CISTER
Research Centre in
Real-Time & Embedded
Computing Systems

Conference Paper

B-TSP: An Advanced Power Safe Management Strategy for modern Multi-core Platforms under Thermal-Aware Design

Javier Pérez Rodriguez
Yilian Ribot
Patrick Meumeu Yomsi
Luis Puente Lam

CISTER-TR-230612

B-TSP: An Advanced Power Safe Management Strategy for modern Multi-core Platforms under Thermal-Aware Design

Javier Pérez Rodriguez, Yilian Ribot, Patrick Meumeu Yomsi, Luis Puente Lam

CISTER Research Centre

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8321159

E-mail:

https://www.cister-labs.pt

Abstract

Thermal management is a crucial aspect of the design and operation of safety-critical multi-core architectures, as their high power density can cause significant heat generation and risk of thermal overload. If not properly managed, thermal overload can lead to system failures and performance degradation, which is a major challenge for system designers. To address this challenge, advanced core mapping solutions have become increasingly popular in both industry and academia. In this paper, we present key insights, techniques and results on thermal management in multi-core architectures. We propose a new per-core power budget strategy called that is scalable and enables system performance optimization while abstracting from mapping concerns. In addition, we present a new strategy called that allows us to derive worst-case mappings as a function of the number of active cores from a power consumption perspective in a thermal-aware design. We demonstrate the effectiveness of our solution through intensive simulations with the homogeneous AMD EPYC 7351 16-cores platform.

B-TSP: An Advanced Power Safe Management Strategy for modern Multi-core Platforms under Thermal-Aware Design

Javier Pérez Rodríguez
CISTER, ISEP, Polytechnic Institute of Porto
Porto, Portugal
perez@isep.ipp.pt

Patrick Meumeu Yomsi
CISTER, ISEP, Polytechnic Institute of Porto
Porto, Portugal
pmy@isep.ipp.pt

Yilian Ribot González
CISTER, ISEP, Polytechnic Institute of Porto
Porto, Portugal
ribot@isep.ipp.pt

Luis Javier Puente Lam
CISTER, ISEP, Polytechnic Institute of Porto
Porto, Portugal
javie@isep.ipp.pt

ABSTRACT

Thermal management is a crucial aspect of the design and operation of safety-critical multi-core architectures, as their high power density can cause significant heat generation and risk of thermal overload. If not properly managed, thermal overload can lead to system failures and performance degradation, which is a major challenge for system designers. To address this challenge, advanced core mapping solutions have become increasingly popular in both industry and academia. In this paper, we present key insights, techniques and results on thermal management in multi-core architectures. We propose a new per-core power budget strategy called B-TSP that is scalable and enables system performance optimization while abstracting from mapping concerns. In addition, we present a new strategy called FBF that allows us to derive worst-case mappings as a function of the number of active cores from a power consumption perspective in a thermal-aware design. We demonstrate the effectiveness of our solution through intensive simulations with the homogeneous AMD EPYC 7351 16-cores platform.

CCS CONCEPTS

• **Hardware → Thermal issues**; • **Computer systems organization → Embedded systems**; *Real-time systems.*

KEYWORDS

Multi-core architectures, thermal-aware mapping and scheduling, DVFS, power budgeting.

ACM Reference Format:
Javier Pérez Rodríguez, Patrick Meumeu Yomsi, Yilian Ribot González, and Luis Javier Puente Lam. 2023. B-TSP: An Advanced Power Safe Management Strategy for modern Multi-core Platforms under Thermal-Aware Design. In *The 31st International Conference on Real-Time Networks and Systems (RTNS 2023), June 7–8, 2023, Dortmund, Germany.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3575757.3593659

RTNS 2023, June 7–8, 2023, Dortmund, Germany

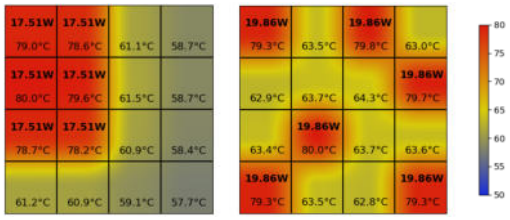
ACM ISBN 978-1-4503-9983-8/23/06.
https://doi.org/10.1145/3575757.3593659

1 INTRODUCTION

The ever-increasing trend towards downsizing safety-critical systems geared by modern multi-core architectures has led to higher power density than ever before. This poses a significant risk, as any failure or malfunction can lead to catastrophic consequences such as environmental damage, material loss, serious injury, or even death. Consequently, in application domains such as avionics and aerospace, thermal issues have become a major challenge in the development of such systems. This is primarily due to thermal coupling and heat transfer between neighboring computing elements. Elevated temperatures can lead to thermal hotspots, which in turn can result in low reliability, shortened lifetime, or even irreversible damage to the underlying platform. In this context, it is of paramount importance to develop robust and flexible *power safe management strategies* that help guiding the core mapping, Dynamic Power Management (DPM), and Dynamic Voltage/Frequency Scaling (DVFS) schemes such that any increase in the assumed per-core power budget results in exceeding the maximum allowable temperature. In other words, we seek to achieve highly predictable system performance under thermal-aware design for modern multi-core architectures.

Generally, chip manufacturers specify what is called *Thermal Design Power* (TDP) [3], i.e., the maximum power for which the system should be designed for. This value provides system designers/integrators with a predefined total power target to aid in the development of accurate thermal-aware solutions. However, TDP uses a *single and constant value* for each core and/or for the entire chip. In recent years, Pagani et al. [12, 13] have pointed out the potentially large performance losses that such an approach entails. To circumvent this hurdle, they proposed a power budget concept called *Thermal Safe Power* (TSP) for a given core mapping, which is an abstraction that provides safe and constant per-core power constraint values as a function of the number of simultaneously active cores. Here, each per-core power budget value guarantees that *at least one* of the active cores reaches the maximum allowable temperature at steady-state (see Figure 1).

Using Hotspot [19], a hypothetical multi-core system with 16 homogeneous cores arranged in 4 rows and 4 columns, and assuming that (*i*) the maximum thermal constraint of the chip is $T_{\max} = 80^{\circ}C$; (*ii*) TDP = 170 W, and (*iii*) the power of each inactive core is zero; Figure 1 shows the resulting *steady-state temperatures* on the chip using TSP for two different core mappings when 6 cores are active



(a) Concentrated cores (b) Distributed cores

Figure 1: Two distinct mappings for 6 active cores under TSP

with consumption of 17.51 W and 19.86 W, respectively. The total power of the chip is then 105.06 W and 119.16 W. In this figure, the top numbers represent the power consumption of each active core, while the bottom numbers indicate the temperatures at the center of each core with respect to the color bar. In Figure 1a, the 6 active cores are *concentrated* in the upper-left corner, while in Figure 1b, the 6 active cores are *distributed* across the chip. In both configurations, $T_{\max}$ is never exceeded, but only one core reaches the actual maximum allowed temperature, while the temperature of the other active cores falls *strictly* below this threshold. In the concentrated configuration, the minimum temperature of all active cores is 78.2°C, while in the distributed configuration it is 79.3°C. Figure 2 shows the thermal gap between the minimum temperature among the active cores and $T_{\max}$ under TSP in configurations where each active core is stressed as much as possible from the thermal point of view (e.g., the gap is 1.8°C when 6 cores are active, as shown in Figure 1a).

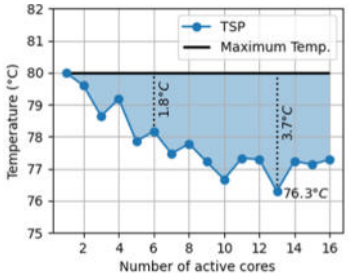


Figure 2: Thermal gap under TSP

In this figure, it is noteworthy that this gap reaches up to 3.7°C for only 13 active cores. Moreover, the trend of TSP performance worsens as the number of active cores increases. Today, it is not uncommon to find platforms with a high number of homogeneous cores on the market, e.g., Xeon W-3275M [8] and Xeon Platinum 9282 [7] from Intel and EPYC 7002 Series [2] from AMD, which handle 28, 56 and 64 cores, respectively.

▹ **Motivation.** Although TSP has led to a significant advance in system performance over the state-of-the-art at the time of its development, it has a number of important limitations that leave room for further improvement.

(1) **About the underutilization of some cores.** TSP assigns constant power to each active core, which may cause some cores to be underutilized. If thermal headroom exists on a given active core for a given core mapping, it is possible to design an efficient alternative approach that *uniformly* redistributes the per-core power budget so that not only some, but *all active cores* reach $T_{\max}$.
(2) **About the ability to handle dynamic workloads.** TSP may not allow system developers to handle a dynamic workload. It is agnostic to what is actually running on each active core. If the power profile of a particular application is known in advance, an active core might consume less power during run-time than other active cores on the same chip. If this is not carefully considered, it can lead to a poor performance-to-power ratio.
(3) **About the scalability of the approach.** TSP may exhibit limited scalability on modern multi-core architectures. When the power assigned to each active core is constant, increasing the number of active cores negatively affects performance (see the trend in Figure 2), making it difficult to use the underlying platform at its maximum computational capabilities.

▹ **This research.** We build on the work of Pagani et al. [13] and propose a new methodology to address the abovementioned problems. Briefly, we propose a novel power budget strategy, called *Beyond - Thermal Safe Power* (B-TSP), along with the associated theoretical evidence that B-TSP: (1) provides safe unbalanced power budgets as a function of the number of simultaneously active cores, such that the maximum thermal constraint $T_{\max}$ is achieved on *all active cores* at steady-state; (2) allows the power budget to be recalculated for each active core to maximize the performance-to-power ratio when the application's power profile is known in advance; and finally (3) is scalable as the number of active cores increases.

