Finally, we analyzed the convergence and the control accuracy of the proposed methods. Simulation results demonstrated that the proposed methods are able to deal with the sequential and random events, and achieve a trade-off between the control accuracy and the computing time. The proposed method is also implemented to the real system. The experimental results showed that the desired control requirements are satisfied by scheduling the robots and controlling their actions.

ACKNOWLEDGMENT

This research was partly supported by the National Natural Science foundation of China (Grant No. 61403340 and 61573103), the ANR ARTEFACT (AppRoximaTivE Flexible Circuits and Computing for IoT) project (Grant No. ANR-15-CE25-0015), the Natural Science Foundation of Jiangsu Province of China (Grant No. BK20180012), and the Lorraine regional project (Grant AME SATELOR).

APPENDIX A PROOF OF LEMMA 4.1

Proof: Although the MP is composed of binary variables \mathbf{S} and a continuous variable $\hat{\Phi}$, this problem can be solved by only considering the binary variables \mathbf{S} . Equation (11) shows that the effective constraints are (3) and C_2 , while C_1 can be treated as the objective function. We assume that at the ς^{th} ($1 \leq \varsigma \leq l$) iteration, the DSP has a bounded solution $(\alpha(\varsigma), \beta(\varsigma), \gamma(\varsigma), \psi(\varsigma))$. Comparing the MP with the following ILP problem:

$$\begin{aligned} \hat{\Phi}_r(\varsigma) = \min_{\mathbf{S}} \quad & \Gamma(\mathbf{S}, \alpha(\varsigma), \beta(\varsigma), \gamma(\varsigma), \psi(\varsigma)) \\ \text{s.t.} \quad & \begin{cases} \sum_{i=1}^{n_i} s_{ij} \leq 1, \quad \forall j \in \mathcal{J}, \\ 0 \geq \Lambda(\mathbf{S}, \hat{\alpha}(\vartheta), \hat{\beta}(\vartheta), \hat{\gamma}(\vartheta), \hat{\psi}(\vartheta)), \quad \forall \vartheta \in \mathcal{B}, \end{cases} \end{aligned} \quad (17)$$

we have $\Phi_L(l) = \hat{\Phi}(l) = \max_{\varrho \in \mathcal{A}} \{\hat{\Phi}_r(\varrho)\}$. Without loss of generality, we assume that $\hat{\Phi}(l) = \hat{\Phi}_r(\rho)$ ($\rho \in \mathcal{A}$). Thus,

$$\begin{aligned} \hat{\Phi}(l) = \min_{\mathbf{S}} \quad & \Gamma(\mathbf{S}, \alpha(\rho), \beta(\rho), \gamma(\rho), \psi(\rho)) \\ & \leq \Gamma(\mathbf{S}^*, \alpha(\rho), \beta(\rho), \gamma(\rho), \psi(\rho)) \end{aligned} \quad (18a)$$

$$\leq \max_{\alpha, \beta, \gamma, \psi \geq 0} \Gamma(\mathbf{S}^*, \alpha, \beta, \gamma, \psi) = \Phi^*, \quad (18b)$$

where inequation (18a) holds since \mathbf{S}^* is not the optimal solution under the given multipliers $(\alpha(\rho), \beta(\rho), \gamma(\rho), \psi(\rho))$. Equation (18b) shows that $\Phi_L(l)$ is a lower bound of Φ^* . Depending on the solution of the DSP, its objective function value can be either finite or infinite. It is obvious that $+\infty$ is an upper bound of Φ^* . Thus, we focus on the case when the DSP has a bounded solution $(\alpha(l), \beta(l), \gamma(l), \psi(l))$. Since

$$\begin{aligned} \Gamma(\mathbf{S}(l), \alpha(l), \beta(l), \gamma(l), \psi(l)) &= \min_{\hat{\mathbf{T}} \geq 0} \Phi(\mathbf{S}(l), \hat{\mathbf{T}}) \\ &\geq \min_{\hat{\mathbf{T}} \geq 0} \Phi(\mathbf{S}^*, \hat{\mathbf{T}}) = \Phi^*, \end{aligned}$$

