Computing and Informatics, Vol. 31, 2012, 939–970

A TOPOLOGY-INDEPENDENT MAPPING TECHNIQUE FOR APPLICATION-SPECIFIC NETWORKS-ON-CHIP

Rafael Tornero, Juan M. Orduña

Departamento de Informática
Universidad de Valencia, Spain
e-mail: Juan.Orduna@uv.es

Maurizio Palesi

Kore University, Italy
e-mail: maurizio.palesi@unikore.it

José Duato

DISCA, Universidad Politécnica de Valencia, Spain
e-mail: jduato@disca.upv.es

Communicated by Ulf Schroeder

Abstract. The design of Networks-on-Chip (NoCs) involves several key issues, including the topological mapping, that is, the mapping of the processing elements or Intellectual Properties (IPs) to the network nodes. Although several proposals have been focused on topological mapping last years, this topic is still an open issue. In this paper, we propose, in an extended manner, a topology-independent mapping technique for application-specific NoCs that can be used with regular or irregular topologies, and with any routing algorithm. This technique globally matches the communication pattern generated by the IPs with the available network bandwidth in the different parts of the network. The evaluation results show that the proposed technique can provide better performance than other mapping techniques not only in terms of average latency and network throughput, but also in terms of power consumption.

Keywords: Networks-on-Chip, topological mapping, performance evaluation

Mathematics Subject Classification 2010: 68M07, 68M10

1 INTRODUCTION

The continuous reduction in the time-to-market required by the telecommunications, multimedia and consumer electronics market makes full-custom design inappropriate for certain application domains, and has led to the definition of design methodologies based on the reuse of Intellectual Properties (IPs). This has caused an increment in complexity and heterogeneity of single chip-based embedded systems. Such systems-on-chip (SoCs) imply the seamless integration of numerous IP cores performing different functions and operating at different clock frequencies. On the one hand, this integration process requires standard interface sockets to allow design reuse of IP components across multiple platforms. On the other hand, it is causing the scalability limitations of state-of-the-art SoC buses to emerge.

The on-chip interconnection system represents one of the major elements which has to be optimized in designing a complex digital system. Networks-on-chip (NoCs) are generally viewed as the ultimate solution for the design of modular and scalable communication architectures, and provide inherent support to the integration of heterogeneous cores through the standardization of the network boundary.

In NoC architectures, each tile of the network contains a resource (either an active processing units or other passive elements like memories, etc.) and a switch. In direct networks, each switch is connected to a resource (a processor, a memory, or any other IP compatible with the NoC interface specifications) and to some adjacent switches, depending on the NoC topology. The design flow for a NoC involves several steps [11]. First, the application has to be split up into a set of concurrent communicating tasks. Then, the IPs are selected from the IP portfolio and the tasks are assigned and scheduled. Once the task are mapped and scheduled, the IPs have to be mapped onto the NoC topology. Some authors denote this mapping as the *topological mapping* [2], and it is known to be NP-hard [9]. As discussed in [18], systematic approaches to NP-complete problems are difficult to scale with the problem size. For example, the branch-and-bound algorithm is memory hungry, while the backtracking is time consuming.

Different mapping techniques have been proposed last years [2, 14, 21, 12, 20]. Some of these proposals are a unified approach of the topological mapping and the problem of routing among network nodes [12, 21]. Therefore, the usefulness of these mapping strategies is guaranteed only for the considered routing scheme. Since one of the key issues in the design of NoCs is the reduction of both area and power dissipation, other proposals focus on these performance metrics [14, 2]. One of these techniques also proposes dynamic re-configuration mechanisms for adapting the NoC to the communications generated by each application [20].

The existing topological mapping proposals for NoCs usually consider regular topologies like 2-D meshes. The reason is that these topologies offer low and constant link delay. However, manufacturing defects or even real-time failures often make the resulting network topology to become irregular, and some routing techniques have been proposed for solving these problems [19]. Additionally, there are also NoCs with inherent irregular topologies [25, 5], and the methodology used in the design

flow can also introduce some irregularities [4]. These scenarios need a topological mapping technique that can adapt the communications exchanged by the IPs to the existing network topology, regardless of the network regularity. Such a topological mapping technique would help reduce network latency and power consumption. In a previous work, we outlined a topological mapping technique for NoCs fulfilling this requirement [27].

In this paper, we propose, in an extended manner, this topology-independent mapping technique that globally adapts the communication pattern generated by the IPs to the available network bandwidth in the different parts of the network. Unlike other mapping techniques, the proposed method uses an abstraction of the network (a table of communication costs) as the model for both the NoC topology and the routing algorithm (the network resources). In this way, the method can be used with either regular or irregular topologies, and with any routing algorithm (different topologies and/or routing algorithms are modeled as tables with different values [28]). This feature can help fully exploit irregular NoCs, provided that the routing function provides alternative paths for reaching the destination. The evaluation results show that the proposed technique can provide better performance than other mapping techniques, not only in terms of average latency but also of power consumption. On the other hand, since the underlying methodology is based on correlating the network model, the proposed technique does not need to experimentally test each solution considered, requiring a relatively low computational effort.

The rest of the paper is organized as follows: Section 2 presents some background and explains the scope and limits of the proposed technique. Section 3 describes the proposed topological mapping technique, including the model of the NoC resources. Next, Section 4 shows the experimental correlation of the network model with the actual network performance. Section 5 shows the performance evaluation of the proposed technique. Finally, Section 6 concludes the paper and draw some directions for future developments.

2 BACKGROUND

The topological mapping problem consists of deciding the exact allocation of each of the existing IPs within the NoC topology. This problem is known to be NP-hard [9]. From the NoC perspective, the topological mapping consists of designing the traffic pattern that will be generated on the existing network topology. In order to achieve this goal, the mapping technique uses the network topology and the routing algorithm as inputs, in order to define a network model (that is, in order to characterize the network performance). The network model can measure the performance as different metrics, like energy [14], communication delay [17] or link bandwidth [21]. On the other hand, the topological mapping technique should use another input, a model of the communications exchanged among the existing IPs, in order to match these communications with the network model, in such a way that the resulting traffic pattern maximizes the network performance.

A pictorial representation of the topological mapping problem is shown in Figure 1. We consider an already mapped and scheduled task graph onto a set of IPs from which a *communication graph* (or core graph) is derived. In this work we do not tackle the problem of task mapping and scheduling as they have been extensively addressed in literature in the area of hardware/software co-design and IP-reuse [6, 17]. The communication graph represents one of the inputs of the topological mapping problem.

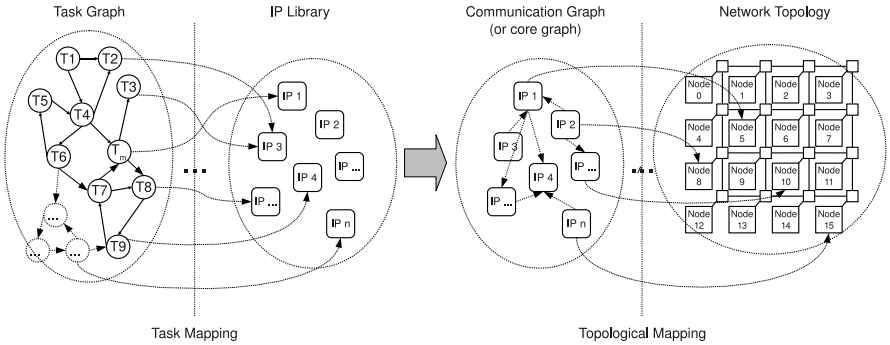


Fig. 1. The topological mapping problem

Following the same philosophy, our idea is to establish a network model (denoted as model of communication cost between each node pair in the network) that is independent of the traffic pattern. On the one hand, this model should take into account only the topology of the network and the routing algorithm. In this way, the model can be used as the basis for an efficient mapping of IPs to network nodes, since it provides a metric based on internode distance (different parameters like latency and energy are directly related to internode distance). On the other hand, the model should be easily correlated with network performance. In this way, if the correlation of the model is high, then it can be used without experimentally checking the performance of each mapping considered, like other techniques do [2]. This feature is essential for reducing the computational cost with respect to other topological mapping techniques [2]. In addition, we propose robustness in front of network irregularities as another major issue of the model of the communication cost. In other words, the model should be valid for both regular and irregular network topologies. In Section 3.1 we describe how Kirchoff's laws can be used to model the network irregularities, in such a way that a robust model can be developed.

Apart from the network model, the other key issue for determining an efficient topological mapping is the traffic pattern. That is, the application being executed on the chip is going to generate a set of communications or traffic pattern that should be taken into account when performing the topological mapping [11]. In this sense, the proposed technique focuses on specific applications where the IPs generate si-

milar communication patterns for different executions of the application, regardless of the input data. For example, self-similar traffic has been observed in the bursty traffic between on-chip modules in typical MPEG-2 video applications [29]. Additionally, the estimation of the information exchanged by the tasks can be extracted from the communication task graph (CTG) of the application [14, 2, 17]. For this kind of applications, the idea is to measure the amount of traffic exchanged between each pair of IPs. For example, real traces can be obtained from the communication task graph in which the application tasks are assigned and scheduled with reference to a library of available IPs/cores [2, 30, 14]. Using these traces, the communications generated by the application can be modeled as a table of messages/packets exchanged among IPs. It should be noticed that such measurement implies a certain task assignment and scheduling (the parallel tasks are assigned and scheduled onto the IPs according to certain criteria), but that scheduling is beyond the scope of the topological mapping technique. Also, different CTGs applications can be considered to be simultaneously executed on the existing IPs, measuring the total aggregated traffic exchanged by the IPs when executing all the CTGs.

Nevertheless, applications requiring the reconfiguration of the topological mapping are beyond the scope of this paper. This reconfiguration may be needed either due to network changes (changes in the the network topology, fault-tolerance, etc.) or due to "dynamic" applications that change their CTGs in time. Although both cases can be considered as different "static" applications that are executed in different time intervals, they would require reconfiguration techniques that are beyond the scope of this paper. Therefore, at this point a table of internode distances models the network topology and the routing function, while another table models the tasks scheduling and the communications pattern generated by that scheduling. The topological mapping technique can be treated as the problem of matching these two tables.

