Thermal-Aware On-Line Task Allocation for 3D Multi-Core Processor Throughput Optimization

Chiao-Ling Lung[1,2], Yi-Lun Ho[1], Ding-Ming Kwai[2], and Shih-Chieh Chang[1]

[1]Department of Computer Science
National Tsing Hua University
HsinChu 30013, Taiwan

[2]Information and Communications Research Laboratories
Industrial Technology Research Institute
HsinChu 31040, Taiwan

cllung0608@gmail.com, fries_421822@hotmail.com, dmkwai@itri.org.tw, scchang@cs.nthu.edu.tw

***Abstract* — Three-dimensional integrated circuit (3D IC) has become an emerging technology in view of its advantages in packing density and flexibility in heterogeneous integration. The multi-core processor (MCP), which is able to deliver equivalent performance with less power consumption, is a candidate for 3D implementation. However, when maximizing the throughput of 3D MCP, due to the inherent heat removal limitation, thermal issues must be taken into consideration. Furthermore, since the temperature of a core strongly depends on its location in the 3D MCP, a proper task allocation helps to alleviate any potential thermal problem and improve the throughput. In this paper, we present a thermal-aware on-line task allocation algorithm for 3D MCPs. The results of our experiments show that our proposed method achieves 16.32X runtime speedup, and 23.18% throughput improvement. These are comparable to the exhaustive solutions obtained from optimization modeling software LINGO. On average, our throughput is only 0.85% worse than that of the exhaustive method. In 128 task-to-core allocations, our method takes only 0.932 ms, which is 57.74 times faster than the previous work.**

Keywords: Multi-core processor, task allocation, thermal awareness, three-dimensional integration, throughput optimization, temperature uniformity.

I. INTRODUCTION

3D ICs have attracted a lot of attention recently because they provide shorter interconnect lengths, smaller form factor, higher packing density, and more flexible heterogeneous integration than their 2D system-on-chip (SOC) counterparts. Nevertheless, the higher power density induced by the closer proximity and thinner thickness of dies, together with the poor thermal conductivity insulation materials between dies, affects the heat dissipation, resulting in temperature rise. High temperature can cause many negative effects, such as expensive cooling cost, slow switching speed, IR drop, and reduced lifetime.

Temperature uniformity is another key issue in yield and reliability. Since the larger temperature gradient exists, the larger mismatch in thermal expansion appears. The thermal stress will occur where temperature uniformity is obviously adverse. In addition, the temperature uniformity induces the formation of hotspots, thermal coupling between adjacent devices, and degradation of reliability in the devices [11], [12]. Therefore, 3D IC designs need to consider performance and thermal constraints simultaneously.

Fig. 1 shows a schema of two-die stacked 3D IC. The package consists of various components including heat sink, heat spreader, thermal interface material (TIM), TSV-equipped dies, micro bumps, underfill between dies, package substrate, solder bumps, and printed circuit board (PCB). The thermal conductivities of these materials vary dramatically from 0.1 to 400 W/mK [1], [5], [6], [9]. Particularly, the insulation materials between dies that lead to dramatically different thermal behaviors. As a result, the task allocation of a 3D MCP needs to take the different characteristics of cores in different dies into consideration.

Previous works which propose task allocation methods with thermal constraints can be categorized into two types: the on-line task allocation requires a fast algorithm to integrate with the scheduler, whereas the off-line task allocation adopts more time-consuming methods for better performance. For example, the off-line method [13] attempts to reduce the peak temperature of a chip. Another off-line method [14] optimizes the throughput according to the information fed from built-in thermal sensors and performance counters.

