A Task Migration Technique for Temperature Control in 3D NoCs

Neda Hassanpour and Shaahin Hessabi  
Dept. of Computer Engineering  
Sharif Univ. of Technology  
Tehran, Iran  
nhassanpour@ce.sharif.edu, hessabi@sharif.edu

Parisa Khadem Hamedani  
Dept. of Electrical and Computer Engineering  
Univ. of Toronto  
Toronto, Canada  
parisa.khademhamedani@mail.utoronto.ca

Abstract—Combination of 3D IC technology and network on chip (NoC) is an effective solution to increase system scalability and also alleviate the interconnect problem in large scale integrated circuits. However, due to the increased power density in 3D NoC systems and the destructive impact of high temperatures on chip reliability, applying thermal management solutions becomes crucial in such circuits. In this paper, we propose a runtime distributed migration algorithm based on game theory to balance the heat dissipation among processing elements (PEs) in a 3D NoC chip multiprocessor (CMP). The objective of this algorithm is to minimize the 3D NoC system’s peak temperature, as well as the overhead imposed on chip performance during migration. Due to the high thermal correlation between adjacent PEs in the same stack in 3D NoCs, we model this multi-objective problem as a cooperative game. Simulation results indicate up to 23% and 27% decrease in peak temperature, for the benchmarks that have the highest communication rate and largest number of tasks, respectively. This comes at the price of slight migration overhead in terms of power-delay product (PDP).

Keywords-3D NoC; task migration; game theory

I. INTRODUCTION

With ever growing computational demands, the need to integrate large numbers of transistors on a single chip seems unavoidable. Recent advances in manufacturing technology has helped designers to move one step closer to achieving main goals such as increasing integration density and application performance [1]. Since interconnects do not follow the same delay trend as transistors, in sub-micrometer design era the delay and power consumption generated by interconnects cannot be neglected [2]. To overcome the interconnect problem, novel approaches have been proposed. Network on chip (NoC) is a reliable communication infrastructure to provide power- and delay-efficient data transfer between different components on a chip. 3D die stacking is another helpful technology with the ability to decrease delay of global wires through a chip. This capability is accomplished by connecting vertical stacked layers upon each other using shorter vertical connections, named Through Silicon Via (TSV) [3] [4]. Merging NoC and 3D integrating design paradigms, which leads to the 3D NoC architecture, enables designers to take advantage of various features of these two different approaches [5]. Despite all advantages of 3D NoC design, such as increased device density, the ability to integrate different technologies on a single chip, reduced delay and power consumption in long global wires, it has one main drawback. The key challenge in 3D NoC technology is the increased power density per unit area of the chip compared to 2D NoCs. This problem raises the concern of thermal issues within a 3D NoC chip, and makes thermal management more essential in such architectures [6]. High peak temperature and imbalanced heat dissipation can seriously affect the performance and reliability of a chip. So, despite all advantages of 3D NoC design, the reliable and efficient functioning of 3D NoCs cannot be achieved without employing an operational thermal control solution [7].

To propose a forceful thermal control method in 3D NoCs, we should consider this architecture’s particular features as listed below [1] [2].

First, due to the high possibility of overheating in 3D NoCs, we need a continuously running distributed thermal management technique with low performance overhead. In this case, by identifying the rising hotspots and taking a proper action as soon as possible, any serious damage to the system can be avoided.

Second, the processing elements (PEs) in different layers of 3D NoC have heterogeneous thermal characteristics. Their cooling efficiencies are highly related to their distance from the heat sink. The farther from the heat sink, the lower is the cooling efficiency.

Third, vertical adjacent PEs have high thermal correlation, which is stronger than that of the PEs in the same layer.

What we need is a dynamic thermal management technique based on the characteristics of 3D NoCs with the least impact on system’s performance. In other words, we need a method to optimize the performance–temperature multi-objective problem. According to the 3D NoC’s special features, using the same 2D NoC thermal control techniques may not be helpful for 3D NoCs. Moreover, employing hardware forms of dynamic thermal management, like DVFS or stop-go, cannot be efficient in 3D NoCs, either.

