Energy Efficient Task Partitioning based on the Single Frequency Approximation Scheme

Santiago Pagani and Jian-Jia Chen
Department of Informatics, Karlsruhe Institute of Technology (KIT), Germany
E-mail: santiago.pagani@kit.edu, jian-jia.chen@kit.edu

***Abstract*—Energy-efficiency is a major concern in modern computing systems. For such systems, the presence of multiple voltage islands, where the voltage of each island can change independently and all cores in an island share the same supply voltage at any given time, is an expected compromise between global and per-core Dynamic Voltage and Frequency Scaling (DVFS). This paper focuses on energy minimization for a set of periodic tasks assigned on a voltage island. We present a simple and practical solution, that assigns the tasks onto cores in the island and then applies a DVFS schedule, particularly the Single Frequency Approximation (SFA) scheme. Furthermore, we provide thorough theoretical analysis of our solution, in terms of energy efficiency, against the optimal task partitioning and optimal DVFS schedule, especially for the state-of-the-art designs, that have a few number of cores per voltage island. The analysis shows that, our task partitioning scheme combined with SFA is a good and practical solution for energy efficiency. Particularly, when the number of cores in each voltage island is limited, the approximation factor is at most 2.01 (2.29, 2.55, 2.80, respectively) when the dynamic power consumption is a cubic function of the frequency and the islands have up to 4 (8, 16, 32, respectively) cores. Moreover, with non-negligible overhead for sleeping, further combination with any uni-core procrastination algorithm that consumes no more energy than keeping a core idle when it has no workload in its ready queue, increases the approximation factor by at most 1.**

I. Introduction

Energy-efficient and low-power design is an important issue in today's and next generation computing systems, for example to prolong the battery lifetime of embedded systems or to reduce the power bills for servers. This is one of the main motivations why single-core computing systems have moved to multi-core platforms, mainly to balance the power consumption and computation performance.

The dynamic power consumption (mainly generated by switching activities) and the static power consumption (mainly generated by the leakage current) are the two major sources of power consumption in CMOS processors, as shown in the literature, e.g. [11]. Executing at low frequencies, by using the *Dynamic Voltage and Frequency Scaling* (DVFS) technique, is in most cases the best solution for energy minimization when the static power consumption is negligible, due to the convexity of the power consumption function.

Nevertheless, executing tasks at frequencies lower than a *critical frequency* might consume more energy for execution in systems with non-negligible static power. This happens because the static energy consumption grows with longer execution times when running at lower frequencies, and it might dominate the energy consumption. This motivates the combination of DVFS and *Dynamic Power Management* (DPM), so that cores can be slowdown and then shutdown after finishing their workloads, e.g., [1], [10], [11].

Even though task scheduling and task partitioning have been explored in the past decade for energy reduction, most researches either assume per-core DVFS (each core can change its own supply voltage and frequency independently from other cores), e.g., [2], [3], [4], [13], [16], or consider global DVFS (there is only one shared supply voltage and the maximum frequency of all cores is limited by it), e.g., [5], [15], [17]. Next-generation many-core systems make a compromise between such extreme cases, adopting multi-core architectures with several voltage islands, in which the cores on a voltage island are consolidated as a cluster, sharing a supply voltage (modifiable by DVFS at any time), but individual islands can have different voltages [7]. For example, Intel's Single-chip Cloud Computer (SCC) [8], [9], is a research multi-core platform with such a feature.

Motivation: For multi-core systems with a shared supply voltage in a voltage island (or a system with a global supply voltage) and periodic real-time tasks, the task partitioning and DVFS schedule play a major role in energy minimization.

Considering the convexity of the dynamic power consumption of a core, the energy consumption of executing some workload in one core at a specific frequency, is bigger than executing the same workload, perfectly distributed in two cores, at half of the frequency. This suggests that in most cases, a task partitioning solution that balances the workloads throughout all cores in the island would have the minimum dynamic energy consumption. However, deriving such a balanced solution is with high complexity and not feasible for most practical cases. Thus, a good option is to use a polynomial time algorithm based on load balancing, e.g., the *Largest-Task-First* (LTF) strategy [17].

Moreover, once the tasks are assigned onto the cores in the island, it is necessary to choose a policy that decides the voltage of the island and the frequencies of the cores for execution. For periodic tasks, the *simplest* and *most intuitive* strategy is to use a single voltage and frequency for executing, particularly, the lowest voltage and frequency that satisfies the timing constraints. Such a scheme, denoted as the *Single Frequency Approximation* (SFA) scheme [14], has linear time complexity, only from evaluating the core with the highest cycle utilization. Naturally, SFA is not the optimal DVFS scheduling strategy for energy minimization, but it significantly reduces the overhead for changing the supply voltage of the islands and frequencies of cores, given that SFA does not require voltage/frequency changes at run time. Moreover, any uni-core DPM technique can be adopted individually in each core together with SFA

1052-8725/13 $26.00 ©2013 IEEE. Published in the 2013 IEEE 34th Real-Time Systems Symposium (RTSS), Vancouver, BC, Canada, December 2013.
Available in the IEEE Xplore Digital Library.

without additional effort, because no frequency alignment for DVFS between cores is needed under SFA.

Combining LTF and SFA for energy minimization is a practical (easy to implement) solution. However, its worst-case performance in terms of energy efficiency is an open problem.

Objective: We focus on energy minimization on a voltage island. We consider a set of periodic tasks assigned to such an island. The objective of this paper is to present and theoretically analyse a simple and practical solution, that assigns the tasks onto cores in the voltage island (task partitioning) and then applies a DVFS schedule (particularly SFA) to decide the voltage of the island and the frequencies of the cores. For the theoretical analysis, we compare our scheme against the optimal task partitioning and optimal DVFS schedule, especially for the state-of-the-art designs, that have a few number of cores per voltage island.

Our Contributions: For periodic real-time tasks in a single voltage island, our contributions are:

- We propose a task partitioning scheme based on LTF, called *Double-Largest-Task-First* (DLTF) scheme, which saves energy for idling and is well suited to combine with SFA for energy minimization. This scheme is presented in Section V.
- We reveal the effectiveness of combining DLTF and SFA for energy efficiency. We show that such a solution has an approximation factor (worst-case ratio of the energy consumption for DLTF combined with SFA, against the optimal task partitioning and optimal DVFS schedule) that can be bounded. Specifically, the factor depends on the *parameters of the power consumption function* and the *number of cores* in the voltage island. When the island has up to 4 (8, 16, 32, respectively) cores, the approximation factor for execution is at most 2.01 (2.29, 2.55, 2.80, respectively). The analysis is presented in Section VI and Section VII.
- In Section VIII, we perform numerical evaluations of several specific case studies, without any approximation in the lower bound for the optimal energy consumption, in order to show the gap between the theoretical analysis and concrete cases.
- In Section IX, when the *overhead for sleeping* is non-negligible, we show the effectiveness of combining DLTF with SFA and *any uni-core procrastination algorithm* that does not consume more energy than putting a core always in idle mode when it has no workload to execute in its ready queue. We show that the overall approximation factor is increased by at most 1 from the previous analysis.

Related Work: Power-aware and energy-efficient scheduling for homogeneous multi-core systems has been widely explored for real-time embedded systems with per-core DVFS, e.g., [2], [3], [4], [13], [16]. In [4], it was shown that applying the *Largest-Task-First* (LTF) strategy for task mapping results in solutions with approximation factors, in terms of energy consumption, in which the factors depend on the hardware platforms. Specifically, by turning off a processor to reduce the energy consumption in homogeneous multiprocessor systems, Xu et al. [16] and Chen et al. [4] propose polynomial-time algorithms to derive task mappings that try to execute at a *critical frequency*. For homogeneous multiprocessor systems with discrete voltage levels and frequencies, Moreno and de Niz [13] present an algorithm that runs in polynomial time and computes the optimal voltage and frequency assignment for systems with uniform frequency steps and negligible static/leakage power consumption. Based on VLSI circuit simulations, Herbert and Marculescu [7] suggest that per-core DVFS suffers from complicated design problems, making it costly for implementation even though it is energy-efficient.

When considering only one global supply voltage, Yang et al. [17] provide results for voltage scaling, to minimize the energy consumption in systems with negligible static power consumption and *frame-based real-time tasks* (all the tasks have the same arrival time and period). Such an approach is highly restricted and it is hard to extend to handle systems with non-negligible static power consumption or periodic real-time tasks (tasks have different arrival times and periodicity). The assumptions in [17] are relaxed in [5], [15], by considering periodic real-time tasks with non-negligible static and voltage-independent power consumptions and non-negligible overhead for turning to low-power idle modes. In [5], the number of active cores is decided first and the frequencies of the active cores are decided in a second stage. Nevertheless, the work in [5] lacks theoretical analysis for the worst-case performance of their approach for energy minimization. Additionally, Seo et al. [15] dynamically balance the task loads of multiple cores and adjust the number of active cores, to optimize the power consumption for execution and to reduce the leakage power consumption for low workloads. For sets of periodic real-time tasks, the analysis in [14] derives the worst-case approximation factor for the *Single Frequency Approximation* (SFA) scheme, in terms of energy consumption.

II. System Model and Problem Definition

This section reviews the power and energy model adopted for the rest of the paper and defines the problem.

A. Hardware Model

This paper focuses on *a single voltage island*, where all the cores in the island have the same supply voltage and run at the same frequency, at any given time point, e.g., one voltage island of SCC [8], [9].[1] The system can change the voltage and frequency of the island by adopting DVFS. This model was also adopted in [5], [17]. For a core to support a frequency, the supply voltage in the island has to be adjusted accordingly, in particular to the least available supply voltage such that stable execution on the core is achievable for the frequency. The available frequencies are in the range of $[s_{\min}, s_{\max}]$.[2]

We denote the power consumption of a core executing a certain task at frequency s as $P(s)$, and the energy consumption during time interval Δt at frequency s is denoted as $E(s) = P(s) \cdot \Delta t$. Particularly, we use

$$P(s) = \beta + \alpha s^{\gamma}, \tag{1}$$

[1] The approximation factor for a system with multiple voltage islands is equal to the approximation factor on one voltage island. The detailed analysis for this argument can be found in Section XI in [14].

[2] For systems with discrete frequencies, all the analysis still holds simply by multiplying the approximation factor by $\psi = \max_{1<i\leq F} \frac{P(f_i)\cdot f_{i-1}}{P(f_{i-1})\cdot f_i}$, where $\{f_1, f_2, \ldots, f_F\}$ are the available frequencies. For example, for $P(s)$ in Equation (1), with $\alpha = 1.76 \frac{\text{Watts}}{\text{GHz}^3}$, $\beta = 0.5$ Watts, $\gamma = 3$, and available frequencies $\{0.1 \text{ GHz}, 0.2 \text{ GHz}, \ldots, 3.0 \text{ GHz}\}$, then ψ is equal to 1.14.