3 A TOPOLOGY-INDEPENDENT MAPPING TECHNIQUE

During the NoC design flow, task mapping and scheduling are carried out in order to properly balance the computational requirements of the application with the available resources in the existing IPs. These procedures are beyond the scope of this paper, since they have been addressed in the area of hardware/software co-design and IP-reuse [6, 17]. However, after these steps are performed, it is clear which tasks will be hosted in each IP. These tasks will exchange information with other tasks assigned to (executed in) other IPs, generating a given network traffic pattern (due to one or more applications). The topological mapping, consisting of assigning each IP to a network node, should exclusively focus on this pattern. Although the tasks can be scheduled for execution in each IP following different temporal sequences, the overall spatial communication pattern exchanged among the IPs will not vary. The topological mapping technique can take into account this pattern to improve the NoC performance.

We propose a topology-independent mapping technique that near-optimally fits the communication pattern of the IPs with the available network bandwidth in the different parts of the network. This technique consists of several steps: first, the available network resources (taking into account the topology and the routing algorithm) must be modeled. Second, the communication pattern generated by the tasks running on the different IPs must be modeled. Third, a criterion that measures the suitability of each mapping is needed. This criterion should use the model of the network resources and also the communication pattern of the tasks. Finally, some mapping technique that tries to minimize/maximize the previous criterion should be developed.

3.1 Network Model

In order to model the network resources, we have defined a metric previously proposed for off-chip networks [23]. This metric is based exclusively on the internode distances jointly provided by the existing topology and the routing algorithm, since this metric is inversely related with latency. Concretely, our model proposes a simple metric, the *equivalent distance* between each pair of nodes (in what follows we will refer to a network router as a node). A *table of equivalent distances* can be obtained by computing the equivalent distance between each pair of nodes in the network. As a first approach, we assume that the link bandwidth is the same for all the links in the network (the area of all the links in the NoC is the same). Therefore, we assign the unit cost for each link, but this limitation can be removed if necessary by assigning different link costs to different links. In addition, we consider the case of adaptive algorithms that supply multiple paths for each origin-destination pair of nodes. In this case, the cost for communicating this origin-destination pair depends on the path chosen by each message sent. We have modeled this average cost by using an analogy to the electrical equivalent resistance.

The equivalent distance for a pair of nodes is computed taking into account all the shortest paths between them supplied by the routing algorithm. Although this model can be applied even when the routing algorithm supplies non-minimal paths, as shown in [1], only the shortest paths are considered. The reason is that the network is assumed to be both fault-free and working below its saturation point on the average. Otherwise, the network itself should be re-designed. With such assumptions, most of the traffic will use the shortest paths. In order to compute the equivalent distance, we use the same rules as for electrical circuits to compute the total communication cost between nodes, applying Kirchoff's laws. The reason for using these rules is that the alternative paths increase the available network bandwidth between a pair of nodes, but the length of each path increases the latency and the number of nodes that share that path. The model of equivalent allows to model the network irregularities by containing different values in the table of distances, in such a way that the same model (table of internode distances) can be used for regular or irregular topologies. Thus, the algorithm to compute the internode distance for a pair of source/destination nodes is the following one:

1. If the routing algorithm provides only one shortest path between a given pair of nodes, then the equivalent distance is the sum of the costs of the links that form the path. That is, this case is similar to computing equivalent resistance of an electrical circuit consisting of serially arranged resistors. Since we have assumed that all the links in the network have unit cost, the communication cost is equal to the number of links in the path. We have chosen only the shortest paths because usually NoCs work below the saturation point, and therefore it is not likely to use the longest paths (even the shortest paths are not necessarily minimal paths). However, this model can be modified to take into account all the possible paths.
2. If there exists more than one shortest path between a given pair of nodes, then the communication cost between them is computed similarly to the electrical equivalent resistance between two points of an electrical circuit, replacing each link in a shortest path with a unit resistor and applying Kirchoff's laws. The paths not supplied by the routing algorithm are not considered.

As an example, let us consider the 2-D mesh network topology shown in Figure 2 a), and let us assume that the routing algorithm can provide two different paths for going from node 0 to node 6: 0-1-6 and 0-5-6. In this case, all the paths supplied by the routing algorithm are shortest paths.

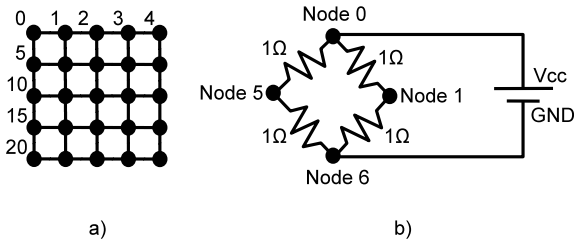


Fig. 2. a) A 2-D network topology b) Equivalent circuit for going from Node 0 to Node 6

Therefore, in order to compute the equivalent distance between nodes 0 and 6, the source and destination nodes must be considered as the V_{cc} and GND points of an electric circuit. Nodes 1 and 5 must be considered as two different intermediate points, and each link must be considered as a resistor with unit resistance, as Figure 2 b) shows. This results in a circuit with two parallel branches, each one composed of two unit resistors serially arranged. Applying Kirchoff's laws to this circuit, an equivalent resistance of 1 Ohm is obtained. Thus, the equivalent distance from node 0 to node 6 would be set to 1. The elements in the table of distances should be computed in this way. It should be noted that this model can be used either with uni-directional or bi-directional channels. Any irregularity produced either by the topology or by the routing algorithm would result in different values in the equivalent distances for communicating the nodes affected by the irregularities.

The table of communication costs contains the equivalent distances for all the origin-destination node pairs. If a given element in this table is located at row i and column j ($i, j \in [0..N-1]$, where N is the number of rows and columns of the table, that is, the number of network nodes), then we will denote that element as d_{ij}. It represents the cost for communicating node i with node j.

3.2 Model of Communication Pattern

The communication pattern generated by the tasks running on the different IPs can be estimated by measuring the number of bytes exchanged by the tasks being executed on each IP. Obviously, measurement should not be done by executing the application, because the mapping is required prior to the execution of the application. Nevertheless, there exists a wide variety of applications whose traffic patterns remain unchanged during different executions. For example, self-similar traffic has been observed in the bursty traffic between on-chip modules in typical MPEG-2 video applications [29]. Additionally, the estimation of the information exchanged by the tasks can be extracted from the communication task graph of the application [14, 2, 17]. The abstraction of the communication pattern in a table easily allows the modeling of the communications required by multiple concurrent applications (it would consists of the sum of the tables corresponding to all the applications to be executed on the NoC).

Therefore, the proposed technique also computes a table of communications between IPs. If the application(s) tasks are mapped on N IPs, then this table will consist of $N \times N$ elements. We have denoted each element (i, j) in this table as com_{ij}. This value represents the number of bytes that the task(s) mapped onto IP i send(s) to the task(s) mapped onto IP j.

3.3 Mapping Technique: Heuristic Search Method

The table of communication costs models the network resources, while the table of communication requirements models the communications shown by the application(s). The next step is to define a quality function that properly measures the suitability of each mapping.

We will represent a given mapping of IPs (hosting one or more tasks) to network nodes as a *mapping array* of N elements. We will denote each element i of this mapping array as m_i. This value means that IP i is assigned to the network node m_i. Thus, for example, if the second element of this mapping array contains the value 5, this will mean that the IP 1 (if they are numbered from 0 to $N-1$, or IP 2 if they are numbered from 1 to N) should be located at the network node 5. The quality function is denoted as the *mapping coefficient* M_c, and it is defined as

$$M_c = \sum_{i=1}^{N} \sum_{j=1}^{N} com_{ij} \cdot d_{m_i m_j}$$

where $d_{m_i m_j}$ is the element in the table of equivalent distances in the m_i row and in the m_j column. M_c represents the sum of all the bytes exchanged between the existing tasks, weighted by their corresponding cost of sending these bytes across the NoC according to the mapping array. Therefore, the best mapping (for a given table of communication costs and a given table of communication requirements) will be the one that minimizes M_c. Effectively, the purpose of the proposed mapping technique is to search the best mapping array for a given network topology and for a given communication pattern. Since the communication cost is defined as inversely proportional to latency, we are actually mapping the IPs hosting the tasks that communicate more frequently to the network nodes with the lowest latency between them (but also using the less amount of resources as possible). By doing so, we fully exploit the existing network resources, delaying network saturation as much as possible.

In a previous work, we had developed a communication-aware task mapping technique for clusters [23]. That mapping technique was based on the same models of network resources and communication pattern as the one proposed here. Also, that technique used the same quality function. Since the mapping problem is NP-hard [9], in that case we tested different heuristic techniques for implementing the search method (searching the best mapping array). Concretely, we tested a Genetic Algorithm [13], a Tabu Search Method [10, 24], a Simulated Annealing search method [16] and Greedy Randomized Adaptive Search Procedures (GRASP) [8]. However, due to the global nature of the optimization problem, we obtained the best results (in terms of both execution times and M_c values) with a random search method.

Although the topological mapping technique proposed here uses the same models and quality function as the task mapping technique [23], it should be noticed that the search specifications are different. First, the topological mapping is computed off-line (it is a separate stage of the NoC design flow) while the task mapping technique was included within the scheduling strategy, requiring much harder time constraints. Additionally, the topological mapping technique inherently considers one-to-one assignments, while the task mapping technique can also consider several-to-one assignments (several tasks can be assigned to a single processor). Finally, the scalability of the search method should be taken into account. The current trend in the NoCs environment is towards large-scale sizes, with hundreds and even thousands of nodes where to map each IP. Therefore, the scalability of the search method should be guaranteed. Due to these differences, we have studied again two different heuristic methods when applied to the topological mapping technique, Genetic Algorithms (GA) and the random search method [26].