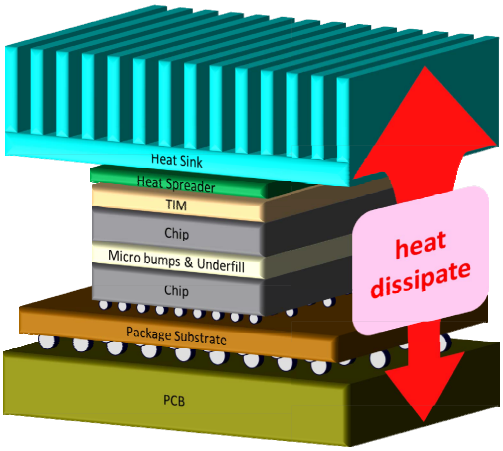

Fig. 1. An example of TSV-based 3D IC

978-3-9810801-7-9/DATE11/©2011 EDAA

The on-line task allocation method, Coolest [2], selects the coolest core for a job one by one, based on its current temperature. Adapt3D [3], balances the rise in temperature by assigning tasks according to the thermal probabilities of cores. The approach presented in [4] assumes that the majority of heat dissipates along the vertical direction and ignores lateral heat dissipation between cores of a 2D MCP. The assumption in [4] is that the thermal behaviors of cores are independent of each other and the throughput optimization can be formulated as a linear assignment problem. However, the assumption cannot directly apply to a 3D IC for that the lateral heat dissipation now is considerable.

Despite several previous works that have proposed thermal-aware task allocation algorithms, it is still difficult to achieve a high-efficiency for on-line task allocation. The major bottleneck of a high-efficiency computation is that many thermal simulations are required. In this paper, we present an extremely fast task allocation algorithm which has the same time complexity as **one** thermal simulation. The fast runtime of our approach is achieved through rewriting thermal equations so that a thermal simulation can be incrementally updated. As shown in experimental results, the runtime for 128 cores is **0.932ms** which is far less than an operation system (OS) scheduling interval (around 10ms [10]). Moreover, our fast algorithm also shows good quality results in terms of throughput.

The rest of this paper is organized as follows. Section II describes the power, thermal, and a performance model used in this paper, and gives a proper problem formulation. Section III presents the proposed task allocation algorithm. Section IV shows the experimental results. Section V contains our conclusions.

II. Problem Formulation

In this section, the power, thermal and performance models used in our method are described. It is followed by the problem formulation of throughput optimization.

A. Power Model

The total power consumed by a node comprises dynamic power and static power. The vector of dynamic power $\boldsymbol{P_{dyn}}$ can be expressed as follows.

$$\boldsymbol{P_{dyn}} = \boldsymbol{V^2 F P_{dyn}^{max}} \quad (1)$$

In (1), $\boldsymbol{P_{dyn}^{max}}$ is a vector of the maximum dynamic power when tasks run at the highest voltage and frequency. $\boldsymbol{V}$ and $\boldsymbol{F}$ are voltage and frequency matrices. Both voltage and frequency are normalized to the range [0, 1] and each voltage value has a corresponding frequency value [14].

We adopt the static power model used in [4] and [10]. The relation between the static power $\boldsymbol{P_s}$ and the temperature $\boldsymbol{T}$ is expressed as

$$\boldsymbol{P_s} = \boldsymbol{P_{s0}} + \boldsymbol{G_{LDT}}\,\boldsymbol{T} \quad (2)$$

where $\boldsymbol{P_{s0}}$ is a vector of the base static power at $\boldsymbol{T} = \boldsymbol{0}$ and, $\boldsymbol{G_{LDT}}$ is a matrix of approximate slopes extracted from the static power-temperature curve. Even though the piecewise linear approximation between static power and temperature may seem inadequate, it remains acceptable as long as the thermal simulation is within the targeted range. As far as we are aware, many previous works simply ignore the leakage power or use a single linear equation for the on-line temperature computation.

B. Thermal Model

To model the heat conduction within a packaged MCP, we adopt the HotSpot thermal RC circuit model [5], [6]. HotSpot uses the well-known duality [8] between thermal and electrical phenomena. The abilities of transferring heat and absorbing heat are analogous to resistance and capacitance in

