Thermal Safe Power (TSP): Efficient Power Budgeting for Heterogeneous Manycore Systems in Dark Silicon

Santiago Pagani, *Student Member, IEEE,* Heba Khdr, Jian-Jia Chen, *Member, IEEE,* Muhammad Shafique, *Member, IEEE,* Minming Li, *Member, IEEE,* and Jörg Henkel, *Fellow, IEEE*

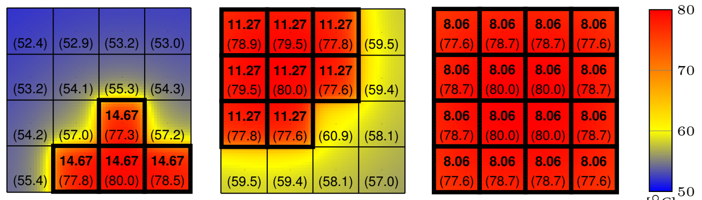
Abstract—Chip manufacturers provide the Thermal Design Power (TDP) for a specific chip. The cooling solution is designed to dissipate this power level. But because TDP is not necessarily the maximum power that can be applied, chips are operated with Dynamic Thermal Management (DTM) techniques. To avoid excessive triggers of DTM, usually, system designers also use TDP as power constraint. However, using a *single* and *constant* value as power constraint, e.g., TDP, can result in significant performance losses in homogeneous and heterogeneous manycore systems. Having better power budgeting techniques is a major step towards dealing with the dark silicon problem. This paper presents a new power budget concept, called Thermal Safe Power (TSP), which is an abstraction that provides safe power and power density constraints as a function of the number of simultaneously active cores. Executing cores at any power consumption below TSP ensures that DTM is not triggered. TSP can be computed offline for the worst cases, or online for a particular mapping of cores. TSP can also serve as a fundamental tool for guiding task partitioning and core mapping decisions, specially when core heterogeneity or timing guarantees are involved. Moreover, TSP results in dark silicon estimations which are less pessimistic than estimations using constant power budgets.

Index Terms—Thermal Safe Power (TSP), Thermal Design Power (TDP), Power Management, Dark Silicon, Heterogeneity

1 Introduction

For a specific chip, the common industry practice is to provide the system designers with the Thermal Design Power (TDP). According to [15], TDP "is the highest expected sustainable power while running known power intensive real applications" and it should be a safe power level in which to run the system. Hence, the cooling solution should be designed to dissipate TDP, such that running at this target power level does not cause any thermal problems. However, TDP is not the maximum achievable power. To avoid the system from possible overheating (and the associated reduction of reliability [1]), chips are provided with Dynamic Thermal Management (DTM) techniques. DTM can power-down cores, gate their clocks, reduce their supply voltage and frequency, boost-up the fan speed, etc. By directly measuring the temperature, if the system heats up above a certain threshold, DTM is triggered such that the temperature is reduced [15].

Usually, system designers also use TDP as a power constraint in order to avoid excessive triggers of DTM. However, on manycore systems, using a *single* and *constant* value as a power constraint for each core or for the entire chip (e.g., TDP), can result in significant performance losses, or in frequent triggers of DTM and thus a total performance much smaller than expected by the task partitioning and core mapping algorithms. Furthermore, given that the continuous increasing performance demands and power consumption issues have led to the emergence of heterogeneous architectures [29], power budgeting techniques should be developed to natively handle core heterogeneity. Moreover, different applications consume different amounts of power depending on the number of executing threads, selected frequency levels, and types of cores. This makes task partitioning and core mapping a very challenging process, specially when involving core heterogeneity or timing guarantees.



(a) 4 active cores (b) 8 active cores (c) 16 active cores

Fig. 1: Example for a maximum temperature of 80°C. Top numbers are the power consumptions (in Watts) of each active core (boxed in black). Bottom numbers in parenthesis are the temperatures in the center of each core (in °C). Detailed temperatures are shown according to the color bar.

Motivational Example: This example provides some insight on the drawbacks of using *single* and *constant* power constraints. For simplicity of presentation, consider a hypothetical manycore system with 16 homogeneous cores of size 2.31×2.31 mm (*simple in-order* Alpha 21264 cores in 45 nm, simulated with McPAT [20]), arranged in 4 rows and 4 columns. Assume a threshold temperature that triggers DTM of 80°C, and a cooling solution taken from the default configuration of HotSpot [14] (detailed in Section 7.1). To account for several *constant* power budgets, we consider four cases. Specifically, 58.7 W per-chip, 90.2 W per-chip, 129.0 W per-chip, and 8.06 W per-core.

Running simulations with HotSpot, Fig. 1a shows the resulting steady-state temperatures on the chip when 4 cores consume 14.67 W each (58.7 W in total). Similarly, Fig. 1b and Fig. 1c show the resulting steady-state temperatures on the chip when 8 cores consume 11.27 W each (90.2 W in total) and when 16 cores consume 8.06 W each (129.0 W in total), respectively. For all cases, although different power budgets are considered, at least one core reaches 80°C in the steady-state. Furthermore, Fig. 2 presents the maximum

This work was partly supported by the German Research Foundation (DFG) as part of the Transregional Collaborative Research Centre Invasive Computing *[SFB/TR 89] and* Resource-Constrained Data Analysis *[SFB 876].*

- *Santiago Pagani, Heba Khdr, Muhammad Shafique, and Jörg Henkel are with the Chair for Embedded Systems (CES), Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany. E-mail: pagani@kit.edu, heba.khdr@kit.edu, shafique@kit.edu, henkel@kit.edu*
- *Jian-Jia Chen is with the Department of Informatics, TU Dortmund, Dortmund, Germany. E-mail: jian-jia.chen@cs.uni-dortmund.de*
- *Minming Li is with the Department of Computer Science, City University of Hong Kong (CityU), Hong Kong, China. E-mail: minming.li@cityu.edu.hk*

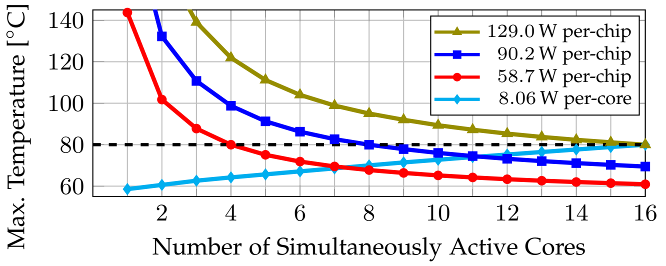


Fig. 2: Maximum steady-state temperature (DTM deactivated) among all cores as a function of the number of active cores, for different per-chip and per-core power budgets.

temperature among all cores (in the steady-state, with DTM is deactivated) as a function of the number of active cores.

From Fig. 1 and Fig. 2, it becomes clear that using a *single* and *constant* power budget can either be a pessimistic approach, or it can result in frequent triggers of DTM. For example, as seen in Fig. 1c, when activating 16 cores each core could safely consume up to 8.06 W (129.0 W in total), which means that using a smaller power budget is pessimistic for such a case. On the other hand, for a power budget of 129.0 W, if the system consumes 129.0 W among any number of cores smaller than 16 the temperature on at least one core will exceed 80°C, triggering DTM and thus resulting in a smaller performance than originally expected. Moreover, Fig. 2 shows that for this case using a power budget of 8.06 W per-core can be a good compromise, because the temperature never exceeds 80°C, but without being too far from it. However, there is still room for improvement, e.g., using different power budgets per-core depending on the number of active cores, such that the maximum steady-state temperature among all cores is 80°C for all cases. That is, according to Fig. 1, a per-core power budget of 14.67 W when activating 4 cores, 11.27 W when activating 8 cores, 8.06 W when activating 16 cores, etc. *Such is the power budget concept which is presented in this paper.*

Objective: The objective of this paper is to present a new power budget concept, called Thermal Safe Power (TSP). TSP is an abstraction that provides safe (but efficient) power constraint values as a function of the number of simultaneously active cores. Executing cores at power consumptions below TSP results in maximum temperatures below the threshold level that triggers DTM. Based on the RC thermal network [14] of a specific chip, we focus on deriving a formal method to compute TSP in an offline manner for the worst cases, as a function of the number of active cores. Moreover, the method should have polynomial-time complexity, such that TSP can also be computed online for a particular mapping of cores and ambient temperature. Thus, TSP can serve as a fundamental tool for guiding task partitioning, core mapping, and voltage/frequency selection algorithms in their attempt to achieve a high predictable performance under thermal constraints.

Our Contributions: Based on the above discussions, our main contributions are polynomial-time algorithms to compute TSP for both *homogeneous* and *heterogeneous* systems. For simplicity of presentation and introduction of the TSP concept, we first present the special case for handling homogeneous cores. We then present the algorithms for the general case, which are suitable for heterogeneous systems. Specifically, Section 5.1 and Section 6.1 present the algorithms that compute TSP for a particular mapping of active cores, which can be used online, thus accounting both for changes in the mapping decisions and ambient temperature. Section 5.2 and Section 6.2 present the computation of TSP for the worst-case mappings of active cores. That is, a mapping of active cores, for every possible number of simultaneously active cores, that results in the lowest TSP values. Such worst-case core mappings are the most pessimistic cases, which result in TSP values that are safe for any other mapping scenario. Therefore, this allows the system designers to abstract from core mapping decisions.

Open-Source Contributions: The algorithms to compute TSP are implemented as an open-source tool available for download at http://ces.itec.kit.edu/download.

Evaluations: We run simulations with gem5, McPAT, and HotSpot, to compare the total performance for using seven different power constraints: TSP for given mappings, worst-case TSP, three constant power budgets per-chip, a constant power budget per-core, and a boosting technique [4], [6], [16], [26]. We also show how to use TSP to estimate the amount of dark silicon [11], [28], resulting in estimations are less pessimistic than those for constant power budgets.

2 Related Work

There are many works that focus on improving performance under a given (constant) per-chip power budget [9], [24], [25], including several that specifically use TDP as power constraint [19], [21]. There is also work on reliability [18] under TDP, on power budgeting based on reinforcement learning [8], and on power budget matching with per-core power budget adaptation and thermal evaluations [5].

In [21], authors propose a control-based framework to obtain optimal trade-off between power and performance for homogeneous multicore systems under a TDP budget. The controllers throttle down the power if TDP is exceeded, and assign tasks to cores to optimize the performance. The work in [25] exploits process variations between cores in a homogeneous system to pick the more suitable cores for an application to improve its performance. Their results show that the performance efficiency can be increased along with the increase in the dark silicon area. The work in [19] presents throughput analysis for the impact of applying per-core power gating and Dynamic Voltage and Frequency Scaling (DVFS) to thermal and power constrained multicore processors, using TDP as power budget. The authors also exploit power and thermal headroom resulting from power-gated idle cores, allowing active cores to increase their frequency. The work in [9] presents a distributed agent-based power management approach that aims at balancing the power consumption on a heterogeneous multicore, under a fixed per-chip power budget. The agent negotiation is based on trading power units, compared to the classical supply/demand model of computational economics.

All of the above mentioned work uses a *single* and *constant* value as the power constraint to abstract from thermal problems. However, the motivational example in Section 1 shows that, depending on the amount of simultaneously active cores, different power consumptions result in the same maximum temperatures. This means that using a single value as the power constraint, e.g., TDP, can result in performance losses for multicore and manycore systems.

A few other technologies, like Intel's Turbo Boost [4], [6], [16], [26] and AMD's Turbo CORE [22], leverage this temperature headroom allowing power consumptions above the

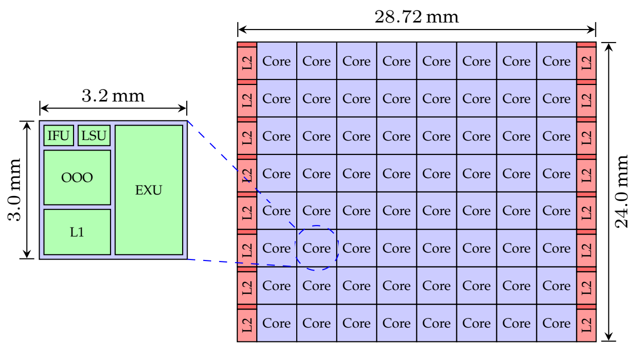


Fig. 3: Floorplan for a 64 core system based on simulations in McPAT [20]. In McPAT, cores are composed by several units: an instruction fetch unit (IFU), an execution unit (EXU), a load and store unit (LSU), an out-of-order (OOO) issue/dispatch, and a private L1 cache.

TDP constraint during *short time intervals* by increasing the voltage and frequency of the cores in an online manner. Due to the increases in power consumption, boosting normally results in an increment of the temperature through time. Once the highest temperature among all cores reaches a predefined threshold, the system must either return to nominal operation (needing some cool-down time before another boosting interval), or use some closed-loop control to oscillate around the threshold (prolonging the boosting time). Unlike these techniques, in TSP the increases in power consumption are constrained (according to the number of cores) such that DTM is not activated, allowing the system to remain *indefinitely* in such power states.

The introduction of TSP as a new power budget concept may motivate researchers to revisit the related work, possibly resulting in extensions of the existing literature, that achieve a better system performance.

3 System Model

This section reviews the system model. A table summarizing all the symbols in the paper is included in Appendix A.

3.1 Hardware Model

The computation of TSP is based on an RC thermal network modeled from the floorplan of any given architecture. The detail of the blocks that conform the floorplan can be freely chosen. For example, consider the 64 cores system presented in Fig. 3, which is based on simulations conducted with gem5 [3] and McPAT [20] for *out-of-order* (OOO) Alpha 21264 cores in 22 nm technology. Each core has an area of $9.6\,\text{mm}^2$, and there is a shared L2 cache and a memory controller every 4 cores. The area of the L2 blocks together with the memory controllers is $4.7\,\text{mm}^2$, which is comparable to the area of the cores. Therefore, it is reasonable to have independent blocks for the L2 cache and the memory controllers. For the rest of the chip, a practical approach is to consider blocks at a core level, and compute the TSP values for such a granularity, specifically, because the power consumptions of the internal blocks of the cores are tightly related with the execution frequency on each core.

For simplicity of presentation, throughout this paper we focus on a manycore system with M cores. However, we consider that there are Z blocks in the floorplan, such that $Z-M$ is the amount of blocks that correspond to other types of components, e.g., L2 caches and memory controllers. When computing TSP, we consider that each one of these $Z-M$ blocks always consumes its highest power.

We refer to a core being *active*, whenever the core is in its execution mode. To maintain a core active at its lowest speed a minimum power consumption is needed, which we define as $P^{\text{core}}_{\text{min}}$. Contrarily, a core is *inactive* when it is in some low-power mode, e.g., sleep or turned off (power-gated). We define the power consumption of an inactive core as $P^{\text{core}}_{\text{inact}}$, and it holds that $P^{\text{core}}_{\text{min}} \geq P^{\text{core}}_{\text{inact}}$. Furthermore, with respect to power consumption of active cores, it should be noted that cores consume different amounts of power depending on the type of core, the executing application, the current (data dependent) workload, the number of running threads, and the selected DVFS levels. As a general rule, when cores execute at high voltage and frequency levels they can achieve a high performance at the cost of high power consumptions, as shown in Fig. 18 and Fig. 19 in Appendix D. Along the paper we talk about *executing cores at power consumptions below TSP (or some other power budget)*. In practice, if cores are equipped with power meters, this can be simply achieved by measuring the actual power consumption on each core and acting accordingly, e.g., by applying DVFS such that the measured power is as close to the power budget as possible (for performance optimization) but without exceeding it. In case power meters are not available in the hardware, offline application profiling or runtime power estimation through performance counters (e.g., [17], [23], [30]) are a reasonable alternatives as an intermediate step to associate core settings to power consumption values. Finally, when simultaneously activating m cores, there are $\binom{M}{m}$ combinations of *which* specific cores in the floorplan to activate. Throughout the paper, we refer to a specific decision of *which* cores to activate as *core mapping* or *mapping of cores*.

Aside of having a power constraint as an abstraction from thermal problems, there can also exist a maximum chip power consumption that cannot be exceeded, which we denote P_{max}. This maximum power is not an abstraction, but an actual electrical constraint, e.g., from the power supply.

The power consumption in a CMOS core is the summation of its dynamic power consumption (mainly generated by switching activities) and its static power consumption (mainly generated by leakage currents). Part of the static power consumption depends on the temperature of the core, which means that higher temperatures cause higher power consumptions. In order to consider *safe margins for offline power profiles* of tasks and their scheduling decisions for safe operation, we assume that the power consumptions of cores are modeled at temperatures near the threshold level that triggers DTM, which we define as T_{DTM}. That is, when experimentally measuring the power consumption of a core executing a task at a specific voltage and frequency, this is done at temperatures near T_{DTM}. If not, we may have underestimated power models of tasks. When making online decisions, if cores are equipped with power meters, then no over- or underestimation occurs, because the system can measure the actual power consumption of each task (including leakage effects), and act accordingly.

3.2 Thermal Model

For our thermal model, we consider the well-known duality between thermal and electrical circuits, i.e., we use an RC thermal network [14]. Such a network is composed by N thermal nodes, where $N \geq Z$. In an RC thermal network, thermal nodes are interconnected between each other through thermal conductances. Each thermal node also has a thermal capacitance associated to it, which accounts for the transient temperatures. The ambient temperature, denoted as T_{amb}, is considered to be constant, and thus there is no capacitance associated with it. The power consumptions of cores and other blocks correspond to heat sources. With these considerations, the temperature of every thermal node is a function of its power consumption, the temperatures of the neighboring nodes, and the ambient temperature. Hence, for any RC thermal network with N thermal nodes, we can build a system of N differential equations associated with it, which can be expressed as

$$\mathbf{A}\mathbf{T}' + \mathbf{B}\mathbf{T} = \mathbf{P} + T_{\text{amb}}\mathbf{G},$$

where matrix $\mathbf{A} = [a_{i,j}]_{N\times N}$ contains the thermal capacitance values, matrix $\mathbf{B} = [b_{i,j}]_{N\times N}$ contains the thermal conductance values in $\left[\frac{\text{Watt}}{\text{Kelvin}}\right]$, column vector $\mathbf{T} = [T_i]_{N\times 1}$ represents the temperature on each node, column vector $\mathbf{T'} = [T'_i]_{N\times 1}$ accounts for the first order derivative of the temperature on each node with respect to time, column vector $\mathbf{P} = [p_i]_{N\times 1}$ contains the power consumption on each node, and column vector $\mathbf{G} = [g_i]_{N\times 1}$ contains the thermal conductance between each node and the ambient. If node i is not in contact with the ambient temperature, e.g., the temperature of a core or an internal node, the value of g_i is set to zero. The thermal conductance values in matrix $\mathbf{B}$ include the thermal conductances between vertical and lateral neighboring nodes. Matrix $\mathbf{A}$ is generally a diagonal matrix, since thermal capacitances are modeled to ground.

