*Journal of Universal Computer Science, vol. 18, no. 7 (2012), 937-969*
*submitted: 11/11/11, accepted: 27/3/12, appeared: 1/4/12 © J.UCS*

# Designing Robust Routing Algorithms and Mapping Cores in Networks-on-Chip: A Multi-objective Evolutionary-based Approach

**Maurizio Palesi**
(Kore University, Enna, Italy
maurizio.palesi@unikore.it)

**Rafael Tornero, Juan Manuel Orduña**
(Universidad de Valencia, Spain
Rafael.Tornero@uv.es, Juan.Orduna@uv.es)

**Vincenzo Catania, Daniela Panno**
(University of Catania, Italy
vcatania@dieei.unict.it, daniela.panno@dieei.unict.it)

**Abstract:** Mainstream electronic designs are realized by Systems-on-Chips (SoCs) that push the limits of integration. The advancement of manufacturing technologies in terms of integration leads us to SoCs with many (*e.g.*, 10–1000) digital units (*e.g.*, processor cores, controllers, storage, application-specific units) that need to be interconnected in an efficient and reliable way. The Network-on-Chip (NoC) design paradigm emerged recently as a promising alternative to classical bus-based communication architectures. Aside from better predictability and lower power consumption, the NoC approach offers greater scalability compared to previous solutions for on-chip communication. The design flow of NoCs include several key issues. Among other parameters, the decision of where cores have to be topologically mapped and also the routing algorithm represent two highly correlated design problems that must be carefully solved for any given application in order to optimize different performance metrics. The strong correlation between the different parameters often makes that the optimization of a given performance metric has a negative effect on a different performance metric. In this paper we propose a new strategy that simultaneously refines the mapping and the routing function to determine the Pareto optimal configurations which optimize average communication delay and routing robustness. The proposed strategy has been applied on both synthetic and real traffic scenarios. The obtained results show how the solutions found by the proposed approach outperforms those provided by other approaches proposed in literature, in terms of both performance and fault tolerance.

**Key Words:** Networks-on-Chip; Topological Mapping; Multi-objective Optimization; Routing algorithm; Genetic Algorithm; Fault-tolerance; Performance Analysis.
**Category:** B.4.3, J.6

## 1 Introduction

The continue improvement in silicon technology, together with the evolution of design methodologies, allow to integrate a complex computing system in a single silicon die nowadays. Such Systems-on-Chip (SoCs) pervade our daily life as almost any modern embedded system today is build around a SoC targeted for a certain application domain. For instance, mobile phones, digital cameras, set-top-box, network appliances, *etc.* represents just a short list of daily-use devices which stress the use of SoCs to deal with cost, performance, and power issues.

As silicon technology shrinks an even more significant number of Intellectual Properties (IPs or cores) is integrated into a SoC. Such IPs span from computational elements (like microprocessor, co-processor, accelerator *etc.*) to storage elements (like embedded DRAM, SRAM, *etc.*), application specific IPs, reconfigurable logic (in the form of embedded FPGA), and analog modules. We have entered the many-core era, in which the number of cores which form the SoC has changed from units to hundreds. The Intel's 80-tiles TeraFLOPS processor [Vangal et al., 2008] represents an emblematic example of the on-chip computational capabilities which can be achieved pushing toward many-cores architectures. As compared to ASCI Red [Mattson et al., 1996], the first computer to reach a Teraflops of processing in 1996 formed by 10,000 Pentium running at 200 MHz and consuming about 500 KWatt of power (plus 500 KWatt to keep the room cool), the TeraFLOPS processor exhibits the same computational capability but dissipating less than 70 Watt.

The International Technology Roadmap for Semiconductors (ITRS) predicts that the number of cores in a SoC will increase from about 80 in 2010, to 270 in 2015, and to 880 in 2020 [ITRS, 2007]. As the number of cores increases, the role played by the on-chip communication systems becomes more and more important. Historically, computation has been expensive and communication cheap. With scaling microchip technologies, this issue has changed. Computation is becoming cheaper, while communication faces fundamental physical limitations such as time-of-flight of electrical signals, power use in driving long wires, *etc.*. On-chip wires do not scale in the same manner as transistors do, and the cost gap between computation and communication is widening.

To mitigate these effects, the Network-on-Chip (NoC) approach emerged recently as a promising alternative to classical bus-based and some point-to-point communication architectures [Dally and Towles, 2001, Jantsch and Tenhunen, 2003, De Micheli and Benini, 2006]. Aside from better predictability and lower power consumption, the NoC approach offers greater scalability compared to previous solutions for on-chip communication [Marculescu et al., 2009].

Modern SoC architectures consist of heterogeneous IP cores such as CPU or DSP modules, video processors, embedded memory blocks, *etc.*. In the NoC approach, each IP is attached to a local router which connects the IP to the neighboring nodes via a NoC. NoC architectures [Benini and De Micheli, 2002, Dally and Towles, 2001, Jantsch and Tenhunen, 2003, Kumar et al., 2002] present a NoC like an interconnection platform based on packet switching composed of routers and point to point links among them. The implementation for the routers vary

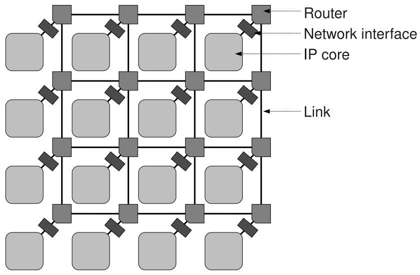



Figure 1: Topological illustration of a 4x4 mesh based NoC architecture, indicating the fundamental elements.

widely according to routing protocols, control flow mechanism and switching technique used. Although several network topologies have been considered, two-dimensional meshes are the preferred topologies. The reasons are their ease of integration on a square surface, the low and constant link delay, and the good parasitic properties they provide. Fig. 1 illustrates the basic elements which characterize a 2-D mesh-based NoC. It shows a NoC structured as a 4-by-4 grid (a.k.a. mesh topology) which provides global chip level communication. Instead of busses and dedicated point-to-point links, a more general scheme is adapted, employing a grid of *routing nodes* spread out across the chip, connected by *communication links*. The most important elements of a NoC are as follows [Bjerregaard and Mahadevan, 2006]. *Network interfaces* implement the interface by which cores connect to the NoC. Their function is to decouple computation (the cores) from communication (the network). *Routing nodes* route the data according to chosen protocols. They implement the routing strategy. *Links* connect the nodes, providing the raw bandwidth. They may consist of one or more logical or physical channels.

### 1.1 Motivation

The use of NoCs as the interconnection infrastructure for complex SoCs has opened several interesting research and design issues [Ogras et al., 2005, Marculescu et al., 2009]. The optimal NoC design depends on multidimensional aspects like topology, routing, buffer allocation, mapping, *etc.*. These aspects determine the overall performance, cost and reliability of the system [Pande et al., 2005]. Since these parameters strongly depend and interact each other, the optimal NoC configuration cannot be found by independently exploring each parameter (*i.e.*, separately from each other). As a result, the design space to be explored when searching the best NoC configuration can be as wide as the Cartesian product of the set of the NoC parameters.

As an example, let us consider the design of a small NoC-based SoC optimized for a given application which is mapped on 16 cores. Suppose that, due to certain design constraints, we can build our NoC-based SoC selecting among three different network topologies (*e.g.*, mesh, torus, and octagon) and two different routing algorithms for each topology. There are 16! possible combinations to map 16 cores onto the NoC. Thus, the size of the design space to be explored in order to find the best topology, the best routing algorithm, and the best mapping for that specific application consists of $6 \times 16!$ network configurations (thousand of billions of configurations). Additionally, the NoC design is rarely aimed to the optimization of a single objective. In practical cases, the parameters to be optimized can be many and conflicting each other [Ascia et al., 2007]. Typical examples are silicon area, energy consumption, performance, *etc.*. This multi-objective design adds more complexity to the search within a vast design space [Marculescu et al., 2009].

### 1.2 Contribution

In order to face the complexity of the NoC design problem, in this paper we propose a new multi-objective design optimization strategy that is able to determine an approximation of the Pareto-optimal set of NoC configurations which optimize multiple performance indices simultaneously. In this way, the design flow can consider multidimensional aspects and it can provide a multi-objective optimization of the NoC. Concretely, we have considered the *routing algorithm* and the *topological mapping* (*i.e.*, on which network node should each core be mapped to) as the multidimensional interacting parameters. We have considered *average communication delay* and *fault tolerance* as the objectives to be optimized. We have used a multi-objective evolutionary approach based on Genetic Algorithms as a strategy for exploring the design space. To the best of our knowledge, there are no work in literature that tackle the mapping and routing problem concurrently in a multi-objective scenario. The results obtained from experiments on both synthetic and real traffic patterns show that the proposed strategy provides better solutions than the sequential optimization of the same parameters. Therefore, these results show that the high degree of interaction among these design parameters require a global system optimization, rather than a separate study.

The rest of the paper is organized as follows. In Section 2, we discuss related research work. In Section 3, we define the required terminology and formulate the problem. Section 4 describes the performance indices to be optimized. Section 5 presents the proposed design space exploration strategy, while its application on both synthetic and real traffic scenarios is presented in Section 6. Finally, Section 7 summarizes the contribution of the paper.

## 2 Related Work

Several approaches have been proposed in literature in the context of topological mapping in NoCs.

Hu and Marculescu present a branch and bound algorithm for mapping cores in a mesh-based NoC architecture that minimizes the total amount of power consumed in communications with the constraint of performance handled via bandwidth reservation [Hu and Marculescu, 2003]. The same authors extend this approach by adding the customization of the routing paths such that the total communication energy is minimized [Hu and Marculescu, 2005]. Murali and De Micheli address the mapping problem under the bandwidth constraint with the aim of minimizing communication delay by exploiting the possibility of splitting the traffic among various paths [Murali and De Micheli, 2004]. Hansson *et al.* present a unified single-objective algorithm which couples path selection, mapping of cores, and channel time-slot allocation to minimize the network required to meet the constraints of the application [Hansson et al., 2007]. Lei and Kumar present an approach that uses genetic algorithms to map an application, described as a parameterized task graph, on a mesh-based NoC architecture [Lei and Kumar, 2003]. Ascia *et al.* use multi-objective genetic algorithms to mapping cores in a mesh-based NoC to optimize performance and energy consumption [Ascia et al., 2004]. Tornero *et al.* present a communication-aware topological mapping technique for NoCs that, based on on the experimental correlation of the network model with the actual network performance, avoids the need to experimentally evaluate each mapping explored [Tornero et al., 2008b]. This technique has been applied for developing a communication-aware routing method with the aim of optimizing the network performance (in terms of average delay) for an application-specific NoCs [Tornero et al., 2008]. Jena and Sharma propose a multi-objective evolutionary algorithm to obtain the Pareto mappings that minimize the energy consumption and link bandwidth requirement under performance constraints [Jena and Sharma, 2008]. Chou and Marculescu present an integer linear programming (ILP) formulation of the contention-aware application mapping problem which aims at minimizing the inter-tile network contention, in order to decrease the average packet latency. Also, they propose a linear programming (LP) approach followed by a mapping heuristic for solving the scalability problem caused by ILP formulation [Chou and Marculescu, 2008].