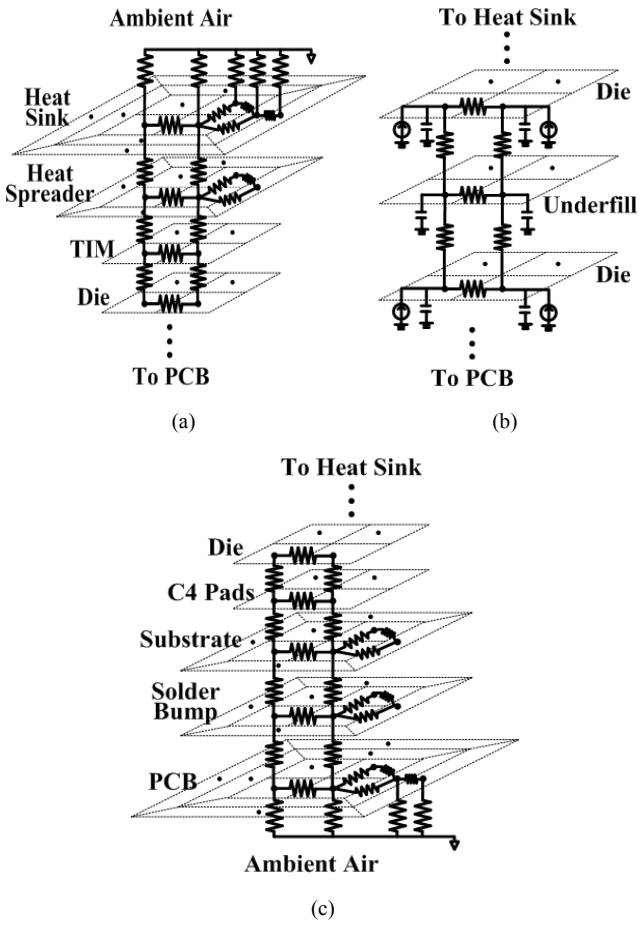

Fig. 2. RC-Equivalent thermal model of a 3D IC. (a) Heat Sink to Die. (b) Die to Die. (c) Die to PCB.

an electric circuit. The heat flow passing through thermal resistances represents current and temperature represents voltage.

Fig. 2 shows a 3D IC thermal model. We divide a die into several nodes, according to the number of grids or functional blocks on the die. The other layers are divided into nodes related to the dies. To avoid cluttering, we do not show all components. The power consumed by a node on the die is modeled as a current source connected to the corresponding node. As shown in Fig. 2, each node has a capacitance and several resistances. The nodes in the heat sink and PCB layer have convection thermal resistances which connect to ground for they maintain the ambient temperature.

Based on the RC-equivalent thermal model, we have

$$\boldsymbol{A}\ \boldsymbol{T} = \boldsymbol{P_{dyn}} + \boldsymbol{P_s} \tag{3}$$

where $\boldsymbol{A}$ is a conductance matrix and $\boldsymbol{T}$ is a vector of node temperatures. Substituting (1) and (2) into the right hand side of (3), $\boldsymbol{T}$ can be expressed as follows.

$$\boldsymbol{T} = \boldsymbol{B}\ (\boldsymbol{V^2 F}\ \boldsymbol{P_{dyn}}^{\boldsymbol{max}} + \boldsymbol{P_{s0}}) \tag{4}$$

where $\boldsymbol{B}$ is equal to $(\boldsymbol{A} - \boldsymbol{G_{LDT}})^{-1}$. In matrix $\boldsymbol{B}$, the element $b_{i,j}$ at row i and column j is the thermal coefficient, representing the temperature rise of node i caused by applying unit power on node j. Given an N-core MCP and each core containing k blocks, the total number of nodes related to the die layers will be $k \times N$. For simplicity, we choose $k = 1$ and use the core count N in the discussion that follows.

C. Performance Model

We assume that each core is a processing island whose operating frequency and voltage can be controlled independently. Without considering simultaneous multi-threading (SMT), each core can run only one single task at a time. The throughput of a 3D MCP with N cores is defined as

$$Throughput = \sum_{i=1..N}(w_i \times f_i) \tag{5}$$

where w_i and f_i are the weight and the frequency of core i, respectively. The weight is related to the task running on core i and can be determined by the instruction per cycle (IPC) or the task priority. When a core has no task assigned, the weight w_i is equal to zero.

