Smart Thermal Management for Heterogeneous Multicores

Jörg Henkel, Heba Khdr, Martin Rapp
Chair for Embedded Systems (CES), Karlsruhe Institute of Technology (KIT), Germany
{henkel, heba.khdr, martin.rapp}@kit.edu

(Invited Paper)

***Abstract*—Heterogeneous multicores have attracted a major focus in recent years, as they provide many possibilities for performance improvements. However, due to the discontinuation of Dennard scaling, on-chip power densities are continuously increasing along with technology scaling, and hence on-chip temperatures are elevated. Therefore, several thermal management techniques have emerged to keep the temperature of the chip within safe limits. These techniques, however, lead to performance losses which ultimately erase a big portion of the expected performance gains from the heterogeneous multicores. Thus, it is indispensable to deploy thermal management techniques that are able to make efficient decisions which satisfy temperature constraints while at the same time maximizing the performance. This paper presents smart thermal management techniques for heterogeneous multicores that exploit relevant information about several heterogeneity parameters at the chip level and at the application level to increase thermal efficiency[1]. Compared to the state of the art, the presented techniques are able to obtain significant performance improvements under the same thermal constraint.**

This paper is part of the DATE 2019 special session on ”Smart Resource Management and Design Space Exploration for Heterogeneous Processors”. The other two papers of this special session are [1] and [2].

***Index Terms*—Thermal Management, Thermal Efficiency, Power Density, Heterogeneous Multicores, Performance Optimization**

I. Introduction

Driven by the ever increasing performance demands in various application domains such as image processing, deep learning and scientific computing, a large body of research focuses on heterogeneous multicores, where a combination of processing elements with different compute capabilities are employed [3]. Heterogeneous multicores [4] can integrate application-specific accelerators, graphics processing units, and general-purpose cores with different voltage and frequency (V/f) levels for each, and thereby providing many possibilities for performance and power/energy efficiency [5], [6]. A common practice for designing such systems is to employ a tiled architecture, where processing elements inside a tile share the same voltage and frequency, but different tiles can execute at different voltage and frequencies at any given time point [7]. An example of such an architecture from commercial chips is the Exynos 5 Octa (5422) chip [8] with ARMs big.LITTLE architecture composed of three tiles: a quad-core Cortex-A7, a quad-core Cortex-A15, and a T-628 GPU.

Although heterogeneous multicores provide many compute capabilities to improve application performance, these capabilities might not be available simultaneously all the time due to the temperature wall [9], [10]. In particular, along with technology scaling, on-chip power densities are continuously increasing and thus on-chip temperature is elevated. Elevated temperature has several negative impacts on chip reliability [11]. Therefore, to keep the chip temperature within safe limits, a thermal management unit (TMU) is implemented on the chip [12] to throttle down the cores and/or power-gate them when a critical temperature T_{crit} is exceeded. That implies, triggering TMU might limit hardware capabilities and thereby leading to significant performance losses, as demonstrated in [13]. As a result, there is a pressing need for *smart thermal management* at the system level that is able to take advantage of the existing heterogeneity to find efficient trade-offs that increase thermal efficiency.

In this paper, we investigate and highlight the key parameters of heterogeneity in multicores that can be exploited by thermal management. Furthermore, we present efficient constraints that are derived considering some heterogeneity parameters in order to reduce the complexity of thermal management solutions and increase their thermal efficiency. Finally, we demonstrate smart thermal management solutions that employ the derived constraints and intelligently exploit the heterogeneity parameters of multicores aiming at maximizing thermal efficiency.

II. Challenges of Thermal Management for Heterogeneous multicores

Heterogeneity in multicores poses new challenges to thermal management techniques. These challenges stem from the different parameters that belong to various types of heterogeneity at both the chip level and the application level. For example, at the chip level, PEs may show heterogeneous characteristics, like power consumption and compute capabilities. At the application level, different applications may have diverse performance and power characteristics, even when they are executed on the same type of PEs. The heterogeneity parameters (will be explained below in details) have direct impact on both the temperature of the chip and the performance of the applications. Therefore, they must be considered by thermal management techniques to enable making thermally efficient decisions. On one hand, each of these parameters presents a challenge, because it adds a new dimension to the thermal management problem and thereby increases the

[1] Thermal efficiency is defined in this work as the ability to keep temperatures within safe limits, while at the same time maximizing the performance.

