2015 23rd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing

Dynamic Application Mapping Algorithm for Wireless Network-on-Chip

Amin Rezaei[1], Masoud Daneshtalab[2,3], Danella Zhao[1], Farshad Safaei[4], Xiaohang Wang[5], Masoumeh Ebrahimi[2,3]
[1] University of Louisiana at Lafayette (ULL), Lafayette, USA (me@aminrezaei.com, dzhao@cacs.louisiana.edu)
[2] University of Turku (UTU), Turku, Finland (masdan@utu.fi)
[3] Royal Institute of Technology (KTH), Stockholm, Sweden
[4] Shahid Beheshti University (SBU), Tehran, Iran (f_safaei@sbu.ac.ir)
[5] Guangzhou Institute of Advanced Technology (GIAT), Guangzhou, China (xh.wang@giat.ac.cn)

***Abstract*— Because of high bandwidth, low latency and flexible topology configurations provided by wireless NoC, this emerging technology is gaining momentum to be a promising future on-chip interconnection paradigm. However, congestion occurrence in wireless routers reduces the benefit of high speed wireless links and significantly increases the network latency; therefore, in this paper, a Dynamic Application Mapping Algorithm (DAMA) is introduced for wireless NoCs in order to reduce both internal and external congestion. DAMA has three key steps: finding the *first node* to map, choosing the *first task* to be mapped onto the *first node*, and allocation of the remaining tasks to the remaining nodes. Simulation results show significant gain in the mapping cost functions compared to state-of-the-art works.**

Keywords—Network-on-Chip; Wireless Network-on-Chip; Dynamic Application Mapping; Latency; Congestion

I. INTRODUCTION

According to the International Technology Roadmap for Semiconductors (ITRS) [1], Many-Core Systems-on-Chips (MCSoCs) may integrate hundreds of Intellectual Property (IP) cores connected based on a Network-on-Chip (NoC) [2] communication infrastructure. Because of high bandwidth, low latency and flexible topology configurations provided by wireless NoC [5], this emerging technology is gaining momentum to be a promising future on-chip interconnection paradigm. However, wireless transceivers along with associated antennas impose considerable area and power overheads in wireless NoCs. Thus, instead of a single NoC spanning the entire system, as is traditional, a hybrid wireless NoC called WiNoC is proposed using both wired and wireless links [4]. Furthermore, a Hierarchical Wireless Network-on-Chip Architecture (HiWA) is introduced where the network is divided into subnets [6]. Intra-subnet nodes communicate through wire links while inter-subnet communications are almost handled by single-hop wireless links.

Such many-core systems will face a highly dynamic workload. Applications, as sets of communicating tasks, will enter and leave the many-core system at run-time. This featured dynamic nature is controlled through the mapping function of the system manager [7]. The duty of the system manager, i.e. application mapping, is to allocate application's tasks onto the system resources efficiently in terms of network latency and power consumption. Since each Wireless Router (WR) is shared by a cluster of processing elements, wireless NoCs are even more vulnerable to congestion than conventional NoCs; therefore, in this paper, a Dynamic Application Mapping Algorithm, DAMA is presented for wireless NoCs where the key contributions are:

- Defining a Square Factor (SF) to find the appropriate *first node* to start mapping the application in wireless NoC.
- Selecting the task with the largest number of edges as the *first task* to be mapped onto the *first node* in order to reduce internal congestion probability.
- Establishing a contiguous area of available nodes around the *first node* to map the rest of the tasks of the application to reduce external congestion probability.

The rest of the paper is organized as follows. Section II reviews backgrounds and related works. HiWA is introduced in section III as a sample of wireless NoCs. A dynamic application mapping approach for HiWA is presented in section IV. Section V presents and discusses experimental results. Finally, some conclusions are given in section VI.