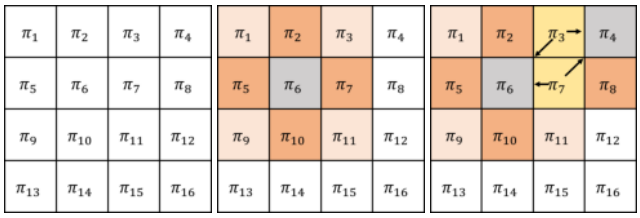
▹ **Paper organization.** We present the adopted model of execution together with the main notations in Section 2. Section 3 specifies our B-TSP solution in detail. In particular, B-TSP for a given core mapping is presented in Subsection 3.1; then, a new strategy for fitting the power budgets of each core when the total power exceeds TDP is proposed in Subsection 3.2; finally, a thorough discussion of worst-case mappings under thermal-aware design for system designers and/or resource management techniques to abstract from core mapping decisions is provided in Subsection 3.3. Section 4 describes how the thermal behavior of the system in the transient phase is integrated into our framework. Section 5 reports on the experiments performed, and Section 6 reviews existing related work on this topic, which, while not exhaustive, is representative. Finally, Section 7 draws the conclusions of the paper and discusses future research directions.

2 SYSTEM MODEL

In this section, we define the hardware and thermal models adopted in this work. We also introduce most of the concepts, notations, and parameters to allow a good understanding of the proposed approach. All matrices and vectors are shown in bold.

▹ **Hardware model.** In this work, we consider a homogeneous multi-core platform $\boldsymbol{\pi}$ with m cores and $M \geq m$ blocks in the floorplan, where $M - m$ is the number of blocks corresponding to other types of components, e.g., L2 caches and memory controllers. Without loss of generality and for simplicity of exposition, we

assume that there are as many blocks in π as there are cores, and that they are arranged in 4 rows and 4 columns, as shown in Figure 3a, i.e., $\boldsymbol{\pi} \stackrel{\text{def}}{=} [\pi_1, \pi_2, \ldots, \pi_{16}]$.



(a) Floorplan (b) Vicinity of a core (c) Direct vs. Indirect

Figure 3: Architecture of the homogeneous 16-cores platform.

We assume that each core $\pi_p \in \pi$ (with $p \in [1, 16]$) is DVFS-enabled and has an *inactive mode* (i.e., when the core is switched-off or power-gated) and an *active mode* (i.e., when the core is in its execution mode). We assume that the power of π_p is $P^{\text{inact}}_{\pi_p} \geq 0$ when it is inactive, and $P^{\text{act}}_{\pi_p} > P^{\text{inact}}_{\pi_p}$, when it is active. We refer to the decision about which cores to activate as *core mapping*. In this process, we define $\mathbf{X}$ as a binary vector in which: $x_p = 1$ means that π_p is active; and $x_p = 0$ means that π_p is inactive. If $k \leq m$ cores are active at the same time, there are $C^m_k = \frac{m!}{k! \cdot (m-k)!}$ possible combinations. We assume that the total power consumption P^{total}_{π} for any mapping must not exceed the maximum power supply of the platform $P_{\max} \stackrel{\text{def}}{=}$ TDP, i.e., we assume that Equation 1 holds true.

$$P^{\text{total}}_{\pi} \stackrel{\text{def}}{=} \sum_{p=1}^{m} \left\{ x_p \cdot P^{\text{act}}_{\pi_p} + (1 - x_p) \cdot P^{\text{inact}}_{\pi_p} \right\} \leq \text{TDP} \tag{1}$$

We recall that TDP is the maximum power for which the system should be designed for. This constraint is relaxed in Section 3.2. It should also be noted that the power consumption of each core is closely related to its execution frequency [14]: The higher the power consumption, the faster the execution and, consequently, the larger the potentially executable workload. For a smooth understanding of the rest of this paper, we introduce below some key definitions for the homogeneous multi-core architecture shown in Figure 3a.

Definition 1 (Vicinity of a core). *For platform π, we define the vicinity of core $\pi_p \in \pi$ as the set $\mathcal{V}(\pi_p)$ consisting of all cores immediately surrounding π_p.*

Definition 2 (Direct contacts of a core). *We define the direct contacts of core π_p as the subset $\mathcal{D}(\pi_p) \subseteq \mathcal{V}(\pi_p)$ of cores sharing an entire edge with π_p.*

Definition 3 (Indirect contacts of a core). *We define the indirect contacts of core π_p as the subset $\mathcal{I}(\pi_p) \stackrel{\text{def}}{=} \mathcal{V}(\pi_p) \backslash \mathcal{D}(\pi_p)$.*

From these definitions, $\mathcal{V}(\pi_6) = \{\pi_1, \pi_2, \pi_3, \pi_5, \pi_7, \pi_9, \pi_{10}, \pi_{11}\}$; $\mathcal{D}(\pi_6) = \{\pi_2, \pi_5, \pi_7, \pi_{10}\}$; and finally $\mathcal{I}(\pi_6) = \{\pi_1, \pi_3, \pi_9, \pi_{11}\}$ (see Figure 3a). From now on, we represent each active core in *gray color*; cores in its direct contact set in *dark orange color*; and cores in its indirect contact set in *light orange color* (see Figure 3b). Remark that a core π_q can belong to $\mathcal{D}(\pi_p)$ and to $\mathcal{I}(\pi_r)$ simultaneously, for any π_p, π_q and $\pi_r \in \boldsymbol{\pi}$. We represent these cores in *yellow color* (see Figure 3c). In this last figure, $\pi_3 \in \mathcal{D}(\pi_4) \cap \mathcal{I}(\pi_6)$ and $\pi_7 \in \mathcal{D}(\pi_6) \cap \mathcal{I}(\pi_4)$.

▹ **Thermal model.** We consider a Resistance-Capacitance (RC) thermal network [15, 19]. Here, the different parts of the chip and the cooling solution are represented by thermal nodes, and there are at least as many thermal nodes as there are blocks in the floorplan. Then, the temperature of a node at any given time is modeled as a function of three factors, namely (*i*) the ambient temperature, (*ii*) the power consumption, and finally (*iii*) the heat transfer between neighboring elements. For a system with $N \geq m$ thermal nodes and using Kirchoff's first law [15], Equation 2 gives the resulting system of *first order non-homogeneous differential equations*.

$$\mathbf{AT'} + \mathbf{BT} = \mathbf{P} + T_{amb}\mathbf{G} \tag{2}$$

In this equation, (*i*) matrix $\mathbf{A} = \left[a_{i,j}\right]_{N \times N}$ contains the values of the thermal capacitances and is generally a diagonal matrix, since the thermal capacitances are modeled to ground; (*ii*) matrix $\mathbf{B} = \left[b_{i,j}\right]_{N \times N}$ contains the values of the thermal conductances between vertical and lateral neighboring nodes and is generally symetric, i.e., $b_{i,j} = b_{j,i}$ for all $i \neq j$; (*iii*) column vector $\mathbf{T} = [T_i(t)]_{N \times 1}$ represents the temperatures at the thermal nodes; (*iv*) column vector $\mathbf{T'} = \left[T'_i(t)\right]_{N \times 1}$ accounts for the first-order derivative of the temperature at each thermal node with respect to time; (*v*) column vector $\mathbf{P} = [P_i]_{N \times 1}$ contains the values of the power consumption at each node; (*vi*) constant value T_{amb} denotes the ambient temperature; and finally (*vii*) column vector $\mathbf{G} = \left[g_{amb_i}\right]_{N \times 1}$ contains the values of thermal conductances between each node and the ambient temperature. If node i is not in contact with the ambient temperature (e.g., the temperature of an internal node), then the value of g_i is set to zero. We assume that the initial temperature of node i (with $i \in [1, N]$) at time instant $t = 0$ is $T_{amb} \stackrel{\text{def}}{=} 45°C$, i.e., $\mathbf{T}_{\text{init}} = [T_{amb}]_{N \times 1}$ for the entire floorplan. The values for matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{G}$ can be calculated by HotSpot [19], and these are the same matrices used internally in the default configuration of HotSpot. Isolating $\mathbf{T'}$ from Equation 2 yields Equation 3.

$$\mathbf{T'} = \mathbf{CT} + \mathbf{A}^{-1}\mathbf{P} + T_{amb}\mathbf{A}^{-1}\mathbf{G}, \text{ where } \mathbf{C} = -\mathbf{A}^{-1}\mathbf{B} \tag{3}$$

Considering only *steady-state*, we have $\mathbf{T'} = [0]_{N \times 1}$ and from Equation 2 it follows that $\mathbf{BT} = \mathbf{P} + T_{amb}\mathbf{G}$, which means that the general form of the temperatures to which the nodes will eventually converge to is provided by Equation 4.

$$\mathbf{T}^{\infty} = \mathbf{B}^{-1}\mathbf{P} + T_{amb}\mathbf{B}^{-1}\mathbf{G} \tag{4}$$

In this equation, note that $\mathbf{T}^{\infty}$ is a function of the power vector $\mathbf{P}$ and is always constant when $\mathbf{P}$ is known. In other words, $\mathbf{T}^{\infty}$ is calculated once for each power change. Moreover, in $\mathbf{B}^{-1}\mathbf{P}$; the expression $b^{-1}_{i,j} \cdot p_j$ is the amount of heat that node j contributes to the steady-state temperature of node i, i.e., T_i. With this in mind, we solve Equation 3 using the so-called method of *undetermined coefficients* [25], for which the general form of the solution is then given by Equation 5.