For specific floorplans, the values for matrices and vector $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{G}$ can be computed at design-time through HotSpot [14], and these are the same matrices used internally by HotSpot. A more practical alternative is to use some thermal modeling method that relies on real measurements on a specific chip and cooling solution, e.g., [10]. Such an option is well suited to deal with process variability and heat sink imperfections. Similarly, since having different fan speeds changes the thermal conductivity of the heat sink, a method like the one in [10] can also help to adapt the thermal model at runtime, or to consider multiple thermal models from which to choose (one for each fan speed).

When only considering the *steady-state*, we have that

$$\mathbf{B}\mathbf{T} = \mathbf{P} + T_{\text{amb}}\mathbf{G} \qquad \text{or} \qquad \mathbf{T} = \mathbf{B}^{-1}\mathbf{P} + T_{\text{amb}}\mathbf{B}^{-1}\mathbf{G},$$

where from $\mathbf{B}^{-1}$ we have that ${b^{-1}}_{i,j} \cdot p_j$ represents the amount of heat contributed by node j into the steady-state temperature of node i, i.e., T_i.

Regarding vector $\mathbf{P}$, we can divide it into three sub-vectors: $\mathbf{P}^{\text{cores}}$ for the power consumption on the cores; $\mathbf{P}^{\text{blocks}}$ for the power consumption on blocks of a different type, which we consider to be always active at their highest power consumption values, e.g., the L2 caches for the manycore system in Fig. 3 as explained in Section 3.1; and $\mathbf{P}^{\text{int}}$ for internal nodes in which $p_i^{\text{int}} = 0$ for all i. It holds that $\mathbf{P} = \mathbf{P}^{\text{cores}} + \mathbf{P}^{\text{blocks}} + \mathbf{P}^{\text{int}}$. Decomposing $\mathbf{P}$, we have that

$$\mathbf{T} = \mathbf{B}^{-1}\mathbf{P}^{\text{cores}} + \mathbf{B}^{-1}\mathbf{P}^{\text{blocks}} + T_{\text{amb}}\mathbf{B}^{-1}\mathbf{G}, \tag{1}$$

where by only focusing on one equation from the system of equations, the steady-state temperature on node i is

$$T_i = \sum_{j=1}^{N} {b^{-1}}_{i,j} \cdot p_j^{\text{cores}} + \sum_{j=1}^{N} {b^{-1}}_{i,j} \left(p_j^{\text{blocks}} + T_{\text{amb}} \cdot g_j\right). \tag{2}$$

For notational brevity, we define set $\mathbf{L} = \{\ell_1, \ell_2, \ldots, \ell_Z\}$, such that the elements in $\mathbf{L}$ include all indexes of the thermal nodes that correspond to blocks of the floorplan; as opposed to thermal nodes that represent the heat sink, internal nodes of the heat spreader, the thermal interface material, etc. Similarly, we define set $\mathbf{K} = \{k_1, k_2, \ldots, k_M\}$, such that $\mathbf{K}$ contains all indexes of nodes that correspond to cores.

Furthermore, we also define column vector $\mathbf{Q} = [q_i]_{M\times 1}$ for a particular mapping of *active* cores. Vector $\mathbf{Q}$ is a binary vector: $q_i = 1$ means that core i corresponds to an *active* core; $q_i = 0$ means that core i corresponds to an *inactive* core.

4 Problem Definition

This paper presents a new power budget concept, called Thermal Safe Power (TSP). TSP is an abstraction that provides power constraint values as a function of the number of simultaneously active cores, according to the definition of *active*/*inactive* from Section 3.1. The values of TSP vary according to the floorplan and which cores are simultaneously active. Some specific core mappings result in the lowest TSP values, and we define such core mappings as the *worst-case mappings*. Executing cores at power consumptions below TSP, for the corresponding mapping and number of active cores, results in maximum temperatures below T_{DTM}.

For a specific chip and its corresponding RC thermal network, the *first objective* of this paper is to provide a numerical method to compute TSP for a given mapping of cores. The mapping of cores is typically determined by an operating/run-time system or by an offline system software. This method should have polynomial-time complexity, such that TSP can be computed online for a particular mapping of cores and ambient temperatures. Formally, for a given core mapping $\mathbf{Q}$, this means obtaining a uniform power constraint for the active cores, defined as $P_{\text{TSP}}(\mathbf{Q})$, such that $T_i \leq T_{\text{DTM}}$ for all $i \in \mathbf{L}$.

The *second objective* is to derive an algorithm to compute the most pessimistic TSP values for a given number of simultaneously active cores, i.e., for the worst-case mappings. Such TSP values can be used as safe power constraints for any possible mapping of cores, thus allowing the system designers to abstract from core mapping decisions. Formally, this means obtaining the most pessimistic uniform power constraint for any m active cores, defined as $P_{\text{TSP}}^{\text{worst}}(m)$, such that $T_i \leq T_{\text{DTM}}$ for all $i \in \mathbf{L}$.

The algorithms presented in Section 5 and Section 6 are derived considering the steady-state. Appendix C explains a method to further consider the transient temperatures.

5 TSP for Homogeneous Multicores

5.1 TSP for a Given Core Mapping (Homogeneous)

This subsection presents a polynomial-time algorithm to compute TSP in an online manner for a particular core mapping and ambient temperature, which results in a uniform value of TSP per-core, for all active cores in $\mathbf{Q}$. That is, one power constraint value for each active core in the specified mapping, that results in a maximum temperature (in the steady-state) among all cores which does not exceed T_{DTM}. We define such a power constraint as $P_{\text{TSP}}(\mathbf{Q})$. *Note that this does not mean that all active cores consume the same power, as this would be an unrealistic assumption. In fact, this means that each active core can consume any amount of power, which can be different for each core, as long as the power consumption of each core is no more than* $P_{\text{TSP}}(\mathbf{Q})$. For completeness, a solution to compute TSP that constrains cores to different power values, depending on their location and the adjacent active cores, is presented in Appendix B.

Here, we summarize the derivation procedure and the properties of the lemmas and theorems in this subsection.

The algorithm that computes $P_{\text{TSP}}(\mathbf{Q})$ is presented in Theorem 1, which is based on Lemma 1. Lemma 1 derives a uniform power constraint for all active cores in mapping $\mathbf{Q}$ such that the maximum temperature in the steady-state among all blocks does not exceed T_{DTM}, by taking into account the floorplan, the power consumption on other blocks, the ambient temperature, and the power consumption of inactive cores. This power constraint is defined as $P^{\star}_{\text{TSP}}(\mathbf{Q})$ and it does not take into consideration the maximum chip power $P_{\max}$. Theorem 1 verifies that the maximum power $P_{\max}$ is not violated. We define auxiliary function $\mathrm{R}(m)$ as

$$\mathrm{R}(m) = P^{\text{core}}_{\text{inact}} + \frac{P_{\max} - \sum_{i=1}^{N} p_i^{\text{blocks}} - P^{\text{core}}_{\text{inact}} \cdot M}{m}, \qquad (3)$$

where m represents the number of active cores. If consuming $P^{\star}_{\text{TSP}}(\mathbf{Q})$ in all active cores violates $P_{\max}$, then Theorem 1 sets $P_{\text{TSP}}(\mathbf{Q})$ to a smaller value, specifically, to $\mathrm{R}\left(\sum_{i=1}^{M} q_i\right)$, such that $P_{\max}$ is satisfied. Moreover, the pseudo-code for computing $P_{\text{TSP}}(\mathbf{Q})$ is presented in Algorithm 1.

Lemma 1. For a given $\mathbf{Q}$, $\mathbf{P}^{\text{blocks}}$, T_{amb}, T_{DTM}, $P^{\text{core}}_{\text{inact}}$, and floorplan, the value of power constraint $P^{\star}_{\text{TSP}}(\mathbf{Q})$ for each active core in the given mapping is computed as

$$P^{\star}_{\text{TSP}}(\mathbf{Q}) = \min_{\forall i \in \mathbf{L}} \left\{ \frac{T_{\text{DTM}} - P^{\text{core}}_{\text{inact}} \cdot \sum_{j=1}^{M} b^{-1}{}_{i,k_j} (1-q_j)}{\sum_{j=1}^{M} b^{-1}{}_{i,k_j} \cdot q_j} + \frac{-\sum_{j=1}^{N} b^{-1}{}_{i,j} \left(p_j^{\text{blocks}} + T_{\text{amb}} \cdot g_j\right)}{\sum_{j=1}^{M} b^{-1}{}_{i,k_j} \cdot q_j} \right\}. \qquad (4)$$

Proof: Considering vector $\mathbf{Q}$, and set $\mathbf{K}$, if all inactive cores consume $P^{\text{core}}_{\text{inact}}$ and all active cores consume equal power P_{equal} at a given time, we have that for this time instant, Equation (2) can be rewritten as

$$T_i = P_{\text{equal}} \cdot \sum_{j=1}^{M} b^{-1}{}_{i,k_j} \cdot q_j + P^{\text{core}}_{\text{inact}} \cdot \sum_{j=1}^{M} b^{-1}{}_{i,k_j} (1-q_j) + \sum_{j=1}^{N} b^{-1}{}_{i,j} \left(p_j^{\text{blocks}} + T_{\text{amb}} \cdot g_j\right). \qquad (5)$$

For a given $\mathbf{Q}$, $\mathbf{P}^{\text{blocks}}$, T_{amb}, and floorplan, the only variables in Equation (5) are T_i and P_{equal}. By setting T_i to T_{DTM}, we can then compute, for every i that belongs in $\mathbf{L}$, what value of P_{equal} would make T_i reach T_{DTM}. The most pessimistic value of P_{equal} is a safe power constraint for all cores in mapping $\mathbf{Q}$, and is therefore the resulting $P^{\star}_{\text{TSP}}(\mathbf{Q})$ for the given $\mathbf{P}^{\text{blocks}}$, T_{amb}, and floorplan, which is expressed in Equation (4). The value of i that results in $P^{\star}_{\text{TSP}}(\mathbf{Q})$ corresponds to the block with the highest temperature for such a case. Thus, the lemma is proven. □

Theorem 1. For a given $\mathbf{Q}$, $\mathbf{P}^{\text{blocks}}$, T_{amb}, T_{DTM}, $P^{\text{core}}_{\text{inact}}$, $P_{\max}$, and floorplan, the value of power constraint $P_{\text{TSP}}(\mathbf{Q})$ for each active core in the specified mapping is computed as

$$P_{\text{TSP}}(\mathbf{Q}) = \begin{cases} P^{\star}_{\text{TSP}}(\mathbf{Q}) & \text{if } P^{\star}_{\text{TSP}}(\mathbf{Q}) \le \mathrm{R}\left(\sum_{i=1}^{M} q_i\right) \\ \mathrm{R}\left(\sum_{i=1}^{M} q_i\right) & \text{otherwise.} \end{cases} \qquad (6)$$

Proof: The first case is based on Lemma 1. As for the second case, the summation of the elements in $\mathbf{Q}$ is equal to the number of active cores in the mapping. Thus, it should hold that $P_{\text{TSP}}(\mathbf{Q}) \cdot \sum_{i=1}^{M} q_i + P^{\text{core}}_{\text{inact}} \left(M - \sum_{i=1}^{M} q_i\right) + \sum_{i=1}^{N} p_i^{\text{blocks}} \le P_{\max}$. If $P^{\star}_{\text{TSP}}(\mathbf{Q})$ is larger than $\mathrm{R}\left(\sum_{i=1}^{M} q_i\right)$, then we simply set $P_{\text{TSP}}(\mathbf{Q})$ to $\mathrm{R}\left(\sum_{i=1}^{M} q_i\right)$, such that $P_{\max}$ is not violated. Thus, the theorem is proven. □

Algorithm 1 TSP for a given mapping

```
Input: $\mathbf{Q}$, $\mathbf{P}^{\text{blocks}}$, $T_{\text{amb}}$, $T_{\text{DTM}}$, $P^{\text{core}}_{\text{inact}}$, $P_{\max}$, and floorplan;
Output: Uniform TSP power constraint for mapping $\mathbf{Q}$;
    {First compute $P^{\star}_{\text{TSP}}(\mathbf{Q})$ according to Equation (4)}
 1: $P^{\star}_{\text{TSP}}(\mathbf{Q}) \leftarrow \infty$;
 2: for all $i \in \mathbf{L}$ do
 3:     $auxP \leftarrow T_{\text{DTM}} - P^{\text{core}}_{\text{inact}} \cdot \sum_{j=1}^{M} b^{-1}{}_{i,k_j} (1 - q_j)$;
 4:     $auxP \leftarrow auxP - \sum_{j=1}^{N} b^{-1}{}_{i,j} \left(p_j^{\text{blocks}} + T_{\text{amb}} \cdot g_j\right)$;
 5:     $auxP \leftarrow \frac{auxP}{\sum_{j=1}^{M} b^{-1}{}_{i,k_j} \cdot q_j}$;
 6:     if $auxP < P^{\star}_{\text{TSP}}(\mathbf{Q})$ then
 7:         $P^{\star}_{\text{TSP}}(\mathbf{Q}) \leftarrow auxP$;
 8:     end if
 9: end for
    {Then compute $P_{\text{TSP}}(\mathbf{Q})$ according to Equation (6)}
10: if $P^{\star}_{\text{TSP}}(\mathbf{Q}) \le \mathrm{R}\left(\sum_{i=1}^{M} q_i\right)$ then
11:     $P_{\text{TSP}}(\mathbf{Q}) \leftarrow P^{\star}_{\text{TSP}}(\mathbf{Q})$;
12: else
13:     $P_{\text{TSP}}(\mathbf{Q}) \leftarrow \mathrm{R}\left(\sum_{i=1}^{M} q_i\right)$;
14: end if
15: return $P_{\text{TSP}}(\mathbf{Q})$;
```

The total time complexity for computing $P_{\text{TSP}}(\mathbf{Q})$ for a given $\mathbf{P}^{\text{blocks}}$, T_{amb}, T_{DTM}, $P_{\max}$, and floorplan, is $O(ZN)$.

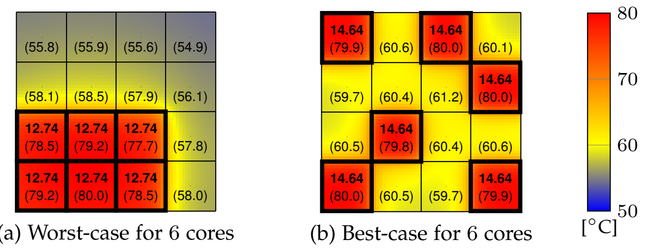
5.2 TSP for Worst-Case Mappings (Homogeneous)

This subsection presents a polynomial-time algorithm to compute TSP for the worst-case core mappings, for m active cores. The algorithm results in a uniform value of TSP per-core for all active cores. That is, one power constraint value for each active core in *any* possible core mapping with m simultaneously active cores, that results in a maximum temperature (in the steady-state) among all cores which does not exceed T_{DTM}. We define such a power constraint as $P^{\text{worst}}_{\text{TSP}}(m)$. *As in Section 5.1, note that this does not mean that all active cores consume the same power, as this would be an unrealistic assumption. It means that each active core,* for any given mapping of m active cores, *can consume any amount of power, which can be different for each core, as long as the power consumption of each core is no more than $P^{\text{worst}}_{\text{TSP}}(m)$.* The purpose for doing this is to allow system designers to abstract themselves from mapping decisions, as opposed to the TSP computations from Section 5.1.

Due to the heat transfer among cores, the mapping of cores, i.e., *which* cores are active and *which* cores are inactive, plays a major role in the computation of the maximum temperatures. This can be seen in the following example. Consider a manycore system of 16 cores with the same settings as in the motivational example from Section 1. Fig. 4 shows two possible mappings when simultaneously activating 6 cores. For Fig. 4a, the maximum temperature among all cores reaches 80°C when each core consumes 12.74 W. However for Fig. 4b, this happens when each core consumes 14.64 W. We have a total of 11.4 W difference between these two core mappings, but with the same maximum temperature.

According to Equation (5), if all the active cores in the system run at the same power consumption, one or more of the active cores will heat up the most with respect to the rest of the cores. For the same value of P_{equal}, different $\mathbf{Q}$ mappings result in different maximum temperatures. Dually, for the same maximum temperature among all cores, different mappings will result in different power consumption values to produce such a temperature.

Generally, as shown in Fig. 4, for the same maximum temperature values with $p_i^{\text{blocks}} = 0$ for all i, activating cores together in a corner of the chip results in lower P_{equal} values, compared to dispersing them throughout the chip. *This*



(a) Worst-case for 6 cores (b) Best-case for 6 cores

Fig. 4: Example of worst-case and best-case mappings. Top numbers are the power consumptions (in Watts) of each active core (boxed in black). Bottom numbers in parenthesis are the temperatures in the center of each core (in °C). Detailed temperatures are shown according to the color bar.

happens because active *cores have higher chances of transferring heat to* inactive *cores when they are dispersed*. The worst-case mappings for TSP are those that produce the lowest power constraints, while no block in the floorplan exceeds (in the steady-state) the threshold temperature that triggers DTM, i.e., $T_i \leq T_{\text{DTM}}$ for all $i \in \mathbf{L}$.

By computing TSP for such cases, system designers can abstract themselves from core mapping decisions. This happens because such worst-case core mappings are the most pessimistic cases. When having m active cores, executing cores at power consumptions below $P_{\text{TSP}}^{\text{worst}}(m)$ will result in maximum temperatures (in the steady-state), among all blocks in the floorplan, below the threshold level that triggers DTM, for any possible mapping of m active cores.

Here, we summarize the derivation procedure and the properties of the lemmas and theorems in this subsection. For a given m, our algorithm is presented in Theorem 2, which is based on auxiliary matrix $\mathbf{H}$ and Lemmas 2, 3, and 4. Lemma 4 derives the worst-case uniform power constraint for m active cores such that the maximum temperature in the steady-state among all blocks does not exceed T_{DTM}, by taking into account the floorplan, the power consumption on other blocks, the ambient temperature, and the power consumption of inactive cores. This power constraint is defined as $P_{\text{TSP}}^{\star\text{worst}}(m)$ and it does not take into consideration the maximum power $P_{\max}$. Theorem 2 verifies that the maximum power $P_{\max}$ is not violated. If consuming $P_{\text{TSP}}^{\star\text{worst}}(m)$ in m cores violates $P_{\max}$, then we set $P_{\text{TSP}}^{\text{worst}}(m)$ to a smaller value such that $P_{\max}$ is satisfied. Auxiliary matrix $\mathbf{H} = [h_{i,j}]_{Z \times M}$ is used in Lemma 4 to compute the maximum amount of heat that any m cores can contribute to the temperature on node i. Matrix $\mathbf{H}$ is built by making a partial copy of matrix $\mathbf{B}^{-1}$, such that $h_{i,j} = b^{-1}{}_{\ell_i,k_j}$ for all $i = 1, 2, \ldots, Z$ and for all $j = 1, 2, \ldots, M$, and then reordering each row of $\mathbf{H}$ in a decreasing manner. Matrix $\mathbf{H}$ needs to be built and ordered only one time, which has a time complexity $O(ZM \log M)$ (the pseudo-code for building matrix $\mathbf{H}$ is later included inside Algorithm 2). Lemmas 2 and 3 prove that using matrix $\mathbf{H} = [h_{i,j}]_{Z \times M}$ results in the most pessimistic power constraints.

