Two-stage Thermal-Aware Scheduling of Task Graphs on 3D Multi-cores Exploiting Application and Architecture Characteristics

Zuomin Zhu
Hong Kong University of Science and Technology
Clear Water Bay, Kowllon, Hong Kong
zzhuae@connect.ust.hk

Vivek Chaturvedi
Nanyang Technological University
P.O. Box 1212, Singapore
vchaturvedi@ntu.edu.sg

Amit Kumar Singh
University of Southampton
Southampton,SO17,1BJ, UK
A.K.Singh@soton.ac.uk

Wei Zhang
Hong Kong University of Science and Technology
Clear Water Bay, Kowllon, Hong Kong
wei.zhang@ust.hk

Yingnan Cui
Nanyang Technological University
N4-02a-32 Nanyang Avenue, Singapore
ycui@e.ntu.edu.sg

Abstract —In this paper, we propose a two-stage thermal-aware task scheduling policy which exploits the application and system architecture characteristics to decouple the mapping of task-graphs for the performance and peak temperature optimization into two stages. At the first stage, the algorithm collects the best mapping of task-graphs exploiting the application and architecture characteristics to minimize the makespan of the task-graphs. At the second stage, a light-weight online algorithm comprised of efficient thermal rank and combined power models is performed to map the task nodes to the real cores for temperature minimization while maintaining the best possible performance achieved in the first stage. Compared to the previous approaches which perform the performance and temperature optimization together, our method can reduce the online mapping algorithm complexity and improve its efficiency. Experiments on real benchmarks show that an average of 6.3°C peak temperature reduction and 6.8% performance improvement can be achieved compared to other existing methods.

I. Introduction

Three-dimensional (3D) chip-multiprocessor (CMP) is a promising multi-core architecture in which multiple active silicon layers are stacked vertically using Through Silicon Vias (TSVs) [7] to gain tremendous performance with energy efficiency and scalability [15]. However, 3D integration causes severe thermal challenges due to increasing power density and rapid heat transfer between vertically placed computational units resulting in soaring chip temperature, causing reliability threat and device aging [14].

In order to address thermal challenges in multi-core, several system level task mapping and scheduling techniques are proposed [4, 12, 2, 3, 8]. Unfortunately, most of the existing approaches are designed for 2D CMP and cannot be applied efficiently to 3D CMP due to its distinctly different thermal characteristics. The reported approaches for 3D CMP either only target independent tasks, or require compute-intensive online procedure for task-graph mapping.. Because they considered performance and temperature optimization together, relatively complicated thermal simulations and searches are required for candidate mappings [12, 8]. Moreover, in many techniques leakage power consideration is missing from the power model that is closely related to temperature [4, 3, 8, 12].

In this paper, we propose a two-stage thermal-aware task scheduling algorithm, namely communication-aware group (CAG) stage and thermal-aware scheduling (TAS) stage, for peak temperature minimization with best possible performance of task-graph mapping on 3D CMP system. The CAG stage is implemented at design-time while TAS stage is implemented at run-time. We decouple the optimization of makespan and temperature into two stages and optimize them separately. Exploiting the application characteristics with brute-force or genetic algorithm method at CAG stage, we bind individual tasks to super tasks assuming different number of available cores to minimize the makespan considering the worst-case communication. Such consideration facilitates the second stage to minimize the peak temperature while not affecting the optimized makespan of the first stage. At TAS stage, super tasks will be mapped to real available cores for peak temperature minimization while maintaining the performance optimization achieved in the first stage. A novel thermal-aware mapping algorithm comprised of thermal rank model and combined power model (defined in Sec.B.2 and Sec.B.3) which captures the characteristics of 3D CMP, especially targeting different layer using different tactics, is proposed to direct the super task mapping with high efficiency and low computation complexity.

Thus, our proposed two-stage scheduling algorithm, called TSS, can achieve the peak temperature minimization of the 3D system while optimizing the makespan of the application with light-weight online complexity.

II. related work

The reported thermal-aware task mapping and scheduling techniques for 3D CMP can be classified into two categories: those targeting independent tasks and others targeting task graphs.