Except [Ascia et al., 2004, Jena and Sharma, 2008], all the other aforementioned works define mapping strategies which optimize a single objective: energy consumption in [Hu and Marculescu, 2003, Hu and Marculescu, 2005], communication cost in [Murali and De Micheli, 2004], power dissipation and worst-case latency in [Hansson et al., 2007], and average delay in [Tornero et al., 2008b, Chou and Marculescu, 2008]. In addition, except [Hu and Marculescu,

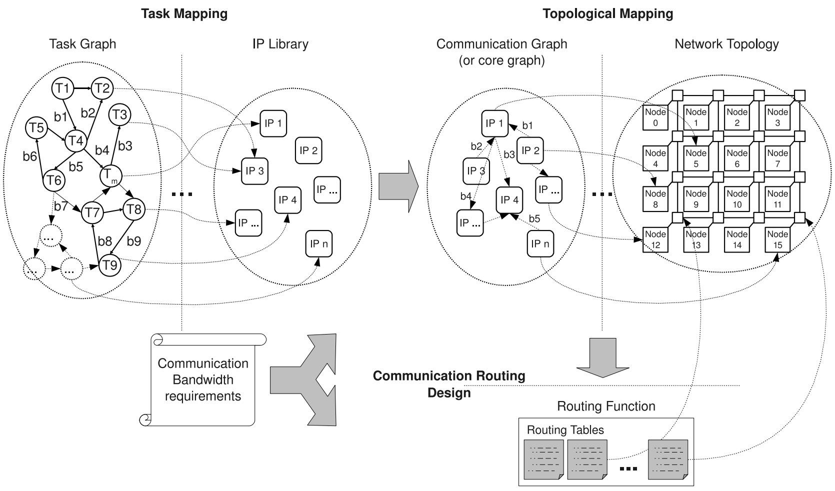



**Figure 2:** Pictorial illustration of the mapping and routing problem.

2005,Hansson et al., 2007], they assume a given routing function when exploring the mapping space.

In this paper we present a new mapping strategy which tackles the mapping and routing problem *concurrently* in a *multi-objective* fashion. The goal of the proposed approach is to obtain the Pareto set of mapping configurations along with their customized routing functions which minimize average delay and maximize the fault tolerance properties of the communication system. This paper is an extension of our previous work [Tornero et al., 2009]. With respect to [Tornero et al., 2009] in this paper the problem formulation is formally stated, new optimization metrics have been considered, and application-specific genetic operators have been introduced.

## 3 Terminology and Problem Formulation

Simply stated, our goal is to achieve a NoC design that minimizes the communication delay and improves the fault tolerance properties of the communication system. In order to achieve this goal, for a given application and a given network topology we must decide on which network node should each IP core be mapped to, as well as how the packets should be routed.

Fig. 2 depicts the mapping problem. We assume that the application has been already divided into a graph of concurrent tasks and that these tasks have been already mapped and scheduled into a set o available IPs. In fact, the above issues are not new to the CAD community, since they have been addressed in the area of hardware/software co-design and IP-reuse [Chang and Pedram, 2002].

Thus, the main problem here is to decide to which network node each selected IP should be mapped and, at the same time, define the communication routing paths in such a way that some metrics of interest are optimized and application communication bandwidth requirements are satisfied.

In order to formulate the problem, we need the following definitions.

**Definition 1.** The *Communication graph*, $CG = G(T, C)$, is a directed graph, where $T$ is the set of IPs and $C$ is the set of communications. Each communication $c_{i,j} = (t_i, t_j) \in C$ connects IP $t_i \in T$ to IP $t_j \in T$. For a communication $c \in C$, the function $B(c)$ returns the bandwidth requirement of $c$. This is the minimum bandwidth that should be allocated by the network in order to meet the performance constraints for communication $c$.

**Definition 2.** The *Topology graph*, $TG = G(N, L)$, is a directed graph which models the network topology. $N$ is the set of network nodes, and $L$ is the set of network channels. Channel $l_{i,j} = (n_i, n_j)$ connects node $n_i \in N$ to node $n_j \in N$. Given a channel $l \in L$, the function $Cap(l)$ returns its capacity.

**Definition 3.** The *Mapping function*, $M : T \to N$, maps IPs to network nodes [*e.g.*, if $M(t_i) = n_j$ then IP $t_i$ is mapped on network node $n_j$].

**Definition 4.** A *Routing Function* for a node $n \in N$, is a function $R(n)$ : $L_{in}(n) \times N \to \wp(L_{out}(n))$. Where, $L_{in}(n)$ and $L_{out}(n)$ are, respectively, the set of input channels and output channels for node $n$, and $\wp$ indicates a power set. Thus, $R(n_s)(l, n_d)$ gives the set of output channels of node $n_s$ that can be used to send a message received from the input channel $l$ and whose destination is $n_d \in N$. We indicate with $R = \{R(n) : n \in N\}$ the set of routing functions (one for each node of the network)

Therefore, the problem we are solving in the paper can be stated as follows. Given a communication graph $CG(T, C)$ and a topology graph $TG(N, L)$, we want to find mapping function $M$ and a routing function $R$ such that the average communication delay is minimized, the network robustness is maximized, and the aggregated communications assigned to any channel do not exceed its capacity. In order to achieve this goal, we have defined two optimization indices appropriated for steering the exploration of the mapping space. The first one, called *mapping coefficient* $MC$ (*cf.* Subsection 4.1), is correlated with the average delay exhibited by a network with topology $TG$ under a communication traffic $CG$ when a routing algorithm $R$ is used. The second one, called *robustness index* $RI$ (*cf.* Subsection 4.2), estimates the ability of the network in tolerating link faults.

Using these indices, we can formally define the problem as to find the Pareto set of mapping and routing functions which:

$$\min MC(CG, TG, R, M), \tag{1}$$
$$\max RI(CG, TG, R, M), \tag{2}$$

such that:

$$\sum_{c \in C} EB(c, l) \leq Cap(l),\ \forall\ l \in L. \tag{3}$$

Where $EB(c, l)$ represents the effective bandwidth of communication $c$ on channel $l$, that is, the fraction of $B(c)$ which is allocated on $l$. With this problem formulation, the effectiveness of the solutions found by the search will rely on the correlation of the optimization indices with the actual performance. The next section describes in detail these indices.

## 4 Optimization Indices

In this section we discuss in details the two objectives, named the *Mapping Coefficient*, $MC$, and the *Robustness Index*, $RI$, introduced in the previous section.

### 4.1 Mapping Coefficient

For a given communication graph, and a given routing algorithm the most accurate way to evaluate the network performance when a certain mapping function is considered is running a simulation. Unfortunately, simulation-based evaluation is a very expensive approach in terms of computational effort [Ascia et al., 2004]. This problem is particularly evident in the context of design space exploration where the number of system configurations[1] to be visited (*i.e.*, simulated) to determine the solutions is very high. To deal with this problem the use of analytical models able to estimate with a certain degree of accuracy the performance indexes to be optimized is preferred. In this paper we propose the use of a metric, named the *Mapping Coefficient*, $MC$, which we found being highly correlated with the average communication latency profile exhibited by a NoC for any traffic pattern, routing and mapping function.

#### 4.1.1 Network Modeling

The $MC$ is based on a simple metric, the *equivalent distance* between each pair of nodes. A *table of equivalent distances* can be obtained by computing the equivalent distance between each pair of nodes in the network. The equivalent

[1] In this paper with the term *system configuration* we intend the set of a mapping function and a routing function.

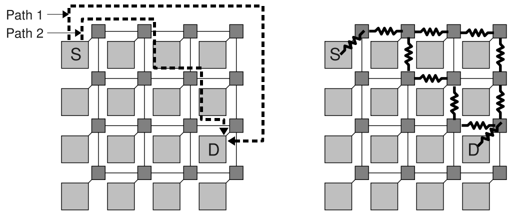



Figure 3: The delay of communication from node $S$ to node $D$ through paths *Path 1* and *Path 2* (left) is correlated with the electric equivalent resistance from $S$ to $D$ (right).

distance for a pair of nodes is computed taking into account all the paths between them supplied by the routing algorithm. It is computed by using the same rules as for electrical circuits to compute the equivalent resistance between two nodes, applying Kirchoff's laws. Let us suppose that the communication between two nodes $S$ and $D$ can be performed using two routing paths, namely *Path 1* and *Path 2*, as shown in Fig. 3. The average communication delay is highly correlated with the electric equivalent resistance from $S$ to $D$ [Tornero et al., 2008b]. That is, network nodes are treated as circuit nodes whereas links as resistances whose value is inversely proportional to the link capacity. The idea behind this model is to take into account the bandwidth added by the different possible paths but also the latency added by each link in these paths. Thus, given a communication graph $CG$, a topology graph $TG$, a mapping function $M$, and a routing function $R$, the mapping coefficient is defined as:

$$MC = \sum_{c=(t_s,t_d)\in C} B(c) \times d(M(t_s), M(t_d)), \tag{4}$$

where, the function $d(x, y)$ returns the equivalent distance between nodes $x$ and $y$.

#### 4.1.2 Correlation Assessment

To get an idea about the correlation degree between the $MC$ and the average communication delay, we have considered a 8x8 mesh network topology with XY routing and wormhole switching [Ni and McKinley, 1993] under *uniform* traffic[2]. We have studied the correlation between each distance in the table of distances and the average latency of the messages exchanged between the

[2] We have considered different network sizes. However, for the sake of shortness, we show here the correlation results for a 8x8 2-D mesh, since this size is large enough to be representative of the NoCs sizes expected for next years. The correlation results obtained for other network sizes were very similar.