Lemma 2. If all active cores consume equal power P_{equal}, the m highest values for $P_{\text{equal}} \cdot b^{-1}{}_{i,j}$ such that $j \in \mathbf{K}$, correspond to the amount of heat contributed to temperature T_i by the m cores that contribute more heat into T_i.

Proof: From the definition of matrix $\mathbf{B}^{-1}$, we know that $b^{-1}{}_{i,j} \cdot p_j$ represents the heat contributed by node j into the steady-state temperature of node i, i.e., T_i. Particularly, if we focus on $j \in \mathbf{K}$, we are referring to the heat contributed by each core j into the steady-state temperature T_i.

From Equation (5), we know that there is one or more mappings $\mathbf{Q}$ that result in the maximum T_i for a given P_{equal}. Given that $\mathbf{B}^{-1}$, $\mathbf{G}$, $\mathbf{P}^{\text{blocks}}$, and T_{amb} are constant, it is clear that, for a given P_{equal}, T_i is maximized when $\sum_{j=1}^{M} b^{-1}{}_{i,k_j} \cdot q_j$ is also maximized. Moreover, it holds that $P_{\text{equal}} \geq P_{\text{min}}^{\text{core}} \geq P_{\text{inact}}^{\text{core}}$, and the summation of the elements in $\mathbf{Q}$ is equal to the number of active cores in the mapping, i.e., $\sum_{j=1}^{M} q_j = m$. Hence, for row i, we have that $\sum_{j=1}^{M} b^{-1}{}_{i,k_j} \cdot q_j$ is maximized when mapping $\mathbf{Q}$ activates the m cores with the highest $b^{-1}{}_{i,k_j}$, for row i. Thus, the lemma is proven. □

Lemma 3. If all active cores consume equal power P_{equal}, multiplying the power consumption on each core with the summation of the first m elements in row i of auxiliary matrix $\mathbf{H}$, that is, computing $P_{\text{equal}} \cdot \sum_{j=1}^{m} h_{i,j}$, results in the maximum amount of heat that any m cores can contribute to the steady-state temperature on node ℓ_i, that is, T_{ℓ_i}.

Proof: Based on the definition of auxiliary matrix $\mathbf{H}$ and Lemma 2, the first m elements in row i of matrix $\mathbf{H}$ correspond to the m highest $b^{-1}{}_{\ell_i,j}$ values, for node ℓ_i, such that $j \in \mathbf{K}$. In turn, multiplied by P_{equal}, this is the amount of heat contributed to temperature T_{ℓ_i} by the m cores that contribute more heat into T_{ℓ_i}. Thus, the lemma is proven. □

Lemma 4. For a given $\mathbf{P}^{\text{blocks}}$, T_{amb}, T_{DTM}, $P_{\text{inact}}^{\text{core}}$, and floorplan, the value of power constraint $P_{\text{TSP}}^{\star\text{worst}}(m)$ for each active core in *any* possible core mapping with m simultaneously active cores is computed as

$$P_{\text{TSP}}^{\star\text{worst}}(m) = \min_{1 \leq i \leq Z} \left\{ \frac{T_{\text{DTM}} - P_{\text{inact}}^{\text{core}} \cdot \sum_{j=m+1}^{M} h_{i,j}}{\sum_{j=1}^{m} h_{i,j}} + \frac{- \sum_{j=1}^{N} b^{-1}{}_{\ell_i,j} \left(p_j^{\text{blocks}} + T_{\text{amb}} \cdot g_j \right)}{\sum_{j=1}^{m} h_{i,j}} \right\}. \tag{7}$$

Proof: Considering matrix $\mathbf{H}$ and Lemma 3, Equation (5) becomes

$$T_{\ell_i} \leq P_{\text{equal}} \cdot \sum_{j=1}^{m} h_{i,j} + P_{\text{inact}}^{\text{core}} \cdot \sum_{j=m+1}^{M} h_{i,j} + \sum_{j=1}^{N} b^{-1}{}_{\ell_i,j} \left(p_j^{\text{blocks}} + T_{\text{amb}} \cdot g_j \right).$$

Similar to Lemma 1, by setting T_{ℓ_i} to T_{DTM}, we can compute, for every $i = 1, 2, \ldots, Z$, what value of P_{equal} would make T_{ℓ_i} reach T_{DTM}. The most pessimistic value of P_{equal} is a safe power constraint for the m active cores, and is therefore the resulting $P_{\text{TSP}}^{\star\text{worst}}(m)$ for the given $\mathbf{P}^{\text{blocks}}$, T_{amb}, and floorplan, which is expressed in Equation (7). The value of i that results in $P_{\text{TSP}}^{\star\text{worst}}(m)$ corresponds to the block with the highest temperature in the worst-case mapping. Thus, the lemma is proven. □

Theorem 2. For a given $\mathbf{P}^{\text{blocks}}$, T_{amb}, T_{DTM}, $P_{\text{inact}}^{\text{core}}$, $P_{\max}$, and floorplan, the value of power constraint $P_{\text{TSP}}^{\text{worst}}(m)$ for each active core in *any* possible core mapping with m simultaneously active cores is computed as

$$P_{\text{TSP}}^{\text{worst}}(m) = \begin{cases} P_{\text{TSP}}^{\star\text{worst}}(m) & \text{if } P_{\text{TSP}}^{\star\text{worst}}(m) \leq \mathrm{R}(m) \\ \mathrm{R}(m) & \text{otherwise.} \end{cases} \tag{8}$$

Proof: The first case is based on Lemma 4, and the second case is based on the proof of Theorem 1. □

Given that matrix $\mathbf{H}$ only needs to be build once for a given chip, the total time complexity for computing $P_{\text{TSP}}^{\text{worst}}(m)$ for a given m is $O(ZN)$, and for computing $P_{\text{TSP}}^{\text{worst}}(m)$ for all $m = 1, 2, \ldots, M$ is $O(MZN)$. The corresponding pseudo-code is presented in Algorithm 2.

Algorithm 2 TSP for all worst-case mappings

```
Input: $\mathbf{P}^{\text{blocks}}$, $T_{\text{amb}}$, $T_{\text{DTM}}$, $P^{\text{core}}_{\text{inact}}$, $P_{\text{max}}$, and floorplan;
Output: Uniform TSP for all worst-case mappings;
    {First build auxiliary matrix $\mathbf{H} = [h_{i,j}]_{Z \times M}$}
 1: for $i = 1$ to $Z$ do
 2:     for $j = 1$ to $M$ do
 3:         $h_{i,j} \leftarrow b^{-1}{}_{\ell_i,k_j}$;
 4:     end for
 5:     Re-order row $i$ of matrix $\mathbf{H}$ in a decreasing manner;
 6: end for
 7: for $m = 1$ to $M$ do
        {Then compute $P^{\star\text{worst}}_{\text{TSP}}(m)$ from Equation (7)}
 8:     $P^{\star\text{worst}}_{\text{TSP}}(m) \leftarrow \infty$;
 9:     for $i = 1$ to $Z$ do
10:         $auxP \leftarrow T_{\text{DTM}} - P^{\text{core}}_{\text{inact}} \cdot \sum_{j=m+1}^{M} h_{i,j}$;
11:         $auxP \leftarrow auxP - \sum_{j=1}^{N} b^{-1}{}_{\ell_i,j} \left(p^{\text{blocks}}_j + T_{\text{amb}} \cdot g_j\right)$;
12:         $auxP \leftarrow \frac{auxP}{\sum_{j=1}^{m} h_{i,j}}$;
13:         if $auxP < P^{\star\text{worst}}_{\text{TSP}}(m)$ then
14:             $P^{\star\text{worst}}_{\text{TSP}}(m) \leftarrow auxP$;
15:         end if
16:     end for
        {Finally compute $P^{\text{worst}}_{\text{TSP}}(m)$ according to Equation (8)}
17:     if $P^{\star\text{worst}}_{\text{TSP}}(m) \leq \text{R}(m)$ then
18:         $P^{\text{worst}}_{\text{TSP}}(m) \leftarrow P^{\star\text{worst}}_{\text{TSP}}(m)$;
19:     else
20:         $P^{\text{worst}}_{\text{TSP}}(m) \leftarrow \text{R}(m)$;
21:     end if
22: end for
23: return $P^{\text{worst}}_{\text{TSP}}(m)$ for all $m = 1, 2, \ldots, M$;
```

6 TSP FOR HETEROGENEOUS MULTICORES

Given that in heterogeneous systems cores have different areas and consume different amounts of power, an efficient power budgeting technique would not use the same power constraint for different types of cores. The two algorithms in Section 5 provide the foundations of TSP for homogeneous cores. In this section, we explain how to use these concepts for the general case of heterogeneous systems.

6.1 TSP for a Given Core Mapping (Heterogeneous)

In this subsection we revise Algorithm 1 (Section 5.1) in order to consider heterogeneous systems. An example of such a heterogeneous system is shown in Fig. 5, consisting of 24 *out-of-order* (OOO) Alpha 21264 cores and 16 *simple in-order* Alpha 21264 cores, based on simulations conducted on gem5 [3] and McPAT [20] for 22 nm, and 16 *in-order* Cortex-A7 cores and 16 OOO Cortex-A15 cores, based on an Odroid-XU3 [13] mobile platform with an Exynos 5 Octa (5422) [27] chip with ARM's "big.LITTLE" architecture. It is important to note that the temperature on a chip is directly related to the *power density*, not to the power consumption. That is, for two cores with different areas, if both cores consume the same power, the core with smaller area will have a higher temperature than the other core. Therefore, we should handle core heterogeneity by focusing on power density instead of power consumption, thus deriving a power density constraint. In order to better understand this concept, we present Fig. 6a, which divides the cores of the floorplan presented in Fig. 5 (after rounding their areas to multiples of $0.25\,\text{mm}^2$ for presentation purposes) into smaller sub-blocks of $0.25\,\text{mm}^2$. Assume now that we activate some cores of different types as shown in Fig. 6b (active cores boxed in thick black lines), such that the power density on all active cores is $1.06\,\text{W/mm}^2$, which is equivalent to having each $0.25\,\text{mm}^2$ sub-block consume $0.265\,\text{W}$. Fig. 6b shows that for such a case the maximum temperature among all cores does not exceed T_{DTM} (in this example 80°C). Conceptually, this is also equivalent to computing TSP through Algorithm 1 for the smaller homogeneous sub-blocks. However, using Algorithm 1 to derive the power

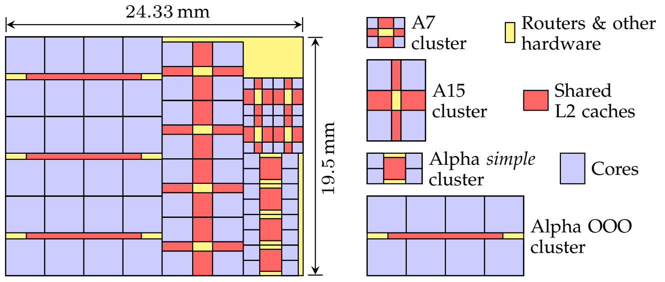
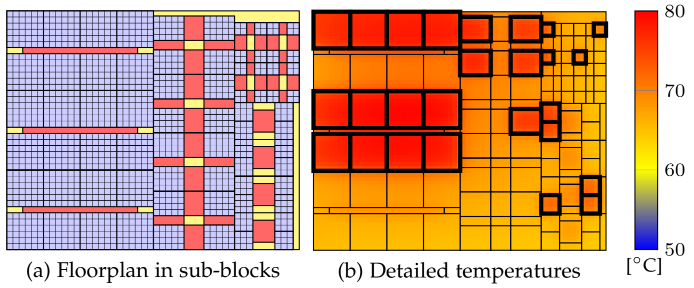


Fig. 5: Floorplan of a heterogeneous 72 core system. For a given cluster type, each cluster is identified as a, b, c, and d, from left to right and from top to bottom.



(a) Floorplan in sub-blocks (b) Detailed temperatures

Fig. 6: Example of power density constraint for given mappings. In (a), the cores are divided in homogeneous sub-blocks. In (b), active cores are boxed in thick black and detailed temperatures are shown according to the color bar.

density constraint is not efficient, specially considering the required size of the sub-blocks (and corresponding RC thermal network) to perfectly fit in all types of cores.

Namely, we derive a uniform *power density* constraint for each active core in the specified mapping (independent of the type of core) that results in a maximum temperature (in the steady-state) among all cores which does not exceed T_{DTM}. We define such a power density constraint as $P^{\rho}_{\text{TSP}}(\mathbf{Q})$. Moreover, the area of core j is defined as $\text{area}^{\text{core}}_j$, for $j = 1, 2, \ldots, M$. To compute the corresponding TSP value for core j, we simply multiply the power density constraint $P^{\rho}_{\text{TSP}}(\mathbf{Q})$ with the area of core j. Note that for heterogeneous systems, the power consumption of inactive cores varies among different types of cores. Hence, we define the power consumption of core j when the core is inactive as $p^{\text{core}}_{\text{inact}\,j}$, for $j = 1, 2, \ldots, M$.

The derivation procedure and properties in this subsection are very similar to those in Section 5.1. Namely, the algorithm that computes $P^{\rho}_{\text{TSP}}(\mathbf{Q})$ is presented in Theorem 3, which is based on Lemma 5. Lemma 5 ignores the maximum chip power and derives a uniform power density constraint for all active cores in mapping $\mathbf{Q}$, defined as $P^{\rho\star}_{\text{TSP}}(\mathbf{Q})$ and independent of the types of cores. Theorem 3 verifies that the maximum power P_{max} is satisfied. We define auxiliary function $\text{R}^{\rho}(\mathbf{Q})$ as

$$\text{R}^{\rho}(\mathbf{Q}) = \frac{P_{\text{max}} - \sum_{i=1}^{N} p^{\text{blocks}}_i - \sum_{j=1}^{M} p^{\text{core}}_{\text{inact}\,j}\,(1-q_j)}{\sum_{j=1}^{M} \text{area}^{\text{core}}_j \cdot q_j}. \quad (9)$$

If having power density $P^{\rho\star}_{\text{TSP}}(\mathbf{Q})$ in all active cores violates P_{max}, then Theorem 3 sets $P^{\rho}_{\text{TSP}}(\mathbf{Q})$ to $\text{R}^{\rho}(\mathbf{Q})$, such that P_{max} is satisfied. The pseudo-code for computing $P^{\rho}_{\text{TSP}}(\mathbf{Q})$ is presented in Algorithm 3.

Lemma 5. For a given $\mathbf{Q}$, $\mathbf{P}^{\text{blocks}}$, T_{amb}, T_{DTM}, $p^{\text{core}}_{\text{inact}\,j}$ for $j = 1, 2, \ldots, M$, and floorplan, the value of power density

Algorithm 3 TSP (density) for a given mapping

```
Input: $\mathbf{Q}$, $\mathbf{P}^{\text{blocks}}$, $T_{\text{amb}}$, $T_{\text{DTM}}$, $p^{\text{core}}_{\text{inact}\,j}$ for $j = 1, 2, \ldots, M$, $P_{\max}$, and floorplan;
Output: Uniform power density constraint for mapping $\mathbf{Q}$;
    {First compute $P^{\rho\star}_{\text{TSP}}(\mathbf{Q})$ according to Equation (10)}
 1: $P^{\rho\star}_{\text{TSP}}(\mathbf{Q}) \leftarrow \infty$;
 2: for all $i \in \mathbf{L}$ do
 3:     $auxP \leftarrow T_{\text{DTM}} - \sum_{j=1}^{M} b^{-1}{}_{i,k_j} \cdot p^{\text{core}}_{\text{inact}\,j}(1 - q_j)$;
 4:     $auxP \leftarrow auxP - \sum_{j=1}^{N} b^{-1}{}_{i,j}\left(p^{\text{blocks}}_j + T_{\text{amb}} \cdot g_j\right)$;
 5:     $auxP \leftarrow \frac{auxP}{\sum_{j=1}^{M} b^{-1}{}_{i,k_j} \cdot \text{area}^{\text{core}}_j \cdot q_j}$;
 6:     if $auxP < P^{\rho\star}_{\text{TSP}}(\mathbf{Q})$ then
 7:         $P^{\rho\star}_{\text{TSP}}(\mathbf{Q}) \leftarrow auxP$;
 8:     end if
 9: end for
    {Then compute $P^{\rho}_{\text{TSP}}(\mathbf{Q})$ according to Equation (12)}
10: if $P^{\rho\star}_{\text{TSP}}(\mathbf{Q}) \leq \mathrm{R}^{\rho}(\mathbf{Q})$ then
11:     $P^{\rho}_{\text{TSP}}(\mathbf{Q}) \leftarrow P^{\rho\star}_{\text{TSP}}(\mathbf{Q})$;
12: else
13:     $P^{\rho}_{\text{TSP}}(\mathbf{Q}) \leftarrow \mathrm{R}^{\rho}(\mathbf{Q})$;
14: end if
15: return $P^{\rho}_{\text{TSP}}(\mathbf{Q})$;
```

constraint $P^{\rho\star}_{\text{TSP}}(\mathbf{Q})$ for each active core in the specified mapping (independent of the core types) is computed as

$$P^{\rho\star}_{\text{TSP}}(\mathbf{Q}) = \min_{\forall i \in \mathbf{L}} \left\{ \frac{T_{\text{DTM}} - \sum_{j=1}^{M} b^{-1}{}_{i,k_j} \cdot p^{\text{core}}_{\text{inact}\,j}(1 - q_j)}{\sum_{j=1}^{M} b^{-1}{}_{i,k_j} \cdot \text{area}^{\text{core}}_j \cdot q_j} + \frac{-\sum_{j=1}^{N} b^{-1}{}_{i,j}\left(p^{\text{blocks}}_j + T_{\text{amb}} \cdot g_j\right)}{\sum_{j=1}^{M} b^{-1}{}_{i,k_j} \cdot \text{area}^{\text{core}}_j \cdot q_j} \right\}. \tag{10}$$

Proof: Considering vector $\mathbf{Q}$, and set $\mathbf{K}$, if each inactive core j consumes $p^{\text{core}}_{\text{inact}\,j}$ and all active cores have equal power density P^{ρ}_{equal} at a given time, we have that for this time instant, Equation (2) can be rewritten as

$$T_i = P^{\rho}_{\text{equal}} \cdot \sum_{j=1}^{M} b^{-1}{}_{i,k_j} \cdot \text{area}^{\text{core}}_j \cdot q_j + \sum_{j=1}^{M} b^{-1}{}_{i,k_j} \cdot p^{\text{core}}_{\text{inact}\,j}(1 - q_j) + \sum_{j=1}^{N} b^{-1}{}_{i,j}\left(p^{\text{blocks}}_j + T_{\text{amb}} \cdot g_j\right). \tag{11}$$

The rest of the proof is equivalent to the proof of Lemma 1 simply by using Equation (11) instead of Equation (5) and P^{ρ}_{equal} instead of P_{equal}. □

***Theorem* 3.** For a given $\mathbf{Q}$, $\mathbf{P}^{\text{blocks}}$, T_{amb}, T_{DTM}, $p^{\text{core}}_{\text{inact}\,j}$ for $j = 1, 2, \ldots, M$, $P_{\max}$, and floorplan, the value of power density constraint $P^{\rho}_{\text{TSP}}(\mathbf{Q})$ for each active core in the specified mapping (independent of the types of cores) is computed as

$$P^{\rho}_{\text{TSP}}(\mathbf{Q}) = \begin{cases} P^{\rho\star}_{\text{TSP}}(\mathbf{Q}) & \text{if } P^{\rho\star}_{\text{TSP}}(\mathbf{Q}) \leq \mathrm{R}^{\rho}(\mathbf{Q}) \\ \mathrm{R}^{\rho}(\mathbf{Q}) & \text{otherwise.} \end{cases} \tag{12}$$

Proof: The first case is based on Lemma 5. For the second case, it should hold that $P^{\rho}_{\text{TSP}}(\mathbf{Q}) \cdot \sum_{i=1}^{M} \text{area}^{\text{core}}_i \cdot q_i + \sum_{i=1}^{M} p^{\text{core}}_{\text{inact}\,i}(1 - q_i) + \sum_{i=1}^{N} p^{\text{blocks}}_i \leq P_{\max}$. If $P^{\rho\star}_{\text{TSP}}(\mathbf{Q})$ is larger than $\mathrm{R}^{\rho}(\mathbf{Q})$ we simply set $P^{\rho}_{\text{TSP}}(\mathbf{Q})$ to $\mathrm{R}^{\rho}(\mathbf{Q})$, such that $P_{\max}$ is not violated. Thus, the theorem is proven. □

The total time complexity for computing $P^{\rho}_{\text{TSP}}(\mathbf{Q})$ for a given $\mathbf{P}^{\text{blocks}}$, T_{amb}, T_{DTM}, $P_{\max}$, and floorplan, is $O(ZN)$. To compute the corresponding TSP value for core j, we simply multiply $P^{\rho}_{\text{TSP}}(\mathbf{Q})$ with the area of core j.