Most of these works targeting independent tasks utilize thermal-aware or power-aware scheduling approaches [12, 18, 17, 16, 9]. In [12, 18], temperature minimization is achieved based on stack power balance. In [17], incremental update of thermal simulation is used. A power-aware partition scheme and thermal-aware speed adjustment policy are introduced in [16]. In [9], supertasks are allocated to 3D core stacks based

on their thermal profile, and dynamic voltage and frequency scaling (DVFS) is applied on selected cores to achieve better cooling in case of thermal emergencies. Although these works show some benefits, some of them ignore influence of neighboring cores and leakage power on temperature[12, 18], or exhibit high computational complexity [17], or consider only power of cores ignoring real time temperature during allocation [16] and incur frequent DVFS switching [9]. In addition, these methods cannot be directly applied to task graphs as task dependencies are not considered.

A few works are proposed to address the challenge of task graph mapping and scheduling on 3D CMP [2, 4, 8, 3, 10]. In [2], hot tasks are first mapped to the layer closest to the heat sink and then tasks are swapped between hot and cold stacks to reduce temperature. However, it might incur huge swapping overhead. In [4], first at run-time, all tasks are scheduled to the bottom layer and then low-power tasks are moved to the top layer. This leads to high online complexity and execution overhead. In [8], tasks are scheduled in the order of assigned priority based on the data dependencies. However, thermal simulations at the scheduling stage lead to high scheduling overhead. In [3], offline optimized thermal profiling results are used during online scheduling, but real-time temperature is not considered. In a recent work [10], layer-by-layer task-to-core mapping and energy-aware voltage scaling are incorporated to reduce peak temperature and temperature gradient without extensive thermal simulation. However it includes several computation steps both for mapping and voltage scaling incurring large online overhead.

III. Preliminaries

A. 3D Multiprocessor Model

The target 3D CMP consists of multiple active silicon layers stacked vertically using TSVs as depicted in Fig.1. An active layer is modeled as a two-dimensional mesh of processing cores, where each core is composed of three functional units: processing element (PE), network interface (NI) and L2 cache (MEM). Communication between any two cores on the same layer takes place via a network-on-chip (NoC), while TSV bundles are employed for vertical communication. We refer to vertically aligned cores as a core stack. The TSV models, communication protocols and NoC routing algorithms are based on the information provided in [3].

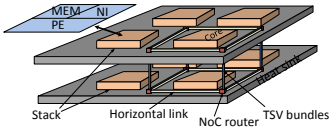

Fig. 1. Two-layer 3D CMP architecture

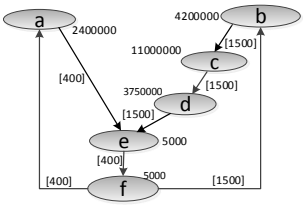

Fig. 2. An example DG model: Autoindust2

B. Application Model

An application is modeled as directed graph (DG), for example, the task graphs used in multimedia applications. The DG can be cyclic or acyclic. A typical DG consists of a set of vertices and directed edges, denoted as $G(V, E)$. The vertices V denote the tasks (nodes) to be executed, and edges E represent the precedence relationship between tasks. Each node is characterized with its execution time and dynamic power consumption and each edge with its communication load, which are assumed to be known at design time. An example DG model of the *Autoindust2* is depicted in Fig.2. The value at each vertex represents its execution time in cycles, and value on edges are the number of bits to be transferred between dependent nodes.

C. Power Model

For each core, the overall power consumption is composed of dynamic power and leakage power. Dynamic power is independent of temperature, while leakage power is sensitive to temperature. Following to the simple yet sufficiently accurate model in [13], we model the leakage power of core i as follows:

$$P_{leak,i} = \alpha \cdot T_i(t) + \beta \quad (1)$$

where α and β are constants depending on the manufacturing technology and can be calculated based on parameters provided in [11].

D. Definitions

To ease the understanding of the paper, we define the necessary terminologies as follows:

1) *Super Task*: It is a group of tasks (nodes) which are mapped to a particular core in a given scheduling interval as shown in Fig.4.
2) *Thermal Rank*: A metric that determines the likelihood of a core to receive new tasks for execution. It essentially calculates the thermal efficiency of each core based on its topological position and thermal influence of adjacent neighbours.
3) *Combined Power*: A metric that evaluates the weighted summation of powers of all cores in a core stack.