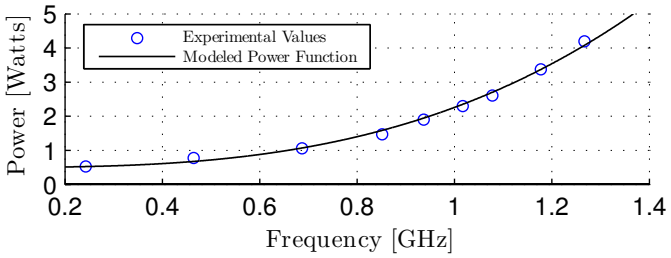

Fig. 1: Power model from experimental results in [8].

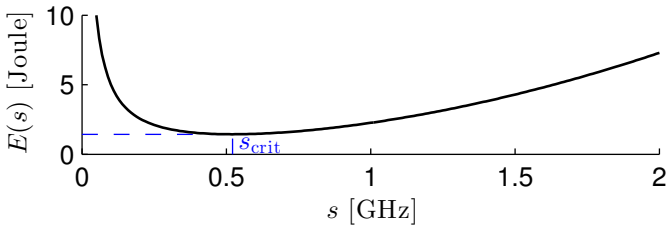

Fig. 2: Energy consumption function for a single core executing 10^9 computer cycles with $\alpha = 1.76 \frac{\text{Watts}}{\text{GHz}^3}$, $\gamma = 3$, $\beta = 0.5$ Watts, and $s_{\text{crit}} = 0.52$ GHz.

where $\alpha > 0$ is a constant dependent on the effective switching capacitance, $\gamma > 1$ is related to the hardware, and $\beta \geq 0$ represents the static power consumption. This power consumption function has been widely used in the literature, e.g., [2], [5], [14], [16], [17]. Therefore, we know that $P(s)$ is a convex and increasing function with respect to s.

During interval Δt, a core running at frequency s executes a certain amount Δc of core cycles, such that $\Delta t = \frac{\Delta c}{s}$ and

$$E(s) = (\beta + \alpha s^{\gamma}) \frac{\Delta c}{s}. \tag{2}$$

This energy consumption is merely a convex function. Hence, by setting the first order derivative of Equation (2) with respect to s to zero, the minimum value for $E(s)$ is found when s is $\sqrt[\gamma]{\frac{\beta}{(\gamma-1)\alpha}}$. In order to consider the case when such a value is smaller than s_{min}, we define the *critical frequency* as $s_{\text{crit}} = \max\left\{s_{\text{min}}, \sqrt[\gamma]{\frac{\beta}{(\gamma-1)\alpha}}\right\}$. The *critical frequency* represents the frequency that minimizes the energy consumption for execution when the overhead for sleeping is considered negligible, as also shown in [3], [11].

For example, to show the validity of the power model, take the 48-core system developed in [8]. The power consumption function from Equation (1) was used in [14] to model the experimental results from [8], resulting in power parameters $\alpha = 1.76 \frac{\text{Watts}}{\text{GHz}^3}$, $\gamma = 3$, $\beta = 0.5$ Watts, and $s_{\text{crit}} = 0.52$ GHz. Figure 1 shows this power model and the original experimental measurements from [8]. Moreover, Figure 2 illustrates function $E(s)$ from Equation (2) for a single core executing 10^9 computer cycles with the same α, γ and β values, where it is easy to observe the concept of a *critical frequency*.

When a core finishes executing all its workload in the ready queue, it has to wait until a new task instance arrives. During this waiting interval, the core can choose to stay idle (in execution mode), consuming β power. Moreover, the core can also choose to go to a low-power mode that consumes $\beta' \geq 0$ power. As we can transfer the power consumption β' to the power consumption of the voltage island for being active, without loss of generality, we can set $P(s)$ as $P(s) - \beta'$, such that we can disregard the effect of the power consumption of a core in a low-power mode. Nevertheless, the core consumes some energy during the transition process of entering/leaving the low-power mode. Given that we handle periodic tasks, which always go back to the execution mode after a certain amount of time, we denote *overhead for sleeping* to the summation of the energy consumption both for entering and leaving the low-power mode.

When the duration of the waiting interval until the arrival of a new task instance is short enough, idling is more energy efficient than sleeping. Contrarily, when the interval is sufficiently long, sleeping results in higher energy savings. Thus, we define the *break-even time* as the time such that the energy consumption for idling is equal to the *overhead for sleeping*.

The analysis in Section VI and Section VII considers negligible *overhead for sleeping*, for which the *break-even time* is 0 and every core goes to sleep immediately when it has no workload to execute. The analysis in Section IX extends such results by considering non-negligible *overhead for sleeping*.

B. Task Model

We consider N periodic real-time tasks with implicit deadlines, i.e., $\{\tau_1, \tau_2, \ldots, \tau_N\}$. Every task τ_j releases an infinite number of task instances with period (and relative deadline) p_j and every instance has worst-case execution cycles e_j. We consider partitioned scheduling, in which each task is assigned onto a core, that is, when a task instance arrives to the system it is executed on the assigned core. Specifically, we use *Earliest-Deadline-First* (EDF) scheduling, where the task instance with the earliest absolute deadline on each core has the highest priority. The least common multiple among all periods is called the *hyper-period*, denoted as L.

After the task partitioning is completed by using M cores, all N tasks are grouped into M task sets $\{\mathbf{T}_1, \mathbf{T}_2, \ldots, \mathbf{T}_M\}$. Without loss of generality, we assume that task set $\mathbf{T}_i$ is assigned on core i, and we define its *cycle utilization* as $w_i = \sum_{\tau_j \in \mathbf{T}_i} \frac{e_j}{p_j}$. Moreover, $\sum_{j=1}^{N} \frac{e_j}{p_j} = \sum_{i=1}^{M} w_i$ denoted as the *total cycle utilization*. By defining w_0 for simplicity and without loss of generality, we order the cores such that $0 = w_0 \leq w_1 \leq w_2 \leq \cdots \leq w_M$. It has been well studied, e.g., [12], that executing core i at frequency higher than or equal to w_i with EDF will meet the timing constraint.

C. Problem Definition

For N periodic tasks that are assigned into a voltage island, the objective of this paper is to present and theoretically analyse a practical solution, that assigns the N tasks onto the M cores in the island and then applies a DVFS schedule, such that the energy consumption in the voltage island is minimized.

We consider the task partitioning that results in the minimum energy consumption by applying the optimal DVFS schedule as the *optimal task partitioning*. Obtaining the *optimal task partitioning* for energy minimization is an $\mathcal{NP}$-hard

problem [17]. For notational purposes, the task sets of the optimal task partitioning are denoted as $\{\mathbf{T}_1^*, \mathbf{T}_2^*, \ldots, \mathbf{T}_M^*\}$ with the corresponding cycle utilizations $0 = w_0^* \leq w_1^* \leq w_2^* \leq \cdots \leq w_M^*$, ordered without loss of generality. In this paper we use a practical task partitioning scheme based on the *Largest-Task-First* (LTF) strategy, which we call *Double-Largest-Task-First* (DLTF). Partitioning task with DLTF results in task sets $\left\{\mathbf{T}_1^{\text{DLTF}}, \mathbf{T}_2^{\text{DLTF}}, \ldots, \mathbf{T}_M^{\text{DLTF}}\right\}$ with the corresponding cycle utilizations $0 = w_0^{\text{DLTF}} \leq w_1^{\text{DLTF}} \leq w_2^{\text{DLTF}} \leq \cdots \leq w_M^{\text{DLTF}}$, ordered without loss of generality. The detailed description and properties of DLTF are presented in Section V.

After the task partitioning by applying DLTF, we consider the *Single Frequency Approximation* (SFA) scheme as our DVFS schedule [14]. Under SFA, all cores in the voltage island always execute at single frequency s_u and each core enters a low-power mode after executing all the workload in its ready queue. The voltage of the island is set to the minimum available voltage so that all M cores in the island can execute stably at frequency s_u. The time complexity of SFA is $O(M)$ to ensure the feasibility, where M is the number of cores in the voltage island and this complexity comes only from evaluating the highest cycle utilization. Clearly, s_u must be at least w_M^{DLTF} to ensure feasible schedules. For completeness, preliminary results for SFA will be presented in Section III.

Our proposed solution is therefore to use DLTF for task partitioning combined with SFA as DVFS schedule. For the theoretical analysis, we provide an approximation factor of such an approach in terms of energy consumption, against the optimal task partitioning and optimal DVFS solution, defined $\text{AF}_{\text{SFA}}^{\text{DLTF}}$ and expressed as

$$\text{AF}_{\text{SFA}}^{\text{DLTF}} = \max \frac{E_{\text{SFA}}^{\text{DLTF}}}{E_{\text{OPT}}^*} \leq \max \frac{E_{\text{SFA}}^{\text{DLTF}}}{E_{\downarrow}^*}, \tag{3}$$

where E_{OPT}^* is the optimal energy consumption for the optimal DVFS schedule and the optimal task partitioning during a hyper-period, $E_{\text{SFA}}^{\text{DLTF}}$ is the energy consumption for our partitioning scheme under SFA during a hyper-period, and $E_{\downarrow}^*$ is a lower bound for the optimal energy consumption for the optimal task partitioning and any feasible DVFS schedule during a hyper-period. Since, E_{OPT}^* is not easily obtained, in the analyses we use its lower bound $E_{\downarrow}^*$, that should not be very far away from E_{OPT}^*.

Note that SFA does not require any capability of voltage/frequency scaling at run time, as it only uses one frequency. However, to explore the approximation factor we need $E_{\downarrow}^*$, in which changing the supply voltage and frequency of the island is with negligible overhead and the available frequencies are continuous between $(0, s_{max}]$. This approach results in a safe lower bound for the optimal energy consumption.

III. Preliminary Results for SFA

In order to obtain the approximation factor for DLTF combined with SFA, three things are required:

- A lower bound for the energy consumption of the optimal task partitioning and the optimal DVFS schedule, i.e., $E_{\downarrow}^*$.
- The energy consumption in the voltage island by using SFA, after partitioning task with DLTF, i.e., $E_{\text{SFA}}^{\text{DLTF}}$.
- The approximation factor of SFA for a single voltage island without considering task partitioning, denoted as $\text{AF}_{\text{SFA}}^{\text{n.p.}}$.