This is due to significant performance degradation imposed on system after applying such hardware methods with high frequency [8]. Thermal control policy based on task migration is another method to balance heat dissipation among different PEs in a multiprocessor chip [9].

In this paper, we propose the use of game theory to implement a runtime distributed migration algorithm in software level, as a thermal management technique for 3D NoCs. The major objective of this algorithm is to minimize
the 3D NoC system’s peak temperature as well as limiting the overhead imposed on chip performance during migration. In this regards, we model this multi-objective problem as a cooperative game, considering each hot core as a player. The proposed algorithm tries to find proper migration destinations for all hot cores in parallel. To restrict the migration overhead, we use temperature prediction to avoid deficient migrations which can raise new hotspots in destination cores. Furthermore, the algorithm considers tasks’ communication weights, trying to keep tasks with high intercommunications close to each other after migration. This consideration results in lower communication cost.

The rest of the paper is organized as follows. In Section II, we discuss previous related works in 3D NoC thermal management. In Section III, we present the basic concepts of game theory. Section IV is dedicated to our proposed migration algorithm. Simulation environment and experimental results are presented in sections V and VI, respectively, and the paper is concluded in Section VII.

II. RELATED WORK

Few works have previously employed thermal aware routing or task scheduling algorithms to alleviate the heat generation problem in 3D NoC systems. The presented method in [3] is a run time thermal management scheme that uses a new, but non-minimal, routing technique, named traffic-aware downward routing. This routing algorithm is adapted to the 3D NoC’s specific characteristics in order to take advantage of these features for reducing heat imbalance in a chip. In [4], a thermal management solution through task scheduling is proposed, which considers both thermal constraints and data delay access. Concentrating on reducing the execution time of tasks, this method tries to increase system’s performance by avoiding placement of a task far from its data during scheduling time, but it does not have any emphasis on reducing the temperature as much as possible. An offline thermal and communication aware mapping method is introduced in [5], which uses Genetic Algorithm in 3D NoCs. In addition, the proposed method in [10] is an ILP-based static thermal-aware mapping algorithm for 3D NoCs to meet system’s thermal constraints. In this paper, we use the concepts of game theory to propose a thermal and communication aware task migration algorithm, as a heat dissipation control tool in 3D NoC systems. To the best of our knowledge, no work has been reported before to employ task migration as a thermal management technique in 3D NoCs, with the first priority on maximizing peak temperature reduction.

III. PRELIMINARIES

A. Game Theory

In this section we present the basic concepts of game theory, which is necessary to understand our method. Game theory was introduced by Von Neumann and Morgenstern [11] for the first time, and is considered as a branch of applied mathematics. Its main objective is to model the interaction between rational agents or decision makers, each viewed as a player in a game. A game is composed of a number of rational players, each having a set of possible actions. Taking each action can produce a specific outcome for a player. Although every player’s aim is to execute the best action — the one that brings the most profit — the final success of each individual is not related to his own actions only, but also depends on the choices taken by other players [12].

B. Cooperative Games

Game theory is divided into two main branches, i.e., non-cooperative and cooperative games. Basically, in a non-cooperative game each rational player individually selects an action from its predefined set, trying to maximize its outcome [13]. In other words, in this type of game the individual profit has priority over the global outcome. By contrast, in cooperative games, global outcome is prioritized over individual benefit. Players are allowed to form groups, which are referred to as coalitions. Coalitions try to enforce the cooperative behavior. So, the game is a competition between coalitions of players rather than between individual players. The members of the coalition interact with each other before making the group’s final decision. The outcome of a cooperative game is specified by the joint actions that the group takes. The main objective in a cooperative N-player game is to form a coalition to maximize the overall outcome. In a grand coalition, all N players accept to cooperate in one union to achieve this goal [14].