$$\mathbf{T} = \mathbf{T}^{\infty} + \sum_{i=1}^{N} \alpha_i \mathbf{K}_i e^{\lambda_i t} \tag{5}$$

In this equation, the tuple $(\alpha_1, \alpha_2, \ldots, \alpha_N)$ stands for N arbitrary constants α_i; the tuple $(\lambda_1, \lambda_2, \ldots, \lambda_N)$ contains the eigenvalues of the matrix $\mathbf{C} = -\mathbf{A}^{-1}\mathbf{B}$; and K_i is an eigenvector associated to λ_i. In Equation 5, note that $\sum_{i=1}^{N} \alpha_i \mathbf{K}_i e^{\lambda_i t}$ captures the transient thermal behavior of the system and models the *transient heat exchanges* between neighboring nodes, while $\mathbf{T}^{\infty}$ captures the steady-state temperatures. The values of α_i (with $i \in [1, N]$) are calculated by setting $t = 0$, i.e., by solving Equation 6.

$$\mathbf{T}_{\text{init}} - \mathbf{T}^{\infty} = \sum_{i=1}^{N} \alpha_i \mathbf{K}_i \tag{6}$$

Note that this is a *classical linear system* with N equations, where α_i (with $i \in [1, N]$) are the only unknown variables. These values are recalculated and the column vector $\mathbf{T}_{\text{init}}$ is updated each time the power vector $\mathbf{P}$ changes. Finally, given the overall structure of the matrices we have for the assumed floorplan, it holds that $\lambda_i < 0,\ \forall i \in [1, N]$ and $rank(C) = N$, i.e., all columns or rows of the matrix $\mathbf{C}$ are independent.

3 SPECIFICATION OF OUR SOLUTION

In this section we introduce the reader to the bulk piece of our contribution. We begin with a description of the inner workings of our proposed thermal-aware solution for a given core mapping in Section 3.1. Then in Section 3.2, we specify our power budget strategy to achieve TDP (the maximum power for which the system should be designed for) whenever the initial maximum power constraint P_{total} exceeds this value, i.e., $P_{\text{total}} > P_{\text{max}}$. In other words, we relax the power constraint in Equation 1. This is done with as little impact as possible on the initial power budgets per-core. Finally, we lead a discussion on how to create the worst-case core mappings (WCMs) in Section 3.3.

3.1 B-TSP for a given core mapping

In this subsection, for a given core mapping $\mathbf{X}$, the ambient temperature T_{amb} and the vector containing the target temperatures for all cores $\mathbf{T}_{\text{max}} = [T_{\text{max}}]_{N\times 1}$, we present our methodology for calculating the power budget for each active core. To this end, we opt for an *exact solution* using traditional *linear algebra* rather than an optimization-based approach, as the latter often leads to a local optimum rather than a global optimum. In addition, optimization-based approaches may find it difficult to evaluate nonlinear constraints, which is generally the case for power as a function of core execution speed. The application of B-TSP can occur either during design-time (i.e., offline), when the designer/integrator has prior knowledge of the target application, or during run-time (i.e., online), when such knowledge is not available.

B-TSP computes the power budget of each active core by using Equation 4 and setting the target temperature of each active core to $\mathbf{T}_{\text{max}}$ since this is the maximum allowable temperature at steady-state. In other words, we solve Equation 7 where the only unknown variable is the column vector $\mathbf{P}$.

$$\mathbf{T}_{\text{max}} = \mathbf{B}^{-1}\mathbf{P} + T_{amb}\mathbf{B}^{-1}\mathbf{G} \tag{7}$$

By defining $\mathbf{Z} \overset{\text{def}}{=} \mathbf{T}_{\text{max}} - T_{amb}\mathbf{B}^{-1}\mathbf{G}$, Equation 7 leads to Equation 8.

$$\mathbf{B}^{-1}\mathbf{P} = \mathbf{Z} \tag{8}$$

Because of the large size of the matrices for the assumed platform topology, we now describe the main intuition of our methodology for solving this equation for a given mapping $\mathbf{X}$ using the following example, which consists of 4 hypothetical cores $\bar{\boldsymbol{\pi}} \overset{\text{def}}{=} [\bar{\pi}_1, \bar{\pi}_2, \bar{\pi}_3, \bar{\pi}_4]$. To this end, we assume:

$$\mathbf{B}^{-1} = \begin{bmatrix} \beta_{1,1} & \beta_{1,2} & \beta_{1,3} & \beta_{1,4} \\ \beta_{2,1} & \beta_{2,2} & \beta_{2,3} & \beta_{2,4} \\ \beta_{3,1} & \beta_{3,2} & \beta_{3,3} & \beta_{3,4} \\ \beta_{4,1} & \beta_{4,2} & \beta_{4,3} & \beta_{4,4} \end{bmatrix};\ \mathbf{P} = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix};\ \mathbf{Z} = \begin{bmatrix} \zeta_1 \\ \zeta_2 \\ \zeta_3 \\ \zeta_4 \end{bmatrix}$$

Before diving into the details, we recall that in $\mathbf{B}^{-1}\mathbf{P}$; the expression $\mathrm{I}_{\bar{\pi}_i}(\bar{\pi}_j) \overset{\text{def}}{=} \beta_{i,j} \cdot p_j$ stands for the amount of heat that $\bar{\pi}_j$ contributes to the steady-state temperature of $\bar{\pi}_i$. For sake of clarity and without any loss of generality, we assume a mapping $\mathbf{X}$ where $x_2 = x_4 = 0$ (i.e., cores $\bar{\pi}_2$ and $\bar{\pi}_4$ are inactive and $p_2 = P^{\text{inact}}_{\bar{\pi}_2}$ and $p_4 = P^{\text{inact}}_{\bar{\pi}_4}$), and $x_1 = x_3 = 1$ (i.e., cores $\bar{\pi}_1$ and $\bar{\pi}_3$ are active). Since p_2 and p_4 are known, the remaining challenge is to calculate p_1 and p_3 such that Equation 1 holds true. By developing Equation 7 we obtain the traditional 2×2 system of linear equations in Equation 9.

$$\begin{bmatrix} \beta_{1,1}p_1 + \beta_{1,2}p_2 + \beta_{1,3}p_3 + \beta_{1,4}p_4 \\ \beta_{2,1}p_1 + \beta_{2,2}p_2 + \beta_{2,3}p_3 + \beta_{2,4}p_4 \\ \beta_{3,1}\beta_1 + \beta_{3,2}p_2 + \beta_{3,3}p_3 + \beta_{3,4}p_4 \\ \beta_{4,1}p_1 + \beta_{4,2}p_2 + \beta_{4,3}p_3 + \beta_{4,4}p_4 \end{bmatrix} = \begin{bmatrix} \zeta_1 \\ \zeta_2 \\ \zeta_3 \\ \zeta_4 \end{bmatrix} \tag{9}$$

In this equation, the *blue shaded* columns represent the amount of heat that $\bar{\pi}_2$ and $\bar{\pi}_4$ contribute to the steady-state temperature of $\bar{\pi}$. Now, since p_2 and p_4 are known, we can discard the *green shaded* rows and turn our attention to the first and third equation to obtain Equation 10.

$$\begin{bmatrix} \beta_{1,1} & \beta_{1,3} \\ \beta_{3,1} & \beta_{3,3} \end{bmatrix} \begin{bmatrix} p_1 \\ p_3 \end{bmatrix} = \begin{bmatrix} \zeta_1 \\ \zeta_3 \end{bmatrix} - \begin{bmatrix} (\mathrm{I}_{\bar{\pi}_1}(\bar{\pi}_2) + \mathrm{I}_{\bar{\pi}_1}(\bar{\pi}_4)) \\ (\mathrm{I}_{\bar{\pi}_3}(\bar{\pi}_2) + \mathrm{I}_{\bar{\pi}_3}(\bar{\pi}_4)) \end{bmatrix} \tag{10}$$

i.e.,

$$\overline{\mathbf{B}^{-1}} \cdot \overline{\mathbf{P}} = \overline{\mathbf{Z}} \tag{11}$$

where

$$\overline{\mathbf{B}^{-1}} \overset{\text{def}}{=} \begin{bmatrix} \beta_{1,1} & \beta_{1,3} \\ \beta_{3,1} & \beta_{3,3} \end{bmatrix};\ \mathbf{P} = \begin{bmatrix} p_1 \\ p_3 \end{bmatrix};\ \overline{\mathbf{Z}} \overset{\text{def}}{=} \begin{bmatrix} \zeta_1 \\ \zeta_3 \end{bmatrix} - \begin{bmatrix} (\mathrm{I}_{\bar{\pi}_1}(\bar{\pi}_2) + \mathrm{I}_{\bar{\pi}_1}(\bar{\pi}_4)) \\ (\mathrm{I}_{\bar{\pi}_3}(\bar{\pi}_2) + \mathrm{I}_{\bar{\pi}_3}(\bar{\pi}_4)) \end{bmatrix}$$

Informally speaking, to calculate the power budgets of the active cores, we remove the rows corresponding to the inactive cores (see the *green shaded* rows) and we consider their contributions to the steady-state temperature of the active cores (see the *blue shaded* rows). Finally, we check that Equation 1 holds. The pseudo-code of the B-TSP strategy for a given mapping $\mathbf{X}$ is provided by Algorithm 1.