The work in [14] provides comprehensive analysis for the approximation factor of SFA for a *given task partitioning*, i.e., the group of task sets is fixed and task partitioning is not considered. Moreover, [14] derives preliminary equations needed to compute the approximation factor $\text{AF}_{\text{SFA}}^{\text{DLTF}}$ in Equation (3). For completeness, this section summarises the results in [14].

A. Lower Bound Energy Consumption

By applying the Lagrange Multiplier Method, the work in [14] provides a lower bound for the energy consumption of *any task partitioning*. Here we are interested in the lower bound of the optimal energy consumption, denoted as $E_{\downarrow}^*$. When $\beta = 0$, by the results in [14], we have

$$E_{\downarrow}^{*(\beta=0)} = \alpha L \left[\sum_{i=1}^{M} \left(w_i^* - w_{i-1}^*\right) \sqrt[\gamma]{M-i+1}\right]^{\gamma}. \tag{4}$$

For the case that $\beta \neq 0$, from [14], we have

$$E_{\downarrow}^*\left(w_M^*\right) = \begin{cases} \alpha\gamma L \left(s_{\text{crit}}{}^{\gamma-1}\right) \sum_{i=1}^{M} w_i^* & \text{if } w_M^* \leq s_{\text{dyn}} \\ \alpha L \left[\sum_{i=1}^{M} \left(w_i^* - w_{i-1}^*\right) \sqrt[\gamma]{M-i+1}\right]^{\gamma} & \text{otherwise,} \end{cases} \tag{5}$$

where s_{dyn} is an auxiliary frequency[3] with $s_{\text{crit}} < s_{\text{dyn}} < s_{\max}$.

B. Energy Consumption for SFA

After partitioning task by using DLTF, the frequency chosen by SFA is either (1) s_{crit} if $w_M^{\text{DLTF}} \leq s_{\text{crit}}$, or (2) w_M^{DLTF} otherwise. The lowest energy consumption for SFA as a function of w_M^{DLTF} is defined as $E_{\text{SFA}}^{\text{DLTF}}\left(w_M^{\text{DLTF}}\right)$.

$$E_{\text{SFA}}^{\text{DLTF}}\left(w_M^{\text{DLTF}}\right) = \begin{cases} \alpha\gamma L \left(s_{\text{crit}}{}^{\gamma-1}\right) \sum_{i=1}^{M} w_i^{\text{DLTF}} & \text{if } w_M^{\text{DLTF}} \leq s_{\text{crit}} \\ \frac{L}{w_M^{\text{DLTF}}} \left(\beta + \alpha {w_M^{\text{DLTF}}}^{\gamma}\right) \sum_{i=1}^{M} w_i^{\text{DLTF}} & \text{otherwise} \end{cases} \tag{6}$$

Clearly, *when $w_M^{DLTF} \leq s_{crit}$, DLTF combined with SFA consumes no more energy than the lower bound $E_{\downarrow}^*$*. Hence, the relation between $s_{\min}$ and $\sqrt[\gamma]{\frac{\beta}{(\gamma-1)\alpha}}$ is of no consequence. Thus, for simplicity in presentation, for the rest of this paper we consider that $s_{\min} = 0$ and therefore set s_{crit} to $\sqrt[\gamma]{\frac{\beta}{(\gamma-1)\alpha}}$. *This results in safe upper bounds for the general case.*

In the case that $\beta = 0$, then s_{crit} is also zero and only the dynamic energy consumption is present for SFA, which is

$$E_{\text{SFA}}^{\text{DLTF}(\beta=0)}\left(w_M^{\text{DLTF}}\right) = \alpha L \left({w_M^{\text{DLTF}}}^{\gamma-1}\right) \sum_{i=1}^{M} w_i^{\text{DLTF}}. \tag{7}$$

Corollary 1: Regardless of the value of β, the energy consumption of all the cores in a voltage island under SFA, for a fixed hardware and a given hyper-period, only depends on the value of w_M and the *total cycle utilization* $\sum_{i=1}^{M} w_i$. In other words, it does not matter how the tasks are distributed among task sets $\{\mathbf{T}_1, \ldots, \mathbf{T}_{M-1}\}$ as long as $w_1 \leq w_2 \leq \cdots \leq w_M$.

[3] The consideration of s_{dyn} is an approximation made in [14] to provide a closed analytical expression for $E_{\downarrow}^*$. Under a specific case study, with known tasks and hardware parameters, a more precise lower bound can be obtained by solving Equation 9 in [14]. Such an approach is taken in Section VIII.

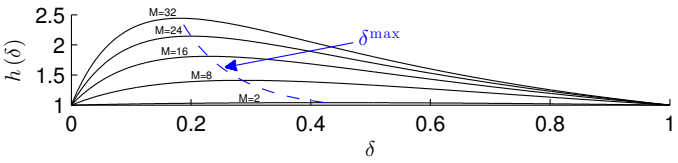

Fig. 3: $h(\delta)$ when $\gamma = 3$, highlighting $\delta^{\max}$ (from [14]).

C. Approximation Factor of SFA for a given task partition

By introducing the scaling factor $r_i = \frac{w_i}{w_M}$, the cycle utilization w_i of task set $\mathbf{T}_i$ for $i = 0, 1, \ldots, M$, can be rephrased as $w_i = r_i \cdot w_M$, where

$$0 = r_0 < r_1 \leq r_2 \leq \cdots \leq r_{M-1} \leq r_M = 1. \tag{8}$$

Then, by defining $H(r_0, \ldots, r_M)$ as

$$H(r_0, \ldots, r_M) = \frac{\sum_{i=1}^{M} r_i}{\left[\sum_{i=1}^{M} (r_i - r_{i-1}) \sqrt[\gamma]{M - i + 1}\right]^{\gamma}}, \tag{9}$$

we have the following lemma.

Lemma 1: For all $r_0, r_1 \ldots, r_M$, defined in Equation (8), and by defining δ as $\frac{\sum_{i=1}^{M-1} r_i}{M-1}$, with $\gamma > 1$, if task sets $\mathbf{T}_1, \mathbf{T}_2, \ldots, \mathbf{T}_{M-1}$ have the same utilization, i.e., $r_i = \delta$ for all $i = 1, 2, \ldots, M-1$, then we have

$$H(r_0, \ldots, r_M) \leq h(\delta) = \frac{1 - \delta + \delta M}{\left(1 - \delta + \delta \sqrt[\gamma]{M}\right)^{\gamma}} \leq h(\delta^{\max}), \tag{10}$$

where

$$\delta^{\max} = \frac{\gamma - 1 + M - \gamma \sqrt[\gamma]{M}}{(\gamma - 1)\left(M \sqrt[\gamma]{M} - M - \sqrt[\gamma]{M} + 1\right)}. \tag{11}$$

Proof: Under a fixed $\sum_{i=1}^{M-1} r_i$, function $H(r_0, \ldots, r_M)$ is maximized if and only if $\sum_{i=1}^{M-1} (r_i - r_{i-1}) \sqrt[\gamma]{M - i + 1}$ is minimized. Such a global minimum happens when $\delta = \frac{\sum_{i=1}^{M-1} r_i}{M-1} = r_{i-1} = r_i$ for all $r_1, \ldots, r_{M-1}$. Hence, by setting r_i to δ for $i = 1, 2, \ldots, M-1$, in Equation (9), we obtain function $h(\delta)$. Moreover, since $h(\delta)$ is a convex function of δ when $\gamma > 1$, by taking the first order derivative of $h(\delta)$ with respect to δ, its maximum value happens when δ is $\delta^{\max}$. The detailed proof can be found in Lemma 2 of [14]. ∎

Function $h(\delta)$, for $\gamma = 3$, is presented in Figure 3.

Lemma 2: Function $h(\delta) = \frac{1-\delta+\delta M}{\left(1-\delta+\delta \sqrt[\gamma]{M}\right)^{\gamma}}$ is strictly decreasing with respect to δ for $M \geq 2$, $\gamma > 1$ and $\delta \geq 0.5$.

Proof: From Equation (10) (illustrated in Figure 3), it is clear that $h(\delta)$ is a decreasing function with respect to δ when $\delta^{\max} \leq \delta \leq 1$, $M \geq 2$ and $\gamma > 1$. Moreover, $\delta^{\max}$ is a decreasing function with respect to M, for all $M \geq 2$ and $\gamma > 1$. Hence, the highest value of $\delta^{\max}$ for a given γ occurs when $M = 2$. Furthermore, $\delta^{\max}$ is an increasing function with respect to γ, for all $M \geq 2$ and $\gamma > 1$. Therefore, the highest value of $\delta^{\max}$ is obtained when $M = 2$ and $\gamma \to \infty$, which converges to $\frac{1}{\ln 2} - 1 = 0.443$. Finally, $h(\delta)$ is a decreasing function after 0.5 for any $M \geq 2$ and $\gamma > 1$. The detailed proof can be found in Lemma 4 of [14]. ∎

Algorithm 1 Largest-Task-First (LTF) strategy

Input: Tasks $\{\tau_1, \tau_2, \ldots, \tau_N\}$;
Output: Task sets $\{\mathbf{T}_1^{\text{LTF}}, \mathbf{T}_2^{\text{LTF}}, \ldots, \mathbf{T}_M^{\text{LTF}}\}$;
1: Sort all tasks in a non-increasing order of their cycle utilizations;
2: **for** $i = 1$ **to** M **do**
3: $\quad \mathbf{T}_i^{\text{LTF}} \leftarrow \emptyset$;
4: **end for**
5: **for** $j = 1$ **to** N **do**
6: $\quad$ Find the smallest w_i^{LTF};
7: $\quad \mathbf{T}_i^{\text{LTF}} \leftarrow \mathbf{T}_i^{\text{LTF}} + \{\tau_j\}$;
8: **end for**
9: Re-order $\mathbf{T}_i^{\text{LTF}}$ by a non-decreasing order of their cycle utilization;
10: Return $\{\mathbf{T}_1^{\text{LTF}}, \mathbf{T}_2^{\text{LTF}}, \ldots, \mathbf{T}_M^{\text{LTF}}\}$;

Considering Lemma 1, the approximation factor of SFA without considering task partitioning is presented in the following Theorems from [14].

Theorem 1: When $\beta = 0$, the approximation factor of SFA without considering task partitioning for periodic real-time tasks, denoted as $\text{AF}_{\text{SFA}}^{\text{n.p. }(\beta=0)}$, is

$$\text{AF}_{\text{SFA}}^{\text{n.p. }(\beta=0)} \leq h(\delta^{\max}),$$

where $\delta^{\max}$ is defined as a function of γ and M in Equation (11) and $h()$ is defined in Equation (10). Therefore, the approximation factor $\text{AF}_{\text{SFA}}^{\text{n.p. }(\beta=0)}$ only depends on γ and M.

