Decomposed Task Mapping to Maximize QoS in Energy-Constrained Real-Time Multicores

Lei Mo, Angeliki Kritikakou, Olivier Sentieys

Decomposed Task Mapping to Maximize QoS in Energy-Constrained Real-Time Multicores

Lei Mo* Angeliki Kritikakou†* and Olivier Sentieys*†
*INRIA/IRISA, France
†University of Rennes 1, France
Email: lei.mo@inria.fr, {angeliki.kritikakou, olivier.sentieys}@irisa.fr

***Abstract*—Multicore architectures are now widely used in energy-constrained real-time systems, such as energy-harvesting wireless sensor networks. To take advantage of these multicores, there is a strong need to balance system energy, performance and Quality-of-Service (QoS). The Imprecise Computation (IC) model splits a task into mandatory and optional parts allowing to tradeoff QoS. The problem of mapping, i.e. allocating and scheduling, IC-tasks to a set of processors to maximize system QoS under real-time and energy constraints can be formulated as a Mixed Integer Linear Programming (MILP) problem. However, state-of-the-art solving techniques either demand high complexity or can only achieve feasible (suboptimal) solutions. In this paper, we develop an effective decomposition-based approach to achieve an optimal solution while reducing computational complexity. It decomposes the original problem into two smaller easier-to-solve problems: a master problem for IC-tasks allocation and a slave problem for IC-tasks scheduling. We also provide comprehensive optimality analysis for the proposed method. Through the simulations, we validate and demonstrate the performance of the proposed method, resulting in an average 55% QoS improvement with regards to published techniques.**

I. Introduction

Multicore architectures have great potential for energy-constrained real-time systems. Energy consumption has become an important design issue due to the increasing need for high computational performance and with the stringent energy constraints of battery powered devices [1], [2]. However, in several application domains, such as multimedia, target tracking and energy harvesting wireless sensor networks [3], [4], approximate results are acceptable as long as the baseline Quality-of-Service (QoS) is satisfied, i.e. the basic tasks are correctly executed in time. The QoS can further be improved by executing additional tasks, if possible.

Such applications can be modeled by Imprecise Computation (IC) task models [3]–[5], where the application tasks are logically decomposed into a mandatory part and an optional part. Both parts must be executed before the deadline, but the optional part can be left incomplete at the cost of reduced quality of results. Usually, the QoS is represented as a linear function of execution cycles allocated to the optional part. The more cycles the optional part executes, the more QoS it generates. Adequately solving the IC-tasks mapping, i.e. allocation and scheduling, on multicores can maximize QoS under real-time and energy constraints. Task allocation refers to the assignment of tasks to suitable processors, while task scheduling refers to the adjustment of optional part in each task.

A set of existing approaches focus on the energy-aware real-time mapping, where the application's timing constraints are met under minimum energy consumption for various task/processor assignment schemes. However, these approaches do not take system QoS into account. Dynamic Voltage Frequency Scaling (DVFS) has been widely used for power and energy optimization [1], [2], [6]–[9]. The tasks mapping problem is usually formulated as a Mixed Non-Linear Programming (MNLP) problem. However, this is not efficient because solving MNLP is time-consuming. The common methods to solve these complex problems are: 1) problem simplification, e.g., the task allocation scheme is fixed and given in advance [2], and each processor has its own specific voltage/frequency [6], 2) problem approximation/relaxation, e.g., the binary integer programming based task allocation problem is relaxed to a convex problem under specific conditions [1], and the mixed integer non-linear programming based task mapping problem is transformed to a mixed integer linear programming problem by linear approximation [7], 3) heuristics [8], and 4) optimization solvers, such as CPLEX [9].

Other existing approaches focus on QoS-aware real-time mapping under energy constraints. Some recent works have comprehensively considered the timing, energy, and QoS factors for optimization. For single processor platform, only IC-tasks scheduling problem should be taken into account [3], [10]. While for multiprocessor scenarios, Yu *et al.* [5] studied the IC-tasks scheduling problem under the given task-processor assignment. Mendez-Diaz *et al.* [11] proposed a dynamic voltage/optional assignment scheme. However, the research of IC-tasks allocation and scheduling joint-design is rare. An exception is the work in [4]. Most of the above methods employ heuristics to find near-optimal solutions. Although heuristics can provide feasible solutions in a short amount of time, they do not provide bounds on solution quality, and are sensitive to changes in the problem structures and parameters [12].

To derive an optimal solution, the IC-tasks allocation and scheduling problems should be jointly addressed. To achieve this, the following questions have to be answered during the system design: 1) *how to find a proper way to formulate the problem of IC-tasks allocation and scheduling such that the system QoS is maximal while guaranteeing the deadlines and the energy budgets?* and 2) *Is there a way to achieve optimal solution while avoiding high computational complexity?*

This paper answers these questions. We first formulate the IC-tasks allocation and scheduling joint-design problem as a Mixed Integer Linear Programming (MILP). The representation of the core problem in an MILP form has a large impact

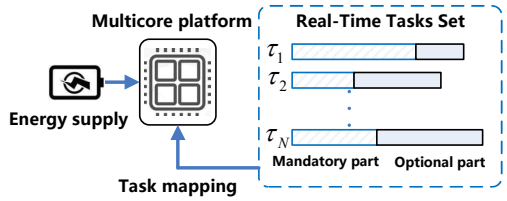


Fig. 1. System architecture and task mapping.

on the solution time required [13], [14]. Different from the previous work, we propose a decomposition-based method to balance solution quality and computational complexity. The original problem is decomposed into two smaller easier-to-solve problems, and through limited iterations between the subproblems, we find the optimal solution. Finally, we provide a comprehensive optimality analysis for the proposed method.

The remainder of this paper is organized as follows: Section II introduces the problem formulation and presents the optimization model. The solution methodology is developed in Section III, and Section IV presents the simulation results. Finally, Section V summarizes the conclusions and future research directions.

II. System Model and Problem Formulation

The typical multicore system we considered in this paper is shown in Fig. 1. In the following, we first present the system model considered in this paper. Then, we formulate the IC-tasks allocation and scheduling joint-design problem as a MILP which takes system QoS, task execution time and energy utilization into account. For better readability of the paper, all the proofs are presented in Appendix.

A. System model

1) Task model: Consider a task set consisting of N independent real-time tasks $\{\tau_1, \ldots, \tau_N\}$. Task τ_j $(1 \le j \le N)$ can be decomposed into a mandatory part and an optional part with M_j and o_j cycles, respectively. All the tasks should be executed with a common deadline D, which is also the scheduling horizon. Under the IC model, each task τ_j must execute mandatorily M_j cycles to generate an acceptable result, and optionally o_j cycles to refine and improve the result of the mandatory part. The optional part o_j should not exceed its upper bound O_j cycles. M_j and O_j are measured in Worst Case Execution Cycles (WCEC). Moreover, a factor $\mu_j \in [0, 1]$ is introduced to describe the activity of task τ_j [4]. Since tasks are assumed to be heterogeneous, different tasks require different power consumptions on the same processor, even when executing at the same speed and temperature.