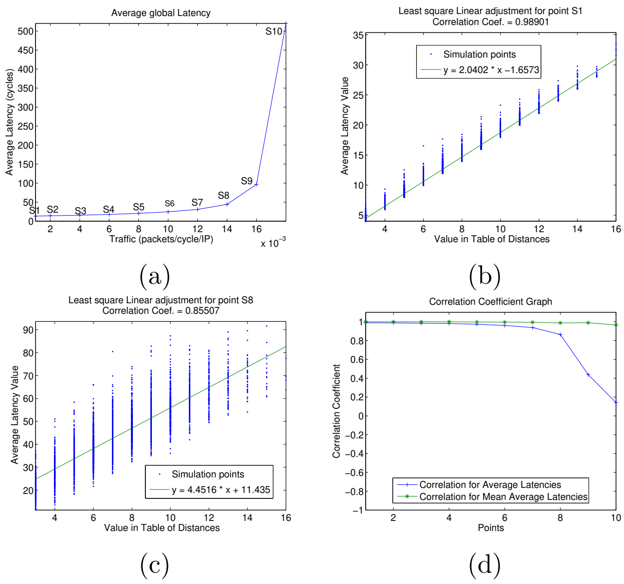



Figure 4: Correlation analysis. Average delay for different *pir* values for a 8x8 mesh based NoC under *uniform* traffic (a). Least square linear adjustment for simulation point $S1$ (b) and $S8$ (c). Correlation of average latencies values and mean values of average latencies (d).

corresponding pair of nodes. Fig. 4(a) shows the simulation results obtained using the cycle-accurate NoC simulator Noxim [Fazzino et al., 2010]. The x-axis reports the packet injection rate (*pir*) that is the rate at which each IP injects packets into the network. The y-axis reports the average communication delay for each value of the *pir*. Each point in Fig. 4 was computed as the average value of fifty different simulations which was enough to guarantee that the 95 percent confidence intervals are mostly within 2 percent of the means. From these measurements, we have computed the correlation between the table of distances and network latency.

In order to establish the correlation between the table of distances and the communication delay values obtained by means of simulation, we have first computed the least square linear adjustment for each point in Fig. 4(a). The results for the points $S1$ and $S8$ are shown in Fig. 4(b) and Fig. 4(c) respectively. As can be observed the correlation indices for $S1$ and $S8$ are 99% and 86% respectively. Such values indicate that the correlation between the proposed metric (the inter-node distance) and the performance values obtained by simulation is very high. It should be pointed out that, the precision of the proposed metric decreases as soon as the saturation region is approached (points $S9$ and $S10$). For instance, the correlation index for $S10$ is about 15%. However, for our purposes, we are not interested in estimating the average delay of a given communication. In fact, what we are interested in is the global average delay which is the mean value of average latencies between all the communications. Fig. 4(d) shows for each point $S1$ to $S10$ the correlation index for both the average communication latency and the mean average communication latency. As can be observed the latter is close to 100% for each *pir* value. An extensive analysis of the correlation between the table of distances and average communication delay figures obtained by means of simulation is presented in [Tornero et al., 2008b].

### 4.2 Robustness Index

After the mapping coefficient $MC$, the second objective to be optimized is the *Robustness Index*, $RI$, as stated in Equation (2). The $RI$ synthesizes the fault tolerant properties of a generic routing algorithm. It is based on the extension of the concept of path diversity [Dally and Towles, 2004]. It should be pointed out that the $RI$ does not directly reflect a fault tolerance measure whereas it is indicative of the effectiveness of the underlying routing algorithm in allowing the delivery of packets in presence of faults. For a given communication $c \in C$, a topology graph $TG$, a mapping function $M$, and a routing function $R$, we define the robustness index for communication $c$, $RI(c)$, as the average number of routing paths available for communication $c$ if a link belonging to the set of links used by communication $c$ is faulty. Formally,

$$RI(c) = \frac{1}{|L(c)|} \sum_{l \in L(c)} |\mathcal{P}(c) \setminus \mathcal{P}(c, l)|, \tag{5}$$

where $\mathcal{P}(c)$ is the set of paths provided by $R$ for communication $c$, $\mathcal{P}(c, l)$ is the set of paths provided by $R$ for communication $c$ that do use link $l$, and $L(c)$ is the set of links belonging to paths in $\mathcal{P}(c)$.

As an example, Fig. 5 shows the routing paths provided by two different routing functions for a communication from an IP mapped on node $s$ to an IP mapped on node $d$. If we consider the routing function in Fig. 5(a), the presence of a faulty link in the upper path does not compromise the communication from $s$ to $d$, since the lower path will be fault free (the two paths are disjoint each other). This does not happen when the routing function in Fig. 5(b) is used. Since the alternative paths share links $l'$ and $l''$, a fault in either $l'$ or $l''$ makes it not possible the communication from $s$ to $d$. Thus, case (a) is more robust than case (b). Such situation is reflected by the robustness index. Computing

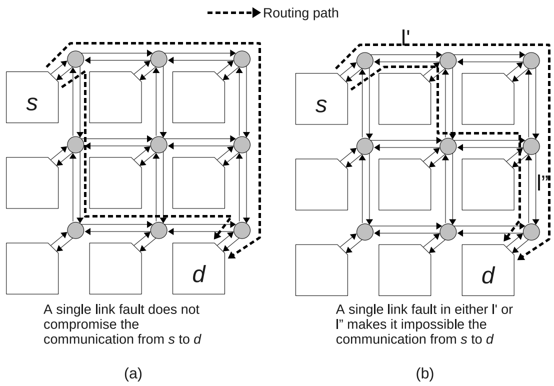



Figure 5: Routing paths provided by two different routing functions for the communicating pair $(s, d)$.

the robustness index for the above cases we have:

$$RI^{(a)}(s \to d) = \frac{1+1+1+1+1+1+1+1}{8} = 1,$$
$$RI^{(b)}(s \to d) = \frac{0+1+1+1+1+0}{6} = 0.67.$$

As $RI^{(a)} > RI^{(b)}$ we conclude that case (a) is more robust than case (b) for communication from $s$ to $d$.

Since there is a robustness index for each communication, we have to aggregate them to obtain a global robustness index which characterizes the network. We aggregated the $RI$s by means of a weighted sum. For a communication $c$, the weight of $RI(c)$ is the degree of adaptivity [Glass and Ni, 1994] of $c$. The degree of adaptivity of a communication $c$ is the ratio of the number of allowed minimal paths to the total number of minimal paths between the source node and the destination node associated to $c$. Thus, given a communication graph $CG$, a topology graph $TG$, a mapping function $M$, and a routing function $R$, the robustness index is defined as:

$$RI = \sum_{c \in C} \alpha(c) RI(c), \tag{6}$$

where $\alpha(c)$ indicates the degree of adaptivity of communication $c$.

## 5 Multi-Objective Design Space Exploration Strategy

The proposed design space exploration (DSE) uses a Multi-objective Evolutionary Algorithm (MOEA) as basic kernel of the optimization loop. The DSE flow

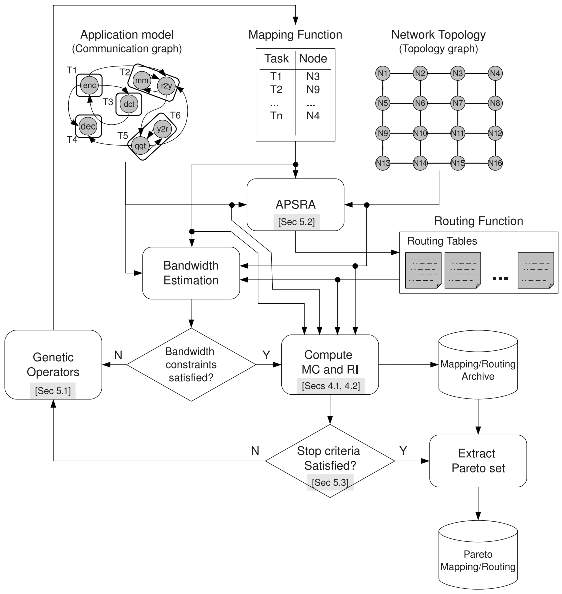



**Figure 6:** Design space exploration flow.

is sketched in Fig. 6. The inputs of the flow are the communication graph, $CG$, the topology graph, $TG$, and an initial mapping function, $M$. APSRA methodology [Palesi et al., 2009] is used to generate an application specific routing function customized for $CG$, $TG$ and $M$ (*cf.* Subsection 5.2). A bandwidth estimation model is used to compute the aggregated bandwidth on each network link. The model assumes that the communication bandwidth of each source-destination pair nodes is uniformly distributed over the routing paths for that communication. If the aggregated bandwidth on each link does not exceed the link capacity, the bandwidth constraints are satisfied and both the mapping coefficient ($MC$), and the robustness index ($RI$), are computed. The current mapping along with the routing tables and the values of $MC$ and $RI$ are collected in an archive. Then, the mapping function is "modified" (*cf.* Subsection 5.1) and the above steps are reiterated until a stop criteria is satisfied (*cf.* Subsection 5.3). Finally, the Pareto set is extracted from the archive.

In the rest of the section the different steps which form the DSE flow are discussed in details.

### 5.1 Genetic Operators

In this subsection we focus on the implementation of the block *Change Mapping* in Fig. 6. This step of the DSE flow specifies the actions taken to "modify" the

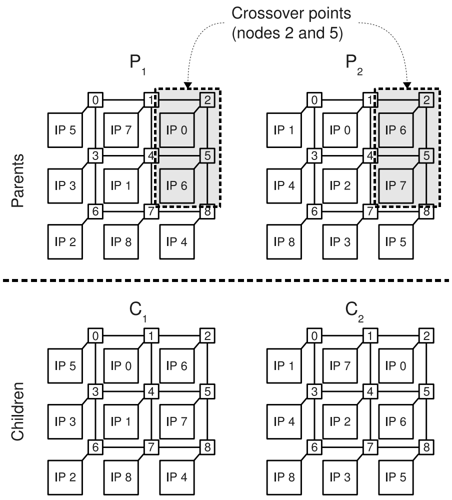



**Figure 7:** Crossover operator.

mapping function. Such modification of the mapping function is carried out by means of the genetic operators, namely crossover and mutation, and a problem specific operator.

