Journal of Systems Architecture 98 (2019) 79–91

Contents lists available at ScienceDirect

Journal of Systems Architecture

journal homepage: www.elsevier.com/locate/sysarc

Minimizing temperature and energy of real-time applications with precedence constraints on heterogeneous MPSoC systems

Tiantian Li[a,*], Tianyu Zhang[b], Ge Yu[c], Jie Song[d], Jing Fan[a,*]

[a] College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou, Zhejiang 310000 China
[b] School of Computer Science and Engineering, Qingdao University, Qingdao, Shandong 110169 China
[c] School of Computer Science and Engineering, Northeastern University, Shenyang, Liaoning 110169 China
[d] Software College, Northeastern University, Shenyang, Liaoning 110169 China

ARTICLE INFO

Keywords:
Temperature and energy minimization
Thermal-aware task assignment
Dynamic voltage and frequency scaling (DVFS)
Heterogeneous MPSoC system
Real-time tasks with precedence constraints

ABSTRACT

The energy issue of real-time applications with precedence-constrained tasks on heterogeneous systems has been studied recently. With the strikingly increasing power density due to the soaring system integration level, severe thermal issues arise which can in turn further aggravate the energy issues due to the strong temperature/leakage dependency. Any optimization should be insufficient if such dependency is not properly addressed. However, the state-of-the-art approaches either treat leakage power as a constant, or only adopt the dynamic power consumption as the heuristic metric to conduct the optimization, both of which cannot fully explore the optimization room for the two issues. To this end, we design an energy/thermal aware task scheduling approach by taking both the thermal and energy factors into consideration. The optimization is conducted from two aspects: first balance the energy/thermal loads of processors by assigning tasks in an energy/thermal aware heuristic way, and that of tasks by the deduced task-level deadlines; then reduce the waiting time between parallel tasks that share the same successor task. Extensive experiments conducted on real-world applications show that, the proposed approach can reduce more temperature by up to about 12°C (depending on the specific application and related parameters) while keeping a competitive energy consumption compared with the state-of-the-arts.

1. Introduction

Energy and thermal issues of multiprocessor system-on-chip (MPSoC) systems have become critical with the ever-increasing power density [1]. Increase in energy incurs serious technical and economic problems, while increase in temperature may greatly damage the performance, reliability and safety of the systems. Further, due to the positive mutual effects between temperature and power and the dominant position of leakage power in nano-era [2–4], it is necessary to study optimization techniques to alleviate the energy and thermal issues. Research work [1] categorizes the optimization techniques into three kinds: floor planning, memory management, and task scheduling. This work focuses on the task scheduling optimization approach for MPSoC systems considering both power and thermal factors.

An MPSoC system has evolved to be heterogeneous to provide better services. As heterogeneous multiprocessors continue to be scaled up, DAG-based (Directed Acyclic Graph) parallel applications keep increasing in number on heterogeneous MPSoC systems [5]. For heterogeneous MPSoC systems, processors have distinctive power and thermal dissipation factors, which should be both considered to better minimize the system energy consumption and temperature. For DAG-based applications, the optimization is more challenging due to the additional precedence constraints between tasks. However, many existing studies fail to take a full consideration of all these related factors, thus cannot fully exploring the optimization space. Research works [6–9] focus on homogeneous systems; [10–12] aims at independent periodic tasks; [13–15] only consider the heterogeneity of processors while ignoring that of tasks in the sense of dynamic power consumption; [5,13,14] treat leakage power as a constant or simply ignore it, while now leakage power has become non-negligible and its changeability with temperature should also be taken into consideration. With the deep sub-micron technology, any optimization technique should be ineffective if the temperature/leakage dependency is not properly addressed [16]. Further, existing task assignment heuristics like [17] only consider the dynamic power factors while ignoring the critical thermal factors.

Due to the reasons above, this work proposes an energy/thermal aware task scheduling for DAG-based applications on heterogeneous MPSoC systems, with the consideration of both the thermal factors and

* Corresponding author.
E-mail addresses: ttli89@zjut.edu.cn, litiantian_neu@163.com (T. Li), ztylll@126.com (T. Zhang), yuge@mail.neu.edu.cn (G. Yu), songjie@mail.neu.edu.cn (J. Song), fangjing@zjut.edu.cn (J. Fan).

https://doi.org/10.1016/j.sysarc.2019.07.001
Received 21 January 2019; Received in revised form 24 June 2019; Accepted 1 July 2019
Available online 2 July 2019

the heterogeneity of processors and tasks. When temperature-dependent leakage power is considered, the optimization of energy consumption coincides with that of temperature. We first deduce the metric to evaluate the energy/thermal profiles of processors, by which providing guides for the subsequent task scheduling. Then, we conduct the optimization from the aspects of balancing the energy/thermal loads of both processors and tasks. For processors, the balance is achieved by a heuristic task assignment strategy that can result in lower temperature values with small standard deviation; For tasks, the balance is achieved by the task-level deadlines deduced in this paper. Finally, we optimize the system by reducing the waiting time between the parallel tasks with the same successor task.

The main contribution of this work is providing a new sight on minimizing both temperature and energy consumption through balancing the energy/thermal loads of processors and tasks. The main differences between the proposed approach and the state-of the art approaches [5,17] lie in the following three aspects: it also takes the thermal factors of processors into consideration when assigning tasks besides the currently used dynamic power factors; it evenly reserves DVFS optimization room for tasks while guaranteeing the schedulability by leveraging the deadlines calculated for individual tasks; it explores the optimization space existing among the waiting between parallel tasks in a thermal aware way.

The rest of this paper is organized as follows: Section 2 introduces the related work; Section 3 describes the system models, the problem formulation and the preliminaries; Section 4 conducts the theoretical optimization analysis, while Section 5 proposes a two-step energy/thermal aware optimization approach correspondingly; Section 6 validates the superiority of the proposed approach through comparison with the state-of-the-arts; Section 7 summarizes this work and points out the future work.

2. Related work

This work focuses on the task scheduling optimization approach for MPSoC systems considering both power and thermal factors. Researchers who are interested in other optimization techniques such as floor planning [18–20], memory management [21–24] are referred to work [1].

When designing the task scheduling approach, both the heterogeneity of tasks and the changeability of leakage power with temperature should be taken into consideration to better minimize the system energy consumption and temperature. However, many existing research works fail to do this. Ahmed et al. [6,7] studied the thermal utilization of homogeneous MPSoC systems to minimize the system peak temperature, which cannot be directly applied to heterogeneous systems; Colin et al. [13,14] studied the optimal load distribution considering the heterogeneity of processors to minimize the energy consumption of heterogeneous MPSoC systems. However, the power model adopted by [6,7,13,14] does not take the heterogeneity of tasks into consideration, and the leakage power is also treated as a constant in [13,14]. Saha et al. [15] proposed a genetic-based task allocation approach considering the impact of temperature on leakage power but ignored the heterogeneity of tasks either. Li et al. [25] considered the heterogeneity of both processors and tasks to minimize the energy consumption, but ignored the temperature constraint. Awan et al. [26] designed a task scheduling algorithm which first decreases the dynamic energy consumption by an energy-aware task allocation, then reduces the leakage power consumption by using efficient sleep states when possible. However, this work only considers the changeability of leakage power with the used sleep states, while ignoring that of temperature.

Considering both the heterogeneity of tasks and the temperature/leakage dependency, Zhou et al. [10] proposed a two-stage task scheduling, which minimizes the dynamic energy consumption by a heuristic task assignment approach in the first stage, and then minimizes the leakage energy consumption through slack distribution. However, this work fails to fully explore the optimization space in both the two stages. First, the task assignment is designed to minimize only the dynamic energy consumption; Second, the processors are fully utilized with its assigning heuristic, reserving little optimization room for slack distribution; Third, the optimization efficiency of slack distribution is weaker than DVFS, and is usually discounted due to the schedulability constraint. To address these issues, our work takes the total energy consumption not only the dynamic part as the metric to heuristically assign the tasks, and balances the computing loads of processors so as to reserve maximal optimization space for the more efficient DVFS technique to minimize the temperature and energy consumption in the second phase. Further, Zhou et al. [11] proposed a method assigning tasks to processors in a decreasing order by the difference of steady state temperature on different processors to minimize the peak temperature, and splitting the tasks with slacks to further minimize the peak temperature. However, this method is based on the thermal profiles of tasks rather than the processors, which may result in huge difference when applied in practice. To address the issues existed in [10,11], Li et al. [12] proposed a thermal-aware task assignment and speed scaling approach to minimize both the energy consumption and temperature through balancing the thermal loads of both processors and tasks. However, as the other works mentioned above, this work also aims at independent tasks, not suitable for the precedence-constrained tasks in the DAG application. Other works like [27–30] also have shortcomings: [27] qualitatively allocated hot tasks to cores that are near to the heat sink rather than quantitatively estimate the thermal loads of processors; [28] only considered the power balance of processors ignoring the thermal balance while [29] considered the temperature balance but only treat temperature as a constraint rather than minimize it; [30] scaled the voltage/frequency in processor level rather than task level.

Aiming at DAG-based parallel applications, Li [8,9] studied the energy saving approaches in homogeneous systems. Further, for heterogeneous systems, Xie et al. [5] proposed a downward-upward energy consumption minimization method (DUECM), which sufficiently explore the optimization space. However, this work still considers the temperature-dependent leakage power as a constant, which is not applicable in nowadays with the continuously increasing power density. Besides, the heterogeneity of tasks is also ignored. Considering the heterogeneity of tasks, Zhou et al. [17] proposed a two-level scheduling to minimize the energy consumption of processors by DVFS. However, this work does not fully exploit the potentials of the DAG-based application on the heterogeneous MPSoC systems to minimize both the temperature and energy consumption. *First*, the task assignment heuristic is designed to minimize only the dynamic energy consumption, rather than the overall energy consumption or the system temperature; *Second*, it keeps assigning tasks to processors until it is overloaded, which can badly decrease the tasks' schedulability; *Third*, the waiting time between tasks are not explored to further minimize the energy consumption.

Different with existing research works, considering both the changeability of leakage power with temperature and the heterogeneity of tasks, our work first adopts a thermal-aware task assignment heuristic by taking the leakage power into consideration rather than the traditional energy-aware one to sufficiently explore the optimization space of both temperature and energy consumption, second uses the task-level deadline when conducting the optimization work by DVFS technique which has little sacrifice on the schedulability, and third further minimizes the energy consumption and temperature by reducing or eliminating the waiting time between tasks that have the same successor task.

3. System models and preliminaries

Consider a heterogeneous MPSoC system $\mathcal{P}$ consisting of M processors $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_M\}$, with each processor characterized as $\mathcal{P}_k =< C_k^{\text{eff}}, f_{k,\max}, R_k, C_k, \alpha_k, \beta_k >$ (see Table 1). $\mathcal{P}_k$ supports a set of discrete supply voltage and frequency pairs $(v_{k,l}, f_{k,l})$, where $v_{k,1} < v_{k,2} < \cdots < v_{k,x_k}$, $f_{k,1} < f_{k,2} < \cdots < f_{k,x_k}$, and x_k is the scaling level supported.

Table 1
Symbols Notation.