3.3.1 Genetic Algorithms

GA start from an initial population, made of R individuals or *chromosomes* (particular solutions, in this case a chromosome is a mapping array) that evolves following certain rules until reaching a convergence condition that maximizes (minimizes)

a fitness function. Each iteration of the algorithm consists of generating a new population from the existing one by recombining and/or mutating chromosomes. The two main types of Genetic Algorithms are generational and stationary. In the generational GA two different generations cannot exist at the same time, emulating the way in which insects reproduce. Thus, the combinations and replacements of the current population pool generate a different population pool. On the contrary, a stationary GA allows the existence of ancestor and offspring chromosomes in the same population pool. The selection operator allows to select those population individuals that will be used for reproduction. The purpose of the selection operator is to give more chances to the most suitable individuals (chromosomes) in the current population. The purpose of the crossover operator consists of mixing the previously selected ancestor chromosomes to generate the offsprings. The mutation operator consists of the random alteration of each of the elements (genes) in the chromosome with a mutation probability. The purpose of mutation is to produce population diversity. Finally, the replacement operator consists of replacing the current population by their offsprings or a mix of both current chromosomes and offsprings.

We used the Evolving Objects (EO) Evolutionary Computation Framework [15]. This framework consists of a evolutionary computation library based on C++ templates, and it allows to select different parameters of the GA to be implemented; kind of GA, selection, crossover and mutation operators, and a replacement strategy. Although they are not shown here due to space limitations, we studied different operators for the selection, crossover, mutation, and replacement. Here we explain the parameters that provided the best results for the mapping problems to be solved. Concretely, we obtained the best results with a generational GA with a selection strategy by ranking, two-point crossover operator with an probability of 1.0, a mutation operator of a single element exchange with a probability of 0.5, and a replacement strategy based on tournament.

The convergence (finishing) condition for this algorithm was fixed as either one hundred generations without improving (decreasing) the fitness function value or one thousand generations. The general pseudo-code of the implemented algorithm could be the one shown in Figure 3.

3.3.2 Random Search Method

The random search method implemented was very similar to the one implemented for task mapping [23]. It starts with a random mapping array and each iteration consists of exchanging two randomly selected values of the mapping array. If the resulting mapping array shows a better mapping coefficient, then this permutation is saved; if not, then it is discarded. The stop condition for the algorithm is to perform a given number of consecutive permutations without decreasing the resulting mapping coefficient. At this point, the minimum mapping coefficient reached and its corresponding mapping array are saved, and another seed (random mapping array) is tried. The different seeds prevent the algorithm to stop when a local minimum is found. The algorithm finishes when a number of different seeds has been explored.

```
1: CreateInitialPopulation(S)
2: ComputeFitnessFunction(S)
3: while convergent condition is not satisfied do
4:   for Population_Size/2 do
5:     RangeSelection(S)
6:     Crossover(Parent1,Parent2)
7:     Mutation(Offsp1,P)
8:     Mutation(Offsp2,P)
9:     ComputeFitnessFunction(Offsp1)
10:    ComputeFitnessFunction(Offsp1)
11:    Replacement(Offsp1,Offsp2,S)
12:  end for
13: end while
```

Fig. 3. Pseudo-code for the implemented algorithm.

The result of this algorithm is the mapping array with the lowest mapping coefficient reached until that moment. The pseudo-code of this random method could be the one shown in Figure 4.

In this pseudocode, the procedure `RandomSelect(`i`,`j`)` randomly selects two indices within the range from 0 to N, and function `CrossMapping(`M`,`i`,`j`)` exchanges the values of elements i and j in mapping array M, providing a mutated partition P. Finally, the procedure `Save(`M`,` $Mc(M)$`)` saves both the mapping array and its associated mapping coefficient for future iterations. It also checks (and saves if necessary) if this mapping coefficient is the minimum historical value (including all the seeds).

3.3.3 Performance Evaluation

For comparison purposes, we have executed the heuristic search with the proposed technique and also with the random search method, and we have measured the fitness function M_c values provided by each of the methods, as well as the execution times required for providing these values. In order to make a fair comparison, we have considered that each seed of the random search method (each random initial population without the local search) is equivalent to one generation of the genetic algorithm. We have considered the Multimedia System (MMS) [2] as the application to be executed. This application is composed of forty parallel tasks that are scheduled in twenty five processing elements (IPs). The mapping technique should find the best topological mapping of these twenty five processing elements within a network topology consisting of a 2-D mesh (the most common topology used for NoCs) of 5×5 nodes (a problem size that can be representative of current NoC sizes).

Figure 5 shows the fitness function values provided by each method. Concretely, it shows six plots. Each plot corresponds to the random search method when us-

```
1: int N; // Mapping array size
2: int P[N], M[N]; // Aux. mapping arrays
3: int Seed; // No. of seeds
4: int i, j, k, count; // Aux. var. and iter. count
5: int no_dec; // No. of failed permutations
6: CreateRandomMapping(M);
7: ComputeMappingCoef(Mc(M));
8: Save(M, Mc(M));
9: for k = 1 to Seed do
10:   while count < max_it_per_seed and no_dec < max_it_without_dec do
11:     RandomSelect(i, j);
12:     P = CrossMapping(M, i, j);
13:     ComputeMappingCoef(Mc(P));
14:     if Mc(P) < Mc(M) then
15:       save(P, Mc(P))
16:       no_dec = 0;
17:     else
18:       no_dec++;
19:     end if
20:     count++;
21:   end while
22:   count = 0;
23:   no_dec = 0;
24:   CreateRandomMapping(M);
25:   ComputeMappingCoef(Mc(M));
26:   Save(M, Mc(M));
27: end for
```

Fig. 4. Algorithm for the random search method.

ing a different search depth (number of iterations per seed without improving the metric). We have labeled the plots for each seed as "dirxxg", for direct method. The "xx" value stands for the number of iterations per seed. Also, it shows the values provided by the proposed method for the same case. We have labeled this plot as "ga". In this figure, the X-axis shows the number of generations (that is, the number of seeds or initial populations for the random search method and the number of generations for the GA method) while the Y-axis shows the M_c values provided by each method.

The shape of the plots in Figure 5 shows that the GA method provides worse fitness function values than the random search method for a given search depth (for a given number of generations). However, the slope of the plot corresponding to the "ga" plot is much higher, in such a way that for a search depth of more than 100 generations it provides better fitness function values than the random

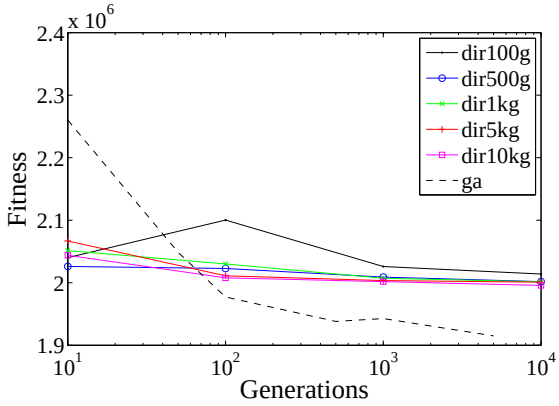


Fig. 5. Fitness function values provided by the heuristic methods

search method. That is, the GA provides better results the deeper the search is, until a "bottom" value is reached. This "bottom" value is lower than the bottom values provided by the random search method. However, the random search method provides values very close to the "bottom" value only with a shallow search, and the maximum percentage difference in the fitness function values achieved by the plot for the GA with respect to the random search method is around 5 %, when searching around one thousand and four hundred generations.

In order to analyze the computational cost required by both heuristic methods, Figure 6 shows the execution times (in seconds) required for providing the results shown in Figure 5. This figure shows that the genetic approach requires similar execution times than the random search method for a search depth lower than five hundred seeds, just the search depth needed by the GA method to provide better function values.

These results show that both heuristic methods can be used for the topological mapping technique. However, the size of current existing NoCs is not greater than a few tens of nodes, and for these sizes the regularity of the 2-D mesh topology makes difficult for the search method to find very different fitness function values in different regions of the search domain. Although the GA method can provide better values, it requires a time-consuming deep search in order to provide an improvement of 5 % in these values. That is, although the GA method seems to have better chances to explore wider regions of the domain space in the same time than the random search method (being more scalable with the problem size), the current problem size and topologies do not allow to exploit this feature. On the other hand, the use of the random search as the heuristic method represents the worst case for the proposed technique (the most inefficient search in terms of fitness function values). Therefore, in the rest of the paper we will denote the random search method as the search method, showing that the proposed technique can improve the performance of application-specific NoCs even when using an inefficient search method. Also, it is

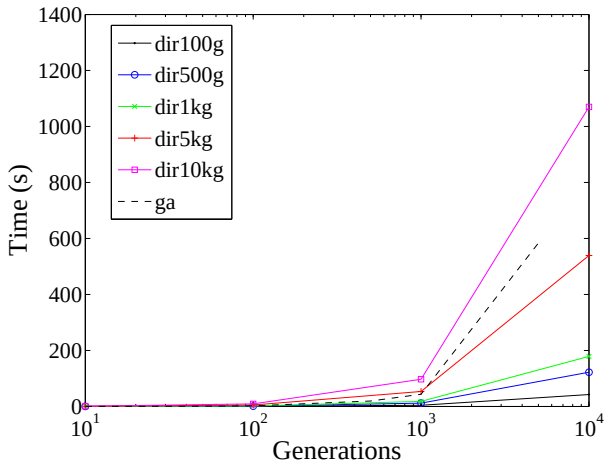


Fig. 6. Execution times required for the heuristic methods

worth mentioning that the proposed mapping technique can use any of the proposed heuristics, as well as any other heuristic method that can be implemented for this particular problem. Thus, even in case of a significant increase in the problem size in the future, the proposed technique could use a different heuristic search like "ga" and still provide good performance.