II. BACKGROUNDS AND RELATED WORKS

New progresses in silicon integrated circuit technology, has permitted the integration of tiny transceivers antennas on a single chip, which results in introducing wireless NoC [5]. Since each WR is shared by many IP cores, an efficient task allocation technique is required to balance the utilization of WRs and reduce congestion. However, as task allocation is known as an NP-hard problem, different heuristics dealing with dynamic management of workload in multi-core systems, such as Nearest Neighbor (NN) and Best Neighbor (BN) are presented [8], [9]. In these heuristics, a clustering mechanism for the *first node* selection is considered. A set of cluster nodes are assumed to select the *first node* of the mapping algorithm among them. In another approach called Incremental Approach (INC) [10], the mapping problem is break down into two steps: the region selection, and the task allocation. In the region selection step, they start from the closest node to the Central Manager (CM) and include it in the region. Then, they iteratively add nodes to the selected region trying to keep both the selected region and remaining nodes contiguous. Afterward, in the task allocation step, application tasks are mapped inside the selected region. As an advanced approach, CoNA [3] selects the closest node to the CM with all its neighbors available. Thus a minimum number of available nodes are assured. Then, application Task Graph (TG) is traversed in breath-first order and tasks are mapped onto the neighboring of their parents in which a smaller square is formed. Moreover, in [11] a Smart Hill Climbing (SHiC) algorithm is introduced. SHiC uses a square factor (SF) model to approximate the contiguous available nodes around a given node.

However, previous papers, which have been mentioned above, proposed task mapping algorithms for conventional NoCs and they didn't cover congestion control mechanisms specifically for wireless NoCs which our paper is targeted.

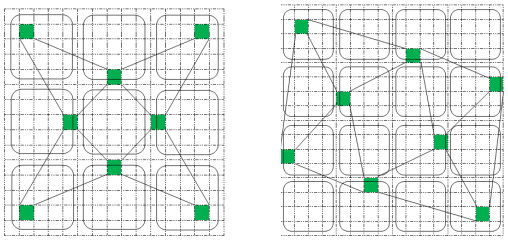


Fig. 1. Illustration of HiWA architecute (Nodes without color are BRs and colored nodes are WRs) (a) 225-node HiWA (b) 256-node HiWA

III. HIWA ARCHITECTURE

In this section, a Hierarchical Wireless Network-on-Chip Architecture, HiWA [6] is introduced as a target platform for DAMA algorithm. The backbone of HiWA is based on 2D mesh NoC. Whenever it is necessary, one of the Baseline Routers (BRs) is replaced by a Wireless Router (WR) which has wireless links with WRs in neighbors' subnets. WRs are able to perform both wired and wireless communication. Fig. 1a shows a 225-node HiWA which is divided into 9 subnets of 5×5 nodes. Fig. 1b shows a 256-node HiWA which is

1066-6192/15 $31.00 © 2015 IEEE
DOI 10.1109/PDP.2015.14

divided into 16 subnets of 4×4 nodes. The challenging issues in each wireless NoC architecture, are optimal number and placement of WRs. This requires designing and solving an optimal model using a meta-heuristic that is well suited for optimizing network topologies, like Simulated Annealing [12] or Ant Colony Optimization [13]. Since our focus is on congestion control, addressing mentioned issues is not in the scope of this work.

IV. Dynamic Application Mapping Algorithm

Congestion has a negative effect on the network performance and greatly increases the network latency. In the absence of an efficient mapping approach, the usability of WRs will be very high. Thus, an efficient application mapping approach is essential in wireless NoCs in order to reduce congestion over WRs. Two types of congestion can be considered from dynamic application mapping perspective: external and internal congestions. External congestion happens when a network channel is contented by edges of different applications. To decrease external congestion probability, the application mapped region should be as compact as possible and minimally fragmented. On the other hand, the internal congestion happens when a network channel is contended by edges of the same application.