$\Phi_U(l) = \min\{\Phi_U(l-1), \Gamma(\mathbf{S}(l), \alpha(l), \beta(l), \gamma(l), \psi(l))\} = \min_{1 \leq i \leq l} \{\Gamma(\mathbf{S}(i), \alpha(i), \beta(i), \gamma(i), \psi(i))\}$ is an upper bound of Φ^* . ■

APPENDIX B PROOF OF THEOREM 4.1

Proof: At the l^{th} iteration, if the DSP has a bounded solution $(\alpha(l), \beta(l), \gamma(l), \psi(l))$, the SP is feasible. The non-optimal solution $\mathbf{S}(l)$ is excluded by the FC: $\hat{\Phi} \geq \Gamma(\mathbf{S}, \alpha(l), \beta(l), \gamma(l), \psi(l))$. On the other hand, if the DSP has an unbounded solution, the SP is infeasible. For the SP, its feasibility is related to the constraints rather than the objective function. This problem may be feasible if the positive variables $\xi^1 = [\xi_{ij}^1]$, $\xi^2 = [\xi_{ij}^2]$, $\xi^3 = [\xi_m^3]$ and $\xi^4 = [\xi_m^4]$ ($\forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \forall m \in \mathcal{M}$) are introduced to relax the constraints. Therefore, we construct a Feasibility Check Problem (FCP):

$$\begin{aligned} \text{FCP: } \min_{\hat{\mathbf{T}}, \xi \geq 0} \quad & \Theta(\xi) = \sum_{k=1}^4 \left(\sum_{i=1}^{n_i} \sum_{j=1}^{n_a} \xi_{ij}^k + \sum_{m=1}^{n_p} \xi_m^k \right) \\ \text{s.t.} \quad & \begin{cases} \hat{t}_{ij} \leq t_j^m s_{ij}(l) + \xi_{ij}^1, \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \\ \hat{t}_{ij} + s_{ij}(l) d_{ij}^s \leq \tau + \xi_{ij}^2, \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \\ \bar{x}_m - \xi_m^3 \leq x_m + \sum_{i=1}^{n_i} \sum_{j=1}^{n_a} u_j \hat{t}_{ij} \tilde{b}_{mi} \\ \leq \bar{x}_m + \xi_m^4, \quad \forall m \in \mathcal{M}, \end{cases} \end{aligned} \quad (19)$$

Introducing the positive Lagrange multipliers $\hat{\alpha} = [\hat{\alpha}_{ij}]$, $\hat{\beta} = [\hat{\beta}_{ij}]$, $\hat{\gamma} = [\hat{\gamma}_m]$ and $\hat{\psi} = [\hat{\psi}_m]$ ($\forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \forall m \in \mathcal{M}$) to the FCP, the Lagrangian is

$$\begin{aligned} \mathcal{L}_2(\mathbf{S}(l), \hat{\mathbf{T}}, \xi, \hat{\alpha}, \hat{\beta}, \hat{\gamma}, \hat{\psi}) &= \sum_{i=1}^{n_i} \sum_{j=1}^{n_a} \left\{ [\hat{\alpha}_{ij} + \hat{\beta}_{ij} + \sum_{m=1}^{n_p} u_j \tilde{b}_{mi} (\hat{\psi}_m - \hat{\gamma}_m)] \hat{t}_{ij} \right. \\ &\quad \left. + (1 - \hat{\alpha}_{ij}) \xi_{ij}^1 + (1 - \hat{\beta}_{ij}) \xi_{ij}^2 \right\} \\ &\quad + \sum_{m=1}^{n_p} [(1 - \hat{\gamma}_m) \xi_m^3 + (1 - \hat{\psi}_m) \xi_m^4] \\ &\quad + \sum_{i=1}^{n_i} \sum_{j=1}^{n_a} [-t_j^m s_{ij}(l) \hat{\alpha}_{ij} + (s_{ij}(l) d_{ij}^s - \tau) \hat{\beta}_{ij}] \\ &\quad + \sum_{m=1}^{n_p} [(x_m - \bar{x}_m) \hat{\psi}_m - (x_m - \bar{x}_m) \hat{\gamma}_m]. \end{aligned}$$