IV. Two-stage thermal aware task mapping algorithm

Our two-stage algorithm consists of the CAG stage, which uses design-time exhaustive or GA approach for makespan minimization exploiting task graph characteristics, and the TAS stage which uses run-time heuristic for peak temperature minimization exploiting the 3D architecture characteristics.

A. CAG Stage: Design-Time Makespan Optimization

For each application to be mapped at run-time, the CAG stage computes the best mappings from makespan point of view while utilizing different number of cores such that one of these mappings can be selected for the real platform. Depending on the dependence of the tasks,, the best mappings are identified and super tasks are generated by employing an exhaustive exploration or genetic algorithm (GA) based exploration.

The exhaustive exploration evaluates all the possible tasks to cores combinations, i.e. mappings, and the tasks assigned in one core form a super task. Since exhaustive exploration evaluates all the possible mappings, it leads to optimal mapping at different number of used cores. However, the number of

mappings increases with the number of tasks n and thus the overall evaluation time. Therefore, for large number of tasks, the evaluation may not be completed within a limited time and GA based heuristic is employed to find optimized mapping.

Applying the GA requires to identify the fitness function and initial population. Since this stage tries to optimize makespan, it has been chosen as the fitness function. Based on the application tasks and number of cores to be used, a set of random and unique tasks-to-cores mappings is assigned as the initial population. To steer the optimization process in order to identify the makespan optimized mapping, we employ NSGA-II as the underlying GA [5]. Assuming different number of cores, the same process is applied for each of the individual applications.

For a given 3D CMP, the cores are assumed to be separated by the maximum hop in the architecture to take maximum communication delay into account. This ensures that in the second stage the dependency and timing constraints between task nodes will be satisfied when the super tasks are mapped to real cores as the distance between real available cores is definitely less than the longest distance.

While searching for the best mappings, the number of cores is varied from one to the number of tasks in the application in order to exploit all the potential parallelism present in the application by assuming that each task can occupy only one core. For each application, the allocation leading to optimized makespan at different number of used cores is stored to be used at run-time so that makespan optimized mapping can be performed at run-time and thermal optimization can be carried out. Fig.3 and Fig.4 show the design-time makespan optimization

#Cores	C1	C2	C3	C4	C5	C6	Period(cycles)
6	e	d	c	b	f	a	19018400
5	*	e	d	c	bf	a	19004900
4	*	*	de	c	bf	a	18991400
3	*	*	*	cde	bf	a	18977900
2	*	*	*	*	bcdef	a	18960000
1	*	*	*	*	*	bacdef	21360000

Fig. 3. Best mappings generated at design-time for Autoindust2

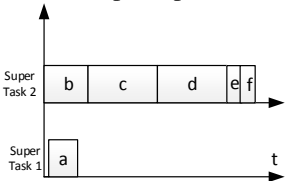

Fig. 4. Two super tasks with the minimum makespan

results after applying exhaustive exploration for the benchmark task graph shown in Fig.2. In the table of Fig.3, each entry from core C1 to core C6 represents a super task. The asterisk means that the specific core is not used. The last column gives the minimum makespan of the application based on different number of available cores. Fig.4 shows the two super tasks corresponding to the mapping using 2 cores with optimal makespan. At TAS stage, run-time thermal optimization algorithm will map these two super tasks to idle cores.

Note that we exploit the characteristics of application in CAG stage, assuming the normal task-graphs with medium communication volume that the communication delay will not dominate the total delay. Thus, during the second stage, when the super task node is mapped to the real cores, although the distance between super task nodes may vary from the first stage mapping, the makespan will improve with better communication delay, however, the best mapping will not be altered. Hence, the makespan optimization will not be compromised.

B. TAS Stage: Run-time Thermal Optimization

Due to significant difference in thermal characteristics of different layers, it is important to consider layers in 3D CMP separately. Assuming a two-layer 3D CMP as illustrated in Fig.5, for bottom layer (the layer closest to heat sink), dissipating heat to ambient is much easier. Thus we should take full advantage of it and allocate intense tasks to cores in this layer. For top layer (the layer far away from heat sink), hotspot and temperature emergency are easily generated. As a result, consideration of power of vertical neighbors is critical to avoid potential hotspots.