2) Energy model: The system consists of M processors $\{\theta_1, \ldots, \theta_M\}$, where processor θ_i $(1 \le i \le M)$ is characterized by a given voltage/frequency pair (v_i, f_i). The power consumption of multicore can be modeled as the sum of static power P_s and dynamic power P_d [2]. Specifically, if task τ_j is assigned to processor θ_i, the overall power consumption is

$$P_c(\theta_i, \tau_j) = P_{s,i} + C_i^e v_i^2 f_i \mu_j,\ 1 \le i \le M,\ 1 \le j \le N, \quad (1)$$

where $P_{s,i}$ and C_i^e are the static power and effective switching capacitance of processor θ_i, respectively.

B. Problem Formulation

Taking the available energy E_s during the scheduling horizon D into account, the system operation can be divided into three states: 1) *Low*: the supplied energy E_s is insufficient to execute all the mandatory parts $\{M_1, \ldots, M_N\}$, 2) *High*: the supplied energy E_s is sufficient to execute all the mandatory and optional parts $\{M_1 + O_1, \ldots, M_N + O_N\}$, and 3) *Medium*: all the mandatory parts are ensured to finish, while not all the optional parts have enough energy to complete their executions. We concentrate on system in medium energy state, and design an IC-tasks allocation and scheduling scheme to maximize QoS.

Constraints

To formulate IC-tasks allocation problem, we introduce an $M \times N$ binary matrix $\boldsymbol{S} = \{s_{ij}\}$. If $s_{ij} = 1$, task τ_j is assigned to processor θ_i, otherwise, $s_{ij} = 0$. Note that each task is executed on one processor. This requirement is represented by the following constraints for the task allocation.

$$\sum_{i=1}^{M} s_{ij} = 1,\ 1 \le j \le N. \quad (2)$$

To formulate IC-tasks scheduling problem, we introduce an $M \times N$ matrix $\boldsymbol{T} = \{t_{ij}\}$, where t_{ij} represents the execution time of optional part o_j on processor θ_i. Note that IC-tasks allocation is made at task level (i.e., a task starts its execution in a certain processor and finishes its execution in the same processor), and the optional part o_j has an upper bound O_j cycles. These requirements are given as

$$0 \le t_{ij} \le s_{ij} \frac{O_j}{f_i},\ 1 \le i \le M,\ 1 \le j \le N. \quad (3)$$

Since all the tasks should be executed within the deadline D, we have

$$\sum_{i=1}^{M} \sum_{j=1}^{N} \left(s_{ij} \frac{M_j}{f_i} + t_{ij} \right) \le D. \quad (4)$$

Based on the energy model (1), the total energy consumed by the system during the scheduling horizon D is bounded by the following constraint

$$\sum_{i=1}^{M} \left[P_{s,i} D + \sum_{j=1}^{N} C_i^e v_i^2 f_i \mu_j \left(s_{ij} \frac{M_j}{f_i} + t_{ij} \right) \right] \le E_s, \quad (5)$$

where $s_{ij} \frac{M_j}{f_i} + t_{ij}$ is the execution time of task τ_j on processor θ_i.

Objective function

The system QoS highly depends on the optional parts of the tasks. Usually, the more optional cycles executed, the higher QoS of the tasks. Hence, we define a QoS function $\sum_{i=1}^{M} \sum_{j=1}^{N} t_{ij}$ for the system, and our aim is to maximize it (or to minimize its negative).

Summarizing the objective and all the constraints mentioned above, the Primal Problem (PP) is formulated as

Primal Problem

$$\min_{\boldsymbol{S}, \boldsymbol{T}}\ Q(\boldsymbol{T}) = -\sum_{i=1}^{M} \sum_{j=1}^{N} t_{ij} \quad (6)$$

$$\text{s.t.}\ \begin{cases} (2), (3), (4), (5), \\ s_{ij} \in \{0, 1\}, 0 \le t_{ij} \le \frac{O_j}{f_i},\ 1 \le i \le M,\ 1 \le j \le N. \end{cases}$$

Since integer and continuous variables are coupled with each other linearly, the PP (6) is an MILP problem.

III. Decomposition-based solution

For the PP (6), finding an optimal IC-tasks allocation scheme is the most important step. If integer variable $\boldsymbol{S}$ is determined, the PP (6) will reduce to a Linear Programming (LP), which has a simpler structure, and is much easier to solve. Benders decomposition is an efficient method to solve a certain class of Mixed Integer Programming (MIP) [15]. The basic idea is decomposing the PP (6) into two smaller easier-to-solve problems called Master Problem (MP) and Slave Problem (SP).

The MP accounts for all the integer variables and the corresponding part of the objective function and the constraints of the PP. It also includes the information regarding the SP part of the PP via a set of constraints called *Benders cuts*. On the other hand, the SP includes all the continuous variables and the associated constraints of the PP. Solving the SP provides the information regarding the SP part of the PP, and this information is considered in the MP via the Benders cuts.

A. Benders Decomposition Based Approach

In each iteration of Benders decomposition, the MP and the SP are formulated as

Master Problem

$$Q_l = \min_{\boldsymbol{S},\hat{Q}} \ \hat{Q} \quad \text{s.t.} \begin{cases} (2), \\ \text{feasibility constraint (16)}, \\ \text{infeasibility constraint (17)}. \end{cases} \tag{7}$$

Slave Problem

$$Q_u = \min_{\boldsymbol{T}} \ Q(\boldsymbol{T}) \quad \text{s.t.} \ \ (3), (4), (5) \text{ under given } \textit{S(l)}, \tag{8}$$

where Q_l and Q_u are the lower and upper bounds of Q^* (the optimal objective function value of the PP (6)), respectively, and $\boldsymbol{S}(l)$ is the solution of the MP (7) at the iteration l. Feasibility constraint (16) and infeasibility constraint (17) serve as Benders cuts to narrow the feasible region of integer variable $\boldsymbol{S}$. Since the objective function of the PP (6) only contains the continuous variable, we introduce an auxiliary variable $\hat{Q}$ for the MP (7). $\hat{Q}$ has the same physical meaning as the objective function of the PP (6), and it is connected with $\boldsymbol{S}$ through the feasibility constraints (see Appendix C).

Lemma 3.1: The SP (8) is convex.

Proof: See Appendix A for the proof. ∎

At each iteration, the solution of the MP (7) gives a tentative IC-tasks allocation scheme. Note that the MP (7) only contains the IC-tasks allocation information, compared with the PP (6), the IC-tasks scheduling related constraints are relaxed, thus solving the MP (7) yields a lower bound Q_l. On the other hand, for the SP (8), $\boldsymbol{S}(l)$ may be just a feasible solution (not optimal yet). Hence, solving the SP (8) yields an upper bound Q_u. In each iteration of the algorithm, a new Benders cut (feasibility/infeasibility constraint) is added into the MP (7) to reduce the gap between Q_l and Q_u. Note that Q^* lies between Q_l and Q_u. By iterating the MP (7) and SP (8), Q^* is found. Hence, the key issue is how to derive proper feasibility and infeasibility constraints, which are highly related to the problem structure.

At the initial iteration $l = 0$, the feasibility and infeasibility constraints are set to null. The lower and upper bounds are set to $Q_l = -\infty$ and $Q_u = \infty$, respectively. The initial solution of the MP (7), i.e., $\boldsymbol{S}(0)$, can be given arbitrarily, as long as it satisfies integer constraint (2).