6.2 TSP for Worst-Case Mappings (Heterogeneous)

This subsection revises Algorithm 2 presented in Section 5.2 in order to consider heterogeneous systems. Similar to Section 6.1, we will focus on deriving a safe *power density* constraint, rather than focus on power consumption. Moreover, in Section 5.2 the value of the power constraint for the worst-case core mappings depends on the number of active cores. Nevertheless, for the heterogeneous case, since different core types have different areas, the value of the power density constraint will depend on the *number of active cores for each type of core*. That is, assume that there are W core types (arbitrarily ordered) and that for every type of core w there are a total of M_w cores, such that $M = \sum_{w=1}^{W} M_w$. We define sets $\mathbf{K}^w = \left\{k^w_1, k^w_2, \ldots, k^w_{M_w}\right\}$ for all core types $w = 1, 2, \ldots, W$, such that $\mathbf{K}^w$ contains the indexes of the thermal nodes that correspond to cores of type w. Furthermore, set $\mathbf{m} = \{m_1, m_2, \ldots, m_W\}$ represents the number of active cores for core types $\{1, 2, \ldots, W\}$. Hence, in this section we derive a safe *power density* constraint for the worst-case core mappings with $\mathbf{m} = \{m_1, m_2, \ldots, m_W\}$ active cores, and we define such a power density constraint as $P^{\rho\,\text{worst}}_{\text{TSP}}(\mathbf{m})$. For example, if we have a system with three types of cores and we would like to activate 4 cores of type 1 and 7 cores of type 3, the safe power density constraint for such a case is defined as $P^{\rho\,\text{worst}}_{\text{TSP}}(\{4, 0, 7\})$. We then simply multiply the power density constraint by the area of each core, thus obtaining a safe power constraint value for each type of core for *any* possible core mapping with $\mathbf{m} = \{m_1, m_2, \ldots, m_W\}$ simultaneously active cores, that results in a maximum temperature (in the steady-state) among all cores which does not exceed T_{DTM}.

We assume that all cores of the same type have equal area and inactive power, and we denote the area and inactive power of core type w as $\text{area}^{\text{type}}_w$ and $p^{\text{type}}_{\text{inact}\,w}$, respectively. Similar to Section 5.2, we define auxiliary matrix $\mathbf{H}^{\rho} = \left[h^{\rho}_{w,i,j}\right]_{W \times Z \times M_w}$ which is used in Lemma 8 to compute the maximum amount of heat that any m_w cores of type w can contribute to the temperature on node i, for all core types $w = 1, 2, \ldots, W$. Matrix $\mathbf{H}^{\rho}$ is built by making a partial copy of matrix $\mathbf{B}^{-1}$, considering the type and the area of the cores, such that $h^{\rho}_{w,i,j} = b^{-1}{}_{\ell_i, k^w_j} \cdot \text{area}^{\text{type}}_w$ for all $w = 1, 2, \ldots, W$, for all $i = 1, 2, \ldots, Z$, and for all $j = 1, 2, \ldots, M_w$, and then, for every w and every i, reordering each row of $\mathbf{H}^{\rho}$ in a decreasing manner. Matrix $\mathbf{H}^{\rho}$ needs to be built and ordered only one time, which requires a time complexity of $O(ZM_w \log M_w)$ for every core type w (the pseudo-code for building matrix $\mathbf{H}^{\rho}$ is later included in Algorithm 4). Furthermore, we also define column vector $\mathbf{Q}^w = [q^w_i]_{M_w \times 1}$ for all core types $w = 1, 2, \ldots W$, which represents a particular mapping of *active* cores of type w. Vector $\mathbf{Q}^w$ is a binary vector: $q^w_i = 1$ means that core i corresponds to an *active* core of type w; $q^w_i = 0$ means that core i corresponds to an *inactive* core of type w.

The derivation procedure and properties in this subsection are very similar to those in Section 5.2. Namely, for a given $\mathbf{m} = \{m_1, m_2, \ldots, m_W\}$, our algorithm is presented in Theorem 4, based on the auxiliary matrix $\mathbf{H}^{\rho}$ and the properties in Lemmas 6, 7, and 8. Lemma 6 and Lemma 7 prove that using matrix $\mathbf{H}^{\rho} = \left[h^{\rho}_{w,i,j}\right]_{W \times Z \times M_w}$ results in the most pessimistic power density constraints. Lemma 8 ignores the maximum chip power and derives the worst-case uniform power density constraint for $\mathbf{m}$ active cores, defined as $P^{\rho\star\text{worst}}_{\text{TSP}}(\mathbf{m})$. Theorem 4 verifies that the maximum power $P_{\max}$ is not violated. We define auxiliary function $\mathrm{R}^{\rho}(\mathbf{m})$ as

$$\mathrm{R}^{\rho}(\mathbf{m}) = \frac{P_{\max} - \sum_{i=1}^{N} p^{\text{blocks}}_i - \sum_{w=1}^{W} p^{\text{type}}_{\text{inact}\,w}(M_w - m_w)}{\sum_{w=1}^{W} \text{area}^{\text{type}}_w \cdot m_w}. \tag{13}$$

If having a power density of $P_{\text{TSP}}^{\rho\star\text{worst}}(\mathbf{m})$ in $\mathbf{m} = \{m_1, m_2, \dots, m_W\}$ cores violates $P_{\max}$, then Theorem 4 sets $P_{\text{TSP}}^{\rho\text{ worst}}(\mathbf{m})$ to $\mathrm{R}^{\rho}(\mathbf{m})$, such that $P_{\max}$ is satisfied.

Lemma 6. If all active cores have equal power density P_{equal}^{ρ}, the m_w highest values for $P_{\text{equal}}^{\rho} \cdot b^{-1}{}_{i,j} \cdot \text{area}_w^{\text{type}}$ such that $j \in \mathbf{K}^w$ for all $w = 1, 2, \dots, W$, correspond to the amount of heat contributed to T_i by the $\mathbf{m} = \{m_1, m_2, \dots, m_W\}$ cores of type $\{1, 2, \dots, W\}$ that contribute more heat into T_i.

Proof: From the definition of matrix $\mathbf{B}^{-1}$, we know that $b^{-1}{}_{i,k_j} \cdot \frac{p_{k_j}}{\text{area}_j^{\text{core}}} \cdot \text{area}_j^{\text{core}}$ represents the heat contributed by core j into the steady-state temperature of node i, i.e., T_i. By grouping the mappings of different types of cores together, Equation (11) becomes

$$T_i = P_{\text{equal}}^{\rho} \cdot \sum_{w=1}^{W} \sum_{j=1}^{M_w} b^{-1}{}_{i,k_j^w} \cdot \text{area}_w^{\text{type}} \cdot q_j^w + \sum_{w=1}^{W} \sum_{j=1}^{M_w} b^{-1}{}_{i,k_j^w} \cdot p_{\text{inact}w}^{\text{type}} \left(1 - q_j^w\right) + \sum_{j=1}^{N} b^{-1}{}_{i,j} \left(p_j^{\text{blocks}} + T_{\text{amb}} \cdot g_j\right), \tag{14}$$

from which we know that there is one or more mappings that result in the maximum T_i for a given P_{equal}^{ρ}. Given that $\mathbf{B}^{-1}$, $\mathbf{G}$, $\mathbf{P}^{\text{blocks}}$, and T_{amb} are constant, and $p_{\text{inact}w}^{\text{type}}$ is smaller than the minimum *active* power consumption of a core of type w, it is clear that, for a given P_{equal}^{ρ}, T_i is maximized when $\sum_{w=1}^{W} \sum_{j=1}^{M_w} b^{-1}{}_{i,k_j^w} \cdot \text{area}_w^{\text{type}} \cdot q_j^w$ is also maximized. Moreover, it holds that the summation of the elements in $\mathbf{Q}^w$ is equal to the number of active cores of type w, i.e., $\sum_{j=1}^{M_w} q_j^w = m_w$. Hence, for row i, we have that $\sum_{w=1}^{W} \sum_{j=1}^{M_w} b^{-1}{}_{i,k_j^w} \cdot \text{area}_w^{\text{type}} \cdot q_j^w$ is maximized when, for all $w = 1, 2, \dots, W$, mapping $\mathbf{Q}^w$ activates the m_w cores with the highest $b^{-1}{}_{i,k_j^w} \cdot \text{area}_w^{\text{type}}$. Thus, the lemma is proven. □

Lemma 7. If all active cores have equal power density P_{equal}^{ρ}, multiplying the power density on each core with the summation of the first m_w elements that correspond to cores of type w in row i of matrix $\mathbf{H}^{\rho}$, that is, computing $P_{\text{equal}}^{\rho} \cdot \sum_{j=1}^{m_w} h_{w,i,j}^{\rho}$, results in the maximum amount of heat that any m_w cores of type w can contribute to the steady-state temperature on node ℓ_i, that is, T_{ℓ_i}.

Proof: Based on the definition of auxiliary matrix $\mathbf{H}^{\rho}$ and Lemma 6, the first m_w elements that correspond to cores of type w in row i of matrix $\mathbf{H}^{\rho}$ are the m_w highest $b^{-1}{}_{\ell_i,j} \cdot \text{area}_w^{\text{type}}$ values, for node ℓ_i, such that $j \in \mathbf{K}^w$. In turn, multiplied by P_{equal}^{ρ}, this is the amount of heat contributed to temperature T_{ℓ_i} by the m_w cores of type w that contribute more heat to T_{ℓ_i}. Thus, the lemma is proven. □

Lemma 8. For a given $\mathbf{P}^{\text{blocks}}$, T_{amb}, T_{DTM}, $p_{\text{inact}w}^{\text{type}}$ for $w = 1, 2, \dots, W$, and floorplan, the value of power density constraint $P_{\text{TSP}}^{\rho\star\text{worst}}(\mathbf{m})$ (independent of the types of cores) for each active core in *any* possible core mapping with $\mathbf{m} = \{m_1, m_2, \dots, m_W\}$ simultaneously active cores is computed as

$$P_{\text{TSP}}^{\rho\star\text{worst}}(\mathbf{m}) = \min_{1 \leq i \leq Z} \left\{ \frac{T_{\text{DTM}} - \sum_{w=1}^{W} \frac{p_{\text{inact}w}^{\text{type}}}{\text{area}_w^{\text{type}}} \sum_{j=m_w+1}^{M_w} h_{w,i,j}^{\rho}}{\sum_{w=1}^{W} \sum_{j=1}^{m_w} h_{w,i,j}^{\rho}} + \frac{-\sum_{j=1}^{N} b^{-1}{}_{\ell_i,j} \left(p_j^{\text{blocks}} + T_{\text{amb}} \cdot g_j\right)}{\sum_{w=1}^{W} \sum_{j=1}^{m_w} h_{w,i,j}^{\rho}} \right\}. \tag{15}$$

Algorithm 4 TSP (density) for all worst-case mappings

Input: $\mathbf{P}^{\text{blocks}}$, T_{amb}, T_{DTM}, $p_{\text{inact}w}^{\text{type}}$ for $w = 1, 2, \dots, W$, $P_{\max}$, and floorplan;
Output: Uniform power density for all worst-case mappings;

```
{First build auxiliary matrix $\mathbf{H}^{\rho} = \left[h_{w,i,j}^{\rho}\right]_{W \times Z \times M_w}$}
for $w = 1$ to $W$ do
    for $i = 1$ to $Z$ do
        for $j = 1$ to $M_w$ do
            $h_{w,i,j}^{\rho} \leftarrow b^{-1}{}_{\ell_i,k_j^w} \cdot \text{area}_w^{\text{type}}$;
        end for
        For these $w$ and $i$, re-order $\mathbf{H}^{\rho}$ (with respect to $j$) decreasingly;
    end for
end for
for $m_1 = 1$ to $M_1$ do
    for $m_2 = 1$ to $M_2$ do
        ⋮
        for $m_W = 1$ to $M_W$ do
            {Compute $P_{\text{TSP}}^{\rho\star\text{worst}}(\mathbf{m})$ from Equation (15)}
            $P_{\text{TSP}}^{\rho\star\text{worst}}(\mathbf{m}) \leftarrow \infty$;
            for $i = 1$ to $Z$ do
                $auxP \leftarrow T_{\text{DTM}} - \sum_{w=1}^{W} \frac{p_{\text{inact}w}^{\text{type}}}{\text{area}_w^{\text{type}}} \cdot \sum_{j=m_w+1}^{M_w} h_{w,i,j}^{\rho}$;
                $auxP \leftarrow auxP - \sum_{j=1}^{N} b^{-1}{}_{\ell_i,j} \left(p_j^{\text{blocks}} + T_{\text{amb}} \cdot g_j\right)$;
                $auxP \leftarrow \frac{auxP}{\sum_{w=1}^{W} \sum_{j=1}^{m_w} h_{w,i,j}^{\rho}}$;
                if $auxP < P_{\text{TSP}}^{\rho\star\text{worst}}(\mathbf{m})$ then
                    $P_{\text{TSP}}^{\rho\star\text{worst}}(\mathbf{m}) \leftarrow auxP$;
                end if
            end for
            {Compute $P_{\text{TSP}}^{\rho\text{ worst}}(\mathbf{m})$ according to Equation (16)}
            if $P_{\text{TSP}}^{\rho\star\text{worst}}(\mathbf{m}) \leq \mathrm{R}^{\rho}(\mathbf{m})$ then
                $P_{\text{TSP}}^{\rho\text{ worst}}(\mathbf{m}) \leftarrow P_{\text{TSP}}^{\rho\star\text{worst}}(\mathbf{m})$;
            else
                $P_{\text{TSP}}^{\rho\text{ worst}}(\mathbf{m}) \leftarrow \mathrm{R}^{\rho}(\mathbf{m})$;
            end if
        end for
        ⋮
    end for
end for
return $P_{\text{TSP}}^{\rho\text{ worst}}(\mathbf{m})$ for all combinations of active cores of different types;
```

Proof: Considering matrix $\mathbf{H}^{\rho}$ and Lemma 7, Equation (14) becomes

$$T_{\ell_i} = P_{\text{equal}}^{\rho} \sum_{w=1}^{W} \sum_{j=1}^{m_w} h_{w,i,j}^{\rho} + \sum_{w=1}^{W} \frac{p_{\text{inact}w}^{\text{type}}}{\text{area}_w^{\text{type}}} \sum_{j=m_w+1}^{M_w} h_{w,i,j}^{\rho} + \sum_{j=1}^{N} b^{-1}{}_{\ell_i,j} \left(p_j^{\text{blocks}} + T_{\text{amb}} \cdot g_j\right).$$

Similar to Lemma 5, by setting T_{ℓ_i} to T_{DTM}, we can compute, for every $i = 1, 2, \dots, Z$, what value of P_{equal}^{ρ} would make T_{ℓ_i} reach T_{DTM}. The most pessimistic value of P_{equal}^{ρ} is a safe power density constraint for the $\mathbf{m}$ active cores, and is therefore the resulting $P_{\text{TSP}}^{\rho\star\text{worst}}(\mathbf{m})$ for the given $\mathbf{P}^{\text{blocks}}$, T_{amb}, $p_{\text{inact}w}^{\text{type}}$ for $w = 1, 2, \dots, W$, and floorplan, which is expressed in Equation (15). Thus, the lemma is proven. □

Theorem 4. For a given $\mathbf{P}^{\text{blocks}}$, T_{amb}, T_{DTM}, $p_{\text{inact}w}^{\text{type}}$ for $w = 1, 2, \dots, W$, $P_{\max}$, and floorplan, the value of power density constraint $P_{\text{TSP}}^{\rho\text{ worst}}(\mathbf{m})$ (independent of the types of cores) for each active core in *any* possible core mapping with $\mathbf{m} = \{m_1, m_2, \dots, m_W\}$ simultaneously active cores is computed as

$$P_{\text{TSP}}^{\rho\text{ worst}}(\mathbf{m}) = \begin{cases} P_{\text{TSP}}^{\rho\star\text{worst}}(\mathbf{m}) & \text{if } P_{\text{TSP}}^{\rho\star\text{worst}}(\mathbf{m}) \leq \mathrm{R}^{\rho}(\mathbf{m}) \\ \mathrm{R}^{\rho}(\mathbf{m}) & \text{otherwise.} \end{cases} \tag{16}$$

Proof: The first case is based on Lemma 8. For the second case, it should hold that $P_{\text{TSP}}^{\rho\text{ worst}}(\mathbf{m}) \cdot \sum_{w=1}^{W} \text{area}_w^{\text{type}} \cdot m_w + \sum_{w=1}^{W} p_{\text{inact}w}^{\text{type}} \left(M_w - m_w\right) + \sum_{i=1}^{N} p_i^{\text{blocks}} \leq P_{\max}$. If $P_{\text{TSP}}^{\rho\star\text{worst}}(\mathbf{m})$ is larger than $\mathrm{R}^{\rho}(\mathbf{m})$, then we simply set it to $\mathrm{R}^{\rho}(\mathbf{m})$, such that $P_{\max}$ is not violated. Thus, the theorem is proven. □

Given that matrix $\mathbf{H}^{\rho}$ only needs to be built once for a given chip, the total time complexity for computing

$P_{\text{TSP}}^{\rho\,\text{worst}}(\mathbf{m})$ for a given $\mathbf{m} = \{m_1, m_2, \ldots, m_W\}$ is $O(ZN)$, and for computing $P_{\text{TSP}}^{\rho\,\text{worst}}(\mathbf{m})$ for all possible combinations of active cores of different types is $O\left(ZN \prod_{w=1}^{W} M_w\right)$. The corresponding pseudo-code is presented in Algorithm 4.