978-3-9819263-2-3/DATE19/©2019 EDAA

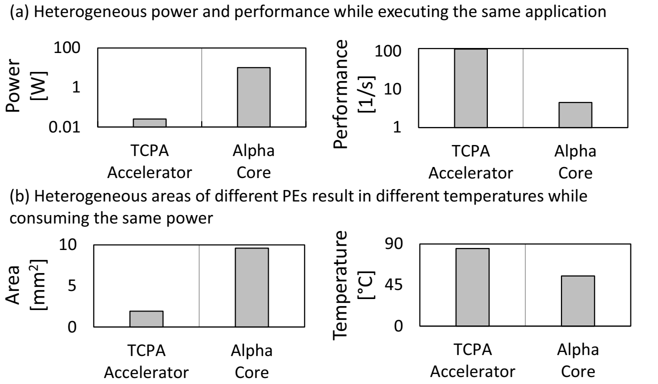


Fig. 1: (a) Heterogeneous PEs result in diverse power and performance values when executing the same applications. (b) PEs with different areas generate diverse temperatures even if they consume the same power.

solution complexity. On the other hand, if they are intelligently exploited, these parameters can provide a great opportunity for performance optimization under a thermal constraint.

In the following, we present various types of heterogeneity in multicores and discuss the key parameters that belong to each type.

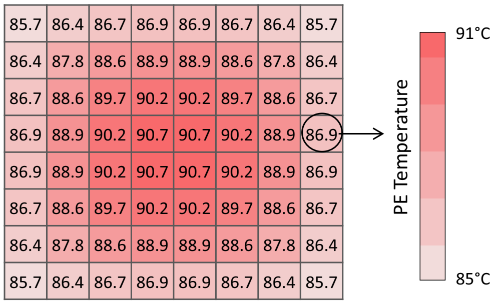
A. *Inter-Tile Heterogeneity:*

The first heterogeneity type indicates that the tiles can be different from each other in the following parameters:

- Processing element type
- Number of processing elements
- Voltage and frequency levels

As aforementioned, different tiles of heterogeneous multicores can integrate different types of processing elements (PEs), such as general-purpose CPUs (cores) or accelerators. For some of these types, an even more fine-grained division is required, e.g., in-order or out-of-order cores. This heterogeneity at different levels will indicate various compute capabilities and thus the resulting performance and power consumption while executing the same application might be significantly diverse. Additionally, the dimensions and thus the area of the PEs might be different. As a result, if two different PEs with different areas consume the same power, the resulting power densities, and therefore the temperatures, will be different. Fig. 1 illustrates a comparison of two different PEs; namely Alpha core [14] and TCPA accelerator [15]. These heterogeneous PEs result in diverse power consumption and performance, as shown in Fig. 1a. Fig. 1b shows the areas of these PEs, which lead to generating different temperatures, for the same power consumptions on the PEs[2].

Besides heterogeneity in PE type, the number of PEs and the available V/f levels play a significant role in specifying the performance potential of executing an application by a given tile and the expected resulting temperature. For example, a tile with a large number of Alpha out-of-order cores and a maximum frequency of 4.0 GHz has a large performance potential because high number of parallel application threads can be executed and also the V/f level can be upscaled to 4.0 GHz. Such a tile, however, might lead to generating high temperatures. Contrarily, a tile with small number of ARM in-order cores and a maximum frequency of 2 GHz will not generate high temperatures like the first tile but obviously it cannot achieve the same performance.



85.7	86.4	86.7	86.9	86.9	86.7	86.4	85.7
86.4	87.8	88.6	88.9	88.9	88.6	87.8	86.4
86.7	88.6	89.7	90.2	90.2	89.7	88.6	86.7
86.9	88.9	90.2	90.7	90.7	90.2	88.9	86.9
86.9	88.9	90.2	90.7	90.7	90.2	88.9	86.9
86.7	88.6	89.7	90.2	90.2	89.7	88.6	86.7
86.4	87.8	88.6	88.9	88.9	88.6	87.8	86.4
85.7	86.4	86.7	86.9	86.9	86.7	86.4	85.7

Fig. 2: Heterogeneous thermal susceptibility of identical PEs inside one tile leads to different temperatures on the PEs, while consuming the same power, i.e., 5 W per PE.

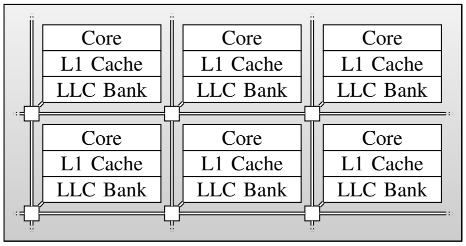


Fig. 3: A tile with S-NUCA architecture with a physically-distributed, yet logically-shared LLC.