### 5.1.1 Crossover

The *crossover* operator is discussed by means of an illustrative example shown in Fig. 7. Let $P_1$ and $P_2$ be two mappings (parents) of the mapping space as shown in the top side of the figure. Two random crossover points, $xo_1$ and $xo_2$, are computed and the subnetworks with opposite corners $xo_1$ and $xo_2$ in $P_1$ and $P_2$ are swapped, generating children $C_1$ and $C_2$. For instance, in Fig. 7, $xo_1 = 2$ and $xo_2 = 5$. $C_1$ is a copy of $P_1$ in which the IPs mapped on nodes 2 and 5 are the same of that mapped on nodes 2 and 5 of $P_2$. Before mapping IP 6 on node 2 of $C_1$, IP 0 is re-mapped on node 5 (*i.e.*, IP 0 and IP 6 are swapped). Similarly, before mapping IP 7 on node 5 of $C_1$, IP 0 is re-mapped on node 1 (*i.e.*, IP 0 and IP 7 are swapped). In the same way, $C_2$ is a copy of $P_2$ in which the IPs mapped on nodes 2 and 5 are the same of that mapped on nodes 2 and 5 of $P_1$. Once again, before mapping IP 0 on node 2 of $C_2$, IP 6 is re-mapped on node 1. Similarly, before mapping IP 6 on node 5 of $C_2$, IP 7 is re-mapped on node 1.

The pseudo-code of the crossover operator is shown in Fig. 8. The function `Subnet`($xo_1$, $xo_2$) returns the set of nodes of the subnetwork whose opposite corners are $xo_1$ and $xo_2$. The function `Map`($N$,$ip$,$n$) maps IP $ip$ on node $n$ of network $N$. The pseudo-code of the `Map` function is shown in Fig. 9. The method `getNode`($ip$) of $N$ returns the node where the IP $ip$ is mapped.

```
Crossover(in: P1, P2, xo1, xo2; out: C1, C2)
{
  C1 ← P1
  C2 ← P2
  for n ∈ Subnet(xo1, xo2)
  {
     Map(C1, P2[n], n)
     Map(C2, P1[n], n)
  }
}
```

**Figure 8:** Pseudo-code of the crossover operator.

```
Map(in/out: N; in: ip, n)
{
   n_tmp ← N.getNode(ip)
   N[n_tmp] ← N[n]
   N[n] ← ip
}
```

**Figure 9:** Pseudo-code of function `Map`.

#### 5.1.2 Mutation

The *mutation* operator is discussed by means of an illustrative example shown in Fig. 10. The mapping of an IP onto a node is mutated with a given user defined mutation probability. A node is mutated by mapping a different IP on it. In the example, nodes 3 and 8 are selected for mutation. Node 3 and node 8 are mutated by mapping IP 0 and IP 2 on them, respectively. When IP 0 is mapped on node 3, IP 3 is re-mapped on node 2 (*i.e.*, IP 0 and IP 3 are swapped). Similarly, when IP 2 is mapped on node 8, IP 4 is re-mapped on node 6 (*i.e.*, IP 4 and IP 2 are swapped).

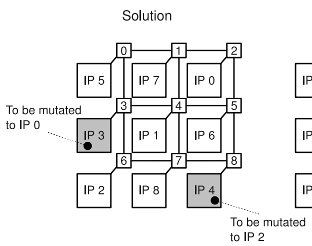



**Figure 10:** Mutation operator.

```
Mutation(in: N; out: mN)
{
  mN ← N
  for n ∈ mN
  {
     if MutateGene()
     {
        ip ← getRandomIP()
        Map(mN, ip, n)
     }
  }
}
```

**Figure 11:** Pseudo-code of the mutation operator.

The pseudo-code of the mutation operator is shown in Fig. 11. The function `MutateGene()`, on the basis of an user defined mutation probability, returns a boolean value indicating whether the current gene must be mutated or not. The function `getRandomIP()` returns a randomly selected IP.

#### 5.1.3 Problem Specific Operator

In addition to the crossover and mutation operators, we define a problem specific operator (PSO) applied to the offspring with a user defined probability. The main goal of the PSO is to modify the mapping function in such a way as to improve the optimization indices $MC$ and $RI$. It works in two phases as follows.

The rationale behind the first phase of the PSO is to reduce the communication cost of the node pair which exhibits the highest communication cost. In fact, the communication cost is related to the overall performance of the network in terms of communication delay [Murali and De Micheli, 2004]. To do this, the communication $c = (t_s, t_d)$ which maximizes the communication cost $B(c) \times d(M(t_s), M(t_d))$ is selected. In case of multiple communications with the same highest communication cost, a random selection is made. Then, the mapping function is updated in such a way that the distance between $M(t_s)$ and $M(t_d)$ is reduced. A pictorial example of this procedure is shown in Fig. 12(a). The IP $M(t_s)$ approaches to IP $M(t_d)$ by one hop following one of its admissible routing paths (randomly chosen).

The basic idea of the second phase is to improve the degree of adaptivity of the communicating pair of nodes with the lowest adaptivity. In order to achieve this goal, the communication $c = (t_s, t_d)$ with the lowest adaptivity $\alpha(c)$ is selected. In case of multiple communications with the same lowest adaptivity, a random selection is made. Then, the mapping function is updated in such a way that the minimum distance between IP $M(t_s)$ and IP $M(t_d)$ does not change

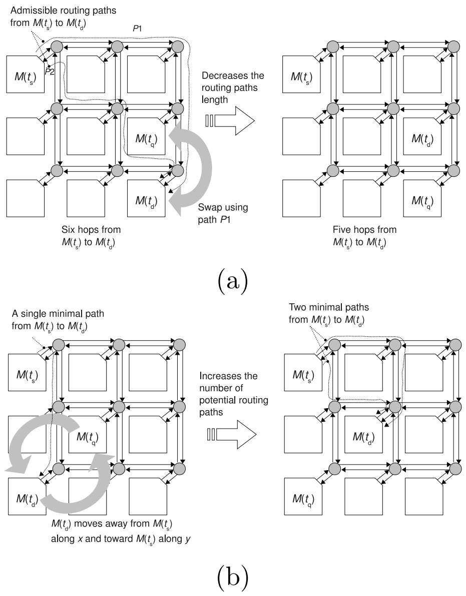



Figure 12: Problem specific operator. (a) Potential decrease of $MC$. (b) Potential increase of $RI$.

but the number of total minimal paths from $M(t_s)$ to $M(t_d)$ increases. For a mesh topology, it is simply obtained by moving one of the two IPs away from the other IP along the dimension with the minimum distance and toward the other IP along the dimension with the maximum distance. A pictorial example of this procedure is shown in Fig. 12(b). The IP $M(t_d)$ moves one hop away from IP $M(t_s)$ along dimension $x$, and one hop toward to IP $M(t_s)$ along dimension $y$.

### 5.2 Generating Routing Paths

Many deadlock free routing algorithms have been proposed in literature in the context of multiprocessing systems (works [Linder and Hardem 1991, Glass and Ni, 1994, Chien and Kim, 1995, Upadhyay et al., 1997, Chiu, 2000] represent just a short list). Whereas some of them have been applied as is in the NoC domain (*e.g.*, Odd-Even has been used in [Hu and Marculescu, 2004]), other have inspired the design of new routing algorithms targeted for NoC architectures (*e.g.*, the turn-prohibition algorithm in [Hansson et al., 2007] has been built on the turn model [Glass and Ni, 1994]). In the general context of multiprocessing systems, an important issue in the design of a routing algorithm is to conjugate deadlock freedom with high adaptiveness. A highly adaptive routing algorithm has the potential of providing high performance (low latency, low packet drop and high throughput), fault tolerance and uniform utilization of network resources.

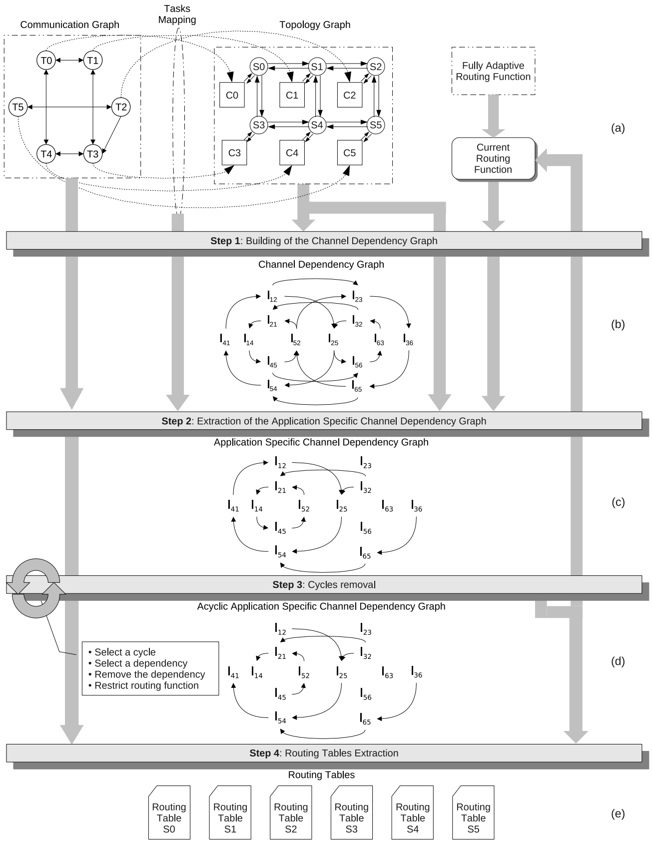



**Figure 13:** Application specific routing algorithm (APSRA) design flow.

Traditionally, in the general purpose domain, a routing algorithm is designed based solely on the network topology. As there is no information about the application that will be mapped on the system, the routing algorithm must conservatively guarantee any two network nodes to communicate. On the contrary, in the embedded system domain, the designer generally has an in-depth knowledge of the application (or the set of applications) that will be mapped on the system. In this work we exploit the knowledge of the application modeled by means of a communication graph to design a routing algorithm which is tailored for that specific application and for a given mapping. For this purpose we used the Application Specific Routing Algorithm (APSRA) design methodology proposed in [Palesi et al., 2009] which is briefly introduced in this subsection.