7 Evaluations for Homogeneous Systems

This section presents evaluations conducted with gem5 [3], McPAT [20], and HotSpot [14], for homogeneous systems. We compare the total system performance of seven different power budget techniques: TSP for given mappings, worst-case TSP, three constant power budgets per-chip, a constant power budget per-core, and an online boosting technique, specifically, Intel's Turbo Boost [15], [26].

7.1 Setup

For our hardware platform, we consider the 64 cores system presented in Fig. 3, based on simulations conducted on gem5 [3] and McPAT [20] for *out-of-order* Alpha 21264 cores in 22 nm technology. The cores are composed by several units: an instruction fetch unit (IFU), an execution unit (EXU), a load and store unit (LSU), an out-of-order (OOO) issue/dispatch, and a private L1 cache. According to our simulations, each core has an area of $9.6\,\text{mm}^2$. There is a shared L2 cache of 2 MB and a memory controller every 4 cores, with a combined area of $4.7\,\text{mm}^2$. We assume available frequencies $\{0.2, 0.4, \ldots, 4.0\}$ GHz, and the voltage settings for each frequency are taken from [12].

For such a floorplan, we consider one thermal block for each core and independent thermal blocks for the L2 caches and other hardware. We then obtain the values for $\mathbf{B}$, $\mathbf{B}^{-1}$, and $\mathbf{G}$, by using HotSpot [14] v5.02 with its default configuration in the block model mode. We consider that the ambient temperature is 45°C, $P_{\max}$ is 250 W (common value in the literature [15]), and the threshold temperature that triggers DTM, T_{DTM}, is 80°C. For all the evaluated power constraints, except when a boosting technique is applied, we consider a standard reactive control-based closed-loop DTM technique [15] that, when the threshold temperature is exceeded anywhere in the chip, it reduces the voltage/frequency levels of all active cores, one step at a time, by using a control period of 1 ms. When the maximum temperature in the chip is below T_{DTM}, the voltage/frequency on the cores is increased one step at a time, by using the same 1 ms control period, until all cores reach their nominal operation according to the established power constraint.

For benchmarks, we use applications from the PARSEC benchmark suite [2]. Specifically, an x264 application (H.264 video encoder), a body track application, an option pricing application with Black-Scholes Partial Differential Equation, and a pricing application of a portfolio of swaptions. All applications can run $1, 2, \ldots, 8$ parallel dependent threads. For each experiment, we conduct closed-loop evaluations involving simulations with gem5 [3], McPAT [20], and HotSpot [14]. We consider that every time an application instance finishes, another instance is immediately executed. Further details about our simulation framework are included in Appendix D.

7.2 Power Constraints

In this subsection we compute the power constraints used in our simulations. Using Algorithm 2, we compute TSP for the worst-case mappings, i.e., $P_{\text{TSP}}^{\text{worst}}(m)$ for all $m = 1, 2, \ldots, M$. Fig. 7 presents the computed TSP values per-core as $\text{TSP}_{\text{worst}}$, resulting in a decreasing function with respect to the number of active cores. For presentation purposes, Fig. 8 shows TSP estimations at a chip level, by multiplying the $\text{TSP}_{\text{worst}}$ values from Fig. 7 with the number of active cores for each case (table representations of these figures are included in Appendix E). In Fig. 8, TSP results in a non-decreasing function with respect to the number of active cores. Moreover, to compare TSP for different core mappings, we consider $64 \cdot 10^5$ random mappings (10^5 for each number of active cores), compute TSP using Algorithm 1 for every given mapping, and present the highest resulting TSP values as TSP_{best} in Fig. 7 and Fig. 8. On a desktop computer with a 64-bit quad-core Intel Sandybridge i5-2400 CPU running at 3.10GHz, we measure the execution time required by Algorithm 1 to compute TSP for every random mapping, resulting in an execution time of at most 5.47 ms among all measurements, *suitable for online usage*.

For the constant per-chip power constraints, since this is a simulated platform, we cannot simply consider TDP because we have no datasheet with that information. Thus, we consider three different per-chip power budgets, which coincide with $m \cdot P_{\text{TSP}}^{\text{worst}}(m)$ for $m = 4$, $m = 16$, and $m = 64$. Specifically, these power budgets (shown as horizontal lines in Fig. 8) are 80 W per-chip, 150 W per-chip, and 225 W per-chip, which are also representative TDP values for current technologies [15]. In Fig. 7, we estimate the maximum power per core when these constraints are equally distributed among the active cores. For the constant per-core power

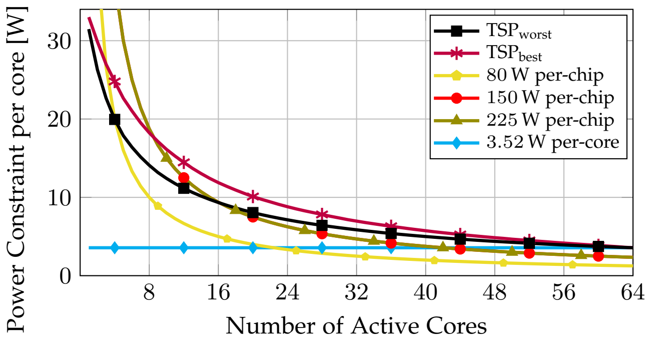


Fig. 7: Worst- and best-case TSP for the floorplan in Fig. 3, compared to a constant per-core budget, and estimations of constant per-chip budgets equally distributed among active cores.

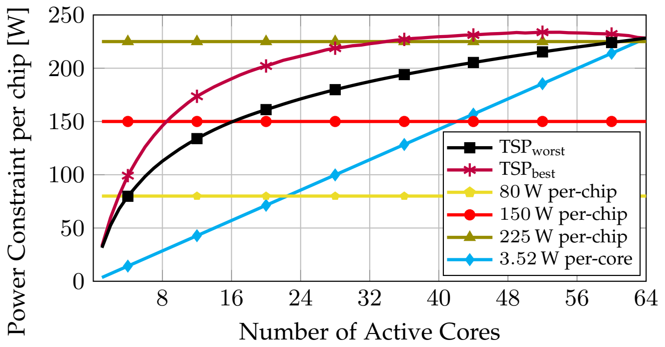


Fig. 8: Constant per-chip budgets, compared to multiplying the number of active cores by a constant per-core budget, and the worst- and best-case TSP for the floorplan from Fig. 3.

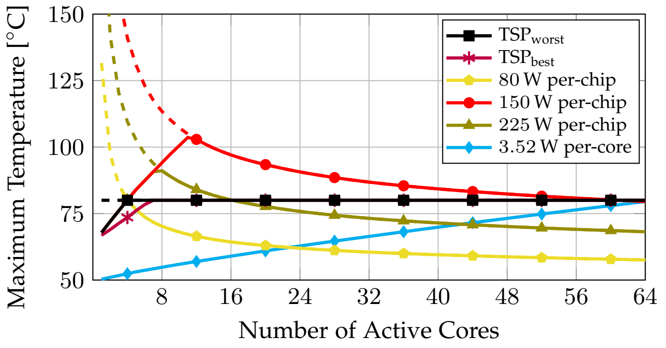


Fig. 9: Maximum steady-state temperatures among all blocks (DTM deactivated), when using TSP, a constant per-core budget, and equally distributed constant per-chip budgets.

constraint, we consider it to be equal to TSP when simultaneously activating all cores, i.e., $P^{\text{worst}}_{\text{TSP}}(m)$ for $m = 64$. This results in a constant power budget of 3.52 W per-core, represented by a horizontal line in Fig. 7 and by an increasing linear function in Fig. 8. *Note that representing TSP and the constant per-core power budget in Fig. 8 does not mean that either constraint should be considered at a per-chip level. Both budgets should be strictly considered at a per-core level. We include them in Fig. 8 only to compare their resulting total system power to the per-chip budgets. Similarly, the opposite applies when representing the constant per-chip budgets in Fig. 7.*

With regards to temperature, Fig. 9 presents simulations conducted in HotSpot that show the maximum temperature (in the steady-state) among all blocks as a function of the number of simultaneously active cores, for the discussed power budgets, in case that DTM is deactivated. As expected, when consuming TSP in all active cores, the maximum temperature among all blocks is 80°C. Moreover, from Fig. 9 we can conclude that whenever TSP is greater than any constant power budget, if such a power budget is used as the power constraint, the system could actually consume more power without reaching T_{DTM}, which accounts for performance losses. Contrarily, when a constant power budget exceeds TSP, this means that if the cores consume more power than TSP, most likely DTM will be triggered almost the entire time. Note that for some cases in Fig. 9, particularly when executing just a few cores, the cores never consume the entire power budget even when executing at maximum voltage/frequency (e.g., in our experiments no individual core ever consumes more than 20 W). Therefore, the dashed lines in the figure show the potential maximum temperature if each power budget would have been entirely consumed, while the solid lines show the maximum temperatures that can be practically achieved.

7.3 Dark Silicon Estimations

Through Fig. 7, we can easily estimate the amount of dark silicon. For example, we assume that the minimum power consumption for activating a core at its lowest speed is 4 W (*not a real practical setting, but chosen for presentation purposes*). Then, considering the values of TSP, no more than 54 cores could be active simultaneously, which results in 15.63% of the chip being dark at all times. If a constant per-chip power budget is used for the same example estimations, e.g., 150 W per-chip, then only 37 cores could be activated simultaneously, resulting in 42.19% of the chip being dark, which is much higher than the estimations for TSP.

7.4 Performance Simulations

Fig. 10 presents the average total IPS count, for considering different numbers of active cores and the different power budgets described in Section 7.2. We assume that at nominal operation every active core runs at the highest possible frequency and performance such that the power budget under consideration is not exceeded.

Furthermore, in order to also compare with an online boosting technique, in Fig. 10 we also consider Intel's Turbo Boost [15], [26]. When the temperature on all blocks is below T_{DTM} the cores boost their voltage and frequency levels in single steps (within a control period of 1ms). Similarly, if some temperature exceeds T_{DTM} the cores reduce their voltage and frequency levels in single steps (within a control period of 1ms) until the thermal constraints are satisfied.

In Fig. 10, we can see that when we activate all 64 cores, the per-core budget and the 225 W per-chip budget achieve the same performance that $\text{TSP}_{\text{worst}}$. This is expected, as in both cases these power budgets coincide with TSP. However, there are many cases in which the fixed power budgets are pessimistic and thus the chip remains underutilized, as explained in Section 7.2. Specifically, this happens with all other cases for the per-core power budget, and with the 80 W and 150 W per-chip budgets when activating more than 4 and 16 cores, respectively. Contrarily, there are many other cases in which the fixed power budgets constantly trigger DTM, specifically, when activating less than 16 or 64 cores with the 150 W or 225 W per-chip budget, respectively. Here, since DTM is constantly triggered, thermal violations are avoided at the cost of reducing the frequency of the cores for long periods of time. Therefore, the resulting performance of the per-chip budgets is in general much worse than originally expected by the task partitioning and mapping algorithms. Furthermore, for such cases there is no simple way of predicting such performance losses, making it almost impossible for the system to provide performance and timing guarantees. Contrarily, TSP never triggers DTM and can thus achieve the expected performance of the task partitioning and mapping decisions. The *average percentage increase in performance* (among all number of active cores and applications) for using $\text{TSP}_{\text{worst}}$ results in a 12% higher average total IPS compared to all constant power budgets. There are just a few cases in which the resulting performance of the per-chip budgets is higher than that of TSP. This happens mostly because a per-chip power budget can potentially execute different cores at different frequencies, reducing the thermal headroom for these few cases, while under TSP all cores share the same power budget and sometimes a certain frequency setting slightly exceeds this budget and a lower one results in large thermal headroom.

Intel's Turbo Boost is a simple but very efficient online technique, achieving a higher performance than $\text{TSP}_{\text{worst}}$ for several cases. Namely, while under TSP the voltage/frequency of the cores remains constant (as does the performance), under Turbo Boost the voltage/frequency levels are constantly changing, thus exploiting the thermal capacitances of the thermal model by knowing that the associated temperature changes require some time to follow the changes in power. In this way, the instantaneous performance of Turbo Boost can sometimes be much higher than TSP for some time intervals (when there is thermal headroom and the voltage/frequency is increased), and

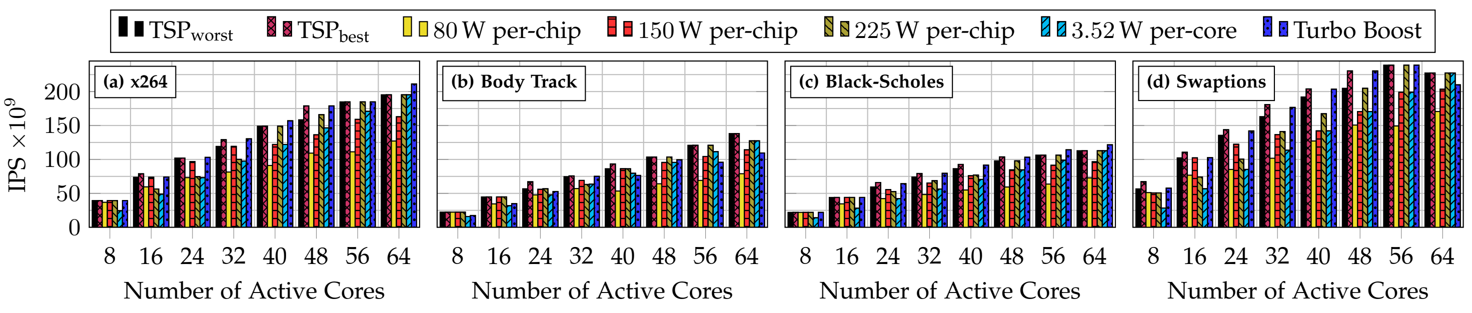


Fig. 10: Evaluation results: average total system performance for *homogeneous* systems when using different power budgets.

much lower for other time intervals (when the critical temperature is reached and the voltage/frequency is reduced). After computing the average performance of Turbo Boost, we observe that $\text{TSP}_{\text{worst}}$ results in the same average total IPS (among all number of active cores and all applications). However, similar to having frequent triggers of DTM, there are no simple offline performance predictions suitable for such boosting techniques, and thus no timing guarantees can be provided in advance. Hence, Turbo Boost cannot guide the task partitioning and mapping algorithms to make intelligent decisions. Contrarily, this can be done with TSP, helping to simplify task partitioning and mapping algorithms to achieve a high *predictable* performance without thermal violations.

Finally, there are no restrictions prohibiting the combination of Turbo Boost and TSP, and in this can potentially result in a higher performance than applying each technique by itself. Namely, TSP can be used to guide the task partitioning, mapping, and DVFS selection algorithms to make intelligent decisions that optimize the performance without incurring in thermal violations at nominal operation. Then, Turbo Boost can be applied on top of such a solution in order to exploit any thermal headroom available at runtime by boosting the cores to execute at power levels higher than TSP. Opposed to standard Turbo Boost, in which once the threshold temperature is reached the voltage/frequency levels are decreased until the temperature is again below the threshold, in this case Turbo Boost can stop decreasing the voltage/frequency of the cores at the nominal operation levels that satisfy TSP, as we know that this is thermally safe. *Therefore, by combining TSP and Turbo Boost, the system can achieve a high predictable performance while also exploiting the thermal headroom available at runtime.*

8 Evaluations for Heterogeneous Systems

8.1 Setup

For our hardware platform, we consider the 72 cores system presented in Fig. 5, consisting of 24 *out-of-order* (OOO) Alpha 21264 cores and 16 *simple in-order* Alpha 21264 cores, based on simulations conducted on gem5 [3] and McPAT [20] for 22 nm, and 16 *in-order* Cortex-A7 cores and 16 OOO Cortex-A15 cores, based on an Odroid-XU3 [13] mobile platform with an Exynos 5 Octa (5422) [27] chip with ARM's "big.LITTLE" architecture. According to our simulations, each OOO Alpha core has an area of $9.6\,\text{mm}^2$ with a shared 2 MB L2 cache every eight cores, and each *simple* Alpha core has an area of $1.6\,\text{mm}^2$ with a shared 2 MB L2 cache every four cores, all connected to a 512 MB RAM (executing at 1 GHz, with 73 GB/s memory bandwidth). The areas of the A7 and A15 cores are estimated from die

Scenario	Alpha OOO	Alpha *simple*	A15	A7
S1	a: 8 threads b: 8 threads c: 2 threads 6 threads	a: - b: - c: 3 threads d: 2 threads	a: 4 threads b: 2 threads c: 2 threads d: -	a: - b: 2 threads c: 2 threads d: 4 threads
S2	a: 5 threads 3 threads b: - c: 7 threads 1 threads	a: 4 threads b: 1 threads c: 2 threads d: -	a: 2 threads b: - c: 1 threads d: 4 threads	a: 4 threads b: - c: - d: 2 threads
S3	a: 4 threads b: 4 threads 4 threads c: -	a: 2 threads b: - c: 3 threads d: -	a: 4 threads b: 1 threads c: - d: -	a: 2 threads b: 2 threads c: - d: -
S4	a: 4 threads 4 threads b: 4 threads 4 threads c: 4 threads 4 threads	a: 4 threads b: 4 threads c: 4 threads d: 4 threads	a: 4 threads b: 4 threads c: 4 threads d: 4 threads	a: 4 threads b: 4 threads c: 4 threads d: 4 threads

TABLE 1: Details of the application mapping scenarios for our experiments. Indexes a, b, c, and d represent the cluster ID as explained in Fig. 5. Every line corresponds to an application instance executed in the corresponding cluster with the indicated number of threads, where "-" means that a cluster is not executing any application.

figures of the Exynos 5 Octa (5422) [27] chip as $0.8\,\text{mm}^2$ and $5.0\,\text{mm}^2$, respectively. There is a shared 512 KB L2 cache every four A7 cores, and a shared 2 MB L2 cache every four A15 cores, connected through two low-power multi-layer 32 bit buses to a 2 GB LPDDR3 RAM PoP (executing at 933 MHz, with 14.9 GB/s memory bandwidth). For the OOO and *simple* Alpha cores, we assume available frequencies $\{0.2, 0.4, \ldots, 4.0\}$ GHz, and the voltage settings for each frequency are taken from the work in [12]. For the A7 and A15 cores, the available frequencies in the Odroid-XU3 platform are $\{0.2, 0.3, \ldots, 1.4\}$ GHz and $\{1.2, 1.3, \ldots, 2.0\}$ GHz, respectively, and the voltage values are automatically selected by the platform.