A directed graph, named as a Task Graph (TG), represents each application in the system. Each vertex represents one task of the application, while each edge stands for a communication between the source task and the destination task as it is shown in Equation 1. TG of an application with 6 tasks is shown in Fig. 2. The amount of data transferred from the source task to the destination task is written on the edge. In order to simplify system modeling , we assume the architecture to be homogenous.

$$\forall t_i \in T, \forall e_{i,j} \in E, A_p = TG(T,E) \quad (1)$$

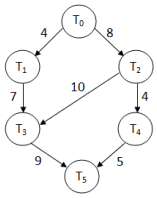


Fig. 2. Task graph of an application with 6 tasks and 7 edges

The proposed mapping algorithm (DAMA), consists of three steps. The first step is selecting the *first node* to map. The second step is picking up the *first task* of the application with the largest number of edges to be mapped onto the *first node*, which reduces internal congestion probability. After all, in order to reduce the external congestion, a contiguous area of available nodes around the *first node* is established. In the following we present each step separately.

A. First Node Selection

The most contiguous area is almost circular [10]. However, because adjacent regions share network links, choosing a circular region for an application in the mesh network increases the external congestion. As an alternative, when tasks are mapped onto a rectangular region of a network with minimal routing, all packets will be routed inside the region border and there will be no external congestion. The most contiguous rectangle is the square, and thus it is preferred in DAMA. The Square Factor (SF) of a node is the estimated number of contiguous, almost square-shaped, available nodes around that node. Accordingly, the suitable *first node* for mapping of an application would be the node with the SF equal to the application size [11].

Each running application in the system is modeled as a rectangle characterized by its corner nodes. Regarding the rectangle model of a running application, there might be some nodes within the rectangle which do not belong to the application. In this work, the rectangle of each application is modeled in such a way that minimizing the number of these nodes while try to keep the model almost square-shaped. The rectangle models of four running applications are shown in Fig. 3a. For instance, the rectangle of the application 1 has two nodes which don't belong to it ($n_{1,2}$ and $n_{0,2}$). Also the rectangle of the application 3 includes one node ($n_{6,1}$) which is not a part of the application. However, they are the best fit rectangles in order to stay close to square-shaped.

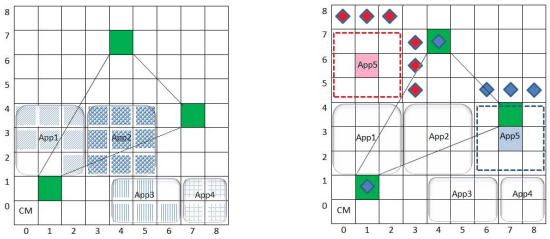


Fig. 3. DAMA square factor calcuation

To calculate the SF for each node:

- First, the largest square centered on the node is found, where it fits within the mesh limits and has no conflict with other running applications of the system. This is shown in Fig. 3b for the node $n_{7,3}$which is the *first node* of the application 5.
- Second, there might be also some more nodes beyond the square borders not belonging to system rectangles, as marked with rhomb in Fig. 3b. These nodes have one-hop links to one of the nodes within the square border. They are counted in order to prevent available nodes from being isolated while keeping the mapped area close to square-shaped.
- Finally, The SF of a given node is calculated by adding the nodes in the area of the largest square, with the available nodes beyond the square borders. For instance, the SF for $n_{7,3}$will be the square area nodes, 9, summed up with marked nodes, 5, which is 14. As it can be seen in Fig. 3b, two WRs ($n_{4,7}$ and $n_{1,1}$) are also counted in SF factor of $n_{7,3}$ because they have one-hop links to the node $n_{7,4}$ that is inside the square border of the application.