D. Problem Formulation

We define the problem of throughput optimization under thermal constraints for a 3D MCP to be *Tout3D*. Given a 3D MCP with N cores and M tasks, our objective is to find the allocation and the voltage/frequency assignment to solve the *Tout3D* problem. It can be easily formulated as a mixed integer nonlinear programming (MINLP) problem.

$$\max \quad \sum_{i=1}^{N}(w_i \times f_i) \tag{6}$$

$$\text{s.t.} \quad \mathbf{T} = \mathbf{B}\left(\mathbf{V}^2\ \mathbf{F}\ \mathbf{L}\ \mathbf{P}_{\mathbf{dyn}}^{\mathbf{max}} + \mathbf{L}\ \mathbf{P}_{\mathbf{s0}}\right) \tag{7}$$

$$\sum_{i=1}^{N}\mathbf{L}_{i,j} \in \{0,1\}, \forall\ j \in \{1,\ldots,M\} \tag{8}$$

$$\sum_{j=1}^{M}\mathbf{L}_{i,j} \in \{0,1\}, \forall\ i \in \{1,\ldots,N\} \tag{9}$$

$$\mathbf{L}_{i,j} \in \{0,1\}\ , \forall\ i \in \{1,\ldots,N\}, \forall\ j \in \{1,\ldots,M\} \tag{10}$$

$$t_i \le T_{max}\ , \forall\ i \in \{1,\ldots,N\} \tag{11}$$

$$f_{min} \le f_i \le 1, v_{min} \le v_i \le 1, \forall\ i \in \{1,\ldots,N\} \tag{12}$$

$$v_i = f_i^{\beta}, \forall\ i \in \{1,\ldots,N\} \tag{13}$$

In the formulation, f_{min} and v_{min} are the minimum frequency and voltage constraints, and T_{max} is the thermal constraint. The elements in the allocation matrix $\boldsymbol{L}$ are described as follows.

$$\mathbf{L}_{i,j} = \begin{cases} 1, & \text{if task } j \text{ is assigned to core } i \\ 0, & \text{otherwise} \end{cases}$$

III. Task Allocation Algorithm

To solve the *Tout3D* problem, we present an effective task allocation approach in this section. Because of its fast runtime property, the proposed task allocation algorithm can be adopted in an on-line scheduler. We first explain our basic idea in Subsection III.A. An illustrative example is given in Subsection III.B and the complexity analysis are summarized in Subsection III.D.

A. Basic Idea

Before describing our algorithm, it is helpful to review the coolest approach [2] which allocates one task at a time. In each step, a thermal simulation is first performed, and then, the task in consideration is allocated to the coolest core. There are two apparent disadvantages in this approach. First, each step needs at least N^2 multiplications for a thermal simulation. Secondly, the allocation could be inaccurate. After assigning the first few tasks, most cores are still without tasks allocated. Because the thermal behavior of a core is affected by virtually all tasks assigned to other cores, this has a large deviation compared to the situation when all tasks have been allocated. A possible violation of the thermal constraint occurs when many tasks run at the same time.

To avoid an inaccurate task allocation by choosing the coolest location, here we propose an *approximated* approach as follows. Let us assume that several tasks are already allocated and we try to determine the location of a new task. We fill in all unallocated cores with the task. Fig. 4 gives an example. Consider allocating four tasks to four cores and that task 1 has been allocated. When determining the location of

task 2, we compute the thermal map by filling all remaining locations with the same task.

We will show later that the thermal simulation on such an approximated task allocation can be incrementally updated from the previous step. It can be performed much faster than a thermal simulation can. After the incremental update, we obtain the temperatures of all locations. We then allocate the task to the core which has the lowest temperature rise among all unallocated ones. The process continues until all tasks are allocated. The approximated approach has a distinct characteristic; i.e., the approximation becomes exact when the last task is assigned.

B. Task Allocation

We now describe our task allocation algorithm. To minimize the temperature of the die stack, we arrange the task powers as $p_1 \geq p_2 \geq \ldots \geq p_N$. The difference between p_m and p_{m-1} is smaller than or equal to zero; i.e., $\Delta p_{m,\,m-1} = p_m - p_{m-1} \leq 0$.

