Journal of Universal Computer Science, vol. 12, no. 4 (2006), 370-394
submitted: 31/10/05, accepted: 15/3/06, appeared: 28/4/06 © *J.UCS*

A Multi-objective Genetic Approach to Mapping Problem on Network-on-Chip

Giuseppe Ascia, Vincenzo Catania, and Maurizio Palesi
Dipartimento di Ingegneria Informatica e delle Telecomunicazioni
Università di Catania, Italy
{gascia,vcatania,mpalesi}@diit.unict.it

Abstract: Advances in technology now make it possible to integrate hundreds of cores (e.g. general or special purpose processors, embedded memories, application specific components, mixed-signal I/O cores) in a single silicon die. The large number of resources that have to communicate makes the use of interconnection systems based on shared buses inefficient. One way to solve the problem of on-chip communications is to use a Network-on-Chip (NoC)-based communication infrastructure. Such interconnection systems offer new degrees of freedom, exploration of which may reveal significant optimization possibilities: the possibility of arranging the computing and storage resources in an NoC, for example, has a great impact on various performance indexes. The paper addresses the problem of topological mapping of intellectual properties (IPs) on the tiles of a mesh-based NoC architecture. The aim is to obtain the Pareto mappings that maximize performance and minimize power dissipation. We propose a heuristic technique based on evolutionary computing to obtain an optimal approximation of the Pareto-optimal front in an efficient and accurate way. At the same time, two of the most widely-known approaches to mapping in mesh-based NoC architectures are extended in order to explore the mapping space in a multi-criteria mode. The approaches are then evaluated and compared, in terms of both accuracy and efficiency, on a platform based on an event-driven trace-based simulator which makes it possible to take account of important dynamic effects that have a great impact on mapping. The evaluation performed on both synthesized traffic and real applications (an MPEG-4 codec) confirms the efficiency, accuracy and scalability of the proposed approach.

Key Words: System-on-chip, Network-on-chip, Mapping, Multi-objective optimization, Evolutionary algorithms, Simulation
Category: B.4.3, D.4.7, G.1.6, I.6

1 Introduction

The possibility of integrating a number of Intellectual Property (IP) blocks in the same die has caused the design paradigm to shift from device-centric to interconnection-centric. Design optimisation strategies in deep-submicron, where the cost of an interconnection is much higher (for area, power consumption, speed and cost) than a cost of logic cells or transistors results in a new design paradigm. New chip and system level synchronisation strategies for complex circuits are required in order to obtain high system performance and standardised way to integrate complex IP to designs. Thus this will define on-chip and off-chip communication architectures, a necessity for platform thinking and efficient IP usage.

Today, the on-chip interconnection system represents one of the major elements which has to be optimized in designing a complex digital system. The International

Technology Roadmap for Semiconductors [ITRS] foresees it will represents the limiting factor for performance and power consumption in next generation Systems-on-a-Chip (SoCs). The continuous reduction in the time-to-market required by the telecommunications, multimedia and consumer electronics market makes full-custom design of an interconnection system inappropriate and has led to the definition of design methodologies focusing on design reuse. This is confirmed by the great standardization effort made by the VSI Alliance [VSI] and the development, by the major EDA and Semiconductor companies, of on-chip interconnection systems that are easy to integrate and scale [CoreConnect, AMBA Spec., PalmChip, WishBone, BackPlane]. Although, however, they are good solutions for current SoCs integrating fewer than 5 processors and rarely more than 10 master buses, their use in next-generation systems, which are likely to integrate hundreds of modules, seems hardly feasible.

