HAL
open science

Real-time Imprecise Computation Tasks Mapping for DVFS-Enabled Networked Systems

Lei Mo, Angeliki Kritikakou, Olivier Sentieys, Xianghui Cao

To cite this version:

Lei Mo, Angeliki Kritikakou, Olivier Sentieys, Xianghui Cao. Real-time Imprecise Computation Tasks Mapping for DVFS-Enabled Networked Systems. IEEE Internet of Things Journal, 2021, 8 (10), pp.8246-8258. 10.1109/JIOT.2020.3044910 . hal-03103821

HAL Id: hal-03103821

https://hal.science/hal-03103821v1

Submitted on 8 Jan 2021

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.

Real-time Imprecise Computation Tasks Mapping for DVFS-Enabled Networked Systems

Lei Mo*, *Member, IEEE,* Angeliki Kritikakou†, *Member, IEEE,* Olivier Sentieys†, *Member, IEEE* and Xianghui Cao*, *Senior Member, IEEE*

***Abstract*—Networked systems are useful for a wide range of applications, many of which require distributed and collaborative data processing to satisfy real-time requirements. On the one hand, networked systems are usually resource constrained, mainly regarding the energy supply of the nodes and their computation and communication abilities. On the other hand, many real-time applications can be executed in an imprecise way, where an approximate result is acceptable as long as the baseline Quality-of-Service (QoS) is satisfied. Such applications can be modeled through Imprecise Computation (IC) tasks. To achieve a better trade-off between QoS and limited system resources, while meeting application requirements, the IC-tasks must be efficiently mapped to the system nodes. To tackle this problem, we firstly construct an IC-tasks mapping problem that aims to maximize system QoS subject to real-time and energy constraints. Dynamic Voltage and Frequency Scaling (DVFS) and multi-path routing are explored to further enhance real-time performance and reduce energy consumption. Secondly, based on the problem structure, we propose an optimal approach to perform IC-tasks mapping and prove its optimality. Furthermore, to enhance the scalability of the proposed approach, we present a heuristic IC-tasks mapping method with low computation time. Finally, the simulation results demonstrate the effectiveness of the proposed methods in terms of the solution quality and the computation time.**

***Index Terms*—Networked Systems, Task Mapping, Imprecise Computation, Quality-of-Service.**

I. Introduction

WIRELESS Sensor and Actuator Networks are networked systems that cannot only measure the physical environment through the sensor nodes, but can also modify it, through the actions performed by the actuator nodes. This characteristic is one of the key elements of the Internet of Things (IoT) [1], [2]. Typical application requirements in such networked systems are low energy consumption, low task execution delay, and high system Quality of Service (QoS) [3]. However, enhancing system QoS often requires more energy consumption and execution time. To balance these contradictory requirements, an efficient mapping of the application tasks on the nodes is required. By properly mapping the tasks on the nodes, we can avoid sending all the data to a central controller that executes the tasks. Part of task execution can be done on-site, on the nodes that have computation and communication capabilities. As a result, only a small part of data is required to be sent, reducing the network traffic and the node workload. This model of computation is also known as "Fog/Edge-computing" [4].

In many application domains [5], such as multimedia processing, mobile target tracking, real-time heuristic search, information gathering and control engineering, an approximate result, obtained before the deadline, is usually acceptable. For example, in video streaming, frames with a lower quality are better than totally missing frames. In target tracking, an approximate estimation of target's location in time is better than an accurate location, obtained too late. In these domains, the applications are usually modeled as Imprecise Computation (IC) tasks [6], where a task is logically decomposed into a mandatory subtask and an optional subtask. The mandatory subtask must be completed before the task deadline to generate the minimum acceptable QoS. Then, the optional subtask is executed, if there are enough free resources in the system. With the IC-tasks model, the longer the optional subtasks are executed, the better the QoS of the result. Dynamic Voltage and Frequency Scaling (DVFS) [7] is an efficient technique that controls both voltage and frequency, thus, the energy and the time required to execute the tasks. By properly mapping IC-tasks on DVFS-enabled nodes of networked systems, the QoS can be further improved, under the limited system sources and the application requirements.

Task mapping is a well-known problem in embedded systems [7]–[16]. However, there are few works that deal with task mapping on the nodes of networked systems [17]–[23]. The majority of these approaches focuses on precise computation tasks (i.e., the tasks without optional subtasks) and systems without DVFS capabilities, see Table I. When both IC-tasks and DVFS are considered, the way to execute IC-tasks is decided by 1) the task mapping, which refers to both the task allocation (on which node a task is executed) and the task scheduling (when each task starts to execute and how long its optional subtask is executed), and 2) the voltage and frequency of the node, which executes the task. In this context, there are three main differences between our work and the existing works:

1) Compared with the task mapping on embedded systems [7]–[16], the task mapping on networked systems is constrained, since some tasks (e.g., sensing and control) have one-to-one allocation on the nodes, whereas the allocation constraints of other tasks (e.g., data processing) may not be restricted.
2) In embedded systems, the task communication cost (time and energy) is usually very small, compared with the task execution cost. Therefore, the impact, that the task mapping decisions have on the task communication cost, is limited. However, in networked systems, the task communication

*L. Mo and X. Cao are with the School of Automation, Southeast University, 210096 Nanjing, China. E-mail: lmo@seu.edu.cn, xhcao@seu.edu.cn.

†A. Kritikakou and O. Sentieys are with the University of Rennes, INRIA, IRISA, CNRS, 35042 Rennes, France. E-mail: angeliki.kritikakou@irisa.fr, olivier.sentieys@irisa.fr.

TABLE I
TASK MAPPING METHOD

		Embedded systems										*Networked systems*							
		[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]	[16]	[17]	[18]	[19]	[20]	[21]	[22]	[23]	Ours
Tasks	Precise	√	√	√	√	√						√	√	√	√	√	√		
	Imprecise						√	√	√	√	√							√	√
	Dependent	√		√	√	√	√			√		√	√	√	√	√	√	√	√
	Independent		√					√	√		√								
Platform	DVFS	√	√	√		√	√			√	√		√				√		√
	Multi-path												√						√
	Comm. cost											√	√	√	√	√	√	√	√
Objective	Max. QoS						√	√	√	√	√							√	√
	Min. Energy	√	√	√	√	√						√	√	√	√	√	√		
Constraint	Real-timeliness	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√
	Energy						√	√	√	√								√	√
Solution	Optimal	√			√		√		√									√	√
	Heuristic		√	√		√		√		√	√	√	√	√	√	√	√		√

cost must be taken into account, and this cost highly depends on the task mapping decision. When tasks are allocated to different nodes, communication delay is introduced, while the nodes consume energy for transmitting the data.

3) In networked systems, the task mapping approaches usually assume that the tasks are precise [17]–[22]. However, since our work considers IC-tasks, a set of new variables (relevant to the execution of optional subtasks) has to be introduced into the task mapping problem. The objective is to maximize the QoS by increasing the execution cycles of the optional subtasks. The longer the optional subtasks are executed, the more energy and time are spent. Hence, the adjustment of optional subtasks affects the objective function as well as the energy and the time constraints of the task mapping problem.

A. Related Work

Existing task mapping approaches focus either on energy-aware mapping or QoS-aware mapping. The majority of energy-aware task mapping approaches usually aim at minimizing energy consumption, under system resource and application constraints [7]–[11]. In order to enhance energy efficiency, DVFS is used. An example considering independent real-time tasks is the work of [8], where the task allocation problem with DVFS is formulated as an Integer Linear Programming (ILP) and solved by a relaxation method based on Linear Programming (LP). When dependent real-time tasks are considered, Mixed Integer Programming (MIP) is usually used to formulate the task mapping problem [7], [9]–[11]. In order to solve an ILP-based task mapping problem, a hybrid Genetic Algorithm (GA) is designed in [9], and an optimal method based on Benders Decomposition (BD) [24] is presented in [10]. Combining DVFS and Dynamic Power Management (DPM), a Mixed-Integer Linear Programming (MILP)-based task mapping problem is formulated in [7]. The number of variables is further reduced by problem refinement, and then, the refined MILP problem is optimally solved by the CPLEX solver. The Mixed-Integer Non-Linear Programming (MINLP)-based task mapping problem in [11] is first relaxed to a MILP by linear approximation, and then, is optimally solved by the Branch and Bound (B&B) method [25].

Existing QoS-aware task mapping approaches usually consider the IC-task model [12]–[16]. Their aim is to maximize system QoS, under real-time and/or energy constraints. In [13], the problem of mapping independent tasks is solved, but the frequency of each processor is decided upfront, whereas task allocation and optional subtasks adjustment are solved one after the other. A similar task mapping problem is studied in [14], where an optimal approach is proposed based on problem decomposition. The work in [15] focuses on task scheduling and optional subtasks adjustment of dependent tasks, whereas task allocation is fixed and given in advance. Other existing approaches focus on bi-objective optimization, e.g., increasing the QoS while reducing the frequency changes under real-time constraints for independent tasks [16]. Dependent tasks are mapped on a DVFS-enabled uni-processor platform in [12]. This problem is first formulated as an Integer Non-Linear Programming (INLP), and then, is relaxed to a convex problem. However, the aforementioned approaches mainly focus on embedded systems, thus, the communication cost is not taken into account. In the networked systems, the nodes also consume energy and time for data transmission.

In networked systems, approaches exist that map precise dependent tasks on the nodes of Wireless Sensor Networks (WSNs) [17]–[22]. For instance, the work of [17] minimizes the overall energy consumption and balances the workload of the system while meeting task deadline through a three-phase heuristic. The task allocation problem with metric – balance the energy consumption of the nodes – is considered in [19]. This problem is first formulated as an INLP, and then, it is transformed to an ILP. Finally, the transformed problem is solved by a greedy algorithm. In [20], [21], the lifetime of system is maximized (i.e., minimize the energy consumption of the node with the lowest energy level) by allocating dependent tasks on the nodes. On this basis, a heuristic method [20] and a game theory method [21] are designed to solve task allocation problems, respectively. However, DVFS is not taken into account in the above approaches. The mapping of dependent tasks and DVFS are jointly addressed in [18], where a two-phase heuristic is proposed. In [22], the energy consumption for communication and task execution is minimized, by allocating the tasks to DVFS-enabled nodes through the ant colony and the bee colony algorithms. However, since the aforementioned approaches mainly focus on the precise computation tasks, the improvement of system QoS is limited. In [23], the dependent IC-tasks mapping problem is formulated as a MILP with the goal of maximizing system QoS and optimally solved by a

BD-based approach. Compared with our preliminary results in [23], the current work takes DVFS and multi-path routing into account and proposes a novel heuristic method to further reduce the computation time. By using DVFS and multi-path routing, we can achieve a better trade-off between real-time performance and energy efficiency.