4 CHARACTERIZATION OF NETWORK RESOURCES

As discussed in Section 1, in the proposed technique the methodology consists of correlating first the model of the NoC with the actual NoC performance. If the correlation is high, then the random search method is used to estimate the quality of each assignment. In this section, we show the correlation of the model with actual network performance. Since the quality function M_c uses this model, if the correlation is high then the experimental validation of each mapping will be no longer needed.

For correlation purposes, we have considered a 2-D mesh network topology with X-Y routing and wormhole switching. The table of equivalent distances for that network is computed as described above. We have studied the correlation between each distance in the table of distances and the average latency of the messages exchanged between the corresponding pair of nodes. The performance evaluation methodology used is based on the one proposed in [7]. In this case, we have used the Noxim simulator [22].

We have considered different network sizes. However, for the sake of shortness, we show here the correlation results for a 8×8 2-D mesh, since this size is large enough to be representative of the NoCs sizes expected for next years. The correla-

tion results obtained for other network sizes were very similar. For characterization purposes, we have used a uniform traffic pattern. We have considered a fixed packet size of 8 flits. Each router has an input buffer capable of allocating 3 flits. We have made simulations with different injection rates, in order to simulate the network with different workloads (ranging from low load to saturation). The simulator provides the global average latency value obtained for each simulation. This value is computed as the average latency value obtained of all messages transmitted through the network during a given simulation. In addition, the simulator provides the average latency value for each source-destination pair in the network. From these measurements, we have computed the correlation between the table of distances and network latency.

Figure 7 shows the performance evaluation results for a 2-D mesh network topology with X-Y routing and wormhole switching. This figure shows on the X-axis the traffic injected to the network, measured in packets per cycle and per node (we assume one that each IP is connected to a network switch). On the Y-axis, it shows the global average message latency. Also, average message latencies for each pair of nodes were computed. However, they are not shown here due to space limitations. Each value in this figure was computed as the average value of fifty different simulations. Figure 7 shows the typical behavior expected for an interconnection network [7]. While the injected traffic is kept below the saturation point, the average latency slightly increases (points $S1$ to $S7$). However, when the injected traffic makes the network to reach saturation (from point $S8$ up) the average latency starts to greatly increase with the traffic rate.

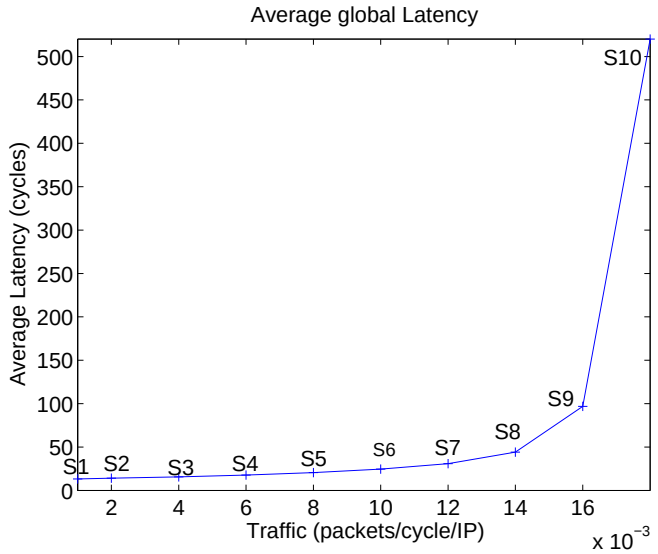


Fig. 7. Global performance results for a 8×8 topology

In order to establish the correlation between the table of distances and these performance evaluation results, we have first computed the least square linear adjustment for each point in Figure 7. The results for the first point $S1$ (traffic equal to 0.001 packets/cycle/IP and global average latency equal to 13.28 cycles) are shown in Figure 8. In this figure, the X-axis shows the values in the table of distance, while the specific average latencies obtained by the simulator for each pair of nodes are shown on the Y-axis. The correlation index obtained in this case has been 98 %, showing that the correlation between the proposed metric (the inter-node distance) and the performance achieved is very high.

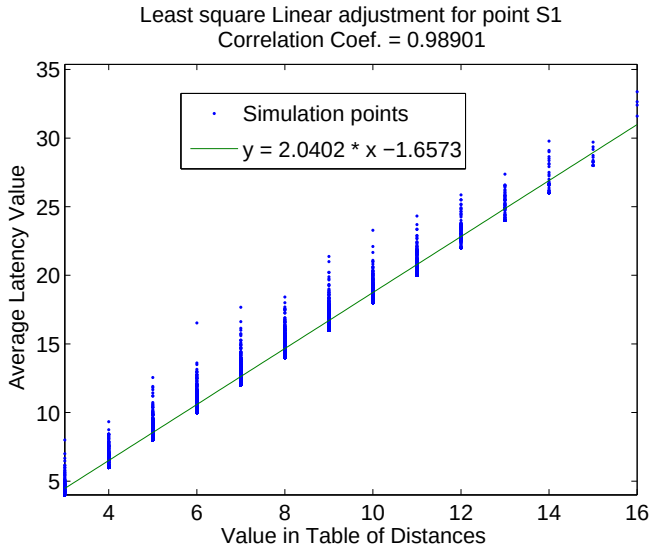


Fig. 8. Least square linear adjustment for simulation point S1

We have computed the same least square linear adjustment for all the points in the plot in Figure 7. Although NoCs usually work below the saturation point, we have considered the correlation for all the workload levels. However, for the sake of shortness we will show here the results for the most representative cases: the lowest workload level, the workload level that makes the network to reach saturation, and the highest workload level that makes the network to work in deep saturation (points $S1$, $S8$ and $S10$). Figure 9 shows the least square linear adjustment for point $S8$ (traffic equal to 0.014 packets/cycle/IP and global average latency equal to 42.73 cycles). The correlation index obtained in this case has been 85 %. These results show that while the network is not under deep saturation, the correlation between the proposed metric and the network performance is high.

Figure 10 shows the regression curve for the point $S10$ Figure 7, with a traffic injection rate equal to 0.018 packets/cycle/IP and an average latency equal to 520.2 cycles. The correlation index is 14 % in this case, showing that (as could be expected)

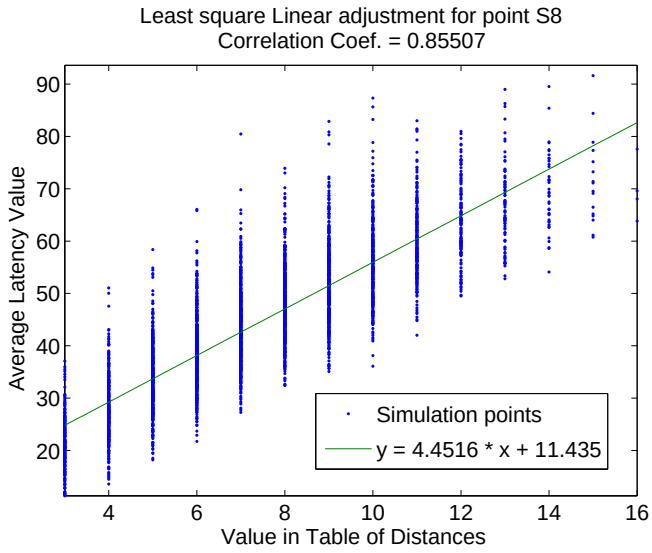


Fig. 9. Least square linear adjustment for simulation point S8

when the network is under deep saturation then the contention prevents the network performance to correlate with inter-node distance.

Nevertheless, in order to study the practical correlation of the proposed metric with real performance, when different values are obtained in different simulations (as this is the case) then we should also consider the long-term network behavior. Therefore, we have also studied the correlation between the equivalent distances and the mean values of the average latencies supplied by the simulator. Figure 11 shows the corresponding regression curve for the simulation point $S10$. In this figure, there is a point for each different value in the table of distances. These values are represented on the X-axis. For information purposes, the number of occurrences for each value in the table of distances is shown on the right side of the figure. For example, point P4 represents value 6 in the table of distances and appears 552 times in the table of distances. The long-term network behavior for this value in the table of distances is much more stable than the network behavior for point P12, whose mean value is computed with 40 different average latencies only. As shown in Figure 11, the mean values of the average latencies show a much higher correlation with the table of distances (96 %) than the average latencies (Figure 10, correlation coefficients of 14 %). The reason for this behavior is that network contention cannot affect this performance metric as it can do for the average latencies.

In order to show the correlation results in a unified way, Figure 12 shows the correlation coefficients provided for each of the points shown in Figure 7. The correlation coefficients for latency values remain about 86–99 % when the network is under a low and a medium load (points $S1$ to $S8$). However, when the network

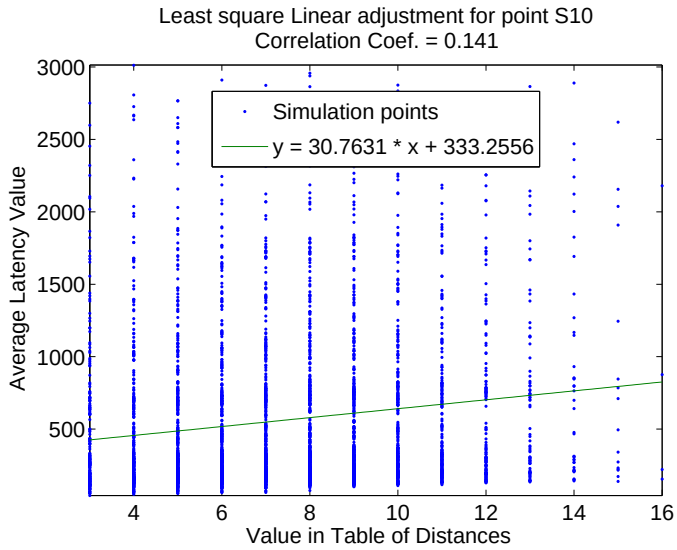


Fig. 10. Least square linear adjustment for simulation point S10

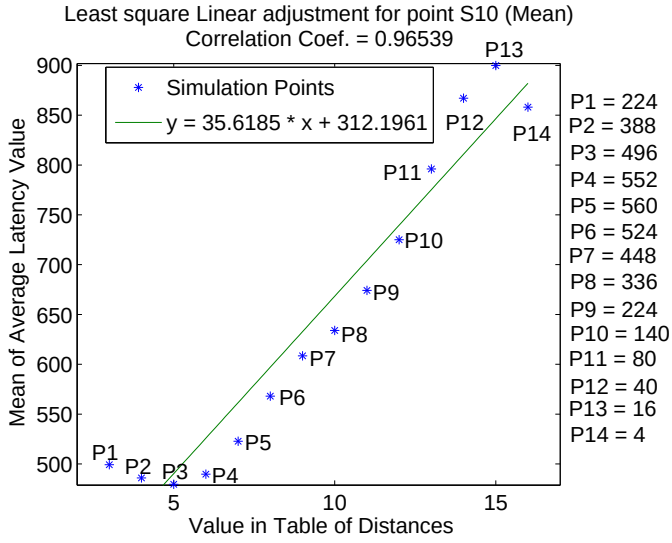


Fig. 11. Least square linear adjustment for averages value of average latencies values (point S10)

enters deep saturation, the correlation coefficients decrease to about 40 %, and even reaching values of 15 %. However, the average values of average latencies obtain a correlation coefficient of about 95 % even when the network is under a deep saturation, showing that the correlation of this parameter is almost independent of the traffic injection rate. These results show that the table of communication costs, computed as shown in Section 3, properly correlates with the actual NoC performance.