Category	Symbol	Meaning	Unit
Task Related	τ_i	Real-time task τ_i	–
	c_i	Average WCET value of τ_i	ms
	$D(\tau_i)$	Deadline of τ_i	ms
	A_i	Switching activity of τ_i	–
	s_i	Executing speed of τ_i	–
	P_{d}^i	Dynamic power consumption of τ_i	W
	δ_i	Power and thermal dissipation factor of τ_i	–
Processor Related	$\mathcal{P}_k$	Processor $\mathcal{P}_k$	–
	P_k	Power consumption of $\mathcal{P}_k$	W
	C_k^{eff}	Efficient power coefficient of $\mathcal{P}_k$	–
	$f_{k,\max}$	The maximal frequency provided by $\mathcal{P}_k$	GHz
	Θ_k	Temperature of $\mathcal{P}_k$	°C
	R_k	Thermal resistance of $\mathcal{P}_k$	°C/W
	C_k	Thermal capacitance of $\mathcal{P}_k$	J/°C
	Θ_a	Ambient temperature	°C
	Λ_k	Power and thermal dissipation factor of $\mathcal{P}_k$	–
Leakage Power Related	α_k	Temperature related coefficient	W/°C
	β_k	Temperature irrelevant coefficient	W

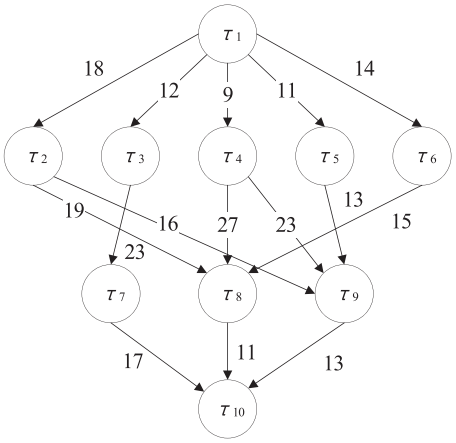


Fig. 1. A classical DAG-based parallel application.

3.1. Application model

A real-time application with precedence-constrained tasks is usually represented by a DAG, in which the nodes represent the tasks and the edges represent the precedence constraints among tasks. For a DAG, $G = (N, E, C)$, N represents a set of nodes with each node representing a task τ_i with different execution times on different processors due to the heterogeneity of the processors; E represents a set of edges with each edge $e_{i,j}$ representing the precedence constraint between task τ_i and τ_j meaning that τ_j cannot start to execute until τ_i completes, and $\omega_{i,j}$ (the weight of $e_{i,j}$) represents the corresponding communication time from τ_i to τ_j; C is a $|N| \times |\mathcal{P}|$ matrix where $c_{i,k}$ denotes the worst-case execution time (WCET) of τ_i on processor $\mathcal{P}_k$ under the maximum frequency $f_{k,\max}$. Denote the sets of immediate predecessor and successor tasks of τ_i as PRED(τ_i) and SUCC(τ_i), respectively. A task without any predecessor is called as entry task, and a task without any successor is called as exit task. We assume the DAG application has only one entry task τ_{entry} and one exit task τ_{exit}. Fig. 1 shows a classical DAG-based real-time application, which has been adopted by numerous studies [5,31–35]. The corresponding WCET values of the 10 tasks on 3 processors are shown in Table 2, and they will change with the scaling of processor frequency as $c_{i,k} f_{k,\max}/f_{k,l_i}$. The power consumption of τ_i not only depends on the processor running speed $s_{i,k}$ ($s_{i,k} = f_{k,l_i}/f_{k,\max}$), but also highly depends upon the circuit activity. Therefore, the task activity factor A_i is introduced to capture this heterogeneity of tasks.

Table 2
The Worst-Case Execution Times (WCET) of tasks in Fig. 1.

	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6	τ_7	τ_8	τ_9	τ_{10}
$\mathcal{P}_1$	14	13	11	13	12	13	7	5	18	21
$\mathcal{P}_2$	16	19	13	8	13	16	15	11	12	7
$\mathcal{P}_3$	9	18	19	17	10	9	11	14	20	16

3.2. Power model

This work considers the state-of-the-art power model for a processor [10], which is modeled as the sum of dynamic power consumption $P_{\mathrm{d}}(t)$ and static power consumption $P_{\mathrm{s}}(t)$:

$$P_{\mathrm{d}}(t) = C^{\mathrm{eff}} \cdot A(t) \cdot s(t)^3, P_{\mathrm{s}}(\Theta(t)) = \alpha\Theta(t) + \beta,$$
$$P(t, \Theta(t)) = P_{\mathrm{d}}(t) + P_{\mathrm{s}}(\Theta(t)), \quad (1)$$

where $P_{\mathrm{d}}(t)$ depends on C^{eff}, $A(t)$ and $s(t)$ (normalized frequency), while $P_{\mathrm{s}}(t)$ depends on $\Theta(t)$ which denotes the processor temperature at moment t.

Here, static power consumption is approximated as the leakage power consumption which consists its main part, and instead of the accurate but complex leakage model given in [36], $P_{\mathrm{s}}(t)$ is modeled as a linear function of temperature, which has been proved accurate enough within typical processor temperature range ($25°C - 120°\mathrm{C}$) under 65 nm manufacturing technology by Liu et al. [37]. The static power model described above has been adopted by many research works [6,7,10,17].

3.3. Thermal model

An MPSoC system consists of multiple processors, and heat will transfer between processors when their thermal loads are not balanced. The heat transfer process can be approximated by Fourier's law [15,38], where the thermal coefficients can be obtained using the RC models [15,37–39].

Denote $R_{k,m}$ as the thermal resistance between two processors $\mathcal{P}_k$ and $\mathcal{P}_m$. If there is no heat transfer between $\mathcal{P}_k$ and $\mathcal{P}_m$, then $R_{k,m} = \infty$. The thermal resistance of $\mathcal{P}_k$ that dissipates heat to the ambient is R_{k}. Let $\Theta_k(t)$ and $P_k(t)$ be the temperature and the power consumption of $\mathcal{P}_k$ at moment t. Then, by Fourier's law, the heat transfer process of $\mathcal{P}_k$ can be described as:

$$\frac{\mathrm{d}\Theta_k(t)}{\mathrm{d}t} = \frac{P_k(t)}{C_k} - \frac{\Theta_k(t) - \Theta_{\mathrm{a}}}{R_k C_k} - \sum_{\mathcal{P}_m \in \mathcal{P}} \frac{\Theta_k(t) - \Theta_m(t)}{R_{km} C_k}. \quad (2)$$

When the thermal loads of processors are balanced, there will be no heat transfer between processors. Then, the thermal model described

by Eq. (2) can be transformed as:

$$\frac{d\Theta_k(t)}{dt} = \frac{P_k(t)}{C_k} - \frac{\Theta_k(t) - \Theta_a}{R_k C_k}. \tag{3}$$

Further, through mathematical transformation and deduction, the temperature function can be derived as follows:

$$\begin{aligned} \theta_k(t) &= C_k\left(\Theta_k(t) - \frac{R_k\beta_k + \Theta_a}{1 - R_k\alpha_k}\right), \\ \theta_k(t) &= \int_0^t P_d^k(u)e^{\lambda_k(u-t)}du + \theta_k(0)e^{-\lambda_k t}, \\ \lambda_k &= \frac{1}{R_k C_k} - \frac{\alpha_k}{C_k}. \end{aligned} \tag{4}$$

where $\theta_k(t)$ is an algebraic transformation of $\Theta_k(t)$.

3.4. Problem formulation

For the task set $\Gamma = \{\tau_1, \ldots, \tau_n\}$ in the given DAG-based application and heterogeneous processor set $\mathcal{P}$, find a feasible partition $\{\Gamma_1, \Gamma_2, \ldots, \Gamma_M\}$ of Γ and the corresponding scheduling plan (including the start execution time and executing speed of each task), that can minimize both the temperature and energy consumption while satisfying the timing and precedence constraints.

3.5. Preliminaries

HEFT (Heterogeneous Earliest Finish Time) is the most popular scheduling algorithm for DAG application to reduce the schedule length to a minimum while achieving both low complexity and high performance in heterogeneous systems [5,33]. It has been widely used by many works [5,40–42]. Since HEFT can efficiently get the minimal schedule length, it will be adopted to determine the deadlines of the precedence-constrained tasks in Section 4.2 so as to reserve the maximal DVFS scaling space to optimize the energy and temperature.

HEFT assigns tasks to the processor with the earliest finish time in a descending order of the tasks' upward rank values (denoted as RANK(τ_i)), which can be calculated by Eq. (5).

$$\mathrm{RANK}(\tau_i) = c_i + \max_{\tau_x \in \mathrm{SUCC}(\tau_i)}\{\mathrm{RANK}(\tau_x) + \omega_{i,j}\}, \tag{5}$$

where c_i represents the average WCET of task τ_i which can be calculated as $c_i = (\sum_{k=1}^{M} c_{i,k})/M$. The earliest start time $\mathrm{EST}(\tau_i, \mathcal{P}_k, f_{k,\max})$ and earliest finish time $\mathrm{EFT}(\tau_i, \mathcal{P}_k, f_{k,\max})$ of task τ_i on processor $\mathcal{P}_k$ under the maximum frequency $f_{k,\max}$ can be calculated as:

$$\begin{cases} \mathrm{EST}(\tau_{\mathrm{entry}}, \mathcal{P}_k, f_{k,\max}) = 0 \\ \mathrm{EST}(\tau_i, \mathcal{P}_k, f_{k,\max}) \\ = \max\{\mathrm{AVAIL}[k], \max\limits_{\tau_x \in \mathrm{PRED}(\tau_i)}\{\mathrm{AFT}(\tau_x) + \hat{\omega}_{x,i}\}\} \end{cases}, \\ \mathrm{EFT}(\tau_i, \mathcal{P}_k, f_{k,\max}) = \mathrm{EST}(\tau_i, \mathcal{P}_k, f_{k,\max}) + c_{i,k}. \tag{6}$$

In Eq. (6), AVAIL(k) is the earliest available time of processor $\mathcal{P}_k$, AFT(τ_x) is the actual finish time of task τ_x, and $\hat{\omega}_{x,i}$ is the communication time between τ_i and τ_x. When τ_i and τ_x are assigned to the same processor, $\hat{\omega}_{x,i} = 0$; otherwise $\hat{\omega}_{x,i} = \omega_{x,i}$.

4. Theoretical analysis

This section analyzes the optimization of temperature and energy for heterogeneous MPSoC systems and finds the metric to evaluate the optimization effect.

From the thermal model presented in Section 3.3, we can get that when the dynamic power of the processor is minimized, the temperature of the processor will be minimized; When the temperature is minimized, the temperature-dependent leakage power will be minimized, then the total energy consumption will be minimized. That is, the optimization of energy consumption coincides with that of temperature when the temperature-dependent leakage power is considered. Hence, we focus on analyzing the optimization of temperature in this section, while it also applies to energy consumption minimization.

Task scheduling for MPSoC systems consists of two parts: task assignment and task scheduling. In addition, based on the research results of our previous works on independent real-time tasks [12,43], processor temperature is prone to be minimized when tasks have similar power profiles. Hence, the following two aspects will be explored to minimize the system energy/temperature: balance the thermal loads of processors by an energy/thermal aware task assignment so as to minimize the peak temperature of the system; balance the thermal loads of tasks by DVFS while satisfying the task-level deadlines so as to minimize the individual processor temperature. We first give the definitions of task thermal load and processor thermal load as follows.

Definition 1. *Task Thermal Load:* The effect on temperature when executing the task.

Definition 2. *Processor Thermal Load:* The effect on temperature when executing all of the tasks assigned to the processor according to the determined scheduling plan.

4.1. Balance the thermal loads of processors

To balance the thermal loads of processors, we first need to find the metric to evaluate the thermal load. For a given task assignment, denote the tasks assigned to $\mathcal{P}_k$ as Γ_k, and the corresponding scheduling plan as $\prod(t)$. $\prod(t)$ is responsible to identify the task τ_i executing at time t with speed s_i. Tasks in Γ_k will be executed by $\prod(t)$ cyclically in each frame of the DAG-based application, i.e., $\prod(t) = \prod(t + D)$, where D is the frame size.