B. Contributions

Complementary to the state-of-the-art, this work solves the following mapping problem: given a networked system with DVFS-enabled wireless nodes and a set dependent IC-tasks, we aim to determine: 1) which node should the task be executed on; 2) what voltage/frequency level should be used for each task; 3) what is the execution sequence of the tasks on each node; 4) which paths transmit the data required for the task execution; 5) when should a task start its execution; and 6) how many cycles of the optional subtasks are needed to be executed, such that the system QoS is maximized. while meeting the energy supply and the task deadline constraints. In this context, the task mapping decides task allocation, frequency assignment, task sequence, multi-path routing, task start time and task adjustment. Our main contributions are summarized as follows:

1) We formulate the IC-tasks mapping problem that simultaneously optimizes task allocation, frequency assignment, task sequence, multi-path routing, task start time and task adjustment as a MINLP. The objective is to maximize QoS without violating the real-time and energy constraints.
2) We prove that, by replacing the nonlinear items with some auxiliary variables and adding additional linear constraints into the optimization problem, the MINLP problem can be equivalently transformed to a MILP.
3) Based on the BD framework and the idea of closed-loop control, we propose an Optimal Task Mapping (OTM) algorithm to efficiently solve the transformed problem. This method decomposes the transformed problem into two subproblems, each with fewer constraints and variables. The first subproblem is an ILP, which is responsible for task allocation, frequency assignment, task sequence and multi-path routing. The second subproblem is an LP, which determines the start time and the optional cycles of each task. The proposed OTM algorithm iterates using the solutions of these two subproblems. We prove that OTM converges to the optimal solution of the transformed problem.
4) In order to enhance the scalability of the proposed approach, we present a Heuristic Task Mapping (HTM) algorithm, which reduces the computation time of OTM algorithm by solving the subproblems in a sequence.
5) Finally, we provide extensive experimental results to evaluate the performance of the proposed optimal and heuristic task mapping algorithms. The obtained results show significant performance improvements compared with the state-of-the-art task mapping methods in terms of solution quality and computation time.

The rest of this paper is organized as follows. Section II presents the system model and formulates the problem under study. Section III, Section IV and Section V describe the problem linearization method, the optimal and the heuristic task mapping algorithms, respectively. Finally, Section VI shows the simulation results and Section VII concludes this work.

II. System Models and Problem Formulation

A. Motivational Example

Let's consider a Heating, Ventilation, and Air Conditioning (HVAC) system [26] as an example. This system includes eight wireless nodes $\{\theta_1, \ldots, \theta_8\}$, as shown in Fig. 1. The nodes θ_1 and θ_8 equipped with temperature sensors. They monitor the average temperature within their sensing range and use this information to determine if a fire has occurred. The nodes θ_3 and θ_7 are equipped with the humidity sensors, able to measure the humidity in the air. The node θ_4 is connected to a sprinkler, capable of extinguishing a fire within its acting range. The node θ_5 is connected to a fan, capable of reducing the humidity in the air. The nodes equipped with temperature sensor, humidity sensor, sprinkler and fan are marked with S_t, S_h, A_s and A_f, respectively.

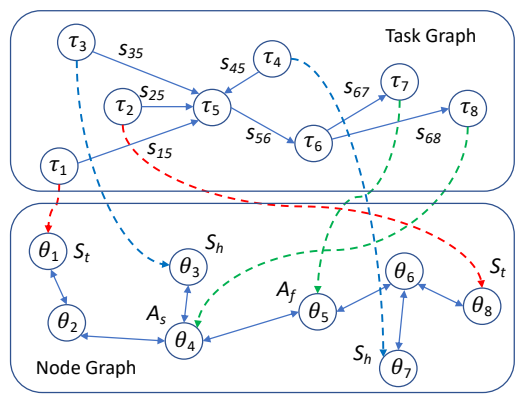


Fig. 1. HVAC system.

The HVAC system periodically measures the temperature and the humidity of the environment, compare the measured values with given thresholds, and decide the corresponding actions. When a fire has been detected, the sprinkler must be activated first, and then, the fan. If these actuators are activated in the opposite order, it will cause serious problems. In the HVAC system, a task can be of sensing, processing, or acting type. For example, τ_1 and τ_2 are the temperature measurement tasks, while τ_3 and τ_4 are the humidity measurement tasks. These sensing tasks generate a set of data, with sizes s_{15}, s_{25}, s_{35}, and s_{45}, that have to be processed by task τ_5. After processing, task τ_5 transmits the result to task τ_6, which determines the control action to be taken. Finally, tasks τ_7 and τ_8 act upon the decision generated by task τ_6 and control the actions of the actuators. The example of Fig. 1 illustrates the task mapping problem addressed in this work.

- Firstly, the allocations of sensing tasks $\{\tau_1, \tau_2, \tau_3, \tau_4\}$ and acting tasks $\{\tau_7, \tau_8\}$ are restricted only to the nodes that have the corresponding capabilities. However, the data processing tasks $\{\tau_5, \tau_6\}$ can be allocated on any node. The dependencies between the tasks define the task graph. Considering that the

communication range of each node is limited, the communication between the nodes is modeled by a node graph. Both task graph and node graph influence the task allocation decision.

- Secondly, each task τ_i is composed of a mandatory subtask and an optional subtask. The optional subtask is executed immediately after its corresponding mandatory subtask. The QoS of the result highly depends on the number of execution cycles of optional subtask. When deciding the task allocation and the task scheduling (i.e., the start time and the end time of each task), the following goals must be achieved at the same time: 1) meet the deadline of the tasks (i.e., each task must be completed within a predefined deadline to generate a result in time), 2) the energy consumption of the nodes (i.e., the energy spent for computation, communication, sensing and acting) should not exceed their energy supply, and 3) increase the execution cycles of optional subtasks to maximize the QoS. These three goals are important, but they contradict with each other, since the real-time and energy constraints may require to sacrifice the system QoS.

B. System Model

1) Task Model: We consider a task set $\mathcal{T}$ of N periodic real-time IC-tasks $\{\tau_1, \ldots, \tau_N\}$. Each task τ_i is described by a tuple $\{o_i, M_i, O_i, t_i^s, d_i, l_i\}$. o_i, M_i and O_i are measured in Worst Case Execution Cycles (WCEC). o_i is the actual WCEC of the optional subtask, M_i is the WCEC of the mandatory subtask, and O_i is the maximum WCEC of the optional subtask o_i (i.e., $0 \leq o_i \leq O_i$). t_i^s and d_i are the start time and the deadline of task τ_i, respectively. l_i is the period of task τ_i, which is also equal to the scheduling horizon H. We introduce a linear QoS function $f_i(o_i)$ for each task τ_i, where the generated QoS increases uniformly with the number of optional cycles, decided for actual execution [5].

We assume that the tasks are dependent and non-preemptive. They are released at the same time 0 and share a common scheduling horizon H. The task set $\mathcal{T}$ is modeled by a Directed Acyclic Graph (DAG) $G_t(\mathcal{V}_t, \mathcal{E}_t)$, where the vertexes $\mathcal{V}_t$ and the edges $\mathcal{E}_t$ represent the set of tasks to be executed and the data dependencies between the tasks, respectively. We consider that the system operates in rounds, which means in each round (i.e., scheduling horizon H) all the tasks in task set $\mathcal{T}$ are executed once. Based on the dependency between the tasks, we introduce a task execution order matrix $\boldsymbol{p} = [p_{ij}]_{N\times N}$. If $p_{ij} = 1$, task τ_i precedes task τ_j and τ_j is the closest task of τ_i, otherwise, $p_{ij} = 0$. As the task graph shown in Fig. 1, we have $p_{15} = p_{25} = p_{35} = p_{45} = p_{56} = p_{67} = p_{68} = 1$.

2) Energy Model: We consider a networked system that contains M wireless DVFS-enabled nodes $\{\theta_1, \ldots, \theta_M\}$, where M_s nodes $\{\theta_1, \ldots, \theta_{M_s}\}$ equipped with sensors and M_a nodes $\{\theta_{M_s+1}, \ldots, \theta_{M_s+M_a}\}$ connected to actuators ($M_s+M_a \leq M$). The processor of each node has L different Voltage/Frequency (V/F) levels $\{(v_1, f_1), \ldots, (v_L, f_L)\}$. The power consumption of the processor under the lth V/F level is computed as

$$P_l^c = P_l^s + P_l^d, \tag{1}$$

where $P_l^s = v_l K_1 e^{K_2 v_l} e^{K_3 v_{bs}} + |v_{bs}| I_j$ is the static power, and $P_l^d = C_{eff} v_l^2 f_l$ is the dynamic power [7]. The constants K_1, K_2 and K_3 are technology dependent. I_j is the approximately constant junction leakage current. v_{bs} is the reverse bias voltage used to reduce the leakage power and can be treated as constant. C_{eff} is the average effective switching capacitance.

We assume that the processor of node θ_k can operate in two modes: *active* mode with the power consumption P_k^c and *idle* mode with the power consumption P_k^0 [8]. The processor goes into idle mode immediately when it has no task to execute. The transition time and energy is incorporated into the task execution time and energy, since they are very small compared to those required to execute a task [8]. For each task τ_i, τ_i starts and ends its execution on the same processor (i.e., no task migration), and the frequency cannot be changed during the execution of the task (i.e., inter-task DVFS).

3) Data Routing Model: Since each node can only communicate directly with the nodes that are within its communication range, we introduce a node graph $\mathcal{G}_n(\mathcal{V}_n, \mathcal{E}_n)$, where the vertexes $\mathcal{V}_n$ represent the nodes, while the edges $\mathcal{E}_n$ represent the communication cost between the nodes. In this paper, we consider multi-path data routing, which means a pair of nodes can communicate with each other through several paths. Specifically, we consider two routing options: the 1st routing path is energy-oriented, while the 2nd routing path is time-oriented, since the system under study is energy and time constrained. For the energy (time)-oriented path, an edge represents the energy (time) required for transmitting and receiving a unit of data between the corresponding two nodes. Therefore, the aim of energy (time)-oriented routing is to find the shortest path. The shortest path can be easily found through the existing methods, such as Dijkstra's algorithm [27].

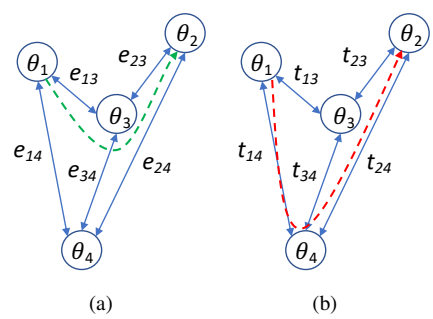


Fig. 2. (a) Energy-oriented data path. (b) Time-oriented data path.

As the examples illustrated in Fig. 2, e_{ij} and t_{ij} is the weight of the edge between the nodes θ_i and θ_j in the energy-oriented and time-oriented graph, respectively. If a data is required to be transmitted from θ_1 to θ_2, we have $\theta_1 \rightarrow \theta_3 \rightarrow \theta_2$ for the energy-oriented path, and $\theta_1 \rightarrow \theta_4 \rightarrow \theta_2$ for the time-oriented path. To describe the time and the energy related to path selection, we introduce 1) a routing energy matrix $\boldsymbol{r} = [r_{\beta\gamma kh}]_{N\times N\times N\times 2}$, where $r_{\beta\gamma kh}$ represents the energy consumed per unit of data at node θ_k, when routing the messages from θ_β to θ_γ through the hth path, and 2) a routing time matrix $\boldsymbol{t} = [t_{\beta\gamma h}]_{N\times N\times 2}$, where $t_{\beta\gamma h}$ denotes the time required to transmit unit of data from θ_β to θ_γ through the hth path. Note that during the data transmission between θ_β to θ_γ, if θ_k is not included in the hth path, we have $r_{\beta\gamma kh} = +\infty$, e.g., $r_{1241} = +\infty$ in Fig. 2(a).