B.1 Overall TAS Algorithm

Our TAS approach is a combination of two heuristics for mapping in bottom and top layer called thermal rank and combined power model, respectively. We handle different layers using different tactics, aiming to dissipate the most possible heat in bottom layer, meanwhile limiting the total power in a stack. In previous approaches, they either only consider the total power in a stack or just take advantage of the heat dissipation efficiency of each core. Our method considers both factors efficiently.

The complete online thermal-aware super tasks mapping algorithm is depicted in Algorithm 1. The time complexity is $O(S)$, where S is the number of super tasks. When an application represented as a task graph arrives, depending upon the number of available cores, the best mapping and super tasks computed at design-time are selected. First, the selected super tasks are classified into hot super tasks and cool super tasks with their power above or below the average power. The average power value is calculated through a sliding window containing the past M tasks, where M is the size of the sliding window. Next, the hot super tasks are scheduled to the idle cores in bottom layer based on the thermal rank model, as described in steps 7-11, and cool super tasks are scheduled to the idle cores in the top layer according to the combined power model, as described in steps 13-16. If there is not enough idle cores in bottom layer, the hot super tasks with less power have to be allocated to the top layer based on the combined power model. Note that we map the cool super tasks to the top layer even if there are available idle cores in the bottom layer. This is because we need to spare the idle cores in the bottom layer for future coming hot super tasks, and the cool tasks in top layer will not increase the peak temperature sharply.

Our method utilizes light-weight models to efficiently model the thermal efficiency of each core for best usage, meanwhile limiting the total power in a core stack. In contrast to methods that simulate the peak temperature for a lot of candidate mappings in the mapping step [8, 12], our method requires no compute-intensive simulations.

B.2 Thermal Rank Model in Bottom Layer

Thermal rank model calculates the thermal efficiency of each core and indicates the priority of the cores for receiving the new task. During super tasks mapping in TAS stage, every idle core in the bottom layer is assigned a thermal rank value. Super tasks of incoming application are allocated to the cores with the

Algorithm 1 Thermal-Aware Mapping Algorithm

Require:
Super tasks set S generated at design-time stage;
Idle cores set $IDLE0$ in top layer;
Idle cores set $IDLE1$ in bottom layer;
Temperature set of all cores T;
Ensure: A super task mapping scheme ;
1: $\vec{TR}$: thermal rank array for idle cores in bottom layer
2: $\vec{P}$: combined power array for idle cores in top layer
3: Calculate thermal rank value for cores in $IDLE1$, get array $\vec{TR}$;
4: Calculate the potential highest combined power for cores in $IDLE0$, get array $\vec{P}$;
5: **while** S is not empty **do**
6: TB = popping up a super task in S which is ready;
7: **if** TB is a hot super task AND $IDLE1 \neq \emptyset$ **then**
8: Assign TB to bottom layer;
9: Find the minimum entry in $\vec{TR}$, and corresponding core id i;
10: Assign TB to core i;
11: Delete core i from $IDLE1$ and $\vec{TR}$.
12: **else**
13: Assign TB to top layer;
14: Find the minimum entry in $\vec{P}$, and corresponding core id j;
15: Allocate TB to core j;
16: Delete core i from $IDLE0$ and $\vec{P}$.
17: **end if**
18: **end while**

minimum rank values. A thermal rank value *TR* is calculated as follows:

$$TR = (T + Lw \cdot T_l + Vw \cdot T_v) \cdot \rho \cdot P_f \quad (2)$$

where T is the temperature of the candidate core, T_l is the average temperature of immediately adjacent cores in the same layer, T_v is the temperature of vertical neighbor in different layers. Lw, Vw, ρ, P_f are predefined constants. Lw and Vw indicate the influence from lateral and vertical neighboring cores, and they are calculated as the lateral and vertical thermal conductance, respectively. ρ denotes the proximity to the heat sink, and it is calculated as the thermal resistance between the specific layer to the ambient. Even if all cores have the same power consumption in a 3D system, the steady-state temperature of cores in the same layer differs due to different absolute position that has different number of surrounding cores. Thus taking the absolute position into consideration is important. P_f represents the influence of absolute position of a core in the horizontal plane. It can be calculated by assigning all cores the average power of task graphs and getting the steady-state temperature of all cores in a layer, then normalizing the steady-state temperature value of each core to the highest temperature. These parameters are determined offline based on the thermal characteristics of 3D CMP.