ALGORITHM I. DAMA *First Node* Algorithm

Input: Current set of applications, Size of the requested application
Output: The first node for mapping the requested application
|T|: size of the requested application
n: nearest free node to the CM
c: next nearest free node to the CM

```
while there is no untested free node
    if [(SF(n) < |T| AND SF(c) > SF(n)] OR [SF(c) >= |T| AND SF(c) < SF(n)] then
        n: choose c
    end
    c: choose the next nearest node to CM
end
return n
```

DAMA starts from the nearest free node to CM and walks through the network to find the appropriate *first node*. DAMA first looks for the node with the smallest SF value which is larger than or equal to the application size. Otherwise, the node with the largest SF value is preferred. Note that, when there are two nodes with equal SF, the one closer to CM is preferred to decrease the incurred defragmentation of remaining nodes. Also in order to reduce congestion over WRs, they are not chosen as *first node* of the application. The two candidates for the *first node* of the application 5 are shown in Fig. 3b. DAMA will choose the node $n_{7,3}$ because the SF factor of this node is 14 which is smallest than the node $n_{1,6}$ with SF factor of 15. Existing WRs, which have express paths to other WRs, will help the application to be mapped as contiguous as possible. In fact, WRs play the role of spreading contiguity across the whole system. Algorithm I shows pseudo-code of DAMA *first node* algorithm.

B. First Task to Map

The task with the largest number of edges is selected to be mapped onto the *first node*. This provides the largest possible number of available nodes around the *first task*; therefore, the edges of the *first task*

can be controlled by one-hop links, which reduces the congestion probability for the *first task*. If there is more than one task having the largest number of edges, then the first task would be the one with the most intensive communication. For example, in Fig. 2, both tasks t_2 and t_3 have 3 edges. Accordingly, since the total communication weight of t_3 is more than that of t_2 (26 vs. 23), t_3 is selected as the *first task* to be mapped. Algorithm II demonstrates pseudo-code of DAMA *first task* algorithm.

ALGORITHM II. DAMA *FIRST TASK* ALGORITHM

Input: Task graph of an application
Output: The first task for mapping to the system
t: task t_0 of the task graph
c: next task of the task graph

while there is no untested task
  if |e(c)| > |e(t)| **then**
    t: choose c
  else if |e(c)| = |e(t)| AND W(c) > W(t) **then**
    t: choose c
  end
end
return t

C. Neighborhood Allocation

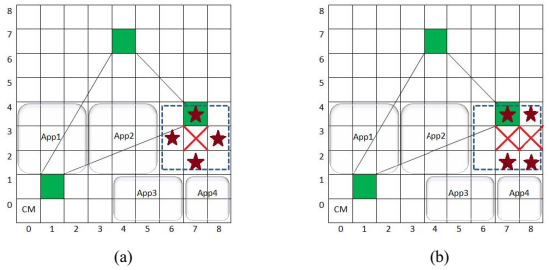
After the *first task* is mapped onto the *first node*, DAMA assumes the TG to be undirected and traverses tasks through their predecessor tasks in the breadth-first order, starting from the *first task*. Considering the set of available nodes in the closest neighborhood of the predecessor task, tasks are mapped onto the nodes which best fit into the smallest square with the *first node*. DAMA algorithm is shown in Algorithm III.

ALGORITHM III. DAMA ALGORITHM

Input: Task graph of an application, the first node to map, and the first task to be mapped
Output: Mapped tasks to the system
t_f: first task to be mapped
n_f: first node to map
t: the current task to be mapped
n: the current node to map

map t_f to n_f
while there is no unmapped task
  t: choose next task from the task graph in breadth-first order
  n: choose randomly from the set of free nodes which best fit to the smallest square
  map t to n
end

Considering the application 5 in Fig. 3b, after mapping the *first task* to the *first node* (which is $n_{7,3}$) the second node will be randomly chosen from one of the nodes of the set A={$n_{6,3}$,$n_{7,2}$, $n_{7,4}$,$n_{8,3}$}. Then supposing that $n_{8,3}$ is chosen, the new set for choosing the third node will be B={$n_{7,2}$, $n_{8,2}$, $n_{7,4}$, $n_{8,4}$}. The demonstration is shown in Fig. 4. As a result, DAMA maps the communicating tasks onto the closest neighborhood, while keeping the mapped area as close to square as possible. As can be seen, the application not only is mapped onto a more contiguous region without any internal congestion, but also imposes no external congestion on its neighboring applications due to the rectangularity of the mapped area.