For balancing the thermal loads of processors, the ideal result that can be achieved is that all processors have the same temperature traces. In this case, there will be no heat transfer between processors, hence the thermal model described by Eq. (3) can be used. For a task τ_i in Γ_k at speed s_i, its dynamic power $P_d^i = C_k^{\mathrm{eff}} A_i s_i^3$. By Eq. (4), the thermal load $\eta_{i,s_i}(t)$ of τ_i in Γ_k at speed s_i can be calculated as:

$$\eta_{i,s_i}(t) = \begin{cases} \dfrac{C_k^{\mathrm{eff}} A_i s_i^3}{\lambda_k}(1 - e^{-\lambda_k t}) & 0 \leqslant t \leqslant \dfrac{c_i}{s_i} \\ \dfrac{C_k^{\mathrm{eff}} A_i s_i^3}{\lambda_k}(1 - e^{-\lambda_k \frac{c_i}{s_i}})e^{-\lambda_k(t - \frac{c_i}{s_i})} & t > \dfrac{c_i}{s_i} \end{cases} \tag{7}$$

Here, the thermal load of only one task is considered, hence treating the initial temperature as zero for now.

Denote the integral of $\eta_{i,s_i}(u)$ until time t as $\varsigma_{i,s_i}(t)$. Then, when $t \to \infty$, $\varsigma_{i,s_i}(t)$ is bounded to be:

$$\varsigma_{i,s_i}(\infty) = \int_0^\infty \eta_{i,s_i}(u)du = \frac{C_k^{\mathrm{eff}} A_i s_i^2 c_i}{\lambda_k}. \tag{8}$$

Denote the processor thermal load with periodic schedule $\prod(t)$ as $\eta_{\prod}(t)$. Then, by superposition, the integral of temperature with initial temperature $\theta(0)$ within the frame can be deduced as:

$$\begin{aligned} &\int_0^D \left(\theta(0)e^{-\lambda_k u} + \eta_{\prod}(u)\right)du \\ &= \int_0^\infty \theta(0)e^{-\lambda_k u}du + \sum_{\tau_i \in \Gamma_k}\int_0^\infty \eta_{i,s_i}(u)du \\ &\quad - \left(\theta(0)e^{-\lambda_k D} + \sum_{\tau_i \in \Gamma_k}\eta_{i,s_i}(D - st_i)\right)\int_0^\infty e^{-\lambda_k u}du \\ &= \frac{\theta(0) - \theta(D)}{\lambda_k} + \sum_{\tau_i \in \Gamma_k}\varsigma_{i,s_i}(\infty). \end{aligned} \tag{9}$$

After repeated executions by $\prod(t)$, the temperature at the end of the frame will converge to the following value:

$$\lim_{j \to \infty}\theta(j \cdot D) = \frac{\eta_{\prod}(D)}{1 - e^{-\lambda_k D}} = \frac{P_{\mathrm{d},k}^D}{\lambda_k}, \tag{10}$$

where $P_{\mathrm{d},k}^{D}$ is the dynamic power of an equivalent constructed task having the same end temperature with that of the tasks in Γ_k executing by $\prod(t)$ in a frame, and can be computed as:

$$P_{\mathrm{d},k}^{D} = \sum_{\tau_i \in \Gamma_k} \frac{P_{\mathrm{d}}^{i}(1 - e^{-\lambda_k c_i})e^{-\lambda_k (D - e_i)}}{1 - e^{-\lambda_k D}}, \tag{11}$$

where e_i is the finish time of τ_i. Further, according to the definition of thermal steady state ($\theta(0) = \theta(D)$), we can calculate the average temperature of processor $\mathcal{P}_k$ in a frame during the steady state (denoted as $\theta_{\mathrm{ss},k}^{\mathrm{avg}}$) as follows:

$$\theta_{\mathrm{ss},k}^{\mathrm{avg}} = \frac{\int_0^D \left(\frac{\eta \prod(D)}{1 - e^{-\lambda_k D}} e^{-\lambda_k u} + \eta \prod(u) \right) du}{D} = \frac{\sum_{\tau_i \in \Gamma_k} \varsigma_{i,s_i}(\infty)}{D}. \tag{12}$$

Then, considering the mathematical transformation conducted by Eq. (4), the real average temperature of processor $\mathcal{P}_k$ under steady state (denoted as $\Theta_{\mathrm{ss},k}^{\mathrm{avg}}$) can be calculated by Eqs. (8) and (12) as:

$$\Theta_{\mathrm{ss},k}^{\mathrm{avg}} = \frac{\sum_{\tau_i \in \Gamma_k} C_k^{\mathrm{eff}} A_i s_i^2 c_i}{\lambda_k C_k D} + \xi_k, \ \xi_k = \frac{R_k \beta_k + \Theta_{\mathrm{a}}}{1 - R_k \alpha_k}. \tag{13}$$

Now, we can say that, to balance the thermal loads of processors, we need to find the optimal task assignment that results in lower $\Theta_{\mathrm{ss},k}^{\mathrm{avg}}$ values with a smaller variance.

4.2. Balance the thermal loads of tasks

To balance the thermal loads of tasks while satisfying the precedence constraints and schedulability, we intend to fairly distribute the deadline slack of the application to tasks based on their task levels. Here, the deadline slack ($DS(G)$) of the application is defined as the difference between the deadline of the application ($D(G)$) and the minimum schedule length (denoted as $LB(G)$) of the application obtained by HEFT, i.e., $DS(G) = D(G) - LB(G)$. $LB(G)$ can be calculated as:

$$LB(G) = \min_{\mathcal{P}_k \in \mathcal{P}} \{\mathrm{EFT}(\tau_{\mathrm{exit}}, \mathcal{P}_k, f_{k,max})\}. \tag{14}$$

The task level of each task can be calculated by Eq. (15).

$$\begin{cases} L(\tau_{\mathrm{entry}}) = 1 \\ L(\tau_i) = \max\limits_{\tau_x \in \mathrm{PRED}(\tau_i)} \{L(\tau_x) + 1\} \end{cases}. \tag{15}$$

For tasks at different levels, to balance their thermal loads, we can evenly allocate the deadline slack time to them to reduce the executing frequency thus decreasing the dynamic power consumption and hence the temperature. Tasks at the same level can be considered as independent, hence share the same slack time. Further, considering the accumulative feature of the slack time, the deadline of each task τ_i is set as Eq. (16).

$$D(\tau_i) = \mathrm{AFT}(\tau_i) + \frac{DS(G)}{L(\tau_{\mathrm{exit}})} \times L(\tau_i). \tag{16}$$

Once the deadlines of tasks are determined, each task can be assigned to a proper processor that can balance the thermal loads of processors (i.e., lower $\Theta_{\mathrm{ss},k}^{\mathrm{avg}}$ values with a smaller variance) based on the analysis in Section 4.1 while satisfying its deadline to balance the thermal loads of tasks. Here, one may notice that the slack time distribution method is similar to that of [5], but the key step of the whole proposed task scheduling approach lies in the thermal-aware task assignment approach which is different from that of [5], i.e., the allocation sequence of tasks to processors, which will be described in the next section.

5. Optimization approach

Based on the theoretical analysis in Section 4, this section proposes an energy/thermal aware task scheduling approach for DAG applications on heterogeneous MPSoC systems by two steps. First, optimize the system by an energy/thermal aware task assignment approach; Then, optimize the system by reducing the waiting time between parallel tasks that have the same successor task.

5.1. By energy/thermal aware task assignment

According to the temperature and energy optimization analysis from the aspect of balancing the thermal loads of processors, an optimal task assignment should result in lower $\Theta_{\mathrm{ss},k}^{\mathrm{avg}}$ values with a smaller variance (see Eq. (13)); according to the temperature and energy optimization analysis from the aspect of balancing the thermal load of tasks, an optimal task assignment should satisfy the deadlines of tasks determined by Eq. (16). This section introduces the approach to achieve the two goals.

For the task assignment of a DAG-based parallel application, we first need to determine the order of the tasks that satisfies the precedence constraints. As mentioned before, the most popular DAG-based scheduling algorithm HEFT aims to reduce the schedule to a minimum, so it uses the upward rank values of tasks as the priority standard, and assigns each task to the processor that obtains the earliest finish time. While this work aims to reduce the temperature and energy of the MPSoC systems, we intend to use the power/thermal dissipation factors of the tasks as the priority standard, and assign each task to the processor that obtains the minimum temperature and energy.

We use task level (see Eq. (15)) to guarantee the precedence constraints between tasks. In general, tasks with lower levels should be ordered before that with higher levels, and tasks with the same level will be ordered in a descending order of their power/thermal dissipation factors. While in practice, the task should be ready for ordering as long as its predecessors have been ordered. Next, we determine the order of the processors.

Since the processors support limited discrete pairs of voltage and frequency, we transform a real processor $\mathcal{P}_k$ to x_k virtual cores with fixed voltage and frequency. The virtual cores belonging to the same real processor share the same parameters except for the frequency/voltage, and contributes to the same computation load. We can get $\hat{M}$ virtual cores in total, where $\hat{M} = \sum_1^M x_k$. Thus, based on the thermal load metric deduced in Eq. (13), we can equivalently transform the optimization object for task assignment from $\Theta_{\mathrm{ss},l}^{\mathrm{avg}}$ to $\hat{\Theta}_{\mathrm{ss},l}^{\mathrm{avg}}$ by lemma 1, where $\Theta_{\mathrm{ss},l}^{\mathrm{avg}}$ represents the average temperature in steady state of the transformed virtual core $\mathcal{P}_l$ belonging to a real processor $\mathcal{P}_k$.

Lemma 1. *(Optimal Assignment Condition) The optimal task assignment that results in lower $\Theta_{\mathrm{ss},l}^{\mathrm{avg}}$ values with a smaller variance is equivalent to that of $\hat{\Theta}_{\mathrm{ss},l}^{\mathrm{avg}}$, where*

$$\hat{\Theta}_{\mathrm{ss},l}^{\mathrm{avg}} = \frac{C_l^{\mathrm{eff}} \hat{s}_l^2}{\lambda_l C_l} \sum_{\tau_i \in \Gamma_l} A_i c_i,$$
$$C_l^{\mathrm{eff}} = C_k^{\mathrm{eff}}, C_l = C_k, \lambda_l = \frac{1}{R_k C_k} - \frac{\alpha_k \cdot \hat{s}_l}{C_k}. \tag{17}$$

Proof. To achieve the optimization goal of resulting in lower $\Theta_{\mathrm{ss},l}^{\mathrm{avg}}$ values with a smaller variance, we need to minimize the expectation (denoted as $EXP(\Theta_{\mathrm{ss},l}^{\mathrm{avg}})$) and variance (denoted as $DEV(\Theta_{\mathrm{ss},l}^{\mathrm{avg}})$) of $\Theta_{\mathrm{ss},l}^{\mathrm{avg}}$s. Therefore, we prove the equivalence of $\Theta_{\mathrm{ss},l}^{\mathrm{avg}}$ and $\hat{\Theta}_{\mathrm{ss},l}^{\mathrm{avg}}$ from the two aspects. By Eq. (13), $EXP(\Theta_{\mathrm{ss},l}^{\mathrm{avg}}) = D \cdot EXP(\hat{\Theta}_{\mathrm{ss},l}^{\mathrm{avg}}) + EXP(\xi_l)$. Since $EXP(\xi_l)$ is fixed for a given set of processors, minimizing $EXP(\Theta_{\mathrm{ss},l}^{\mathrm{avg}})$ is equivalent to minimizing $EXP(\hat{\Theta}_{\mathrm{ss},l}^{\mathrm{avg}})$. By Eq. (13), $DEV(\Theta_{\mathrm{ss},l}^{\mathrm{avg}}) = \Sigma_1^{\hat{M}}(\Theta_{\mathrm{ss},l}^{\mathrm{avg}} - EXP(\Theta_{\mathrm{ss},l}^{\mathrm{avg}}))^2 = D^2 \cdot DEV(\hat{\Theta}_{\mathrm{ss},l}^{\mathrm{avg}})$. Since D is a constant for a given DAG-based application, minimizing $DEV(\Theta_{\mathrm{ss},l}^{\mathrm{avg}})$ is equivalent to minimizing $DEV(\hat{\Theta}_{\mathrm{ss},l}^{\mathrm{avg}})$. □

Since task assignment is a well known NP-hard problem, this work focuses on finding a sub-optimal heuristic approach. According to

Eq. (17), the optimization object $\hat{\Theta}_{ss,l}^{avg}$ contains two kinds of parameters: core-dependent and task-dependent. Let $\Lambda^{\hat{M}} = [\Lambda_1, \Lambda_2, \dots, \Lambda_{\hat{M}}]$ capture the core-dependent parameters, then $\Lambda_l = C_l^{eff}\hat{s}_l^2/\lambda_l C_l$, which can be termed as the power/thermal dissipation factor of core $\mathcal{P}_l$. In a similar way, let $B^{\hat{M}} = [B_1, B_2, \dots, B_{\hat{M}}]^T$ capture the task dependent parameters, then $B_l = \sum_{\tau_i \in \Gamma_l} \delta_i = \sum_{\tau_i \in \Gamma_l} (A_i c_i)$, which can be termed as the power/thermal dissipation factor of task subset Γ_l (δ_i can be termed as the power/thermal dissipation factor of task τ_i). Then, $\hat{\Theta}_{ss,l}^{avg}$ can be expressed as $\hat{\Theta}_{ss,l}^{avg} = \Lambda_l \times B_l$, and the optimal task assignment solution should result in lower $\hat{\Theta}_{ss,l}^{avg}$ values with a smaller variance. Since $\Sigma_1^{\hat{M}} B_l = \Sigma_1^N \delta_i$ is a fixed value for a given task set, to achieve a better task assignment, a task with higher power/thermal dissipation factor δ_i should be assigned to a core with lower power/thermal dissipation factor Λ_l. The reasonableness of this heuristic can be proved by simple deduction, hence we omit the details due to the space limit.