B.3 Combined Power Model in Top Layer

Considering thermal efficiency of individual core is not enough to achieve efficient peak temperature reduction, because it might map two hot super tasks into the same stack simultaneously. As a result, blocks with a high power consumption that are stacked on top of each other will generate hotspot. For the bottom layer, the consideration of thermal efficiency dominates as it is easier in this layer to dissipate heat to the ambient. While for the top layer, the consideration of controlling hotspots and reducing peak temperature dominates because peak temperature is usually generated in this layer. Thus it is imperative to consider cores in the same stack together for mapping tasks to the cores in the top layer. Here we propose a combined power model for cores in top layer to limit the total power in a core stack as follows:

$$P_{combined} = (R_{0\_amb} \cdot P_0 + R_{1\_amb} \cdot P_1) \cdot P_f \quad (3)$$

Where R_{0_amb} and R_{1_amb} denote the thermal resistance from silicon layer 0 to the ambient, and layer 1 to the ambient, respectively. P_f is exactly defined as in Sec.B.2. P_0 and P_1 denote the power consumption of cores in the same stack. For peak temperature reduction, the lateral influence in the top layer is not much significant. Thus we neglect the influence from lateral neighbors for reducing model complexity. Considering a core located in silicon layer j, its equivalent thermal resistance to the ambient R_{j_ambint} can be calculated as,

$$R_{j\_amb} = \sum_{k=1}^{j} R_{k,k-1} + R_{0,hsk} + R_{hsk,amb} \quad (4)$$

where $R_{k,k-1}$ represents the thermal resistance between silicon layers k and $k-1$. $R_{0,hsk}$ denotes the thermal resistance between silicon layer 0 and heat sink. $R_{hsk,amb}$ represents the thermal resistance from heat sink to the ambient. When a super task is to be assigned to top layer, a sequence of potential highest combined power is generated by assuming it is mapped to a candidate core. The core with the minimum combined power is selected to receive this super task. Fig.5 shows the target 3D CMP with cores in gray being unavailable. Assume that super task 1 in Fig.4, $SP1$, is hot and super task 2, $SP2$, is cool. Thus we assign $SP1$ to the bottom layer, and $SP2$ to the top layer. For bottom layer, we calculate thermal rank values of $C5$ and $C6$, and then allocate $SP1$ to the core with minimum thermal rank value. Similarly, for top layer, we calculate the combined power values of $C1$ and $C4$, and then allocate $SP2$ to the core with minimum combined power.

C. Extension To Multi-Layer 3D CMP

The earlier introduced algorithm for two-layer 3D CMP can be easily extended to 3D CMP with more than two layers. Since there is no industrial chip with more than two layers so far, we limit the exploration of the layers to be no more than four.

Combined power model in Eq.(3) can be adjusted as follows to incorporate more layers:

$$P_{combined} = \sum_{j=0}^{N} R_{j\_amb} \cdot P_j \quad (5)$$

where N is the number of layers. R_{j_amb} and P_j are exactly the same as defined in Section B.

In the extended thermal-aware task mapping algorithm, we should also take the same consideration of thermal efficiency for these layers which can fast dissipate heat to the ambient, and limiting the total power in a stack for those layers where hotspots usually arise. For a three-layer 3D CMP, we view the

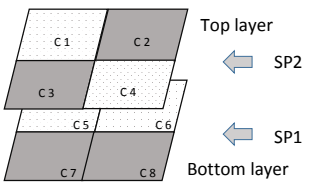

Fig. 5. An example of target 3D CMP

Parameter	Value
V_w	0.016
L_w	0.348
R_{0_amb}	3.25
R_{1_amb}	1.625

TABLE I
PARAMETERS IN THE MODEL

bottom two layers as a hyper-bottom layer. For a four-layer 3D CMP, we view the bottom two layers and the top two layers as a hyper-bottom layer and hyper-top layer, respectively. Then we can apply the same thermal rank model for tasks mapped to the hyper-bottom layer, and apply the combined power model for tasks mapped to the hyper-top layer.