C. Using Game Theory in Multiprocessor Chips

There are always some conflicting constraints in multiprocessor systems. As an example, we can refer to the contrast between performance and power consumption while adjusting the processors’ frequency in a multiprocessor chip. In order to achieve a global optimization in such systems with multi-conflicting objective functions, we need a dynamic distributed technique. Game theory concepts can help designers to reach this goal. As an example, a game theory-based approach can be used to adjust the frequency of each PE in an MPSoc to optimize the power consumption and the task synchronization in the system [11]. Applying game theory-based methods is a practical power management tool in real time embedded systems [13]. It can also be helpful for task scheduling to achieve simultaneous optimization of performance and energy [15]. To employ game theory for solving multi-objective problems in multiprocessor chips, each PE can be considered as a player. The objective functions which are defined for each processor should be compatible with overall objective of the system. In this manner, the attempt of each PE to optimize its own objective function by adjusting a specific variable results in global optimization for the entire system.
Although most case studies mentioned above have concentrated on non-cooperative games, some references suggest that cooperative games could lead to greater outcomes with the cost of small overhead increase [16]. Besides, all these works apply game theory in 2D systems. In the next section we illustrate our proposal to employ cooperative game model for solving thermal issues in 3D NoCs.

IV. THE PROPOSED ALGORITHM

Our goal is to use task migration as a dynamic thermal management technique in 3D NoCs and to implement it with the aid of game theory. The optimization problem that we are going to solve by game theory is a tradeoff between two goals. As the first goal, we attempt to maximize the peak temperature reduction by employing migration. In the meanwhile, as the second goal, we want to minimize the performance overhead imposed on the system due to migration. This overhead is composed of three parts as follows.

1. The first part is related to the change in distance between the migrated task and the tasks it needs to communicate with, after migration. We try to limit this part with taking into account the task communication weights in order to find proper destination cores.

2. The second part is caused by repeated migrations, which can happen if the migration process chooses an improper destination. In that case, the destination position can become a new hotspot. We use temperature prediction to prevent such migrations.

3. The third part is due to transferring the migrated task’s data and code from the current position to the new destination core. We use BookSim simulator [17] to estimate this part and report it in the form of power-delay product (PDP). Avoiding inefficient migrations can also be beneficial to limit the third part overhead.

A. Type of Game

By checking core temperatures in specified intervals, new hotspots can be rapidly determined. Upon detecting all hot cores in an interval, the migration algorithm applies to all of them simultaneously to avoid any damage to the system. Due to the high temperature dependency that exists among vertical adjacent cores, assigning a destination core to a hot PE can affect the remaining assignments. Therefore, to prevent the occurrence of repeated non-efficient migrations, it is crucial to inform each hot PE of the others’ choices. Furthermore, this idea can prevent selection of one core for more than one hot PE as a destination. Game cycles will repeat until an appropriate destination is found for each hot core. Therefore, we choose a simultaneous repeated cooperative game model to implement the migration method. To minimize the communication overhead between PEs in each game cycle, we limit the circulated data between all players to a list of found proper destinations for hot cores.

B. Problem Formulation

Similar to [10], we make some assumptions about the systems that are to be managed by our algorithm. The system is composed of similar processing elements, which is a common assumption in chip multiprocessors. As a result, none of them is specialized for a particular task. Furthermore, we assume that the applications that run on the chip can be broken down into tasks with known communication requirements.

The aim of all hot PEs is to find the best migration destinations to solve the optimization problem in the 3D NoC system. In order to quickly detect new hotspots in the system, the algorithm checks the temperature of cores that have been measured by sensors in specified intervals. To consider the effects of thermal capacitances on PEs temperature, we use HotSpot thermal model [18] as a sensor in our simulations. Having the list of cores’ temperatures, we use an existing thermal model [10] [19] as shown in (1) to estimate the power consumption of each core in 3D NoC. The specific characteristic of 3D NoC, i.e., the high thermal dependencies between vertical adjacent cores, has been considered in this thermal model.

\[
Th_{i,j,k} = T_{Amb} + \sum_{m=1}^{R_{i,j,m}} \frac{R_{i,j,m}}{A} \times \left( \sum_{s=m}^{n} \left( P_{i,j,s} + PR_{i,j,s} \right) \right) \tag{1}
\]