B. *Intra-Tile Heterogeneity:*

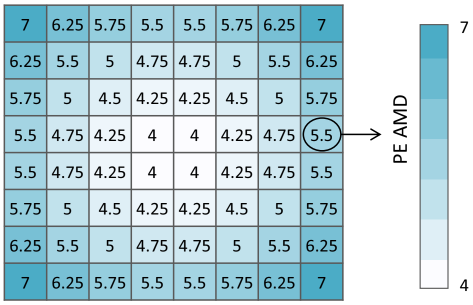
This type of heterogeneity refers to the heterogeneity of PEs within one tile. Although all PEs inside one tile belong to the same type, i.e. the same architecture, the same compute capabilities, and the same area, they still differ from each other in two parameters:

- Thermal susceptibility
- Last-Level-Cache (LLC) latency

The difference in thermal susceptibility means that PEs inside one tile, which have the same type and area, might generate different temperatures, even if they consume the same amount of power. Thermal susceptibility depends on the location of the PEs within the chip. For example, the PEs near the chip center generate higher temperature than the PEs near the chip edges, even if they consume the same amount of power. The reason is that the PEs near the chip center have higher thermal contributions from the rest of the PEs on the chip. Fig. 2 shows the different generated temperatures on the PEs of one tile composed of 8x8 cores.

Depending on the cache architecture deployed inside a tile, there can also be heterogeneity in the LLC latency of the PEs. This is the case for Static Non-Uniform Cache Access (S-NUCA) architectures [18]. Figure 3 shows a tile with S-NUCA caches. It consists of multiple cores that are connected by a NoC. The physically-distributed LLC is divided into several LLC banks – each co-located with a core, but they are logically shared among all cores. In such a tile, the latency of a single LLC access by an application thread depends on

[2] The performance, power, area and temperatures of the PEs are obtained using simulation tools which are explained in details in [16], [17].



7	6.25	5.75	5.5	5.5	5.75	6.25	7
6.25	5.5	5	4.75	4.75	5	5.5	6.25
5.75	5	4.5	4.25	4.25	4.5	5	5.75
5.5	4.75	4.25	4	4	4.25	4.75	5.5
5.5	4.75	4.25	4	4	4.25	4.75	5.5
5.75	5	4.5	4.25	4.25	4.5	5	5.75
6.25	5.5	5	4.75	4.75	5	5.5	6.25
7	6.25	5.75	5.5	5.5	5.75	6.25	7

Fig. 4: Heterogeneous AMD to the LLC banks of identical PEs inside an S-NUCA tile leading to different average LLC latency. The higher the AMD, the higher is the average LLC latency.

the hop count (Manhattan Distance) on the NoC between the core where the thread is running and the LLC bank where the data resides and therefore is non-uniform. Consequently, the average LLC latency among many LLC accesses depends on the Average Manhattan Distance (AMD) between the core where the thread is running and *all LLC banks* [19]. The AMD is a static property of every core. Fig. 4 shows the AMD values for a tile with 8×8 cores. The lower is the AMD, the lower is the average LLC latency.

By comparing Fig. 2 and Fig. 4, it can be noticed that there is an inherent trade-off between thermal susceptibility and AMD of a core [20]. The closer a core is to the center, the lower is its AMD, but the higher is its thermal susceptibility. Contrarily, cores near the corners have low thermal susceptibility, but high AMD.

C. Inter-Application Heterogeneity:

This type of heterogeneity indicates the diverse characteristics of different applications even if they are executed on the same type of PEs. These characteristics are:

- Thread-level parallelism (TLP)
- Instruction-level parallelism (ILP)
- Sensitivity to LLC latency
- Power consumption

Fig. 5 illustrates the difference w.r.t. the aforementioned parameters between two applications; *x264* and *canneal*, from PARSEC benchmark suite [21].

The applications differ from each other with their thread-level parallelism (TLP). Particularly, each application has a sequential and a parallelizable part. The dominance of the parallelizable part of the application indicates its TLP. That implies, the performance of applications with high TLP will be significantly increased when more parallel threads and thus more PEs are assigned to the application. Contrarily, for applications with low TLP, the performance gain when assigning more PEs to the application will be limited. Fig. 5a shows the resulting performance speedup of the two different applications when increasing the parallelism level. The application with higher TLP (*x264*) gains more performance. In addition to TLP, applications differ from each other with their instruction-level-parallelism (ILP). ILP indicates how many instructions of the application can be executed simultaneously. Applications with high ILP will gain significant performance when the V/f levels of the executing PEs are increased (Fig. 5b).