We perform an incremental update of thermal simulation on the approximated task allocation so that all cores run the same task with the largest power p_1. Then, we assign the task to the core with the lowest temperature, based on the following two reasons.

1. The remaining steps will apply $\Delta p_{m,m-1}$, for $m = 2 \ldots N - 1$, to all unallocated cores. Because all of them are non-positive, the temperature of all cores will decrease monotonically at the remaining steps.
2. The objective of our method is to minimize the hotspot temperature. We assign the task to the core with the lowest temperature. The other cores with higher temperatures will then be assigned with lower powers to achieve the reduction.

In the second step, we perform the incremental update of thermal simulation on the approximated task allocation with all unallocated $N - 1$ cores running the same task with power of p_2. We assign the task to the core with the lowest temperature. The process continues until all tasks are allocated.

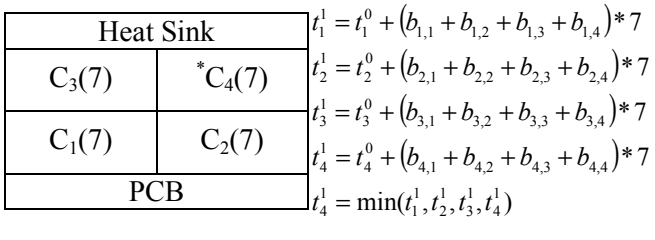

Fig. 3. The first step of the example.

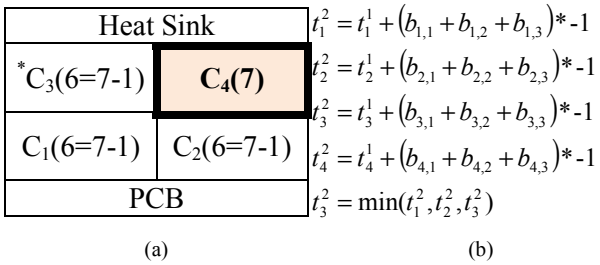

Fig. 4. The second step of the example.

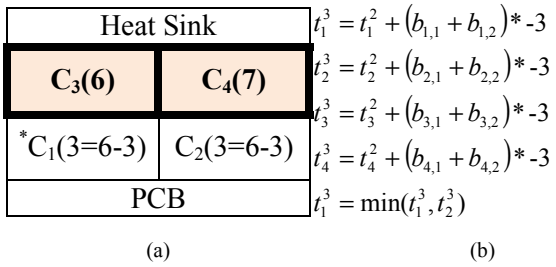

Fig. 5. The third step of the example.

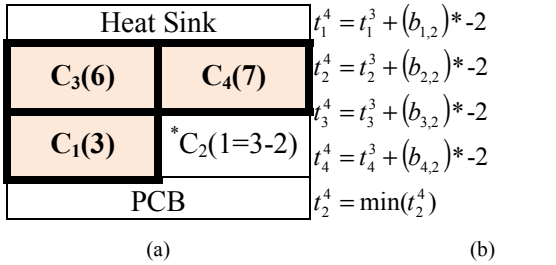

Fig. 6. The fourth step of the example.

C. Illustrative Example

In this section, we demonstrate our overall algorithm using an example. We attempt to assign four tasks to a two-die stack of 3D MCP with four cores. Given that the initial temperature of core C_i is t_i^0 and the powers of the four tasks are $\boldsymbol{P} = \{7, 6, 3, 1\}$. The proposed task allocation algorithm is performed in the following steps.

In the first step, we apply $p_1 = 7$ to all cores as shown in Fig. 3(a). Then, we obtain core temperatures t_i^1 by computing the equations shown in Fig. 3(b). We assign the task to core C_4 which has the lowest temperature among all cores. It is marked with a "*" in Fig. 3(a).

The second step is shown in Fig. 4(a) and the bold border represents C_4 is already allocated with the task whose power is 7. We apply $\Delta p_{1,2} = -1$ to all unallocated cores. By computing the equations listed in Fig. 4(b), we obtain the core temperatures t_i^2 and assign the task to core C_3 which has the lowest temperature among all unallocated cores.