The parameters in 3D NoC structure are defined as follows: \(Th_{i,j,k}\) shows the temperature of the PE at position \((i, j, k)\). The ambient temperature is shown with \(T_{Amb}\), and \(n\) is the number of vertical layers in 3D NoC. \(R_{i,j,m}\) represents thermal resistance of the PE at position \((i, j, m)\). The area of each PE is shown with \(A\). \(P_{i,j,s}\) is the power consumption of the PE at position \((i, j, s)\), which is calculated based on the power consumption of the task that runs on it. In order to evaluate the proposed algorithm, the power for each task is estimated based on the average power consumption of the core (at the range of 10 to 60 W/cm²) at the 45 nm technology node [10] [20]. \(PR_{i,j,s}\) is the power consumption of the router at position \((i, j, s)\), and is calculated based on the router power consumption per bit and the amount of data which is routed from this router [10]. We use an estimate from the Orion2 library [21] to quantify this parameter.

Each core in 3D NoC takes necessary information, such as the lists of initial system’s task assignments, cores’ temperatures, tasks’ communication weights, tasks’ and routers’ power consumptions as input, and starts to run the steps of the proposed algorithm. Fig. 1 shows the flow chart that explains the functioning of the algorithm during its execution in the system. In order to locate hot cores in the system, all core temperatures are compared to a thermal threshold \(TH_{t}\). \(TH_{t}\) is defined as the maximum tolerable PE temperature in 3D NoC. In case any hotspot is detected in the system, all detected hot cores start the game in parallel. Each hot PE tries to find the best migration destination for its current task, and pursues the following process in every game cycle.

**First step.** Each player finds all possible migration destinations for the task being processed. A possible
migration destination should meet two thermal conditions. First, exchanging the task running on the hot PE with the one running on the candidate migration destination is not allowed to increase the temperature of currently cool PE to more than TH1. The second condition is that it should keep the present hot core temperature below a predefined threshold TH2. To consider the effect of thermal capacitance as well as the base temperature in the hot core, we select TH2 such that the constraint TH2 < TH1 is held. By using temperature prediction, at the end of this step, a set of all valid destination candidates is obtained for each player. To find a list of low power tasks suitable for exchange, and also to minimize computation overhead, we sort the tasks based on their power consumption in ascending order. Starting with the task with the lowest power consumption in the sorted list, we continue to check the TH2 thermal constraint until the first element in the list violates this condition. Up to this point, the core corresponding to each task that also verifies the thermal condition TH2 will be added to a set named γ as a valid candidate for exchange. In other words, the cores that are added to γ are the thermally safe destinations for migration.

**Second step.** Each hot core determines an optimal position for migration destination without considering the thermal constraints. This position is designated based on the task running on each hot PE and the weight of its thermal constraints. This position is designated based on the select performance overhead as the tasks communication performance overhead after migration. In this work, we use (2) and its derivatives – (3), (4) and (5) – to find such optimal position. In (2), \((x_i, y_i, z_i)\) represents the current position of the task running on the hot PE, and \((x_i, y_i, z_i)\) is the coordination of the optimal migration position, which we name it \(\beta\). The parameter \(w_{T_i,T_j}\) is the communication rate between the tasks \(T_i\) and \(T_j\).

\[
L = \sum_{i\neq j} w_{T_i,T_j}((x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2) \tag{2}
\]

\[
\frac{\partial L}{\partial x_i} = 0, x_i = \frac{\sum_{j=1}^{n} w_{T_i,T_j} x_j}{\sum_{j=1}^{n} w_{T_i,T_j}} \tag{3}
\]

\[
\frac{\partial L}{\partial y_i} = 0, y_i = \frac{\sum_{j=1}^{n} w_{T_i,T_j} y_j}{\sum_{j=1}^{n} w_{T_i,T_j}} \tag{4}
\]

\[
\frac{\partial L}{\partial z_i} = 0, z_i = \frac{\sum_{j=1}^{n} w_{T_i,T_j} z_j}{\sum_{j=1}^{n} w_{T_i,T_j}} \tag{5}
\]