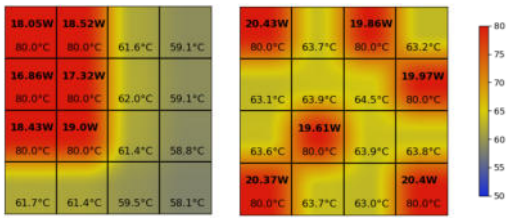
This algorithm receives as inputs (1) the vector of target temperatures for all cores $\mathbf{T}_{\text{max}}$; (2) the ambient temperature T_{amb}; (3) the core mapping $\mathbf{X}$; (4) the platform $\boldsymbol{\pi}$ with m cores; (5) the array of

Algorithm 1: B-TSP specification

Inputs : $T_{\max}$, $\mathbf{T}_{amb}$, mapping $\mathbf{X}$, $\boldsymbol{\pi}$, $\mathbf{P}_{\pi}^{\text{inact}}$, Matrices $\mathbf{B}$ and $\mathbf{G}$.
Output : Power vector distribution $\mathbf{P}^{\text{B-TSP}}$.

1 $\mathbf{Z} \leftarrow \mathbf{T}_{\max} - T_{amb}\mathbf{B}^{-1}\mathbf{G}$ (see Equation 7)
2 $\mathbf{totI} == [0]_{m\times 1}$
3 **foreach** $x_i \in \boldsymbol{X}$ **do**
4 **if** $x_i == 0$ **then**
5 **foreach** $\pi_p \in \boldsymbol{\pi}$ **do**
6 $\mathrm{I}_{\pi_p}(\pi_i) \leftarrow \beta_{p,i} P_{\pi_i}^{\text{inact}}$
7 $\mathrm{totI}_{\pi_p} \leftarrow \mathrm{totI}_{\pi_p} + \mathrm{I}_{\pi_p}(\pi_i)$
8 $\overline{\mathbf{Z}} \leftarrow \texttt{ReduceVector}(\mathbf{Z} - \mathbf{totI}, \mathbf{X})$
9 $\overline{\mathbf{B}^{-1}} \leftarrow \texttt{ReduceMatrix}(\mathbf{B}^{-1}, \mathbf{X})$
10 $\overline{\mathbf{P}} \leftarrow \texttt{ReduceVector}(\mathbf{P}, \mathbf{X})$
11 $\overline{\mathbf{P}}^{\text{B-TSP}} \leftarrow \texttt{ComputeBTSP}(\overline{\mathbf{B}^{-1}}, \overline{\mathbf{Z}})$
12 $\mathbf{P}^{\text{B-TSP}} \leftarrow \texttt{Merge}(\overline{\mathbf{P}}^{\text{B-TSP}}, \mathbf{P}_{\pi}^{\text{inact}}, \mathbf{X})$
13 **if** $\texttt{Sum}(\boldsymbol{P}^{\text{B-TSP}}) \leq$ TDP *(see Equation 1)* **then**
14 **return** $\mathbf{P}^{\text{B-TSP}}$
15 **else**
16 **return** TDP exceeded;

powers of inactive cores $\mathbf{P}_{\pi}^{\text{inact}}$; and finally (6) matrices $\mathbf{B}$ and $\mathbf{G}$ fed to the thermal model. After the initializations (lines 1 and 2), we check for each $x_i \in \mathbf{X}$ (line 3) whether the core π_i is inactive or not. If yes (line 4), we calculate its thermal contribution to all cores in π (lines 5 and 6) and update **totI** (line 7). Then, we construct $\overline{\mathbf{B}^{-1}}$; $\overline{\mathbf{P}}$ and $\overline{\mathbf{Z}}$ to solve the resulting traditional system of linear equations and obtain the power distribution $\overline{\mathbf{P}}^{\text{B-TSP}}$ of the active cores (line 11). Finally, we build the power distribution $\mathbf{P}^{\text{B-TSP}}$ by merging $\overline{\mathbf{P}}^{\text{B-TSP}}$ and $\mathbf{P}_{\pi}^{\text{inact}}$ (line 12). If Equation 1 is true (line 13), we return $\mathbf{P}^{\text{B-TSP}}$ (line 14). Otherwise, TDP is exceeded and no power distribution could be found for the active cores. Figure 4 illustrates the steady-state temperatures obtained by using B-TSP for the core mappings with six active cores discussed in Section 1. The total power for these mappings is 108.18 W and 120.64 W, which means an increase of 2.97% and 1.24%, respectively.



(a) Concentrated cores (b) Distributed cores

Figure 4: Two distinct mappings for 6 **active cores under** B-TSP

It follows from the description of B-TSP that if we know the power profile of a particular application in advance and can assign fixed power budgets to some active cores, say π_{a_1}, π_{a_2}, ... and π_{a_k} ($k \leq m$), we can use this information for a more efficient power budget distribution. To this end, we add these special active cores to the original set of inactive cores but keep their power budgets $P_{\pi_{a_1}}^{\text{act}}$, $P_{\pi_{a_2}}^{\text{act}}$, ..., $P_{\pi_{a_k}}^{\text{act}}$ unchanged in $\mathbf{P}_{\pi}^{\text{inact}}$, and apply the B-TSP methodology. In this way, we potentially increase the power budget of the other active cores as well as their performance, as we enable them to handle larger workloads.

3.2 Relaxing the power constraint

The value of TDP can vary drastically from one platform to another. For AMD multi-core architectures with homogeneous 16 cores, this value varies between 85 W and 320 W. Ideally, the total power of a chip should remain permanently below TDP, i.e., Equation 1 must always hold, since TDP is the maximum power the system should be designed for. In practice, however, this is not always the case; this value may be exceeded during short time intervals. In such situations, appropriate actions must be taken to bring the system back to a power level below TDP as quickly as possible. This is done by *adjusting* the power budgets assigned to some individual active cores. In this subsection, we describe such a strategy for a given mapping with minimal impact on system performance. To this end, we start from Equation 12.

$$P_{\pi}^{\text{total}} = \sum_{p=1}^{m} \left\{ x_p \cdot P_{\pi_p}^{\text{act}} + (1 - x_p) \cdot P_{\pi_p}^{\text{inact}} \right\} > \text{TDP} \tag{12}$$

From this equation, we define $\Delta P \stackrel{\text{def}}{=} P_{\pi}^{\text{total}} - \text{TDP}$, i.e., the *total overbudgeting power*. Since we want to adjust the power budgets of some cores so that Equation 1 is satisfied again every time this operation is legally possible, we proceed as follows.

After the initial assignment of power budgets per-core, the budgets of inactive cores as well as some active cores cannot be reduced under any circumstances. The latter may be the case due to the amount and/or nature of workloads executed by these specific cores, otherwise a deadline will be missed. For this reason, we organize the cores in π into three different categories: (1) the inactive cores $\pi_{i_1}, \pi_{i_2}, \ldots, \pi_{i_{\ell_1}}$ with power budgets $P_{\pi_{i_1}}^{\text{inact}}, P_{\pi_{i_2}}^{\text{inact}}, \ldots, P_{\pi_{i_{\ell_1}}}^{\text{inact}}$(2) the active cores $\pi_{a_1}, \pi_{a_2}, \ldots, \pi_{a_{\ell_2}}$ with power budgets $P_{\pi_{a_1}}^{\text{act}}, P_{\pi_{a_2}}^{\text{act}}, \ldots, P_{\pi_{a_{\ell_2}}}^{\text{act}}$, which *cannot* be reduced; and finally (3) the active cores $\pi_{b_1}, \pi_{b_2}, \ldots, \pi_{b_{\ell_3}}$ with power budgets $P_{\pi_{b_1}}^{\text{act}}, P_{\pi_{b_2}}^{\text{act}}, \ldots, P_{\pi_{b_{\ell_3}}}^{\text{act}}$, which *can* be reduced. We have $\ell_1 + \ell_2 + \ell_3 = m$. If there is no core in the last category, then there is no solution and there is nothing we can do to satisfy Equation 1 again. Otherwise, $\ell_3 \neq 0$, we turn our attention to the cores in the last category and reduce the original power budget of each of these cores in a "fair manner" by using their original power budgets as their weights. Formally, we apply Equation 13.

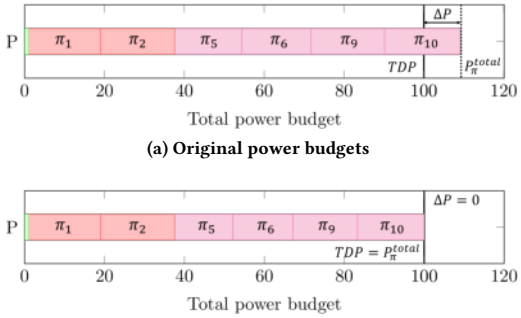
$$P_{\pi_{b_j}}^{\text{ajust}} \stackrel{\text{def}}{=} P_{\pi_{b_j}}^{\text{act}} \left(1 - \frac{\Delta P}{\sum_{j=1}^{\ell_3} P_{\pi_{b_j}}^{\text{act}}} \right) \tag{13}$$

From this equation, exactly ΔP of P_{π}^{total} is reduced and the total adjusted power P_{π}^{ajust} is given by Equation 14, which means that Equation 1 is again satisfied.

$$P_{\pi}^{\text{ajust}} \stackrel{\text{def}}{=} \sum_{j=1}^{\ell_1} P_{\pi_{i_j}}^{\text{inact}} + \sum_{j=1}^{\ell_2} P_{\pi_{a_j}}^{\text{act}} + \sum_{j=1}^{\ell_3} P_{\pi_{b_j}}^{\text{ajust}} = \text{TDP} \tag{14}$$

Figure 5 illustrates the case for the concentrated core mapping considered in Section 3.1) when TDP = 100 W. In this example,

cores π_1, π_2, π_5, π_6, π_9 and π_{10} are active and the remaining cores are inactive. Then the total power budget $P_\pi^{\text{total}} = 108.18\ W$ exceeds the TDP value (see Figure 5a). Without loss of generality, we assume that the cores π_1 and π_2 belong to the second category, while π_5, π_6, π_9 and π_{10} belong to the third category. Using our specifications, the power budgets of the cores in this last category are adjusted, and the result of this operation is shown in Figure 5b.