Based on the above analysis, one intuitive heuristic way to obtain the optimal task scheduling is as follows: sort the tasks in a decreasing order by δ_i, and then assign them to the virtual cores in an increasing order by Λ_l. However, compared with the task sorting by the metric of the upward rank value adopted by HEFT, this kind of task sorting by the metric of the power/dissipation factor can better balance the thermal loads among processors, but result in performance loss, thus reducing the optimization space for the subsequent DVFS scaling. Further, for DAG-based applications, the advantages of sorting tasks by their power dissipation factors cannot be given a full play due to the precedence constraints between tasks, which may result in failing to offset the negative effects on the optimization room for DVFS. Actually, this guess has been proved right in the first group of the experiments. To sum up, based on all of the analysis above, we propose a novel energy/thermal aware task scheduling by considering the advantages of both as follows: for the virtual cores, sort them in an increasing order by their power/thermal dissipation factors, while for the tasks, sort them in a decreasing order by their upward rank values, then assign the tasks sequentially to the feasible virtual cores. See Algorithm 1 for details.

Algorithm 1 consists of three main parts. The first part (lines 2–3) transforms the real processors into virtual cores by their DVFS scaling levels. The second part (lines 4–6) sorts the virtual cores and tasks in an optimal sequence, and calculates the deadlines of tasks. The third part (lines 8–27) allocates tasks to virtual cores while satisfying the deadlines of tasks. When two (virtual) processors have the same thermal dissipation factor, first-fit policy is adopted. In particular, when the condition shown in line 13 cannot be satisfied, we assign the task to the processor with the earliest finish time (lines 22–26). While assigning tasks in a way to balance the energy/thermal loads of processors, the deadlines of tasks are used to guarantee both the tasks' schedulability and the DVFS scaling room. In specific, without the deadline constraints, tasks will be continuously allocated to the virtual cores with lower power/thermal dissipation factors until the deadline of the application is violated. This can on one hand delay the completion time of the application due to the precedence constraints between tasks, and on the other hand reduce the scaling room of DVFS due to the load imbalance of processors.

In Algorithm 1, $\text{AET}(\Gamma_{tmp})$ represents the sum of the actual execution time of tasks in Γ_{tmp}; $\tilde{\Gamma}_l$ represents the tasks assigned to the other virtual cores that belong to the same processor with $\mathcal{P}_l$, and $\text{AET}(\tilde{\Gamma}_l)$ represents the sum of the actual execution time of tasks in $\tilde{\Gamma}_l$. In addition, as we mentioned before, the virtual cores belonging to the same real processor share the same available time by setting $\text{AVAIL}[l] = \text{AVAIL}[k]$ (see line 17). The time complexity of Algorithm 1 is $O(|N|^2 \times \hat{M})$.

As to the processor level utilization, the task level deadline (see Eq. (16)) that we transform from the application deadline can guarantee the schedulability of the application, i.e., not exceeding the application deadline and the processor computing capacity. The optimization goal of this work is to minimize the temperature and energy consumption of the system through balancing the thermal loads of the system, which inevitably have some impacts on the balance of the computing loads. However, the task level deadline can also help keeping the balance of processor utilizations to some extent, thus avoiding the occurrence of too imbalance of the processor computing loads.

ALGORITHM 1: Energy/Thermal Aware Task Assignment.

Input: $G = (N, E, C)$ and $\mathcal{P} = \{\mathcal{P}_1, \cdots, \mathcal{P}_M\}$.
Output: Task assignment $\{\Gamma_1, \Gamma_2, \cdots, \Gamma_M\}$ and the task scheduling plan $\prod(G)$ ($\text{AST}(\tau_i)$ and $s_{i,k}$.

1 $\{\Gamma_1, \Gamma_2, \cdots, \Gamma_M\} = \{\emptyset, \emptyset, \cdots, \emptyset\}$;$\Theta_{\text{init}} = \Theta_a$;
2 Transform each real processor $\mathcal{P}_k$ into x_k virtual cores; Calculate the number of virtual cores as $\hat{M} = \sum_1^M x_k$;
3 Denote Γ_l as the task set assigned to virtual core $\mathcal{P}_l$, $\Gamma_l = \emptyset$;
4 Calculate the power/thermal dissipation factor of $\mathcal{P}_l$ as $\Lambda_l = C_l^{eff}\hat{s}_l^2/\lambda_l C_l$, and sort them in the non-decreasing order;
5 Calculate the upward rank value of τ_i by Eq. (5) as RANK_i, and sort them in the non-increasing order by RANK_i;
6 Calculate the task levels and deadlines by Eq. (15) and (16);
7 Initialize each $\text{AVAIL}[l]$ and $\text{AVAIL}[k]$ as 0;$\text{AST}(\tau_{\text{entry}}) = 0$;
8 **while** $\Gamma \neq \emptyset$ **do**
9 $l = 1; flag = false$;
10 Create a temporary empty subset Γ_{tmp};
11 **while** $l \leqslant \hat{M}$ **do**
12 $\Gamma_{\text{tmp}} = \Gamma_l + \tau_i$;
13 **if** $(AET(\Gamma_{tmp}) + AET(\tilde{\Gamma}_l)) \leqslant D(G)$ && $AFT(\tau_i) \leqslant D(\tau_i)$ **then**
14 Set the actual start time of τ_i as $\text{AST}(\tau_i) = \max\{\text{AVAIL}[l], \max_{\tau_x \in \text{PRED}(\tau_i)} \{\text{AFT}(\tau_x) + \hat{\omega}_{x,i}\}$;
15 Set the executing frequency of τ_i as f_l;
16 Set the actual finish time of τ_i as $\text{AFT}(\tau_i) = \text{AST}(\tau_i) + \text{AET}(\tau_i)$, where $\text{AET}(\tau_i) = c_{i,l} = c_{i,k} \cdot f_{k,\max}/f_l$;
17 $\text{AVAIL}[k] = \text{AVAIL}[k] + \text{AET}(\tau_i)$, and synchronize it to all its virtual cores ($\text{AVAIL}[l] = \text{AVAIL}[k]$);
18 $\Gamma_l = \Gamma_l + \tau_i$;$\Gamma = \Gamma - \tau_i$;$flag = true$;break;
19 **end**
20 $l = l + 1$;
21 **end**
22 **if** $(!flag)$ **then**
23 Assign τ_i to the processor with the earliest finish time;
24 Update the values of the task executing frequency, $\text{AST}(\tau_i), \text{AFT}(\tau_i), \text{AET}(\tau_i), \text{AVAIL}[k]), \text{AVAIL}[l])$;
25 $\Gamma = \Gamma - \tau_i$;
26 **end**
27 **end**
28 **if** $(AFT(\tau_{exit}) > D(G))$ **then**
29 Exit(1);
30 **end**
31 Combining all Γ_l belonging to the same $\mathcal{P}_k$ into Γ_k;
32 **return** $\{\Gamma_1, \Gamma_2, \cdots, \Gamma_M\}$;

5.2. By reducing the waiting time between tasks

In Algorithm 1, the strategy of evenly allocating the deadline slack time to tasks according to their levels cannot completely explore the optimization space, since there still may exist waiting between the parallel tasks that have the same successor task. This section explores such optimization space by reducing or eliminating the waiting time between tasks.

The main idea of the optimization method proposed in this section is to further stretch the $\text{AFT}(\tau_i)$ obtained by Algorithm 1 to $\text{LFT}(\tau_i)$, which

can be calculated by Eq. (18).

$$\begin{cases} \mathrm{LFT}(\tau_{\mathrm{exit}}) = D(G) \\ \mathrm{LFT}(\tau_i) \\ = \min\{\mathrm{AST}(\tau_{\mathrm{DN}(\tau_i)}), \min\limits_{\tau_x \in \mathrm{SUCC}(\tau_i)}\{\mathrm{AST}(\tau_x) - \hat{\omega}_{i,x}\}\} \end{cases}, \tag{18}$$

where $\tau_{\mathrm{DN}(\tau_i)}$ refers to the downward neighbor task of τ_i on the same processor. When $\mathrm{LFT}(\tau_i) > \mathrm{AFT}(\tau_i)$, adjust the executing frequency of τ_i to the maximal frequency supported by the corresponding processor that can satisfy the timing constraint of $\mathrm{LFT}(\tau_i)$. In particular, when the adjusted frequency cannot fully utilize the slack time between $\mathrm{AFT}(\tau_i)$ and $\mathrm{LFT}(\tau_i)$ due to the discreteness of frequency, we will delay the execution of this task by half of the slack time remained (see lines 8–9 in Algorithm 2). This behavior aims to reserve slack time between tasks to further cool down the processor temperature. The process above is conducted in a descending order of $\mathrm{AFT}(\tau_i)$.

ALGORITHM 2: Optimization by LFT.

Input: The task scheduling result by Algorithm 1.
Output: Adjusted task scheduling result.
1 Sort the tasks in the non-increasing order of $\mathrm{AFT}(\tau_i)$;
2 **for** $\tau_i \in \Gamma$ **do**
3 Calculate $\mathrm{LFT}(\tau_i)$ by Eq. (18);
4 Calculate the new frequency ($\hat{f}_l$) by considering both the slack time and the discreteness of the frequency;
5 **if** ($\hat{f}_l < f_l$) **then**
6 Set the executing frequency of τ_i as $\hat{f}_l$;
7 Update $\mathrm{AET}(\tau_i)$;
8 $\mathrm{AST}(\tau_i) = (\mathrm{LFT}(\tau_i) + \mathrm{AST}(\tau_i) - \mathrm{AET}(\tau_i))/2$;
9 $\mathrm{AFT}(\tau_i) = (\mathrm{LFT}(\tau_i) + \mathrm{AST}(\tau_i) + \mathrm{AET}(\tau_i))/2$;
10 **end**
11 **end**

6. Experiments

This section designs experiments to validate the superiority of the proposed energy/thermal aware task scheduling (ETA-TS) approach with respects to the optimization on both temperature and energy consumption. To validate the optimization efficiency, we compare our proposed energy/thermal aware task scheduling approach ETA-TS with the state-of-the-arts, DUECM [5] and EA-TS [17], under the following four groups of applications: the standard DAG-based application shown in Fig. 1, the real-world application of a molecular dynamics code adopted by Topcuoglu et al. [33], the real-world problem of the Gaussian elimination [5,33], and the real-world application of Laser Interferometer Gravitational Wave Observatory (LIGO) provided by Pegasus Workflow Generator [44,45] and adopted by many works like [46–48]. All the four applications chosen above have their own distinctive features which can be seen in the corresponding figures in this section.