Proof: For a given task partitioning, $\text{AF}_{\text{SFA}}^{\text{n.p. }(\beta=0)}$ is equal to $\max H(r_0, \ldots, r_M)$. Based on Equation (10), $H(r_0, \ldots, r_M) \leq h(\delta^{\max})$ and thus the theorem is proven. The detailed proof can be found in Theorem 1 of [14]. ∎

Theorem 2: When $\beta \neq 0$, the approximation factor of SFA without considering task partitioning for periodic real-time tasks, denoted as $\text{AF}_{\text{SFA}}^{\text{n.p.}}$, is

$$\text{AF}_{\text{SFA}}^{\text{n.p.}} \leq \frac{\gamma - 1}{\left[\gamma \gamma h(\delta^{\max})\right]^{\frac{1}{\gamma - 1}}} + h(\delta^{\max}),$$

where $\delta^{\max}$ is defined as a function of γ and M in Equation (11) and $h()$ is defined in Equation (10). Since $h(\delta^{\max})$ is only a function of γ and M, the approximation factor $\text{AF}_{\text{SFA}}^{\text{n.p.}}$ is independent of α and β.

Proof: The proof can be found in Theorem 2 of [14]. ∎

IV. Preliminary Results for Largest-Task-First

Our DLTF scheme is based on LTF. For completeness, we present the description and properties of LTF from [17].

A. Description of LTF

A good and widely used algorithm for task partitioning is the *Largest-Task-First* (LTF) strategy, with time complexity $O(N(\log N + \log M) + M)$ [17]. LTF partitions tasks $\{\tau_1, \tau_2, \ldots, \tau_N\}$ into M groups of task sets $\{\mathbf{T}_1^{\text{LTF}}, \mathbf{T}_2^{\text{LTF}}, \ldots, \mathbf{T}_M^{\text{LTF}}\}$ with the corresponding cycle utilizations $0 = w_0^{\text{LTF}} \leq w_1^{\text{LTF}} \leq w_2^{\text{LTF}} \leq \cdots \leq w_M^{\text{LTF}}$, ordered without loss of generality. Naturally, it holds that $\sum_{i=1}^{M} w_i^{\text{LTF}} = \sum_{i=1}^{M} w_i^*$, because the *total cycle utilization* of all tasks is constant, regardless of how they are partitioned. The pseudo-code for LTF is presented in Algorithm 1.

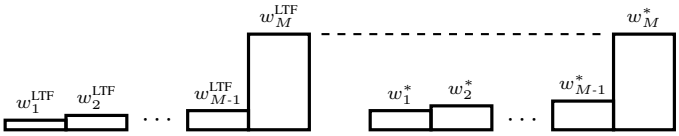

Fig. 4: Utilization relation for LTF when $w_M^* \geq w_M^{\text{LTF}}$.

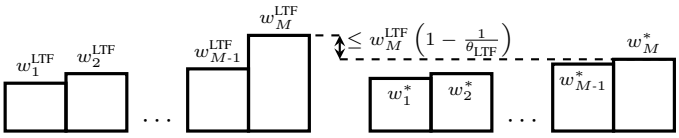

Fig. 5: Utilization relation for LTF when there are at least two tasks in the highest utilization task set and $w_M^* < w_M^{\text{LTF}}$.

B. Properties of LTF

The *Largest-Task-First* strategy is mainly a reformulation of the *Longest-Processing-Time* (LPT) algorithm from [6] for the Makespan problem. Consider the resulting task partitioning from LTF, previously defined as $\{\mathbf{T}_1^{\text{LTF}}, \mathbf{T}_2^{\text{LTF}}, \ldots, \mathbf{T}_M^{\text{LTF}}\}$, in which w_M^{LTF} is the maximum cycle utilization in the resulting solution. Similarly, consider the optimal task partitioning for energy minimization, previously defined as $\{\mathbf{T}_1^*, \mathbf{T}_2^*, \ldots, \mathbf{T}_M^*\}$, in which w_M^* is the maximum cycle utilization in the optimal solution. Clearly, if after the task partitioning there is *only one* task in $\mathbf{T}_M^{\text{LTF}}$, then $w_M^{\text{LTF}} \leq w_M^*$, since the cycle utilization of the task in $\mathbf{T}_M^{\text{LTF}}$ is the lower bound of the maximum utilization in the optimal task partitioning.

Moreover, if there are *at least two* tasks in $\mathbf{T}_M^{\text{LTF}}$, then it holds that

$$\frac{w_M^{\text{LTF}}}{w_M^*} \leq \theta_{\text{LTF}} = \frac{4}{3} - \frac{1}{3M}, \tag{12}$$

where θ_{LTF} is the approximation factor of LTF in terms of task partitioning, due to the approximation factor of LPT for the makespan problem due from [6]. Furthermore, under this condition, in [17], it is also proven that

$$\frac{w_1^{\text{LTF}}}{w_M^{\text{LTF}}} \geq \frac{1}{2}. \tag{13}$$

From Corollary 1, for a set of tasks where the *total cycle utilization* is constant, the energy consumption under SFA can be reduced when the value of w_M is reduced. Therefore, considering Equation (12), we can conclude that LTF is a good task partitioning scheme for energy minimization under SFA.

Figure 4 and Figure 5 illustrate the properties of LTF.

Throughout the rest of this paper, we consider that $w_M^{\text{LTF}} \leq s_{\max}$; otherwise, LTF does not derive a feasible solution. Such infeasibility of the solution of LTF can be resolved by a resource augmentation scheme to augment speed $s_{\max}$ to $(\frac{4}{3} - \frac{1}{3M})s_{\max}$ for ensuring the feasibility if feasible solutions exist under the original $s_{\max}$ frequency constraint.

V. Double-Largest-Task-First Scheme (DLTF)

This section presents our task partitioning scheme for energy consumption reduction suited to apply in combination with SFA, called *Double-Largest-Task-First* (DLTF) scheme.

Algorithm 2 Double-Largest-Task-First (DLTF) scheme

Input: Tasks $\{\tau_1, \tau_2, \ldots, \tau_N\}$;
Output: Task sets $\{\mathbf{T}_1^{\text{DLTF}}, \mathbf{T}_2^{\text{DLTF}}, \ldots, \mathbf{T}_M^{\text{DLTF}}\}$;
Execute LTF for $\{\tau_1, \tau_2, \ldots, \tau_N\}$;
$w_{\max} \leftarrow \max\{s_{\text{crit}}, w_M^{\text{LTF}}\}$;
$\{\mathbf{T}_1^{\text{DLTF}}, \mathbf{T}_2^{\text{DLTF}}, \ldots, \mathbf{T}_M^{\text{DLTF}}\} \leftarrow \{\mathbf{T}_1^{\text{LTF}}, \mathbf{T}_2^{\text{LTF}}, \ldots, \mathbf{T}_M^{\text{LTF}}\}$;
for $i = 1$ **to** $M-1$ **do**
  for all $\tau_k \in \mathbf{T}_i^{\text{DLTF}}$ **do**
    for $j = M$ **to** $i+1$ **do**
      if $\frac{e_k}{p_k} + w_j^{\text{DLTF}} \leq w_{\max}$ **then**
        $\mathbf{T}_j^{\text{DLTF}} \leftarrow \mathbf{T}_j^{\text{DLTF}} + \{\tau_k\}$;
        break for loop j;
      end if
    end for
  end for
end for
Return $\{\mathbf{T}_1^{\text{DLTF}}, \mathbf{T}_2^{\text{DLTF}}, \ldots, \mathbf{T}_M^{\text{DLTF}}\}$;

A. Description of DLTF

Task partitioning scheme DLTF considers LTF as an initial solution, which means that after executing LTF, $\{\mathbf{T}_1^{\text{DLTF}} = \mathbf{T}_1^{\text{LTF}}, \mathbf{T}_2^{\text{DLTF}} = \mathbf{T}_2^{\text{LTF}}, \ldots, \mathbf{T}_M^{\text{DLTF}} = \mathbf{T}_M^{\text{LTF}}\}$, with the corresponding utilizations $0 = w_0^{\text{DLTF}} \leq w_1^{\text{DLTF}} = w_1^{\text{LTF}} \leq w_2^{\text{DLTF}} = w_2^{\text{LTF}} \leq \cdots \leq w_M^{\text{DLTF}} = w_M^{\text{LTF}}$, ordered without loss of generality. Then, to reduce the energy consumption for idling under SFA when considering non-negligible overhead for sleeping, we regroup the tasks, shutting-down as many cores as possible. For such a purpose, we define the auxiliary cycle utilization $w_{\max}$ as $\max\{s_{\text{crit}}, w_M^{\text{LTF}}\}$. The value of $w_{\max}$ is used as a maximum utilization for the task regrouping.

For the regrouping procedure, we consider a *source task set* and a *destination task set*, denoted as $\mathbf{T}_i^{\text{DLTF}}$ and $\mathbf{T}_j^{\text{DLTF}}$ respectively. The regrouping is an iterative procedure, where we iterate through all the *source task sets* (in an increasing manner with respect to their cycle utilizations), all the tasks inside each *source task set*, and all the possible *destination task sets* (in an decreasing manner with respect to their cycle utilizations) for each one of these tasks. Particularly, we start by choosing the *source task set* $\mathbf{T}_i^{\text{DLTF}}$, such that w_i^{DLTF} is the smallest cycle utilization among all task sets, that is, $i = 1$. We then iterate (with no specific order) through all tasks τ_k in $\mathbf{T}_i^{\text{DLTF}}$, with cycle utilization $\frac{e_k}{p_k}$. For every τ_k, we further iterate through the *destination task sets* $\mathbf{T}_j^{\text{DLTF}}$ for all $j = M, M-1, \ldots, i+2, i+1$, that is, we start from the task set with the highest cycle utilization. If τ_k fits in $\mathbf{T}_j^{\text{DLTF}}$ without exceeding $w_{\max}$, that is, if it holds that $\frac{e_k}{p_k} + w_j^{\text{DLTF}} \leq w_{\max}$, then we move τ_k to $\mathbf{T}_j^{\text{DLTF}}$ and update the value of w_j^{DLTF}. If τ_k does not fit inside $\mathbf{T}_j^{\text{DLTF}}$ without exceeding $w_{\max}$, then we update j to $j-1$ and try again with a new *destination task set*, until j reaches the value of i, point when we move to the next task inside $\mathbf{T}_i^{\text{DLTF}}$. The process is repeated for all tasks in the *source task set* $\mathbf{T}_i^{\text{DLTF}}$, and then i is updated to $i+1$, until i reaches M.