V. Experimental Results

A. Experiment Setup

Benchmark Applications: We consider 7 DG based applications picked from the real-life SDFG and E3S benchmarks as taken in some prior studies[3]. In each DG, the execution cycles for node is set to the default value provided by the SDFG and E3S benchmarks. The active power consumption of each node is set randomly between $2W$ to $4W$ based on model derived in [6].

Target 3D CMPs:The parameters required in Eq. (2) and (3) are defined in Sec.B.2 and calculated based on the physical parameters and summarized in Table I. The target thermal model of 3D CMP is built using HotSpot 6.0 [1], which is a well established thermal simulation platform. The physical properties of the target 3D CMPs including the area of each block in a core are provided in table II. The other necessary parameters used in our experiments are set to default in Hotspot. For the communication protocol and routing algorithm, we refer to the work in [3]. For the leakage power of each core that is modeled by Eq.(1), we calculate the curve fitting constants $\alpha = 0.011867, \beta = -2.4722$, based on the parameters in [11] for the temperature range of 45-110°C.

Overall Experiment Flow: The experiments are conducted to compare the effectiveness of the proposed algorithm with other state-of-the-art thermal-aware approaches listed in Table III. Thermal profiling in [12] uses a representative and efficient temperature prediction model to guide the task mapping for independent tasks. PTLS in [8] targets to schedule real applications for peak temperature minimization under deadline constraint. TR in [12] and coolest-first algorithm [4] are considered for only temperature comparison, validating the TAS stage. While PTLS is considered for both performance and temperature comparison to validate the whole TSS algorithm. We measure the efficiency of approaches on two different metrics which are peak temperature and performance. We define the average execution time of all the applications as the measure of performance improvement. As for the CAG design-time step, when the application has more than 8 tasks, GA is employed. Otherwise, exhaustive exploration is employed. Moreover, the scheduling interval varies from 1ms to 2ms to explore the performance of proposed algorithm under different workload pressure. The scheduler dispatchs task graphs to the cores every scheduling interval.

B. Peak Temperature Reduction

Fig.6 shows the absolute peak temperature of 7 benchmarks at the fixed scheduling interval of 1ms when mapped on a $4 \times 4 \times 2$ CMP. To make a fair comparison, we assume that the design time makespan optimization results are available to the approaches targeting independent tasks. From the histogram, we can see that TSS can reduce the peak temperature

TABLE II
Physical Properties & Hospot Parameters

Parameter	Value
Core size $[mm]$	2×2
Processing element size $[mm]$	2×1
Network interface size $[mm]$	1×1
L2 cache size $[mm]$	1×1
Thermal interface material thickness $[um]$	20
Silicon layer thickness $[um]$	150
Ambient temperature $[^\circ C]$	45

TABLE III
Methodologies considered for comparison

Methods	Abbreviation	References
Coolest First	CF	[4]
Thermal Profiling	TR	[12]
Peak Temperature List Scheduling	PTLS	[8]
Two-stage scheduling	TSS	proposed

up to 5.0°C compared to CF, and up to 4.39°C compared to TR. Fig.7 shows the observed relative average peak temperatures at

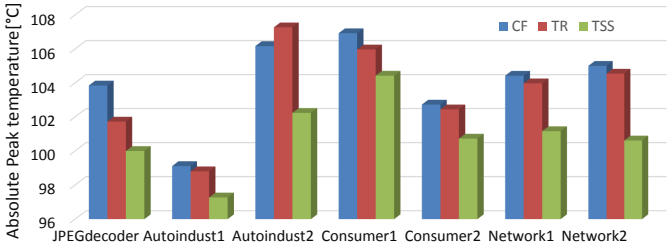

Fig. 6. Absolute peak temperature of individual benchmarks

varying scheduling interval for the benchmarks shown above. We can see that our run-time scheduling algorithm achieves significant peak temperature reduction compared to the other two algorithms. The average reduction is 3.6°C, and the maximum reduction is 6.1°C. This is due to the fact that our algorithm maps super tasks to different layers based on different decision models, dissipating as much heat as possible in bottom layer, meanwhile limiting the combined power in a stack. Such power distribution is thermal-efficient as explained in Sec. B.