The limiting factor is mainly the topological organization of the interconnection between the various units, which will substantially remain bus-based. From the point of view of a module, the behavior of a bus is unpredictable: as it is a shared resource, it can be used by other interconnected modules. Another problem is connected with the physics of deep submicron technology. Long, global wires and buses are undesirable due to their unpredictable performance, high power consumption and problems of reliability due to noise. As regards performance, the continuous reduction in gate delays and increase in wiring delays will cause significant synchronization problems. In 50 nm technology, the projected chip die edge will be around 22 mm, with a clock frequency of 10 GHz. An optimistic estimate of the propagation delay for a signal crossing a chip diagonally ranges between 6 and 10 clock cycles [Sylvester and Keutzer. 2000]. In addition, in a deep submicron regime, the increase in sensitivity to sources of on-chip noise (e.g. crosstalk, power supply noise, electromagnetic interference, intersymbol interference, etc.) makes communications unreliable [Bertozzi et al. 2003].

At any rate, Moore's law will remain valid for the next 10 years and single processors will not be able to use all the transistors on a chip. Synchronous regions will occupy an increasingly lower fraction of a chip [Sylvester and Keutzer. 1998] giving rise to locally synchronous, globally asynchronous solutions [Hemani et al. 1999]. Applications will be modelled as a set of communicating tasks with different characteristics (e.g. control-dominated, data-dominated) and origins (reused from previous projects or acquired from third parties), which will make implementations extremely heterogeneous.

A type of architecture which lays emphasis on modularity and is intrinsically oriented towards supporting such heterogeneous implementations is represented by Network-on-Chip (NoC) architectures [Dally and Towels 2001]. These architectures loosen the bottleneck due to delays in signal propagation in deep-submicron technologies and provide a natural solution to the problem of core reuse by standardizing on-chip communications. In this paper we will focus on mesh-based NoC architectures, in which resources communicate with each other via a mesh of switches that route and buffer messages. A resource is generally any core: a processor, a memory, an FPGA, a spe-

cific hardware block or any other IP compatible with the NoC interface specifications. A two-dimensional mesh interconnection topology is simplest from a layout perspective and the local interconnections between resources and switches are independent of the size of the network. Moreover, routing in a two-dimensional mesh is easy, resulting in potentially small switches, high bandwidth, short clock cycles, and overall scalability [Jantsch and Tenhunen 2003].

One of the most onerous tasks in this context is the topological mapping of the resources on the mesh in such a way as to optimize certain performance indexes (e.g. power, performance). Mapping is, in fact, a problem of quadratic assignment that is known to be NP-hard [Garey and Johnson 1979a]. The search space of the problem increases factorially with the system size. It is therefore of strategic importance to define methods to search for a mapping that will optimize the desired performance indexes. In addition, these strategies have to a multi-criteria exploration of the space of possible architectural mapping alternatives. The objectives to be optimized are, in fact, frequently multiple rather than single, and are almost always in contrast with each other. There is therefore no single solution to the problem of exploration (i.e. a single mapping) but a set of equivalent (i.e. not dominated) possible architectural alternatives, featuring a different trade-off between the values of the objectives to be optimized (Pareto-set).

In this paper we present a multi-objective exploration approach for the mapping space of a mesh-based NoC architecture. The approach, based on evolutionary computing techniques, is an efficient and accurate way to obtain the Pareto mappings that optimize performance and power consumption. In addition, two of the most widely known approaches to topological mapping of IPs in a mesh-based NoC architecture [Hu and Marculescu 2003, Murali and De Micheli 2004] have been extended to achieve multi-criteria optimization and have been compared with the approach proposed here. In contrast with the approaches in the existing literature which use static analysis to evaluate a mapping, here we use an event-driven trace-based simulator which makes it possible to take account of important dynamic effects that have a great impact on performance indexes to be optimise. To the best of our knowledge this work is the first attempt to attack the topological mapping problem for NoC architectures from a multi-objective point of view taking care of model important dynamic effect such as contention for outgoing links, backpressure effects, influence of buffer size, packet size, etc.