(a) Original power budgets

(b) Adjusted power budgets

Figure 5: Power budget redistribution

3.3 WCM: The worst-case core mappings

In this subsection, we report our results on the worst-case mappings (WCMs) for a given number of simultaneously active cores. The WCMs are formally defined as follows.

Definition 4 (WCMs). *For a given number of simultaneously active cores, a WCM is defined as a core mapping for which the per-core power budgets, and hence the total power budgets determined by the* B-TSP *scheme, are the most pessimistic, i.e., a core mapping that leads to the lowest power values while no node in the floorplan (at steady state) exceeds the critical threshold temperature* $T_{\max}$.

From Definition 4, it follows that this step plays a central role in the analysis of any safety-critical system geared by a multi-core platform, and the situation worsens under thermal-aware design. Here we provide the reader with a *constructive* approach that describes how to create these WCMs. Unlike B-TSP computations from Section 3.1 for a given core mapping, system designers/integrators and/or resource management techniques will have a compelling tool with this WCMs information to abstract from core mapping decisions. Any alternative core mapping for the same number of simultaneously active cores will inevitably result in more efficient utilization of resources.

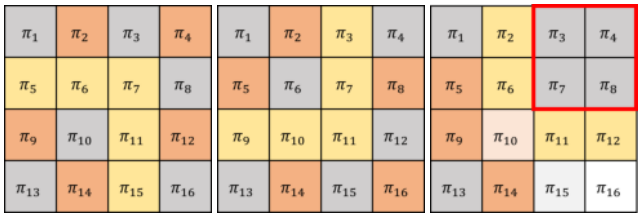
To better understand our approach, we introduce below some definitions for a hypothetical platform π with m^2 cores arranged in m rows and m columns (see Figure 3 for an example with $m = 4$).

Definition 5 (Topologically equivalent cores). *Two cores π_i and π_j in π are called topologically equivalent if they have the same number of direct and indirect contacts in their vicinity.*

Definition 6 (Topologically equivalent core mappings). *Two core mappings X_1 and X_2 are called topologically equivalent if they have the same number of active cores and there exists a one-to-one function from X_1 to X_2 that preserves all properties in each mapping (i.e., each active core in X_1 is topologically equivalent to one and only one active core in X_2 with the same number of active and inactive cores in its vicinity).*

Definition 7 (block-core). *A block-core B_k is defined as a cluster of k^2 adjacent active cores (with $k \leq m$) arranged in k rows and k columns.*

Figure 6 illustrates Definitions 5, 6 and 7. In Figure 6a, core π_1 and core π_4 are topologically equivalent. Core mapping X_1 (Figure 6a) and core mapping X_2 (see Figure 6b) are topologically equivalent. In fact, both mappings have the same number of active cores (here 6) and each active core in X_1 is topologically equivalent to one and only one active core in X_2. For example, the cores π_8 and π_{10} in X_1 are topologically equivalent to the cores π_{15} and π_6 in X_2, respectively, etc. But the core mapping X_1 is not equivalent to X_3 (see Figure 6c). Note here that the active core π_7 in X_3 is not topologically equivalent to any active core in X_1. In fact, this core has three active cores (here π_3, π_4, and π_8) in its vicinity and none of the active cores in X_1 satisfy this constraint. Finally, Figure 6c shows a block-core B_2 with four active cores (here π_3, π_4, π_7 and π_8) inside the red box.



(a) Mapping X_1 **(b) Mapping X_2** **(c) X_3 and Block core**

Figure 6: Topologically equivalent mappings and Block core.

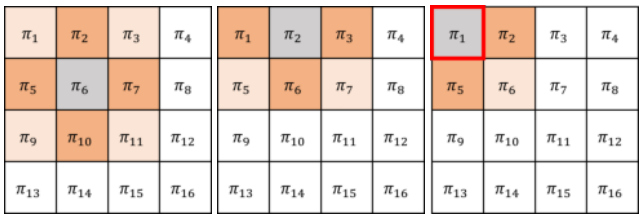
Here we have all the materials we need to build our WCMs. Without any loss of generality and for the sake of simplicity of exposition, we assume that $P_{\pi_p}^{\text{inact}} = 0$ for all $\pi_p \in \pi$. From equation 1 it follows that

$$P_\pi^{\text{total}} = \sum_{p=1}^{m} \left\{ x_p \cdot P_{\pi_p}^{\text{act}} + (1 - x_p) \cdot P_{\pi_p}^{\text{inact}} \right\} = \sum_{p=1}^{m} x_p \cdot P_{\pi_p}^{\text{act}}$$

For the WCMs, the challenge is to find the positions of the active cores that *minimizes* P_π^{total}. To this end, we start with only *one active core* and follow an *iterative approach.*

▹ **Locating one active core.** In this case $P_\pi^{\text{total}} = P_{\pi_p}^{\text{act}}$ and from Definition 5, there are only 3 distinct positions where to locate this core since any other position is topologically equivalent to one of these (see Figure 7).

From these positions it follows that placing the active core in the corner (see Figure 7c) leads to the lowest P_π^{total}. This is because at this position it has the least possibility of transferring heat to the inactive cores in its vicinity. Moreover, this position defines the smallest block-core B_1 from this corner.



(a) Center location (b) Border location (c) Corner location

Figure 7: Possible locations of a core.

▹ **Locating two active cores.** We assume that the two cores to be placed are π_i and π_j (with $i \neq j$). In this case, $P_\pi^{\text{total}} = P_{\pi_i}^{\text{act}} + P_{\pi_j}^{\text{act}}$. We simply write $P_\pi = p_i + p_j$ to avoid heavy and overloaded notations. From Equation 8 and by applying the methodology presented in Section 3.1, we get:

$$\overline{\mathbf{B}^{-1}} \cdot \overline{\mathbf{P}} = \overline{\mathbf{Z}} \tag{15}$$

where

$$\overline{\mathbf{B}^{-1}} \stackrel{\text{def}}{=} \begin{bmatrix} \beta_{i,i} & \beta_{i,j} \\ \beta_{j,i} & \beta_{j,j} \end{bmatrix};\ \mathbf{P} = \begin{bmatrix} p_i \\ p_j \end{bmatrix};\ \overline{\mathbf{Z}} \stackrel{\text{def}}{=} \begin{bmatrix} \zeta_i \\ \zeta_j \end{bmatrix} - \begin{bmatrix} \sum_{\pi_u^{\text{inact}}} \mathrm{I}_{\pi_i}(\pi_u) \\ \sum_{\pi_u^{\text{inact}}} \mathrm{I}_{\pi_j}(\pi_u) \end{bmatrix}$$

Note that: (1) $\beta_{i,j} = \beta_{j,i} \stackrel{\text{def}}{=} \beta$, i.e., the heat transfer from π_i to π_j is the same as the heat transfer from π_j to π_i; (2) $\zeta_i = \zeta_j = T_{\max}$, since $g_i = 0$ for all cores; (3) $\mathrm{I}_{\pi_i}(\pi_u) = \beta_{i,u} \cdot p_u = 0$, as $p_u = 0$ for each inactive core π_u by assumption. Similarly, $\mathrm{I}_{\pi_j}(\pi_u) = \beta_{j,u} \cdot p_u = 0$. From these observations we obtain:

$$\overline{\mathbf{B}^{-1}} = \begin{bmatrix} \beta_{i,i} & \beta \\ \beta & \beta_{j,j} \end{bmatrix};\ \mathbf{P} = \begin{bmatrix} p_i \\ p_j \end{bmatrix};\ \overline{\mathbf{Z}} \stackrel{\text{def}}{=} \begin{bmatrix} T_{\max} \\ T_{\max} \end{bmatrix}$$

Solving Equation 15 for p_i and p_j, we have:

$$\begin{bmatrix} p_i \\ p_j \end{bmatrix} = \frac{1}{\beta_{i,i}\beta_{j,j} - \beta^2} \begin{bmatrix} \beta_{j,j} & -\beta \\ -\beta & \beta_{i,i} \end{bmatrix} \begin{bmatrix} T_{\max} \\ T_{\max} \end{bmatrix}$$

This means that:

$$\begin{aligned} P_\pi &= p_i + p_j \\ &= \frac{T_{\max}}{\beta_{i,i}\beta_{j,j} - \beta^2} \cdot (\beta_{i,i} + \beta_{j,j} - 2\beta) \end{aligned}$$

This is a continuously decreasing function of β, i.e., the larger the value of β, the smaller the total power (see the trend in Figure 8).

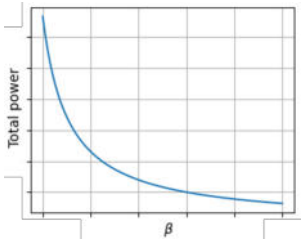
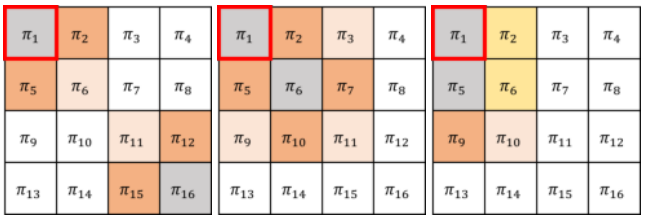


Figure 8: P_π as a function of β.