Therefore, the Dual FCP (DFCP) is formulated as

$$\begin{aligned} \text{DFCP: } \max_{\hat{\alpha}, \hat{\beta}, \hat{\gamma}, \hat{\psi} \geq 0} \quad & \Lambda(\mathbf{S}(l), \hat{\alpha}, \hat{\beta}, \hat{\gamma}, \hat{\psi}) \\ \text{s.t.} \quad & \begin{cases} \hat{\alpha}_{ij} + \hat{\beta}_{ij} + \sum_{m=1}^{n_p} u_j \tilde{b}_{mi} (\hat{\psi}_m - \hat{\gamma}_m) \geq 0, \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \\ 1 - \hat{\alpha}_{ij} \geq 0, 1 - \hat{\beta}_{ij} \geq 0, \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \\ 1 - \hat{\gamma}_m \geq 0, 1 - \hat{\psi}_m \geq 0, \quad \forall m \in \mathcal{M}, \end{cases} \end{aligned} \quad (20)$$

Denote $\xi(l)$ and $(\hat{\alpha}(l), \hat{\beta}(l), \hat{\gamma}(l), \hat{\psi}(l))$ as the solutions of the FCP and the DFCP at the l^{th} iteration, respectively.

If the SP exists the infeasible constraints, the related relax variables are non-zero, while the others are zero. Hence, we have $\Theta(\xi(l)) > 0$. Since the FCP is a LP, the strong duality is guaranteed between the FCP and its dual problem, i.e., $\Theta(\xi(l)) = \Lambda(\mathcal{S}(l), \hat{\alpha}(l), \hat{\beta}(l), \hat{\gamma}(l), \hat{\psi}(l)) > 0$. Therefore, the infeasible solution $\mathcal{S}(l)$ is excluded by the IC: $0 \geq \Lambda(\mathcal{S}, \hat{\alpha}(l), \hat{\beta}(l), \hat{\gamma}(l), \hat{\psi}(l))$. The reasons why we construct the IC by the solution of the DFCP rather than the solution of the FCP is that $\Lambda(\mathcal{S}, \hat{\alpha}, \hat{\beta}, \hat{\gamma}, \hat{\psi})$ is a function with respect to the binary variables \mathcal{S} but not $\Theta(\xi)$, i.e., $0 \geq \Theta(\xi(l))$ is an invalid constraint for the MP.

Since the non-optimal values of the binary variables \mathcal{S} found by previous l iterations have been excluded, and as l increases more constraints are added into the MP (i.e., the feasible region of the MP will shrink), $\Phi_L(l+1)$ is larger than the previous lower bounds $\{\Phi_L(0), \dots, \Phi_L(l)\}$. On the other hand, since the upper bound is updated by $\Phi_U(l) = \min\{\Phi_U(l-1), \Gamma(\mathcal{S}(l), \alpha(l), \beta(l), \gamma(l), \psi(l))\}$, $\Phi_U(l+1)$ is smaller than the previous upper bounds $\{\Phi_U(0), \dots, \Phi_U(l)\}$. According to the characteristics of these two sequences, the global optimal solution of the P2 is guaranteed when $|\Phi_U(l) - \Phi_L(l)| \leq \varepsilon$. Note that at each iteration a new constraint is added into the MP to exclude those non-optimal or infeasible values of the binary variables \mathcal{S} . In addition, the dimension of the binary variables \mathcal{S} is finite. The gap between the lower and the upper bounds converges to ε in a finite number of iterations. ■