Finally, task sets with high cycle utilization after LTF will have even higher cycle utilization after regrouping, and vice-versa. Any core with utilization equal to 0, after partitioning with DLTF, can be further shut-down. A pseudo-code for DLTF is presented in Algorithm 2. *When we consider negligible overhead for sleeping, based on Corollary 1, the energy consumption of LTF under SFA is the same as the*

energy consumption of DLTF under SFA. However, when such overhead is non-negligible, DLTF saves some energy for idling compared to plain LTF.

B. Properties of DLTF

Given that the *total cycle utilization* of all the tasks is a constant for all possible task partitions, it holds that

$$\sum_{i=1}^{M} w_i^{\text{DLTF}} = \sum_{i=1}^{M} w_i^{\text{LTF}} = \sum_{i=1}^{M} w_i^*. \tag{14}$$

Furthermore, when $w_M^{\text{LTF}} \geq s_{\text{crit}}$, we have that $\mathbf{T}_M^{\text{DLTF}} = \mathbf{T}_M^{\text{LTF}}$ and therefore $w_M^{\text{DLTF}} = w_M^{\text{LTF}}$. Thus, when $w_i^{\text{LTF}} \geq s_{\text{crit}}$, the only property of LTF that does not hold in DLTF is Equation (13). In other words, for this case, if after the task partitioning there is *only one* task in $\mathbf{T}_M^{\text{DLTF}}$, then $w_M^{\text{DLTF}} \leq w_M^*$, as the cycle utilization of the task in $\mathbf{T}_M^{\text{DLTF}}$ is the lower bound of the maximum utilization in the optimal task partitioning; and, if there are *at least two* tasks in $\mathbf{T}_M^{\text{DLTF}}$, then it holds that $\frac{w_M^{\text{DLTF}}}{w_M^*} \leq \theta_{\text{LTF}} = \frac{4}{3} - \frac{1}{3M}$.

When $w_M^{\text{LTF}} < s_{\text{crit}}$, given that we might add some tasks into task set $\mathbf{T}_M^{\text{DLTF}}$ (in comparison with the original $\mathbf{T}_M^{\text{LTF}}$), then most likely we will have that $w_M^{\text{DLTF}} \geq w_M^{\text{LTF}}$. Nevertheless, if LTF results in $w_1^{\text{LTF}} \leq w_1^{\text{LTF}} \leq \cdots \leq w_M^{\text{LTF}} < s_{\text{crit}}$, then all cores will be executed at s_{crit} and the energy consumption for execution will be then minimized. Clearly, the optimal task partitioning and optimal DVFS schedule will consume the same energy for execution, resulting in an approximation factor of 1. Therefore, *for the analysis of the approximation factor we will only focus in the case that* $w_M^{LTF} \geq s_{crit}$.

VI. Approximation Factor Analysis when $\beta = 0$

This section presents the analysis of the approximation factor of DLTF combined with SFA, against the optimal task partitioning and optimal DVFS solution when $\beta = 0$, defined as $\text{AF}_{\text{SFA}}^{\text{DLTF}(\beta=0)}$. *Note that even though this is not a practical setting, we do incremental analysis for simplicity in presentation. The properties derived in this section will also be used in Section VII when* $\beta \neq 0$.

The expression for $\text{AF}_{\text{SFA}}^{\text{DLTF}(\beta=0)}$ is obtained by replacing Equation (4) and Equation (7) inside Equation (3). By also considering Equation (14), we have

$$\text{AF}_{\text{SFA}}^{\text{DLTF}(\beta=0)} \leq \max \left\{ \frac{\left(w_M^{\text{DLTF}}\right)^{\gamma-1} \sum_{i=1}^{M} w_i^*}{\left[\sum_{i=1}^{M} \left(w_i^* - w_{i-1}^*\right) \sqrt[\gamma]{M-i+1}\right]^{\gamma}} \right\}.$$

To obtain the worst-case of $\text{AF}_{\text{SFA}}^{\text{DLTF}(\beta=0)}$, we use the scaling factor r_i^* from Equation (8), such that $0 = r_0^* < r_1^* \leq r_2^* \leq \cdots \leq r_{M-1}^* \leq r_M^* = 1$ and $w_i^* = r_i^* \cdot w_M^*$ for all $i = 1, 2, \ldots, M$. Moreover, by considering the definition of $H(r_0, \ldots, r_M)$ and Lemma 1, $\text{AF}_{\text{SFA}}^{\text{DLTF}(\beta=0)}$ becomes

$$\text{AF}_{\text{SFA}}^{\text{DLTF}(\beta=0)} \leq \max \left\{ \left(\frac{w_M^{\text{DLTF}}}{w_M^*}\right)^{\gamma-1} \cdot h\left(\frac{\sum_{i=1}^{M-1} r_i^*}{M-1}\right) \right\}, \tag{15}$$

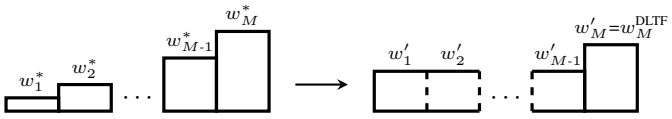

Fig. 6: Example of Cycle Utilization Adjustment in Lemma 3, when $w_M^* \geq w_M^{\text{DLTF}}$.

where $h(\delta)$ is defined in Lemma 1.

The following lemma presents a cycle utilization adjustment in the optimal task partitioning, needed for the rest of this section, that results in a reduced energy consumption lower bound. An example of such a cycle utilization adjustment, when $w_M^* \geq w_M^{\text{DLTF}}$, is shown in Figure 6.

Lemma 3: After applying DLTF for task partitioning, by adjusting $w_1^*, w_2^*, \ldots, w_M^*$ to $w_1', w_2', \ldots, w_M'$, then the energy consumption lower bound in Equation (4) by using $w_1^*, w_2^*, \ldots, w_M^*$ is further reduced by using $w_1', w_2', \ldots, w_M'$, in which $w_1' = w_2' = \cdots = w_{M-1}' = \frac{\sum_{i=1}^{M-1} w_i'}{M-1}$ and

$$w_M' = \begin{cases} w_M^{\text{DLTF}} & \text{if } w_M^* \geq w_M^{\text{DLTF}} \\ \max\left\{\frac{w_M^{\text{DLTF}}}{\theta_{\text{LTF}}}, \frac{\sum_{i=1}^{M} w_i^*}{M}\right\} & \text{if } w_M^* < w_M^{\text{DLTF}}. \end{cases}$$

Proof: This lemma can be first proved by reorganizing the cycle utilization $w_1^*, w_2^*, \ldots, w_M^*$ to $w_1', w_2', \ldots, w_M'$ such that $w_M' = w_M^*$, $w_1' = w_2' = \cdots = w_{M-1}' = \frac{\sum_{i=1}^{M-1} w_i^*}{M-1}$. The new cycle utilization $w_1', w_2', \ldots, w_M'$ has less energy consumption by using Equation (4). Then, further, we can easily reduce w_M' by a constant ϵ while increasing the others $w_1', w_2', \ldots, w_{M-1}'$ by a constant $\frac{\epsilon}{M-1}$. It can be easily observed that this also reduces the energy consumption under SFA.

Therefore, for the first case $w_M^* \geq w_M^{\text{DLTF}}$, this cycle utilization adjustment to $w_1', w_2', \ldots, w_M'$ clearly reduces the energy consumption lower bound in Equation (4). An example of such a cycle utilization adjustment is shown in Figure 6.

For the second case, in which $w_M^* < w_M^{\text{DLTF}}$, we know that $w_M^* \geq \frac{w_M^{\text{DLTF}}}{\theta_{\text{LTF}}}$ due to Equation (12). Therefore, we can greedily reduce the cycle utilization w_M', until (a) w_M' reaches $\frac{w_M^{\text{LTF}}}{\theta_{\text{LTF}}}$, point in which we cannot reduce further without violating $w_M^* \geq \frac{w_M^{\text{DLTF}}}{\theta_{\text{LTF}}}$; or either until (b) w_M' reaches the average cycle utilization, for which $w_1' = w_2' = \cdots = w_M'$ and we cannot reduce further without having that $w_M' < w_{M-1}'$. According to the above cycle utilization adjustment, both reduce the lower bound of the energy consumption in Equation (4) and ensure that $w_1' \leq w_2' \leq \cdots \leq w_M'$. Thus, the lemma is proven. ■

A. Approximation Factor when $\beta = 0$ and $w_M^* \geq w_M^{DLTF}$

This subsection analyses the approximation factor when $\beta = 0$ and $w_M^* \geq w_M^{\text{DLTF}}$. The following lemma provides a closed form for $\text{AF}_{\text{SFA}}^{\text{DLTF}(\beta=0)}$ for such a case.

Lemma 4: When applying DLTF for task partitioning, if $\beta = 0$ and $w_M^* \geq w_M^{\text{DLTF}}$, the approximation factor of DLTF combined with SFA in terms of energy consumption, is equal to the approximation factor of SFA without considering task partitioning with $\beta = 0$, presented in Theorem 1.

Proof: From the definition of $h(\delta)$ in Equation (10) and Lemma 1, it holds that Equation (15) is maximized according to Lemma 3, by adjusting $w_1^*, w_2^*, \ldots, w_M^*$ to $w'_1, w'_2, \ldots, w'_M$. Moreover, from Lemma 1, we know that $h(\delta)$ is maximized when $\delta = \delta^{\max}$, defined in Equation (10) and Equation (11). Thus, under this condition, the approximation factor is less or equal than $h(\delta^{\max})$, and the lemma is proven. ■

This is clearly the best case when $\beta = 0$, since the approximation factor of DLTF combined with SFA cannot be lower than the approximation factor of SFA without considering task partitioning.

B. Approximation Factor when $\beta = 0$ and $w_M^* < w_M^{DLTF}$

This subsection analyses the approximation factor for the case when $\beta = 0$ and $w_M^* < w_M^{\text{DLTF}}$.

Even though Lemma 1 states that $h(\delta)$ is maximized in $\delta^{\max}$, Lemma 6 shows that for this case, relation $\frac{w'_1}{w'_M}$ is constrained so that it never reaches the value of $\delta^{\max}$.

Lemma 5: When applying DLTF for task partitioning, if $w_M^* < w_M^{\text{DLTF}}$, it holds that

$$\frac{w'_1}{w'_M} \geq \frac{4M+1}{6M},$$

with $w'_1, w'_2, \ldots, w'_M$ defined in Lemma 3.