C. Problem Formulation

The IC-tasks mapping problem has the objective of maximizing system QoS subject to real-time and energy constraints, by determining task allocation, frequency assignment, task sequence, multi-path routing, task start time and task adjustment. In order to formulate this problem, we introduce the following variables: 1) binary variable $q_{ik} = 1$ if task τ_i is allocated to node θ_k, otherwise, $q_{ik} = 0$; 2) binary variable $c_{il} = 1$ if task τ_i is executed with frequency f_l, otherwise, $c_{il} = 0$; 3) binary variable $u_{ij} = 1$ if task τ_i proceeds τ_j, otherwise, $u_{ij} = 0$; 4) binary variable $x_{eh} = 1$ if data over edge e in $\mathcal{E}_t$ is routed along the hth path, otherwise, $x_{eh} = 0$; 5) continuous variable t_i^s denotes the start time of task τ_i; 6) continuous variable o_i represents the optional subtask of task τ_i.

We assume that the following matrices are known: 1) task execution order matrix $\boldsymbol{p}$, 2) data size matrix $\boldsymbol{s}$, 3) routing energy matrix $\boldsymbol{r}$, and 4) routing time matrix $\boldsymbol{t}$. The parameters and the variables used in the problem formulation are summarized in Table II. For tractability reasons, during the problem formulation, we consider o_i as continuous variables. When the problem is solved, we round the result down. Since the tasks are typically executed in hundreds of thousands of cycles, one cycle is a very fine-grained unit [12]. Let $\mathcal{N} = \{1, \ldots, N\}$, $\mathcal{M} = \{1, \ldots, M\}$, $\mathcal{M}_s = \{1, \ldots, M_s\}$, $\mathcal{M}_a = \{M_s + 1, \ldots, M_s + M_a\}$ and $\mathcal{H} = \{1, 2\}$. The constraints involved in the task mapping problem can be explained as follows.

1) Task Allocation Constraint: Since each task τ_i is assigned to only one node and the task allocation decisions related to the sensors and the actuators are restricted, we have

$$\sum_{k \in \mathcal{K}} q_{ik} = 1, \ \forall i \in \mathcal{N}, \tag{2}$$

$$q_{ik} = 1, \ \forall (i, k) \in \mathcal{M}_m, \tag{3}$$

where $\mathcal{M}_m$ denote the set of index pairs (tasks and nodes) that have fixed matching. As the example shown in Fig. 1, since $q_{11} = q_{28} = q_{33} = q_{47} = q_{75} = q_{84} = 1$, we have $\mathcal{M}_m = \{(1, 1), (2, 8), (3, 3), (4, 7), (7, 5), (8, 4)\}$.

2) Frequency Assignment Constraint: Since each task τ_i is executed with only one V/F level, we get

$$\sum_{l \in \mathcal{L}} c_{il} = 1, \ \forall i \in \mathcal{N}. \tag{4}$$

3) Data Routing Constraint: Since the data over edge e in $\mathcal{E}_t$ is routed through one path, we obtain

$$\sum_{h \in \mathcal{H}} x_{eh} = 1, \ \forall e = (i, j) \in \mathcal{E}_t. \tag{5}$$

4) Task Sequence Constraints: Before executing a task τ_j, we need to collect all the data generated from its previous dependent tasks. If $p_{ij} = p_{kj} = 1$, tasks τ_i and τ_k precede τ_j and they are the tasks closest to τ_j. When these tasks are allocated to different nodes, the node that executes τ_j should collect the data from the other nodes. In order to avoid communication collision [28], the data are received in sequence, since one node cannot receive the data from multiple nodes simultaneously. For the dependent tasks, e.g., τ_i and τ_j in Fig. 3 ($p_{ij} = 1$), if we allocate these tasks to different nodes θ_β and θ_γ (i.e., $q_{i\beta} = q_{j\gamma} = 1$) and choose the hth path (i.e., $x_{eh} = 1$, $e = (i, j) \in \mathcal{E}_t$) to transmit the data from θ_β to θ_γ, the time required for θ_γ to receive the data from θ_β is $s_{ij} p_{ij} q_{i\beta} q_{j\gamma} x_{eh} t_{\beta\gamma h}$. Therefore, the time spent for receiving the data required by the execution of task τ_j is

$$t_j^r = \sum_{e \in \mathcal{E}_t} \sum_{\beta \in \mathcal{M}} \sum_{\gamma \in \mathcal{M}} \sum_{h \in \mathcal{H}} s_{ij} p_{ij} q_{i\beta} q_{j\gamma} x_{eh} t_{\beta\gamma h}, \ \forall j \in \mathcal{N}. \tag{6}$$

For the dependent tasks, e.g., τ_i and τ_j in Fig. 3 ($p_{ij} = 1$), no matter if they are allocated to the same node or to different

TABLE II
SYMBOLS USED IN THE PROBLEM FORMULATION

Parameters	
M_a	number of actuator nodes
M_s	number of sensor nodes
M	number of nodes
N	number of tasks
L	number of voltage/frequency levels
H	scheduling horizon
θ_k	the kth node
τ_i	the ith task
(v_l, f_l)	the lth voltage/frequency level
P_k^s	static power of node θ_k
P_k^d	dynamic power of node θ_k
P_k^c	active power of node θ_k
P_k^0	idle power of node θ_k
E_k^l	available energy of node θ_k at the lth round
M_i	mandatory cycles of task τ_i
O_i	maximum optional cycles of task τ_i
d_i	deadline of task τ_i
p_{ij}	$= \begin{cases} 1 & \text{if task } \tau_i \text{ proceeds } \tau_j \text{ and } \tau_j \text{ is the nearest task of } \tau_i \\ 0 & \text{else} \end{cases}$
s_{ij}	size of data that task τ_i produces for task τ_j
$r_{\beta\gamma kh}$	energy consumed of node θ_k when routing unit of data from θ_β to θ_γ through the hth path
$t_{\beta\gamma h}$	time required to transmit unit of data from θ_β to θ_γ through the hth path
Binary Variables	
q_{ik}	$= \begin{cases} 1 & \text{if task } \tau_i \text{ is allocated to node } \theta_k \\ 0 & \text{else} \end{cases}$
c_{il}	$= \begin{cases} 1 & \text{if task } \tau_i \text{ is executed with frequency } f_l \\ 0 & \text{else} \end{cases}$
u_{ij}	$= \begin{cases} 1 & \text{if task } \tau_i \text{ proceeds task } \tau_j \\ 0 & \text{else} \end{cases}$
x_{eh}	$= \begin{cases} 1 & \text{if data over edge } e \text{ in } \mathcal{E}_t \text{ is routed along with the } h^{th} \text{ path} \\ 0 & \text{else} \end{cases}$
Continuous Variables	
o_i	optional cycles of task τ_i
t_i^s	start time of task τ_i

nodes, the execution sequence between these tasks is fixed. Therefore, the start and end time of the tasks is bounded by

$$t_j^s \geq p_{ij} t_i^e + t_j^r, \ \forall i, j \in \mathcal{N}, \ i \neq j, \tag{7}$$

where $t_i^c = \sum_{l \in \mathcal{L}} c_{il} \frac{M_i + o_i}{f_l}$ and $t_i^e = t_i^s + t_i^c$ are the execution time and the end time of task τ_i, respectively. If $p_{ij} = 1$, we have $t_j^s \geq t_i^e + t_j^r$, else, (7) is always satisfied.

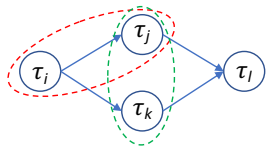


Fig. 3. Task dependency.

On the other hand, for the independent tasks, e.g., τ_j and τ_k in Fig. 3 ($p_{jk} = 0$), if they are allocated to different nodes, the execution of τ_j doesn't affect the execution of τ_k. However, if τ_j and τ_k are allocated to the same node, we need to schedule these tasks so as to make sure that their executions do not overlap with each other, since one processor executes no more than one task at the same time. To this end, we introduce the following constraint:

$$t_i^e + t_j^r \leq t_j^s + (2 - q_{ik} - q_{jk})H + (1 - u_{ij})H, \quad \forall i, j \in \mathcal{N}, \ i \neq j, \ \forall k \in \mathcal{M}. \tag{8}$$

If τ_i and τ_j are executed on the different nodes (i.e., $q_{ik} + q_{jk} \leq 1$), (8) is always satisfied, else, (8) is meaningful: with the task allocation decision $q_{ik} = q_{jk} = 1$, if $u_{ij} = 1$ (i.e., τ_i precedes τ_j), we have $t_i^e + t_j^r \leq t_j^s$; if $u_{ij} = 0$, we get $u_{ji} = 1$ due to $u_{ji} + u_{ij} = 1$. Note that (8) can be rewritten as $t_j^e + t_i^r \leq t_i^s + (2 - q_{jk} - q_{ik})H + (1 - u_{ji})H = t_i^s + (2 - q_{jk} - q_{ik})H + u_{ij}H$. Therefore, we have $t_j^e + t_i^r \leq t_i^s$.

5) Task Deadline Constraint: Since each task τ_i must be finished within a given time threshold d_i, we have

$$t_i^e \leq d_i, \ \forall i \in \mathcal{N}. \tag{9}$$

6) Energy Supply Constraint: With the task allocation decision q_{ik}, the time required to execute all the tasks assigned to node θ_k is $\sum_{i \in \mathcal{N}} q_{ik} t_i^c$, and the idle time of node θ_k being in the scheduling horizon H is $H - \sum_{i \in \mathcal{N}} q_{ik} t_i^c$. Based on the energy model (1), the computation energy of node θ_k in each round is

$$E_k^c = \sum_{i \in \mathcal{N}} q_{ik} e_i^c + \left(H - \sum_{i \in \mathcal{N}} q_{ik} t_i^c \right) P_k^0, \ \forall k \in \mathcal{M}. \tag{10}$$

where $e_i^c = \sum_{l \in \mathcal{L}} c_{il} \frac{M_i + o_i}{f_l} P_l^c$ is the energy required to execute task τ_i. On the other hand, the energy consumed for node θ_k to transmit data in each round is

$$E_k^t = \sum_{e \in \mathcal{E}_t} \sum_{\beta \in \mathcal{M}} \sum_{\gamma \in \mathcal{M}} \sum_{h \in \mathcal{H}} s_{ij} q_{i\beta} q_{i\gamma} x_{eh} r_{\beta\gamma kh}, \ \forall k \in \mathcal{M}. \tag{11}$$

Since the total energy consumed by node θ_k during the scheduling horizon H cannot exceed the energy supply E_k^l, we have

$$E_k^c + E_k^t + E_k^s \leq E_k^l, \ \forall k \in \mathcal{M}_s, \tag{12}$$

$$E_k^c + E_k^t + E_k^a \leq E_k^l, \ \forall k \in \mathcal{M}_a, \tag{13}$$

$$E_k^c + E_k^t \leq E_k^l, \ \forall k \in \mathcal{M}, \ \forall k \notin \mathcal{M}_s, \mathcal{M}_a, \tag{14}$$

where E_k^s is the sensing energy consumption and E_k^a is the acting energy consumption.