B. Slave problem and Its Dual

In this paper, rather than solving the SP (8) directly, we solve its *dual problem*. This is because 1) the SP (8) is convex, the optimal objective function values of SP (8) and its dual problem are equivalent due to the strong duality [16], and 2) we can construct the feasibility constraint (16) and infeasibility constraint (17) based on the solution of the dual slave problem. In the following, we will explain how to formulate and solve the dual slave problem.

If $\boldsymbol{S}(l)$ is given, by introducing positive Lagrange multipliers $\boldsymbol{\alpha}(l) = \{\alpha_{ij}(l)\}$, $\beta(l)$ and $\gamma(l)$ to the constraints (3), (4) and (5), respectively, the Lagrangian is

$$\begin{aligned} &\mathcal{L}_1(\boldsymbol{T}(l), \boldsymbol{\alpha}(l), \beta(l), \gamma(l)) \\ &= -\sum_{i=1}^{M}\sum_{j=1}^{N} t_{ij}(l) + \sum_{i=1}^{M}\sum_{j=1}^{N} \alpha_{ij}(l)\left(t_{ij}(l) - s_{ij}(l)\frac{O_j}{f_i}\right) \\ &\quad + \beta(l)\left[\sum_{i=1}^{M}\sum_{j=1}^{N}\left(s_{ij}(l)\frac{M_j}{f_i} + t_{ij}(l)\right) - D\right] + \gamma(l)\cdot \\ &\left\{\sum_{i=1}^{M}\left[P_{s,i}D + C_i^e v_i^2 \sum_{j=1}^{N}\mu_j\left(s_{ij}(l)M_j + t_{ij}(l)f_i\right)\right] - E_s\right\}. \end{aligned} \tag{9}$$

Defining the dual function $\mathcal{R}(\boldsymbol{\alpha}(l), \beta(l), \gamma(l))$ as the minimum value of $\mathcal{L}_1(\boldsymbol{T}(l), \boldsymbol{\alpha}(l), \beta(l), \gamma(l))$ over $\boldsymbol{T}(l)$ [16], i.e., for $\boldsymbol{\alpha}(l)$, $\beta(l)$ and $\gamma(l)$, we have

$$\begin{aligned} &\mathcal{R}(\boldsymbol{\alpha}(l), \beta(l), \gamma(l)) \\ &= \min_{\boldsymbol{T}(l)} \left\{\sum_{i=1}^{M}\sum_{j=1}^{N}\left(\alpha_{ij}(l) + \beta(l) + \gamma(l)C_i^e v_i^2 f_i \mu_j - 1\right) t_{ij}(l)\right. \\ &\quad + \sum_{i=1}^{M}\sum_{j=1}^{N} s_{ij}(l)\left(\beta(l)\frac{M_j}{f_i} - \alpha_{ij}(l)\frac{O_j}{f_i}\right) - \beta(l)D \\ &\quad \left. + \gamma(l)\left[\sum_{i=1}^{M}\left(P_{s,i}D + C_i^e v_i^2 \sum_{j=1}^{N}\mu_j s_{ij}(l)M_j\right) - E_s\right]\right\}. \end{aligned} \tag{10}$$

Hence, the dual problem associated with the SP (8) is

Dual Slave Problem

$$\max_{\boldsymbol{\alpha}(l),\beta(l),\gamma(l)} \mathcal{R}(\boldsymbol{\alpha}(l), \beta(l), \gamma(l)). \tag{11}$$

Based on the structure of the DSP (11), we design a two-layer subgradient-based algorithm to solve this problem. For simplicity and generality, we remove Benders iteration counter l from $\boldsymbol{\alpha}(l)$, $\beta(l)$, $\gamma(l)$ and $\boldsymbol{T}(l)$, and introduce indexes m and

k to count outer-layer and inner-layer iterations, respectively. The inner-layer iteration aims to update $\boldsymbol{T}$ under the given $\boldsymbol{\alpha}$, β and γ, while the outer-layer iteration aims to update $\boldsymbol{\alpha}$, β and γ under the given $\boldsymbol{T}$.

1) Inner-layer Iteration: Assume current outer-layer iteration is m. Since $0 \leq t_{ij} \leq \frac{O_j}{f_i}$, based on the update result of previous outer-layer iteration, i.e., $\boldsymbol{\alpha}(m)$, $\beta(m)$ and $\gamma(m)$, t_{ij} is iteratively updated by

$$t_{ij}(m,k+1) = \left[t_{ij}(m,k) - \delta \frac{\partial \mathcal{L}_1(\boldsymbol{T}, \boldsymbol{\alpha}, \beta, \gamma)}{\partial t_{ij}(m,k)} \right]_0^{\frac{O_j}{f_i}}$$
$$= \left[t_{ij}(m,k) + \delta \Big(1 - \alpha_{ij}(m) - \beta(m) - \gamma(m) C_i^e v_i^2 f_i \mu_j \Big) \right]_0^{\frac{O_j}{f_i}}, \quad 1 \leq i \leq M,\ 1 \leq j \leq N. \quad (12)$$

where $[x]_a^b = \max\{a, \min\{x, b\}\}$ and δ is a positive small step-size.

Since $\mathcal{L}_1(\boldsymbol{T}, \boldsymbol{\alpha}, \beta, \gamma)$ is a convex function with respect to $\boldsymbol{T}$, the inner-layer iteration can start from an arbitrary initial point $\boldsymbol{T}(m,0)$, and stops when $||\boldsymbol{\theta}(m,k+1) - \boldsymbol{\theta}(m,k)||_2 \leq \epsilon$, where $\boldsymbol{\theta} \triangleq [\boldsymbol{T}'_1, \ldots, \boldsymbol{T}'_N]'$, $\boldsymbol{T}_j$ is the j^{th} column of matrix $\boldsymbol{T}$, and $\epsilon \geq 0$ is a small tolerance. Hence, by iterating (12), we can obtain $\boldsymbol{T}(m)$, i.e., the optimal solution of the problem (10) under the given $\boldsymbol{\alpha}(m)$, $\beta(m)$ and $\gamma(m)$.

2) Outer-layer Iteration: Based on the update results of previous inner-layer and outer-layer iterations, i.e., $\boldsymbol{T}(m)$, $\boldsymbol{\alpha}(m)$, $\beta(m)$ and $\gamma(m)$, the positive Lagrange multipliers α_{ij}, β and γ are iteratively updated by

$$\alpha_{ij}(m+1) = \left[\alpha_{ij}(m) + \delta \frac{\partial \mathcal{R}(\boldsymbol{\alpha}, \beta, \gamma)}{\partial \alpha_{ij}(m)} \right]^+ = \left[\alpha_{ij}(m) + \delta \left(t_{ij}(m) - s_{ij}(l) \frac{O_j}{f_i} \right) \right]^+, \quad 1 \leq i \leq M,\ 1 \leq j \leq N, \quad (13)$$

$$\beta(m+1) = \left[\beta(m) + \delta \frac{\partial \mathcal{R}(\boldsymbol{\alpha}, \beta, \gamma)}{\partial \beta(m)} \right]^+ = \left[\beta(m) + \delta \left[\sum_{i=1}^{M} \sum_{j=1}^{N} \left(s_{ij}(l) \frac{M_j}{f_i} + t_{ij}(m) \right) - D \right] \right]^+, \quad (14)$$

$$\gamma(m+1) = \left[\gamma(m) + \delta \frac{\partial \mathcal{R}(\boldsymbol{\alpha}, \beta, \gamma)}{\partial \gamma(m)} \right]^+ = \left[\gamma(m) + \delta \left\{ \sum_{i=1}^{M} \left[P_{s,i} D + C_i^e v_i^2 \sum_{j=1}^{N} \mu_j (s_{ij}(l) M_j + t_{ij}(m) f_i) \right] - E_s \right\} \right]^+, \quad (15)$$

where $[x]^+ = \max\{0, x\}$.