**Third step.** Finally, each player selects the best option in its γ set which is determined as the set of candidate migration destinations at the first step. The best option as a migration destination is the one which minimizes performance overhead after migration. In this work, we select performance overhead as the tasks communication cost. The objective function of each hot core is the square root of F as described in (6), which calculates the distance between optimal destination point β in the position \((x_i, y_i, z_i)\) and any candidate core \(C_k\) in γ with the coordination \((x_k, y_k, z_k)\), as a destination. Each player tries to minimize this function by choosing the closest thermally suitable PE to its optimal migration position. Therefore, the best option for migration destination is the closest core to the optimal point obtained in step two, which satisfies both thermal constraints defined in step one. It should be noted that we set the temperature thresholds TH1 and TH2 to the proper values to guaranty system’s thermal safety after picking up any selected PE in the set γ as a final destination for migration.

\[
F = (x_i - x_k)^2 + (y_i - y_k)^2 + (z_i - z_k)^2 \tag{6}
\]

After each hot core specifies the best possible destination for migrating its task, players should communicate in order to get informed of each other’s choices. This communication is necessary to prevent selection of the same destination core for more than one hot PE during migration. If there is no overlap among selected destinations, all task migrations will be performed in parallel for the hot cores, and the game will be over. In the case of overlap, we will pursue the following policy. Due to the main objective of the thermal management algorithm, i.e., alleviating the peak temperature in the system, we choose the hottest core with the most critical thermal condition to execute its migration towards the selected destination. It is the end of the current

![Flow chart for the proposed algorithm](image-url)
game cycle. Due to possible effect of the occurred migration on other hot cores’ decisions for determining a destination for migration, the information of each remaining hot core, such as the task mapping and tasks’ power list, will be updated. The thermal dependency between PEs in the same vertical stack makes this updating more essential. All remaining hot PEs will then repeat the three steps of the game with the updated information. Repeating the game cycles will continue until it reaches an overlap-free condition for the rest of the hot spots. At this time, the game comes to end.

The output of the game is a new task mapping for the 3D NoC. According to the previously discussed analysis we expect the new task mapping to be capable of improving heat balancing in the 3D NoC system. It should be noted that we consider the employed routing algorithm in 3D NoC as an adaptive routing algorithm. So, migration of tasks to different positions of the network would not cause any bandwidth problem.

C. Game Convergence

By applying the proposed algorithm in a 3D NoC system with N total cores and M existing hotspots in a specific time interval, in the worst case scenario the overlap situation occurs in every game cycle, which means only the hottest core is allowed to migrate its related task. By exploiting temperature prediction in each game cycle, which prevents creation of new hot cores, the maximum number of game cycles that would be needed for solving all hot PEs’ problems and achieving game convergence will be equal to M. According to the simulation results for applying the proposed thermal management algorithm to different types of benchmarks, we can conclude that the game converges to the solution point at most in four game cycles.

V. SIMULATION ENVIRONMENT

In order to evaluate the performance of our algorithm, we used BookSim simulator and HotSpot thermal model to simulate a real system.

A. BookSim Simulator

We implemented the proposed algorithm in C++ and used BookSim to simulate the 3D NoC and calculate the network’s delay and power consumption. Using BookSim augmented with Orion2 library, we modeled the power consumption of NoC’s routers and interconnections at the architectural level. Since Power modeling in Orion2 has been designed for 2D NoCs, the data and the method used in this library are not adequate for modeling the 3D NoC power. Therefore, we exploited a modified version of orion2 [10], which takes into account the extra technology information – such as TSV area – to enhance the precision of power consumption modeling in 3D systems. We also used BookSim to measure the delay of NoC and to report its performance as PDP. We used the 45 nanometer technology parameters for the NoC with 2 GHZ router operating frequency and 128 flit size. Each simulation runs for 3 million cycles, which specifies the length of interval between two consecutive temperature measurements.