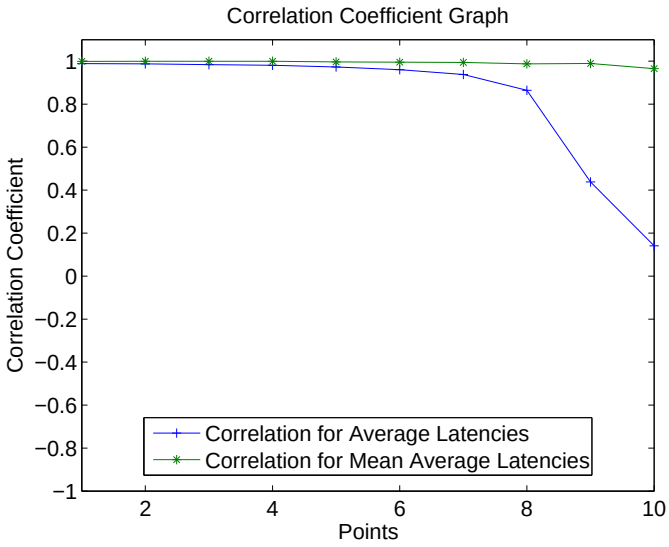


Fig. 12. Correlation of average latencies values and average values of average latencies for a 8×8 NoC topology

5 PERFORMANCE EVALUATION

We propose the evaluation of the proposed technique by simulation. For comparison purposes, we have also evaluated the performance of a previously proposed mapping technique based on Genetic Algorithm (GA) [2]. In order to obtain comparable results, we have considered both the network and the same generic Multi-Media System (MMS) as the application running on the machine [2]. We started from the Communication Task Graph (CTG) of the MMS, shown in Figure 13.

From this CTG, which contains 40 different tasks assigned and scheduled onto 25 IPs, we have computed the table of communication requirements (that is, the 40 original tasks are finally grouped in 25 IPs). Concretely, we have simply taken the communication bandwidth required between each pair of processing nodes as the number of messages exchanged between that pair of processing nodes. Although the required bandwidth is not the same as the number of messages exchanged between

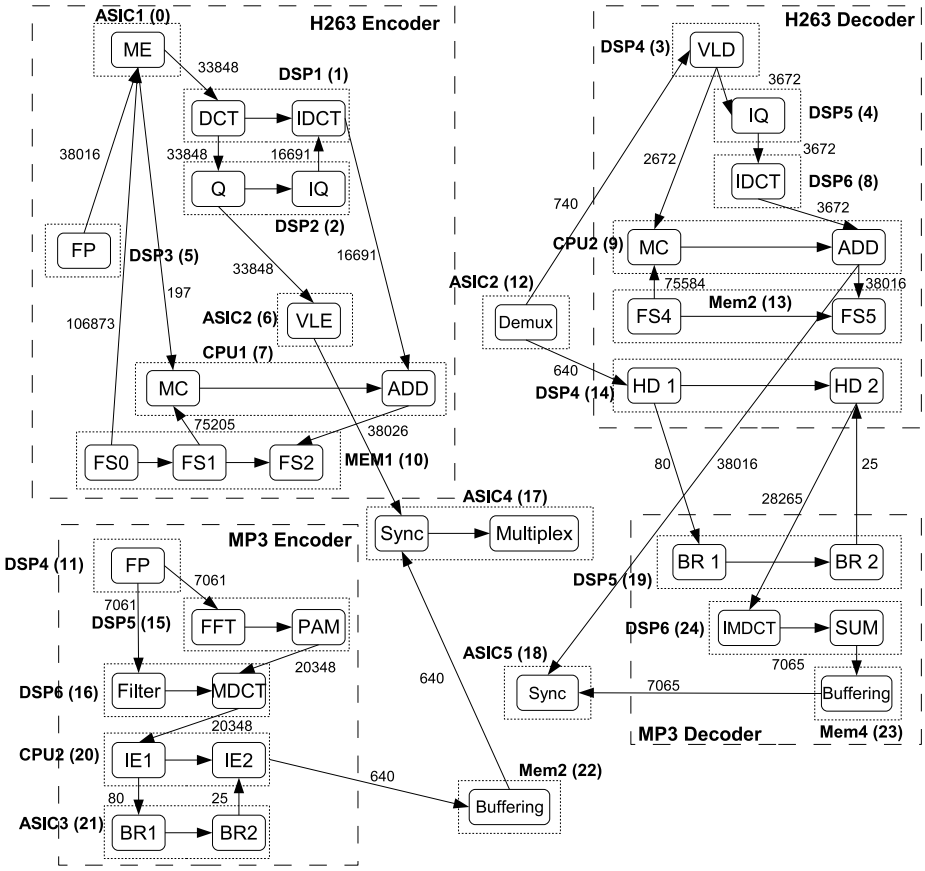


Fig. 13. Communication task graph for the MMS

nodes, in this way we ensure that the proposed technique is evaluated under worse conditions (with approximate values in the table of communications between IPs).

At this point it is worth mentioning that the proposed technique is designed for application-specific NoCs, instead of general-purpose NoCs. Although there are available standard processors of 48 cores like Intel Cloud chip [3] and some forecasts for coming years 2 017 predict processors with thousands of cores on each chip, the current and expected NoC sizes in specific-applications for a near future are of tens of nodes. This number is significantly smaller than for general-purpose application NoCs (the ones in processors like Intel Cloud chip), that are beyond the scope of our technique (it is almost impossible to obtain the communication pattern generated by different applications that continuously change in time).

Although the proposed technique is topology-independent and can be used with both regular and irregular topologies [28], we have used a regular topology for mea-

suring the performance. The reason is that the benefits of the proposed technique increase as so does the irregularity of the network topology, since more differences appear between assigning the IPs to different network nodes. In order to show the performance in the worst case, we have considered a regular network topology for evaluation purposes. The NoC consists of a 5×5 mesh topology with the X-Y routing algorithm and wormhole switching. Figure 14 shows a graphic representation of the table of communication costs for the 5×5 mesh network with X-Y routing, assuming a unit cost for the internal link between each router in the NoC and the corresponding IP. The graph shows the cost assigned by the proposed model to the path for communicating each pair of network nodes. The value at the generic coordinates (x, y) represents the costs for communicating the source network node x with the destination node y.

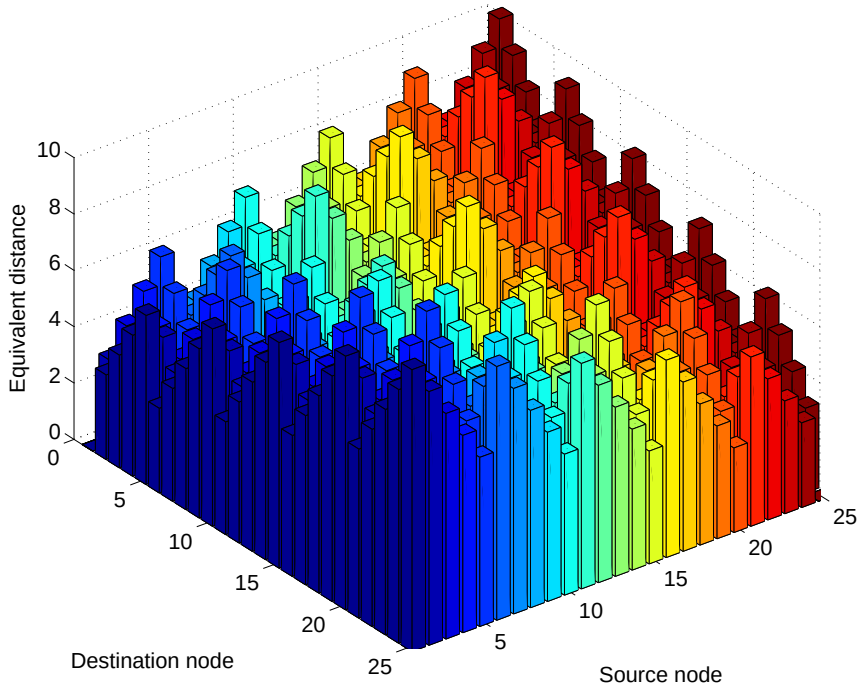


Fig. 14. Graphic representation of the table of equivalent distances between network nodes for a 5×5 network topology using X-Y routing

Using the random search method described in Section 3, we have obtained a near-optimal mapping of IPs to network nodes. Using the same NoC simulator used for the characterization of the network model [22], we have evaluated the performance of the mapping provided by the proposed mapping technique and also the mapping provided by the GA-based technique [2]. In addition, for comparison purposes, we have evaluated the performance provided by two other mapping strategies: a random