One additional heterogeneity parameter of applications is their sensitivity to the LLC latency. Particularly, compared to computation-intensive applications, memory-intensive ones expose high sensitivity to the LLC latency, because such applications have to frequently read/write from/to the LLC during their execution. The performance of such applications is reduced if they are executed on PEs with high LLC latency. Now the mapping of applications to cores determines the average LLC latency experienced by the application. This is the case for example with S-NUCA tiles (see Section II-B), where the average LLC latency depends on the AMD of the core. Fig. 5c illustrates the performance sensitivity of two different applications to different AMD which resulting in different average LLC latency. Besides their heterogeneity in performance characteristic, applications have heterogeneous power consumptions as shown in Fig. 5d.

D. Intra-Application Heterogeneity:

This kind of heterogeneity refers to the variation in the characteristics within one application through two parameters:

- Thread type (e.g. master or slave)
- Execution phases

As an example, these two parameters affect the performance sensitivity to LLC. In particular, different threads of a single application may show different characteristics. Figure 6a shows the susceptibility of PARSEC *blackscholes* master and slave threads to the AMD. The master thread prepares data, which is processed by the slave threads. Due to more frequent LLC accesses, the master thread is more sensitive to the AMD and therefore its performance drops significantly with increasing AMD (-32 %), whereas the performance of the slave threads only decreases slightly (-7 %).

Not only the characteristics of different threads may vary, but also characteristics of a single thread may vary over time. This may e.g. stem from the application traversing different execution phases, or changes in the input data. Figure 6b shows the relative number of cycles spent waiting for the LLC during the execution of single-threaded *x264*. The execution is divided into five distinct phases. In this example, the phases stem from the input data, which are the different frames to be encoded. In Phases 2, 3 and 5, *x264* accesses the LLC more frequently and therefore its susceptibility to the LLC latency is higher compared to Phases 1 and 4. A thermal management solution needs to make sure that during Phases 2, 3 and 5, *x264* is mapped to a core with low AMD, whereas during Phases 1 and 4, the AMD plays only a minor role and cores with low AMD can be used to speed up other applications.

Opportunity for Thermal Efficiency: Any of the above-mentioned heterogeneity parameters of the applications can be exploited to find a trade-off between performance and temperature. For example, we should not assign a large number of PEs for applications with low TLP because that will not increase their performance, and thus unnecessary temperature increase can be avoided. Moreover, applications with low sensitivity to LLC can be mapped to cores with high AMD, which helps reducing temperature increase and might give a thermal margin for other applications that need to be mapped to cores with low AMD. Analogue discussion can be applied on any of these heterogeneity parameters. That emphasizes

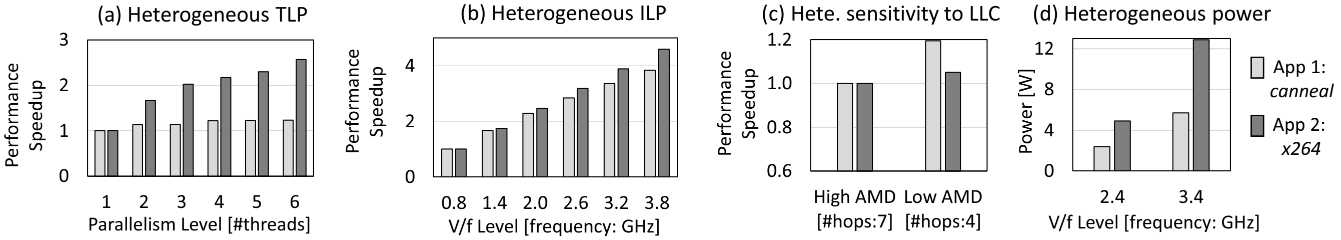


Fig. 5: Inter-application heterogeneity between two different applications w.r.t. TLP, ILP, sensitivity to LLC, and power consumption.

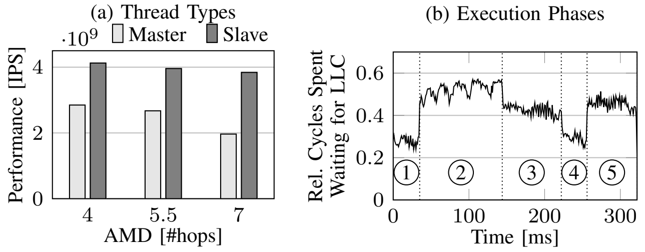


Fig. 6: Intra-application heterogeneity: (a) *Blacksholes* master thread exhibits stronger susceptibility to the AMD than the slave threads. (b) *X264* execution exhibits different phases that affect the LLC accesses.

the need for smart thermal management that is conscious of the existed opportunities behinds these parameters and able to exploit them to achieve thermal efficiency.