An overview of the APSRA design methodology is depicted in Fig. 13. After the task mapping phase of the NoC design flow, we have a complete knowledge about the pairs of cores which communicate and other pairs which never communicate. This information is captured by means of a *communication task graph* ($CG$). The $CG$ is a direct graph where each vertex represents a computational module in the application (*i.e.*, a task) and each directed arc between two tasks characterizes either data or control dependencies. The communication graph represents the first input of the APSRA methodology. The second input of the methodology is the *topology graph* ($TG$). The $TG$ is a direct graph where each vertex represents a network node (either a core or a switch), and each directed arc between two nodes represents an unidirectional channel.

For the sake of example here we consider a simple mesh-based $3 \times 2$ topology. We also consider a simple task graph with only six vertices and a mapping function which maps task $T_i$ onto core $C_i$, $i = 0, \ldots, 5$.

The methodology starts assuming a fully adaptive routing function. Based on it, the channel dependency graph ($CDG$) [Dally and Seitz, 1987] is built. In the second step, the *Application Specific Channel Dependency Graph* ($ASCDG$) is extracted from $CDG$. The $ASCDG$ is a sub-graph of $CDG$ in which only a subset of direct dependencies are retained which fall on the routing paths connecting the communicating node pairs.

For instance, looking at Fig. 13(b), the direct dependency from channel $l_{1,2}$ to channel $l_{2,3}$ is not an application specific dependency. In fact, such a dependency is generated by routing paths connecting nodes $S1$ and $S3$, nodes $S1$ and $S6$, and nodes $S4$ and $S3$. As these nodes do not communicate (as can be seen from the $CG$) such a dependency does not exist. As the $ASCDG$ is a sub-graph of the $CDG$ there are more chances that it will have less number of cycles.

If the $ASCDG$ is not acyclic, in the third step, all the cycles are iteratively broken. To break a cycle, at least one of the dependencies which forms the cycle has to be removed. This step consists of two phases. In the first phase the dependency to be removed is selected. In [Palesi et al., 2009] a heuristic was presented to select the dependency in such a way that the adverse impact on adaptivity is minimized. In the second phase, the dependency is removed simply by restricting the routing function. For example, referring to the $ASCDG$ in Fig. 13(d), the application specific channel dependency $l_{4,1} \rightarrow l_{1,2}$ is due to the communication $T4 \rightarrow T2$. Such communication can be realized by both paths $C4 \rightarrow S4 \rightarrow S5 \rightarrow S2 \rightarrow C2$ and $C4 \rightarrow S4 \rightarrow S1 \rightarrow S2 \rightarrow C2$. If the routing function is restricted in such a way as the latter path is prohibited, the application specific channel dependency $l_{4,1} \rightarrow l_{1,2}$ does not exist any longer. In a similar way it is possible to break the second cycle, removing, for instance, the dependency $l_{1,4} \rightarrow l_{4,5}$ due to communication $T1 \rightarrow T5$.

Finally, in the fourth step, the routing tables are populated by using the information provided by the $CG$ and the routing function.

### 5.3 Convergence Criterion

In the proposed DSE strategy, the longer the evolution of the algorithm, the closer the approximation is to the Pareto-optimal set. For how many generations is it therefore necessary to iterate the GA? We have used a convergence-based stop criteria that stops the process of evolution when no appreciable improvements in the species have been observed for a few generations.

The convergence criterion we propose uses the function of coverage [Zitzler and Thiele, 1999] between two sets to establish when the GA has reached convergence. Let $P'$ and $P''$ be two Pareto-fronts. The coverage function is defined as follows:

$$f_{cov}(P', P'') = \frac{|\{p'' \in P'';\ \exists\ p' \in P' : p' \preceq p''\}|}{|P''|}, \tag{7}$$

where $p' \preceq p''$ indicates the coverage of $p'$ by $p''$, that is each component (*i.e.*, objective) of $p'$ is less than or equal to the corresponding component of $p''$:

$$p' \preceq p'' \Leftrightarrow p'_i \leq p''_i\ \forall\ i \in \{1, 2, \ldots, n\},$$

where $n$ is the number of objectives (in our case $n = 2$, that is, $MC$ and $RI$). If $f_{cov}(P', P'') = 1$ all the points in $P''$ are dominated by or equal to those in $P'$. The reverse situation, $f_{cov}(P', P'') = 0$, is one in which none of the points in $P''$ is covered by $P'$.

If $P_i$ is the Pareto-front at generation $i$ and $G \geq 0$ we define the following convergence index:

$$q(i, G) = f_{cov}(P_{i+G}, P_i) - f_{cov}(P_i, P_{i+G}).$$

As the evolutionary process improves the solutions found $f_{cov}(P_{i+G}, P_i) \geq f_{cov}(P_i, P_{i+G}) \Rightarrow q(i, G) \geq 0$, *i.e.*, the fraction of points in $P_i$ that are dominated by or equal to points in $P_{i+G}$ will be greater than or equal to the fraction of points in $P_{i+G}$ dominated by or equal to the points in $P_i$. We could therefore envisage an observation every $G$ generations to evaluate $q(g, G)$ and stop iteration when it takes a value of 0 or at any rate is lower than a user-defined threshold $T$.

In practice, the algorithm may converge prematurely if no improvement has been found in the non-dominated set in an interval of $G$ generations. This can be solved by defining a stop criterion such as to confirm convergence if and only if the condition $q(g, G) \leq T$ holds for a further $O$ observations. To be more specific, if $\overline{g}$ is the first generation for which $q(\overline{g}, G) \leq T$, the stop criterion is taken as being met if and only if

$$q(\overline{g}, G) \leq T, q(\overline{g} + G, G) \leq T, \ldots, q(\overline{g} + OG, G) \leq T. \tag{8}$$

A qualitative example of the proposed convergence-based stop criteria is shown in Fig. 14. The graph on the left side of the figure shows a set of Pareto-front at different generations (from generation 1 to generation $n$). On the right

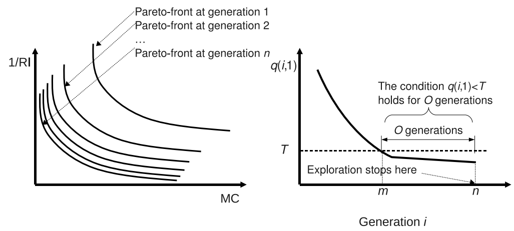



Figure 14: Qualitative example of the proposed convergence-based stop criteria.

side of the figure is shown a qualitative profile of the convergence index $q(i, G)$ for $G = 1$. As can be observed, $q$ decreases with the number of generations. At generation $m$ the value of the convergence index becomes smaller than a user-defined threshold $T$. If it remains smaller than $T$ for the next $O$ generations (as shown in the example) the convergence is assumed and the exploration algorithm is concluded.

## 6 Experiments and Results

In this section the proposed DSE strategy is assessed and compared with other DSE strategies. We first present the simulation setup and the traffic scenarios used in the experiments. Then, we perform a static performance analysis in which we evaluate statically computable performance indices under different traffic conditions and fault probabilities. Finally, we perform a detailed performance analysis using a simulator and average delay *vs. pir* curves are discussed.

### 6.1 Simulation Setup and Traffic Scenarios

We have evaluated the performance of the proposed approach on both synthetic an real traffic scenarios. Considering a $N \times N$ mesh, the following types of synthetic traffics are used. In *uniform* traffic, a randomly generated communication graph is used. Precisely, for a given source task, the destination task is selected randomly. The number of vertices (tasks) of the communication graph is equal to the number of nodes in the network whereas the average number of edges (communications) per node is 3 (*i.e.*, on average, a task communicates with three tasks). In *hot-spot* traffic, a randomly generated communication graph is used but some tasks are designated as *hot spot tasks*, which receive hot spot traffic in addition to regular uniform traffic [Boppana and Chalasani, 1993].

With regard to the real traffic scenario, we consider the traffic patterns generated by the execution of a a generic MultiMedia System (MMS) which includes an H.263 video encoder, an H.263 video decoder, an MP3 audio encoder and

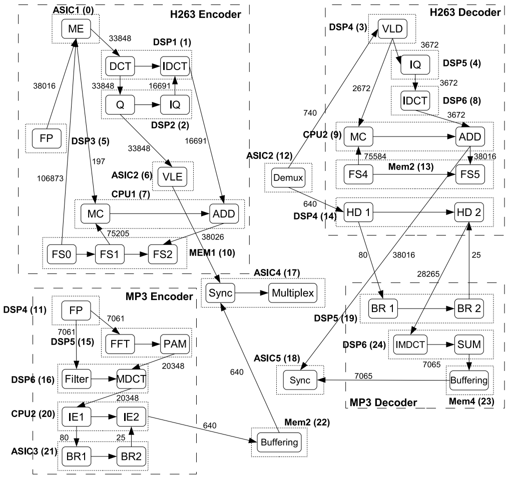



**Figure 15:** Communication graph of the MultiMedia System (MMS).

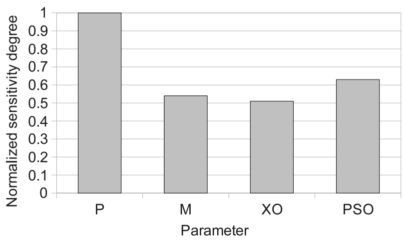



**Figure 16:** Normalized sensitivity degrees of SPEA2's parameters.

an MP3 audio decoder [Hu and Marculescu, 2005]. The communication graph for this application is shown in Fig. 15. The application is partitioned into 40 distinct tasks, which then were assigned and scheduled onto 25 selected IPs.

### 6.2 Algorithm Parameters Tuning Phase

The genetic algorithm implementing the DSE loop is based on SPEA2 [Zitzler et al., 2001]. The experimental analysis has been carried out using the following parameters. The convergence threshold $T$, the generation step $G$, and the observation period $O$ (see Sec. 5.3) have been set to 0.02, 5, and 5 respectively. The rest of the parameters (population size, crossover probability, mutation probability, and PSO probability) have been set after having carried out an extended tuning phase based on a sensitivity analysis [Montgomery, 2000] as follows.