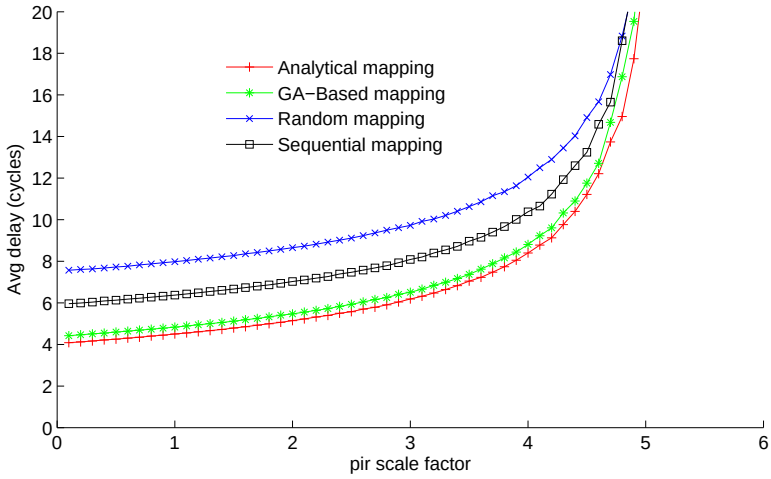


Fig. 15. Average latencies provided by the mapping techniques

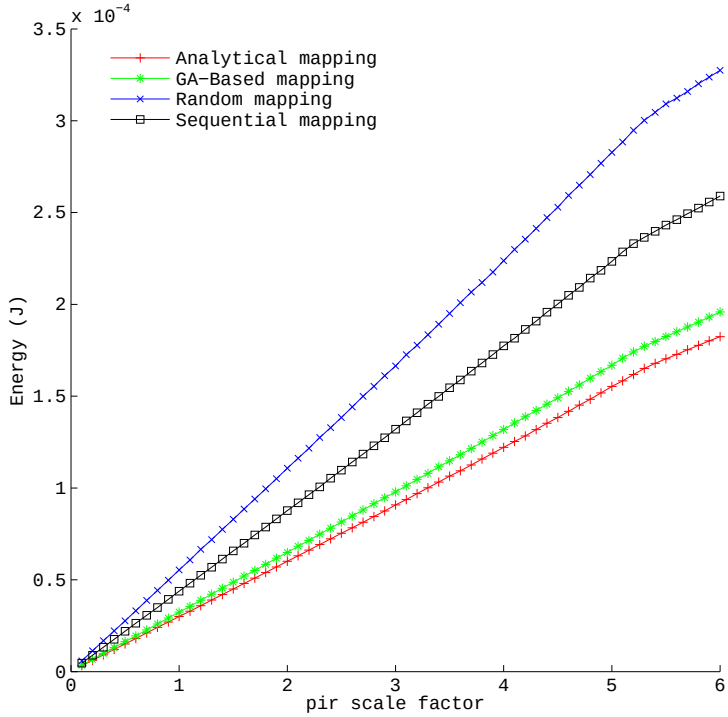


Fig. 16. Energy consumption required by the mapping techniques

mapping as well as a sequential mapping of the MMS. The first one simply computes a random mapping of IPs to network nodes, and the second one assigns the IP labeled with each number to the network node with the same number (the IP labeled as 0 would be assigned to network node 0, IP 1 to network node 1, etc.). Although these strategies are far way from optimal, they can be used as a reference for illustrating the performance improvement that the proposed technique can provide with respect to straightforward mapping strategies that do not require computational efforts.

Figure 15 shows the latency comparative results for the MMS application mapped onto a 5×5 mesh. Each point in these figures has been computed as the average value of 50 different simulations. The plot corresponding to the mapping provided by the communication-aware mapping technique has been labeled as "Analytical mapping", the plot corresponding to the mapping provided by the GA-based technique [2] has been labeled as "GA-Based mapping", and the plots corresponding to the random and the sequential mapping have been labeled with these names. Figure 15 shows the packet injection rates (PIR) on the X-axis. This value indicates the scaling factor used for injecting traffic into the network. Thus, a PIR scale factor of one indicates that the total volume of injected traffic during the simulation is the sum of all the values in the table of communication requirements. A PIR scale factor of two indicates that each element of the table has been increased twice, and therefore the volume of injected traffic during the simulation has been doubled. On the Y-axis, Figure 15 shows the average message latencies obtained during the simulations for each of the considered mappings.

Figure 15 shows that the mapping provided by the proposed technique obtains the lowest latencies. Although the improvement achieved is not significant if compared with the GA-based mapping, it provides similar or better results in terms of latency. If we take into account that the proposed method does not require the simulation of each mapping considered, then these results show that the proposed method adds a much lower overhead than the GA-based mapping technique while providing similar or better results.

Also, we have analyzed the network performance in terms of power consumption. Figure 16 shows the NoC energy consumption for each one of the considered mappings. In this case there are significant differences among the different plots. Figure 16 shows that the mappings provided by the proposed technique require the lowest power consumption. The differences among the considered mapping increase as the traffic injection rates are higher. This behavior is due to the fact that the proposed mapping technique maps as closely as possible those tasks that communicate the highest amount of traffic. As the traffic generated by the tasks increases, the energy saving with respect to other techniques is greater.

Taking into account that the network size is relatively small, these results suggest that the communication-aware mapping technique can help to significantly improve the NoC performance. In order to show that the proposed technique can provide better results for larger network sizes, we have simulated the same application (MMS) with a different scheduling algorithm. In this case, the task scheduling results in 40 tasks mapped each one on a different IP. We have used the proposed technique

to map the 40 IPs onto a 10×4 mesh. Figures 17 and 18 show the evaluation results for different mappings: the sequential mapping, a random mapping and also the mapping provided by the proposed technique. Unfortunately, we have not been able to find out which mapping would provide the GA-based technique in this case, due to the high computational effort needed by the GA-based approach. Therefore, we have not included this mapping for comparison purposes.

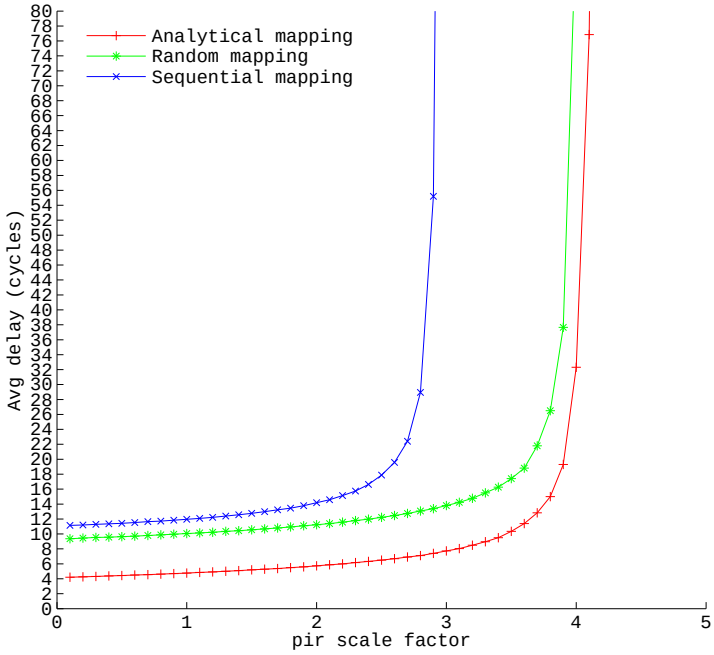


Fig. 17. Latencies provided by different mappings for a 10×4 mesh

Figure 17 shows that there is a significant difference among the latencies provided by the considered mappings. When comparing this figure with Figure 15, it can be seen that while the latency values provided by the proposed technique in Figure 17 are very similar to the ones shown in Figure 15, the values provided by the other mappings are significantly higher.

Figure 18 shows the energy consumption required for the same network when the considered mappings are used. Again, the best results are those provided by the proposed technique. When comparing this figure with Figure 16, it can be seen that the differences between the values provided by the proposed technique and the values provided by the other two mappings are higher in Figure 18 than in Figure 16. These results suggest that the proposed technique can provide a better

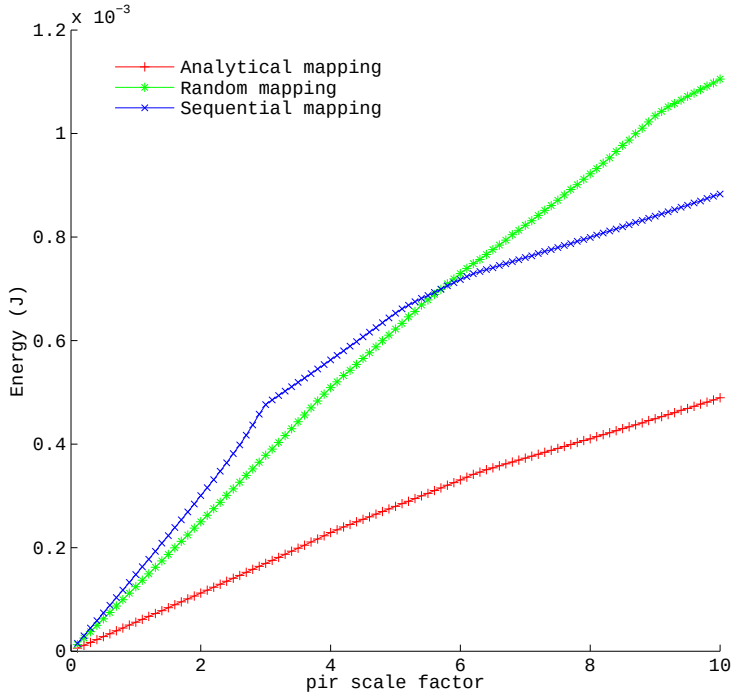


Fig. 18. Energy consumption required by different mappings for a 10×4 mesh

network performance, particularly in terms of energy consumption, as the network size increases.