III. Efficient Thermal Management Constraints

The complexity of decision making process within thermal management is continuously increasing due to the increase in the number of PEs integrated into the chip and also due to the inherent heterogeneity within the chip. One essential source of this complexity is the need to consider the heat transfer between the PEs. Particularly, the power consumption on one PE influences the temperatures of all other PEs. Thus, several state-of-the-art system-level techniques, e.g., [22], employ the Thermal Design Power (TDP) of the chip as a power constraint which helps avoiding thermal violations without the need to directly deal with PE temperatures and heat transfer issues.

TDP is defined as the power amount that the cooling system of the chip can dissipate [12]. However, it is specified independent of the number of active PEs on the chip and therefore, it cannot guarantee avoiding thermal violations. For instance, if just few PEs on the chip consume power equal to TDP, thermal violations may occur, because TDP has been consumed on small area on the chip, resulting in high power density and thus high temperature. As a result, employing TDP as a power constraint might reduce thermal violations but does not guarantee avoiding them.

Therefore, we present three efficient constraints that help in simplifying the complexity of thermal management techniques but at the same time they are thermally safe.

A. Thermal Safe Power (TSP)

In [13], we present a constraint called Thermal Safe Power (TSP), which is an abstraction that provides a safe power constraint as a function of the number and location, i.e., the mapping, of simultaneously active PEs. Executing PEs at any power consumption below TSP ensures that no thermal violation occurs in the steady-state; i.e., the critical temperature T_{crit} on the chip will not be exceeded. To extract TSP, we employ an RC thermal network that represents the targeted chip and can be used to calculate the steady-state temperatures as shown in Eq. (1).

$$T_i = \sum_{j=1}^{N} b_{i,j} \cdot p_j + c_i \tag{1}$$

p_j is the power consumption of PE j, $b_{i,j}$ represents the heat contribution of PE j into the temperature of PE i. N is the number of PEs on the chip. c_i represents the heat contribution from the ambient temperature to PE i. To differentiate between active and inactive PEs, a vector $\mathbf{Q} = [q_j]_{N\times 1}$ is considered, where $q_i = 1$, when PE j is active, and $q_j = 1$, otherwise. Assuming that all active PEs consume equal power P_{equal}, and the inactive PEs are power-gated ($p_j = 0$), Eq. 1 can be written as follows:

$$T_i = P_{equal} \cdot \sum_{j=1}^{N} b_{i,j} \cdot q_j + c_i \tag{2}$$

If T_i is set to T_{crit}, we can derive the power budget (P_{equal}) that lets the temperature of PE i equal to T_{crit}. However, the temperature of the rest of the PEs might be higher or lower than T_{crit}, even if these PEs consume the same power P_{equal} (due to the heterogeneous thermal susceptibility explained in Section II-B). Therefore, to obtain a thermally safe power constraint (TSP), P_{equal} needs to be computed for all PEs $i = 1, 2, \ldots, N$ and then we adopt the minimum computed power value as our constraint TSP.

$$\text{TSP} = \min_{1\leq i\leq N} \left\{ (T_{crit} - c_i)/(\sum_{j=1}^{N} b_{i,j} \cdot q_j) \right\} \tag{3}$$

TSP considers one heterogeneity parameter of the inter-tile heterogeneity, which is the heterogeneous susceptibility of temperature increase of the PEs, as it depends on the RC-thermal network, which considers all the thermal characteristics of the PEs. However, TSP assumes that all PEs can consume the same power budget, i.e., P_{equal} in Eq. 3. This assumption is not suitable for heterogeneous PEs as applying the same power constraint on them might result in a huge difference in their temperatures due to the difference in their areas (see Fig. 1b). To overcome this limitation, a power density constraint is proposed in the next section.

B. Thermal Safe Power Density (TSPD)

To consider the potential heterogeneity in the areas of PEs, we derive in [16], [23] a thermally safe power density constraint (TSPD). Applying a power density constraint will allow consuming low power in small PEs and high power in large PEs. To derive TSPD, the area of the PEs need to be involved in the equation as follows:

$$\mathrm{TSPD} = \min_{1\leq i\leq N}\left\{(T_{crit}-c_i)/(\sum_{j=1}^{N} b_{i,j}\cdot area_j\cdot q_j)\right\} \qquad (4)$$

TSPD considers the heterogeneity in the PE area, but it does not consider the heterogeneity in the resulting power consumption of executing different applications, the heterogeneity in the compute capabilities of the PE, and the heterogeneity in the maximum allowable V/f levels of the tile. In fact, these three parameters might lead to scenarios, in which TSPD cannot be fully utilized by some PEs, whose resulting power consumption from executing applications, although they run at their maximum V/f level. Such scenarios results in an unexploited thermal margin on that PE leading to wasting a performance potential. To avoid this inefficiency, we propose to adapt TSPD to consider the above mentioned heterogeneity in the power consumptions, as explained in the next section.