Similarly, Fig. 5 and Fig. 6 show the remaining two steps to complete the thermal simulation and task allocation. Fig. 5 shows the third step, in which $task_3$ is assigned to $core_1$, and Fig. 6 shows the fourth step, in which $task_4$ is assigned to $core_2$.. In addition, the final temperature of C_i is equal to t_i^4. We prove the correctness of the incremental updates by deriving the final temperature as follows.

$$
\begin{aligned}
t_1^4 &= t_1^3 + (b_{1,2}) * -2 \\
&= (t_1^2 + (b_{1,1} + b_{1,2}) * -3) + (b_{1,2}) * -2 \\
&= t_1^2 + b_{1,1} * -3 + (b_{1,2}) * -5 \\
&= (t_1^1 + (b_{1,1} + b_{1,2} + b_{1,3}) * -1) + b_{1,1} * -3 + (b_{1,2}) * -5 \\
&= t_1^1 + b_{1,1} * -4 + b_{1,2} * -6 + b_{1,3} * -1 \\
&= (t_1^0 + (b_{1,1} + b_{1,2} + b_{1,3} + b_{1,4}) * 7) + b_{1,1} * -4 + b_{1,2} * -6 + b_{1,3} * -1 \\
&= t_1^0 + b_{1,1} * 3 + b_{1,2} * 1 + b_{1,3} * 6 + b_{1,4} * 7
\end{aligned}
$$

D. Overall Algorithm and Complexity Analysis

The time complexity of the proposed task allocation in terms of the number of multiplications is analyzed as follows. For ease of explanation, we only consider the nodes related to the die layers in the previous example. If each core has k nodes, then updating a core temperature requires k^2 multiplications. For N cores, the total number of multiplication required for solving t_i^x is k^2N. The procedure of incremental task allocation takes N steps; therefore, the total number of multiplications is k^2N^2. Notice that a complete thermal simulation requires the same number of multiplications.

Although we use the assignment of N tasks to N cores as an example, it is easy to see that the proposed approach can be applied to the situation when certain cores have unfinished tasks. For this situation, those cores are simply treated as allocated cores and new tasks are assigned according to our approach.

IV. Experimental Results

To solve the *Tout3D* problem in Section II, we implement a framework integrating both our incremental task allocation (ITA) method and the extended on-line optimal voltage and frequency assignment algorithm. We assume that each task is running at the maximally allowable voltage and frequency at first. All tasks are allocated by our algorithm presented in Section III. After the task allocation, if there is a core whose temperature exceeds the thermal constraint, the procedure of voltage and frequency assignment is invoked. The optimal voltage and frequency of each core can be solved.

We consider a sixteen-core in a four-die stack of 3D MCP. In the given architecture, each die contains four Alpha 21264 microprocessor cores, each occupying the area of 7.95mm × 7.95mm. The material parameters of the die stack are shown in Table I. Convection thermal resistances of the heat sink and PCB are 0.01°C/W and 1°C/W, respectively. The thermal RC model is extracted from HotSpot5.0 by applying the material parameters listed in Table I. The thermal constraint is set to 110°C and the range of frequency is from 1.5GHz to 3GHz. The power consumptions of tasks are obtained from [10] and those tasks are selected from SPEC CPU2000 benchmarks.

Table I. The material parameters

Layer	Conductivity (W/mK)	Thickness (mm)	Width (m)	Height (m)
Heat Sink	400	6.9	0.06	0.06
Heat Spreader	400	1	0.03	0.03
TIM	4	0.02	0.159	0.159
TSV-equipped Dies	142.8	0.05	0.159	0.159
Underfill	1.25	0.01	0.159	0.159
C4 Pads	2.5	0.1	0.159	0.159
Package Substrate	2	1	0.021	0.021
Solder Bumps	16.666	0.94	0.021	0.021
PCB	3	2	0.1	0.1

We consider five cases, each of which has sixteen tasks with different power profiles.