Since $\mathcal{R}(\boldsymbol{\alpha}, \beta, \gamma)$ is a concave function with respect to $\boldsymbol{\alpha}$, β and γ, the outer-layer iteration can start from the arbitrary initial points $\boldsymbol{\alpha}(0)$, $\beta(0)$ and $\gamma(0)$, and stops when $||\boldsymbol{\lambda}(m+1) - \boldsymbol{\lambda}(m)||_2 \leq \epsilon$, where $\boldsymbol{\lambda} \triangleq [\boldsymbol{\alpha}'_1, \ldots, \boldsymbol{\alpha}'_N, \beta, \gamma]'$, and $\boldsymbol{\alpha}_j$ is the j^{th} column of matrix $\boldsymbol{\alpha}$. Note that $t_{ij}(m,k)$, $\alpha_{ij}(m)$, $\beta(m)$ and $\gamma(m)$ are updated individually, the DSP (11) is solved in a distributed manner.

Theorem 3.1: With the iterations between outer-layer and inner-layer, $\boldsymbol{\alpha}$, β and γ statistically converges to their optimal values $\boldsymbol{\alpha}^*$, β^* and γ^* when δ is a small enough value.

Proof: See Appendix B for the proof. ■

C. Master Problem and Benders Cut

The iteration counter l increases. $\mathcal{A}$ and $\mathcal{B}$ are defined as the sets of iterations in which the DSP (11) has the bounded and unbounded solutions, respectively. According to the solution of the DSP (11), a new feasibility or infeasibility constraint (Benders cut) is added into the MP (7) at iteration $l+1$.

1) Case 1: If the DSP (11) is *infeasible*, the SP (8) has an unbounded solution. Hence, the PP (6) has no feasible solution.

2) Case 2: If the DSP (11) has a *bounded solution*, e.g., $\boldsymbol{\alpha}(l)$, $\beta(l)$ and $\gamma(l)$. Due to the duality, the SP (8) is feasible. Hence, iteration l should be added into $\mathcal{A}$, i.e., $\mathcal{A} \leftarrow \{l\} \cup \mathcal{A}$. Substituting $\boldsymbol{\alpha}(l)$, $\beta(l)$ and $\gamma(l)$ into the DSP (11), we can obtain $Q_u(l)$. Since $\boldsymbol{S}(l)$ is a feasible solution (not optimal) of the PP (6), a better solution may exist. Hence, the upper bound of Q^* at iteration l is updated by $Q_u = \min\{Q_u, Q_u(l)\}$. Moreover, to raise the lower bound Q_l, based on the bounded solution, a new

Feasibility Constraint

$$\hat{Q} \geq \sum_{i=1}^{M} \sum_{j=1}^{N} s_{ij}(l+1) \left(\beta(l) \frac{M_j}{f_i} - \alpha_{ij}(l) \frac{O_j}{f_i} \right) - \beta(l) D + \gamma(l) \left[\sum_{i=1}^{M} \left(P_{s,i} D + C_i^e v_i^2 \sum_{j=1}^{N} \mu_j s_{ij}(l+1) M_j \right) - E_s \right] \quad (16)$$

is generated and added into the MP (7) at iteration $l+1$.

3) Case 3: If the DSP (11) has an *unbounded solution*, due to the duality, the SP (8) has no feasible solution under the given $\boldsymbol{S}(l)$. Hence, iteration l should be added into $\mathcal{B}$, i.e., $\mathcal{B} \leftarrow \{l\} \cup \mathcal{B}$. To obtain the direction of the unbounded solution, we construct a Feasibility Check Problem (FCP) (31) and solve its dual problem. Based on the solution of the Dual Feasibility Check Problem (DFCP) (33), i.e., $\hat{\boldsymbol{\alpha}}(l)$, $\hat{\beta}(l)$ and $\hat{\gamma}(l)$, a new

Infeasibility Constraint

$$0 \geq \sum_{i=1}^{M} \sum_{j=1}^{N} s_{ij}(l+1) \left(\hat{\beta}(l) \frac{M_j}{f_i} - \hat{\alpha}_{ij}(l) \frac{O_j}{f_i} \right) - \hat{\beta}(l) D + \hat{\gamma}(l) \left[\sum_{i=1}^{M} \left(P_{s,i} D + C_i^e v_i^2 \sum_{j=1}^{N} \mu_j s_{ij}(l+1) M_j \right) - E_s \right] \quad (17)$$

is generated and added into the MP (7) at iteration $l+1$ to avoid selecting those infeasible integers again.

When the MP (7) is solved, the above iteration is repeated, and the iteration stops until $|Q_u - Q_l| \leq \varepsilon$. The algorithm convergence is based on *Theorem* 3.2.

Theorem 3.2: At each iteration with feasibility constraint (16) or infeasibility constraint (17) added into the MP (7), the solution converges.

Proof: See Appendix C for the proof. ■

IV. Simulation Results

The simulations are based on a 2×3 multicore architecture, i.e., $M = 6$. The processor model is built on 65 nm technology [4], and the processor parameters are shown in Table I. The task activity factor μ_j is randomly distributed in $[0.4, 1]$ to demonstrate the characteristics of heterogeneous tasks. The WCEC of tasks are assumed to be in the range $[4 \times 10^7, 6 \times 10^8]$. IC-task τ_j is generated by randomly picking two WCECs from this range, one for the mandatory part M_i and one for the maximum optional part O_i. Moreover, we assume $D = \sum_{j=1}^{N} \left(\min_i \frac{M_j + O_j}{f_i} \right)$ and $E_s = \eta \cdot E_h$, where $\min_i \frac{M_j + O_j}{f_i}$ is the minimum execution time of task $M_j + O_j$ on processor $\{\theta_1, \ldots, \theta_M\}$, $E_h = \sum_{i=1}^{M} (P_{s,i} D) + \sum_{j=1}^{N} \left(\min_i \frac{M_j + O_j}{f_i} P_{d,i} \right)$ is the minimum energy required to execute the tasks $\{M_1 + O_1, \ldots, M_N + O_N\}$, and $\eta \in [0, 1]$ is an energy efficiency factor. The simulations are performed on a laptop with quad-core 2.5 GHZ Intel i7 processor and 16 GB RAM, and the algorithms are implemented in Matlab 2016a.

v (V)	0.00	0.85	0.90	0.95	1.00	1.05	1.10
f (GHZ)	0.0	0.8010	0.8291	0.8553	0.8797	0.9027	1.0

TABLE I
Processor parameters

We present the following evaluation results: 1) the QoS, the energy consumption, the convergence and the computational complexity of the proposed Optimal Decomposition method (OD) with that of Adaptive Task Allocation [4] (ATA) – a two-step heuristic method – for the PP (6), and 2) the QoS and the energy consumption of the OD solving another IC-task mapping problems: a) Minimize Energy (ME) (18), and b) Work-Load Balance (WLB) (19).