C. Adaptive Thermal Safe Power Density (ATSPD)

To adapt TSPD to the actual power consumption of the PEs after running given applications, it is necessary to find which PEs underutilize TSPD, which are the ones that reach their maximum V/f level and their power consumption is still less than TSPD. After finding these PEs, TSPDA is calculated while fixing their actual power consumptions in Eq. (4). In particular, a binary vector $\mathbf{X} = [x_j]_{N\times 1}$ is defined, so that $x_j = 1$ implies that the actual power consumption of PE j is given and fixed, while $x_j = 0$ implies that the new adapted power density constraint will be computed for PE j. Thus, the following equation calculates the Adaptive Thermal Safe Power Density (ATSPD) as follows:

$$\mathrm{ATSPD} = \min_{1\leq i\leq N}\left\{\frac{T_{crit}-(\sum\limits_{j=1}^{N} b_{i,j}\cdot p_j\cdot x_j\cdot q_j)-c_i}{\sum\limits_{j=1}^{N} b_{i,j}\cdot area_j\cdot(1-x_j)\cdot q_j}\right\} \qquad (5)$$

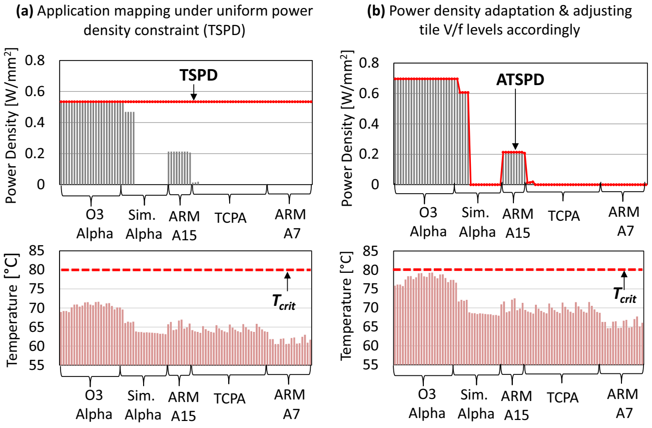
Thereby, ATSPD avoids unexploited thermal headroom by increasing the power budget of PEs, which ultimately allows improving performance, while still being thermally safe.

IV. Smart Thermal Management Solutions

The aforementioned constraints are efficient and reduce the complexity of thermal management, because they abstract thermal issues as power and power density constraints and also consider some heterogeneity parameters. In the following, two thermal management techniques that employ these constraints, are demonstrated.

A. Power-Density-Aware Thermal Management

This section presents a power-density-aware thermal management (*PdTM*) technique [16] that enforces the ATSPD constraint while determining the application mapping on the chip and the V/f levels of the PEs, in order to increase thermal efficiency. To achieve its goal, *PdTM* considers and exploits several heterogeneity parameters: First, by enforcing ATSPD, heterogeneity in thermal susceptibility and in power characteristics of the PEs is implicitly considered. Additionally, *PdTM* considers and exploits inter-tile and inter-application heterogeneity parameters, which are the application performance at different PE types, the application TLP and ILP, and the available number of PEs and V/f levels inside different tiles.


(a) Application mapping under uniform power density constraint (TSPD)

(b) Power density adaptation & adjusting tile V/f levels accordingly

Fig. 7: The resulting power densities and the temperatures of PEs in heterogeneous multicores after applying the two steps of *PdTM*.

The first step of *PdTM* determines the application mapping, i.e., application-to-tile and thread-to-PE assignments, as well as the V/f levels of the PEs, so that the overall system performance is maximized under TSPD. Fig. 7a illustrates the resulting power densities on the different PEs, after applying this step of *PdTM*, which employs TSPD. The second step of *PdTM* considers the actual power consumption resulting from executing the mapped applications on the PEs to adapt the power density constraint according to ATSPD (Fig. 7b). Then, the V/f levels of the PEs are upscaled to exploit the new potential margin below the new ATSPD constraint, leading to significant performance improvements. As evaluated in [16], *PdTM* outperforms several state-of-the-art techniques, due to its consideration of several heterogeneity parameters.