REFERENCES

- [1] Y. Liu, Y. Peng, B. Wang, S. Yao, and Z. Liu, "Review on cyber-physical systems," *IEEE/CAA J. of Autom. Sinica*, vol. 4, no. 1, pp. 27–40, 2017.
- [2] L. Yeh, C. Lu, C. Kou, Y. Tseng, and C. Yi, "Autonomous light control by wireless sensor and actuator networks," *IEEE Sensors J.*, vol. 10, no. 6, pp. 1029–1041, 2010.
- [3] A. E. D. Mady and G. Provan, "Co-design of wireless sensor-actuator networks for building controls," in *Proc. of IEEE CDC-ECC*, 2011, pp. 5266–5273.
- [4] H. Li, L. Lai, and H. V. Poor, "Multicast routing for decentralized control of cyber physical systems with an application in smart grid," *IEEE J. Sel. Areas Commun.*, vol. 30, no. 6, pp. 1097–1107, 2012.
- [5] C. Wang, J. Li, F. Ye, and Y. Yang, "NETWRAP: an NDN based real-time wireless recharging framework for wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 13, no. 6, pp. 1283–1297, 2014.
- [6] L. Xie, Y. Shi, T. Hou, W. Lou, H. D. Sherali, and S. F. Midkiff, "Multi-node wireless energy charging in sensor networks," *IEEE/ACM Trans. Netw.*, vol. 23, no. 2, pp. 437–450, 2015.
- [7] X. Ye and W. Liang, "Charging utility maximization in wireless rechargeable sensor networks," *Wirel. Netw.*, pp. 1–13, 2016.
- [8] J. Jia, J. Chen, Y. Deng, X. Wang, and A.-H. Aghvami, "Joint power charging and routing in wireless rechargeable sensor networks," *Sensors*, vol. 17, no. 10, pp. 1–17, 2017.
- [9] J. Chen, X. Cao, P. Cheng, Y. Xiao, and Y. Sun, "Distributed collaborative control for industrial automation with wireless sensor and actuator networks," *IEEE Trans. Ind. Electron.*, vol. 57, no. 12, pp. 4219–4230, 2010.
- [10] X. Cao, J. Chen, Y. Xiao, and Y. Sun, "Building-environment control with wireless sensor and actuator networks: centralized versus distributed," *IEEE Trans. Ind. Electron.*, vol. 57, no. 11, pp. 3596–3605, 2010.
- [11] K. Ota, M. Dong, Z. Cheng, J. Wang, X. Li, and X. S. Shen, "ORACLE: mobility control in wireless sensor and actor networks," *Comput. Commun.*, vol. 35, no. 9, pp. 1029–1037, 2012.
- [12] L. Mo, X. Cao, Y. Song, and A. Kritikakou, "Distributed node coordination for real-time energy-constrained control in wireless sensor and actuator networks," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 4151–4163, 2018.
- [13] H. Nakayama, Z. M. Fadlullah, N. Ansari, and N. Kato, "A novel scheme for WSA sink mobility based on clustering and set packing techniques," *IEEE Trans. Autom. Control*, vol. 56, no. 10, pp. 2381–2389, 2011.
- [14] T. Melodia, D. Pompili, and I. F. Akyldiz, "Handling mobility in wireless sensor and actor networks," *IEEE Trans. Mobile Comput.*, vol. 9, no. 2, pp. 160–173, 2010.
- [15] X. Cao, P. Cheng, J. Chen, and Y. Sun, "An online optimization approach for control and communication co-design in networked cyber-physical systems," *IEEE Trans. Ind. Informat.*, vol. 9, no. 1, pp. 439–450, 2013.
- [16] A. Aminifar, P. Eles, Z. Peng, and A. Cervin, "Control-quality driven design of cyber-physical systems with robustness guarantees," in *Proc. of IEEE/ACM DATE*, 2013, pp. 1093–1098.