A. OD-ATA Comparison

The QoS and energy consumption using OD and ATA to solve the PP (6) under different supplied energy E_s and task number N are shown in Fig. 2. Fig. 2(a) demonstrates that the proposed method achieves higher QoS (55% on average) than that of ATA, while both of them consume the same energy, i.e., the supplied energy E_s, as shown in Fig. 2(b).

Fig. 3(a) shows the algorithm convergence, with task number $N = 40$ and energy efficiency factor $\eta = 0.8$. By introducing the feasibility and infeasibility constraints into the MP (7) during the iterations 1-8 and 9, respectively, the upper and lower bounds quickly converge to the optimal value Q^*. Fig. 3(b) compares the computing time required by OD and ATA under different task number N, with $\eta = 0.8$. From it we can see that the computing time of OD increases linearly with task number N, while the computing time of ATA is almost unchanged with task number N. With task number N increases, more constraints will be added into the PP (6). Hence, more iterations are involved in OD to search the optimal solution. However, even if the computational complexity increases with task number N, the computing time of OD still within an acceptable bound.

B. ME and WLB IC-tasks mapping problems

The problems under study are defined as follows:

–Minimize Energy (ME)

$$\min_{\boldsymbol{S},\boldsymbol{T}} \quad \sum_{i=1}^{M} \left[P_{sta,i} D + C_i^{eff} v_i^2 \sum_{j=1}^{N} \mu_j \left(s_{ij} M_j + t_{ij} f_i \right) \right] \tag{18}$$
$$\text{s.t.} \quad (2), (3), (4).$$

–Work-Load Balance (WLB)

$$\min_{\boldsymbol{S},\boldsymbol{T},U} \quad \kappa \left(\sum_{i=1}^{M} \sum_{j=1}^{N} t_{ij} \right) + (1 - \kappa) U \tag{19}$$
$$\text{s.t.} \quad \begin{cases} U \geq \sum_{j=1}^{N} \left(s_{ij} \frac{M_j}{f_i} + t_{ij} \right), \ 1 \leq i \leq M, \\ (2), (3), (4), (5). \end{cases}$$

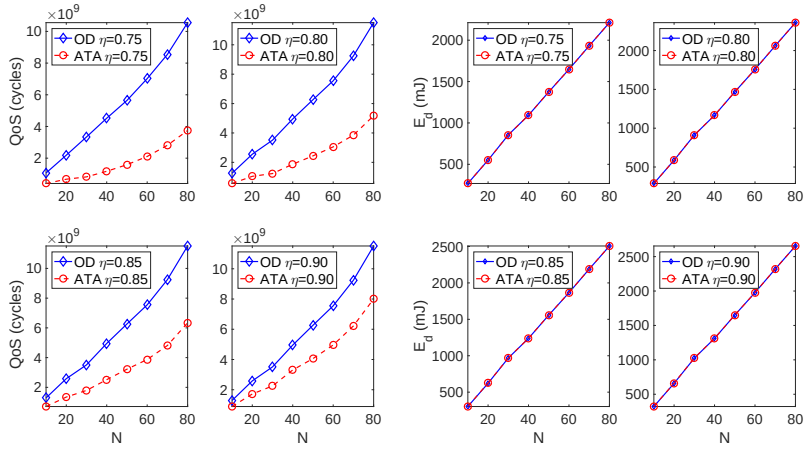
where κ is a balance factor, and U is the maximum processor work-load.

Fig. 4(a) shows the QoS achieved by using OD to solve the ME (18) and the PP (6) under different supplied energy E_s and task number N. We observe that following the solution of the ME (18) to perform tasks mapping, the QoS is always 0. This is because the mandatory part of each task is given and fixed. Hence, the smaller optional part is, the lower energy is consumed for task execution. As shown in Fig. 4(b), although the consumed energy of the PP (6) is 35% higher than that of the ME (18) on average, the consumed energy of the PP (6) is always equal to E_s. Hence, using the PP (6) to perform tasks mapping can provide a better balance between QoS-enhancing and energy-utilizing.

The QoS and maximum processor work-load of the WLB (19) under different supplied energy E_s and balance factor κ are shown in Fig. 5, with task number $N = 40$. Fig. 5(a) and Fig. 5(b) show that QoS and U increase with κ and η. To be specific, $\sum_{i=1}^{M} \sum_{j=1}^{N} t_{ij} = 0$ and $U = 0$ if $\kappa = 0$, and $\sum_{i=1}^{M} \sum_{j=1}^{N} t_{ij}$ and U are maximum if $\kappa = 1$. Usually, under the given supplied energy E_s, a smaller U represents a better work-load balance between the processors. Here, the objectives of maximizing QoS and balancing work-load are contradicted with each other. From energy model (5), we can see that to maximize the total optional parts $\sum_{i=1}^{M} \sum_{j=1}^{N} t_{ij}$ under the given supplied energy E_s, the tasks should be assigned to the processors with small power dissipation factor $C_i^e v_i^2$, which means these processors have higher work-load than the others.

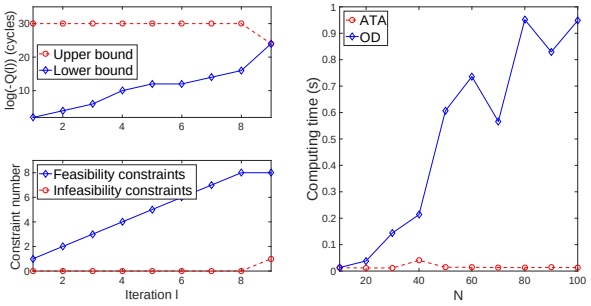
V. Conclusion

In this paper, we have studied the joint-design problem of IC-tasks allocation and scheduling on multicore platforms with the goal of maximizing the QoS without violating the real-time and energy constraints. Then, we built an MILP model for global optimization. To reduce the computational complexity and find an optimal solution, we proposed a decomposition-based method to solve MILP and analyzed the algorithm convergence. Finally, we compared the QoS and energy consumption with several most closely related work on randomly generated tasks. The simulation results showed the effectiveness of our proposed algorithm, which outperforms other algorithms with respect to QoS-enhancing and energy-utilizing in all cases. In the future, we plan to extend our method to multicore platforms that support local/global DVFS and consider the dependency between the tasks.



(a) QoS. (b) Energy consumption.

Fig. 2. QoS and energy consumption using OD and ATA under different energy supply and task number.



(a) Algorithm convergence. (b) Computing time of OD and ATA under different task number.

Fig. 3. Algorithm convergence and computational complexity.

ACKNOWLEDGMENT

This research is funded by INRIA post-doctoral research fellowship program, and partly sponsored by the NSF of China with Grant 61403340.

APPENDIX A
PROOF OF LEMMA 3.1

Proof: For the SP (8), the previous problem formulation can be reformulated in an abstract manner as follows

$$\min_{\boldsymbol{x}} \ \boldsymbol{a}'\boldsymbol{x} \qquad (20)$$
$$\text{s.t.} \ \begin{cases} \boldsymbol{c}_i'\boldsymbol{x} \le b_i, \ 1 \le i \le \bar{m}, \\ 0 \le x_j \le d_j, \ 1 \le j \le \bar{n}. \end{cases}$$

where $\boldsymbol{x} \in R^{\bar{n}\times 1}$, $\boldsymbol{a} \in R^{\bar{n}\times 1}$, $\boldsymbol{c}_l \in R^{\bar{n}\times 1}$, $\bar{m}$ and $\bar{n}$ are the number of constraints and continuous variables, respectively.