B. TSP-Aware & LLC-Latency-Aware Thermal Management

This section presents a thermal management technique *PCMig* [17] that aims at achieving further thermal efficiency within the tiles, by considering intra-tile, inter-application and intra-application heterogeneity w.r.t. average LLC latency. Moreover, *PCMig* employs the TSP constraint, and thus, the intra-tile thermal susceptibility is implicitly considered. As Section II-B demonstrated, when mapping threads to cores in an S-NUCA tile, there is a trade-off between thermal susceptibility and AMD of a core. Section II-D showed that the susceptibility of threads to these factors differs among threads and may even change over time. *PCMig* addresses this by dynamically adapting the mapping using task migrations.

The run-time migration algorithm periodically traverses the following steps: (1) Create migration candidates, (2) rate these candidates using a performance prediction model and (3) execute the migration candidate with the highest rating. The performance prediction model predicts the performance of all threads before and after a migration based on the mappings before migration M_0 and after migration M_m. The

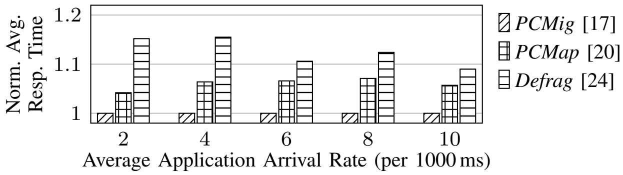


Fig. 8: *PCMig* improves the performance on S-NUCA tiles by up to 16 % over the state-of-the-art by considering the existing intra-tile, inter-application and intra-application heterogeneity.

migration candidates are rated by the predicted average relative performance improvement among all threads:

$$\Delta_{IPS}(m) = \sum_{t \in T} \left(\frac{\text{IPS}(t, M_m)}{\text{IPS}(t, M_0)} - 1 \right)$$

No actual migration is performed to calculate the rating. Finally, the migration candidate m with the highest rating $\Delta_{IPS}(m)$ is executed.

Fig. 8 shows the performance improvements obtained with *PCMig* over the state-of-the-art approaches *Defrag* [24] and *PCMap* [20] for different application arrival rates. Thereby, the average (peak) utilization is varied from 10 % (58 %) to 38 % (100 %). Details on the experimental setup can be found in [17]. *Defrag* performs *defragmentation* of the mapping. It migrates running applications in order to let the idle cores form a contiguous shape that is available for new arriving applications. Under message-passing, this reduces the communication latency and thus increases the performance. However, this is not the case with S-NUCA. *Defrag* is not aware of the heterogeneity present in S-NUCA tiles and therefore results in suboptimal performance. *PCMap* is a run-time mapping algorithm tailored for S-NUCA tiles, that exploits the intra-tile heterogeneity w.r.t. thermal susceptibility and LLC latency. However, *PCMap* does not consider the inter- and intra-application heterogeneity, and therefore is not able to achieve optimal performance. *This demonstrates that performance cannot be maximized without accounting for all existing heterogeneity.*

V. Conclusion

Multicore processors exhibit different kinds of heterogeneity. This comprises heterogeneity between tiles of multicores, between processing elements within one tile, between different applications and even within a single application. Thermal management can only be thermally efficient, i.e., maximize the performance under a temperature constraint, if the existing heterogeneity is considered. This work demonstrates efficient constraints that account for some of the heterogeneity and serve as an abstraction to simplify thermal management techniques. Furthermore, two smart thermal management techniques are presented, which employ these constraints and take advantage of existing heterogeneity to find trade-offs that increase thermal efficiency. The presented techniques significantly outperform the state of the art.

Acknowledgment

This work was partly supported by the German Research Foundation (DFG) as part of the Transregional Collaborative Research Centre *Invasive Computing* (SFB/TR 89).

References

[1] G. M. Bhat, S. Gumussoy, and U. Y. Ogras, "Power and Thermal Analysis of Commercial Mobile Platforms: Experiments and Case Studies," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2019.

[2] B. K. Joardar, R. G. Kim, J. R. Doppa, and P. P. Pande, "Design and Optimization of Heterogeneous Manycore Systems enabled by Emerging Interconnect Technologies: Promises and Challenges," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2019.

[3] R. Ubal, D. Schaa, P. Mistry, X. Gong *et al.*, "Exploring the heterogeneous design space for both performance and reliability," in *51st Design Automation Conference (DAC)*, 2014, pp. 1–6.

[4] J. Henkel, A. Herkersdorf, L. Bauer, T. Wild *et al.*, "Invasive manycore architectures." in *ASP-DAC*, 2012, pp. 193–200.