Proof: When $w_M^* < w_M^{\text{DLTF}}$, using $w'_1, w'_2, \ldots, w'_M$ from Lemma 3, there are two cases to consider (1) $w'_M = \frac{\sum_{i=1}^{M} w_i^*}{M}$ or (2) $w'_M = \frac{w_M^{\text{LTF}}}{\theta_{\text{LTF}}}$. For the first case, it is clear that $\frac{w'_1}{w'_M} = 1$. Hence, we focus on the second case, in which starting from Equation (14) and remembering that $w_M^{\text{DLTF}} = w_M^{\text{LTF}}$, we have

$$\begin{aligned} w_M^{\text{LTF}} + \sum_{i=1}^{M-1} w_i^{\text{LTF}} &= w'_M + \sum_{i=1}^{M-1} w'_i = \frac{w_M^{\text{LTF}}}{\theta_{\text{LTF}}} + (M-1)\, w'_1 \\ \Rightarrow w'_1 &= w_M^{\text{LTF}} \frac{\theta_{\text{LTF}} - 1}{\theta_{\text{LTF}}\,(M-1)} + \frac{\sum_{i=1}^{M-1} w_i^{\text{LTF}}}{(M-1)} \\ \Rightarrow \frac{w'_1}{w'_M} &= \frac{\theta_{\text{LTF}} - 1}{M-1} + \frac{\theta_{\text{LTF}}}{w_M^{\text{LTF}}} \frac{\sum_{i=1}^{M-1} w_i^{\text{LTF}}}{(M-1)}. \end{aligned}$$

Thus, considering Equation (13), it holds that

$$\begin{aligned} \frac{w'_1}{w'_M} &\geq \frac{\theta_{\text{LTF}} - 1}{M-1} + \frac{\theta_{\text{LTF}}}{2} = \frac{\frac{4}{3} - \frac{1}{3M} - 1}{M-1} + \frac{\frac{4}{3} - \frac{1}{3M}}{2} \\ &\geq \frac{4M+1}{6M}, \end{aligned}$$

and thus the lemma is proven. ■

Lemma 6: When applying DLTF for task partitioning, if $\beta = 0$ and $w_M^* < w_M^{\text{DLTF}}$, the approximation factor of DLTF combined with SFA in terms of energy consumption, is less or equal than $\theta_{\text{LTF}}^{\gamma-1} \cdot h\left(\frac{4M+1}{6M}\right)$.

Proof: This lemma is proven by considering Lemma 2, Lemma 3, Lemma 5 and Equation (15). ■

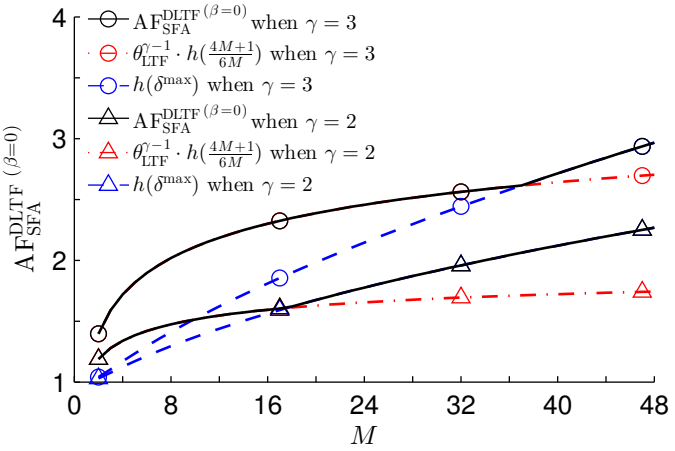

Fig. 7: Approximation factors for SFA combined with DLTF when $\beta = 0$.

C. Worst-case Approximation Factor when $\beta = 0$

Finally, considering Lemma 4 and Lemma 6, the approximation factor for DLTF combined with SFA in terms of energy consumption, when $\beta = 0$, is formalized in Theorem 3.

Theorem 3: When applying DLTF for task partitioning, if $\beta = 0$, the approximation factor of DLTF combined with SFA in terms of energy consumption is

$$\text{AF}_{\text{SFA}}^{\text{DLTF}(\beta=0)} \leq \max\left\{ h\left(\delta^{\max}\right), \theta_{\text{LTF}}^{\gamma-1} \cdot h\left(\frac{4M+1}{6M}\right) \right\}.$$

Proof: This comes from Lemma 4 and Lemma 6. ■

For a hardware platform, e.g., with $\gamma = 2$ or $\gamma = 3$, functions $h(\delta^{\max})$, $h\left(\frac{4M+1}{6M}\right)$ and θ_{LTF} are dependent on the value of M. Therefore, it is necessary to explore the impact of M on the approximation factor of DLTF combined with SFA, when $\beta = 0$. Theoretically, the approximation factor can go up to ∞ when $M \to \infty$. However, practically, the number of cores in a voltage island is not a very large number. Figure 7 presents the approximation factor, based on Theorem 3, for M up to 48 when $\gamma = 2$ and $\gamma = 3$. *Note that we do not expect next-generation platforms to have more than* 8 *or* 16 *cores per island, but it is important to show where the statements of Lemma 4 and Lemma 6 cross each other*. Whenever the voltage island has at most 4 (8, 16, 32, respectively) cores and negligible overhead for sleeping is considered, DLTF combined with SFA, when $\beta = 0$, has an approximation factor that can be bounded to at most 1.72 (2.02, 2.30, 2.56, respectively) when $\gamma = 3$ and to at most 1.34 (1.47, 1.59, 1.96, respectively) when $\gamma = 2$.

VII. Approximation Factor Analysis when $\beta \neq 0$

This section presents the analysis of the approximation factor of DLTF combined with SFA, against the optimal task partitioning and optimal DVFS solution, defined as $\text{AF}_{\text{SFA}}^{\text{DLTF}}$.

Clearly, Lemma 3 not only holds for Equation (4), but it also holds for Equation (5). This happens because when $w_M^* \leq s_{\text{dyn}}$, it does not matter whether we adjust the utilizations of the optimal task partitioning to fit Lemma 3 or not. Moreover, when $w_M^* > s_{\text{dyn}}$, Equation (5) is equal to Equation (4).

A. Approximation Factor as a function of s_{dyn}

According to Equation (3), $\text{AF}_{\text{SFA}}^{\text{DLTF}}$ is the ratio between Equation (6) and Equation (5), which expressed as a function of w_M^{DLTF} and w_M^* is

$$\frac{E_{\text{SFA}}^{\text{DLTF}}\left(w_M^{\text{DLTF}}\right)}{E_{\downarrow}^*\left(w_M^*\right)} \leq \begin{cases} \frac{\beta+\alpha {w_M^{\text{DLTF}}}^{\gamma}}{\alpha\gamma w_M^{\text{DLTF}}\left(s_{\text{crit}}{}^{\gamma-1}\right)} & \text{if } s_{\text{crit}} < w_M^{\text{DLTF}} \\ & \text{and } w_M^* \leq s_{\text{dyn}} \\ \frac{\frac{\beta+\alpha {w_M^{\text{DLTF}}}^{\gamma}}{\alpha {w_M^{\text{DLTF}}}} \cdot \sum_{i=1}^{M} w_i^*}{\left[\sum_{i=1}^{M}\left(w_i^* - w_{i-1}^*\right)\sqrt[\gamma]{M-i+1}\right]^{\gamma}} & \text{if } s_{\text{dyn}} < w_M^* \\ & \leq w_M^{\text{DLTF}} \leq s_{\max} \\ 1 & \text{otherwise.} \end{cases} \tag{16}$$

Lemma 7: By defining s_{dyn} as

$$s_{\text{dyn}} = s_{\text{crit}}\left[\gamma H\left(r_0^*, \ldots, r_M^*\right)\right]^{\frac{1}{\gamma-1}}, \tag{17}$$

we have

$$\frac{E_{\text{SFA}}^{\text{DLTF}}\left(w_M^{\text{DLTF}}\right)}{E_{\downarrow}^*\left(w_M^*\right)} \leq \frac{\beta + \alpha\left(\frac{w_M^{\text{DLTF}}}{w_M^*} s_{\text{dyn}}\right)^{\gamma}}{\alpha\gamma \frac{w_M^{\text{DLTF}}}{w_M^*} s_{\text{dyn}}\left(s_{\text{crit}}{}^{\gamma-1}\right)}. \tag{18}$$

Proof: Since α, β, γ and s_{crit} are all constants, we know that $\frac{E_{\text{SFA}}^{\text{DLTF}}\left(w_M^{\text{DLTF}}\right)}{E_{\downarrow}^*\left(w_M^*\right)}$ is a convex and increasing function with respect to w_M^{DLTF} when $s_{\text{crit}} < w_M^{\text{DLTF}}$ and $w_M^* \leq s_{\text{dyn}}$. Therefore, for this case, it holds that $\frac{E_{\text{SFA}}^{\text{DLTF}}\left(w_M^{\text{DLTF}}\right)}{E_{\downarrow}^*\left(w_M^*\right)} \leq \frac{E_{\text{SFA}}^{\text{DLTF}}\left(s_{\text{dyn}}\right)}{E_{\downarrow}^*\left(w_M^*\right)}$.

With the scaling factor r_i^* from Equation (8) and the definition of $H\left(r_0, \ldots, r_M\right)$ from Equation (9), we can rephrase Equation (16), when $s_{\text{dyn}} < w_M^* \leq w_M^{\text{DLTF}} < s_{\max}$, to

$$\frac{\frac{\beta}{{w_M^*}^{\gamma}} + \alpha\frac{{w_M^{\text{DLTF}}}^{\gamma}}{{w_M^*}^{\gamma}}}{\alpha\frac{w_M^{\text{DLTF}}}{w_M^*}} H\left(r_0^*, \ldots, r_M^*\right). \tag{19}$$

In the optimal task partitioning, function $H\left(r_0^*, \ldots, r_M^*\right)$ is a constant for a fixed set of tasks. Together with the fact that α, β, M and γ are constants, we know that (19) is a decreasing function with respect to w_M^*.

With the above analysis, function $\frac{E_{\text{SFA}}^{\text{DLTF}}\left(w_M^{\text{DLTF}}\right)}{E_{\downarrow}^*\left(w_M^*\right)}$ reaches the (lowest) upper bound when the above two cases intersect with each other. That is, when

$$\frac{\beta + \alpha {w_M^{\text{DLTF}}}^{\gamma}}{\gamma w_M^{\text{DLTF}}\left(s_{\text{crit}}{}^{\gamma-1}\right)} = \frac{\frac{\beta}{{w_M^*}^{\gamma}} + \alpha\frac{{w_M^{\text{DLTF}}}^{\gamma}}{{w_M^*}^{\gamma}}}{\frac{w_M^{\text{DLTF}}}{w_M^*}} H\left(r_0^*, \ldots, r_M^*\right),$$

which results in the definition of s_{dyn} in Equation (17), when $w_M^* = s_{\text{dyn}}$. Thus, the inequality in Equation (18) holds when $w_M^* = s_{\text{dyn}}$, and this lemma is proven. ■

B. Approximation Factor when $\beta \neq 0$ and $w_M^* \geq w_M^{DLTF}$

This subsection analyses the approximation factor when $\beta \neq 0$ and $w_M^* \geq w_M^{\text{DLTF}}$. Lemma 8 provides a closed form for $\text{AF}_{\text{SFA}}^{\text{DLTF}}$ for such case.