In all of the experiments, the same system models are adopted for the three approaches for the sake of fairness. As to the temperature, we calculate it according to Eqs. (2)–(4) (see Section 3.3), which is essentially an enforced version of the existing method MatEx [49] by taking the dynamic change feature of leakage power with the temperature into consideration. MatEx has been proved having enough accuracy and lower computing complexity compared with the widely acknowledged computing method HotSpot [50], which is actually derived from the same differential equation of temperature. The variable symbols we use are quite different with that of MatEx, but the computing results have the same accuracy and efficiency.

As to the three algorithms (DUECM, EA-TS and ETA-TS) we compare, the optimization ideas behind them are as follows:

- *DUECM (Downward and Upward Energy Consumption Minimization)*: First assign tasks to processors by HEFT [33], then use the slack between the schedule length and the deadline of the application to scale down the executing frequency of tasks, and finally further minimize the energy consumption from a "downward" perspective by leveraging the latest finish time.
- *EA-TS (Energy Aware Task Scheduling)*: First transform the real processors to virtual cores according to the frequency levels they support, then adopt an energy-aware heuristic to assign tasks in a descending order of their power dissipation factors ($\delta'_i = A_i c_i$) to the virtual cores in an increasing order of their power dissipation factors ($\Lambda'_l = C_l^{\mathrm{eff}} \hat{s}_l^2$).
- *ETA-TS (Energy/Thermal Aware Task Scheduling)*: First transform the real processors to virtual cores according to the frequency levels they support; then adopt an energy/thermal-aware heuristic to assign tasks in a descending order of their upward rank values to the virtual cores in a non-decreasing order of their power/thermal dissipation factors ($\Lambda_l = C_l^{\mathrm{eff}} \hat{s}_l^2 / \lambda_l C_l$), while satisfying the deadlines of tasks to balance the thermal loads of tasks and guarantee the schedulability; finally further minimize the temperature and energy consumption by exploring the optimization space due to the waiting between the parallel tasks that have the same successor task.

In particular, the EA-TS algorithm proposed by Zhou et al. in [17] keeps assigning tasks to a virtual core until violating its computing capacity or the task's timing constraint or the temperature constraint. Such assignment strategy may severely damage the schedulability of the subsequent tasks due to the precedence constraints between tasks in a DAG-based application. Besides, the calculation of the peak temperature is also time consuming, which decreases the efficiency of the algorithm. To this end, we slightly make a modification of the original algorithm by adopting a decision condition in the task assignment heuristic to our proposed approach (ETA-TS) while reserving the critical energy-aware feature (i.e., δ'_i and Λ'_l keeps unchanged).

6.1. Setup

We perform our experimental simulations based on a heterogeneous MPSoC system with eight processors. Processor-related parameters are shown in Table 3, which have been adopted by many works [15–17] for simulation. The number of frequency levels (denoted as x_k) varies from 3 to 11. The frequency levels are derived by the decrements of $f_{k,\max}$ with a step size of 0.2. The ambient temperature Θ_a is set as 45°C. The mean value (c_i^{avg}) we use to generate the WCETs of a task in the DAG-based parallel application is randomly generated from the range of [20 ms, 200 ms]. Meanwhile, $c_{i,k}$ is controlled by a range percentage parameter σ, which represents the heterogeneity factor of processor speeds. The value of $c_{i,k}$ is randomly selected from the range $[c_i^{\mathrm{avg}}(1-\sigma/2), c_i^{\mathrm{avg}}(1+\sigma/2)]$. A high value of σ indicates a significant difference in the WCETs of a task among the processors. All of the tasks share a common initial deadline D. The communication time $\omega_{i,j}$ between tasks is controlled by the parameter CCR, which is the ratio of the average communication cost to the average computation cost. A low value of CCR represents a computation-intensive application. The task switching activity factor A_i is uniformly distributed in the range of [0.4, 1], which demonstrates the heterogeneity of tasks. The values of the parameters basically can reflect

Table 3
Processor parameters.

$\mathcal{P}_k$	$f_{k,\max}$	α_k	β_k	C_k^{eff}	R_k	C_k
$\mathcal{P}_1$	3.3	0.1666	20.5060	3.656	0.282	340
$\mathcal{P}_2$	3.4	0.1942	5.01870	2.138	0.487	295
$\mathcal{P}_3$	3.3	0.2043	12.7880	3.645	0.288	320
$\mathcal{P}_4$	3.0	0.1942	15.6262	4.556	0.238	320
$\mathcal{P}_5$	3.2	0.1574	20.6393	3.204	0.278	295
$\mathcal{P}_6$	3.1	0.1586	11.9759	2.719	0.480	255
$\mathcal{P}_7$	3.0	0.1124	10.3490	2.074	0.661	335
$\mathcal{P}_8$	2.6	0.1754	13.1568	2.332	0.680	380

Table 4
Two metrics comparison under the application shown in Fig. 1.

	Peak temperature (°C)	Standard deviation	Energy consumption(J)	
EA-TS	By Delta	63.5	3.31	82.3
	By Rank	61.6	1.76	86.3
ETA-TS	By Delta	60.7	1.65	86.1
	By Rank	58.1	1.06	81.8

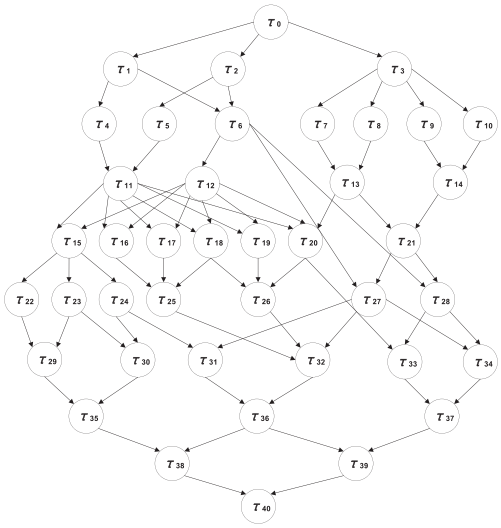


Fig. 2. The DAG representation of the molecular dynamics code [33].

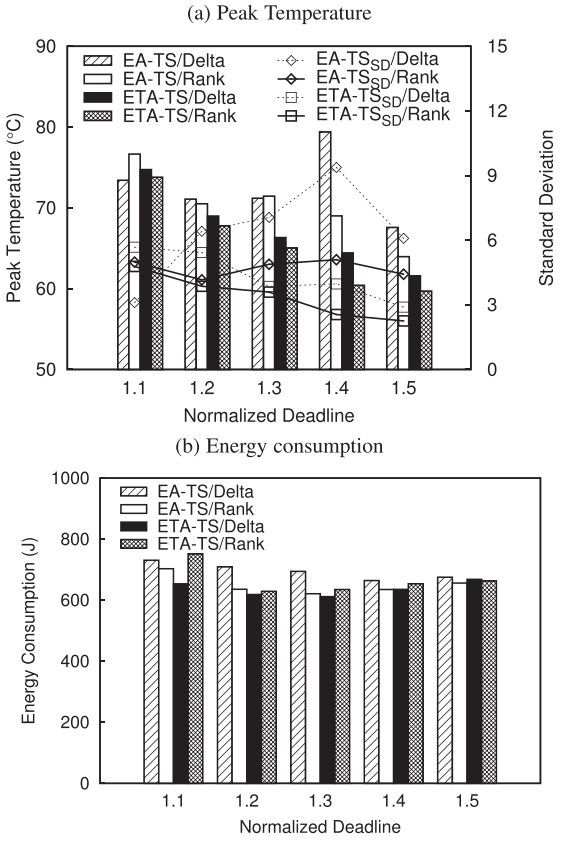


Fig. 3. Two metrics comparison under the Molecular application.

the task features on some high-performance embedded processors, such as Intel Mobile Pentium III and ARM Cortex-A9.

Although the experiments are conducted by simulations like many studies [5,17,32], the parameters of processors and the DAG-based applications are all based on real-world ones.The benefit of such simulations is that it can significantly reduce the development cost during the design phase, and provide efficient optimization guide for the implementation phase.

6.2. Experimental results

6.2.1. Comparison between two task ordering metrics

As analyzed in Section 5.1, assigning tasks in a order by the metric of power/thermal dissipation factor can balance the energy/thermal loads of processors, thus minimizing the temperature and energy consumption. However, due to the precedence constraints, this metric cannot give a full play to its advantages while sacrificing the DVFS optimization room compared with the metric of upward rank value. This section evaluates the effects of the two metrics on the scheduling results.

As we mentioned before in Section 5.1, the order of tasks has significant effect on the task assignment results. By adopting the upward rank values of tasks (see Eq. (5)) as the metric, HEFT sorts the tasks in a decreasing order to reduce the schedule length to a minimum, thus reserving the maximal optimization room for the subsequent DVFS scaling. Another task sorting metric is the power/thermal dissipation factors of tasks (denoted as $\delta_i = A_i c_i$).

We evaluate the effects of the two metrics (denoted as "By Delta" and "By Rank") under two applications from three aspects: peak temperature of the system, standard deviation of the peak temperature of different processors, and the energy consumption of the system. We first analyze the effects under the standard DAG-based application (see Fig. 1) on a heterogeneous MPSoC system with three processors ($\mathcal{P}_1$, $\mathcal{P}_2$, and $\mathcal{P}_7$ in Table 3 with frequency levels of 5, 7 and 3). The schedule length ($LB(G)$) of this application obtained by HEFT algorithm is 80, hence we set the deadline of this application as $D(G) = 100$ here. With the metric of power/thermal dissipation factor, the task order is determined as $\{\tau_1, \tau_2, \tau_5, \tau_6, \tau_3, \tau_7, \tau_4, \tau_9, \tau_8, \tau_{10}\}$; With the metric of upward rank value, the task order is determined as $\{\tau_1, \tau_3, \tau_4, \tau_2, \tau_5, \tau_6, \tau_9, \tau_7, \tau_8, \tau_{10}\}$. Table 4 shows the comparison results under this application. From Table 4, we can see that in general the scheduling result under the metric of upward rank value is better than that of power/thermal dissipation factor.

Further, we analyze the effects under a real-world application with a much irregular task graph (Molecular application in [33], see Fig. 2) on a heterogeneous MPSoC system with eight processors (see Table 3, with frequency levels of 5, 7, 9, 11, 8, 10, 3 and 6). Here, CCR is set as 1, and σ is set as 0.75. Fig. 3 shows the comparison results under this application with varying deadlines. Adopting the minimum scheduling length obtained by HEFT algorithm as the baseline deadline, we respectively set the normalized deadlines as 1.1, 1.2, 1.3, 1.4 and 1.5. From Fig. 3, we also find the same comparison result.

The above experimental results demonstrate the superiority of the proposed approach by combining the energy/thermal aware task scheduling with the metric of upward rank value. The EA-TS and ETA-TS algorithms in the following experiments refer to the ones with the this metric without specified otherwise.