From this figure it can be seen that P_π is minimal for the largest value of β, i.e., when cores π_i and π_j are in direct contact. In fact, assuming the corner position for core π_i as discussed in the previous case, there are only two alternative positions for π_j – at π_6 (see Figure 9b) and at π_5 (see Figure 9c) – so that it is in contact with π_i. The latter position is topologically equivalent to π_2. At these two possible positions, the heat transfer between the two cores is the largest (i.e., the largest value of β) when these cores are in direct contact[1]. Without loss of generality, we place π_j at π_5.



(a) Non-contact (b) Indirect contact (c) Direct contact

Figure 9: Impact of the contacts.

At this point, we have everything we need to describe our methodology for constructing the WCMs for a platform with m^2 number of cores arranged in m rows and m columns. We call the methodology "*Fill Edge Block-core First*" (FBF) because it aims to activate cores to fill the edge of each block-core. Algorithm 2 presents the pseudo-code, which is illustrated in Figure 10 for the 16-core homogeneous architecture considered in this paper.

This algorithm receives as inputs (1) the vector of target temperatures for all cores $\mathbf{T}_{\max}$; (2) the ambient temperature T_{amb}; (3) the platform $\boldsymbol{\pi}$ with m cores; and finally (4) matrices $\mathbf{B}$ and $\mathbf{G}$ fed to the thermal model. After the initializations (lines 1 and 2), we locate the first active core at a corner and build the first block-core (line 3 and line 4). Then we build the WCMs as the number of active cores increases. In particular, we check if there is any space left at an edge of B_1. Since this is the case (at π_2 and π_5), we check if the new core fills the edge. The answer is affirmative, we activate π_5 (see Figure 10a) and increase the number of active cores (line 28).

We repeat the previous steps for the new active core. Since it completes the other edge of B_1, we activate π_2 (see Figure 10b) and increase the number of active cores to 4. With this new core, there is no more space at any of the edges of B_1, so the algorithm jumps to line 25 to complete the new block by activating π_6; we increment the variable k and build the new block B_2 (see Figure 10c). Then, the number of active cores is increased to 5. As there is still space in one edge of B_2 and the new core does not complete one edge and the list of segregated cores is empty we jump to line 19 and activate π_9 (see Figure 10d). This is the position where the difference between active and inactive cores in the vicinity is the more favorable (2 to 3). We increase the number of active cores and find that by activating π_{10} we fill the edge, we do it (see Figure 10h) and increase the number of active cores to 7. There is space in an edge of B_2 (at π_3 and π_7), but activating one of these cores will not fill the edge. In this case, we activate π_{13} (see Figure 10f). Since this core is not in direct contact with B_2, it is added to the list of segregated cores (line 12), and the number of active cores is increased to 8. Note that

[1]Figure 9a shows the most favorable configuration from the power point of view.

Algorithm 2: FBF specification

Inputs : $T_{\max}$, $\mathbf{T}_{amb}$, $\boldsymbol{\pi}$, Matrices **B** and **G**.
Output : WCMs.

1 $n_{\text{act}} \leftarrow 1$
2 Segregated $\leftarrow$ [ ]
3 Locate first active core at a corner (e.g., at π_1)
4 Build the first block-core B_1
5 **do**
6 **if** *There is room in one edge of* B_k **then**
7 **if** *The new core completes one edge* **then**
8 Fill the edge
9 **else**
10 **if** *There are* Segregated *cores* **then**
11 Locate the new core alongside
12 Segregated $\leftarrow$ Segregated +new core
13 **if** *This new group fill one edge* **then**
14 Fill the edge
15 Empty Segregated
16 **else**
17 **go to line** 19
18 **else**
19 Locate the new core where the difference between active and inactive cores in the vicinity is the more favorable
20 **if** *Two positions return the same value* **then**
21 Locate the new core at the position with more active cores in the vicinity
22 **if** *The new core is not in contact with* B_k **then**
23 Segregated $\leftarrow$ this core
24 **else**
25 Complete a new block-core
26 $k \leftarrow k + 1$
27 Build the new block-core B_k
28 $n_{\text{act}} \leftarrow n_{\text{act}} + 1$
29 **while** $n_{\text{act}} < m$
30 **return** WCMs for all active cores;

there is still space at the edge of B_2 (at π_3 and π_7), but the new core does not complete the edge by itself. Since the segregated list is not empty, we temporarily activate π_{14}. The new group of cores in Segregated can fill the other edge of B_2. We deactivate π_{13} and π_{14} and activate π_3 and π_7 (see Figure 10g). Finally, we increase the number of cores and repeat this process until all cores are activated (see from Figure 10h to Figure 10o).

4 TRANSIENT PHASE THERMAL BEHAVIOR

In this section, the effects of the thermal behavior of the system in the transient phase on our proposed B-TSP strategy are examined and addressed. Figure 11 illustrates two different scenarios of challenging the temperature fluctuations in this phase. Specifically, we consider the same settings as in Section 1, where the maximum temperature is $T_{\max} = 80°C$. Then we assume a case study where the system temperature is already at $T_{\max}$ from time instant $t = 0$. Figure 11a shows the issue on a particular active core when the number of active cores drops from 6 to 5; while Figure 11b shows the issue on the same core when the number of

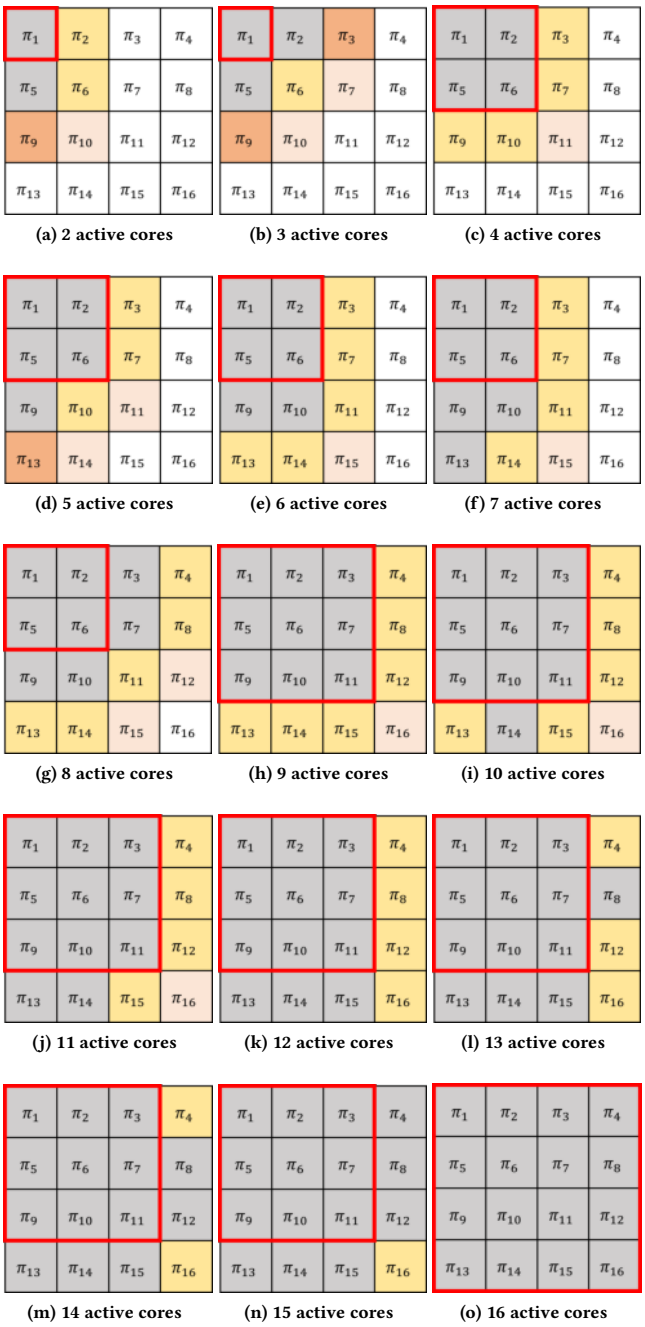
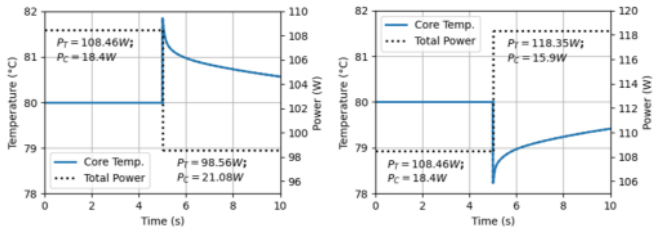


Figure 10: Worst case mapping from 2 to 16 active cores.

active cores jumps from 6 to 7 at time instant $t = 5s$. In the first scenario, $T_{\max}$ is exceeded, because the reduction in the number of active cores decreases the total power at the chip level (it drops from $P_T = 108.46\ W$ to $98.56\ W$); however, the power budget per-core increases (it jumps from $P_C = 18.4\ W$ to $21.08\ W$). Since the core under scrutiny was already operating at $T_{\max}$, any increase in the per-core power budget inevitably leads to thermal violation.