The RC thermal network for such a floorplan is obtained as detailed in Section 7.1. We also use the same ambient temperature, P_{max}, T_{DTM}, DTM technique, and benchmarks described in Section 7.1. Furthermore, as different applications have different power consumptions depending on the type of cores and number of threads in which they are executed, we run the applications from the PARSEC benchmark suite under different scenarios. Particularly, we focus on different applications individually, considering multiple instances of the same application, with different number of threads per instance and also different thread-to-core mappings. Details can be found in Table 1. For each scenario in Table 1, we conduct closed-loop evaluations involving simulations with gem5 [3] and McPAT [20] for the Alpha cores, power and performance traces from real measurements in the

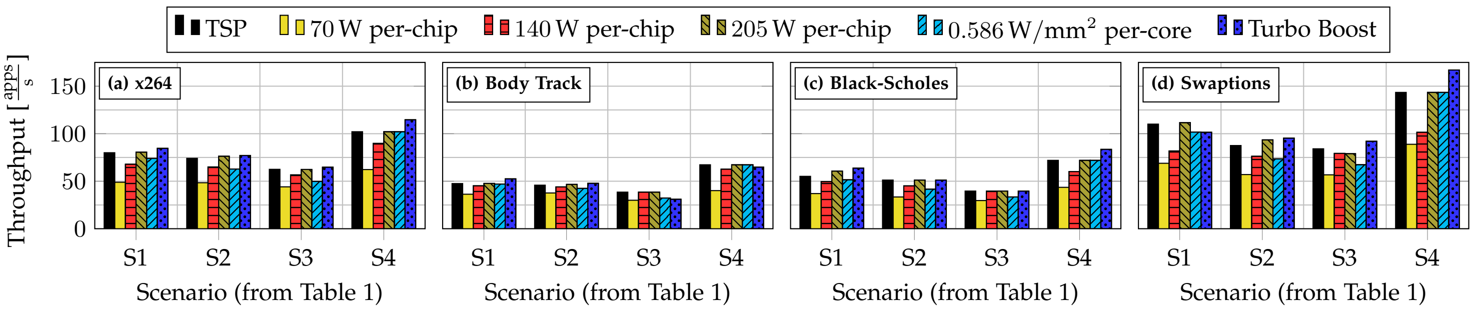


Fig. 11: Evaluation results: average total system performance on *heterogeneous* systems when using different power budgets.

Odroid-XU3 platform for the A7 and A15 cores, and thermal simulations with HotSpot [14]. Given that the Odroid-XU3 platform has no performance counters to measure the total number executed instructions, we use throughput as our performance metric, where throughput is defined as the total number of application instances finished every second.

8.2 Power Constraints

We use similar power constraints as those described in Section 7.2, but extended for heterogeneous systems. Using Algorithm 3 we compute TSP for the given mappings in Table 1. For the per-chip power constraints, we choose 205 W per-chip (total active power of using TSP when all cores active), and 140 W and 70 W per-chip (to have a similar relation with the per-chip constraints from Section 7.2). In the evaluations in Section 7, the total per-chip power budgets were equally divided among the number of active cores for each experiment. For the heterogeneous case, the per-chip power constraints are *proportionally* divided according to the *area of the active cores*. For example, when activating two OOO Alpha cores and three A7 cores under the 140 W per-chip power constraint, the total active area for these cores is $2 \cdot 9.6\,\text{mm}^2 + 3 \cdot 0.8\,\text{mm}^2 = 21.6\,\text{mm}^2$, such that each OOO Alpha core can consume up to $\frac{9.6\,\text{mm}^2}{21.6\,\text{mm}^2} \cdot 140\,\text{W} = 62.2\,\text{W}$ and each A7 core can consume up to 5.2 W. Finally, we use $0.586\,\text{W/mm}^2$ as the per-core power constraints, by simply dividing 205 W by the total core area in the chip, i.e., $350.36\,\text{mm}^2$.

8.3 Performance Simulations

Fig. 11 presents the average total throughput for considering the different mappings from Table 1 and the different power budgets described in Section 8.2. Similar to Section 7.4, we also compare with Intel's Turbo Boost [15], [26]. The observations of the results are similar to those in Section 7.4. Additionally, developing intelligent and efficient task partitioning and mapping algorithms is much more challenging for heterogeneous systems. Therefore, TSP can dearly help reduce the complexity of such decisions, achieving a high *predictable* performance without thermal violations.

9 Conclusions

Using a *single* and *constant* power constraint, e.g., TDP, is a pessimistic approach for homogeneous and heterogeneous manycore systems. This paper presents a new power budget concept, called Thermal Safe Power (TSP), which results in a high total system performance, while the maximum temperature among all cores remains below the threshold level that triggers DTM. TSP is a fundamental new step and advancement towards dealing with the dark silicon problem as it alleviates the pessimistic bounds of TDP and enables system designers and architects to explore new avenues for performance improvements in the dark silicon era.

For a specific floorplan and ambient temperature, TSP can be computed offline to obtain safe power and power density constraints for the worst cases, allowing the system designers to abstract from mapping decisions. Moreover, TSP can also be computed online, for a particular mapping of active cores and ambient temperature. The simulations show the validity of our arguments, comparing the total performance of using TSP, several constant power constraints, and a boosting technique. TSP can also be used to estimate the amount of dark silicon, which results in less pessimistic estimations than those using constant power budgets.

References

[1] H. Amrouch, B. Khaleghi, A. Gerstlauer, , and J. Henkel, "Reliability-aware design to suppress aging," in *the 53rd IEEE/ACM Design Automation Conference (DAC)*, June 2016.
[2] C. Bienia, S. Kumar, J. P. Singh, and K. Li, "The PARSEC benchmark suite: Characterization and architectural implications," in *PACT*, 2008, pp. 72–81.
[3] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, and D. A. Wood, "The gem5 simulator," *ACM SIGARCH Computer Architecture News*, vol. 39, no. 2, pp. 1–7, May 2011.
[4] J. Casazza, "First the tick, now the tock: Intel microarchitecture (nehalem)," Intel Corporation, White Paper, 2009.
[5] J. M. Cebrián, "Effcient power and thermal management using fine-grain architectural approaches in multicores," Ph.D. dissertation, University of Murcia, June 2011.
[6] J. Charles, P. Jassi, N. S. Ananth, A. Sadat, and A. Fedorova, "Evaluation of the intel core i7 turbo boost feature," in *IISWC*, 2009, pp. 188–197.
[7] G. Dantzig and M. Thapa, *Linear Programming 2: Theory and Extensions*. Springer-Verlag, 2003.
[8] T. Ebi, D. Kramer, W. Karl, and J. Henkel, "Economic learning for thermal-aware power budgeting in many-core architectures," in *CODES+ISSS*, 2011, pp. 189–196.
[9] T. Ebi, M. A. Al Faruque, and J. Henkel, "TAPE: Thermal-aware agent-based power economy for multi/many-core architectures," in *the International Conference on Computer-Aided Design (ICCAD)*, 2009, pp. 302–309.
[10] T. J. A. Eguia, S. X.-D. Tan, R. Shen, E. H. Pacheco, and M. Tirumala, "General behavioral thermal modeling and characterization for multi-core microprocessor design," in *Proceedings of the 18th Design, Automation and Test in Europe (DATE)*, 2010, pp. 1136–1141.
[11] H. Esmaeilzadeh, E. Blem, R. St.Amant, K. Sankaralingam, and D. Burger, "Dark silicon and the end of multicore scaling," in *ISCA*, 2011, pp. 365–376.
[12] A. Grenat, S. Pant, R. Rachala, and S. Naffziger, "5.6 adaptive clocking system for improved power efficiency in a 28nm x86-64 microprocessor," in *ISSCC*, 2014, pp. 106–107.
[13] Hardkernel Co., Ltd., "Odroid-XU3," www.hardkernel.com.
[14] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, and M. R. Stan, "HotSpot: A compact thermal modeling methodology for early-stage VLSI design," *IEEE Transactions on VLSI Systems*, vol. 14, no. 5, pp. 501–513, May 2006.

[15] Intel Corporation, “Dual-core intel xeon processor 5100 series datasheet, revision 003,” August 2007.

[16] ——, “Intel turbo boost technology in intel CoreTM microarchitecture (nehalem) based processors,” White Paper, November 2008.

[17] C. Isci and M. Martonosi, “Runtime power monitoring in high-end processors: Methodology and empirical data,” in *MICRO*, 2003, pp. 93–104.

[18] F. Kriebel, S. Rehman, D. Sun, M. Shafique, and J. Henkel, “ASER: Adaptive soft error resilience for reliability-heterogeneous processors in the dark silicon era,” in *DAC*, 2014, pp. 12:1–12:6.

[19] J. Lee and N. S. Kim, “Optimizing throughput of power- and thermal-constrained multicore processors using DVFS and per-core power-gating,” in *DAC*, 2009, pp. 47–50.

[20] S. Li, J.-H. Ahn, R. Strong, J. Brockman, D. Tullsen, and N. Jouppi, “McPAT: An integrated power, area, and timing modeling framework for multicore and manycore architectures,” in *MICRO*, 2009, pp. 469–480.

[21] T. Muthukaruppan, M. Pricopi, V. Venkataramani, T. Mitra, and S. Vishin, “Hierarchical power management for asymmetric multi-core in dark silicon era,” in *DAC*, 2013, pp. 174:1–174:9.

[22] S. Nussbaum, “AMD trinity APU,” in *Hot Chips*, 2012.

[23] M. Powell, A. Biswas, J. Emer, S. Mukherjee, B. Sheikh, and S. Yardi, “CAMP: A technique to estimate per-structure power at run-time using a few simple parameters,” in *HPCA*, 2009, pp. 289–300.

[24] B. Raghunathan and S. Garg, “Job arrival rate aware scheduling for asymmetric multi-core servers in the dark silicon era,” in *CODES+ISSS*, 2014.

[25] B. Raghunathan, Y. Turakhia, S. Garg, and D. Marculescu, “Cherry-picking: Exploiting process variations in dark-silicon homogeneous chip multi-processors,” in *DATE*, 2013, pp. 39–44.

[26] E. Rotem, A. Naveh, D. Rajwan, A. Ananthakrishnan, and E. Weissmann, “Power-management architecture of the intel microarchitecture code-named sandy bridge,” *IEEE Micro*, vol. 32, no. 2, pp. 20–27, March 2012.

[27] Samsung Electronics Co., Ltd., “Exynos 5 Octa (5422),” www.samsung.com/exynos.

[28] M. Shafique, S. Garg, J. Henkel, and D. Marculescu, “The EDA challenges in the dark silicon era: Temperature, reliability, and variability perspectives,” in *DAC*, 2014, pp. 185:1–185:6.

[29] C. Tan, T. Muthukaruppan, T. Mitra, and L. Ju, “Approximation-aware scheduling on heterogeneous multi-core architectures,” in *ASP-DAC*, Jan 2015, pp. 618–623.

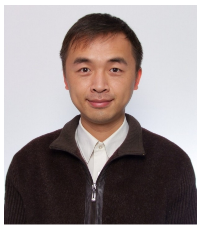
[30] W. Wu, L. Jin, J. Yang, P. Liu, and S. X.-D. Tan, “Efficient power modeling and software thermal sensing for runtime temperature monitoring,” *TODAES*, vol. 12, no. 3, pp. 25:1–25:29, May 2008.

Santiago Pagani is a Ph.D. student and part of the research staff at the Chair for Embedded Systems (CES) in Karlsruhe Institute of Technology (KIT) in Germany. He received his Diploma in Electronics Engineering from the Department of Electronics, National Technological University (UTN), Argentina in 2010. From 2003 until 2012, he worked as a hardware and software developer in the industry sector for several companies in Argentina. He joined KIT and started his doctoral research in March 2012. His research interests include embedded systems, real-time systems, energy-efficient scheduling, power-aware designs and temperature-aware scheduling. He received Best Paper Awards from IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA) in 2013, and from IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS) in 2014.

Heba Khdr is a Ph.D. student at the Chair for embedded Systems (CES) in Karlsruhe Institute of Technology (KIT) in Germany. She received her B.Sc in Informatics Engineering from University of Aleppo, Syria in 2005 with excellent grade and the first rank. From 2005 until 2007 worked as a software engineer in software company in Syria. From 2008 until 2010 she worked as an assistant in Aleppo university. In 2011 she did an equivalent master thesis at KIT and started her PhD in July 2011 at the Chair for Embedded System (CES). Her research interests are thermal management and resource management in many core systems. In 2012 she received Research Student Award from KIT. She received Best Paper Award from IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS) in 2014.

Jian-Jia Chen is a Professor in the Department of Informatics in TU Dortmund University in Germany. He was a Juniorprofessor in the Department of Informatics in Karlsruhe Institute of Technology (KIT) in Germany from May 2010 to March 2014. He received his Ph.D. degree from Department of Computer Science and Information Engineering, National Taiwan University, Taiwan in 2006. He received his B.S. degree from the Department of Chemistry at National Taiwan University 2001. Between Jan. 2008 and April 2010, he was a postdoc researcher at Computer Engineering and Networks Laboratory (TIK) in ETH Zurich, Switzerland. His research interests include real-time systems, embedded systems, energy-efficient scheduling, power-aware designs, temperature-aware scheduling, and distributed computing. He received Best Paper Awards from ACM Symposium on Applied Computing (SAC) in 2009, IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA) in 2005 and 2013, and IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS) in 2014.

Muhammad Shafique (M’11) received the Ph.D. degree in computer science from the Karlsruhe Institute of Technology (KIT), Germany, in 2011. He is currently a Research Group Leader at the Chair for Embedded Systems, KIT. He has over ten years of research and development experience in power-/performance-efficient embedded systems in leading industrial and research organizations. He holds one U.S. patent. His current research interests include design and architectures for embedded systems with focus on low power and reliability. Dr. Shafique was the recipient of 2015 ACM/SIGDA Outstanding New Faculty Award, six gold medals, the CODES+ISSS 2011 and 2014 Best Paper Awards, AHS 2011 Best Paper Award, DATE 2008 Best Paper Award, DAC 2014 Designer Track Poster Award, ICCAD 2010 Best Paper Nomination, several HiPEAC Paper Awards, and the Best Master Thesis Award. He is the TPC co-Chair of ESTIMedia 2015 and has served on the TPC of several IEEE/ACM conferences like ICCAD and DATE.



Minming Li received the BE and PhD degrees from the Department of Computer Science and Technology, Tsinghua University, Beijing, China, in 2002 and 2006, respectively. He is currently an assistant professor in the Department of Computer Science, City University of Hong Kong. His research interests include wireless ad hoc networks, algorithm design and analysis, and combinatorial optimization.

Jörg Henkel is currently with Karlsruhe Institute of Technology (KIT), Germany, where he is directing the Chair for Embedded Systems CES. Before, he was a Senior Research Staff Member at NEC Laboratories in Princeton, NJ. He received his PhD from Braunschweig University with "Summa cum Laude". Prof. Henkel has/is organizing various embedded systems and low power ACM/IEEE conferences/symposia as General Chair and Program Chair and was a Guest Editor on these topics in various Journals like the IEEE Computer Magazine. He was Program Chair of CODES'01, RSP'02, ISLPED'06, SIPS'08, CASES'09, Estimedia'11, VLSI Design'12, ICCAD'12, PATMOS'13, NOCS'14 and served as General Chair for CODES'02, ISLPED'09, Estimedia'12, ICCAD'13 and ESWeek'16. He is/has been a steering committee member of major conferences in the embedded systems field like at ICCAD, ESWeek, ISLPED, CODES+ISSS, CASES and is/has been an editorial board member of various journals like the IEEE TVLSI, IEEE TCAD, IEEE TMSCS, ACM TCPS, JOLPE etc. In recent years, Prof. Henkel has given around ten keynotes at various international conferences primarily with focus on embedded systems dependability. He has given full/half-day tutorials at leading conferences like DAC, ICCAD, DATE etc. Prof. Henkel received the 2008 DATE Best Paper Award, the 2009 IEEE/ACM William J. Mc Calla ICCAD Best Paper Award, the CODES+ISSS 2015, 2014, and 2011 Best Paper Awards, and the MaXentric Technologies AHS 2011 Best Paper Award as well as the DATE 2013 Best IP Award and the DAC 2014 Designer Track Best Poster Award. He is the Chairman of the IEEE Computer Society, Germany Section, and was the Editor-in-Chief of the ACM Transactions on Embedded Computing Systems (ACM TECS) for two consecutive terms. He is an initiator and the coordinator of the German Research Foundation's (DFG) program on 'Dependable Embedded Systems' (SPP 1500). He is the site coordinator (Karlsruhe site) of the Three- University Collaborative Research Center on "Invasive Computing" (DFG TR89). He is the Editor-in-Chief of the IEEE Design & Test Magazine since January 2016. He holds ten US patent and is a Fellow of the IEEE.

Appendix A
Summary of Symbols

For simple reference, Table 2 and Table 3 summarize all the symbols used in the paper.

Appendix B
TSP for a Given Mapping: Homogeneous (Different Power Constraint per Core)

This section presents a solution to compute TSP for a particular core mapping and ambient temperature, which results in different TSP values per core. That is, cores are constrained to different power values, depending on their location and the adjacent active cores. The objective here is to maximize the total power consumption.

For a given $\mathbf{Q}$, $\mathbf{P}^{\text{blocks}}$, T_{amb}, T_{DTM}, $P^{\text{core}}_{\text{inact}}$, $P_{\max}$, and floorplan, the TSP computation for this case can be formulated as a linear programming. Note that this is not the main contribution of our paper, and we only present this solution for completeness. This formulation is expressed as

$$\text{Maximize} \quad \sum_{\forall i \in \mathbf{K}} p_i^{\text{cores}}$$

such that:

$$\begin{aligned}
&\mathbf{B}\mathbf{T} - \mathbf{P}^{\text{cores}} = \mathbf{P}^{\text{blocks}} + T_{\text{amb}}\mathbf{G} \\
&\textstyle\sum_{i=1}^{N} p_i^{\text{cores}} \leq P_{\max} - \sum_{i=1}^{N} p_i^{\text{blocks}} \\
&T_i \leq T_{\text{DTM}} && \text{for all } i \in \mathbf{L} \\
&T_i \geq 0 && \text{for } i = 1, \ldots, N \\
&p_i^{\text{cores}} = P^{\text{core}}_{\text{inact}} && \text{for } i = 1, \ldots, M \text{ if } q_i = 0 \\
&p_i^{\text{cores}} \geq P^{\text{core}}_{\text{min}} && \text{for } i = 1, \ldots, M \text{ if } q_i = 1.
\end{aligned}$$

Any standard method, e.g., Simplex or an Interior-Point method [7], can be used to solve this programming, resulting in a vector with a different power constraint for each active core in vector $\mathbf{Q}$, which we define as $\mathbf{P}^{\text{cores}}_{\text{TSP}}(\mathbf{Q})$.