B. HotSpot Thermal Model

We used HotSpot 5.2 and its 3D stacking feature in order to accomplish thermal simulations and calculate temperatures in 3D NoC. HotSpot thermal model – which has the ability to consider the effects of thermal capacitances in temperature calculation – takes a power trace file and floorplan information as inputs, and returns the corresponding temperatures as outputs. To enhance the precision of temperature calculation, all simulations have been performed with the grid model. To adjust HotSpot parameters, we used constants listed in Table I. The complete technology information, which is also used to configure HotSpot, is represented in Table II. We exploited HotSpot as a thermal sensor as well as a temperature predictor in our simulations.

VI. EXPERIMENTAL RESULTS

To evaluate the performance of our algorithm, we used various types of benchmarks with different numbers of tasks and communication rates between them. The algorithm used a 3D NoC mesh structure [3] [4]. The benchmarks information and the size of the 3D NoC for each of them are listed in Table III.

A. Thermal Thresholds

Considering the high base temperature in a hot core, if we increase $TH_1$ up to $TH_2$, the thermal capacitance effect on core temperature is not taken into account. On the other hand, decreasing this parameter leads to reduction in the number of thermally safe candidates as migration destination, which limits the algorithm’s choices. To observe the effect of thermal capacitances on core temperature, we exploit (1). Using the thermal model in this equation, we estimated cores’ temperatures without taking into account the impact of thermal capacitance. By comparing resulted temperatures of HotSpot and those of the thermal model in (1) with the same parameters as inputs, we observed near 10 $^o$C temperature differences on average. As a result, after defining the quantity of $TH_1$ for benchmarks, we set the thermal threshold $TH_2$ to $TH_1-10$ $^o$C.

For all benchmarks except GSM_VC, the thermal control thresholds $TH_1$ and $TH_2$ are set to 80 $^o$C and 70 $^o$C, respectively. Because of the high number of tasks and the large number of vertical layers in the 3D NoC which GSM_VC is mapped on, the average temperature of the cores would be higher for this benchmark. So, we increased the values of thermal thresholds in GSM_VC to 90 $^o$C and 80 $^o$C as $TH_1$ and $TH_2$, respectively.

B. Peak Temperature

The main goal of our algorithm is to reduce the peak temperature in order to mitigate thermal problems. Fig. 2
compares the peak temperature before and after employing the proposed algorithm in different types of benchmarks. As shown in this figure, increasing tasks’ communication weight, number of tasks and also the number of vertical layers, result in higher PEs’ temperature in 3D NoC. This is why benchmarks such as GSM_VC, GSM_VD and MMS have higher peak temperatures. The upper bound of peak temperature after applying the algorithm to 3D NoC is specified by TH\(_1\) value. It can be seen in Fig. 2 that the resulting peak temperatures after employing the algorithm are always below the TH\(_1\) thermal threshold which is 90°C in GSM_VC and 80°C for the others.

To demonstrate the effectiveness of our algorithm in alleviating critical thermal situations, the percentage of reduction in peak temperature for each benchmark is reported in Fig. 3. It can be seen that the effect of the algorithm in improving system’s thermal situation is more profound in the applications with larger number of tasks and higher communication rates between them. As a result, the benchmarks such as GSM_VC and MMS, with the highest number of tasks and communication rate, are more affected by employing the algorithm. As reported in Fig. 3, the reductions of peak temperature in GSM_VC and MMS are close to 27 and 23 percent, respectively. In other words, migrating a task with heavy communication links in such benchmarks can dramatically impact system’s overall temperature.