Let $f \triangleq \boldsymbol{a}'\boldsymbol{x}$ and $g_i \triangleq \boldsymbol{c}_i\boldsymbol{x} - b_i$. Since

$$\mathcal{H} = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_{\bar{n}}} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_{\bar{n}} \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_{\bar{n}}^2} \end{bmatrix} = \mathbf{0}_{\bar{n}\times\bar{n}},$$

$$\hat{\mathcal{H}}_i = \begin{bmatrix} \frac{\partial^2 g_i}{\partial x_1^2} & \cdots & \frac{\partial^2 g_i}{\partial x_1 \partial x_{\bar{n}}} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 g_i}{\partial x_{\bar{n}} \partial x_1} & \cdots & \frac{\partial^2 g_i}{\partial x_{\bar{n}}^2} \end{bmatrix} = \mathbf{0}_{\bar{n}\times\bar{n}}, \ 1 \le i \le \bar{m},$$

the Hessian of functions f and g_i are positive semi-definite. Thus, the objective function and the constraints of the SP (8) are convex. According to the definition of convex problem [16], the SP (8) is a convex problem. ■

APPENDIX B
PROOF OF THEOREM 3.1

Proof: Let $w_j(m) \triangleq \frac{\partial \mathcal{R}(\boldsymbol{\lambda}(m))}{\partial \lambda_j(m)}$. We construct a Lyapunov function

$$\mathcal{V}(\boldsymbol{\lambda}(m)) = \sum_{j=1}^{MN+2} \left(\lambda_j^* - \lambda_j(m)\right)^2. \qquad (21)$$

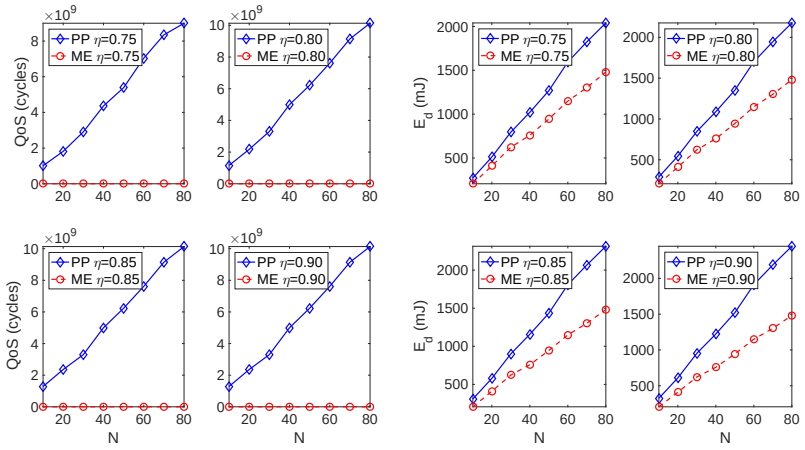
By $\lambda_j(m+1) = [\lambda_j(m) + \delta w_j(m)]^+$, we have $\lambda_j(m+1) \ge 0$ and $\lambda_j(m) \ge 0$. Since $\delta > 0$ and $w_j(m) \le 0$, we get $\lambda_j(m+1) \ge \lambda_j(m) + \delta w_j(m)$. And further, $\left(\lambda_j^* - \lambda_j(m+1)\right)^2 \le \left(\lambda_j^* - \lambda_j(m) - \delta w_j(m)\right)^2$ due to $\lambda_j^* \ge 0$. Hence, we have

$$\begin{aligned} &\mathcal{V}(\boldsymbol{\lambda}(m+1)) \\ =&\sum_{j=1}^{MN+2} \left(\lambda_j^* - \lambda_j(m+1)\right)^2 \le \sum_{j=1}^{MN+2} \left(\lambda_j^* - \lambda_j(m) - \delta w_j(m)\right)^2 \\ =&\mathcal{V}(\boldsymbol{\lambda}(m)) + \sum_{j=1}^{MN+2} \left[2\delta w_j(m)\left(\lambda_j(m) - \lambda_j^*\right) + \delta^2 w_j(m)^2\right] \\ \le&\mathcal{V}(\boldsymbol{\lambda}(m)) + 2\delta\left(\mathcal{R}(\boldsymbol{\lambda}(m)) - \mathcal{R}(\boldsymbol{\lambda}^*)\right) + \sum_{j=1}^{MN+2} \delta^2 w_j(m)^2 \quad (22) \\ \le&\mathcal{V}(\boldsymbol{\lambda}(1)) + \sum_{\tau=1}^{m} \left[2\delta\left(\mathcal{R}(\boldsymbol{\lambda}(\tau)) - \mathcal{R}(\boldsymbol{\lambda}^*)\right) + \sum_{j=1}^{MN+2} \delta^2 w_j(\tau)^2\right]. \quad (23) \end{aligned}$$

Since $\mathcal{R}(\boldsymbol{\lambda})$ is concave, according to the definition of subgradient, we have $\sum_{j=1}^{MN+2} w_j(m)\left(\lambda_j(m) - \lambda_j^*\right) \le \mathcal{R}(\boldsymbol{\lambda}(m)) - \mathcal{R}(\boldsymbol{\lambda}^*)$. Hence, (22) holds.

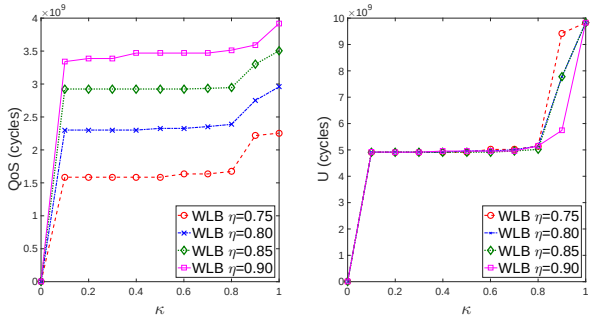
Let $\overline{\boldsymbol{\lambda}}(m) = \frac{1}{m}\sum_{\tau=1}^{m} \boldsymbol{\lambda}(\tau)$. Based on the concavity of $\mathcal{R}(\boldsymbol{\lambda})$ and Jensen's inequality, we get

$$\sum_{\tau=1}^{m} \left(\mathcal{R}(\boldsymbol{\lambda}(\tau)) - \mathcal{R}(\boldsymbol{\lambda}^*)\right) \le m\left(\mathcal{R}(\overline{\boldsymbol{\lambda}}(m)) - \mathcal{R}(\boldsymbol{\lambda}^*)\right). \qquad (24)$$



(a) QoS. (b) Energy consumption.

Fig. 4. QoS and energy consumption of OD solving the PP (6) and the ME (18) under different energy supply and task number.



(a) QoS. (b) Maximum work-load assigned to the processor.

Fig. 5. QoS and maximum processor work-load of OD solving the WLB (19) under different energy supply and balance factor.