- [17] D. Goswami, R. Schneider, and S. Chakraborty, "Co-design of cyber-physical systems via controllers with flexible delay constraints," in *Proc. of IEEE ASP-DAC*, 2011, pp. 225–230.
- [18] C. Wu, G. S. Tewolde, W. Sheng, B. Xu, and Y. Wang, "Distributed multi-actuator control for workload balancing in wireless sensor and actuator networks," *IEEE Trans. Autom. Control*, vol. 56, no. 10, pp. 2462–2467, 2011.
- [19] K. Genova and V. Guliashki, "Linear integer programming methods and approaches - a survey," *Cybern. and Inf. Technol.*, vol. 11, no. 1, pp. 1–23, 2011.
- [20] J. F. Benders, "Partitioning procedures for solving mixed-variables programming problems," *Numer. Math.*, vol. 4, no. 1, pp. 238–252, 1962.
- [21] S. Boyd and L. Vandenberghe, "Convex optimization," *Cambridge University Press*, 2004.
- [22] O. Landsiedel, K. Wehrle, and S. Gotz, "Accurate prediction of power consumption in sensor networks," in *Proc. of IEEE EmNetS*, 2005, pp. 37–44.
- [23] Y. Peng, Z. Li, W. Zhang, and D. Qiao, "Prolonging sensor network lifetime through wireless charging," in *Proc. of IEEE RTSS*, 2010, pp. 129–139.
- [24] S. Boyd and J. Matingley, "Branch and bound methods," *Notes for EE364b, Stanford University*, pp. 1–11, 2007.
- [25] S. Albert, "Solving mixed integer linear programs using branch and cut algorithm," Master's thesis, North Carolina State University, 1999.
- [26] E. Rothberg, "An evolutionary algorithm for polishing mixed integer programming solutions," *INFORMS J. Comput.*, vol. 19, no. 4, pp. 534–541, 2007.
- [27] C. Randazzo and H. P. L. Luna, "A comparison of optimal methods for local access uncapacitated network design," *Ann. Oper. Res.*, vol. 106, no. 1, pp. 263–286, 2001.
- [28] <http://optitrack.com/>.
- [29] K. Yoshizawa, H. Hashimoto, M. Wada, and S. Mori, "Path tracking control of mobile robots using a quadratic curve," in *Proc. of IEEE IV*, 1996, pp. 58–63.
- [30] http://www.inpharmix.com/jps/PID_Controller_For_Lego_Mindstorms_Robots.html.
- [31] K. L. Moore, Y. Chen, and Z. Song, "Diffusion-based path planning in mobile actuator-sensor networks (mas-net): some preliminary results," in *Proc. of IEEE SPIE*, vol. 5421, 2004, pp. 58–69.
- [32] M. Marin-Perianu, S. Bosch, R. Marin-Perianu, H. Scholten, and P. Havinga, "Autonomous vehicle coordination with wireless sensor and actuator networks," *ACM Trans. Auton. Adapt. Syst.*, vol. 5, no. 4, p. 13, 2010.
- [33] A. K. Singh, M. Shafique, A. Kumar, and J. Henkel, "Mapping on multi/many-core systems: survey of current and emerging trends," in *Proc. of IEEE/ACM DAC*, 2013, pp. 1–10.
- [34] G. Chen, K. Huang, and A. Knoll, "Energy optimization for real-time multiprocessor system-on-chip with optimal DVFS and DPM combination," *ACM Trans. Embed. Comput. Syst.*, vol. 13, no. 3, pp. 111:1–111:21, 2014.
- [35] K. Cao, G. Xu, J. Zhou, T. Wei, M. Chen, and S. Hu, "QoS-adaptive approximate real-time computation for mobility-aware IoT lifetime optimization," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, pp. 1–12, 2018. DOI 10.1109/TCAD.2018.2873239.