The rest of the paper is organized as follows. Section 2 summarizes some of the most important contributions in the field of topological mapping of IPs/cores in mesh-based NoC architectures. Section 3 presents the simulation and evaluation framework used and highlight the necessity of using evaluation tools that do not neglect important dynamic effects which have a great impact on the performance indexes to be optimized. Section 4 our approach for exploration of the mapping space is presented. In the same section we discuss the multiobjective extension of two other algorithms proposed in literature we compare to. Experimental results are reported in Section 5. Finally, Section 6 summarizes our contribution and outlines some directions for future work.

2 Previous Work and Our Contribution

A large amount of research in the area of system-level synthesis of application specific architectures can be found in the literature. Hardware and software partitioning is followed by mapping an application task graph on a set of pre-designed cores and application-specific hardware blocks [Gajski et al. 1994]. Increasingly greater interest is being shown in communications as well as the optimization of processing and storage. The interconnection system, in fact, has a fundamental impact on the most significant performance indexes and is the object of great standardization efforts by the Virtual Socket Interface Alliance (VSIA) [VSI] and the interconnection solutions proposed by several microelectronics corporations (IBM CoreConnect [CoreConnect], ST Microelectronics STBus [STBus]), core vendors (ARM AMBA [AMBA Spec.]), interconnect IP vendors (Palmchip CoreFrame [PalmChip], Silicore WishBone [WishBone], Sonics SiliconBackPlane [BackPlane]), and others.

To address the problems linked with the long, highly capacitive wires typical of bus-based architectures, one proposal is the use of communication architectures with regular topologies to interconnect a number of elements [Dally and Towels 2001]. These complex networks create significant challenges in determining the best way to map system communications to an underlying network topology.

The problem of mapping in mesh-based NoC architectures has been addressed in three previous papers. Hu and Marculescu [Hu and Marculescu 2003] present a branch and bound algorithm for mapping IPs/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. The same authors extend the approach to constructs a deadlock-free deterministic routing function such that the total communication energy is minimized [Hu and Marculescu 2005]. Murali and De Micheli [Murali and De Micheli 2004] address the problem under the bandwidth constraint with the aim of minimizing communication delay by exploiting the possibility of splitting traffic among various paths. Lei and Kumar [Lei and Kumar 2003] present an approach that uses genetic algorithms to map an application, described as a parameterized task graph, on a mesh-based NoC architecture. The algorithm finds a mapping of the vertices of the task graph on the available cores so as to minimize the execution time.

These papers do not, however, solve certain important issues. The first relates to the mapping evaluation model used, which can be defined as “static”. The exploration algorithm decides which mapping to explore without taking important dynamic effects of the system into consideration. For example, failure to model the effects of bus contention causes components which communicate with each other more frequently to be clustered, whereas it may be more effective to separate components whose traffic flows overlap in time so as to increase the degree of concurrency. In the above-mentioned works, in fact, the application to be mapped is described using task graphs, as in [Lei and Kumar 2003], or simple variations such as the the *application characterization graph* [Hu and Marculescu 2003] or the *core graph* [Murali and De Micheli 2004].

These formalisms do not, however, capture important dynamics of communication traffic. They hypothesize worst-case conditions, which leads to several mappings being discarded and thus a highly conservative exploration.

In agreement with [Pestana et al. 2004], we believe that analytical methods make too many assumptions about the network and traffic to get accurate values for real systems. This is also confirmed in [Lahiri et al. 2004]: although the application scenario refers to bus-based organizations, it is shown that approaches using only static information (such as the frequency of communication between the various nodes in a system) and neglecting dynamic characteristics (such as overlapping communications leading to contention on the bus and waiting times for synchronization) often provide mediocre solutions. In any case, the methodology proposed in [Lahiri et al. 2004] together with the evaluation framework described in [Lahiri et al. 2001] is only applicable to memoryless topologies. In these topologies, once data transfer has started it has to be completed before any other transfer can take place as the data cannot be stored inside the communication architecture. This hypothesis obviously does not apply to the communication architectures dealt with in this paper.