Assume $\sum_{j=1}^{MN+2} w_j(\tau)^2 \leq B$ since $w_j(\tau)$ is bounded. Substituting (24) into (23) and noting that $\mathcal{V}(\boldsymbol{\lambda}(m+1)) \geq 0$, we have

$$\mathcal{R}(\boldsymbol{\lambda}^*) - \mathcal{R}(\overline{\boldsymbol{\lambda}}(m)) \leq \frac{\mathcal{V}(\boldsymbol{\lambda}(1)) + m\delta^2 B}{2m\delta}.$$

Hence, $\limsup_{m\to\infty} \left(\mathcal{R}(\boldsymbol{\lambda}^*) - \mathcal{R}(\overline{\boldsymbol{\lambda}}(m))\right) \leq \frac{\delta B}{2}$. According to the definition of statistical convergence [17], given a small enough δ, $\boldsymbol{\lambda}$ statistically converges to $\boldsymbol{\lambda}^*$. ■

Appendix C
Proof of Theorem 3.2

Proof: For the problem (10), since $t_{ij}(l) \geq 0$, $t_{ij}(l)$ is finite only when $\alpha_{ij}(l) + \beta(l) + \gamma(l) C_i^e v_i^2 f_i \mu_j - 1 \geq 0$. Hence, the DSP (11) can be rewrote as follows:

$$\max_{\boldsymbol{\alpha}(l),\beta(l),\gamma(l)} \sum_{i=1}^{M}\sum_{j=1}^{N} s_{ij}(l)\left(\beta(l)\frac{M_j}{f_i} - \alpha_{ij}(l)\frac{O_j}{f_i}\right) - \beta(l)D + \gamma(l)\left[\sum_{i=1}^{M}\left(P_{sta,i}D + C_i^{eff} v_i^2 \sum_{j=1}^{N}\mu_j s_{ij}(l) M_j\right) - E_{sup}\right]$$

$$\text{s.t.} \begin{cases} \alpha_{ij}(l) + \beta(l) + \gamma(l) C_i^e v_i^2 f_i \mu_j - 1 \geq 0, \\ 1 \leq i \leq M,\ 1 \leq j \leq N, \\ \alpha_{ij}(l) \geq 0, \beta(l) \geq 0, \gamma(l) \geq 0, 1 \leq i \leq M, 1 \leq j \leq N. \end{cases} \tag{25}$$

At iteration l, if the problem (25) has a bounded solution, e.g., $\boldsymbol{\alpha}(l)$, $\beta(l)$ and $\gamma(l)$, we have

$$\min_{\boldsymbol{S}} \left\{ \sum_{i=1}^{M}\sum_{j=1}^{N} s_{ij}\left(\beta(l)\frac{M_j}{f_i} - \alpha_{ij}(l)\frac{O_j}{f_i}\right) - \beta(l)D + \gamma(l)\left[\sum_{i=1}^{M}\left(P_{s,i}D + C_i^e v_i^2 \sum_{j=1}^{N}\mu_j s_{ij} M_j\right) - E_s\right]\right\} \tag{26}$$

$$\leq \sum_{i=1}^{M}\sum_{j=1}^{N} s_{ij}^*\left(\beta(l)\frac{M_j}{f_i} - \alpha_{ij}(l)\frac{O_j}{f_i}\right) - \beta(l)D + \gamma(l)\left[\sum_{i=1}^{M}\left(P_{s,i}D + C_i^e v_i^2 \sum_{j=1}^{N}\mu_j s_{ij}^* M_j\right) - E_s\right] \tag{27}$$

$$\leq \max_{\boldsymbol{\alpha},\beta,\gamma} \sum_{i=1}^{M}\sum_{j=1}^{N} s_{ij}^*\left(\beta\frac{M_j}{f_i} - \alpha_{ij}\frac{O_j}{f_i}\right) - \beta(l)D + \gamma\left[\sum_{i=1}^{M}\left(P_{s,i}D + C_i^e v_i^2 \sum_{j=1}^{N}\mu_j s_{ij}^* M_j\right) - E_s\right]\Bigg\} = Q^*, \tag{28}$$

where (27) holds since under the given $\boldsymbol{\alpha}(l)$, $\beta(l)$ and $\gamma(l)$, $\boldsymbol{S}^*$ may not the optimal solution, and (28) holds since Q^* is obtained by solving problem (25) under the given $\boldsymbol{S}^*$.

From the above inequalities, we can see that (26) is a lower bound of Q^*. To raise the lower bound, an auxiliary variable $\hat{Q}$ is introduced and we have

$$\min_{\boldsymbol{S},\hat{Q}} \ \hat{Q} \tag{29}$$

$$\text{s.t.} \ \ \hat{Q} \geq \sum_{i=1}^{M}\sum_{j=1}^{N} s_{ij}\left(\beta(l)\frac{M_j}{f_i} - \alpha_{ij}(l)\frac{O_j}{f_i}\right) - \beta(l)D + \gamma(l)\left[\sum_{i=1}^{M}\left(P_{s,i}D + C_i^e v_i^2 \sum_{j=1}^{N}\mu_j s_{ij} M_j\right) - E_s\right]. \tag{30}$$

Recall the objective function of the MP (7), we obtain a corresponding feasibility constraint (30). Moreover, (26)-(30) show that auxiliary variable $\hat{Q}$ has the same physical meaning as the objective function of the PP (6).

On the other hand, if the problem (25) has an unbounded solution, that means the given $\boldsymbol{S}(l)$ and the SP (8) conflicted with each other. To find out the direction of the unbounded solution, we construct a Feasibility Check Problem (FCP) (31). For the SP (8), its feasibility is related to the constraints rather than the objective function. This problem may be feasible if positive variables $\boldsymbol{\xi} = \{\xi_{ij}\}$, ρ and ν are introduced to relax its constraints. Based on this idea, the FCP can be formulated as

Feasibility Check Problem

$$\min_{\boldsymbol{T},\boldsymbol{\xi},\rho,\nu} \ \mathcal{F}(\boldsymbol{T},\boldsymbol{\xi},\rho,\nu) = \sum_{i=1}^{M}\sum_{j=1}^{N}\xi_{ij} + \rho + \nu \tag{31}$$

$$\text{s.t.} \begin{cases} 0 \le t_{ij} \le s_{ij}\frac{O_j}{f_i} + \xi_{ij},\ 1 \le i \le M,\ 1 \le j \le N, \\ \sum_{i=1}^{M}\sum_{j=1}^{N}\left(s_{ij}\frac{M_j}{f_i} + t_{ij}\right) \le D + \rho, \\ \sum_{i=1}^{M}\left[P_{s,i}D + C_i^e v_i^2 \sum_{j=1}^{N}\mu_j\left(s_{ij}M_j + t_{ij}f_i\right)\right] \le E_s + \nu. \end{cases}$$