Let $P, XO, M, PSO$ be the population size (we assume internal population size equal to external population size), the crossover probability, the mutation probability, and the PSO probability, respectively. The first phase of the sensitivity analysis sorts the parameters in descending order based on their sensitivity degree as follows. $XO$, $M$, and $PSO$ are fixed to a reference value $XO_0$, $M_0$, and $PSO_0$ respectively[3]. The, $P$ is made to vary in the set $\{10, 20, 30, 40, 50\}$. For each value of $P$, the design space exploration flow shown in Fig. 6 is performed and the minimum and maximum values of the objective functions (*i.e.*, the robustness index, $RI$, and the mapping coefficient, $MC$) are collected. Let $RI_m$ ($RI_M$) and $MC_m$ ($MC_M$) be the minimum (maximum) value of $RI$ and $MC$ respectively. The sensitivity degree of parameter $P$ is $\sigma_P = (RI_M - RI_m) \times (MC_M - MC_m)$. The same procedure is repeated for the rest of the parameters to obtain the sensitivity degrees $\sigma_{XO}, \sigma_M, \sigma_{PSO}$. Fig. 16 shows the normalized sensitivity degrees for *uniform* traffic scenario (similar trend has been observed for the rest of traffic scenarios considered in this paper). The most sensitive parameter is $P$, followed by $PSO$, $M$, and $XO$. Thus, in the second phase of the sensitivity analysis, the parameters are tuned following the above sequence. First, $P$ is made to vary and the other parameters are fixed to their reference value. Let $P_{best}$ the value of $P$ which determines the best value for $RI$ and $MC$. Then, $P$ is fixed to $P_{best}$, and the second most sensitive parameter ($PSO$) is made to vary while the remaining parameters ($M$ and $XO$) are fixed to their reference value. Let $PSO_{best}$ the value of $PSO$ which determines the best value for $RI$ and $MC$. And so on for the rest of the parameters.

We found $P_{best} = 20$, $XO_{best} = 0.2$, $M_{best} = 0.01$, $PSO_{best} = 0.1$. Although these values have been computed considering *uniform* traffic scenarios, it has been observed that the same values still continue to represent a good choice also for the other traffic scenarios. This makes it reasonable to assume that the parameters tuning phase only needs to be performed once.

### 6.3 Static Performance Analysis

The first experiment is aimed to show the trade-off between the mapping coefficient and the robustness index under different traffic scenarios. Fig. 17 shows the Pareto-front obtained by the proposed multi-objective exploration strategy (MOEARMAP) for a $8 \times 8$ mesh topology under different traffic scenarios. The same graph also shows the solution found by the approach in [Tornero et al., 2008b] (MCBMAP-XY), and by a variant of [Tornero et al., 2008b] (MCBMAP-AS) in which an application specific routing function is generated starting from the mapping found by MCBMAP-XY. The x-axes shows the normalized value of the mapping coefficient, whereas y-axes shows the normalized values of the inverse of the robustness index. That is, small values mean better performance

[3] In our experiments we considered as reference value for a generic parameter the mean value of its range of variation.

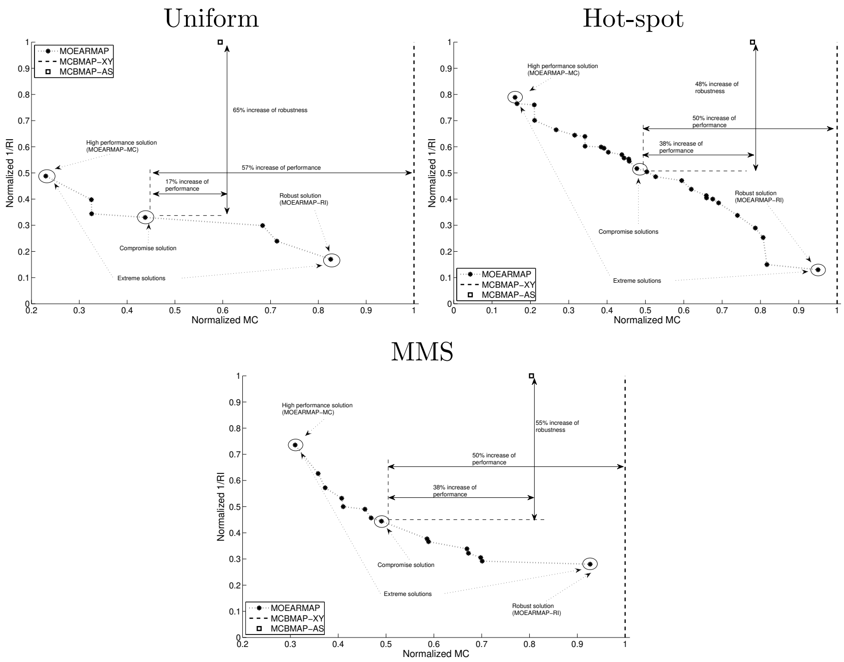



Figure 17: Pareto fronts obtained by MOEARMAP and solutions found by MCBMAP-XY and MCBMAP-AS for *uniform*, *hot-spot* and MMS traffic.

and higher robustness. Please note that, differently from MOEARMAP which returns a Pareto set of solutions, both MCBMAP-XY and MCBMAP-AS returns a single solution. In particular, for MCBMAP-XY only the $MC$ value is considered as the $RI$ cannot be computed because it implements a deterministic routing.

As it can be observed, in all the experiments the solutions found by MOEARMAP dominate those found by the other approaches. In terms of mapping coefficient, the best solution found by the proposed approach outperforms that found by MCBMAP-XY and MCBMAP-AS by 77% and 40% respectively for *uniform* traffic, by 82% and 63% respectively for *hot-spot* traffic, and by 69% and 49% respectively for MMS traffic. With regard to the robustness index, the proposed approach outperforms MCBMAP-AS by 80%, 90% and 70% for *uniform*, *hot-spot*, and MMS traffic, respectively. It is worth mention that the solution space for the *hot-spot* traffic is not convex. This is an additional point in favor of multi-objective design space exploration techniques. In fact, the use of mono-objective optimization techniques based on the optimization of an aggregation function would not be able to discover non-convex regions of the Pareto front [Coello, 1999].

Table 1: Percentage improvement in terms of performance and robustness of the compromise solution found by MOEARMAP as compared with the solution found by MCBMAP-XY and MCBMAP-AS.

| Traffic scenario | Percentage improvement Performance *vs.* MCBMAP-XY | Performance *vs.* MCBMAP-AS | Robustness *vs.* MCBMAP-AS |
|---|---|---|---|
| Uniform | 57% | 17% | 65% |
| Hot-spot | 50% | 38% | 48% |
| MMS | 50% | 38% | 55% |
| Average | 52% | 31% | 56% |

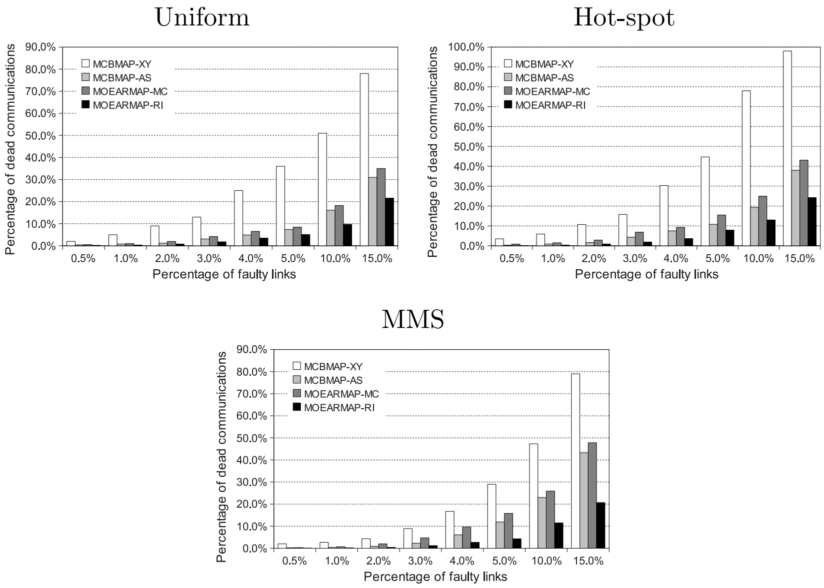



Figure 18: Percentage of dead communications for different percentage of faulty links.

Table 1 summarizes the results by comparing the solution found by MCB-MAP-XY and MCBMAP-AS with the compromise solution (*i.e.*, the solution in the middle of the Pareto-front) found by MOEARMAP. On average, the compromise solution found by MOEARMAP outperforms by 52% the solution found by MCBMAP-XY and by 31% and 56% that found by MCBMAP-AS in terms of performance and robustness respectively.

Fig. 18 shows the number of *dead communications* for different percentages of link faults. That is, a number of network links are randomly marked as faulty and the percentage of communications which do not have any available routing paths (dead communications) is computed. We compare the solution found by MCBMAP-XY and MCBMAP-AS with the extreme two solutions of the Pareto front: that which minimizes the mapping coefficient (MOEARMAP-MC), and that which maximizes the robustness index (MOEARMAP-RI). As expected, the solution which minimizes the $RI$ outperforms the other ones. On average, for any faulty link percentage the number of dead communications for MOEARMAP-RI is less than half of that of MCBMAP-AS and less than a quarter of that of MCBMAP-XY.

### 6.4 Dynamic Performance Analysis

The performance evaluation has been carried out using Noxim [Fazzino et al., 2010], a cycle-accurate NoC simulator. We have used wormhole switching with a packet size randomly distributed between 2 and 16 flits, and we have simulated routers with input buffers size of 4 flits[4]. We have used the source packet injection rate ($pir$) as load parameter. Poisson packet injection distribution is used for *uniform* and *hot-spot* traffic whereas self-similar packet injection distribution is used for MMS scenario (self-similar traffic has been observed in the bursty traffic between on-chip modules in typical multimedia applications [Varatkar and Marculescu, 2002]). For each load value, latency values are averaged over 100,000 clock cycles after a warm-up session of 10,000 clock cycles. The 95 percent confidence intervals are mostly within 2 percent of the means.

Fig. 19 shows average delay for different packet injection rate ($pir$) factors under *uniform*, *hot-spot*, and MMS traffic scenarios. That is, starting from the communication graph of the application, we compute the $pir$ of any communication $c$ as:

$$pir(c) = \frac{communication\ bandwidth\ of\ c}{packet\ size \times flit\ size \times clock\ frequency}.$$

Thus, a point in the graph at a given $pir$ factor $p$ is computed simulating the network using a $pir$ value of $p \times pir(c)$ for a communication $c$. For all traffic scenarios and low $pir$s MCBMAP-XY performs slightly better than the other. For higher values of $pir$, adaptivity plays a significant role and the performance improvement is evident. Overall, the best solution in terms of $MC$ found by the proposed approach outperforms the others. On the other side, the best solution in terms of $RI$ found by the proposed approach, exhibits delay characteristics close to MCBMAP-AS.