Due to the heat transfer among cores, for the same number of active cores different mappings result in different power constraints. This can be seen in the following example. Consider a manycore system of 16 cores with the same settings as in the motivational example from Section 1, with $P^{\text{core}}_{\text{inact}}$ of 0 W and $P^{\text{core}}_{\text{min}}$ of 0.5 W. For a T_{amb} of 45°C and two different mappings $\mathbf{Q}$, both with 4 active cores, we compute $\mathbf{P}^{\text{cores}}_{\text{TSP}}(\mathbf{Q})$ and present the results in Fig. 12a and Fig. 12b. The maximum total power consumption for the mapping in Fig. 12a and Fig. 12b is 62.2 W and 67.4 W, respectively.

Appendix C
Transient State Considerations

Depending on the executed applications and the task partitioning, mapping, and DVFS policies implemented in the system, the power consumption and the number of active cores throughout the chip could change very frequently (in the order of milliseconds) or it could change rarely (in the order of several seconds). The former will happen very often in normal systems, e.g., when there are context switches inside different cores due to task preemption, when some cores become idle waiting for data from memory or waiting for some other thread to finish its computation before being able to continue, etc. The latter could occur in scenarios with long running application that have no more running threads than cores (such that preemption is not needed). In either case, when drastic power changes occur and depending on the adopted DVFS policy, the temperature of some cores may exceed the value of T_{DTM} due to the effects of transient temperatures. When this happens, DTM is triggered to avoid damages to the chip, resulting in a lower performance than expected. *Unless care is taken, these transient effects can be observed in any power budgeting technique derived for the steady-state temperatures, e.g., for constant power constraints like TDP, and also for some cases with TSP.*

The following example uses HotSpot [14] with its default configuration (detailed in Section 7.1) to show how this effect could possibly occur for systems constrained both by

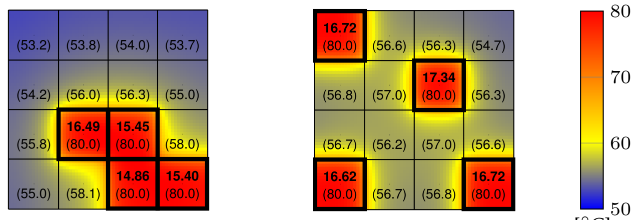
Symbol	Description
M	Total number of cores in the system
Z	Total number of blocks in the floorplan, such that $Z - M$ is the amount of blocks that correspond to components that are not cores
N	Total number of nodes in RC thermal network
T_{amb}	Ambient temperature
T_{DTM}	Threshold temperature that triggers DTM
$P^{\text{core}}_{\text{min}}$	In a homogeneous system, the minimum power consumption to maintain a core in the active state (at its lower speed)
$P^{\text{core}}_{\text{inact}}$	In a homogeneous system, the power consumption of an inactive core
$P_{\max}$	Maximum sustainable chip power consumption. An electrical constraint and not an abstraction
$\mathbf{A} = [a_{i,j}]_{N\times N}$	Matrix containing the thermal capacitance values of the RC thermal network
$\mathbf{B} = [b_{i,j}]_{N\times N}$	Matrix containing the thermal conductance values of the RC thermal network
$\mathbf{B}^{-1} = [b^{-1}{}_{i,j}]_{N\times N}$	Inverse of matrix $\mathbf{B}$
$\mathbf{T} = [T_i]_{N\times 1}$	Column vector representing the temperature on each node of the RC thermal network
$\mathbf{T}' = [T'_i]_{N\times 1}$	Column vector accounting for the first order derivative of the temperature on each node of the RC thermal network with respect to time
$\mathbf{P} = [p_i]_{N\times 1}$	Column vector containing the power consumption on each node of the RC thermal network. It holds that $\mathbf{P} = \mathbf{P}^{\text{cores}} + \mathbf{P}^{\text{blocks}} + \mathbf{P}^{\text{int}}$
$\mathbf{P}^{\text{cores}} = [p_i^{\text{cores}}]_{N\times 1}$	Sub-vector containing the power consumption on the cores. Elements of other nodes are set to 0
$\mathbf{P}^{\text{blocks}} = [p_i^{\text{blocks}}]_{N\times 1}$	Sub-vector containing the power consumption on blocks of the floorplan that are not cores. Elements of other thermal nodes are set to 0
$\mathbf{P}^{\text{int}} = [p_i^{\text{int}}]_{N\times 1}$	Sub-vector containing the power consumption on internal nodes, such that $p_i^{\text{int}} = 0$ for all i
$\mathbf{G} = [g_i]_{N\times 1}$	Column vector containing the thermal conductance between each node of the RC thermal network and T_{amb}. If node i is not in contact with the ambient g_i is set to zero
$\mathbf{L} = \{\ell_1, \ldots, \ell_Z\}$	Set containing all indexes of thermal nodes that correspond to blocks of the floorplan
$\mathbf{K} = \{k_1, \ldots, k_M\}$	Set containing all indexes of thermal nodes that correspond to cores
$\mathbf{Q} = [q_i]_{M\times 1}$	Binary vector for a particular mapping of *active* cores: $q_i = 1$ means that core i is *active*; $q_i = 0$ means that core i is *inactive*
$P_{\text{TSP}}(\mathbf{Q})$	For homogeneous systems, a power constraint for each active core in mapping $\mathbf{Q}$ that results in a maximum temperature (in the steady-state) among all cores which does not exceed T_{DTM}
$P^{\star}_{\text{TSP}}(\mathbf{Q})$	Equivalent to $P_{\text{TSP}}(\mathbf{Q})$ but ignoring the maximum chip power $P_{\max}$
$P^{\text{worst}}_{\text{TSP}}(m)$	For homogeneous systems, a power constraint for each active core in *any* possible core mapping with m simultaneously active cores that results in a maximum temperature (in the steady-state) among all cores which does not exceed T_{DTM}

TABLE 2: Table listing all symbols.

Symbol	Description
$P_{\text{TSP}}^{\star\text{worst}}(m)$	Equivalent to $P_{\text{TSP}}^{\text{worst}}(m)$ but ignoring the maximum chip power $P_{\max}$
$\mathbf{H} = [h_{i,j}]_{Z \times M}$	Auxiliary matrix used in Lemma 4 to compute the maximum amount of heat that any m cores can contribute to the temperature on node i
$\text{R}(m)$	Auxiliary function that helps guarantee that $P_{\text{TSP}}(\mathbf{Q})$ and $P_{\text{TSP}}^{\text{worst}}(m)$ do not exceed $P_{\max}$
W	For heterogeneous systems, total number of types of cores
M_w	For heterogeneous systems, total number of cores of type w
$\text{area}_j^{\text{core}}$	For heterogeneous systems, the area of core j, for $j = 1, 2, \ldots, M$
$\text{area}_w^{\text{type}}$	For heterogeneous systems, the area of cores of type w, for $w = 1, 2, \ldots, W$
$p_{\text{inact}\,j}^{\text{core}}$	For heterogeneous systems, the inactive power consumption of core j, for $j = 1, 2, \ldots, M$
$p_{\text{inact}\,w}^{\text{type}}$	For heterogeneous systems, the inactive power consumption of cores of type w, for $w = 1, 2 \ldots, W$
$\mathbf{K}^w = \left\{k_1^w, \ldots, k_{M_w}^w\right\}$	For heterogeneous systems, set containing all indexes of thermal nodes that correspond to cores of type w
$\mathbf{Q}^w = \left[q_i^w\right]_{M_w \times 1}$	Binary vector for a particular mapping of *active* cores of type w: $q_i^w = 1$ means that core i is an *active* core of type w; $q_i^w = 0$ means that core i is an *inactive* core of type w
$P_{\text{TSP}}^{\rho}(\mathbf{Q})$	For heterogeneous systems, a uniform power density constraint for each active core in mapping $\mathbf{Q}$ (independent of the core types) that results in a maximum temperature (in the steady-state) among all cores which does not exceed T_{DTM}
$P_{\text{TSP}}^{\rho\star}(\mathbf{Q})$	Equivalent to $P_{\text{TSP}}^{\rho\star}(\mathbf{Q})$ but ignoring the maximum chip power $P_{\max}$
$\text{R}^{\rho}(\mathbf{Q})$	Auxiliary function that helps guarantee that $P_{\text{TSP}}^{\rho}(\mathbf{Q})$ does not exceed $P_{\max}$
$\mathbf{m} = \{m_1, \ldots, m_W\}$	For heterogeneous systems, set representing the number of active cores for core types $\{1, \ldots, W\}$
$\mathbf{H}^{\rho} = \left[h_{w,i,j}^{\rho}\right]_{W \times Z \times M_w}$	Auxiliary matrix used in Lemma 8 to compute the maximum amount of heat that any m_w cores of type w can contribute to the temperature on node i, for all core types $w = 1, 2, \ldots, W$
$P_{\text{TSP}}^{\rho\,\text{worst}}(\mathbf{m})$	For heterogeneous systems, a power density constraint (independent of the core types) for each active core in *any* possible core mapping with $\mathbf{m}$ simultaneously active cores that results in a maximum temperature (in the steady-state) among all cores which does not exceed T_{DTM}
$P_{\text{TSP}}^{\rho\star\text{worst}}(\mathbf{m})$	Equivalent to $P_{\text{TSP}}^{\rho\,\text{worst}}(\mathbf{m})$ but ignoring the maximum chip power $P_{\max}$
$\text{R}^{\rho}(\mathbf{m})$	Auxiliary function that helps guarantee that $P_{\text{TSP}}^{\rho\,\text{worst}}(\mathbf{m})$ does not exceed $P_{\max}$

TABLE 3: Table listing all symbols (continued).

(1) TDP and (2) TSP. The same example applies for both cases. Consider a manycore system of 16 cores with the same settings as in the motivational example from Section 1, with $P_{\text{inact}}^{\text{core}}$ of 0 W, and (1) TDP of 90.2 W per-chip, or (2) $P_{\text{TSP}}^{\text{worst}}(4) = 14.67\,\text{W}$ and $P_{\text{TSP}}^{\text{worst}}(8) = 11.27\,\text{W}$, i.e., TSP of 14.67 W and 11.27 W per-core when activating 4 and 8 cores, respectively. Under these assumptions, Fig. 13 presents simulations in which during $t = [0s, 0.5s]$ there are 8 active cores according to Fig. 1b, consuming 11.27 W each (90.2 W in total). During $t = [0.5\,\text{s}, 1\,\text{s}]$, these cores are shut-down and we activate 4 cores according to Fig. 1a, consuming 14.67 W each (58.7 W in total). That is, if constrained by (1) TDP, the system consumes TDP during $t = [0\,\text{s}, 0.5\,\text{s}]$, and less than TDP during $t = [0.5\,\text{s}, 1\,\text{s}]$. Moreover, if constrained by (2) TSP, the system consumes the corresponding



(a) Total power consumption of 62.2 W

(b) Total power consumption of 67.4 W

Fig. 12: Example of TSP with different power constraints. Top numbers are the power consumptions (in Watts) of each active core (boxed in black). Bottom numbers in parenthesis are the temperatures in the center of each core (in °C). Detailed temperatures are shown according to the color bar.

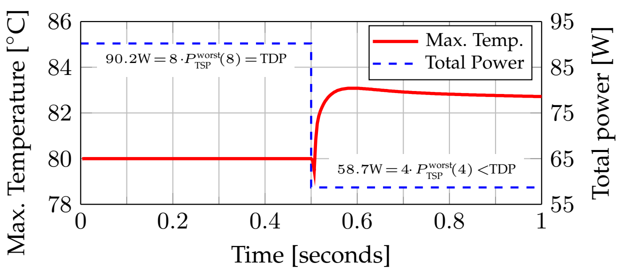


Fig. 13: Transient example for both TSP and TDP (the red line shows the maximum temperature among all cores). During $t = [0\,\text{s}, 0.5\,\text{s}]$ there are 8 active cores according to Fig. 1b, consuming 11.27 W each. During $t = [0.5\,\text{s}, 1\,\text{s}]$, these cores are shut-down and we activate 4 cores according to Fig. 1a, consuming 14.67 W each.

TSP according to the number of active cores all the time. Nevertheless, although for both cases the power budgets are not violated, Fig. 13 shows that during $t = [0.5\,\text{s}, 1\,\text{s}]$ the temperature of at least one core would exceed the 80°C threshold for triggering DTM. *This transient effect in which we have transient temperature peaks that are higher than the corresponding steady-state temperatures normally occurs when the power density of the cores is increased during a change in power. For example, for the opposite case in which we transition from the mapping in Fig. 1a to that in Fig. 1b, since the power density in the active cores decreases, the transient temperatures would remain below both steady-states, as seen in Fig. 14.*

If the frequency of the changes in power that produce this effect is very high, then DTM could be triggered frequently, and the associated performance losses would be noticeable. There are several ways to deal with this issue in regards to TSP, and here we detail two of them. Specifically, we can adjust the temperature for which we compute TSP such that the transient peak temperatures remain below T_{DTM} even if we always adjust the DVFS levels to consume the entire TSP budget according to the number of active cores (Appendix C.1), or the DVFS levels can be kept at nominal operation for a given mapping by ignoring the partial number of active cores due to cores being idle while waiting for data from memory or waiting for other threads to finish (Appendix C.2).

C.1 Adjusting the Temperature for Computing TSP

One strategy for dealing with transient peak temperatures is to quantify the maximum values that these temperatures can actually reach during the transient state. The difference

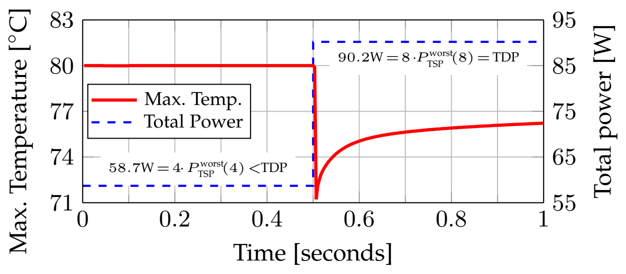
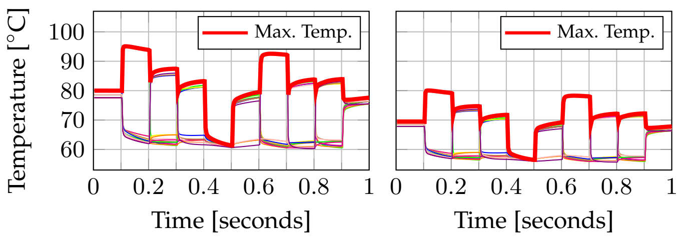


Fig. 14: Transient example for both TSP and TDP (the red line shows the maximum temperature among all cores). During $t = [0\,\mathrm{s}, 0.5\,\mathrm{s}]$ there are 4 active cores according to Fig. 1a, consuming $14.67\,\mathrm{W}$ each. During $t = [0.5\,\mathrm{s}, 1\,\mathrm{s}]$, these cores are shut-down and we activate 8 cores according to Fig. 1b, consuming $11.27\,\mathrm{W}$ each.



(a) TSP for 80°C (b) TSP for 69.5°C

Fig. 15: Transient example for 16 cores. The number of active cores and their power consumption changes every 0.1 seconds. The mapping and power values adopted are those of worst-case TSP computed for (a) 80°C and (b) 69.5°C.

between such maximum transient temperatures and T_{DTM} is denoted as $\Delta T^{\max}_{\mathrm{transient}}$, with value 3.08°C for the example in Fig. 13. Therefore, by computing TSP for $T_{\mathrm{DTM}} - \Delta T^{\max}_{\mathrm{transient}}$, we can make sure that the transient temperatures never reach T_{DTM}. Nevertheless, depending on the floorplan and the resulting thermal capacitances, it can happen that the transient temperatures for this new TSP are too pessimistic compared to T_{DTM}. For such cases, with just a few iterations, a near optimal value for which to compute TSP can be achieved. This method should be applied offline, due to the overheads for obtaining $\Delta T^{\max}_{\mathrm{transient}}$ for each case. *A similar method should be adopted for systems that use constant power budgets, e.g., TDP.*

Procedure Example: Consider a hypothetical manycore system of 16 cores with the same settings as in the motivational example from Section 1, and an ambient temperature of 45°C. For the power consumptions, consider a hypothetical scenario in which the power of the cores changes every 0.1 seconds. The changes in power are for a random number of active cores. The mapping and power values adopted in all cases are those of TSP for the worst-case, computed through Algorithm 2, according to the number of active cores at each time instant. Under these assumptions, we run simulations with HotSpot for such a case, and present what the associated temperatures would look like in Fig. 15a. In Fig. 15a, the temperature on each core is presented with a different color, and the maximum temperature among all cores at any given time is highlighted by the bold-red curve. We can observe that for these assumptions and floorplan the thermal capacitances are not negligible, which results in long transient state periods. Therefore, for such a case, TSP should be recomputed for some temperature below T_{DTM}.

Looking at Fig. 15a, we can quantify $\Delta T^{\max}_{\mathrm{transient}}$, which we set to 15°C, and then recompute TSP for 65°C. Given that this results in a maximum transient temperature of 74°C, which is too pessimistic, we need a higher value. Thus, we iterate computing TSP and running transient temperature simulations. After just 5 iterations, specifically, computing TSP for 80°C, 65°C, 71°C, 69°C, and 69.5°C, we reach a near optimal value for which to compute TSP, which for this floorplan is 69.5°C. Clearly, the new TSP values are smaller than those for a TSP computed for 80°C.

Finally, we run the transient simulations with the same settings and assumptions, but with power states according to the new TSP values. The results are presented in Fig. 15b, which shows that when computing TSP for 69.5°C, the temperature would always be below T_{DTM}. When computing TSP online for particular mapping scenarios, the target temperature should also be 69.5°C, and not the original 80°C.

If, *unlike* Fig. 15, the changes in power are *not* too frequent, such that DTM is triggered with low frequency during short time intervals, then computing TSP for T_{DTM} could still a better approach that results in higher total performance.