Lei Mo (S'13–M'17) is currently a Postdoctoral Fellow with INRIA Rennes research center, France. He received the B.S. degree from College of Telecom Engineering and Information Engineering, Lanzhou University of Technology, Lanzhou, China, in 2007, and the Ph.D. degree from College of Automation Science and Engineering, South China University of Technology, Guangzhou, China, in 2013. From 2013 to 2015, he was a research fellow with the Department of Control Science and Engineering, Zhejiang University, China. From 2015 to 2017, he was a research fellow with INRIA Nancy research center, France. His current research interests include networked estimation and control in wireless sensor and actuator networks, cyber-physical systems, task mapping and resources allocation in multi-core systems. He serves as an Associate Editor for *KSI Transactions on Internet and Information Systems*, *Journal of Computer and Journal of Electrical and Electronic Engineering*. He also serves as a Guest Editor for *IEEE Access* and *Journal of Computer Networks and Communications* and a TPC Member for several international conferences.



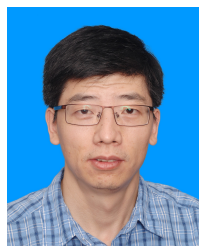
Yeqiong Song received the B.S. degree from Beijing University of Posts and Telecommunications, Beijing, China, in 1984, the Master degree from the Ecole Nationale Supérieure des Tlcommunications (ENST), Paris, France, in 1987, the M.Sc. degree (DEA) from the University of Paris 6 in 1988, the Ph.D. degree from the Institut National Polytechnique de Lorraine (INPL), Nancy, France, in 1991, and the Habilitation to lead research (HdR) degree from the University of Henri Poincaré Nancy 1 in 2004, all in telecommunications and computer science. In 1988, he joined Lorraine Laboratory of IT Research and Applications (LORIA) in Nancy, France. From 1992 to 2005, he was an Associate Professor with the University of Henri Poincaré Nancy 1. From 2001 to 2003, he was a Full-Time Researcher with INRIA Lorraine. Since 2005, he is a full Professor of computer science with the INPL and now with the University of Lorraine. His research interests include modeling and performance evaluation of networks and real-time distributed systems including networked control systems, and the implementation of real-time quality-of-service mechanisms in industrial networks, in-vehicle networks, and wireless IoT and sensor networks. He authored/co-authored more than 100 scientific papers.



Pengcheng You (S'14) received his B.S. degree (with honors) in electrical engineering and Ph.D. degree in control, both from Zhejiang University, China, in 2013 and 2018, respectively. Currently he is a Postdoctoral Fellow with the Whiting School of Engineering (ME & ECE) at Johns Hopkins University, USA. Previously he was a visiting student at California Institute of Technology, USA and also a research intern at Pacific Northwest National Laboratory, USA. His research focuses on smart grid, especially electric vehicle and power market.



Angeliki Kritikakou is currently an Associate Professor at University of Rennes 1 and IRISA - INRIA Rennes Bretagne Atlantique research center. She received her Ph.D. in 2013 from the Department of Electrical and Computer Engineering at University of Patras, Greece and in collaboration with IMEC Research Center, Belgium. She worked for one year as a Postdoctoral Research Fellow at the Department of Modelling and Information Processing (DTIM) at ONERA in collaboration with Laboratory of Analysis and Architecture of Systems (LAAS) and the University of Toulouse, France. Her research interests include embedded systems, real-time systems, mixed-critical systems, hardware/software co-design, mapping methodologies, design space exploration methodologies, memory management methodologies, low power design and fault tolerance.



Xianghui Cao (S'08–M'11–SM'16) received the B.S. and Ph.D. degrees in control science and engineering from Zhejiang University, Hangzhou, China, in 2006 and 2011, respectively. From 2012 to 2015, he was a Senior Research Associate with the Department of Electrical and Computer Engineering, Illinois Institute of Technology, Chicago, IL, USA. He is currently an Associate Professor with the School of Automation, Southeast University, Nanjing, China. His current research interests include cyber-physical systems, wireless network performance analysis, wireless networked control, and network security. He served as the Organization Chair for The Youth Academic Annual Conference of Chinese Association of Automation in 2018, the Symposium Co-Chair for ICNC 2017, the Publicity Co-Chair for ACM MobiHoc 2015 and IEEE/CIC ICC 2015. He was a recipient of the Best Paper Runner-Up Award from ACM MobiHoc in 2014 and the First Prize of Natural Science Award of Ministry of Education of China in 2017. He also serves as an Associate Editor for *ACTA Automatica Sinica* and *IEEE/CAA JOURNAL OF AUTOMATIC SINICA*.