(a) (b)

Fig. 4. DAMA Neighborhood Allocation

V. EXPERIMENTAL RESULTS

In this section, we assess the impact of the DAMA method on improving the mapping results. Several set of applications with 4 to 35 tasks are generated using TGG [14] where the amount of data transferred from the source task to the destination task are randomly distributed between 4 to 16 flits of data. Experiments are performed using an open-source simulator called *XMulator* [15] an integrated simulation platform for interconnection networks. Different mapping and *first node* selection methods are evaluated over the network size varying from 9×9 to 16×16 nodes. A random sequence of applications is entered into the scheduler FIFO according to the desired rate, λ. The sequence is kept fixed in all experiments for the sake of fair comparison. Applications are scheduled based on First Come First Serve (FCFS) policy and the maximum possible scheduling rate is called λ_{full}. An allocation request for the scheduled application is sent to the CM of the platform residing in the node $n_{0,0}$. In order to have a holistic view of the results and enable real case comparisons, each set of experiments are performed over 10 million cycles where hundreds of applications enter and leave the system.

A. Evaluation Metrics

AWD: Cost of a packet delivery is related to the number of hops it traverses. Hence, a metric to evaluate a mapping is the Average Manhattan Distance (AMD) between tasks of the mapped application edges. The smaller AMD value, the lower average packet latency is expected:

$$AWD_{map(A_p)} = \frac{\sum_{\forall e_{i,j} \in E} MD(map(t_i), map(t_j))}{|E|} \quad (2)$$

AWMD: The packet delivery cost depends not only on the length of its path, but also on the size of the packet; therefore, a more precise metric is to include the weight of edges in it. Average Weighted Manhattan Distance (AWMD) is the sum product of Manhattan Distance (MD) and all edges' weight of the mapped application, averaged by the total communication weights:

$$AWMD_{map(A_p)} = \frac{\sum_{\forall e_{i,j} \in E} W_{i,j} \times MD(map(t_i), map(t_j))}{\sum W_{i,j}} \quad (3)$$

MRD: To assess how contiguous the mapped region of an application is, Mapped Region Dispersion (MRD) factor is defined that is the mean value of all possible node pairs MD in the mapped region:

$$MRD_{map(A_p)} = \frac{\sum_{t_{i,j} \in T} MD(map(t_i), map(t_j))}{\left(\frac{|T|}{2}\right)} \quad (4)$$

NMRD: To decrease external congestion probability, the application mapped region should be as compact as possible and minimally fragmented. As it is mentioned before, the best mapped area would be square as it is the rectangle with the smallest MRD. It can be shown that the MRD of a square with |T| nodes will be:

$$AWD_{SQ(|T|)} = \frac{2 \times \sqrt{|T|}}{3} \quad (5)$$

Therefore, the Normalized Mapped Region Dispersion (NMRD) metric is defined which assesses the squareness of the mapped region independent of the size of the application. NMRD increases as the mapped area is getting less similar to a square:

$$NMRD_{map(A_p)} = 1 + \frac{\left| MRD_{map(A_p)} - MRD_{SQ(|T|)} \right|}{\left| MRD_{SQ(|T|)} \right|} \quad (6)$$

On the other hand, for measuring the internal congestion, Internal Congestion Ratio (ICR) is introduced. ICR is the number of edges of an application using the same communication channel (according to the XY algorithm) with respect to its total number of edges (|E|). Of note, we do not count overlapped edges that are originated from the same source. In such case, their injection is limited by source injection rate limit [16], and they will never contend.