C.2 Nominal DVFS Operation for a Given Mapping

Another strategy for dealing with transient peak temperatures is to keep the DVFS levels at nominal operation for a given mapping by ignoring the partial number of active cores due to cores being idle while waiting for data from memory or waiting for other threads to finish. For example, considering the experimental setup for the homogeneous experiments in Section 7, assume a situation in which the operating system partitions tasks and maps them to cores such that 32 cores have threads assigned to them and the other 32 cores are power gated. For such a case, the DVFS levels of the 32 active cores can be set such that the power consumption in each active core is below the TSP power budget, which is $5.86\,\mathrm{W}$ if we consider the worst-case TSP value as reported in Fig. 7 and Table 4 (we could have also computed TSP online for the specific mapping, but we use the worst-case TSP values for simplicity of presentation). Naturally, there will be time intervals (which could last some milliseconds or entire seconds) during which some of these 32 active cores will actually remain idle, e.g., when a thread is locked waiting for data from another thread to continue. When this happens, there are two possible alternatives of how the system could operate: (1) change the DVFS levels in order to satisfy TSP for the partial number of active cores, or (2) keep the same DVFS levels that satisfy TSP when there are 32 active cores (e.g., $5.86\,\mathrm{W}$ for the worst-case TSP).

It is important to remember at this point that TSP is a decreasing function with respect to the number of active cores, as shown in Fig. 7 and Table 4. Therefore, since TSP for less than 32 active cores would be larger than $5.86\,\mathrm{W}$ and thus cores can execute at faster frequencies while satisfying TSP, this means that operating under alternative (1) we can potentially achieve a higher system performance by dynamically changing the DVFS levels. However, this precisely the case in which we can many transient thermal violations as explained above.

Contrarily, operating under alternative (2) is a conservative approach which needs little further considerations. Namely, since TSP for less than 32 active cores is larger than

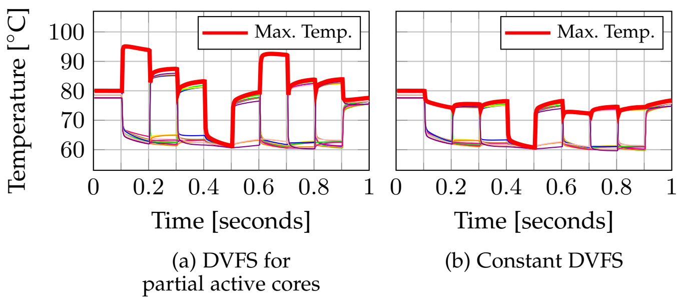


Fig. 16: Transient example for 16 active cores when using different DVFS policies for idling. The partial number of active cores changes every 0.1 seconds, e.g., when threads become idle waiting for other threads to finish. In (a), DVFS is used to match the worst-case TSP for the partial number of active cores in each time interval. In (b), DVFS levels are kept constant to match the worst-case TSP when no core is idle, i.e., for 16 active cores.

5.86 W, if the DVFS levels are not changed then individual cores will not consume more than 5.86 W and this is safe for any scenario with less than 32 active cores. Furthermore, since the power density of the active cores remains constant under alternative (2), in this scenario the transient temperatures will generally remain under T_{DTM} and thus DTM will not be triggered. This statement is supported by Fig. 16. Fig. 16a shows the same simulations as in Fig. 15a, in which the DVFS levels are constantly changing to match the TSP values for the partial number of active cores in each time interval. On other hand, Fig. 16b shows what the resulting temperatures would be for the same assumptions and scenario, but keeping the DVFS levels constant at nominal operation, were it can be observed that all transient temperatures would be below T_{DTM}. *Finally, note that under alternative (2) the DVFS levels are not kept always constant, but are rather set to nominal values. That is, when an application finishes execution or a new application arrives, the mapping of threads changes and new nominal DVFS levels to satisfy TSP are computed for the new mapping. Alternative (2) simply keeps the DVFS levels constant when the partial number of active cores changes due to core idling or similar situations.*

APPENDIX D
ADDITIONAL EXPERIMENTAL SETUP DETAILS

In this section we discuss additional details of our experimental setup and simulation framework, for which Fig. 17 presents an overview. Our experimental evaluations are conducted considering detailed transient temperatures, such that we can conduct accurate full system simulations. Specifically, our simulation framework integrates gem5, McPAT, real measurements on the Odroid-XU3 platform, and HotSpot in a closed-loop transient simulator. First, the PARSEC applications are executed in gem5 for some specific frequency configuration. We parse the output statistics from gem5 in order to generate the xml files required as inputs by McPAT, adding at this point the voltage information (not required before by gem5). The output results from McPAT are then also parsed in order to obtain the necessary power consumption data for every part of the chip. For the Odroid-XU3 platform, we simply execute the PARSEC applications for different frequency configurations and measure the execution time and power consumption.

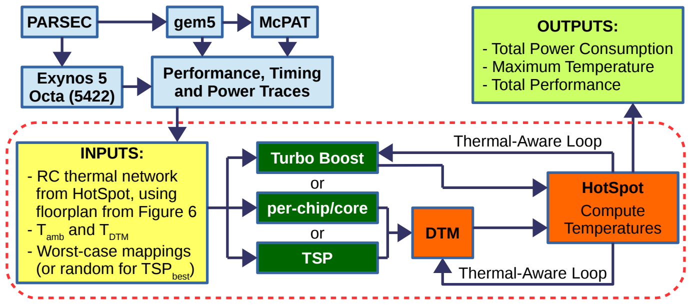

PARSEC
gem5
McPAT
Exynos 5 Octa (5422)
Performance, Timing and Power Traces
OUTPUTS:
- Total Power Consumption
- Maximum Temperature
- Total Performance
INPUTS:
- RC thermal network from HotSpot, using floorplan from Figure 6
- T_{amb} and T_{DTM}
- Worst-case mappings (or random for TSP_{best})
Turbo Boost
or
per-chip/core
or
TSP
DTM
HotSpot
Compute Temperatures
Thermal-Aware Loop
Thermal-Aware Loop

Fig. 17: Overview of our simulation framework.

For example, Fig. 18 and Fig. 19 show execution time and average power consumption values for applications from the PARSEC benchmark suite, where it can be clearly observed that different applications have different power consumptions, and the specific power values depend on the application, the voltage/frequency, and the selected number of threads (not shown in the figure). All of this power data is then fed to HotSpot in order to conduct the transient thermal simulations.

For each floorplan discussed in Section 7.1 and Section 8.1, we consider one thermal block for each core and independent thermal blocks for the L2 caches and other hardware. We then obtain the values for $\mathbf{B}$, $\mathbf{B}^{-1}$, and $\mathbf{G}$, by using HotSpot [14] v5.02 with its default configuration in the block model mode. That is, chip thickness of 0.15 mm, silicon thermal conductivity of 100 $\frac{\text{W}}{\text{m}\cdot\text{K}}$, silicon specific heat of $1.75 \cdot 10^6 \frac{\text{J}}{\text{m}^3\cdot\text{K}}$, a heat sink of 6×6 cm and 6.9 mm thick, heat sink convection capacitance of 140.4 $\frac{\text{J}}{\text{K}}$, heat sink convection resistance of 0.1 $\frac{\text{K}}{\text{W}}$, heat sink and heat spreader thermal conductivity of 400 $\frac{\text{W}}{\text{m}\cdot\text{K}}$, heat sink and heat spreader specific heat of $3.55 \cdot 10^6 \frac{\text{J}}{\text{m}^3\cdot\text{K}}$, a heat spreader of 3×3 cm and 1 mm thick, interface material thickness of 20 um, interface material thermal conductivity of 4 $\frac{\text{W}}{\text{m}\cdot\text{K}}$, and interface material specific heat of $4 \cdot 10^6 \frac{\text{J}}{\text{m}^3\cdot\text{K}}$.

When evaluating TSP and the per-chip/per-core power budgets, if the temperature anywhere in the chip exceeds the predefined threshold, then DTM is triggered, operating as described in Section 7.1. The voltage/frequency reduction or increment is achieved by changing the voltage and frequency in gem5 and McPAT or in the Odroid-XU3 platform. These changes in the voltage and frequency translate to different execution times for each application thread and also on different power consumption values. Thanks to our closed-loop simulations, these new power consumption data is fed to HotSpot for the next simulation steps, thus changing the resulting transient temperature.

When evaluating Turbo Boost, the transient simulations are conducted in a similar fashion, only that now Turbo Boost will always attempt to increase the voltage/frequency whenever the temperature is below the threshold, as already explained in Section 7.1. In our simulation framework, the voltage/frequency reduction or increment is achieved in the same way as done by the DTM technique. Finally, the performance results presented in Fig. 10 and Fig. 11 show *average* total IPS count and the *average* total throughput, respectively.

As an additional comment with respect to the considered DTM technique, note that although in Fig. 9 we are only plotting the maximum temperature among all cores, there are many cores that reach temperatures above the threshold temperature that triggers DTM. This point is very rele-

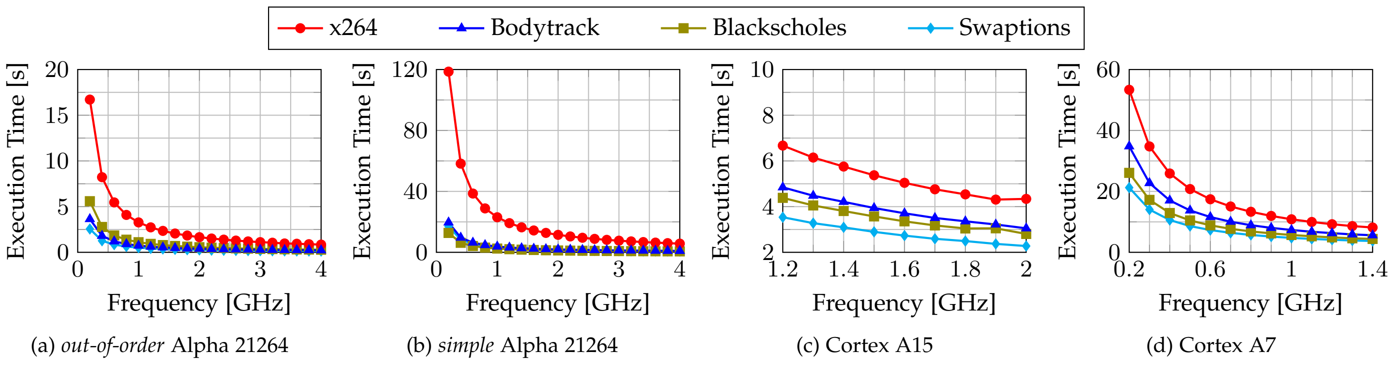


Fig. 18: Execution time based on simulations in gem5 [3], and measured on the Exynos 5 Octa (5422) processor, for some applications from the PARSEC benchmark suite [2].

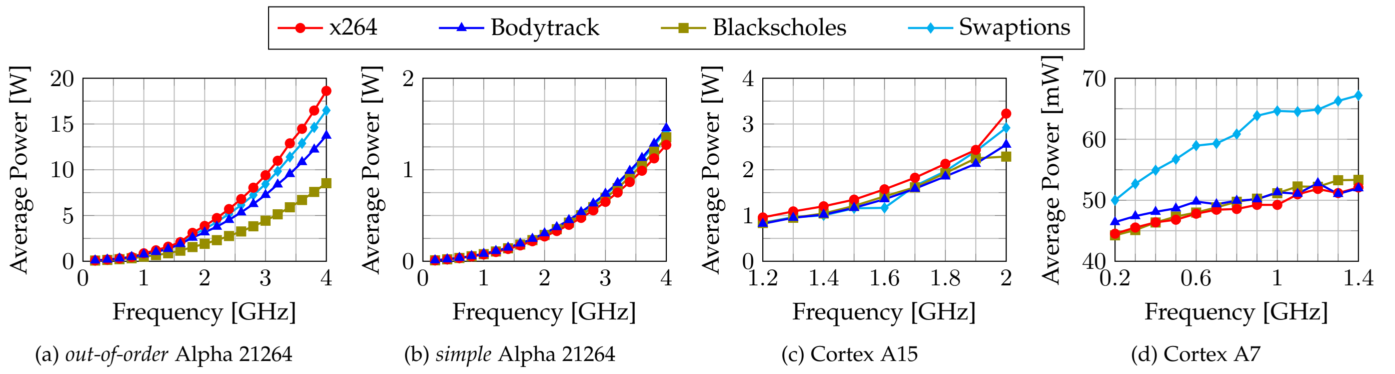


Fig. 19: Average power values based on simulations in gem5 [3] and McPAT [20] (for 22 nm), and measured on the Exynos 5 Octa (5422) processor, for some applications from the PARSEC benchmark suite [2].

vant from the perspective of DTM. Namely, for particular mappings, only a few cores might exceed the threshold temperature and thus reducing the voltage/frequency of all active cores is not the most efficient DTM approach. This is most common in real systems executing applications that consume different amounts of power, or when different cores are configured to run at different DVFS levels. However, conducting such experiments would make for a very hard interpretation of the results. Because of this reason, we choose to conduct more uniform evaluations, with applications of the same type and all cores of the same type running at similar frequencies. In this way, when there is a thermal violation that triggers DTM, all (or almost all) active cores have temperatures above the allowed threshold. For example, Fig. 20 presents a thermal snapshot for the worst-case mapping when activating 12 cores and equally distributing the 225 W per-chip power budget among the active cores. As seen in the figure, for such a case the highest temperature among all cores is 102.9°C. Nevertheless, the lowest temperature among the active cores is also much higher than 80.0°C, specifically, it reaches 97.0°C. Therefore, in our experiments we can conservatively assume that DTM is triggered for all active cores without incurring in much pessimism.

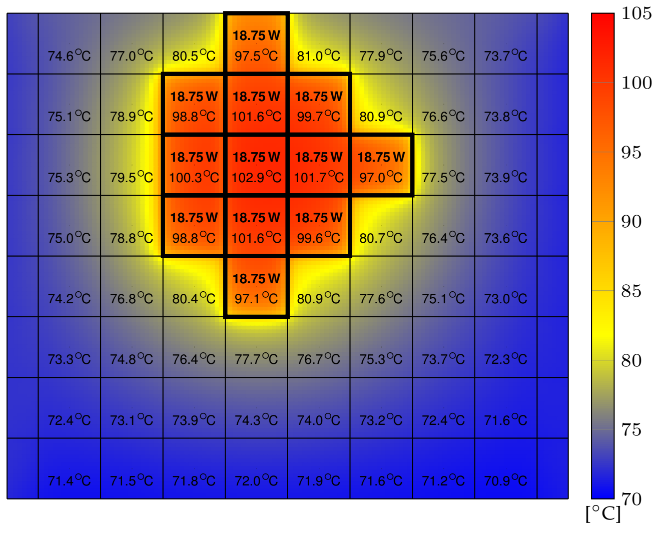


Fig. 20: Temperature snapshot for 12 active cores when equally distributing the 225 W per-chip power budget, with a resulting highest temperature of 102.9°C.

Appendix E
Power Constraints for Homogeneous Evaluations

In this section we include table representations of Fig. 7, Fig. 8, and Fig. 9.

Power Constraint per core [W]						
Active Cores	TSP_{worst}	TSP_{best}	3.52 W per-core	80 W per-chip	150 W per-chip	225 W per-chip
4	19.95	24.79	3.52	20.00	37.50	56.25
8	14.09	18.24	3.52	10.00	18.75	28.13
12	11.16	14.47	3.52	6.67	12.50	18.75
16	9.34	11.88	3.52	5.00	9.38	14.06
20	8.06	10.11	3.52	4.00	7.50	11.25
24	7.13	8.81	3.52	3.33	6.25	9.38
28	6.42	7.81	3.52	2.86	5.36	8.04
32	5.86	6.97	3.52	2.50	4.69	7.03
36	5.39	6.31	3.52	2.22	4.17	6.25
40	5.00	5.74	3.52	2.00	3.75	5.63
44	4.67	5.25	3.52	1.82	3.41	5.11
48	4.39	4.85	3.52	1.67	3.13	4.69
52	4.14	4.50	3.52	1.54	2.88	4.33
56	3.93	4.16	3.52	1.43	2.68	4.02
60	3.73	3.87	3.52	1.33	2.50	3.75
64	3.52	3.52	3.52	1.25	2.34	3.52

TABLE 4: Worst- and best-case TSP for the floorplan in Fig. 3, compared to a constant per-core budget, and estimations of constant per-chip budgets equally distributed among active cores.

Power Constraint per chip [W]						
Active Cores	TSP_{worst}	TSP_{best}	3.52 W per-core	80 W per-chip	150 W per-chip	225 W per-chip
4	79.80	99.17	14.08	80.00	150.00	225.00
8	112.76	145.94	28.16	80.00	150.00	225.00
12	133.91	173.63	42.24	80.00	150.00	225.00
16	149.45	190.06	56.32	80.00	150.00	225.00
20	161.14	202.10	70.40	80.00	150.00	225.00
24	171.18	211.35	84.48	80.00	150.00	225.00
28	179.86	218.80	98.56	80.00	150.00	225.00
32	187.43	223.00	112.64	80.00	150.00	225.00
36	194.02	227.04	126.72	80.00	150.00	225.00
40	199.96	229.57	140.80	80.00	150.00	225.00
44	205.36	231.21	154.88	80.00	150.00	225.00
48	210.51	232.91	168.96	80.00	150.00	225.00
52	215.33	233.80	183.04	80.00	150.00	225.00
56	219.82	233.13	197.12	80.00	150.00	225.00
60	224.08	232.10	211.20	80.00	150.00	225.00
64	225.28	225.28	225.28	80.00	150.00	225.00

TABLE 5: Constant per-chip budgets, compared to multiplying the number of active cores by a constant per-core budget, and the worst- and best-case TSP for the floorplan from Fig. 3.

Maximum Temperature [°C]						
Active Cores	TSP_{worst}	TSP_{best}	3.52 W per-core	80 W per-chip	150 W per-chip	225 W per-chip
4	80.00	73.52	52.38	80.10	80.10	80.10
8	80.00	80.00	54.81	70.25	91.10	94.08
12	80.00	80.00	56.97	66.47	84.05	102.89
16	80.00	80.00	58.98	64.34	80.13	97.05
20	80.00	80.00	60.96	62.99	77.67	93.40
24	80.00	80.00	62.83	61.96	75.82	90.66
28	80.00	80.00	64.64	61.16	74.37	88.53
32	80.00	80.00	66.41	60.51	73.21	86.83
36	80.00	80.00	68.16	59.98	72.27	85.45
40	80.00	80.00	69.87	59.52	71.47	84.28
44	80.00	80.00	71.56	59.12	70.78	83.28
48	80.00	80.00	73.20	58.75	70.15	82.36
52	80.00	80.00	74.82	58.42	69.58	81.55
56	80.00	80.00	76.41	58.11	69.07	80.82
60	80.00	80.00	77.98	57.83	68.60	80.15
64	80.00	80.00	79.51	57.57	68.16	79.51

TABLE 6: Maximum steady-state temperatures among all blocks (DTM deactivated), when using TSP, a constant per-core budget, and equally distributed constant per-chip budgets.