Taking the objective (maximize QoS function $\sum_{i \in \mathcal{N}} f_i(o_i)$) and all the constraints mentioned above into account, the Primal Problem (PP) can be formulated as

$$\textbf{PP}: \min_{\substack{\boldsymbol{q}, \boldsymbol{c}, \boldsymbol{u}, \boldsymbol{x}, \\ \boldsymbol{t}^s, \boldsymbol{o}}} -\sum_{i \in \mathcal{N}} f_i(o_i) \tag{15}$$

$$\text{s.t.} \begin{cases} (2) - (14), \\ q_{ik}, c_{il}, u_{ij}, x_{eh} \in \{0, 1\}, \\ 0 \leq t_i^s \leq H, \ 0 \leq o_i \leq O_i. \end{cases}$$

III. Problem Linearization

Since the products $q_{ik} c_{il}$, $q_{i\beta} q_{j\gamma} x_{eh}$, $c_{il} o_i$ and $q_{ik} c_{il} o_i$ are included in the constraints (6)–(14), (15) is a MINLP problem, which is hard to solve directly. Note that q_{ik}, c_{il} and x_{eh} are binary variables, while o_i is a continuous variable. In order to linearize above nonlinear terms, we introduce the following lemmas.

Lemma 3.1: Let b_1, b_2 and g denote the binary variables. The nonlinear constraint $g = b_1 b_2$ can be equivalently replaced by the linear constraints: $g \leq b_1$, $g \leq b_2$ and $g \geq b_1 + b_2 - 1$.

All the proofs of lemmas and theorems are presented in the Appendices for better readability of the main manuscript.

Based on *Lemma 3.1*, we first introduce an auxiliary (binary) variable g_{ikl} to replace the nonlinear term $q_{ik} c_{il}$. Then, we add the following constraints into the PP:

$$\{g_{ikl} \leq q_{ik}, \ g_{ikl} \leq c_{il}, \ g_{ikl} \geq q_{ik} + c_{il} - 1\}, \quad \forall i \in \mathcal{N}, \ \forall k \in \mathcal{M}, \ \forall l \in \mathcal{L}. \tag{16}$$

Similarly, the nonlinear term $q_{i\beta} q_{j\gamma}$ in $q_{i\beta} q_{j\gamma} x_{eh}$ is first replaced by the auxiliary variable $b_{i\beta j\gamma}$ and the following constraints:

$$\{b_{i\beta j\gamma} \leq q_{i\beta}, \ b_{i\beta j\gamma} \leq q_{j\gamma}, \ b_{i\beta j\gamma} \geq q_{i\beta} + q_{j\gamma} - 1\}, \quad \forall e \in \mathcal{E}_t, \ \forall \beta, \gamma \in \mathcal{M}. \tag{17}$$

Then, the nonlinear term $b_{i\beta j\gamma} x_{eh}$ is replaced by the auxiliary variable $w_{i\beta j\gamma h}$ and the following constraints:

$$\{w_{i\beta j\gamma h} \leq b_{i\beta j\gamma}, \ w_{i\beta j\gamma h} \leq x_{eh}, \ w_{i\beta j\gamma h} \geq b_{i\beta j\gamma} + x_{eh} - 1\}, \quad \forall e \in \mathcal{E}_t, \ \forall \beta, \gamma \in \mathcal{M}, \ \forall h \in \mathcal{H}. \tag{18}$$

Note that $q_{ik} c_{il} o_i = g_{ikl} o_i$, where g_{ikl} is a binary variable, while o_i is a continuous variable bounded by $0 \leq o_i \leq O_i$. To deal with the nonlinear terms $c_{il} o_i$ and $g_{ikl} o_i$, we introduce the following lemma.

Lemma 3.2: The spaces $S_1 = \{[y, b, x] | y = bx, -s_1 \leq x \leq s_2\}$ and $S_2 = \{[y, b, x] | -bs_1 \leq y \leq bs_2, y + bs_1 - x - s_1 \leq 0, y - bs_2 - x + s_2 \geq 0\}$ are equivalent, where x is a continuous variable, b is a binary variable, and $s_1, s_2 > 0$ are constants.

Based on *Lemma 3.2*, the nonlinear terms $c_{il} o_i$ and $g_{ikl} o_i$ are replaced by the auxiliary (continuous) variables y_{il} and z_{ikl}, and the following constraints:

$$\{y_{il} \leq c_{il} O_i, \ y_{il} - o_i \leq 0, \ y_{il} - c_{il} O_i - o_i + O_i \geq 0\},$$

$$\forall i \in \mathcal{N},\ \forall l \in \mathcal{L}, \quad (19)$$

$$\{z_{ikl} \le g_{ikl}O_i,\ z_{ikl} - o_i \le 0,\ z_{ikl} - g_{ikl}O_i - o_i + O_i \ge 0\},$$
$$\forall i \in \mathcal{N},\ \forall k \in \mathcal{M},\ \forall l \in \mathcal{L}. \quad (20)$$

Therefore, with the auxiliary variable $w_{i\beta j\gamma h}$, (6) and (11) are rewritten as follows:

$$t_j^r = \sum_{e\in\mathcal{E}_t}\sum_{\beta\in\mathcal{M}}\sum_{\gamma\in\mathcal{M}}\sum_{h\in\mathcal{H}} s_{ij}p_{ij}w_{i\beta j\gamma h}t_{\beta\gamma h},\ \forall j \in \mathcal{N}, \quad (21)$$

$$E_k^t = \sum_{e\in\mathcal{E}_t}\sum_{\beta\in\mathcal{M}}\sum_{\gamma\in\mathcal{M}}\sum_{h\in\mathcal{H}} s_{ij}w_{i\beta j\gamma h}r_{\beta\gamma kh},\ \forall k \in \mathcal{M}. \quad (22)$$

On this basis, with the auxiliary variables g_{ikl}, y_{il} and z_{ikl}, (7)–(10) are reformulated as follows:

$$t_j^s \ge p_{ij}\left(t_i^s + \sum_{l\in\mathcal{L}}\frac{c_{il}M_i + y_{il}}{f_l}\right) + t_j^r,\ \forall i,j \in \mathcal{N},\ i \ne j, \quad (23)$$

$$t_i^s + \sum_{l\in\mathcal{L}}\frac{c_{il}M_i + y_{il}}{f_l} + t_j^r \le t_j^s + (2 - q_{ik} - q_{jk})H + (1 - u_{ij})H,$$
$$\forall i,j \in \mathcal{N},\ i \ne j,\ \forall k \in \mathcal{M}, \quad (24)$$

$$t_i^s + \sum_{l\in\mathcal{L}}\frac{c_{il}M_i + y_{il}}{f_l} \le d_i,\ \forall i \in \mathcal{N}, \quad (25)$$

$$E_k^c = HP_k^0 + \sum_{i\in\mathcal{N}}\sum_{l\in\mathcal{L}}\frac{g_{ikl}M_i + z_{ikl}}{f_l}(P_l^c - P_k^0),\ \forall k \in \mathcal{M}. \quad (26)$$

Finally, (15) is transformed to the following MILP problem:

$$\textbf{PP1}: \min_{\substack{\boldsymbol{q},\boldsymbol{c},\boldsymbol{u},\boldsymbol{x},\boldsymbol{g},\boldsymbol{b},\boldsymbol{w},\\ \boldsymbol{o},\boldsymbol{t}^s,\boldsymbol{y},\boldsymbol{z}}} -\sum_{i\in\mathcal{N}} f_i(o_i) \quad (27)$$
$$\text{s.t.} \begin{cases} (2)-(5), (12)-(14), (16)-(26), \\ q_{ik}, c_{il}, u_{ij}, x_{eh}, g_{ikl}, b_{i\beta j\gamma}, w_{i\beta j\gamma h} \in \{0,1\}, \\ 0 \le o_i, y_{il}, z_{ikl} \le O_i,\ 0 \le t_i^s \le H. \end{cases}$$

Remark 3.1: Lemma 3.1 and *Lemma 3.2* show that the linearization does not change the feasible region and the objective function of the problem. Therefore, solving PP and solving PP1 are equivalent.

IV. Optimal Task Mapping Algorithm

In this section, we present an OTM algorithm to optimally solve the PP1. Our method decomposes the PP1 into two smaller subproblems with less variables: a *Master Problem* (MP) for task allocation $\boldsymbol{q}$, frequency assignment $\boldsymbol{c}$, task sequence $\boldsymbol{u}$, and data routing $\boldsymbol{x}$, and a *Slave Problem* (SP) for task start time $\boldsymbol{t}^s$ and optional cycles adjustment $\boldsymbol{o}$. Then, the optimal solution of PP1 is found by solving the subproblems iteratively. With this decomposition structure, the computational complexity can be greatly reduced.

For the sake of presentation, (27) is reformulated as follows:

$$\textbf{PP2}: \min_{\boldsymbol{x},\boldsymbol{y}}\ \Phi = \boldsymbol{f}^T\boldsymbol{y} \quad (28)$$
$$\text{s.t.} \begin{cases} \boldsymbol{A}\boldsymbol{x} \preceq \boldsymbol{b}_1, \\ \boldsymbol{C}\boldsymbol{x} + \boldsymbol{D}\boldsymbol{y} \preceq \boldsymbol{b}_2, \end{cases}$$

where $\boldsymbol{x}$ and $\boldsymbol{y}$ represent the vector of binary and continuous variables, respectively. The vector $\boldsymbol{f}$ represents the coefficients in the objective function. The matrices $\boldsymbol{A}$, $\boldsymbol{C}$, $\boldsymbol{D}$ and the vectors $\boldsymbol{b}_1$, $\boldsymbol{b}_2$ represent the coefficients in the constraints.

A. MP and SP formulation

According to the structure of PP2, at the kth iteration, the MP and the SP have the forms:

$$\textbf{MP}: \Phi_l(k) = \min_{\boldsymbol{x},\hat{\Phi}}\ \hat{\Phi} \quad (29)$$
$$\text{s.t. } \boldsymbol{A}\boldsymbol{x} \preceq \boldsymbol{b}_1,\ C_1,\ C_2,$$
$$\textbf{SP}: \Phi_u(k) = \min_{\boldsymbol{y}\succeq 0}\ \boldsymbol{f}^T\boldsymbol{y} \quad (30)$$
$$\text{s.t. } \boldsymbol{C}\boldsymbol{x}(k) + \boldsymbol{D}\boldsymbol{y} \preceq \boldsymbol{b}_2,$$

where $\boldsymbol{x}(k)$ is the MP solution at the kth iteration, $C_1: \hat{\Phi} \ge \boldsymbol{\mu}(i)^T(\boldsymbol{C}\boldsymbol{x} - \boldsymbol{b}_2)$, $\forall i \in \mathcal{A}$, $C_2: 0 \ge \hat{\boldsymbol{\mu}}(j)^T(\boldsymbol{C}\boldsymbol{x} - \boldsymbol{b}_2)$, $\forall j \in \mathcal{B}$.

Lemma 4.1: $\Phi_l(k)$ and $\Phi_u(k)$ are the lower and upper bounds on Φ^*, respectively, where Φ^* is the optimal value of Φ.