B. Latency Evaluation

Latency evaluation for different mapping algorithms is summarized in Table I. Evaluation metrics are normalized to the DAMA results to ease comparison. The mapping results for applications with different packet sizes are the same, because the application TG and system behavior remains the same. The application injection rate is 2/3 λ_{full}. As can be seen, DAMA outperforms the BN algorithm by 40% and 20% reduction in external and internal congestion factors, respectively. It, also, obtains 50% gain over the INC in internal congestion and more than 15% in external congestion. In addition, DAMA has 25% gain in average for all evaluation aspects in comparison with the NN algorithm.

TABLE I. LATENCY EVALUATION

	NN [8]	BN [9]	INC [10]	DAMA
AWD	1.25	1.30	1.42	1.00
AWMD	1.29	1.32	1.36	1.00
NMRD	1.24	1.62	1.21	1.00
ICR	1.52	1.21	2.01	1.00

As shown in [10], decreasing MD between tasks of application edges is an effective way to minimize the communication energy consumption of the application. As another evaluation, we illustrate the percentage of packets that are delivered over different path lengths (MD). The experiments have been run for different algorithms in the injection rate of 2/3λ_{full}. As depicted in Fig. 5, more than 80% of the packets are delivered by one hop distance using DAMA scheme.

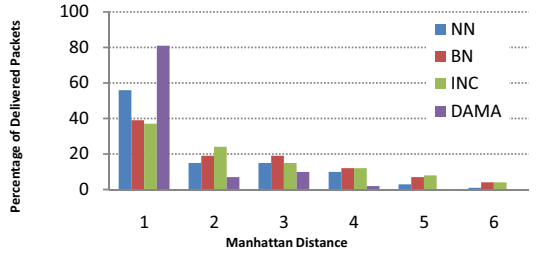


Fig. 5. Percentage of deliverd packets in different path lengths

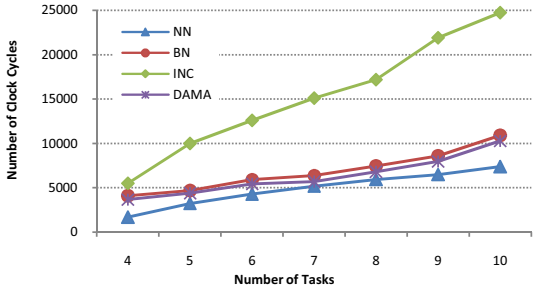


Fig. 6. Running time of mapping algorithms over application sizes

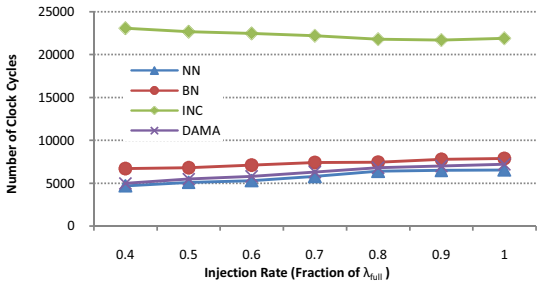


Fig. 7. Running time of mapping algorithms over different injection rates

C. Running Time Evaluation

The average number of clock cycles that is elapsed in CM to map applications with number of tasks varying from 4 to 10 is presented in Fig. 6. The injected rate is set to 0.75 λ_{full}. Furthermore, the time complexity of different mapping algorithms for applications with 8 tasks is presented in Fig. 7, when the injection rate varies from 0.4 λ_{full} to λ_{full}. As can be seen, DAMA provides a reasonable time complexity next to NN. As it is shown in Fig. 7, all mapping algorithms scale well when the injection rate is increased.

VI. CONCLUSION

In this paper, we proposed an efficient Dynamic Application Mapping Algorithm, DAMA, for wireless NoC. DAMA targets at reducing both internal and external congestions, and consists of three steps. First a network node with the best suitable Square Factor (SF) is selected as the *first node* to map. The SF of a node is the estimated number of contiguous, almost square-shaped, available nodes around that node. This guarantees a minimum free region around the *first node*, which results in a more contiguous region. Second, DAMA maps the task with the largest number of edges onto the *first node*, which reduces internal congestion probability. Third DAMA maps the rest of the tasks onto the nearest neighborhood, trying to form the most contiguous square-shaped region. Existing WRs, which have express paths to other WRs, help the system area to stay as contiguous as possible. In fact, WRs play the role of spreading contiguity across the whole system. Experimental results showed that DAMA is accomplished a reduced internal and external congestion probability as targeted.