In order to ensure that the NoC communications can take advantage of the proposed technique regardless of the communication pattern, we have also evaluated the proposed technique under synthetic traffic. However, the corresponding tables of communication requirements are not shown here for the sake of shortness. Concretely, we have used the bit-reversal, the butterfly and the shuffle distribution [7]. Also, for the sake of shortness we will show here the results for the bit-reversal and the shuffle distribution.

Figure 19 shows the latencies provided by different mappings for the bit-reversal traffic distribution. For comparison purposes, we have used a random mapping as well as the sequential mapping. Figure 19 shows that the proposed method provides the best results not only in terms of latency (it provides the lowest latencies) but also (and mainly) in terms of network throughput. The reason for that behavior is to locate as close as possible those IPS exchanging the highest amount of information, adapting the locality in the communication pattern to the network topology.

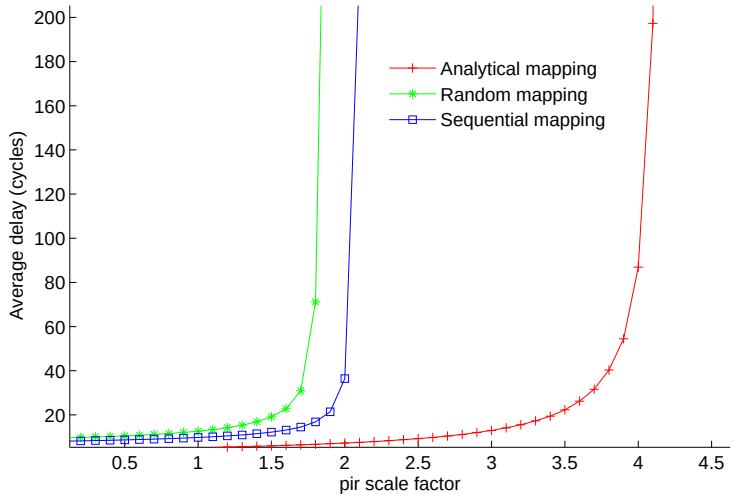


Fig. 19. Latencies provided by different mappings for the bit-reversal traffic distribution

As a result, the most frequent paths are also the shortest ones, and the network throughput is increased.

Figure 20 shows the energy consumption required by different mappings for the bit-reversal traffic distribution. Since the power consumption is directly related to the length of the path traversed by the messages exchanged by the application, this

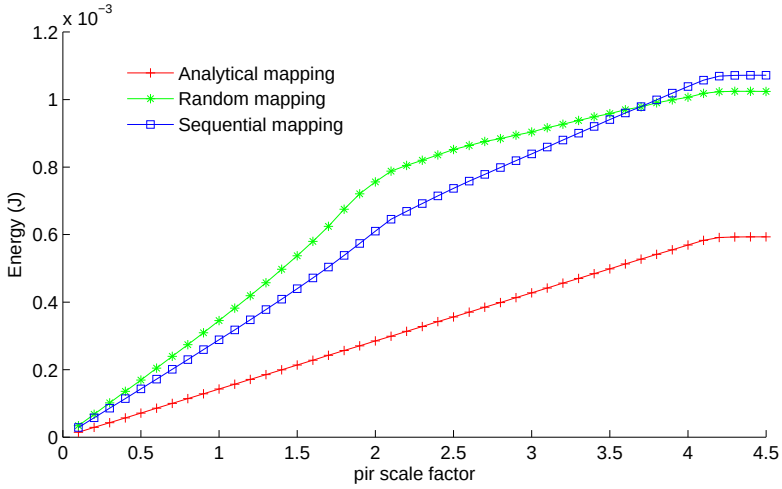


Fig. 20. Energy consumption required by different mappings for the bit-reversal traffic distribution

figure shows that the mapping provided by the proposed technique requires lower energy consumption than the rest of the mappings. As the traffic injection rate increases, so does this trend.

Figure 21 shows the latencies provided by different mappings for the shuffle traffic distribution. In this case, the differences in terms of network throughput are even larger than in the case of the bit-reversal traffic distribution (Figure 19). The reason is that the traffic pattern of the former distribution is even more clustered than the traffic pattern of the latter one.

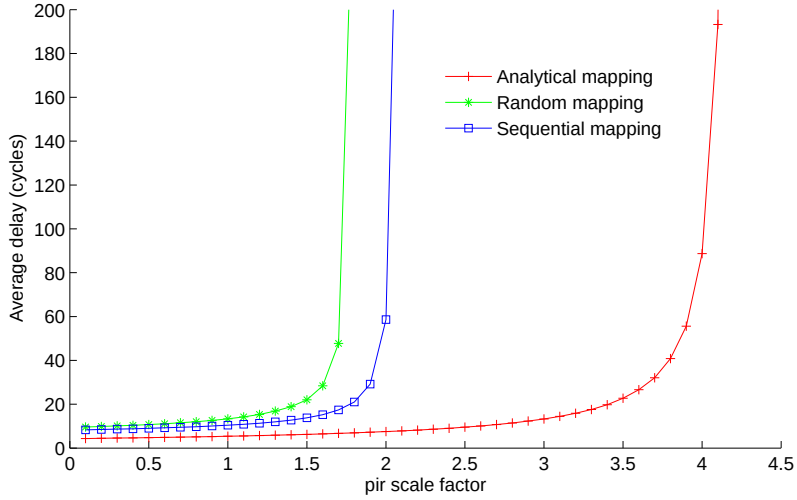


Fig. 21. Latencies provided by different mappings for the shuffle traffic distribution

Finally, Figure 22 shows the energy consumption required by different mappings for the shuffle traffic distribution. As could be expected, the behavior shown in this figure is very similar to that shown in Figure 20, showing that the proposed technique is able to provide a significant power saving in regard to the rest of mappings.

6 CONCLUSIONS

In this paper, we have proposed, in an extended manner, a topology-independent mapping technique for NoCs that globally matches the communication pattern generated by the IPs with the available network bandwidth in the different parts of the network. Unlike other mapping techniques, the proposed method uses a table of communication costs as the model of the NoC topology and routing algorithm (the network resources). As a result, this method can be used with either regular or irregular topologies. This feature can help fully exploit those irregular NoCs resulting from manufacturing defects, even real-time failures or even the methodology design

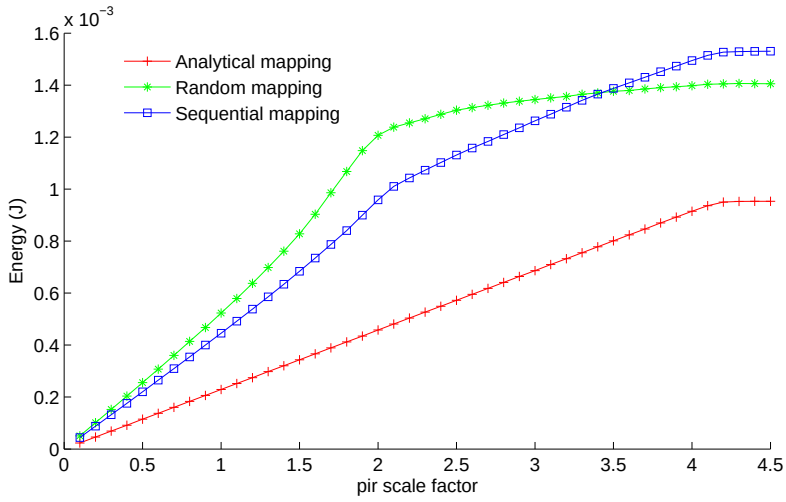


Fig. 22. Energy consumption required by different mappings for the shuffle traffic distribution

flow. The underlying methodology in the proposed technique consists of correlating first the model of the NoC with the actual NoC performance.

The characterization results show that the proposed model of network resources properly correlates with actual network performance, and therefore this model can be used to estimate the quality of each assignment. A random search method is used to obtain the best global assignment of IPs to network nodes.

The performance evaluation results show that the proposed technique provides better performance than other mapping techniques not only in terms of average latency and network throughput, but also in terms of power consumption.

Acknowledgements

This work has been jointly supported by the Spanish MICINN and the European Commission FEDER funds under grants Consolider-Ingenio 2010 CSD2006-00046 and TIN2009-14475-C04-04.

REFERENCES

[1] Arnau, V.—Orduña, J. M.—Ruiz, A.—Duato, J.: On the Characterization of Interconnection Networks with Irregular Topology: A New Model of Communication Cost. In XI IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS '99), IASTED Press 1999, pp. 1–6.

[2] Ascia, G.—Catania, V.—Palesi, M.: Mapping Cores on Network-on-Chip. International Journal of Computational Intelligence Research, Vol. 1, 2005, Nos. 1–2, pp. 109–126.

[3] Baron, M.: The Single-Chip Cloud Computer: Intel Networks 48 Pentiums on a Chip. Technical report, Intel Corporation: Microprocessor Report 2010.

[4] Benini, L.: Application Specific NoC Design. In DATE'06: Proceedings of the Conference on Design, Automation and Test in Europe, Leuven, Belgium 2006, pp. 491–495.

[5] Bolotin, E.—Cidon, I.—Ginosar, R.—Kolodny, A.: Routing Table Minimization for Irregular Mesh NoCs. In Proceedings of Design, Automation andTest in Europe, DATE'07 Conference and Exhibition 2007, pp. 1–6.

[6] Chang, J.-M.—Pedram, M.: Codex-dp: Co-Design of Communicating Systems Using Dynamic programming. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 19, 2002, No. 7, pp. 732–744.

[7] Duato, J.—Yalamanchili, S.—Ni, L.: Interconnection Networks: An Engineering Approach. IEEE Computer Society 2003.

[8] Festa, P.—Resende, M. G. C.: Grasp: An Annotated Bibliography. In P. Hansen and C. C. Ribeiro (Eds.): Essays and Surveys on Metaheuristics, Kluwer Academic Publishers 2002, pp. 325–367.

[9] Garey, M. R.—Johnson, D. S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. Freeman and Company 1979.