### TABLE I.  Thermal simulation configuration [22] [23]

<table>
<thead>
<tr>
<th>Grid size</th>
<th>64 × 64</th>
</tr>
</thead>
<tbody>
<tr>
<td>Silicon thermal conductivity (w/m.k)</td>
<td>150</td>
</tr>
<tr>
<td>Interface material thermal conductivity (w/m.k)</td>
<td>0.25</td>
</tr>
<tr>
<td>Heat sink thermal conductivity (w/m.k)</td>
<td>400</td>
</tr>
<tr>
<td>Interface material thickness (m)</td>
<td>2e-6</td>
</tr>
<tr>
<td>Silicon resistivity (m.k/w)</td>
<td>0.006</td>
</tr>
<tr>
<td>Thermal interface material resistivity (m.k/w)</td>
<td>4</td>
</tr>
<tr>
<td>Silicon specific heat (m³.k)</td>
<td>1750000</td>
</tr>
<tr>
<td>Spreader side (m)</td>
<td>0.005</td>
</tr>
<tr>
<td>Heat sink side (m)</td>
<td>0.004</td>
</tr>
<tr>
<td>Ambient temperature (K)</td>
<td>298.5</td>
</tr>
<tr>
<td>Chip thickness with three vertical layers (µm)</td>
<td>36.47</td>
</tr>
<tr>
<td>Chip thickness with four vertical layers (µm)</td>
<td>48.63</td>
</tr>
</tbody>
</table>

### TABLE II.  Technology information (45nm) [24]

| Cu local thickness (m)        | 81e-9     |
| Cu semi-global thickness (m)  | 1.1       |
| Cu global thickness (m)       | 162e-9    |
| Glue thickness (m)            | 2e-6      |
| ILD local thickness (m)       | 72e-9     |
| Si layer thickness (m)        | 10e-6     |
| ILD semi-global thickness (m) | 72e-9     |
| Si bulk thickness (m)         | 500e-6    |
| ILD global thickness (m)      | 148e-9    |
| TSV area (m²)                | 25e-12    |

### TABLE III.  Characteristics of benchmarks [10]

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>NoC size (x,y,z)</th>
<th>Num. of tasks</th>
<th>Num. of edges</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPEG4</td>
<td>2 × 2 × 3</td>
<td>12</td>
<td>23</td>
</tr>
<tr>
<td>VOPD</td>
<td>2 × 2 × 3</td>
<td>12</td>
<td>15</td>
</tr>
<tr>
<td>MWD</td>
<td>2 × 2 × 3</td>
<td>12</td>
<td>13</td>
</tr>
<tr>
<td>MMS</td>
<td>3 × 4 × 3</td>
<td>29</td>
<td>23</td>
</tr>
<tr>
<td>GSM_VD</td>
<td>3 × 4 × 3</td>
<td>34</td>
<td>48</td>
</tr>
<tr>
<td>GSM_VC</td>
<td>4 × 4 × 4</td>
<td>53</td>
<td>81</td>
</tr>
</tbody>
</table>

### C. Migration Overhead

Migration overhead is related to the delay and power consumption imposed on 3D NoC to transfer the data and code of the task under migration to its new position. Although the proposed algorithm tries to minimize the migration overhead by preventing improper migrations, reducing overhead to zero is not possible. To observe the impact of task migration on 3D NoC’s PDP, we measured the network’s PDP during application of the proposed algorithm to the benchmarks. The measured PDPs in 3D NoC during the migration are illustrated in Fig. 4 for each benchmark. As shown in this figure, benchmarks with high intercommunication rates and large number of tasks such as GSM_VC, GSM_VD and MMS have the highest network PDP in comparison with the others.

To consider the performance degradation of system due to migration and in order to estimate the migration overhead, we performed two series of simulations. As the first simulation, we employed the proposed migration algorithm during the runtime of each benchmark and measured the PDP of 3D NoC. In the second simulation, we repeated this procedure under normal conditions, without applying our algorithm to the system. Then, we normalized the PDPs of 3D NoC from the first simulation to the PDPs obtained from the second simulation. The outcome, which is demonstrated in Fig. 5, depicts the small effect of migration on the 3D NoC’s PDP which is used as a representative of the system performance factor. It should be noted that the imposed overhead on the system is directly related to the number of migrations, which is kept small by using temperature prediction in our algorithm.