The MP considers all the binary variables $\boldsymbol{x}$, and the associated part of the constraints (i.e., $\boldsymbol{A}\boldsymbol{x} \preceq \boldsymbol{b}_1$). It also includes the information regarding the SP through a set of constraints (i.e., C_1 and C_2) called *Benders cuts*. Note that the objective function of the PP2 only considers the continuous variables $\boldsymbol{y}$. To facilitate the iterations between the MP and the SP, an *auxiliary* (continuous) variable $\hat{\Phi}$ is introduced into the MP as the objective function, where $\hat{\Phi}$ and Φ have the same physical meaning. At each iteration, based on the SP solution, a new constraint is added into C_1 or C_2 to reduce the gap between the upper and the lower bounds. The problem iterations stop when $\Phi_u(k) - \Phi_l(k) \le \epsilon$, where ϵ is a small positive tolerance.

B. Iterations Between MP and SP

Initially, we set the iteration counter $k = 0$ and the MP solution $\boldsymbol{x}(0)$ that satisfies $\boldsymbol{A}\boldsymbol{x}(0) \preceq \boldsymbol{b}_1$. In addition, we assume that the constraint sets C_1 and C_2 are null, the lower bound $\Phi_l(0) = -\infty$ and the upper bound $\Phi_u(0) = +\infty$.

Since the SP is a LP problem, the strong duality [29] exists between the SP and its dual problem (DSP), i.e., solving SP and solving DSP are equivalent. Instead of solving the SP directly, we solve its dual problem:

$$\max_{\boldsymbol{\mu}\succeq 0}\ \boldsymbol{\mu}^T(\boldsymbol{C}\boldsymbol{x}(k) - \boldsymbol{b}_2) \quad (31)$$
$$\text{s.t. } \boldsymbol{f} + \boldsymbol{D}^T\boldsymbol{\mu} \succeq 0,$$

where $\boldsymbol{\mu} = [\mu_i]_{v\times 1}$ are the dual variables. According to the solution of DSP, we can construct the constraints in C_1 and C_2. Specifically, if (31) has a bounded solution $\boldsymbol{\mu}(k)$, (30) is feasible under given $\boldsymbol{x}(k)$. Therefore, $\mathcal{A} \leftarrow \{k\} \cup \mathcal{A}$ and a new constraint:

$$\hat{\Phi} \ge \boldsymbol{\mu}(k)^T(\boldsymbol{C}\boldsymbol{x} - \boldsymbol{b}_2), \quad (32)$$

is added into C_1. On the other hand, if (31) has an unbounded solution, i.e., $\boldsymbol{\mu}(k)^T(\boldsymbol{C}\boldsymbol{x}(k) - \boldsymbol{b}_2) = +\infty$, (30) is infeasible under given $\boldsymbol{x}(k)$. Therefore, $\mathcal{B} \leftarrow \{k\} \cup \mathcal{B}$ and a new constraint:

$$0 \ge \hat{\boldsymbol{\mu}}(k)^T(\boldsymbol{C}\boldsymbol{x} - \boldsymbol{b}_2), \quad (33)$$

is added into C_2, where $\hat{\boldsymbol{\mu}}(k)$ is the optimal solution to (45) at the kth iteration. Based on the solution of (31), at the kth iteration, the upper bound is updated by

$$\Phi_u(k) = \min\{\Phi_u(k-1), \boldsymbol{\mu}(k)^T(\boldsymbol{C}\boldsymbol{x}(k) - \boldsymbol{b}_2)\}. \tag{34}$$

Finally, with the updated constraint sets C_1 and C_2, (29) is solved again to obtain a solution $\boldsymbol{x}(k+1)$ for the next round iteration.

Theorem 4.1: Constraints (32) and (33) exclude the non-optimal and infeasible solutions of the binary variables $\boldsymbol{x}$, respectively.

Lemma 4.2: With updating the constraints in C_1 and C_2, the lower bound sequence $\{\Phi_l(0), \ldots, \Phi_l(k)\}$ is increasing while the upper bound sequence $\{\Phi_u(0), \ldots, \Phi_u(k)\}$ is decreasing.

Theorem 4.2: The solution found by OTM converges to the global optimal one within a finite number of iterations.

V. Heuristic Task Mapping Algorithm

At each iteration, a new constraint is generated and added into the MP. Therefore, with an increasing number of iterations, the computational complexity and the size of MP both increase. In order to enhance the scalability of the proposed approach, we provide a novel heuristic approach HTM to efficiently solve the PP1. The basic idea of HTM is similar to OTM: both of them are based on problem decomposition. However, the HTM solves the master and the slave problems in sequence (without iteration), thus, the HTM contains two steps. During the first step, we only consider the mandatory subtasks. By balancing the energy consumption of the nodes, while meeting deadline constraints, we obtain the feasible task allocation, frequency assignment, task sequence, and data routing decisions. Under these decisions, in the second step, we determine the start time of the tasks and the cycles of optional subtasks so as to maximize QoS function.

A. Task Allocation Problem

Since the mandatory subtasks must be executed, we initially only consider the allocation, the frequency, the sequence and the routing of the mandatory subtasks (i.e., $o_i = 0$, $\forall i \in \mathcal{N}$). The difficulty during the problem formulation is how to deal with the continuous variables $\{t_1^s, \ldots, t_N^s\}$ (i.e., the start time of the tasks). If the task start time is determined, the PP1 reduces to an ILP problem, since $o_i = 0$. (9) shows that the task deadline constraint must be satisfied (i.e., $t_i^e \le d_i$). In the worst case, we have $t_i^e = d_i$. Thus, the start time of task τ_i is

$$t_i^s = d_i - \sum_{l \in \mathcal{L}} \frac{c_{il} M_i}{f_l}, \ \forall i \in \mathcal{N}. \tag{35}$$

On the other hand, under different optional cycles o_i, task τ_i generates a set of data with size s_{ij} for task τ_j. Therefore, the time required to collect the data for the execution of task τ_j is given by (21), and the energy consumed for node θ_k to transmit the task data is calculated as (22). With $o_i = 0$ and $g_{ikl} = q_{ik} c_{il}$, (7), (8) and (10) have the forms:

$$t_j^s \ge p_{ij} \left(t_i^s + \sum_{l \in \mathcal{L}} \frac{c_{il} M_i}{f_l} \right) + t_j^r, \ \forall i, j \in \mathcal{N}, \ i \ne j, \tag{36}$$

$$t_i^s + \sum_{l \in \mathcal{L}} \frac{c_{il} M_i}{f_l} + t_j^r \le t_j^s + (2 - q_{ik} - q_{jk})H + (1 - u_{ij})H, \quad \forall i, j \in \mathcal{N}, \ i \ne j, \ \forall k \in \mathcal{M}, \tag{37}$$

$$E_k^c = H P_k^0 + \sum_{i \in \mathcal{N}} \sum_{l \in \mathcal{L}} \frac{g_{ikl} M_i}{f_l} (P_l^c - P_k^0), \ \forall k \in \mathcal{M}. \tag{38}$$

Hence, the total energy consumption of node θ_k is

$$E_k^{all} = \begin{cases} E_k^c + E_k^t + E_k^s, \ \forall k \in \mathcal{M}_s, \\ E_k^c + E_k^t + E_k^a, \ \forall k \in \mathcal{M}_a, \\ E_k^c + E_k^t, \ \forall k \in \mathcal{M}, \ \forall k \notin \mathcal{M}_s, \mathcal{M}_a. \end{cases}$$

In order to make sure that we have enough energy to enhance the QoS in the next step, the aim of the first step is to balance the energy consumption of the nodes. Therefore, the Task Allocation Problem (TAP) can be formulated as the following ILP:

$$\textbf{TAP}: \min_{\substack{\boldsymbol{q}, \boldsymbol{c}, \boldsymbol{u}, \boldsymbol{x}, \\ \boldsymbol{g}, \boldsymbol{b}, \boldsymbol{w}}} \left\{ \max_{\forall k \in \mathcal{M}} \frac{E_k^{all}}{E_k^l} \right\} \tag{39}$$

$$\text{s.t.} \begin{cases} (2) - (5), (12) - (14), (16) - (18), (21), (22), (35) - (38), \\ q_{ik}, c_{il}, u_{ij}, x_{eh}, g_{ikl}, b_{i\beta j\gamma}, w_{i\beta j\gamma h} \in \{0, 1\}. \end{cases}$$

When solving the TAP, the polynomial-time method such as the Feasibility Pump (FP) method [30] can be used to reduce the computational complexity. In particular, we introduce a set of auxiliary (continuous) variables $\{\hat{\boldsymbol{q}}, \hat{\boldsymbol{c}}, \hat{\boldsymbol{u}}, \hat{\boldsymbol{x}}, \hat{\boldsymbol{g}}, \hat{\boldsymbol{b}}, \hat{\boldsymbol{w}}\}$, which are assumed to be within the range $[0, 1]$, to replace the original integer variables $\{\boldsymbol{q}, \boldsymbol{c}, \boldsymbol{u}, \boldsymbol{x}, \boldsymbol{g}, \boldsymbol{b}, \boldsymbol{w}\}$. Then, we solve the relaxed TAP (LP) and round the solution to the nearest binary matrix that is feasible to the TAP.

B. Task Scheduling Problem

Based on the solution of TAP, we obtain the decisions for the allocation, the frequency, the sequence and the routing of the mandatory subtasks (i.e., $\boldsymbol{q}$, $\boldsymbol{c}$, $\boldsymbol{u}$ and $\boldsymbol{x}$). When the values of binary variables are determined, the PP1 reduces to a LP problem. Note that the continuous variables, i.e., the task start time t_i^s and the optional subtask cycles o_i, affect the constraints (7)-(14). With the aim to maximize QoS under real-time and energy constraints, the Task Scheduling Problem (TSP) can be formulated as

$$\textbf{TSP}: \min_{\boldsymbol{o}, \boldsymbol{t}^s} \ -\sum_{i \in \mathcal{N}} g_i(o_i) \tag{40}$$

$$\text{s.t.} \begin{cases} t_j^s \ge p_{ij} t_i^e + t_j^r, \ \forall i, j \in \mathcal{N}, \ i \ne j, \\ t_i^e + t_j^r \le t_j^s + (2 - q_{ik} - q_{jk})H + (1 - u_{ij})H, \\ \qquad \forall i, j \in \mathcal{N}, \ i \ne j, \ \forall k \in \mathcal{M}, \\ t_i^e \le d_i, \ \forall i \in \mathcal{N}, \\ E_k^c + E_k^t + E_k^s \le E_k^l, \ \forall k \in \mathcal{M}_s, \\ E_k^c + E_k^t + E_k^a \le E_k^l, \ \forall k \in \mathcal{M}_a, \\ E_k^c + E_k^t \le E_k^l, \ \forall k \in \mathcal{M}, \ \forall k \notin \mathcal{M}_s, \mathcal{M}_a, \\ 0 \le o_i \le O_i, \ 0 \le t_i^s \le H. \end{cases}$$

where $t_i^c = \sum_{l \in \mathcal{L}} c_{il} \frac{M_i + o_i}{f_l}$ and $t_i^e = t_i^s + t_i^c$. E_k^c and E_k^t are given by (10) and (11), respectively.