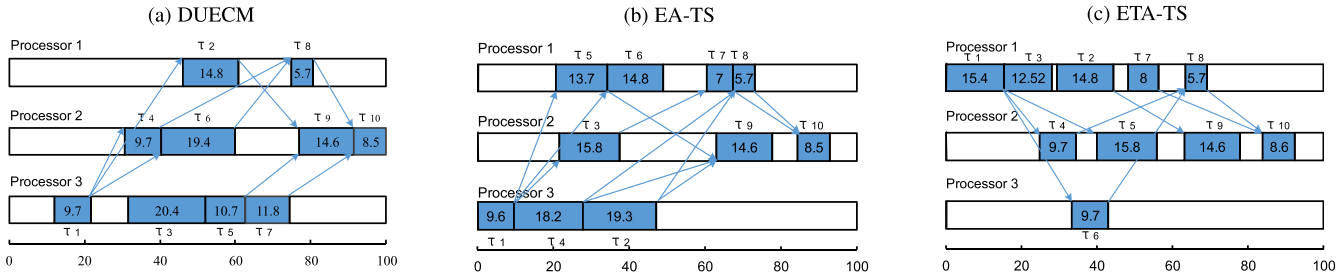


Fig. 4. Task allocation and execution plan comparisons under the application shown in Fig. 1.

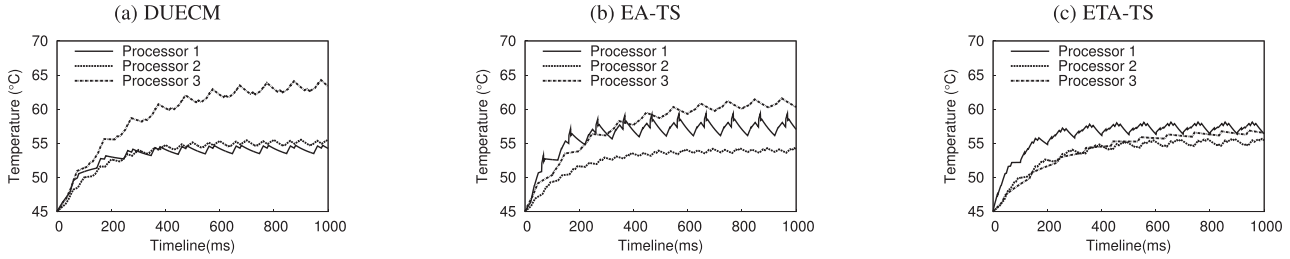


Fig. 5. Processor temperature comparison under the application shown in Fig. 1.

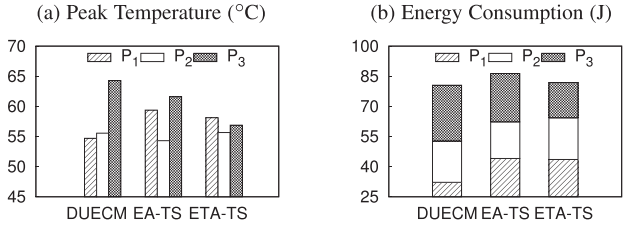


Fig. 6. Comparison results under the application shown in Fig. 1.

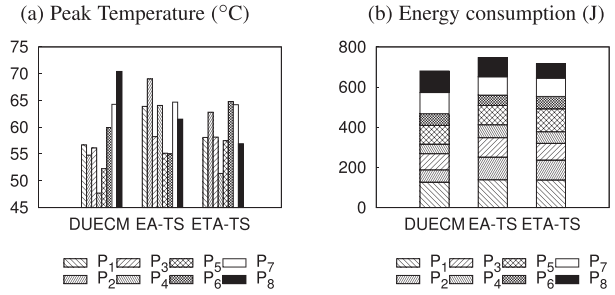


Fig. 7. Comparison results under the Molecular application.

6.2.2. Standard DAG-based application

This section evaluates the optimization efficiency of ETA-TS under the standard DAG-based application shown in Fig. 1. Similarly, the application is executed on a heterogeneous MPSoC system with three processors ($\mathcal{P}_1$, $\mathcal{P}_2$, and $\mathcal{P}_7$ with frequency levels of 5, 7 and 3) and the deadline $D(G)$ is set as 100 here.

Fig. 4 shows the results of task assignment and scheduling plan determined by the three algorithms. Here, the reason we only choose three processors is that there are only ten tasks in this application, and there are at most five tasks in each level. From Fig. 4, we can intuitively see that the scheduling results of the three algorithms are quite different. This is because of the different scheduling heuristics they adopt: DUECM aims to assign tasks to the processors with the earliest finish time, EA-TS aims to assign tasks to the processors with a lower value of dynamic power consumption, while ETA-TS aims to assign task to the processors with a lower value of energy/thermal load. Since the task assignments by the three algorithms are different, the frequency scheduling results are different accordingly, which can be seen from the actual execution time of tasks shown in Fig. 4.

Fig. 5 shows the corresponding temperature curves of the three processors. The observation time length is set ten times of $D(G)$ here, i.e., 1000, by which the system can reach the thermal steady state. Fig. 6 further shows the peak temperature and energy consumption of the three processors respectively. From the two figures, we can see that compared with both DUECM and EA-TS, our proposed algorithm ETA-TS can better balance the energy/thermal loads of the heterogeneous processors. In specific, Fig. 6(a) shows that ETA-TS has the lowest peak temperature which is 4.7°C lower than that by EA-TS and 7.4°C lower than that by DUECM; Fig. 6(b) shows that ETA-TS achieves a competitive energy consumption optimization compared with DUECM while significantly decreasing the peak temperature.

6.2.3. Molecular dynamics code

This section evaluates the optimization efficiency of ETA-TS under the Molecular application [33] (see Fig. 2). The application is executed on a heterogeneous MPSoC system with eight processors with frequency levels of 5, 7, 9, 11, 8, 10, 3 and 6.

Fig. 7 shows the peak temperature and energy consumption of the eight processors by DUECM, EA-TS and ETA-TS, with a normalized deadline of 1.5. Fig. 7(a) shows that our proposed approach ETA-TS has the lowest peak temperature which is 4.2°C lower than that by EA-TS and 5.6°C lower than that by DUECM; From Fig. 7(b), we can see that ETA-TS achieves a competitive energy consumption optimization compared with DUECM while significantly decreasing the peak temperature. To sum up, our approach can efficiently minimize the peak temperature of the system through balancing the thermal loads of processors, while ensuring a competitive energy consumption.

Further, we evaluate ETA-TS under this application with varying deadlines. Similarly, the deadlines are separately set as 1.1, 1.2, 1.3, 1.4 and 1.5 times of the baseline deadline obtained by HEFT. Experimental results are shown in Fig. 8, from which we can make the following conclusions: (1) With all five setups of deadline, ETA-TS always achieves the lowest peak temperature while ensuring a competitive energy consumption compared with DUECM and EA-TS; (2) With the increasing of deadline, all the three algorithms achieve better optimization results; (3) ETA-TS can better balance the thermal

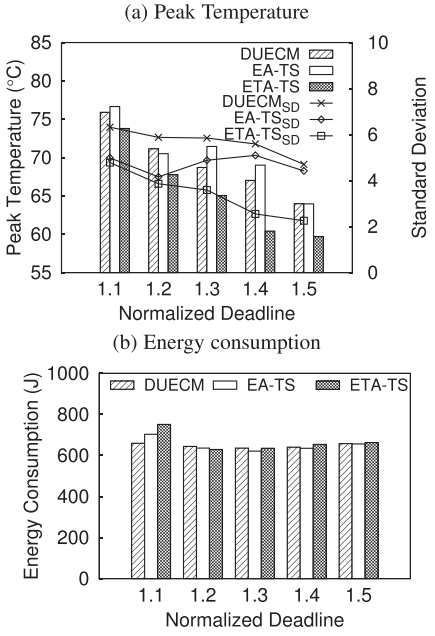


Fig. 8. Under Molecular application with varying deadlines.

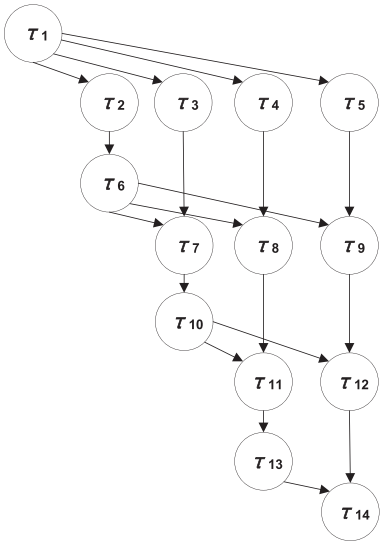


Fig. 9. The DAG representation of the Gaussian elimination application with dimension $\rho = 5$ ($N = 14$) [5,33].

loads of different processors, which is indicated by a much smaller standard deviation shown in Fig. 8(a). In specific, the peak temperature by ETA-TS is 2.9/2.7/6.4/8.6/4.3°C lower than that by EA-TS and 2.1/3.3/3.7/6.6/4.3°C lower than that by DUECM. From Fig. 8(b), we can still see that, when the normalized deadline is 1.1, the energy consumption by ETA-TS is slightly larger than that by DUECM and EA-TS. This is because that, when the deadline is too short, the optimization room is not enough for ETA-TS to give full play to balance the thermal loads thus offsetting the slightly sacrifice on dynamic energy consumption.

6.2.4. Gaussian elimination

This section evaluates ETA-TS under the Gaussian application with varying scales. The number of tasks with dimension ρ can be calculated as $|N| = (\rho + 2)(\rho - 1)/2$ (see Fig. 9 for an example). We respectively set the number of tasks as 65, 135, 230, 350, 495, 665. We first generate the task graph of this application according to the number of tasks, then

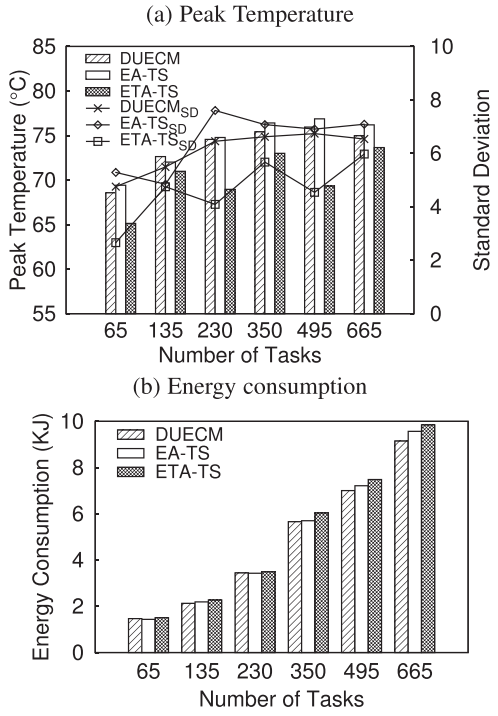


Fig. 10. Under Gaussian application with varying scales.

use the parameters described in Section 6.1 to generate the time profiles and activity factors for the tasks. Here, the value of CCR is set as 1, and the value of σ is set as 0.75. In addition, the frequency levels of the eight heterogeneous processors are set as 5, 7, 9, 11, 8, 10, 3 and 6, respectively.

Experimental results are shown in Fig. 10, from which we can make the following conclusions: (1) With all six setups of scale, ETA-TS always achieves the lowest peak temperature while ensuring a competitive energy consumption compared with DUECM and EA-TS; (2) Our approach ETA-TS can better balance the thermal loads of different processors, which is indicated by a much smaller standard deviation shown in Fig. 10(a); (3) With the increasing of scale, the optimization room of all the three algorithms is reduced. In specific, the peak temperature by ETA-TS is 4.4/1/5.8/3.4/7.6/2.6°C lower than that by EA-TS and 3.5/1.6/5.6/2.4/6.6/1.4°C lower than that of DUECM. From Fig. 10(b), we can still see that, when the number of tasks is scaled to 665, the energy consumption of ETA-TS is slightly larger than that of DUECM and EA-TS. This is because when there are too many tasks, the computation loads of processors are too high, leaving little optimization room for ETA-TS to give full play to balance the thermal loads thus offsetting the slightly sacrifice on dynamic energy consumption.