To demonstrate the effectiveness of our method, we obtained an optimal solution from the MINLP formulation listed in Section II using LINGO. Since the runtime to obtain such an optimal solution is relatively long, we have limited the runtime to be one day. For comparison, two previous works are implemented as following descriptions. According to the current temperature, the on-line task allocation method Coolest [2] selects the coolest core for a job in the job queue. The task-to-core allocation algorithm proposed by [4], called DAC09, assigns tasks based on the following two assumptions. First, since the temperatures of caches are relatively low and the cores are surrounded by L2 caches, the heat dissipation path between the cores can be neglected. Second, the lateral thermal resistances can be ignored, due to the relatively lower lateral thermal resistances, and the larger thermal gradient in vertical direction. Based on the assumptions, the voltage and frequency of a core running a task can be computed independently. Therefore, a linear assignment problem is formulated and solved in polynomial time.

Table II. Throughput, and runtime comparison

Case	Throughput			
	LINGO	Coolest	DAC09	Our ITA
Case1	12.99 (1.01X)	10.98 (0.86X)	10.23 (0.80X)	12.82 (1 X)
Case2	12.24 (1.01X)	9.65 (0.79X)	9.01 (0.74X)	12.15 (1 X)
Case3	10.79 (1.00X)	8.29 (0.77X)	8.02 (0.75X)	10.74 (1 X)
Case4	12.02 (1.01X)	10.01 (0.84X)	9.32 (0.79X)	11.85 (1 X)
Case5	11.72 (1.00X)	10.06 (0.86X)	9.31 (0.80X)	11.7 (1 X)
AVG	**11.95 (1.01X)**	**9.8 (0.82X)**	**9.18 (0.77X)**	**11.85 (1 X)**
Runtime (ms)	**1 day**	**0.13 (6.74X)**	**0.31 (16.32X)**	**0.019 (1X)**

Throughput, maximal temperature difference, and runtime of 16 cores and their comparisons are shown in Table II. The first column shows the name of the test case. The second column shows the local optimal solution generated by LINGO. The third to sixth columns are the throughputs (penalties compared to our approach) of Coolest, DAC09, and ITA, respectively. The seventh column shows the maximum temperature differences by LINGO. The eighth to eleventh columns are the maximum temperature differences of Coolest, DAC09 and ITA, respectively. The last row shows the runtime of each algorithm. Because the runtime depends solely on the number of cores and is independent of the power profile of a test case, each algorithm has a runtime in the condition of 16 cores.

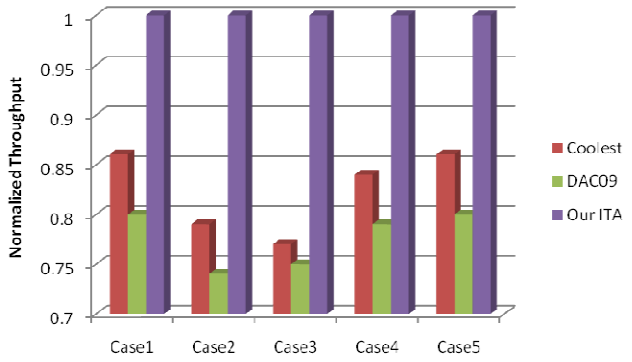

Fig. 7. Cmparison of normalized throughput.

The average throughput of our ITA algorithm is 11.85, whereas that of LINGO is 11.95. Compare to LINGO, ITA improves the runtime from one day to 0.019ms and only costs a trivial throughput penalty (less than 1%). On average, the throughput penalties of Coolest and DAC09 are 18% and 23%, respectively. Furthermore, the runtime of Coolest and DAC09, respectively, are 6.74 time and 16.32 times larger than ITA. Fig. 7 shows the throughput, which is normalized to that of ITA, in a bar chart.

To demonstrate the fast runtime of our algorithm, we also created a test case of 128 cores. To finish the task allocation for 128 cores, Coolest takes 53.814ms and DAC09 takes 5.978ms. Our algorithm can finish in **0.932ms**. To allocate 128 tasks, the runtime of Coolest and DAC09 are 57.74 times and 6.41 times larger, respectively, than that of ITA. In addition, to exhibit the throughput impact caused by neglecting the heat removed from dies to the PCB, known as the second path effect, we also implement our task allocation algorithm without considering the effect. The result shows that the cost of ignoring the second path effect results, on average, is 5.73% throughput underestimation.