C. Performance Evaluation

Fig.8 demonstrates the results of comparing our approach with PTLS of 7 benchmarks on a two-layer 3D CMP arranged in the $4 \times 4 \times 2$ pattern. The broken line demonstrates the performance improvement, which clearly shows that our algorithm can achieve an average of 4.9% performance improvement, with a maximum performance saving of 6.8%. This is owing to the fact that our design-time algorithm is effective in optimizing the makespan of task graphs based on different available cores and results in super tasks which when mapped during run-time improves the performance. Meanwhile, note that the histogram indicates that our algorithm greatly outperforms PTLS in the peak temperature as well. An average peak temperature reduction of 6.3°C is achieved. Peak temperature reduction increases with larger scheduling interval because there are more idle cores giving bigger exploration space for TSS with the increase of scheduling interval.

D. Results For Multi-layer 3D CMP

Fig.9 manifests the results of comparing TSS with PTLS on a four-layer 3D CMP arranged in the $2 \times 4 \times 4$ pattern. We

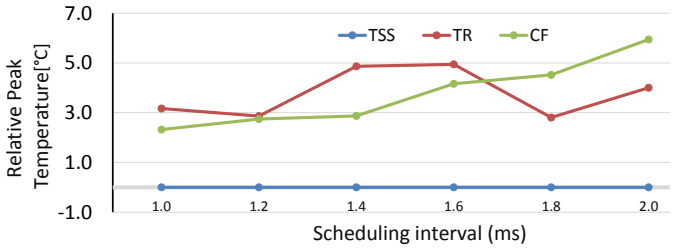

Fig. 7. Relative peak temperature on two layer 3D CMP

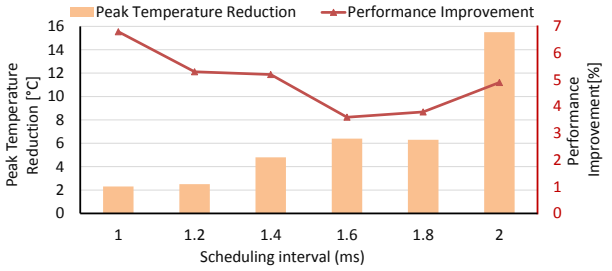

Fig. 8. Improvement compared to PTLS for two layers

use the same application set tested in the above experiments as input. We can see that TSS achieves an average peak temperature reduction of 10.0°C and an average performance improvement of 6.78% compared to PTLS. From Fig.8 and Fig.9, it is shown that greater improvement is achieved with the increasing of number of layers. This is because the thermal challenge is more severe in four-layer 3D CMP, thus TSS has bigger exploration space to reduce the peak temperature. To validate

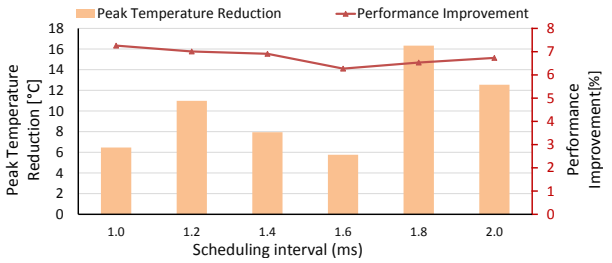

Fig. 9. Improvement compared to PTLS for four layers

our proposed TAS approach which incorporates the consideration of both thermal efficiency and power stack against the ones considering only one factor. Fig.10 gives the relative peak temperature at fixed scheduling interval of 1ms on two-layer, three-layer and four-layer 3D CMPs. Every layer is comprised of 4×4 identical cores. "Thermal rank only"means using thermal rank model for all layers, while "combined power only" means using combined power model for all layers. We can see that our proposed TAS approach outperforms them by an average of 7.31°C and 5.7°C reduction, respectively. Thus we can conclude that the extension of TAS stage approach to multi-layer 3D CMP work efficiently in peak temperature reduction.