6.2.5. LIGO application

This section validates ETA-TS under the LIGO application with varying scales of 30 tasks and 50 tasks, and varying normalized deadlines ranging in [1.1, 2.0] with a step size of 0.1. For a given scale like 30 or 50, the corresponding task graph, the communication time profiles, and the average time profiles together with their standard deviations can be obtained from works [44,45]; the remained parameters like the activity factors of the tasks are generated as described in Section 6.1, and the value of σ is set as 0.75. Further, considering that the task graph provided in works [44,45] has more than one entry task, we add one virtual idle task (the execution time is 0) for the graph to make it adapt to our algorithms without loss of generality. Fig. 11 shows the adapted graph for a LIGO application with scale of 30 tasks.

For the scale of 30 tasks, considering that there are at most 7 tasks executed in parallel, we only choose three of the eight processors that

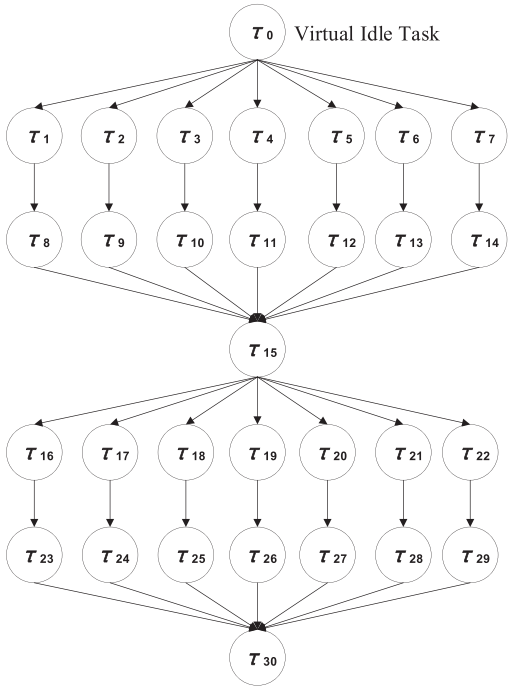


Fig. 11. The DAG representation of the LIGO application with scale of $N = 30$ [44,45].

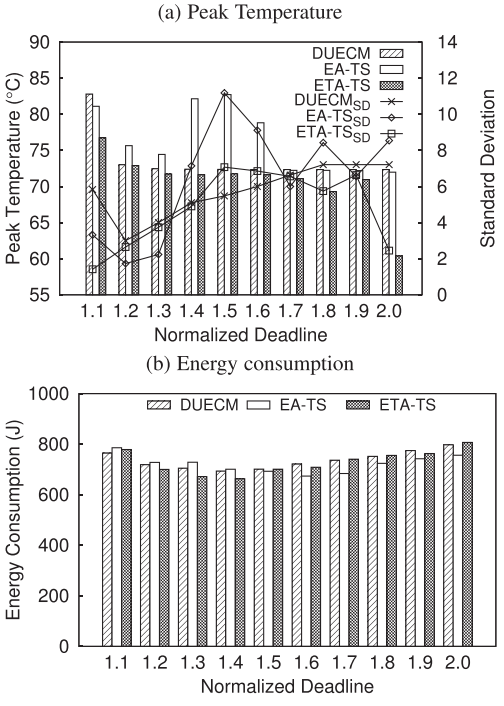


Fig. 12. Under LIGO application with varying deadlines with 30 tasks.

have distinct thermal features to better exhibits the scheduling results of the three algorithms. Here,The chosen processors are $\mathcal{P}_1, \mathcal{P}_2$, and $\mathcal{P}_7$ with frequency levels of 5, 7 and 3. Fig. 12 shows the peak temperature and energy consumption of the system scheduled by DUECM, EA-TS and ETA-TS under different normalized deadlines, from which we can make the following conclusions: (1) With all ten setups of deadline, ETA-TS always achieves the lowest peak temperature while ensuring a competitive energy consumption compared with DUECM (less than 1.02) and EA-TS (less than 1.08); (2) With the increasing

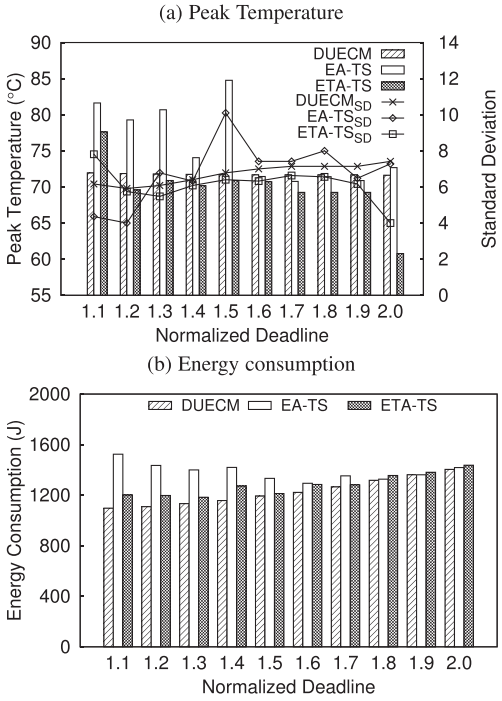


Fig. 13. Under LIGO application with varying deadlines with 50 tasks.

of deadline, ETA-TS can better balance the thermal loads of different processors, which is indicated by a much smaller standard deviation shown in Fig. 13(a). In specific, the peak temperature by ETA-TS is 4.4/2.8/2.7/10.5/11.0/7.2/1.2/3.0/1.3/11.6°C lower than that by EA-TS and 6.1/0.2/0.7/0.8/0.6/0.7/1.3/3.1/1.4/12.0°C lower than that by DUECM.

For the scale of 50 tasks, we execute them on the whole eight processors with frequency levels of 5, 7, 9, 11, 8, 10, 3 and 6. Fig. 13 shows the peak temperature and energy consumption of the system scheduled by DUECM, EA-TS and ETA-TS under different normalized deadlines, from which we can make the following conclusions: (1) With all ten setups of deadline, ETA-TS almost always achieves the lowest peak temperature while ensuring a competitive energy consumption compared with DUECM (less than 1.09) and EA-TS (less than 1.02); (2) With the increasing of deadline, ETA-TS can better balance the thermal loads of different processors, which is indicated by a much smaller standard deviation shown in Fig. 13(a). In specific, the peak temperature by ETA-TS is 4.0/9.7/9.8/3.9/13.9/0.8/1.5/2.2/1.7/12.0°C lower than that by EA-TS and -5.6/2.3/0.9/1.6/0.8/1.0/2.4/2.4/2.4/11.9°C lower than that by DUECM. From Fig. 13(a), we can still see that, when the normalized deadline is 1.1, the peak temperature by ETA-TS is larger than that by DUECM. This is because that, when the deadline is too short, the optimization room is not enough for ETA-TS to give full play to balance the thermal loads thus offsetting the slightly sacrifice on dynamic energy consumption.

6.3. Discussion

The extensive experimental results shown in Section 6.2 can sufficiently prove the efficiency of our approach (ETA-TS) on minimizing the peak temperature of the system through balancing the thermal loads of processors, while ensuring a competitive energy consumption. Through analysis, the main reasons behind lie in that:

- Compared with DUECM, ETA-TS can better minimize the system temperature through balancing the thermal loads of processors which is done by assigning tasks according to their power/thermal dissipation factors. DUECM adopts the typical HEFT algorithm to

assign tasks, which can reserve the maximal optimization room for dynamic power consumption. However, through minimizing the processor temperature with its energy/thermal aware task assignment strategy, ETA-TS minimizes the leakage energy consumption, which almost offsets the sacrifice on dynamic energy consumption.

- Compared with EA-TS, ETA-TS achieves a better optimization result by considering not only the dynamic power consumption (that EA-TS considers), but also the leakage power consumption and the thermal dissipation factors into when assigning tasks to processors. As a result, ETA-TS can explore the optimization space more accurately and more sufficiently.

7. Conclusions

Taking the strong leakage/temperature dependency into consideration, an effective energy/thermal aware task scheduling algorithm called ETA-TS is presented for real-time applications with precedence-constrained tasks on heterogeneous MPSoC systems. First, ETA-TS considers not only the power factors but also the thermal factors in the task assignment heuristic to balance the energy/thermal loads of processors. By this way, it demonstrates more effective temperature minimization while keeping a competitive energy consumption compared with the state-of-the-art approaches. Second, ETA-TS evenly reserves DVFS optimization space for tasks while guaranteeing the schedulability through transforming the deadline of the application to each task according to their levels. By this way, it can better minimize the temperature and the energy consumption through balancing the energy/thermal loads of tasks. Third, ETA-TS explores the optimization space by reducing or eliminating the waiting between tasks with the same successor task in a thermal aware way. With the three efforts above, the proposed approach can sufficiently explore the optimization room for both temperature and energy consumption. Extensive experiments conducted on the real-world DAG-based applications show that, the proposed approach can reduce more temperature by up to 12°C (depending on the specific application and related parameters) while keeping a competitive energy consumption compared with the state-of-the-arts.

In our future work, we will explore the possibility of assigning the slack time to tasks according to their power dissipation factors, which can further minimize the temperature and energy consumption of the systems according to our previous studies. The challenge of this study mainly lies in the trade-off between this slack assignment and the DVFS optimization space while guaranteeing the tasks' schedulability.

Declaration of competing interest

The authors declare that we do not have any financial and personal relationships with other people or organizations that can inappropriately influence this work. There is no professional or other personal interest of any nature or kind in any product, service and/or company that might lead to a conflict of interest for any author of the manuscript entitled.

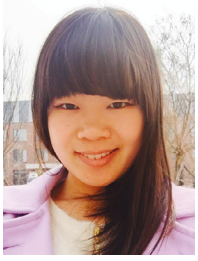
Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant nos. 61672143, 61433008.