Lemma 8: When applying DLTF for task partitioning, if $\beta \neq 0$ and $w_M^* \geq w_M^{\text{DLTF}}$, the approximation factor of DLTF combined with SFA in terms of energy consumption, is equal to the approximation factor of SFA without considering task partitioning with $\beta \neq 0$, presented in Theorem 2.

Proof: We define $s_{\text{dyn}}^{\max}$ as the maximum value for s_{dyn}, which from Equation (9), Lemma 1 and Lemma 3, it happens when $s_{\text{dyn}}^{\max} = s_{\text{crit}}\left[\gamma h\left(\delta^{\max}\right)\right]^{\frac{1}{\gamma-1}}$. Moreover, when $w_M^* \geq w_M^{\text{DLTF}}$, the maximum value for Equation (18) is obtained when s_{dyn} is equal to $s_{\text{dyn}}^{\max}$ and when $w'_M = w_M^{\text{DLTF}}$, according to Lemma 3. Thus, by replacing in Equation (18), we obtain the expression from Theorem 2, and the lemma is proven. ■

Similar to Lemma 4, this is clearly the best case when $\beta \neq 0$, since the approximation factor of DLTF combined with SFA cannot be lower than the approximation factor of SFA without considering task partitioning.

C. Approximation Factor when $\beta \neq 0$ and $w_M^* < w_M^{DLTF}$

The following lemma provides a closed form for this case.

Lemma 9: When applying DLTF for task partitioning, if $\beta \neq 0$ and $w_M^* < w_M^{\text{DLTF}}$, the approximation factor of DLTF combined with SFA in terms of energy consumption, is less or equal than $\frac{\gamma-1}{\theta_{\text{LTF}}\left[\gamma^{\gamma} h\left(\frac{4M+1}{6M}\right)\right]^{\frac{1}{\gamma-1}}} + \theta_{\text{LTF}}^{\gamma-1} h\left(\frac{4M+1}{6M}\right)$.

Proof: From Lemma 1 we know that $H\left(r_0^*, \ldots, r_M^*\right)$ is less or equal than $h\left(\delta\right)$, defined in Equation (10). Moreover, from Lemma 3 and Lemma 5, when $w_M^* < w_M^{\text{DLTF}}$, it holds that $H\left(r_0^*, \ldots, r_M^*\right) \leq h\left(\frac{4M+1}{6M}\right)$. Thus, when $w_M^* < w_M^{\text{DLTF}}$, it holds that $s_{\text{dyn}} \leq s_{\text{dyn}}^* = s_{\text{crit}}\left[\gamma h\left(\frac{4M+1}{6M}\right)\right]^{\frac{1}{\gamma-1}}$. Moreover, for this case, the maximum value for Equation (18) is obtained when s_{dyn} is equal to s_{dyn}^* and when $w'_M = \frac{w_M^{\text{DLTF}}}{\theta_{\text{LTF}}}$, according to Lemma 3 and Lemma 5. Therefore, by replacing these values in Equation (18), the lemma is proven. ■

D. Worst-case Approximation Factor when $\beta \neq 0$

Finally, considering Lemma 8 and Lemma 9, the approximation factor for DLTF combined with SFA in terms of energy consumption, when $\beta \neq 0$, is formalized in Theorem 4.

Theorem 4: When applying DLTF for task partitioning, if $\beta \neq 0$, the approximation factor of DLTF combined with SFA in terms of energy consumption is

$$\text{AF}_{\text{SFA}}^{\text{DLTF}} \leq \max\left\{\frac{\gamma-1}{\left[\gamma^{\gamma} h\left(\delta^{\max}\right)\right]^{\frac{1}{\gamma-1}}} + h\left(\delta^{\max}\right), \right.$$
$$\left. \frac{\gamma-1}{\theta_{\text{LTF}}\left[\gamma^{\gamma} h\left(\frac{4M+1}{6M}\right)\right]^{\frac{1}{\gamma-1}}} + \theta_{\text{LTF}}^{\gamma-1} h\left(\frac{4M+1}{6M}\right)\right\}.$$

Proof: This comes from Lemma 8 and Lemma 9. ■

Similar to Section VI-C, for a fixed hardware platform, $\text{AF}_{\text{SFA}}^{\text{DLTF}}$ is a function of M. Figure 8 presents the approximation factor, based on Theorem 4, for M up to 48 when $\gamma = 2$ and $\gamma = 3$. Whenever the voltage island has at most 4 (8, 16, 32, respectively) cores and negligible overhead for sleeping is considered, DLTF combined with SFA has an approximation factor that can be bounded to at most 2.01 (2.29, 2.55, 2.80, respectively) when $\gamma = 3$ and to at most 1.53 (1.64, 1.75, 2.09, respectively) when $\gamma = 2$. These values are just slightly higher than those of Section VI-C.

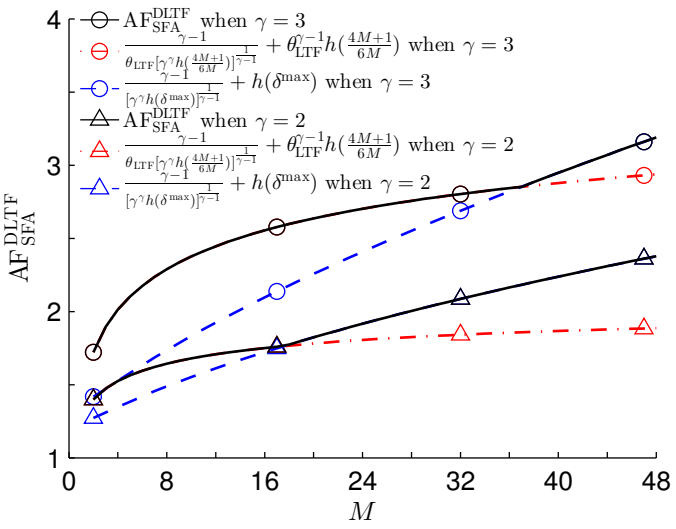

Fig. 8: Approximation factors for SFA combined with DLTF when $\beta \neq 0$.

VIII. Numerical Evaluations

This section provides numerical results for specific case studies, without any approximation in the lower bound for the optimal energy consumption, instead of using Equation (5).

A. Numerical Evaluation Setup

Similar to Section II-A, the parameters of $P(s)$ are chosen as $\alpha = 1.76 \frac{\text{Watts}}{\text{GHz}^3}$, $\beta = 0.5$ Watts, $\gamma = 3$, and $s_{\text{crit}} = 0.52$ GHz, modelled from the experimental measurements from [8].

We consider 150 cases of synthetic tasks for every M. Every case considers a different amount of tasks, different cycle utilizations, the existence or non-existence of *one considerably large* task, and different resulting hyper-periods. When using SFA, the tasks are partitioned using DLTF. As for the optimal task partitioning, given that obtaining it is an $\mathcal{NP}$-hard problem, we consider two lower bounds as proved in Section VI and Section VII: (1) when there is *only one* task in task set $\mathbf{T}_M^{\mathrm{DLTF}}$ we consider a low bound with $w_1^* = w_2^* = \cdots = w_{M-1}^* = \delta^{\max} \cdot w_M^*$ and $w_M^* = w_M^{\mathrm{DLTF}}$ (from Lemma 1, Lemma 3 and Lemma 8), and (2) when there are *at least two* tasks in task set $\mathbf{T}_M^{\mathrm{DLTF}}$ we consider a lower bound with $w_1^* = w_2^* = \cdots = w_{M-1}^* = w_M^* \cdot \frac{4M+1}{6M}$ and $w_M^* = \frac{w_M^{\mathrm{DLTF}}}{\theta_{\mathrm{LTF}}}$ (from Lemma 3, Lemma 5 and Lemma 9).

For the analysis in Section VII, we used the lower bound in Equation (5) for the optimal energy consumption and the optimal task partitioning, derived in [14]. The reason for using this approximated lower bound is due to the difficulty of obtaining the actual optimal values. A more precise expression, specifically Equation (9) in [14], can be numerically solved by using Newton's method for a specific input instance. Such an approach can help obtain less pessimistic and concrete energy consumption values for the optimal DVFS schedule.

B. Numerical Evaluation Results

The maximum concrete factor among the 150 evaluated cases of synthetic tasks for every M is reported as the peak factor for approximation, denoted by $\left.\frac{E_{\mathrm{SFA}}^{\mathrm{DLTF}}(w_M^{\mathrm{DLTF}})}{E_{\downarrow}^*(w_M^*)}\right|_{\text{peak}}$. Figure 9 presents the results of this peak factor, together with the analytical upper bound derived in Theorem 4, for $\gamma = 3$.

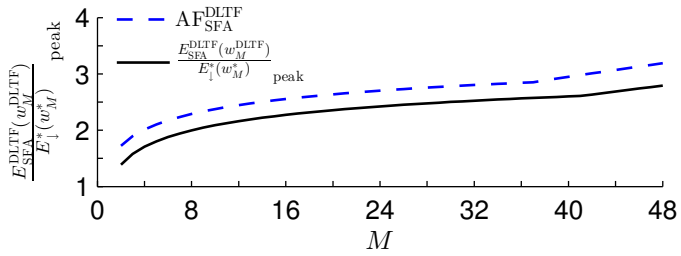

Fig. 9: Numerical results of DLTF combined with SFA for $\alpha = 1.76 \frac{\text{Watts}}{\text{GHz}^3}$, $\beta = 0.5$ Watts and $\gamma = 3$.

We observe that the peak factor provides a lower analytical bound for approximation for these concrete cases. The difference between this peak factor and the theoretical $\mathrm{AF}_{\mathrm{SFA}}^{\mathrm{DLTF}}$ from Theorem 4 comes from the precise computation of Equation (9) in [14], by using Newton's method for specific case studies, instead of using the approximated lower bound from Equation (5).