By introducing the Lagrange multipliers $\hat{\boldsymbol{\alpha}}$, $\hat{\beta}$ and $\hat{\gamma}$ to the FCP (31), the Lagrangian is

$$\begin{aligned} \mathcal{L}_2(\boldsymbol{T},\boldsymbol{\xi},\rho,\nu,\hat{\boldsymbol{\alpha}},\hat{\beta},\hat{\gamma}) &= \sum_{i=1}^{M}\sum_{j=1}^{N}\left(\hat{\alpha}_{ij} + \hat{\beta} + \hat{\gamma}C_i^e v_i^2 f_i \mu_j\right) t_{ij} \\ &+ \sum_{i=1}^{M}\sum_{j=1}^{N}\left(1-\hat{\alpha}_{ij}\right)\xi_{ij} + \left(1-\hat{\beta}\right)\rho + \left(1-\hat{\gamma}\right)\nu \\ &+ \sum_{i=1}^{M}\sum_{j=1}^{N} s_{ij}\left(\hat{\beta}\frac{M_j}{f_i} - \hat{\alpha}_{ij}\frac{O_j}{f_i}\right) - \hat{\beta}D \\ &+ \hat{\gamma}\left[\sum_{i=1}^{M}\left(P_{s,i}D + C_i^e v_i^2 \sum_{j=1}^{N}\mu_j s_{ij} M_j\right) - E_s\right]. \end{aligned} \tag{32}$$

Since $t_{ij} \ge 0$, $\xi_{ij} \ge 0$, $\rho \ge 0$ and $\nu \ge 0$, the dual problem associated with the FCP (31) is

Dual Feasibility Check Problem

$$\begin{aligned} \max_{\hat{\boldsymbol{\alpha}},\hat{\beta},\hat{\gamma}} \ \mathcal{P}(\hat{\boldsymbol{\alpha}},\hat{\beta},\hat{\gamma}) &= \sum_{i=1}^{M}\sum_{j=1}^{N} s_{ij}\left(\hat{\beta}\frac{M_j}{f_i} - \hat{\alpha}_{ij}\frac{O_j}{f_i}\right) - \hat{\beta}D \\ &+ \hat{\gamma}\left[\sum_{i=1}^{M}\left(P_{s,i}D + C_i^e v_i^2 \sum_{j=1}^{N}\mu_j s_{ij} M_j\right) - E_s\right] \end{aligned} \tag{33}$$

$$\text{s.t.} \begin{cases} \hat{\alpha}_{ij} + \hat{\beta} + \hat{\gamma}C_i^e v_i^2 f_i \mu_j \ge 0,\ 1 \le i \le M,\ 1 \le j \le N, \\ 1-\hat{\alpha}_{ij} \ge 0, 1-\hat{\beta} \ge 0, 1-\hat{\gamma} \ge 0, 1 \le i \le M, 1 \le j \le N, \\ \hat{\alpha}_{ij} \ge 0, \hat{\beta} \ge 0, \hat{\gamma} \ge 0,\ 1 \le i \le M,\ 1 \le j \le N. \end{cases}$$

Based on the solution of the DFCP (33), i.e., $\hat{\boldsymbol{\alpha}}(l)$, $\hat{\beta}(l)$ and $\hat{\gamma}(l)$, we can construct the direction of the unbounded solution to help us to avoid selecting those infeasible integers again. If the SP (8) exists infeasible constraints, the related relax variables are non-zero, while the others are zero. Hence, we have $\min_{\boldsymbol{T},\boldsymbol{\xi},\rho,\nu}, \mathcal{F}(\boldsymbol{T},\boldsymbol{\xi},\rho,\nu) > 0$. To avoid selecting $\boldsymbol{S}(l)$ again, we should set $\min_{\boldsymbol{T},\boldsymbol{\xi},\rho,\nu}, \mathcal{F}(\boldsymbol{T},\boldsymbol{\xi},\rho,\nu) \le 0$. Since the FCP (33) is an LP, the strong duality is guaranteed, i.e., $\min_{\boldsymbol{T},\boldsymbol{\xi},\rho,\nu}, \mathcal{F}(\boldsymbol{T},\boldsymbol{\xi},\rho,\nu) = \max_{\hat{\boldsymbol{\alpha}},\hat{\beta},\hat{\gamma}} \mathcal{P}(\hat{\boldsymbol{\alpha}},\hat{\beta},\hat{\gamma})$. Hence, we obtain a corresponding infeasibility constraint

$$\begin{aligned} 0 \ge &\sum_{i=1}^{M}\sum_{j=1}^{N} s_{ij}\left(\hat{\beta}(l)\frac{M_j}{f_i} - \hat{\alpha}_{ij}(l)\frac{O_j}{f_i}\right) - \hat{\beta}(l)D \\ &+ \hat{\gamma}(l)\left[\sum_{i=1}^{M}\left(P_{s,i}D + C_i^e v_i^2 \sum_{j=1}^{N}\mu_j s_{ij} M_j\right) - E_s\right]. \end{aligned} \tag{34}$$

Note that in feasibility constraint (30) and infeasibility constraint (34), all the parameters are constant except s_{ij}. If feasibility/infeasibility constraint is added into the MP (7) at iteration $l+1$, $s_{ij} \to s_{ij}(l+1)$. Hence, we have (16) and (17). ■

REFERENCES

[1] 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.

[2] 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.

[3] 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.

[4] J. Zhou, J. Yan, T. Wei, M. Chen, and X. S. Hu, "Energy-adaptive scheduling of imprecise computation tasks for qos optimization in real-time mpsoc systems," in *IEEE DATE*, 2017, pp. 1402–1407.

[5] H. Yu, B. Veeravalli, Y. Ha, and S. Luo, "Dynamic scheduling of imprecise-computation tasks on real-time embedded multiprocessors," in *IEEE CSE*, 2013, pp. 770–777.

[6] Y. Zhang, Y. Wang, and H. Wang, "Energy-efficient task scheduling for dvfs-enabled heterogeneous computing systems using a linear programming approach," in *IEEE IPCCC*, 2016, pp. 1–8.

[7] 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 *IEEE ISCAS*, 2003, pp. 309–312.

[8] P. Zhou and W. Zheng, "An efficient biobjective heuristic for scheduling workflows on heterogeneous dvs-enabled processors," *J. Appl. Math.*, vol. 2014, pp. 1–15, 2014.

[9] Y. Liu, G. Cox, Q. Deng, S. C. Draper, and R. Bianchini, "Fastcap: An efficient and fair algorithm for power capping in many-core systems," in *IEEE ISPASS*, 2016, pp. 57–68.

[10] H. Yu, B. Veeravalli, and Y. Ha, "Dynamic scheduling of imprecise-computation tasks in maximizing qos under energy constraints for embedded systems," in *IEEE ASP-DAC*, 2008, pp. 452–455.

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

[12] T. Chantem, X. S. Hu, and R. P. Dick, "Temperature-aware scheduling and assignment for hard real-time applications on mpsocs," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 19, no. 10, pp. 1884–1897, 2011.

[13] A. Davare, J. Chong, Q. Zhu, D. M. Densmore, and A. L. Sangiovanni-Vincentelli, "Classification, customization, and characterization: Using milp for task allocation and scheduling," EECS Dep., UC Berkeley, Tech. Rep., 2006.

[14] 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.

[15] J. Benders, "Partitioning procedures for solving mixed-variables programming problems," *Comput. Manag. Sci.*, vol. 2, no. 1, pp. 3–19, 2005.

[16] S. Boyd and L. Vandenberghe, "Convex optimization," *Cambridge University Press*, 2004.

[17] L. Chen, S. Low, M. Chiang, and J. Doyle, "Cross-layer congestion control, routing and scheduling design in ad hoc wireless networks," in *IEEE INFOCOM*, 2006.