VI. Conclusion and Future Work

This paper proposes a novel two-stage thermal-aware task scheduling algorithm to solve the thermal challenges faced by modern 3D CMP system. Our algorithm is constituted of two steps: the communication-aware group step at design-time, and thermal-aware scheduling step at run-time. The experimental results show that our algorithm outperforms other existing

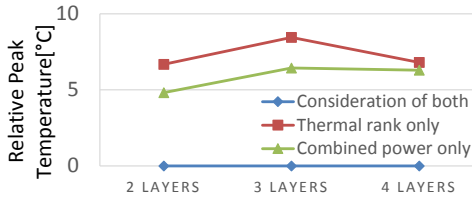

Fig. 10. Relative peak temperature on 3D CMPs with different layers

methods and achieves average peak temperature reduction up to 6.3°C. In future, we plan to apply DVFS into our thermal-aware scheduling algorithm to further reduce the peak temperature.

References

[1] Hotspot 6.0 temperature modeling tool, `http://lava.cs.virginia.edu/HotSpot/`.

[2] V. Chaturvedi et al. Thermal-aware task scheduling for peak temperature minimization under periodic constraint for 3D-MPSoCs. In *Proc. of Int. Symp. on Rapid System Prototyping*, pages 107–113, 2014.

[3] M. Cox et al. Thermal-aware mapping of streaming applications on 3D multi-processor systems. In *Proc. of Symp. on Embedded Systems for Real-time Multimedia*, pages 11–20, 2013.

[4] Y. Cui et al. Thermal-aware task scheduling for 3D network-on-chip: A Bottom-to-Top scheme. In *Proc. Int. Symp. on Integrated Circuits*, pages 224–227, 2014.

[5] K. Deb et al. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. on Evolutionary Computation*, 6(2):182–197, 2002.

[6] X. Feng et al. Power and energy profiling of scientific applications on distributed systems. In *Proc. of Int. Parallel and Distributed Processing Symposium*, pages 34–34, 2005.

[7] J. U. Knickerbocker et al. 3-D silicon integration and silicon packaging technology using silicon through-vias. *IEEE Journal of Solid-State Circuits*, 41(8):1718–1725, 2006.

[8] J. Li et al. Thermal-aware task scheduling in 3D chip multiprocessor with real-time constrained workloads. *ACM Trans. on Embedded Computing Systems*, 12(2):24, 2013.

[9] C.-H. Liao et al. An online thermal-constrained task scheduler for 3d multi-core processors. In *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 351–356, 2015.

[10] C.-H. Liao et al. Thermal-constrained task scheduling on 3-d multicore processors for throughput-and-energy optimization. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 23(11):2719–2723, 2015.

[11] W. Liao et al. Temperature and supply voltage aware performance and power modeling at microarchitecture level. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 24(7):1042–1053, 2005.

[12] S. Liu et al. Thermal-aware job allocation and scheduling for three dimensional chip multiprocessor. In *Proc. of Int. Symp. on Quality Electronic Design*, pages 390–398, 2010.

[13] Y. Liu et al. Accurate temperature-dependent integrated circuit leakage power estimation is easy. In *Proc. of Conf. on Design, Automation & Test in Europe*, pages 1526–1531, 2007.

[14] R. Rao et al. Statistical estimation of leakage current considering inter- and intra-die process variation. In *Proc. of Int. Symp. on Low Power Electronics and Design*, pages 84–89, 2003.

[15] A. W. Topol et al. Three-dimensional integrated circuits. *IBM Journal of Research and Development*, 50(4.5):491–506, 2006.

[16] T.-H. Tsai et al. Thermal-aware real-time task scheduling for three-dimensional multicore chip. In *Proc. of ACM Symposium on Applied Computing*, pages 1618–1624, 2012.

[17] C. H. Yu et al. Thermal-aware on-line scheduler for 3-D many-core processor throughput optimization. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 33(5):763–773, 2014.

[18] X. Zhou et al. Thermal-aware task scheduling for 3D multicore processors. *IEEE Trans. on Parallel and Distributed Systems*, 21(1):60–71, 2010.