IX. Non-negligible Sleeping Overhead

For systems with non-negligible *overhead for sleeping*, our scheme can be further combined with DPM schemes to decide when to switch a core into a low-power mode. There are two cases to be considered: (1) the total cycle utilization is less than s_{crit}, i.e., $\sum_{j=1}^{N} \frac{e_j}{p_j} < s_{crit}$, and (2) the total cycle utilization is no less than s_{crit}, i.e., $\sum_{j=1}^{N} \frac{e_j}{p_j} \geq s_{crit}$. For the first case, DLTF assigns all the tasks on one core and executes them at the critical speed, making this a uniprocessor scheduling problem. For example, we can then use the *Left-To-Right* (LTR) algorithm in [10] (only for the DPM) to achieve a 2-approximation for such a case.

For the second case, i.e., $\sum_{j=1}^{N} \frac{e_j}{p_j} \geq s_{crit}$, our scheme can be further combined with *any uni-core procrastination algorithm* that does not consume more energy than putting a core always in idle mode when it has no workload to execute in its ready queue. The rest of this section analyses the approximation factor of such an approach, against the optimal task partitioning and optimal DVFS and DPM schedule.

For simplicity in presentation, we assume that under the scheme that uses DLTF combined with SFA, every core stays idle (in execution mode) when it has no workload to execute in its ready queue, consuming β power. This provides a safe upper bound for the analysis, since by considering the *break-even time* entering a low-power mode optimally, will not consume more energy. The energy consumption for idling in a hyper-period when using DLTF combined with SFA and *any uni-core procrastination algorithm*, is denoted as $\text{idle}\left(E_{\text{SFA-DPM}}^{\mathrm{DLTF}}\right)$. Furthermore, suppose that $M^{\dagger}$ cores are activated for execution in DLTF. The energy consumption for execution in the optimal task partitioning and optimal DVFS and DPM schedule, is denoted as $\text{active}\left(E_{\text{DVFS-DPM}}^{*}\right)$. Moreover, we have the following lemmas for the upper bound of $\text{idle}\left(E_{\text{SFA-DPM}}^{\mathrm{DLTF}}\right)$ and the lower bound of $\text{active}\left(E_{\text{DVFS-DPM}}^{*}\right)$.

Lemma 10: When $\sum_{j=1}^{N} \frac{e_j}{p_j} \geq s_{crit}$, it holds that

$$\text{idle}\left(E^{\text{DLTF}}_{\text{SFA-DPM}}\right) < 0.5M^{\dagger}L\beta.$$

Proof: In DLTF with $\sum_{j=1}^{N} \frac{e_j}{p_j} \geq s_{crit}$, we know that $w_i^{\text{DLTF}} + w_\ell^{\text{DLTF}} > \max\{w_i^{\text{LTF}}, s_{crit}\}$ for any two cores i and ℓ with $w_i^{\text{DLTF}} > 0$ and $w_\ell^{\text{DLTF}} > 0$; otherwise, the tasks on either core i or ℓ will be moved to other cores. Therefore, by executing at speed $\max\{w_i^{\text{LTF}}, s_{crit}\}$ for all these $M^{\dagger}$ cores, we know that the total idle time in DLTF with SFA for the activated cores in the hyper-period is

$$L \sum_{i=M-M^{\dagger}+1}^{M} \left(1 - \frac{w_i^{\text{DLTF}}}{\max\{w_M^{\text{DLTF}}, s_{crit}\}}\right) < \frac{LM^{\dagger}}{2}.$$

Therefore, the lemma is proved. ■

Lemma 11: When $\sum_{j=1}^{N} \frac{e_j}{p_j} \geq s_{crit}$, it holds that

$$\text{active}\left(E^{*}_{\text{DVFS-DPM}}\right) \geq 0.5M^{\dagger}L\frac{\gamma}{\gamma-1}\beta.$$

Proof: There are two cases: (1) $w_M^{\text{DLTF}} < s_{crit}$, or (2) $w_M^{\text{DLTF}} \geq s_{crit}$. For the first case, similar to the analysis shown in Lemma 10, we know that $\sum_{j=1}^{N} \frac{e_j}{p_j} > \frac{M^{\dagger}}{2} s_{crit}$. Therefore, by the definition of the speed s_{crit}, we know that $\text{active}\left(E^{*}_{\text{DVFS-DPM}}\right)$ is at least the energy consumption for executing all the tasks at s_{crit}. By the above argument, we have $\text{active}\left(E^{*}_{\text{DVFS-DPM}}\right) \geq 0.5M^{\dagger}L \cdot P(s_{crit}) = 0.5M^{\dagger}L\frac{\gamma}{\gamma-1}\beta$ for the first case.

For the second case, similarly, we have $\sum_{j=1}^{N} \frac{e_j}{p_j} > \frac{M^{\dagger}}{2} w_M^{\text{DLTF}} \geq \frac{M^{\dagger}}{2} s_{crit}$. Therefore, the statement for $\text{active}\left(E^{*}_{\text{DVFS-DPM}}\right) \geq 0.5M^{\dagger}L \cdot P(s_{crit}) \geq 0.5M^{\dagger}L\frac{\gamma}{\gamma-1}\beta$ also holds for this case. ■

Finally, the following theorem provides the approximation factor when considering non-negligible *overhead for sleeping*, defined as $\text{AF}^{\text{DLTF}}_{\text{SFA-DPM}}$.

Theorem 5: By using DLTF combined with SFA and *any uni-core procrastination algorithm* that does not consume more energy than putting a core always in idle mode when it has no workload to execute in its ready queue, when $\sum_{j=1}^{N} \frac{e_j}{p_j} \geq s_{crit}$, it holds that

$$\text{AF}^{\text{DLTF}}_{\text{SFA-DPM}} \leq \text{AF}^{\text{DLTF}}_{\text{SFA}} + \frac{\gamma-1}{\gamma}.$$

Proof: Considering Theorem 4, Lemma 10, Lemma 11, and that by definition $\text{AF}^{\text{DLTF}}_{\text{SFA}} \geq 1$, we have

$$\begin{aligned}
\text{total}\left(E^{\text{DLTF}}_{\text{SFA-DPM}}\right) &= \text{active}\left(E^{\text{DLTF}}_{\text{SFA-DPM}}\right) + \text{idle}\left(E^{\text{DLTF}}_{\text{SFA-DPM}}\right) \\
&\leq \text{AF}^{\text{DLTF}}_{\text{SFA}} \cdot \text{active}\left(E^{*}_{\text{DVFS-DPM}}\right) + \frac{\gamma-1}{\gamma}\text{active}\left(E^{*}_{\text{DVFS-DPM}}\right) \\
&\leq \left(\text{AF}^{\text{DLTF}}_{\text{SFA}} + \frac{\gamma-1}{\gamma}\right) \cdot \text{active}\left(E^{*}_{\text{DVFS-DPM}}\right),
\end{aligned}$$

where $\text{total}\left(E^{\text{DLTF}}_{\text{SFA-DPM}}\right)$ is the total energy consumption for using DLTF combined with SFA and *any uni-core procrastination algorithm* that does not consume more energy than putting a core always in idle mode when it has no workload to execute in its ready queue. Thus, the theorem is proven. ■

X. Conclusions

In this paper we have presented a practical solution to partition and schedule periodic real-time task in a voltage island, that combines DLTF and SFA for energy minimization. Most importantly, we provide thorough analysis of the approximation factor of such solution, in terms of energy efficiency, against the optimal task partitioning and optimal DVFS schedule. The analysis shows that the approximation factor can be bounded to a value depending on γ and the number of cores in a voltage island. For systems with non-negligible overhead for sleeping, further combination with *any uni-core procrastination algorithm* (that does not consume more energy than putting a core always in idle mode when it has no workload to execute in its ready queue) increases the approximation factor by $\frac{\gamma-1}{\gamma}$.

Acknowledgements: This work is supported in part by Baden Württemberg MWK Juniorprofessoren-Programms.

References

[1] S. Albers and A. Antoniadis, "Race to idle: new algorithms for speed scaling with a sleepstate," in *Symposium on Discrete Algorithms (SODA)*, 2012, pp. 1266–1285.

[2] H. Aydin and Q. Yang, "Energy-aware partitioning for multiprocessor real-time systems," in *IPDPS*, 2003, pp. 113 – 121.

[3] J.-J. Chen, H.-R. Hsu, and T.-W. Kuo, "Leakage-aware energy-efficient scheduling of real-time tasks in multiprocessor systems," in *Proceedings of the 12th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 2006, pp. 408–417.

[4] J.-J. Chen and L. Thiele, "Energy-efficient scheduling on homogeneous multiprocessor platforms," in *SAC*, 2010, pp. 542–549.

[5] V. Devadas and H. Aydin, "Coordinated power management of periodic real-time tasks on chip multiprocessors," in *Proceedings of the International Conference on Green Computing*, 2010, pp. 61 –72.

[6] R. Graham, "Bounds on multiprocessing timing anomalies," *SIAM Journal on Applied Mathematics*, vol. 17, pp. 263–269, 1969.

[7] S. Herbert and D. Marculescu, "Analysis of dynamic voltage/frequency scaling in chip-multiprocessors," in *ISLPED*, 2007, pp. 38–43.

[8] J. Howard and others, "A 48-core ia-32 processor in 45 nm cmos using on-die message-passing and dvfs for performance and power scaling," *IEEE Journal of Solid-State Circuits*, vol. 46, no. 1, pp. 173–183, 2011.

[9] Intel Corporation, "Single-chip cloud computer (SCC)."

[10] S. Irani, S. Shukla, and R. Gupta, "Algorithms for power savings," in *Symposium on Discrete Algorithms (SODA)*, 2003, pp. 37–46.

[11] R. Jejurikar, C. Pereira, and R. Gupta, "Leakage aware dynamic voltage scaling for real-time embedded systems," in *Proceedings of the 41st Design Automation Conference (DAC)*, 2004, pp. 275–280.

[12] C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment," *Journal of the ACM (JACM)*, vol. 20, no. 1, pp. 46–61, 1973.

[13] G. Moreno and D. de Niz, "An optimal real-time voltage and frequency scaling for uniform multiprocessors," in *Proceedings of the 18th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, 2012, pp. 21–30.

[14] S. Pagani and J.-J. Chen, "Energy efficiency analysis for the single frequency approximation (SFA) scheme," in *Proceedings of the 19th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, 2013.

[15] E. Seo, J. Jeong, S.-Y. Park, and J. Lee, "Energy efficient scheduling of real-time tasks on multicore processors," *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 11, pp. 1540–1552, 2008.

[16] R. Xu, D. Zhu, C. Rusu, R. Melhem, and D. Mossé, "Energy-efficient policies for embedded clusters," in *LCTES*, 2005, pp. 1–10.

[17] C.-Y. Yang, J.-J. Chen, and T.-W. Kuo, "An approximation algorithm for energy-efficient scheduling on a chip multiprocessor," in *Conference on Design, Automation, and Test in Europe (DATE)*, 2005, pp. 468–473.