Due to the elevated power density in 3D NoCs, exploiting a fast online method for thermal management is crucial in such systems. Our algorithm, as a runtime technique, makes it possible to continuously check the system and detect new hotspots as early as possible. Therefore, in few cases where migration introduces minor negative impact on network’s PDP, such as the VOPD benchmark in Fig. 5, the small performance overhead is well justified for obtaining this advantage.

### D. Improved Task Mapping

As we mentioned before, the main goal of the proposed algorithm is mitigating the thermal situation in 3D NoCs by reducing the peak temperature of PEs. However, comparing the PDP of 3D NoC before and after occurring migration in the system shows that our algorithm not only is capable of reducing cores’ temperatures but also has the ability to improve system’s performance. Over the time, applying the proposed algorithm to the system leads in a decrease in distance between highly communicating tasks. In other words, a new task mapping will be obtained by employing the algorithm, which can improve the system’s performance by reducing PDP. Fig. 6 demonstrates the PDP of the network with new generated task assignment, normalized to the PDP of the system with initial task mapping. The results
reveal that the effect of algorithm in improving system’s PDP is more evident for highly communicating benchmarks, such as MMS. Because of the high communication rate between tasks in such benchmarks, reallocation of a task can have significant effect on performance key factors such as NoC delay and its power consumption.

E. Comparison to Previous Works

We chose the algorithm proposed in [10] to evaluate its efficiency in comparison to our thermal management scheme. The algorithm introduced in [10] is an ILP-based static thermal-aware mapping algorithm for 3D NoCs, which generates a semi-optimal task mapping to meet system’s thermal constraints. In contrast to our online dynamic migration method, the semi-optimal offline static mapping in [10] is not capable of adapting its operation to the dynamic behavior of applications during runtime. However, it tries to find an optimal core for each task to run, by taking into account information related to the tasks, which is known at design time. Moreover, due to complexity of the mapping algorithm and its long execution time, it is not suitable as a runtime dynamic thermal control technique.

Since the proposed method in [10] is an offline method, it does not impose any overhead on the system during runtime. So, we cannot compare our proposed scheme with this mapping algorithm in this regard. Instead, we compared the resulted peak temperatures after running both algorithms on the same benchmarks and 3D NoC structures, and the results are demonstrated in Fig. 7.

It can be seen that for small benchmarks such as VOPD, MWD and MPEG4, the semi-optimal mapping algorithm shows better performance in reducing hot spots and alleviating critical thermal situation. However, for benchmarks that have more number of tasks and communication rates among them, our proposed migration algorithm surpasses the mapping method in omitting hotspots and minimizing peak temperature. We conclude that in large benchmarks such as MMS, GSM_VD and GSM_VC, employing a dynamic mapping that can be changed according to system’s requirements would be more beneficial than using a fix task mapping that is generated based on information which is available at design time.

I. CONCLUSIONS

Scalable nanotechnology development exacerbates thermal issues in VLSI circuits. 3D NoC, which is the combination of 3D IC technology and NoC, is an effective solution for interconnects and ever growing computational demand problem. Regarding the increased power density in 3D NoCs, thermal management becomes more vital in such systems. In this paper, we introduced a dynamic distributed task migration algorithm, based on game theory, as a thermal control tool for 3D NoCs. To ensure thermal safety as well as graceful performance degradation, we modeled this multi-objective problem with cooperative game theory. Simulation results for different benchmarks indicate up to 27% decrease in PEs peak temperature after applying the proposed algorithm to the system. The effectiveness of the algorithm is more obvious in benchmarks with higher number of tasks and communication rates between them. Comparing the simulation results after and before employing the migration algorithm shows reduction in NoC’s PDP, which indicates the efficiency of the proposed algorithm in improving the system’s performance by changing its initial task mapping. This performance improvement comes at the price of negligible migration overhead, which has been reported as PDP in the paper.

In the rare cases in which the proposed algorithm is unable to find a proper destination core due to the thermal constraints, the integration of the algorithm with a hardware thermal management method like DVFS can be beneficial.
REFERENCES