Fig. 20 shows the percent improvement in saturating $pir$[5] and percent improvement in average delay assuming MCBMAP-XY as baseline. Average delay

[4] These are the most common NoC parameters.
[5] A network is said to start saturating when increase in applied load does not result in linear increase in throughput [Pande et al., 2005].

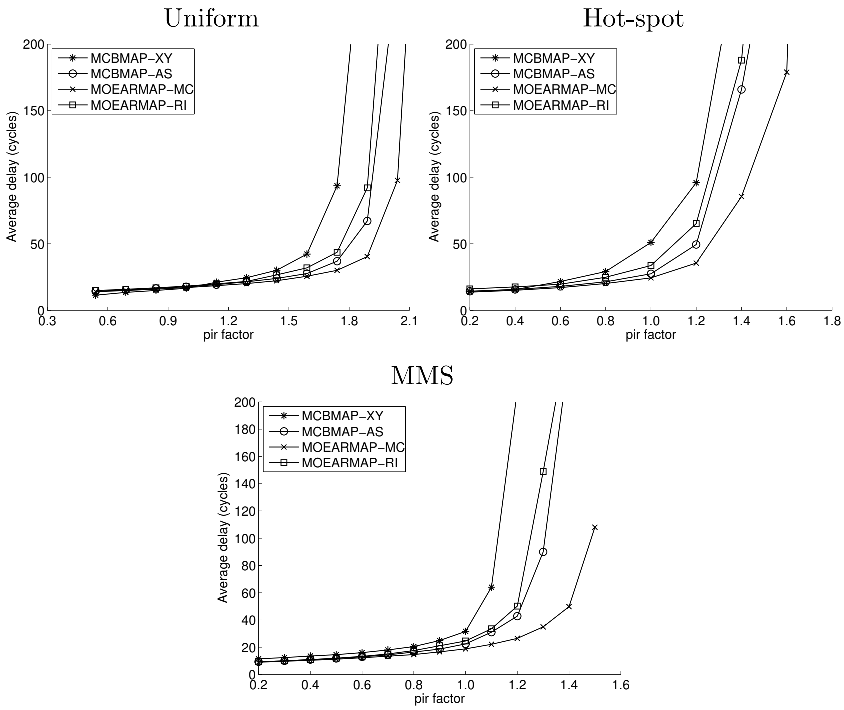
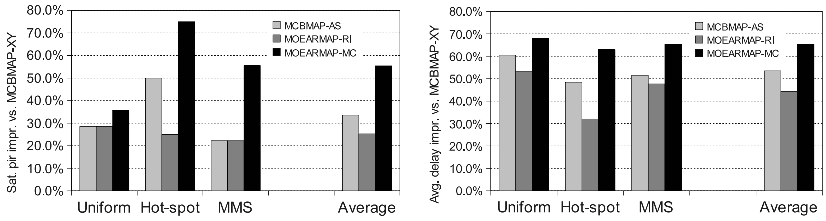



**Figure 19:** Delay variation for different packet injection rate factors.

has been measured at a packet injection rate where none of the algorithms are saturated. On average, as compared to MCBMAP-XY [Tornero et al., 2008b], MOEARMAP-MC improves saturation point and average delay by 55% and 65%, respectively. Also the solution which maximizes the robustness index, MOEARMAP-RI, exhibits important improvements over MCBMAP-XY. On average, MOEARMAP-RI is 25% and 44% better than MCBMAP-XY in terms of saturation point and average delay respectively.

### 6.5 Energy Optimization

The multi-objective nature of the proposed DSE technique make it simple its extension to perform energy optimization. We used the analytical energy estimation model proposed in [Hu and Marculescu, 2005] to compute the energy consumption of the communication sub-system. Based on this, MCBMAP-XY and MCBMAP-AS have been updated by replacing the optimization function with the energy model. MOEARMAP has been updated with and additional objective, namely energy consumption.




(a) (b)

Figure 20: Percent improvement in saturation *pir* (a) and average delay (b) assuming MCBMAP-XY as baseline.

Table 2: Percentage improvement in terms of energy consumption of the compromise solution found by MOEARMAP as compared with the solution found by MCBMAP-XY and MCBMAP-AS.

| Traffic scenario | Percentage energy reduction *vs.* MCBMAP-XY | MCBMAP-AS |
|---|---|---|
| Uniform | 38% | 33% |
| Hot-spot | 47% | 42% |
| MMS | 45% | 39% |
| Average | 43% | 38% |

The mappings found by the three approaches are simulated on a modified version of Noxim which is back-annotated with power data obtained by a gate-level power analysis of the main components of the NoC, namely router and network interface. They have been designed in Verilog HDL described at the RTL level, synthesized with Synopsys Design Compiler and mapped onto an UMC 65 nm technology library. The power dissipated by the network links has been computed using the model presented in [Palesi et al., 2011].

Overall, the following parameters are used. The NoC is clocked at 700 MHz. The baseline NI (*i.e.*, without the encoding/decoding logic) dissipates 5.3 mW. The average power dissipated by the wormhole-based router is 5.7 mW. An inter-routers wire has a total capacity of 592 fF/mm in a 65 nm UMC technology in which about 80% is due to crosstalk. We assume 2 mm 32-bits links and packet size of 16 bytes (8 flits).

Table 2 reports the percentage energy reduction of the compromise solution found by MOEARMAP as compared to the solutions found by MCBMAP-XY and MCBMAP-AS respectivelly. As can be observed, up to 43% of energy consumption can be saved without any degradation in terms of performance. For instance, Fig. 21 shows the Pareto points found by MOEARMAP for *uniform* traffic scenario. The graph also shows the solutions found by MCBMAP-

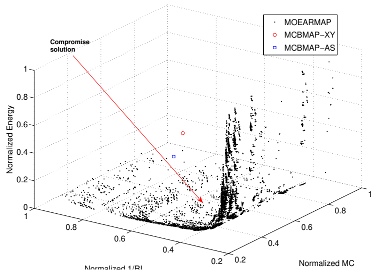



Figure 21: Pareto solutions found by MOEARMAP and solutions found by MCBMAP-XY and MCBMAP-AS whose cost function is energy consumption.

XY and MCBMAP-AS. It can be noticed how most of the solutions found by MOEARMAP dominate those found by MCBMAP-XY and MCBMAP-AS.

### 6.6 A More Complex Case Study

To assess the scalability of the proposed approach, let us consider a more complex case study in which five different applications, which share the same memory banks, are considered.

- MMS: A generic MultiMedia System which includes a H.263 video encoder, a H.263 video decoder, a MP3 audio encoder and a MP3 audio decoder [Hu and Marculescu, 2005].
- MIMO-OFDM: A MIMO-OFDM receiver in which, to support the maximum data rate of WWiSE (World-Wide Spectrum Efficiency) proposal for the next-generation wireless LAN systems, some of the IPs have been parallelized to multiple IPs [Yoon et al., 2006].
- PIP and MWD: A Picture-In-Picture application and a Multi-Window Display application [Jaspers and With, 1999, Tol and Jaspers, 2002].

The mapping found by MOEARMAP is shown in Fig. 22. The compromise solution found by MOEARMAP dominates those found by MCBMAP-XY and MCBMAP-AS. Table 3 summarizes the results. The performance statistics (average communication delay) are presented separetly for each of the five applications. As can be observed, for all the applications, on average, the communication delay for the mapping found by MOEARMAP is 29% and 14% smaller than that exhibited by MCBMAP-XY and MCBMAP-AS respectively. Even in terms of total energy consumption saving, MOEARMAP outperforms the other approaches. The experiment also gives some insight about the scalability of the proposed approach which maintains its advantages over the other approaches as network size and system complexity increase.

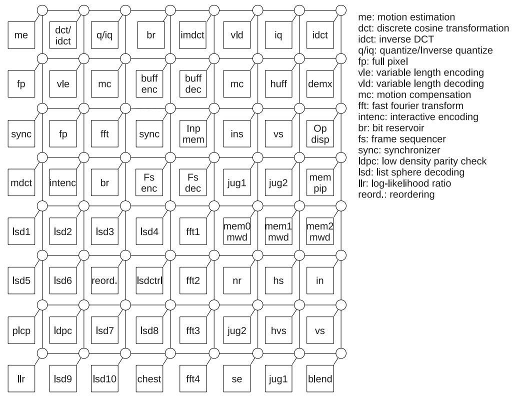



Figure 22: Mapping found by MOEARMAP for a system composed by a multimedia sub-system, a MIMO-OFDM receiver, a PIP and a MWD module.

Table 3: Comparison between the compromise solution found by MOEARMAP (Fig. 22) and the solutions found by MCBMAP-XY and MCBMAP-AS.

| Decrease of | MOEARMAP *vs.* | |
|---|---|---|
| | MCBMAP-XY | MCBMAP-AS |
| Dead communications with 5% faulty links | 28% | 9% |
| Average delay (MMS-enc) | 40% | 23% |
| Average delay (MMS-dec) | 43% | 21% |
| Average delay (MIMO-OFDM) | 48% | 27% |
| Average delay (PIP) | 8% | 0% |
| Average delay (MWD) | 5% | 1% |
| Total energy consumption | 39% | 28% |

## 7 Conclusions

In this paper, we have proposed a new design strategy that is able to determine an approximation of the Pareto-set of NoC configurations. In this way, the design flow can consider multidimensional aspects and it can provide a multi-objective optimization of the NoC. The optimization loop is implemented by means of a Multi-Objective Evolutionary Algorithm (MOEA) which provides a good compromise between the quality of the solutions found and the exploration time. The effectiveness of the MOEA has been improved by incorporating problem specific genetic operators. Concretely, we have considered the routing algorithm and the topological mapping (finding the best placement of cores within the NoC topology) as the multidimensional interacting parameters.

The results obtained from experiments on both synthetic and real traffic patterns show that the proposed strategy provides better solutions than the sequential optimization of the same parameters. Overall, these results show that the high degree of interaction among these design parameters require a global system optimization, rather than a separate study.