[5] R. Ubal, D. Schaa, P. Mistry, X. Gong *et al.*, "Exploring the heterogeneous design space for both performance and reliability," in *Design Automation Conference (DAC), 2014 51st ACM/EDAC/IEEE*. IEEE, 2014, pp. 1–6.

[6] J. Henkel and Y. Li, "Energy-conscious hw/sw-partitioning of embedded systems: A case study on an mpeg-2 encoder," in *Hardware/Software Codesign, 1998.(CODES/CASHE'98) Proceedings of the Sixth International Workshop on*. IEEE, 1998, pp. 23–27.

[7] S. Pagani, A. Pathania, M. Shafique, J.-J. Chen *et al.*, "Energy efficiency for clustered heterogeneous multicores," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 5, pp. 1315–1330, 2017.

[8] "Exynos 5 octa (5422)," http://www.samsung.com/exynos.

[9] H. Esmaeilzadeh, E. Blem, R. St. Amant, K. Sankaralingam *et al.*, "Dark silicon and the end of multicore scaling," in *38th International Symposium on Computer Architecture (ISCA)*, 2011, pp. 365–376.

[10] J. Henkel, H. Khdr, S. Pagani, and M. Shafique, "New trends in dark silicon," in *52nd Annual Design Automation Conference (DAC)*, 2015.

[11] J. Henkel, T. Ebi, H. Amrouch, and H. Khdr, "Thermal management for dependable on-chip systems," in *Design Automation Conference (ASP-DAC), 2013 18th Asia and South Pacific*. IEEE, 2013, pp. 113–118.

[12] "Intel xeon phi coprocessor datasheet," ser. June, 2013.

[13] S. Pagani, H. Khdr, W. Munawar, J.-J. Chen *et al.*, "TSP: Thermal Safe Power - efficient power budgeting for many-core systems in dark silicon," in *International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, 2014, pp. 10:1–10:10.

[14] R. Kessler, "The alpha 21264 microprocessor," *IEEE Micro*, vol. 19, no. 2, pp. 24–36, Mar. 1999.

[15] F. Hannig, V. Lari, S. Boppu, A. Tanase *et al.*, "Invasive tightly-coupled processor arrays: A domain-specific architecture/compiler co-design approach," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 13, no. 4s, pp. 133:1–133:29.

[16] H. Khdr, S. Pagani, E. Sousa, V. Lari *et al.*, "Power-Density-Aware Resource Management for Heterogeneous Tiled Multicores," *IEEE Transactions on Computers*, vol. 66, no. 3, pp. 488–501, 2017.

[17] M. Rapp, A. Pathania, T. Mitra, and J. Henkel, "Prediction-Based Task Migration on S-NUCA Many-Cores," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2019.

[18] C. Kim, D. Burger, and S. W. Keckler, "An Adaptive, Non-Uniform Cache Structure for Wire-Delay Dominated On-Chip Caches," in *Architectural Support for Programming Languages and Operating Systems (ASPLOS)*. ACM, 2002, pp. 211–222.

[19] A. Pathania and J. Henkel, "Task scheduling for many-cores with s-nuca caches," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2018*. IEEE, 2018, pp. 557–562.

[20] M. Rapp, A. Pathania, and J. Henkel, "Pareto-Optimal Power- and Cache-Aware Task Mapping for Many-Cores with Distributed Shared Last-Level Cache," in *Int. Symp on Low Power Electronics and Design (ISLPED)*. ACM/IEEE, 2018, pp. 16:1–16:6.

[21] C. Bienia, S. Kumar, J. P. Singh, and K. Li, "The PARSEC benchmark suite: Characterization and architectural implications," in *Proceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques (PACT)*, 2008, pp. 72–81.

[22] T. Muthukaruppan, M. Pricopi, V. Venkataramani, T. Mitra *et al.*, "Hierarchical power management for asymmetric multi-core in dark silicon era," in *50th Design Automation Conference (DAC)*, 2013, pp. 174:1–174:9.

[23] S. Pagani, H. Khdr, J.-J. Chen, M. Shafique *et al.*, "Thermal safe power (tsp): Efficient power budgeting for heterogeneous manycore systems in dark silicon," *IEEE Transactions on Computers*, vol. 66, no. 1, pp. 147–162, 2017.

[24] J. Ng, X. Wang, A. K. Singh, and T. Mak, "Defragmentation for Efficient Runtime Resource Management in NoC-Based Many-Core Systems," *Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 24, no. 11, pp. 3359–3372, 2016.