[10] Glover, F.—Laguna, F.: Tabu Search. Kluwer Academic Publishers, Norwell, MA, USA 1997.

[11] Goossens, K.—Dielissen, J.—Gangwal, O. P.—Pestana, S. G.—Rădulescu, A.—Rijpkema, E.: A Design Flow for Application-Specific Networks on Chip with Guaranteed Performance to Accelerate SoC Design and Verification. In DATE'05: Proceedings of the Conference on Design, Automation and Test in Europe, Washington, DC, USA, 2005, IEEE Computer Society, pp. 1182–1187.

[12] Hansson, A.—Goossens, K.—Rădulescu, A.: A Unified Approach to Constrained Mapping and Routing on Network-on-Chip Architectures. In CODES + ISSS'05: Proceedings of the 3rd IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis, New York, NY, USA 2005, pp. 75–80.

[13] Haupt, R. L.—Haupt, S. E.: Practical Genetic Algorithms. Willey 1997.

[14] Hu, J.—Marculescu, R.: Energy- and Performance-Aware Mapping for Regular NoC Architectures. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 24, 2005, No. 4, pp. 551–562.

[15] Keijzer, M.—Merelo Guervós, J. J.—Romero, G.—Schoenauer, M.: Evolving Objects: A General Purpose Evolutionary Computation Library. In Selected Papers from the 5th European Conference on Artificial Evolution, London, UK 2002, pp. 231–244.

[16] Laarhoven, P. J.—Aarts, E. H.: Simulated Annealing: Theory and Applications. Volume 37 of Mathematics and its Applications. Springer-Verlag 1987.

[17] Lei, T.—Kumar, Sh.: A Two-Step Genetic Algorithm for Mapping Task Graphs to a Network on Chip Architecture. In Digital System Design, 2003, Proceedings of Euromicro Symposium, September 2003, pp. 180–187.

[18] Lu, Zh.—Xia, L.—Jantsch, A.: Cluster-Based Simulated Annealing for Mapping Cores Onto 2D Mesh Networks on Chip. In 11th IEEE Workshop on Design and Diagnostics of Electronic Circuits and Systems – DDECS 2008, pp. 1–6.

[19] Mejia, A.—Flich, J.—Duato, J.—Reinemo, S.-A.—Skeie, T.: Segment-Based Routing: An Efficient Fault-Tolerant Routing Algorithm for Meshes and Tori. In 20th International Symposium on Parallel and Distributed Processing – IPDPS 2006, p. 10.

[20] Murali, S.—Coenen, M.—Radulescu, A.—Goossens, K.—de Micheli, G.: A Methodology for Mapping Multiple Use-Cases Onto Networks on Chips. In DATE '06: Proceedings of the Conference on Design, Automation and Test in Europe, 3001 Leuven, Belgium, pp. 118–123.

[21] Murali, S.—de Micheli, G.: Bandwidth-Constrained Mapping of Cores Onto NoC Architectures. In DATE '04: Proceedings of the Conference on Design, Automation and Test in Europe, Washington, DC, USA 2004, p. 20896.

[22] Noxim. Network-on-Chip Simulator. In `http://noxim.sourceforge.net`.

[23] Orduña, J. M.—Silla, F.—Duato, J.: On the Development of a Communication-Aware Task Mapping Technique. Journal of Systems Architecture, Vol. 50, 2004, No. 4, pp. 207–220.

[24] Porto, S.—Ribeiro, C.: A Tabu Search Approach to Task Scheduling on Heterogeneous Processors Under Precedence Constraints. International Journal of High Speed Computing, Vol. 7, 1995, No. 1, pp. 45–71.

[25] Schafer, M. K. F.—Hollstein, T.—Zimmer, H.—Glesner, M.: Deadlock-Free Routing and Component Placement for Irregular Mesh-Based Networks-on-Chip. In Proceedings of IEEE/ACM International Conference on Computer-Aided Design 2005 (ICCAD 2005), pp. 238–245.

[26] Soriano, R.—Orduña, J.-M.: Improving the Scalability of Communication-Aware Task Mapping Techniques. In Advanced Information Networking and Applications Workshops, International Conference, Los Alamitos, CA, USA 2009, pp. 1061–1066.

[27] Tornero, R.—Orduña, J.-M.—Palesi, M.—Duato, J.: A Communication-Aware Topological Mapping Technique for NoCs. In: Proceedings of the 14th international Euro-Par Conference on Parallel Processing – Euro-Par '08, pp. 910–919.

[28] Tornero, R.—Orduña, J.-M.—Mejía, A.—Flich, J.—Duato, J.: CART: Communication-Aware Routing Technique for Application-Specific NoCs. In Digital System Design Architectures, Methods and Tools – DSD '08, 11th EUROMICRO Conference, pp. 26–31.

[29] Varatkar, G.—Marculescu, R.: Traffic Analysis for On-Chip Networks Design of Multimedia Applications. In ACM/IEEE Design Automation Conference, June 2002, pp. 510–517.

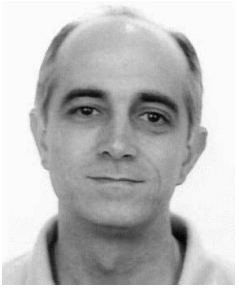
[30] Wu, Ch.-M.—Chi, H.-Ch.—Lee, M.-Ch.: Mapping of IP Cores to Network-On-Chip Architectures Based on Communication Task Graphs. In 6th International Conference ASIC 2005, ASICON 2005, Vol. 2, pp. 953–956.

Rafael Tornero is a Ph. D. student at University of Valencia, Spain. He received his Bachelor's degree in Computer and Telecommunication Engineering at University of Valencia in 2005 and 2006 respectively. Currently, he belongs to the Networks and Virtual Environments Group (GREV) at the University of Valencia, which is part of the ACCA Group. His research addresses Core and Task Mapping onto Network on Chip (NoCs). He has published his works in international conferences like EuroPar, DSD and IPDPS.

Juan M. Orduña received the M. Sc. degree in computer engineering from the Technical University of Valencia, Valencia, Spain, in 1990 and the Ph. D. degree in computer engineering from the University of Valencia in 1998. His research has been developed within the ACCA team (`http://www.acca-group.info/`). He was a Computer Engineer with Telefónica de España, Manpel Electrónica, S. A. and the Technical University of Valencia. He is currently a Lecturer Professor with the Department of Informatics, University of Valencia, where he leads the Networking and Virtual Environments Group. He is a member of the HiPEAC network of excellence, and his research is currently supported by the Spanish MEC and the European Commission through several projects. His research currently focuses on networks on chip, collaborative augmented reality, multiagent systems, and crowd simulations. He has published papers about his research in a number of international journals and conferences. Dr. Orduña has served as a Program Committee Member in different conferences and workshops (e.g., IEEE International Conference on Parallel Processing, European Conference on Parallel Processing, IEEE Virtual Reality Conference, and International Conference on Parallel and Distributed Systems), as well as a reviewer for journals like IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED S YSTEMS, Journal of Network and Computer Applications, IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, etc.

Maurizio Palesi received the M. Sc. and Ph. D. degrees in computer engineering from the University of Catania, Italy, in 1999 and 2003, respectively. Since November 2010 he is Assistant Professor at Kore University, Italy. He serves on the Editorial Board of VLSI Design journal as an Associate Editor since May 2007. He has served as a Guest Editor for the VLSI Design Journal, the International Journal of High Performance Systems Architecture, Elsevier MICPRO Journal and ACM Transactions on Embedded Computing Systems. He is in the Technical Program Committee of several IEEE/ACM International Conferences including DATE, RTAS, CODES+ISSS, ESTIMedia, NOCS, SOCC, VLSI, ISC, and SITIS.

He has been co-organizer of the four editions of the International Workshop on Network-on-Chip Architectures (from 2008 to 2011). He is member of the European Network of Excellence on High Performance and Embedded Architecture and Compilation (HiPEAC).



José Duato received the M. Sc. and Ph. D. degrees in electrical engineering from the Technical University of Valencia, Valencia, Spain, in 1981 and 1985, respectively. He is currently a Professor with the Department of Computer Engineering (DISCA), Technical University of Valencia. He was an Adjunct Professor with the Department of Computer and Information Science, The Ohio State University, Columbus. He has published over 400 refereed papers. He proposed a powerful theory of deadlock-free adaptive routing for wormhole networks. Versions of this theory have been used in the design of the routing algorithms for the MIT Reliable Router, the Cray T3E supercomputer, the on-chip router of the Alpha 21364 microprocessor, and the IBM BlueGene/L supercomputer. He also developed RECN, the only truly scalable congestion management technique proposed to date, and a very efficient routing algorithm for fat trees that has been incorporated into Sun Microsystem's 3 456-port InfiniBand Magnum switch. He leads the Advanced Technology Group in the HyperTransport Consortium, whose main result to date has been the development and standardization of an extension to HyperTransport (High Node Count HyperTransport Specification 1.0) that extends the device addressing capabilities of HyperTransport in several orders of magnitude. He is the first author of the book Interconnection Networks: An Engineering Approach (IEEE Computer Society Press, 1997). His current research interests include interconnection networks and multiprocessor architectures. He has served as a member of the Editorial Boards of the IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Computers, and IEEE Computer Architecture Letters. He has been the General Cochair for the 2001 IEEE International Conference on Parallel Processing (ICCP), the Program Committee Chair for the Tenth International Symposium on High Performance Computer Architecture (HPCA-10), and the Program Cochair for the 2005 IEEE ICCP. He also served as a Cochair, a Steering Committee Member, a Vice Chair, or a Program Committee Member in more than 60 conferences, including the most prestigious conferences in his area (e.g., HPCA, International Symposium on Computer Architecture, International Parallel Processing Symposium/Symposium on Parallel and Distributed Processing, IEEE International Parallel and Distributed Processing Symposium, IEEE International Conference on Parallel Processing, International Conference on Distributed Computing System, European Conference on Parallel Processing, and International Conference on High Performance Computing).