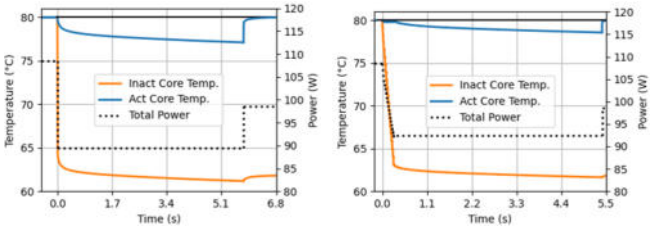
In the second case study, the picture is different. Here, the total power at the chip level increases (it jumps from $P_T = 108.46\ W$ to $118.35\ W$), but the power budget per-core decreases (it drops from $P_C = 18.4\ W$ to $15.9\ W$), which is beneficial to the core from a thermal point of view.



(a) From 6 to 5 active cores (b) From 6 to 7 active cores

Figure 11: Transient behavior when varying the number of active cores

Pagani et al [12, 13] has illuminated the above problem and proposed a good solution to it. However, his approach is only applicable to TSP. In this work, we have shown the limitations of the methodology and demonstrated the dominance of B-TSP, which necessitates a reconsideration of the thermal behavior in the transient phase. We approach the problem from a different perspective and propose two solutions (see Figure 12). The first solution is rather simple and intuitive, while the second solution is more sophisticated.



(a) Two phases approach (b) Gradual approach

Figure 12: Different approaches for the transient phase

Figure 12a illustrates the first solution, using a two-phase approach to avoid thermal violations. Here, we turn off all cores that were meant to be switched off from the initial mapping and keep the power of the active cores unchanged (at 89.98 W) long enough so that $T_{\max}$ is not violated after the active cores are assigned the new power budgets $\mathbf{P}^{\text{B-TSP}}$.

In contrast, Figure 12b presents the second solution, which uses an iterative approach in which the power of the inactive cores is reduced to $P^{\text{inact}}_{\pi_p}$ *gradually* rather than all at once. During the reduction, the power of the active cores is also gradually increased. When all inactive cores are at $P^{\text{inact}}_{\pi_p}$, we also leave the power unchanged (at 92.34 W), allowing the platform to run larger workloads during this time window.

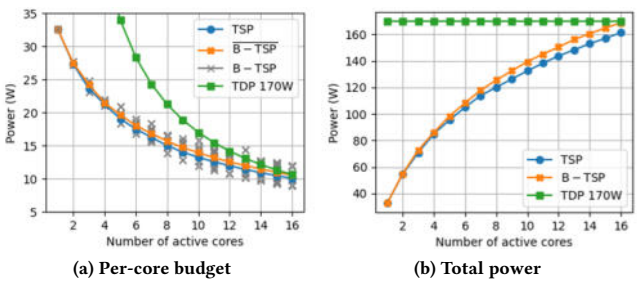
5 EXPERIMENTAL RESULTS

In this section, we report on the experiments performed to validate the theoretical results presented.

▹ **Setup.** Our simulations are performed using the AMD EPYC 7351 [1] platform specifications. This is a $2D$-homogeneous 16-core architecture in 14 nm technology with TDP = 170 W. The AMD EPYC 7351 platform uses the Zen (Naples) architecture with Socket SP3 and operates by default at 2.4 GHz. The available frequencies range from 0.1 GHz to 2.9 GHz. We consider one thermal block for each core and independent thermal blocks for the L2 caches and other hardware components, with a combined area of 213 mm^2. The values for matrices A and B and vector G for this target platform are obtained using HotSpot [19] in the default configuration in block model mode. We set the ambient temperature $T_{amb} = 45°C$ and $T_{\max} = 80°C$.

▹ **Evaluation metrics.** We evaluate the effectiveness of our B-TSP strategy by comparing it to two other strategies for allocating per-core power budgets using the worst-case mappings discussed in Section 3.3: (1) TDP, which uses a fixed power budget for each core and/or for the entire chip, (2) TSP of Pagani et al. [13], which assigns fixed power budget constraints based on the number of active cores to ensure that at least one active core reaches its maximum allowable temperature at steady-state. Recall that our B-TSP strategy, on the other hand, assigns safe and unbalanced power budgets based on the number of active cores to ensure that all active cores reach their maximum allowable temperature at steady-state. It is worth noting that TDP also defines an upper bound on the power budget per-core.

▹ **Interpretation of the results.** Figure 13 compares all three strategies for a per-core power budget (see Figure 13a) and total power at the chip level (see Figure 13b). As can be seen from these figures, B-TSP consistently dominates TSP in both aspects. In Figure 13a, the power budgets per core decrease with the number of simultaneously active cores for all approaches. However, the gap between B-TSP and TSP in the best case scenario (achieved with 13 active cores) is about 2.91 W, which corresponds to an increase of 25.46%. At the chip level, the gap between the two approaches increases as the number of cores increases. It reaches 7.4 W with 16 active cores, which corresponds to an increase of 4.59%. It is important to point out that as long as the per-core power budgets computed by B-TSP are not exceeded, running different applications will not result in thermal violation.



(a) Per-core budget (b) Total power

Figure 13: Power distribution: TSP **vs.** B-TSP

Figure 14 evaluates the performance of our WCM strategy. Here we compared to an exhaustive search and considered all possible combinations of core mappings (RDMs) for any number of active cores (see "gray crosses"). As can be seen from the figure, our FBF strategy (see "orange curve") consistently assigns the lowest total power budgets to the active cores at the chip level for any number of active cores. The best-case core mapping (BCM) differs from WCMs in that it prioritizes maximizing the power budget per core while ensuring that no node in the floorplan exceeds the critical threshold temperature $T_{\max}$ (as indicated by the "green curve") under steady-state conditions. Essentially, BCM minimizes thermal interference between active cores.

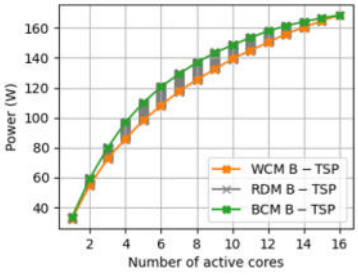


Figure 14: All possible mappings under B-TSP

The thermal values attained by each active core under the maximum possible thermal stress in configurations with TSP and B-TSP are depicted in Figure 15. In the case of TSP, the figure displays all the thermal values of each active core (indicated by "gray crosses") and highlights the average chip temperature (see "green curve") and the temperature of the coolest core (see "blue curve"). Unlike TSP, where the thermal gap increases with the number of active cores, B-TSP (see "orange curve") actually manages to achieve $T_{\max}$ in all active cores. This enables system designers/integrators to assign higher power consumption tasks without exceeding the maximum thermal limit $T_{\max}$, ultimately leading to higher performance gains.

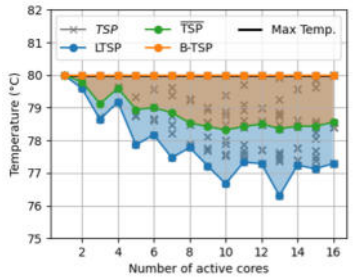


Figure 15: Thermal distribution TSP **vs.** B-TSP

6 STATE OF THE ART

Every year, a considerable body of work is presented on techniques for thermal-aware management and core mapping strategies for multi-core platforms. This section provides an overview of the state-of-the-art. Although not exhaustive, it provides a representative view of the research conducted in this paper. Core mapping is a critical step that directly impacts system efficiency, power consumption, and temperature. Depending on when tasks are assigned to cores, mapping methodologies can be broadly categorized as (1) *run-time mapping*, which is more appropriate for dynamic workloads, or (2) *design-time mapping*, which is more appropriate for predetermined workloads.

About run-time mapping strategies: These approaches handle the task-to-core mapping online based on the current state of the system. A variety of run-time thermal-aware mapping approaches have been proposed in the literature, including (1) Load Balancing [11, 20], in which tasks are dynamically reallocated between cores based on their current temperature and workload to distribute the heat load evenly across the system; and (2) Task Migration [6, 9, 22, 23], in which tasks are migrated from high-temperature cores to low-temperature cores to reduce the risk of thermal overload. In [11], Enric Musoll focused on the power consumption and thermal behavior of the core pool under different workload conditions and different load-balancing schemes for mesh- and ring-based multi-core architectures. Wu et al. [22] developed a task allocation algorithm along with a migration and exchange algorithm to prevent thermal saturation by creating a power density map with more uniform distribution. Yun et al. [23] took this direction one step further and proposed an efficient run-time thermal-aware scheduler using job migration and power-gating techniques to avoid the thermal hotspots for critical applications on a multi-core platform.

About design-time mapping strategies: These strategies optimize the task-to-core mapping offline, i.e., before the system is deployed. They aim to minimize the risk of thermal overload by finding the "optimal" core mapping that maximizes performance while keeping the temperature within acceptable limits. A range of design-time thermal-aware mapping approaches have been proposed in the literature, including (1) Mathematical Programming [4, 5, 16], which typically involve formulating the thermal-aware mapping problem as an optimization problem and solving it using specialized algorithms; (2) Heuristics [17, 18], which are algorithms that aim to find a good, but not necessarily optimal solutions to the thermal-aware mapping problem in a reasonable amount of time; and finally (3) Machine Learning techniques [10, 24], which learn the relationship between the task-to-core mapping and the thermal behavior of the system to predict the temperature at a given horizon and find an "optimal" solution. Benedikt et al. [4] proposed an optimization-based task mapping for thermal-aware allocation of safety-critical avionics workloads under time isolation constraints, with the goal of minimizing average power consumption. Shehzad et al. presented a heuristic in their paper [17] which adjusts the thermal threshold value dynamically, resulting in a reduction of thermal violations and power consumption. On another front, Zhang et al. [24] utilized advanced deep reinforcement learning to develop a novel hotspot-aware task-to-core mapping technique to improve lifetime and reliability of the platform.