The second problem relates to the optimization method used. It refers in all cases to a single performance index (power in [Hu and Marculescu 2003], performance in [Murali and De Micheli 2004, Lei and Kumar 2003]). As we will see in Section 5, optimization of one performance index may lead to unacceptable values for another performance index (e.g. high performance levels but unacceptable power consumption). We therefore think that the problem of mapping can be more usefully solved in a multi-objective environment, i.e. one in which there is no single solution but a set of mapping alternatives (which we will indicate as Pareto mapping), each featuring a different tradeoff between performance indexes, from which the designer (or decision maker) will choose the most suitable.

The contribution we intend to make in this paper is to propose a multi-objective approach to solving the problem of mapping IPs/cores in mesh-based NoC architectures. The approach will use evolutionary computing techniques to explore the mapping space with the goal to optimize performace and power consumption. The mappings visited during the exploration process will be evaluated using a trace-based approach which gives an excellent combination of accuracy and efficiency features.

3 Evaluation of a Mapping and Exploration Framework

In this section we will discuss aspects connected with the evaluation of a mapping. We will present the reference communication architecture and highlight the necessity of using evaluation tools that do not neglect important dynamic effects which have a great impact on the performance indexes to be optimized. We will then present the exploration framework.

3.1 Reference Architecture

Figure 1 shows the NoC topology we will refer to. It is a two-dimensional mesh of

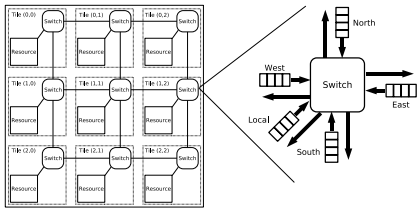


Figure 1: Structure of a 3x3 mesh-based NoC architecture

processing resources. Each processing resource is connected to the communication network by a switch. We will call the pair formed by a resource and a switch a *tile*. The term *mapping* will be used to indicate assignment of an IP/core to each tile in the NoC. Each switch in the NoC is connected to the four adjacent switches except for those at the network boundaries. Switches send data from one network interface to the other by means of packets. Such a packet consists of one or more *flow control digits* (or *flits*), were a flit is the minimal transmission unit. On each side of a switch there is an output and an input port. The input port has a finite-length FIFO buffer in which flits to be routed are queued. The use of the FIFO is regulated by back-pressure mechanism [Hu and Marculescu 2004]. Under this scheme, a flit will be held in the buffer until the downstream router has empty space in the corresponding input FIFO. Thus, the network will not drop any packet in transit. This is extremely important for NoC architectures which may not implement very advanced end-to-end protocol.

The routing algorithm features *static XY* routing in which a flit is first routed in a horizontal direction (X) and then, when it reaches the column where the destination tile is located, it is routed in a vertical direction (Y). Of course the XY routing is a *minimal* path routing algorithm and is *free* of deadlock and livelock [Glass and Ni 1998]. As a transmission scheme we use wormhole routing because of the low cost (the buffer capacity can be less than the length of a packet) and low latency (the router can start forwarding the first flit of a packet without waiting for the tail).

3.2 Simulation Model

The key to applying any design space exploration methodology is to have the tools necessary for rapid, accurate evaluation of any instance of the design space. Research into system-level performance analysis of on-chip communication comprises both approaches based on *simulation* of the whole system at different levels of abstraction [Rowson and Sangiovanni-Vincentelli 1997] and *static* approaches based on estimation