For the HTM algorithm, the binary variables $\{\boldsymbol{q}, \boldsymbol{c}, \boldsymbol{u}, \boldsymbol{x}\}$ and the continuous variables $\{\boldsymbol{o}, \boldsymbol{t}^s\}$ are solved in the TAP and the TSP separately. Therefore, we can avoid adding the auxiliary variables $\{\boldsymbol{y}, \boldsymbol{z}\}$ and the additional constraints (19) and (20) into the problem to linearize the nonlinear items $c_{il}o_i$ and $q_{ik}c_{il}o_i$. In addition, the problem sizes of TAP and TSP are fixed and these problems are solved only once. Therefore, compared with OTM algorithm, the computation time of HTM algorithm can be greatly reduced.

Remark 5.1: The structure of PP1 shows that the feasibility of this problem is mainly determined by the real-time and the energy related constraints, but these constraints are relaxed in the TAP (i.e., $t_i^e = d_i$ and $o_i = 0$). Therefore, if the PP1 is feasible, the TAP is also feasible, and further, if the TAP is feasible, the TSP is also feasible. This is because in the worst case, we have $t_i^s = d_i - t_i^c$ and $o_i = 0$, which is consistent with the assumption of the TAP.

VI. Simulation Results

For evaluating our approach, we consider a HVAC system with 25 nodes, where 10 nodes equipped with sensors and 5 nodes equipped with actuators. The values and the tuned parameters of the experimental set-up are summarized in Table III. The processor of the node is based on 70 nm technology [7], and the WCECs of the mandatory subtask M_i and the maximum optional subtask O_i are calculated from the MiBench and the MediaBench benchmarks [31]. We consider a linear QoS function $g_i(o_i) = o_i$ [13] and assume that all data items are unit size (i.e., $s_{ij} = 1$) [19].

In order to set a deadline d_i of task τ_i, we introduce a temporary data receiving time $\hat{t}_i^r$ and a relative deadline $\hat{d}_i$. If $p_{ij} = 1$, τ_i precedes τ_j and τ_j is the closest task of τ_i, $d_j = \max_{\forall i, p_{ij} \neq 1}\{d_i + \hat{t}_i^r + \hat{d}_i\}$. If τ_j is an entry task, $d_j = \hat{d}_j$. Let $\min_{\forall \beta,\gamma,h}\{t_{\beta\gamma h}\}$ and $\max_{\forall \beta,\gamma,h}\{t_{\beta\gamma h}\}$ denote the minimum and the maximum time required to transmit the data between the nodes. Since the number of tasks precede task τ_i is $\sum_{j\in\mathcal{N}} p_{ji}$, the temporary data receiving time $\hat{t}_i^r$ is assumed to be within the range $[\hat{t}_{i,\min}^r, \hat{t}_{i,\max}^r]$, where $\hat{t}_{i,\min}^r$ and $\hat{t}_{i,\max}^r$ are the minimum and the maximum time required to transmit the data related to τ_j's previous dependent tasks, respectively. The relative deadline $\hat{d}_i$ is assumed to be within the range $[\hat{d}_{i,\min}, \hat{d}_{i,\max}]$, where $\hat{d}_{i,\min}$ and $\hat{d}_{i,\max}$ are the minimum and the maximum time required to execute a task with $M_i + O_i$ cycles, respectively.

We assume that the scheduling horizon $H = \max_{\forall i}\{d_i\}$ and $E_k^h = E_{k,\min}^t + E_{k,\min}^c$, where $E_{k,\min}^t$ and $E_{k,\min}^c$ are the minimum energy required for the node θ_k to transmit all the task data and execute all the mandatory subtasks, respectively. The energy supply of node θ_k is set to $E_k^l = \eta E_k^h$, where $\eta \in [0, 1]$ is an energy efficiency factor. For the sensor and the actuator nodes, we set $E_k^l = 2\eta E_k^h$ and $E_k^l = 3\eta E_k^h$, respectively, since these nodes consume more energy for sensing and acting tasks. Note that different processor and task parameters lead to different values in the parameters $\{\boldsymbol{A}, \boldsymbol{C}, \boldsymbol{D}, \boldsymbol{f}, \boldsymbol{b}_1, \boldsymbol{b}_2\}$ for the PP1. However, the problem structures under different values of parameters are the same. Therefore, the proposed methods are still applicative.

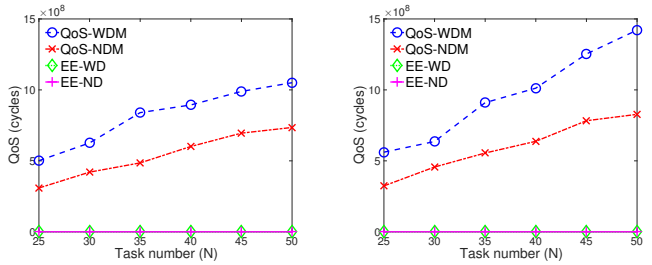
TABLE III
Simulation parameters

Processor θ_k characteristics					
v_l (V)	0.65	0.7	0.75	0.8	0.85
f_l (GHz)	1.01	1.26	1.53	1.81	2.10
P_l^d (mW)	184.9	266.7	370.4	498.9	655.5
P_l^s (mW)	246	290.1	340.3	397.6	462.7
P_0^s (μW)	80				
$M = 25$		$M_s = 10$		$M_a = 5$	
Task τ_i characteristics					
$M_i, O_i \in [4 \times 10^7, 6 \times 10^8]$					
Objective function					
$\sum_{i\in\mathcal{N}} g_i(o_i) = \sum_{i\in\mathcal{N}} o_i$					
Constraints					
$\hat{t}_{i,\min}^r = \sum_{j\in\mathcal{N}} p_{ji} \min_{\forall\beta,\gamma,h}\{t_{\beta\gamma h}\}$					
$\hat{t}_{i,\max}^r = \sum_{j\in\mathcal{N}} p_{ji} \max_{\forall\beta,\gamma,h}\{t_{\beta\gamma h}\}$					
$\hat{d}_{i,\min} = \min_{\forall l}\{\frac{M_i+O_i}{f_l}\}$			$\hat{d}_{i,\max} = \max_{\forall l}\{\frac{M_i+O_i}{f_l}\}$		
$\hat{t}_i^r \in [\hat{t}_{i,\min}^r, \hat{t}_{i,\max}^r]$			$\hat{d}_i \in [\hat{d}_{i,\min}, \hat{d}_{i,\max}]$		
$d_j = \max_{\forall i, p_{ij}\neq 1}\{d_i + \hat{t}_i^r + \hat{d}_i\}$			$H = \max_{\forall i}\{d_i\}$		
$E_{k,\min}^t = N \min_{\forall\beta,\gamma,h}\{r_{\beta\gamma kh}\}$					
$E_{k,\min}^c = HP_k^0 + \sum_{i\in\mathcal{N}}[\min_{\forall l}\{\frac{M_i}{f_l}(P_l^s + P_l^d - P_k^0)\}]$					
$E_k^l = \eta E_k^h$					
Tuned parameters					
Min/Max/Step		$N : 25/50/5$		$\eta : 0.8/0.9/0.1$	

Firstly, we compare the system performance (i.e., the system energy consumption and the system QoS) with the proposed task mapping method (i.e., PP) and other task mapping methods [18], [19], [23]. Secondly, we explore the algorithm performance (i.e., the computation time and the system QoS) of the proposed OTM and HTM methods with: Branch and Bound (B&B) [25] and Branch and Cut (B&C) [32], which are known to provide the optimal solution for the MILP problem. The simulations are performed on a PC with quad-core 2.5 GHz Intel i7 processor and 16 GB RAM, and the algorithms are implemented in Matlab 2016a with CPLEX solver.

Let QoS-WDM, QoS-NDM, EE-ND and EE-WD denote the methods proposed in this paper, [23], [19] and [18], respectively. Specifically, QoS-WDM and QoS-NDM are QoS-aware task mapping methods, while EE-ND and EE-WD are energy-aware task mapping methods. Compared with QoS-NDM, DVFS and multi-path routing are considered in our QoS-WDM method. In addition, compared with EE-ND, DVFS is considered in EE-WD. Fig. 4 and Fig. 5 show the system performance under these task mapping methods. From Fig. 4, we observe that the QoS increases with the values of N and η in QoS-WDM and QoS-NDM, while the QoS is always equal to 0 in EE-ND and EE-WD. This is because the aims of QoS-WDM and QoS-NDM are to maximize QoS under energy and real-time constraints, while the aims of EE-ND and EE-WD are to minimize energy under real-time constraints. Therefore, with values of N and η increasing, more optional subtasks are executed in QoS-WDM and QoS-NDM, thus, a higher QoS is achieved. The cycles of mandatory subtasks are fixed and they must be always executed. However, the less optional subtask cycles are executed, the less is the energy consumed to execute the tasks. Thus, the execution cycles of the optional subtasks are 0 in EE-ND and EE-WD. Fig. 4 also shows that the QoS achieved by QoS-

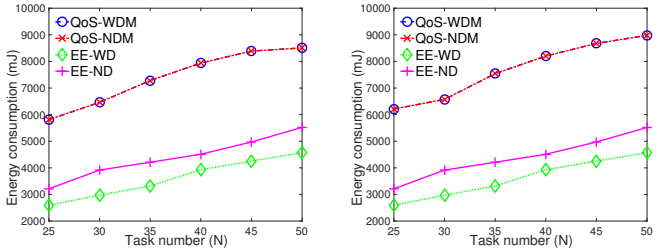
WDM is higher than QoS-NDM. This is because DVFS and multi-path routing are used in QoS-WDM. Compared with QoS-NDM, where the frequency assignment and the routing path selection are fixed, QoS-WDM is able to find better decisions for frequency assignment and routing path selection, increasing QoS under time and energy constraints.



(a) $\eta = 0.8$ (b) $\eta = 0.9$

Fig. 4. QoS under different task mapping methods with η and N varying.

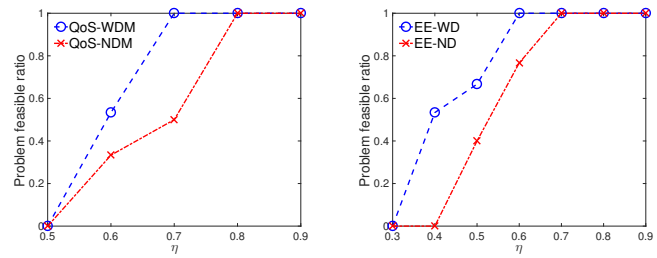
The consumed energy under the different task mapping methods is compared in Fig. 5. Although QoS-WDM and QoS-NDM require more energy than EE-ND and EE-WD, the consumed energy is always smaller than the supplied energy, as the energy related constraints (12)–(14) must be satisfied. From Fig. 4 and Fig. 5, we observe that QoS-aware task mapping method provides a better usage of system resources to enhance QoS. Fig. 5 also shows that the energy efficiency factor η doesn't affect the energy consumption of EE-ND and EE-WD, under the same number of tasks N, since EE-ND and EE-WD aim to minimize the energy consumption. In addition, EE-WD consumed less energy than EE-ND. Typically, for the energy-aware task mapping problem, as long as the constraints (e.g., deadline and energy) allow it, methods applying DVFS is able to achieve a better energy efficiency compared to methods without DVFS.