REFERENCES

[1] ITRS. International Technology Roadmap for Semiconductors, 2011 edition.

[2] L. Benini and G. D. Micheli, "Networks on chips: a new SoC paradigm," In *IEEE Computer,* Vol. 35, Issue 1, pp. 70-78, 2002.

[3] M. Fattah, M. Ramirez, M. Daneshtalab, P. Liljeberg, and J. Plosila, "CoNA: Dynamic application mapping for congestion reduction in many-core systems," In *IEEE International Conference on Computer Design(ICCD)*, pp. 364-370, 2012.

[4] A. Ganguly, K. Chang, S. Deb, P. P. Pande, B. Belzer, and C. Teuscher, "Scalable hybrid wireless network-on-chip architectures for multicore systems," In *IEEE Transactions on Computers*, Vol. 60, Issue 10, pp. 1485-1502, 2011.

[5] M. F. Chang, J. Cong, A. Kaplan, M. Naik, G. Reinman, E. Socher, and S. W. Tam, "CMP network-on-chip overlaid with multi-band RF-interconnect," In *Proceedings of IEEE International Symposium on High Performance Computer Architecture*, pp. 191-202, 2008.

[6] A. Rezaei, F. Safaei, M. Daneshtalab, and H. Tenhunen "HiWA: A Hierarchical Wireless Network-on-Chip Architecture," In *IEEE International Conference on High Performance Computing & Simulation (HPCS)*, pp. 499-505, 2014.

[7] I. Tcarenko, M. Fattah, P. Liljeberg, J. Plosila, and H. Tenhunen, "Multi rectangle modeling approach for application mapping on a many-core system," In *Euromicro International Conference on parallel, Distributed and Network-Based Processing (PDP)*, pp. 452-457, 2014.

[8] E. L. Carvalho, N. L. V. Calazans, and F. G. Moraes, "Heuristics for dynamic task mapping in NoC-based heterogeneous MPSoCs," In *IEEE/IFIP International Workshop on Rapid System Prototyping (RSP)*, pp. 34-40, 2007.

[9] E. L. Carvalho, N. L. V. Calazans, and F. G. Moraes, "Dynamic task mapping for MPSoCs," In *IEEE Design & Test of Computers*, Vol. 27, Issue 5, pp. 26-35, 2010.

[10] C. L. Chou, U. Y. Ogras, and R. Marculescu, "Energy- and performance-aware incremental mapping for networks on chip with multiple voltage levels," In *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 27, Issue 10, pp. 1866-1879, 2008.

[11] M. Fattah, M. Daneshtalab,P. Liljeberg, and J. Plosila "Smart hill climbing for agile dynamic mapping in manycore systems," In *ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 1-6, 2013.

[12] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing." In *Science*, Vol. 220, Issue 4598, pp. 671-680, 1983.

[13] M. Dorigo and T. Stutzle, *Ant Colony Optimization*, MIT Press, 2004.

[14] "Task graph generator (TGG)." [Online]. Available: http://sourceforge.net/projects/taskgraphgen/.

[15] A. Nayebi, S. Meraji, A. Shamaei, and H. Sarbazi-Azad, "XMulator: A listener-based integrated simulation platform for interconnection networks," In *First Asia International Conference on Modeling & Simulation*, pp. 128-132, 2007.

[16] C. -L. Chou and R. Marculescu, "Contention-aware application mapping for Network-on-Chip communication architectures," In *IEEE International Conference on Computer Design (ICCD)*, pp. 164-169, 2008.