References

[1] K. Cao, J. Zhou, T. Wei, M. Chen, S. Hu, K. Li, A survey of optimization techniques for thermal-aware 3d processors, J. Syst. Archit. 97 (2019) 397–415.
[2] M. Scrbak, M. Islam, K.M. Kavi, M. Ignatowski, N. Jayasena, Exploring the processing-in-memory design space, J. Syst. Archit. Embed. Syst. Des. 75 (2017) 59–67.
[3] K. Cao, J. Zhou, P. Cong, L. Li, T. Wei, M. Chen, S. Hu, X.S. Hu, Affinity-driven modeling and scheduling for makespan optimization in heterogeneous multiprocessor systems, IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. 37 (6) (2018) 1–14.
[4] A.A. Andreev, A. Sridhar, M.M. Sabry, M. Zapater, P. Ruch, B. Michel, D. Atienza, Powercool: simulation of cooling and powering of 3d MPSoCs with integrated flow cell arrays, IEEE Trans. Comput. 67 (1) (2018) 73–85.
[5] G. Xie, J. Jiang, Y. Liu, R. Li, K. Li, Minimizing energy consumption of real-time parallel applications using downward and upward approaches on heterogeneous systems, IEEE Trans. Ind. Informat. 13 (3) (2017) 1068–1078.
[6] R. Ahmed, P. Ramanathan, K.K. Saluja, On thermal utilization of periodic task sets in uni-processor systems, in: Proceedings of the IEEE 19th International Conference on Embedded and Real-Time Computing Systems and Applications, RTCSA, 2013, pp. 267–276. Taipei, Taiwan, August 19–21, 2013
[7] R. Ahmed, P. Ramanathan, K.K. Saluja, Necessary and sufficient conditions for thermal schedulability of periodic real-time tasks under fluid scheduling model, ACM Trans. Embed. Comput. Syst. 15 (3) (2016) 49:1–49:26.
[8] K. Li, Scheduling precedence constrained tasks with reduced processor energy on multiprocessor computers, IEEE Trans. Comput. 61 (12) (2012) 1668–1681.
[9] K. Li, Power and performance management for parallel computations in clouds and data centers, J. Comput. Syst. Sci. 82 (2) (2016) 174–190.
[10] J. Zhou, T. Wei, M. Chen, J. Yan, X.S. Hu, Y. Ma, Thermal-aware task scheduling for energy minimization in heterogeneous real-time MPSoC systems, IEEE Trans. CAD Integr. Circuits Syst. 35 (8) (2016) 1269–1282.
[11] J. Zhou, J. Yan, J. Chen, T. Wei, Peak temperature minimization via task allocation and splitting for heterogeneous MPSoC real-time systems, Signal Process. Syst. 84 (1) (2016) 111–121.
[12] T. Li, G. Yu, J. Song, Minimizing energy by thermal-aware task assignment and speed scaling in heterogeneous MPSoC systems, J. Syst. Archit. Embed. Syst. Des. 89 (2018) 118–130.
[13] A. Colin, A. Kandhalu, R. Rajkumar, Energy-efficient allocation of real-time applications onto heterogeneous processors, in: Proceedings of the IEEE 20th International Conference on Embedded and Real-Time Computing Systems and Applications, 2014, pp. 1–10. Chongqing, China, August 20–22, 2014
[14] A. Colin, A. Kandhalu, R. Rajkumar, Energy-efficient allocation of real-time applications onto single-ISA heterogeneous multi-core processors, Signal Process. Syst. 84 (1) (2016) 91–110.
[15] S. Saha, Y. Lu, J.S. Deogun, Thermal-constrained energy-aware partitioning for heterogeneous multi-core multiprocessor real-time systems, in: Proceedings of the IEEE International Conference on Embedded and Real-Time Computing Systems and Applications, RTCSA, 2012, pp. 41–50. 2012, Seoul, Korea (South), August 19–22, 2012
[16] G. Quan, V. Chaturvedi, Feasibility analysis for temperature-constraint hard real–time periodic tasks, IEEE Trans. Ind. Informat. 6 (3) (2010) 329–339.
[17] J. Zhou, J. Yan, K. Cao, Y. Tan, T. Wei, M. Chen, G. Zhang, X. Chen, S. Hu, Thermal-aware correlated two-level scheduling of real-time tasks with reduced processor energy on heterogeneous MPSoC, J. Syst. Archit. 82 (0) (2018) 1–11.
[18] J. Knechtel, J. Lienig, I.A.M. Elfadel, Multi-objective 3d floorplanning with integrated voltage assignment, ACM Trans. Des. Autom. Electr. Syst. 23 (2) (2018) 22:1–22:27.
[19] D. Saha, S. Sur-Kolay, Multi-objective optimization of placement and assignment of TSVs in 3D ICs, in: Proceedings of the 30th International Conference on VLSI Design and 16th International Conference on Embedded Systems, VLSID, 2017, pp. 372–377. Hyderabad, India, January 7–11, 2017
[20] Q. Xu, S. Chen, Fast thermal analysis for fixed-outline 3d floorplanning, Integration 59 (2017) 157–167, doi:10.1016/j.vlsi.2017.06.013.
[21] H. Han, J. Chung, J. Yang, READ: reliability enhancement in 3d-memory exploiting asymmetric SER distribution, IEEE Trans. Comput. 67 (8) (2018) 1193–1201.
[22] H.H. Shin, Y.M. Park, D. Choi, B.J. Kim, D. Cho, E. Chung, EXTREME: exploiting page table for reducing refresh power of 3d-stacked DRAM memory, IEEE Trans. Comput. 67 (1) (2018) 32–44.
[23] M. Guan, L. Wang, Improving DRAM performance in 3-d ICs via temperature aware refresh, IEEE Trans. VLSI Syst. 25 (3) (2017) 833–843.
[24] A. Asad, O. Ozturk, M. Fathy, M.R. Jahed-Motlagh, Optimization-based power and thermal management for dark silicon aware 3d chip multiprocessors using heterogeneous cache hierarchy, Microprocess. Microsyst. Embed. Hardw. Des. 51 (2017) 76–98.
[25] D. Li, J. Wu, Minimizing energy consumption for frame-based tasks on heterogeneous multiprocessor platforms, IEEE Trans. Parallel Distrib. Syst. 26 (3) (2015) 810–823.
[26] M.A. Awan, S.M. Petters, Energy-aware partitioning of tasks onto a heterogeneous multi-core platform, in: Proceedings of the 19th IEEE Real-Time and Embedded Technology and Applications Symposium, RTAS, 2013, pp. 205–214. Philadelphia, PA, USA, April 9–11, 2013
[27] T. Tsai, Y. Chen, Thermal-aware real-time task scheduling for three-dimensional multicore chip, in: Proceedings of the ACM Symposium on Applied Computing, SAC, 2012, pp. 1618–1624. Riva, Trento, Italy, March 26–30, 2012
[28] V. Chaturvedi, A.K. Singh, W. Zhang, T. Srikanthan, Thermal-aware task scheduling for peak temperature minimization under periodic constraint for 3d-MPSoCs, in: Proceedings of the 25nd IEEE International Symposium on Rapid System Prototyping, RSP, 2014, pp. 107–113. New Delhi, India, October 16–17, 2014
[29] Y. Cheng, L. Zhang, Y. Han, X. Li, Thermal-constrained task allocation for interconnect energy reduction in 3-d homogeneous MPSoC, IEEE Trans. VLSI Syst. 21(2) 239–249.
[30] S. Jin, Y. Wang, T. Liu, On optimizing system energy of voltage-frequency island based 3-d multi-core SoCs under thermal constraints, Integration 48 (2015) 36–45.
[31] M. Hu, J. Luo, Y. Wang, M. Lukasiewycz, Z. Zeng, Holistic scheduling of real-time applications in time-triggered in-vehicle networks, IEEE Trans. Ind. Informat. 10 (3) (2014) 1817–1828.
[32] B. Zhao, H. Aydin, D. Zhu, Shared recovery for energy efficiency and reliability enhancements in real-time applications with precedence constraints, ACM Trans. Des. Autom. Electr. Syst. 18 (2) (2013) 23:1–23:21.

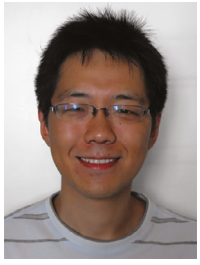
[33] H. Topcuoglu, S. Hariri, M. Wu, Performance-effective and low-complexity task scheduling for heterogeneous computing, IEEE Trans. Parallel Distrib. Syst. 13 (3) (2002) 260–274.
[34] G. Xie, R. Li, K. Li, Heterogeneity-driven end-to-end synchronized scheduling for precedence constrained tasks and messages on networked embedded systems, J. Parallel Distrib. Comput. 83 (2015) 1–12.
[35] J. Liu, K. Li, D. Zhu, J. Han, K. Li, Minimizing cost of scheduling tasks on heterogeneous multicore embedded systems, ACM Trans. Embed. Comput. Syst. 16 (2) (2017) 36:1–36:25.
[36] W. Liao, L. He, K.M. Lepak, Temperature and supply voltage aware performance and power modeling at microarchitecture level, IEEE Trans. CAD Integr. Circuits Syst. 24 (7) (2005) 1042–1053.
[37] Y. Liu, R.P. Dick, L. Shang, H. Yang, Accurate temperature-dependent integrated circuit leakage power estimation is easy, in: Proceedings of the Design, Automation and Test in Europe Conference and Exposition, DATE, 2007, pp. 1526–1531. Nice, France, April 16–20, 2007
[38] T. Chantem, X.S. Hu, R.P. Dick, Temperature-aware scheduling and assignment for hard real-time applications on MPSoCs, IEEE Trans. VLSI Syst. 19 (10) (2011) 1884–1897.
[39] K. Skadron, M.R. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, D. Tarjan, Temperature-aware microarchitecture: modeling and implementation, TACO 1 (1) (2004) 94–125.
[40] G. Xie, G. Zeng, L. Liu, R. Li, K. Li, High performance real-time scheduling of multiple mixed-criticality functions in heterogeneous distributed embedded systems, J. Syst. Archit. Embed. Syst. Des. 70 (2016) 3–14.
[41] Q. Huang, S. Su, J. Li, P. Xu, K. Shuang, X. Huang, Enhanced energy-efficient scheduling for parallel applications in cloud, in: Proceedings of the 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, 2012, pp. 781–786. CCGrid 2012, Ottawa, Canada, May 13–16, 2012
[42] Z. Tang, L. Qi, Z. Cheng, K. Li, S.U. Khan, K. Li, An energy-efficient task scheduling algorithm in DVFS-enabled cloud environment, J. Grid Comput. 14 (1) (2016) 55–74.
[43] T. Li, T. Zhang, G. Yu, J. Song, Temperature minimization and thermal-driven scheduling for real-time periodic tasks, J. Signal Process. Syst. (2018) 1–16.
[44] S. Bharathi, A. Chervenak, E. Deelman, G. Mehta, M.-H. Su, K. Vahi, Characterization of scientific workflows, in: Proceedings of the Third Workshop on Workflows in Support of Large-Scale Science, 2008, pp. 1–10.
[45] G. Juve, A.L. Chervenak, E. Deelman, S. Bharathi, G. Mehta, K. Vahi, Characterizing and profiling scientific workflows, Future Gener. Comput. Syst. 29 (3) (2013) 682–692.
[46] H. Wu, X. Hua, Z. Li, S. Ren, Resource and instance hour minimization for deadline constrained DAG applications using computer clouds, IEEE Trans. Parallel Distrib. Syst. 27 (3) (2016) 885–899.
[47] R. Medina, E. Borde, L. Pautet, Scheduling multi-periodic mixed-criticality dags on multi-core architectures, in: Proceedings of the IEEE Real-Time Systems Symposium, RTSS, 2018, pp. 254–264. Nashville, TN, USA, December 11–14, 2018
[48] T. Wu, H. Gu, J. Zhou, T. Wei, X. Liu, M. Chen, Soft error-aware energy-efficient task scheduling for workflow applications in DVFS-enabled cloud, J. Syst. Archit. Embed. Syst. Des. 84 (2018) 12–27.
[49] S. Pagani, J. Chen, M. Shafique, J. Henkel, Matex: efficient transient and peak temperature computation for compact thermal models, in: Proceedings of the Design, Automation & Test in Europe Conference & Exhibition, DATE, 2015, pp. 1515–1520. Grenoble, France, March 9–13, 2015
[50] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, M.R. Stan, Hotspot: a compact thermal modeling methodology for early-stage VLSI design, IEEE Trans. VLSI Syst. 14 (5) (2006) 501–513.



Tiantian Li received the Ph.D. degree from Northeastern University, China, in 2018. She currently works in Zhejiang University of Technology, China. Her fields of interest are energy efficient computing, distributed computing, and dynamic thermal and power management in real-time systems.

Tianyu Zhang received the Ph.D. degree from Northeastern University, China, in 2018. He was a visiting scholar in the Department of Computer Science and Engineering at the University of Notre Dame during 2015.11 - 2017.05. He currently works in Qingdao University, China. His research interests include real-time systems, cyber-physical systems and wireless sensor networks.

Ge Yu received the Ph.D. degree from Kyushu University, Japan, in 1996. He is currently a professor in Northeastern University, China. His research interests include distributed and parallel database, big data storage and management, real-time systems optimization, etc. He has published more than 200 papers in refereed journals and conferences. He is a member of the IEEE Computer Society, IEEE, ACM, and a Senior Member of the China Computer Federation.



Jie Song received the Ph.D. degree from Northeastern University, China, in 2010. He is currently an associate professor in Northeastern University, China. His research interests include green computing, big data management and machine learning.

Jing Fan received the Ph.D. degree from Zhejiang University, China, in 2003. She is now a Professor of School of Computer Science and Technology, and Director of Institute of Computer Software at Zhejiang University of Technology, China. She is a Director of China Computer Federation (CCF), and Member of CCF Technical Committee on Service Computing. She also is Vice-Director of Key Laboratory of Visual Media Intelligent Processing Technology of Zhejiang Province, China. Her current research interests include Service Computing, Software Middleware, Virtual Reality and Visualization.