(a) $\eta = 0.8$ (b) $\eta = 0.9$

Fig. 5. Energy consumption of the nodes under different task mapping methods with η and N varying.

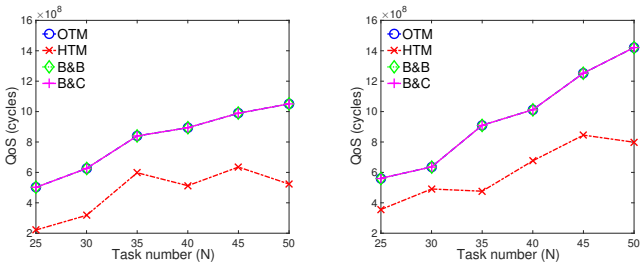
In order to further evaluate the behavior of the proposed QoS-aware task mapping QoS-WDM, we compare the schedulability of QoS-WDM and QoS-NDM, using the problem feasible ratio as a metric. We set the number of tasks $N = 30$ and change the value of energy efficiency factor η. Under a given η, we independently construct and solve the task mapping problems 30 times (i.e., $N_f = 30$) for QoS-WDM and QoS-NDM. For each experiment the parameters are randomly generated and same values are used for QoS-WDM and QoS-NDM. Let N_w and N_n denote the times that the task mapping problems in QoS-WDM and QoS-NDM are feasible, respectively. Therefore, the problem feasible ratios for QoS-WDM and QoS-NDM are defined as N_w/N_f and N_n/N_f, respectively. From Fig. 6(a), we observe that with the value of η increasing, the problem feasible ratios of QoS-WDM and QoS-NDM increase as well. This is because with a higher energy supply, a processor can use a higher frequency to execute faster an assigned task. Therefore, the time and the energy related constraints are easier to be satisfied. Fig. 6(a) shows that the problem feasible ratio of QoS-WDM is always higher than that of QoS-NDM. This is due to fact that, by considering DVFS and multi-path routing in QoS-WDM, the explored design space is larger, allowing QoS-WDM to find solutions, even for cases where QoS-NDM is not able to. Similar are the results when we compare the methods with and without DVFS for the energy-aware task mapping problem, as shown in Fig. 6(b).



(a) QoS-aware task mapping. (b) Energy-aware task mapping.

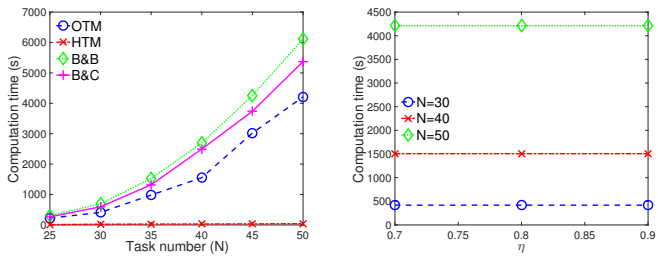
Fig. 6. Problem feasible ratio under different task mapping methods with η varying.

The QoS and the computation time of using OTM, HTM, B&B and B&C to solve task mapping problem PP1 under different number of tasks N and energy efficiency factor η are compared in Fig. 7 and Fig. 8. Fig. 7 shows that the solutions found by OTM, B&B and B&C are the same. This means that OTM is able to find the optimal solution, verifying our analysis about the convergence of OTM. In addition, the achieved QoS increases with the value of η, since more optional subtasks are executed. On the other hand, since HTM just provides a feasible solution, OTM achieves higher QoS than HTM. As shown in Fig. 8(a), with the value of N increasing, more variables and constraints are involved into the problem, thus, the algorithm computation time increases. However, compared with OTM, HTM has a negligible computation time, since HTM only needs to solve two polynomial-time problems in sequence. In addition, OTM has a shorter computation time than that of B&B and B&C. B&C combines the benefits of B&B and Gomory cutting scheme and can better balance optimality, efficiency and stability. Usually, B&C has a faster convergence speed than that of B&B [32]. Note that the computational complexity of an optimization problem is highly related to the number of variables and constraints. Solving smaller problems with less variables and constraints (i.e., MP and SP) iteratively is more efficient than solving a single large problem [33]. Fig. 8(b) shows that the influence of the energy efficiency factor η on the computation time of OTM is limited, since η doesn't change the problem size (i.e., the number of variables and constraints).



(a) $\eta = 0.8$ (b) $\eta = 0.9$

Fig. 7. QoS under different algorithms with η and N varying.



(a) $\eta = 0.8$ (b) Computation time of OTM.

Fig. 8. Computation time under different algorithms with η and N varying.

VII. Conclusion

In this paper, we study the IC-tasks mapping problem for the networked system. We consider dependent IC-tasks executed on the wireless DVFS-enabled nodes with real-time and energy constraints. The design objective is to assign the IC-tasks to the nodes and adjust the start and the end time of the tasks so as to maximize QoS without violating the real-time and the energy constraints. By introducing DVFS and multi-path routing, we are able to achieve a better trade-off between real-time performance and energy efficiency. We first develop a MINLP model to describe this task mapping problem. Then, we propose a MILP description of this model without performance degradation. Through the problem transformation, the problem structure can be simplified, thus, the optimal solution is easier to find. This problem is optimally solved by the proposed OTM algorithm. A novel algorithm, HTM, is proposed to reduce the computation time. Our numerical results show that OTM is guaranteed to converge to the optimal solution, while HTM is able to find a feasible solution within a negligible computation time compared with OTM. Moreover, the proposed QoS-aware task mapping strategy outperforms other task mapping strategies in term of QoS-enhancing and energy-utilizing.

Acknowledgment

This work was supported in part by the Open Research Project of the State Key Laboratory of Industrial Control Technology, Zhejiang University, China under Grant ICT20058, in part by the Fundamental Research Funds for the Central Universities, China under Grant 2242020R10059, in part by the Outstanding Young Scholarship of Jiangsu Province under Grant BK20180012, in part by the Southeast University "Zhongying Young Scholars" and "Zhishan Scholars" Projects.

Appendix A
Proof of Lemma 3.1

Proof: The inequalities $g \le b_1$ and $g \le b_2$ ensure that $g = 0$ if either $b_1 = 0$ or $b_2 = 0$. On the other hand, the inequality $g \ge b_1 + b_2 - 1$ guarantees that $g = 1$ if both variables b_1 and b_2 are set to 1. ■

Appendix B
Proof of Lemma 3.2

Proof: Since $y = bx$ and $-s_1 \le x \le s_2$, we have $-bs_1 \le y \le bs_2$. And further, we obtain $(b-1)(x+s_1) \le 0$ and $(b-1)(x-s_2) \ge 0$ due to $-s_1 \le x \le s_2$ and $b \in \{0,1\}$. Therefore, we have $y + bs_1 - x - s_1 \le 0$ and $y - bs_2 - x + s_2 \ge 0$. $S_1 \to S_2$ holds.

If $b = 0$, since $-bs_1 \le y \le bs_2$, $y + bs_1 - x - s_1 \le 0$ and $y - bs_2 - x + s_2 \ge 0$, we have $y = 0$ and $-s_1 \le x \le s_2$. On the other hand, if $b = 1$, we have $-s_1 \le y = x \le s_2$. $S_2 \to S_1$ holds. ■

Appendix C
Proof of Lemma 4.1

Proof: Note that the MP can be reformulated as follows

$$\hat{\Phi}(k) = \min_{\boldsymbol{x}} \left\{ \max_{\forall i \in \mathcal{A}} \ \boldsymbol{\mu}(i)^T (\boldsymbol{C}\boldsymbol{x} - \boldsymbol{b}_2) \right\} \quad \text{s.t. } \boldsymbol{A}\boldsymbol{x} \preceq \boldsymbol{b}_1, \ C_2. \tag{41}$$

It can be solved by only considering the binary variables $\boldsymbol{x}$. Comparing (41) with the following problem

$$\hat{\Phi}'(i) = \min_{\boldsymbol{x}} \ \boldsymbol{\mu}(i)^T (\boldsymbol{C}\boldsymbol{x} - \boldsymbol{b}_2) \quad \text{s.t. } \boldsymbol{A}\boldsymbol{x} \preceq \boldsymbol{b}_1, \ C_2,$$

we have $\hat{\Phi}(k) \ge \hat{\Phi}'(i)$. Without loss of generality, we assume that

$$\hat{\Phi}(k) = \hat{\Phi}'(l) = \max_{\forall i \in \mathcal{A}} \left\{ \hat{\Phi}'(i) \right\}.$$

Since

$$\hat{\Phi}(k) = \min_{\boldsymbol{x}} \boldsymbol{\mu}(l)^T (\boldsymbol{C}\boldsymbol{x} - \boldsymbol{b}_2) \le \boldsymbol{\mu}(l)^T (\boldsymbol{C}\boldsymbol{x}^* - \boldsymbol{b}_2) \le \max_{\boldsymbol{\mu}} \boldsymbol{\mu}^T (\boldsymbol{C}\boldsymbol{x}^* - \boldsymbol{b}_2) = \Phi^*,$$

where $\boldsymbol{x}^*$ is the optimal value of $\boldsymbol{x}$, we get $\Phi_l(k) = \hat{\Phi}(k)$ is a lower bound of Φ^* and $\hat{\Phi}$ has the same physical meaning as Φ.

On the other hand, according to (34), we have

$$\Phi_u(k) = \min_{1 \le i \le k} \left\{ \boldsymbol{\mu}(i)^T (\boldsymbol{C}\boldsymbol{x}(i) - \boldsymbol{b}_2) \right\}. \tag{42}$$

In addition, due to the strong duality between the SP and its dual problem, we get

$$\boldsymbol{\mu}(i)^T (\boldsymbol{C}\boldsymbol{x}(i) - \boldsymbol{b}_2) = \min_{\boldsymbol{y} \succeq 0} \boldsymbol{f}^T \boldsymbol{y} | \boldsymbol{x}(i) \ge \min_{\boldsymbol{y} \succeq 0} \boldsymbol{f}^T \boldsymbol{y} | \boldsymbol{x}^* = \Phi^*. \tag{43}$$

Based on (42) and (43), $\Phi_u(k)$ is an upper bound of Φ^*. ■

Appendix D
Proof of Lemma 4.2

Proof: Note that the MP is a minimization problem, and the non-optimal values of binary variables $\boldsymbol{x}^*$ have been excluded by the constraints in the set C_1. In addition, with iteration number k increasing, more constraints are added into the MP. Therefore, the feasible region of the MP will shrink. Accordingly, $\Phi_l(k+1) = \hat{\Phi}(k+1)$ is larger than the previous lower bounds $\{\Phi_l(0), \ldots, \Phi_l(k)\}$. On the other hand, based on (34), $\Phi_u(k+1)$ is smaller than the previous upper bounds $\{\Phi_u(0), \ldots, \Phi_u(k)\}$. ■

Appendix E
Proof of Theorem 4.1

Proof: If (31) has a bounded solution, the SP is feasible under the given MP solution $\boldsymbol{x}(k)$. However, since $\boldsymbol{x}(k)$ is a non-optimal solution, we have $\hat{\Phi}(k) < \boldsymbol{\mu}(k)^T(\boldsymbol{C}\boldsymbol{x}(k) - \boldsymbol{b}_2)$. Therefore, the non-optimal solution $\boldsymbol{x}(k)$ is excluded by $\hat{\Phi} \geq \boldsymbol{\mu}(k)^T(\boldsymbol{C}\boldsymbol{x} - \boldsymbol{b}_2)$.