V. CONCLUSIONS

In this paper, we proposed a highly efficient task allocation algorithm to solve the *Tout3D* problem by utilizing incremental updates of thermal simulation. As a result of the fast runtime property, the proposed algorithm is able to realize the task allocation for 128 cores within **1ms** which enables the integration of our approach into an on-line thermal-awareness scheduler. And, even more importantly, ITA has shown high quality results in terms of throughput.

ACKNOWLEDGMENT

This work is partially support by Industrial Technology Research Institute (ITRI) under Contract 52-99S3-01.

REFERENCES

[1] J. H. Chien, C. L. Lung, C. C. Hsu, Y. F. Chou, and D. M. Kwai, "Floorplanning 1024 cores in a 3D-stacked networkon- chip with thermal-aware redistribution," *12th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems* (*ITherm*), 2010, pp. 1–6.

[2] A. K. Coskun, T. S. Rosing, K. A. Whisnant and K. C. Gross, "Static and dynamic temperature-aware scheduling for multiprocessor SoCs," *IEEE Trans. VLSI Systems*, vol. 16, no. 9, pp. 1127–1140, 2008

[3] A. K. Coskun, T. S. Rosing, J. Ayala, D. Atienza, and Y. Leblebici. "Dynamic thermal management in 3D multicore architectures," *in Proc. Design Automation and Test in Europe Conf.*, 2009, pp. 1410–1415.

[4] V. Hanumaiah, R. Rao, S. Vrudhula and K. S. Chatha, "Throughput optimal task allocation under thermal constraints for Multi-core Processors," *in Proc. Design Automation Conf.*, 2009, pp. 776–781.

[5] W. Huang, K. Sankaranarayanan, R. J. Ribando, M. R. Stan and K. Skadron, "An improved block-based thermal model in HotSpot 4.0 with granularity considerations," *in Proc. Workshop Duplicating, Deconstructing, and Debunking*, June 2007.

[6] W. Huang, K. Skadron, S. Gurumurthi, R. J. Ribando, and M. R. Stan, "Differentiating the roles of IR measurement and simulation for power and Temperature-Aware Design," in *Proc. ISPASS*, Apr. 2009, pp. 1–10.

[7] M. Kadin and S. Reda, "Frequency and voltage planning for Multi-Core Processors Under Thermal Constraints," in *Proc. Int. Conf. Computer Design*, pp. 463–470, 2008.

[8] A. Krum, "Thermal management," in *The CRC handbook of thermal engineering*, F. Kreith ed., Chap. 2, CRC Press, 2000.

[9] C. L. Lung, Y. L. Ho, S. H. Huang, C. W. Hsu, J. L. Liao; S. Y. Huang, and S. C. Chang, "Thermal analysis experiences of a tri-core SoC system," *International Conference on Green Circuits and Systems* (*ICGCS*), 2010, pp. 1–6.

[10] R. Rao and S. Vrudhula. "Efficient online computation of core speeds to maximize the throughput of thermally constrained multi-core processors," *in Proc. ICCAD*, pp. 537–542, 2008.

[11] N. Rinaldi, "On the modeling of the transient thermal behavior of semiconductor devices," *IEEE Trans. Electron Devices*, vo1. 48, no. 12, pp. 2796–2802, 2001.

[12] K. Sheng, S. J. Finney, and B. W.Williams, "Thermal stability of IGBT high-frequency Operation," *IEEE Trans. Industrial Electronics*, vo1. 47, no. 1, pp. 9–16, 2000.

[13] C. Sun, L. Shang, and R. P. Dick, "Three-dimensional multiprocessor system-on-chip thermal optimization," in *Proc. Int. Conf. Hardware/ Software Codesign System Synthesis*, pp. 117–122, Oct. 2007.

[14] C. Zhu, Z. Gu, L. Shang, R. P. Dick, and R. Joseph, "Three-dimensional chip-multiprocessor run-time thermal management," *IEEE Trans. CAD*, vol. 27, no. 8, pp 1479–1492, Aug. 2008.