In general, thermal-aware mapping approaches at both run-time and design-time have the potential to enhance the performance and reliability of multi-core systems by mitigating the risk of thermal overload. Nonetheless, these approaches rely on the workload and

fail to equip the system designer with a mechanism for establishing a per-core power budget solely based on the number of active cores. Consequently, such methods do not alleviate the designer's concerns associated with core mapping. In this context, Pagani et al.[13] introduced a power budget concept named Thermal Safe Power (TSP). This approach allocates the same power budget to every active core, but it has certain restrictions, such as leaving some cores underutilized and struggling to handle dynamic workloads effectively. Wang et al. [21] designed a reactive power budget strategy combined with dynamic thermal management (DTM) that migrates tasks from "hot" to "cool" cores to ensure that no temperature constraint is violated. However, the authors formulate an optimization problem in the discrete domain, resulting in a loss of accuracy. In this paper, we fill all these gaps.

7 CONCLUSION

This paper addresses the issue of thermal management of safety-critical multi-core architectures. Within this framework, a new per-core power budget strategy is proposed that is scalable and enables system performance optimization while abstracting from mapping concerns. A new strategy is presented that allows us to derive worst-case mappings as a function of the number of active cores from a power consumption perspective. Finally, the effectiveness of our solution has been demonstrated through intensive simulations with the homogeneous AMD EPYC 7351 16 cores platform. Last but not least, the gains over the most advanced solution on this topic in the literature reach up to 25.46% in the best case. In our roadmap, we plan to extend our strategy to heterogeneous architectures and conduct experiments on real platforms to show the differences due to model abstraction.

ACKNOWLEDGMENTS

This work was supported by the CISTER Research Unit (UIDP/UIDB /04234/2020), financed by National Funds through FCT/MCTES (Portuguese Foundation for Science and Technology); also by the scholarship grant ref. 2020.09635.BD, financed by FCT through the European Social Fund (ESF) and the Regional Operational Programme (ROP) Norte 2020.

REFERENCES

[1] AMD. 2017. AMD EPYC™ 7351. https://www.amd.com/en/product/1986.
[2] AMD. 2019. AMD EPYC™ 7742. https://www.amd.com/en/product/8761.
[3] Mohsen Ansari, Sepideh Safari, Amir Yeganeh-Khaksar, Mohammad Salehi, and Alireza Ejlali. 2019. Peak Power Management to Meet Thermal Design Power in Fault-Tolerant Embedded Systems. *IEEE Transactions on Parallel and Distributed Systems* 30, 1 (2019), 161–173. https://doi.org/10.1109/TPDS.2018.2858816
[4] Ondřej Benedikt, Michal Sojka, Pavel Zaykov, David Hornof, Matěj Kafka, Přemysl Šůcha, and Zdeněk Hanzálek. 2021. Thermal-Aware Scheduling for MPSoC in the Avionics Domain: Tooling and Initial Results. In *27th Int. Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*. IEEE, Houston, TX, USA, 159–168. https://doi.org/10.1109/RTCSA52859.2021.00026
[5] T. Chantem, R. P. Dick, and X. S. Hu. 2008. Temperature-Aware Scheduling and Assignment for Hard Real-Time Applications on MPSoCs. In *Design, Automation and Test in Europe*. IEEE, Munich, Germany, 288–293. https://doi.org/10.1109/DATE.2008.4484694
[6] Ting-Hsuan Chien and Rong-Guey Chang. 2016. A thermal-aware scheduling for multicore architectures. *Journal of Systems Architecture* 62 (2016), 54–62. https://doi.org/10.1016/j.sysarc.2015.12.003
[7] Intel. 2019. Intel® Xeon® Platinum 9282 Processor. https://www.intel.com/content/www/us/en/products/sku/194146/intel-xeon-platinum-9282-processor-77m-cache-2-60-ghz/specifications.html.
[8] Intel. 2019. Intel® Xeon® W-3275M Processor. https://www.intel.com/content/www/us/en/products/sku/193754/intel-xeon-w3275m-processor-38-5m-cache-2-50-ghz/specifications.html.
[9] Deguang Li, Ruiling Zhang, Shijie Jia, Yanling Jin, Youzhong Ma, and Junke Li. 2019. An Improved Dynamic Power Management Approach by Process Migration for Multi-Core Systems. In *International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*. IEEE, Atlanta, GA, USA, 368–372. https://doi.org/10.1109/iThings/GreenCom/CPSCom/SmartData.2019.00081
[10] Shiting (Justin) Lu, Russell Tessier, and Wayne Burleson. 2015. Reinforcement Learning for Thermal-Aware Many-Core Task Allocation. In *Proceedings of the 25th Edition on Great Lakes Symposium on VLSI* (Pittsburgh, Pennsylvania, USA) *(GLSVLSI '15)*. ACM, NY, USA, 379–384. https://doi.org/10.1145/2742060.2742078
[11] Enric Musoll. 2008. A Thermal-Friendly Load-Balancing Technique for Multi-Core Processors. In *International Symposium on Quality Electronic Design (ISQED)*. IEEE, San Jose, CA, USA, 549–552. https://doi.org/10.1109/ISQED.2008.4479794
[12] S. Pagani. 2016. *Power, Energy, and Thermal Management for Clustered Manycores*. Ph. D. Dissertation. Karlsruher Institut für Technologie.
[13] Santiago Pagani, Heba Khdr, Jian-Jia Chen, Muhammad Shafique, Minming Li, and Jörg Henkel. 2017. Thermal Safe Power (TSP): Efficient Power Budgeting for Heterogeneous Manycore Systems in Dark Silicon. *IEEE Trans. Comput.* 66, 1 (2017), 147–162. https://doi.org/10.1109/TC.2016.2564969
[14] Javier Pérez Rodríguez and Patrick Meumeu Yomsi. 2021. An Efficient Proactive Thermal-Aware Scheduler for DVFS-Enabled Single-Core Processors. In *29th Int. Conference on Real-Time Networks and Systems* (NANTES, France) *(RTNS'2021)*. ACM, NY, USA, 144–154. https://doi.org/10.1145/3453417.3453430
[15] J. P. Rodriguez and P. M. Yomsi. 2020. WiP: Towards a fine-grain thermal model for uniform multi-core processors. In *RTSS*. IEEE, Houston, TX, USA, 403–406.
[16] Andrea Rudi, Andrea Bartolini, Andrea Lodi, and Luca Benini. 2014. Optimum: Thermal-aware task allocation for heterogeneous many-core devices. In *International Conference on High Performance Computing and Simulation (HPCS)*. IEEE, Bologna, Italy, 82–87. https://doi.org/10.1109/HPCSim.2014.6903672
[17] Muhammad Naeem Shehzad, Qaisar Bashir, Umer Farooq, Ghufran Ahmed, Mohsin Raza, Priyan Malarvizhi Kumar, and Muhammad Khalid. 2020. Threshold temperature scaling: Heuristic to address temperature and power issues in MPSoCs. *Microprocessors and Microsystems* 77 (2020), 103124. https://doi.org/10.1016/j.micpro.2020.103124
[18] Hafiz Fahad Sheikh and Ishfaq Ahmad. 2014. Efficient heuristics for joint optimization of performance, energy, and temperature in allocating tasks to multi-core processors. In *International Green Computing Conference*. IEEE, Dallas, TX, USA, 1–8. https://doi.org/10.1109/IGCC.2014.7039178
[19] K. Skadron, M. R. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan. 2003. Temperature-Aware Microarchitecture. In *Int. Symp. on Comp. Arch.* ACM, New York, NY, USA, 2–13.
[20] Ting-Hao Tsai and Ya-Shu Chen. 2016. Thermal-throttling server: A thermal-aware real-time task scheduling framework for three-dimensional multicore chips. *Journal of Systems and Software* 112 (2016), 11–25. https://doi.org/10.1016/j.jss.2015.10.038
[21] Hai Wang, Ming Zhang, Sheldon X.-D. Tan, Chi Zhang, Yuan Yuan, Keheng Huang, and Zhenghong Zhang. 2016. New power budgeting and thermal management scheme for multi-core systems in dark silicon. In *29th International System-on-Chip Conference (SOCC)*. IEEE, Seattle, WA, USA, 344–349. https://doi.org/10.1109/SOCC.2016.7905507
[22] Guowei Wu, Zichuan Xu, Qiufen Xia, Jiankang Ren, and Feng Xia. 2010. Task Allocation and Migration Algorithm for Temperature-Constrained Real-Time Multi-Core Systems. In *Int. Conference on Green Computing and Communications and Int. Conference on Cyber, Physical and Social Computing*. IEEE/ACM, Hangzhou, China, 189–196. https://doi.org/10.1109/GreenCom-CPSCom.2010.27
[23] Buyoung Yun, Kang G. Shin, and Shige Wang. 2011. Thermal-Aware Scheduling of Critical Applications Using Job Migration and Power-Gating on Multi-core Chips. In *International Conference on Trust, Security and Privacy in Computing and Communications*. IEEE, Changsha, China, 1083–1090. https://doi.org/10.1109/TrustCom.2011.148
[24] Jinwei Zhang, Sheriff Sadiqbatcha, Yuanqi Gao, Michael O'Dea, Nanpeng Yu, and Sheldon X.-D. Tan. 2020. HAT-DRL: Hotspot-Aware Task Mapping for Lifetime Improvement of Multicore System using Deep Reinforcement Learning. In *2nd Workshop on Machine Learning for CAD (MLCAD)*. ACM/IEEE, New York, NY, USA, 77–82. https://doi.org/10.1145/3380446.3430623
[25] Dennis G Zill. 2012. *A first course in differential equations with modeling applications*. Cengage Learning, USA.