## References

[ITRS, 2007] ITRS 2007 edition. International Technology Roadmap for Semiconductors, 2007.

[Ascia et al., 2007] G. Ascia, V. Catania, A. G. D. Nuovo, M. Palesi, and D. Patti. Efficient design space exploration for application specific systems-on-a-chip. *Journal of Systems Architure*, 53(10):733–750, 2007.

[Ascia et al., 2004] G. Ascia, V. Catania, and M. Palesi. Multi-objective mapping for mesh-based NoC architectures. In *Second IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*, pages 182–187, Stockholm, Sweden, Sept. 8–10 2004.

[Benini and De Micheli, 2002] L. Benini and G. D. Micheli. Networks on chips: a new SoC paradigm. *IEEE Computer*, 35(1):70–78, Jan. 2002.

[Bjerregaard and Mahadevan, 2006] T. Bjerregaard and S. Mahadevan. A survey of research and practices of network-on-chip. *ACM Computing Surveys*, 38(1):1–51, 2006.

[Boppana and Chalasani, 1993] R. V. Boppana and S. Chalasani. A comparison of adaptive wormhole routing algorithms. In *International Symposium on Computer Architecture*, pages 351–360, San Diego, California, USA, May 1993.

[Chang and Pedram, 2002] J.-M. Chang and M. Pedram. Codex-dp: Co-design of communicating systems using dynamic programming. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 19(7):732–744, July 2002.

[Chien and Kim, 1995] A. A. Chien and J. H. Kim. Planar-adaptive routing: Low-cost adaptive networks for multiprocessors. *Journal of the ACM*, 42(1):91–123, Jan. 1995.

[Chiu, 2000] G.-M. Chiu. The odd-even turn model for adaptive routing. *IEEE Transactions on Parallel Distributed Systems*, 11(7):729–738, 2000.

[Chou and Marculescu, 2008] C.-L. Chou and R. Marculescu. Contention-aware application mapping for network-on-chip communication architectures. In *Proc. 26th International Conference on Computer Design, ICCD 2008, Lake Tahoe, CA, USA*, pages 164–169, 12–15 October 2008.

[Coello, 1999] C. A. C. Coello. An updated survey of evolutionary multiobjective optimization techniques: State of the art and future trends. In *Special Session on Multiobjective Optimization at the Congress on Evolutionary Computation*, volume 1, pages 3–13, July 1999.

[Dally and Seitz, 1987] W. J. Dally and C. Seitz. Deadlock-free message routing in multiprocessor interconnection networks. *IEEE Transactions on Computers*, C(36):547–553, 1987.

[Dally and Towles, 2001] W. J. Dally and B. Towles. Route packets, not wires: On-chip interconnection networks. In *ACM/IEEE Design Automation Conference*, pages 684–689, Las Vegas, Nevada, USA, 2001.

[Dally and Towles, 2004] W. J. Dally and B. Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, San Francisco, CA, 2004.

[Fazzino et al., 2010] F. Fazzino, M. Palesi, and D. Patti. Noxim: Network-on-Chip simulator. `http://noxim.sourceforge.net`.

[Glass and Ni, 1994] C. J. Glass and L. M. Ni. The turn model for adaptive routing. *Journal of the Association for Computing Machinery*, 41(5):874–902, Sept. 1994.

[Hansson et al., 2007] A. Hansson, K. Goossens, and A. Rădulescu. A unified approach to mapping and routing on a network-on-chip for both best-effort and guaranteed service traffic. *VLSI Design*, 2007, 2007.

[Hu and Marculescu, 2003] J. Hu and R. Marculescu. Energy-aware mapping for tile-based NoC architectures under performance constraints. In *Asia & South Pacific Design Automation Conference*, pages 233–239, Jan. 2003.

[Hu and Marculescu, 2004] J. Hu and R. Marculescu. DyAD - smart routing for networks-on-chip. In *ACM/IEEE Design Automation Conference*, pages 260–263, San Diego, CA, USA, June 7–11 2004.

[Hu and Marculescu, 2005] J. Hu and R. Marculescu. Energy- and performance-aware mapping for regular NoC architectures. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 24(4):551–562, Apr. 2005.

[Jantsch and Tenhunen, 2003] A. Jantsch and H. Tenhunen, editors. *Networks on Chip*, chapter 1. Kluwer Academic Publishers, 2003.

[Jaspers and With, 1999] E. G. T. Jaspers and P. H. N. de With. Chip-set for video display of multimedia information. *IEEE Transactions on Consumer Electronics*, 45(3):706–715, Aug. 1999.

[Jena and Sharma, 2008] R. Jena and G. Sharma. Application mapping of mesh based noc using multi-objective genetic algorithm. *International Journal of Computers and Applications*, 30(1):17–22, January 2008.

[Kumar et al., 2002] S. Kumar, A. Jantsch, J.-P. Soininen, M. Forsell, M. Millberg, J. Oberg, K. Tiensyrja, and A. Hemani. A network on chip architecture and design methodology. In *IEEE Computer Society Annual Symposium on VLSI*, page 117, 2002.

[Lei and Kumar, 2003] T. Lei and S. Kumar. A two-step genetic algorithm for mapping task graphs to a network on chip architecture. In *Euromicro Symposium on Digital Systems Design*, Sept. 1–6 2003.

[Linder and Hardem 1991] D. Linder and J. Harden. An adaptive and fault-tolerant wormhole routing strategy for k-ary n-cubes. *IEEE Transactions on Computers*, 40(1):2–12, Jan. 1991.

[Marculescu et al., 2009] R. Marculescu, U. Y. Ogras, L.-S. Peh, N. E. Jerger, and Y. Hoskote. Outstanding research problems in NoC design: System, microarchitecture, and circuit perspectives. *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, 28(1):3–21, Jan. 2009.

[Mattson et al., 1996] T. G. Mattson, D. Scott, and S. R. Wheat. A TeraFLOP in 1996: The ASCI TeraFLOP supercomputer. In *International Parallel Processing Symposium*, pages 84–93, Apr. 15–19 1986.

[De Micheli and Benini, 2006] G. D. Micheli and L. Benini. *Networks on Chips: Technology and Tools.* Morgan Kaufmann, 2006.

[Montgomery, 2000] D. C. Montgomery. *Design and Analysis of Experiments.* Wiley Text Books, 5th edition, 2000.

[Murali and De Micheli, 2004] S. Murali and G. D. Micheli. Bandwidth-constrained mapping of cores onto NoC architectures. In *Design, Automation, and Test in Europe*, pages 896–901. IEEE Computer Society, Feb. 16–20 2004.

[Ni and McKinley, 1993] L. M. Ni and P. K. McKinley. A survey of wormhole routing techniques in direct networks. *IEEE Computer*, 26:62–76, Feb. 1993.

[Ogras et al., 2005] U. Y. Ogras, J. Hu, and R. Marculescu. Key research problems in noc design: A holistic perspective. In *International Conference on Hardware-Software Codesign and System Synthesis*, pages 69–74, Sept. 2005.

[Palesi et al., 2011] M. Palesi, G. Ascia, F. Fazzino, and V. Catania. Data encoding schemes in networks on chip. *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, 30(5), May 2011.

[Palesi et al., 2009] M. Palesi, R. Holsmark, S. Kumar, and V. Catania. Application specific routing algorithms for networks on chip. *IEEE Transactions on Parallel and Distributed Systems*, 20(3):316–330, Mar. 2009.

[Pande et al., 2005] P. P. Pande, C. Grecu, M. Jones, A. Ivanov, and R. Saleh. Performance evaluation and design trade-offs for network-on-chip interconnect architectures. *IEEE Transactions on Computers*, 54(8):1025–1040, Aug. 2005.

[Tornero et al., 2008] R. Tornero, J. M. Orduña, A. Mejía, J. Flich, and J. Duato. Cart: Communication-aware routing technique for application-specific nocs. In *Proc. 11th EUROMICRO Conference on Digital System Design Architectures, Methods and Tools*, pages 26–31, 3–5 Sept. 2008.

[Tornero et al., 2008b] R. Tornero, J. M. Orduña, M. Palesi, and J. Duato. A communication-aware topological mapping technique for NoCs. In *14th International Conference on Parallel and Distributed Computing*, volume 5168 of *Lecture Notes in Computer Science*, pages 910–919, Las Palmas de Gran Canaria, Spain, 26–29Aug. 2008.

[Tornero et al., 2009] R. Tornero, V. Sterrantino, M. Palesi, and J. M. Orduña. A multi-objective strategy for concurrent mapping and routing in networks on chip. In *International Workshop on Nature Inspired Distributed Computing held in conjunction with The 23th IEEE/ACM International Parallel and Distributed Processing*, page 122, 2009.

[Upadhyay et al., 1997] J. Upadhyay, V. Varavithya, and P. Mohapatra. A traffic-balanced adaptive wormhole routing scheme for two-dimensional meshes. *IEEE Transactions on Computers*, 46(2):190–197, Feb. 1997.

[Tol and Jaspers, 2002] E. B. van der Tol and E. G. Jaspers. Mapping of MPEG-4 decoding on a flexible architecture platform. *Media Processors*, 4674:362–375, 2002.

[Vangal et al., 2008] S. R. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, A. Singh, T. Jacob, S. Jain, V. Erraguntla, C. Roberts, Y. Hoskote, N. Borkar, and S. Borkar. An 80-tile sub-100-W TeraFLOPS processor in 65-nm CMOS. *IEEE Journal of Solid-State Circuits*, 43(1):29–41, Jan. 2008.

[Varatkar and Marculescu, 2002] G. Varatkar and R. Marculescu. Traffic analysis for on-chip networks design of multimedia applications. In *ACM/IEEE Design Automation Conference*, pages 510–517, June 2002.

[Yoon et al., 2006] S.-R. Yoon, J. Lee, and S.-C. Park. Case study: Noc based next-generation wlan receiver design in transaction level. In *International Conference on Advanced Communication Technology*, pages 1125–1128, 2006.

[Zitzler et al., 2001] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the performance of the strength pareto evolutionary algorithm. Technical Report TIK-Report 103, Computer Engineering and Communication Networks Lab, Swiss Federal Institute of Technology (ETH) Zurich, Gloriastrasse 35, CH-8092, May 2001.

[Zitzler and Thiele, 1999] E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 4(3):257–271, Nov. 1999.