On the other hand, if (31) has an unbounded solution, the SP is infeasible under the given MP solution $\boldsymbol{x}(k)$. However, this problem is feasible if the positive variables $\boldsymbol{\xi} = [\xi_i]_{v\times 1}$ are introduced to relax the constraints. In order to minimize $\boldsymbol{\xi}$, we construct the following problem:

$$\begin{aligned} \min_{\boldsymbol{y},\boldsymbol{\xi}\succeq 0} \ & \mathbf{1}^T\boldsymbol{\xi} \\ \text{s.t.} \ & \boldsymbol{C}\boldsymbol{x}(k) + \boldsymbol{D}\boldsymbol{y} \preceq \boldsymbol{b}_2 + \boldsymbol{\xi}. \end{aligned} \tag{44}$$

Since (44) is a LP problem, we solve its dual problem:

$$\begin{aligned} \max_{\hat{\boldsymbol{\mu}}\succeq 0} \ & \hat{\boldsymbol{\mu}}^T(\boldsymbol{C}\boldsymbol{x}(k) - \boldsymbol{b}_2) \\ \text{s.t.} \ & \mathbf{1} - \hat{\boldsymbol{\mu}} \succeq 0, \ \boldsymbol{D}^T\hat{\boldsymbol{\mu}} \succeq 0, \end{aligned} \tag{45}$$

where $\hat{\boldsymbol{\mu}} = [\hat{\mu}_i]_{v\times 1}$ are the dual variables.

Let $\boldsymbol{\xi}(k)$ and $\hat{\boldsymbol{\mu}}(k)$ denote the solutions of (44) and (45), respectively. Since the relaxation variables with respect to infeasible constraints are non-zero, we have $\mathbf{1}^T\boldsymbol{\xi}(k) > 0$. On this basis, due to the strong duality between (44) and (45), we get $\mathbf{1}^T\boldsymbol{\xi}(k) = \hat{\boldsymbol{\mu}}(k)^T(\boldsymbol{C}\boldsymbol{x}(k) - \boldsymbol{b}_2) > 0$. Therefore, the infeasible solution $\boldsymbol{x}(k)$ is excluded by $0 \geq \hat{\boldsymbol{\mu}}(k)^T(\boldsymbol{C}\boldsymbol{x} - \boldsymbol{b}_2)$. ■

Appendix F
Proof of Theorem 4.2

Proof: Note that the gap between the lower and upper bounds gradually reduces. In addition, the dimension of binary variables $\boldsymbol{x}$ is finite, and the non-optimal and infeasible values are excluded. Therefore, the solution $(\boldsymbol{x}(k), \boldsymbol{y}(k))$ converges to optimal one $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ within a finite number of iterations. ■

References

[1] M. Chiang and T. Zhang, "Fog and IoT: an overview of research opportunities," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 854–864, 2016.

[2] 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.

[3] M. A. Razzaque, M. Milojevic-Jevric, A. Palade, and S. Clarke, "Middleware for internet of things: a survey," *IEEE Internet Things J.*, vol. 3, no. 1, pp. 70–95, 2016.

[4] H. Zahaf, A. E. H. Benyamina, R. Olejnik, and G. Lipari, "Energy-efficient scheduling for moldable real-time tasks on heterogeneous computing platforms," *J. Syst. Architect.*, vol. 74, pp. 46–60, 2017.

[5] H. Aydin, R. Melhem, D. Mosse, and P. Mejia-Alvarez, "Optimal reward-based scheduling for periodic real-time tasks," *IEEE Trans. Comput.*, vol. 50, no. 2, pp. 111–130, 2001.

[6] J. W. S. Liu, W. K. Shih, K. J. Lin, R. Bettati, and J. Y. Chung, "Imprecise computations," *Proc. IEEE*, vol. 82, no. 1, pp. 83–94, 1994.

[7] 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.

[8] D. Li and J. Wu, "Minimizing energy consumption for frame-based tasks on heterogeneous multiprocessor platforms," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 3, pp. 810–823, 2015.

[9] A. Mahmood, S. A. Khan, F. Albalooshi, and N. Awwad, "Energy-aware real-time task scheduling in multiprocessor systems using a hybrid genetic algorithm," *Electron.*, vol. 6, no. 2, 2017.

[10] A. Emeretlis, G. Theodoridis, P. Alefragis, and N. Voros, "A logic-based Benders decomposition approach for mapping applications on heterogeneous multicore platforms," *ACM Trans. Embed. Comput. Syst.*, vol. 15, no. 1, pp. 19:1–19:28, 2016.

[11] L. F. Leung, C. Y. Tsui, and W. H. Ki, "Simultaneous task allocation, scheduling and voltage assignment for multiple-processors-core systems using mixed integer nonlinear programming," in *Proc. IEEE International Symposium on Circuits and Systems*, 2003, pp. 309–312.

[12] L. A. Cortes, P. Eles, and Z. Peng, "Quasi-static assignment of voltages and optional cycles in imprecise-computation systems with energy considerations," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 14, no. 10, pp. 1117–1129, 2006.

[13] T. Wei, J. Zhou, K. Cao, P. Cong, M. Chen, G. Zhang, X. S. Hu, and J. Yan, "Cost-constrained QoS optimization for approximate computation real-time tasks in heterogeneous MPSoCs," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 37, no. 9, pp. 1733–1746, 2018.

[14] L. Mo, A. Kritikakou, and O. Sentieys, "Decomposed task mapping to maximize QoS in energy-constrained real-time multicores," in *Proc. IEEE International Conference on Computer Design*, 2017, pp. 493–500.

[15] H. Yu, Y. Ha, and B. Veeravalli, "Quality-driven dynamic scheduling for real-time adaptive applications on multiprocessor systems," *IEEE Trans. Comput.*, vol. 62, no. 10, pp. 2026–2040, 2013.

[16] I. Mendez-Diaz, J. Orozco, R. Santos, and P. Zabala, "Energy-aware scheduling mandatory/optional tasks in multicore real-time systems," *Int. Trans. Oper. Res.*, vol. 24, no. 12, pp. 173–198, 2017.

[17] W. Li, F. C. Delicato, and A. Y. Zomaya, "Adaptive energy-efficient scheduling for hierarchical wireless sensor networks," *ACM Trans. Sen. Netw.*, vol. 9, no. 3, pp. 33:1–33:34, 2013.

[18] Y. Tian and E. Ekici, "Cross-layer collaborative in-network processing in multihop wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 6, no. 3, pp. 297–310, 2007.

[19] A. Pathak and V. K. Prasanna, "Energy-efficient task mapping for data-driven sensor network macroprogramming," *IEEE Trans. Comput.*, vol. 59, no. 7, pp. 955–968, 2010.

[20] W. Li, F. C. Delicato, P. F. Pires, Y. C. Lee, A. Y. Zomaya, C. Miceli, and L. Pirmez, "Efficient allocation of resources in multiple heterogeneous wireless sensor networks," *J. Parallel Distrib. Comput.*, vol. 74, no. 1, pp. 1775–1788, 2014.

[21] N. Edalat, C.-K. Tham, and W. Xiao, "An auction-based strategy for distributed task allocation in wireless sensor networks," *Comput. Commun.*, vol. 35, no. 8, pp. 916 – 928, 2012.

[22] W. Zhang, B. Song, and E. Bai, "A trusted real-time scheduling model for wireless sensor networks," *J. Sensors*, vol. 2016, pp. 1–8, 2016.

[23] L. Mo and A. Kritikakou, "Mapping imprecise computation tasks on cyber-physical systems," *Peer-to-Peer Netw. Appl.*, vol. 2019, pp. 1–15, 2019.

[24] J. F. Benders, "Partitioning procedures for solving mixed-variables programming problems," *Numer. Math.*, vol. 4, no. 1, pp. 238–252, 1962.

[25] S. Boyd and J. Mattingley, "Branch and bound methods," *Notes for EE364b, Stanford University*, pp. 1–11, 2007.

[26] R. Vedantham, Z. Zhuang, and R. Sivakumar, "Hazard avoidance in wireless sensor and actor networks," *Comput. Commun.*, vol. 29, no. 13, pp. 2578–2598, 2006.

[27] S. Zhang, J. Wu, and S. Lu, "Distributed workload dissemination for makespan minimization in disruption tolerant networks," *IEEE Trans. Mobile Comput.*, vol. 15, no. 7, pp. 1661–1673, 2016.

[28] H. Lee and A. Keshavarzian, "Towards energy-optimal and reliable data collection via collision-free scheduling in wireless sensor networks," in

Proc. IEEE Conference on Computer Communications, 2008, pp. 2029–2037.

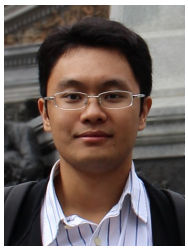
[29] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge University, 2004.

[30] M. Fischetti, F. Glover, and A. Lodi, "The feasibility pump," *Math. Program.*, vol. 104, no. 1, pp. 91–104, 2005.

[31] J. Ramkumar and M. Tulika, "Temperature aware task sequencing and voltage scaling," in *Proc. IEEE International Conference On Computer Aided Design*, 2008, pp. 618–623.

[32] S. Albert, "Solving mixed integer linear programs using branch and cut algorithm," Master's thesis, North Carolina State University, 1999.

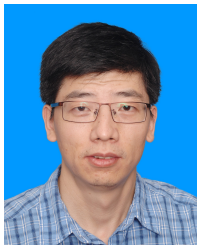
[33] C. Randazzo and H. P. L. Luna, "A comparison of optimal methods for local access uncapacitated network design," *Ann. Oper. Res.*, vol. 106, no. 1, pp. 263–286, 2001.



Lei Mo (S'13–M'17) is currently an associate professor with the School of Automation, Southeast University, Nanjing, China. 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–Grand Est, France. From 2017 to 2019, he was a research fellow with INRIA Rennes–Bretagne Atlantique, 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 embedded systems. He serves as an Associate Editor for KSII Transactions on Internet and Information Systems, International Journal of Ad Hoc and Ubiquitous Computing, 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.

Angeliki Kritikakou is currently an Associate Professor at University of Rennes 1 and IRISA - INRIA Rennes 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.

Olivier Sentieys (M'94) is a Professor at the University of Rennes holding an INRIA Research Chair on Energy-Efficient Computing Systems. He is leading the Cairn team common to INRIA (French research institute dedicated to computational sciences) and IRISA Laboratory. He is also the head of the Computer Architecture department of IRISA. From 2012 to 2017 he was on secondment at INRIA as a Senior Research Director. His research interests are in the area of computer architectures, embedded systems and signal processing, with a focus on system-level design, energy-efficiency, reconfigurable systems, hardware acceleration, approximate computing, and power management of energy harvesting sensor networks. He authored or co-authored more than 250 journal or conference papers, holds 6 patents, and served in the technical committees of several international IEEE/ACM/IFIP conferences.



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 a 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 ICCC 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 AUTOMATICA SINICA.