of the system performance indexes using analytical models [Gasteier and Glesner 1999]. Whereas the former guarantee high levels of accuracy at the cost of performance, the latter, although efficient from a computational point of view, are not capable of modeling significant dynamic effects, giving a rough estimate of the *communication latency*. Communication latency is one of the most important performance metrics. It has three components: *start-up latency*, *network latency* and *blocking time*. Start-up latency is the time required for the system to handle the message at the source and destination nodes and depends primarily on the design of the interface between the cores and switches. Network latency is defined as the time a message takes to transit through the network. It is calculated as the time between the instant at which the message header is put out over the network and the instant at which the tail is enters he destination node. Blocking time is defined as the time a message has to wait before it can use a communication channel currently being used for another message. Whereas the first two components are fixed for a given network and can thus be determined statically, the third depends on the resource contentions a message encounters along its path. Blocking time can thus not be statically determined as it depends on the distribution of traffic on the network and on the path a message takes. A third approach is trace-based performance-analysis techniques [Lahiri et al. 2001]. Their accuracy and performance generally come between those of the other approaches. They guarantee accurate modeling of the dynamic effects and at the same time greater efficiency than simulation-based techniques [Lahiri et al. 2001].

The system simulation strategy used in this work is event-based and simulation is performed by stimulating the network with concurrent trace files. The network architecture is defined using *Behavioral Annotated Graphs* (BAGs) [Ascia et al. 2004b] which model the cores and switches of the NoC and characterize them as regards timing and power. The simulation infrastructure used makes it possible to evaluate a number of performance indexes. From a global viewpoint, it is possible to evaluate both the total amount of energy consumed and the time needed ti handle incoming traffic (completion time). As regards energy consumption, it is possible to evaluate separately the three types of consumption due to computing (cores), communication (switches) and transmission (wires). From a local viewpoint it is possible to evaluate the performance and features of each element making up the NoC; for example, the energy consumed by each core and switch, the mean buffer occupancy for each switch, the communication bandwidth on each link, statistical information about delay and jitter, etc..

3.3 Exploration Framework

Figure 2 shows the framework for exploration of the space of possible mappings in mesh-based NoC architectures.

It comprises two macro blocks: a *NoC simulator* (to evaluate the performance indexes to be optimized for any mapping), and an *Exploration engine* (which determines the next mapping to be evaluated). The inputs to the framework are:

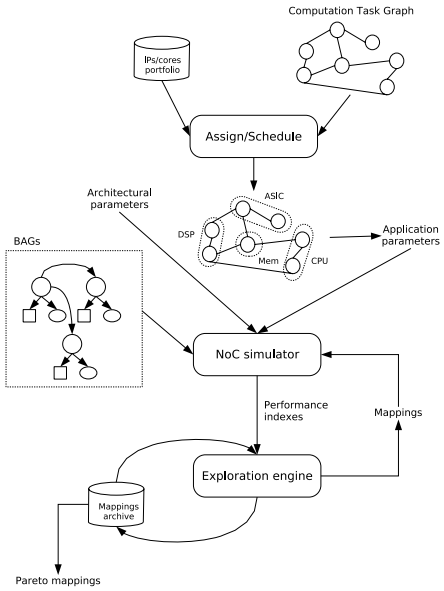


Figure 2: Framework for simulation and exploration of the mapping space

– *Architectural parameters*: for example, topology, network size, communication protocols, size of buffers in switches, priority assignment schemes, etc.

– *Application parameters*: these mainly refer to the characteristics of the communication traffic involved in the application being considered. They may relate to both the characterization of statistical models of the traffic exchanged between the various network resources, and real traces obtained by measuring the communication traffic during execution of the application. Useful application parameters to specify traffic in statistical models are: packet generation rate (packets can be generated at random or periodical intervals, or in a bursty or uniform fashion); the statistical distribution of the destination addresses (random, or polarized towards a certain group of resources (hot spot) etc.). For the real traces, they 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.

– *Set of BAGs* [Ascia et al. 2004b]: these specify the functional behavior of each element in the NoC and also contain characterization information for estimation of the timing and power consumption parameters.

The flow of operations involved in exploration generally consists of repeating two phases: evaluation of one or more mapping alternatives, and determination of the next mapping/s to be evaluated. The first phase is carried out using a NoC simulator, which

evaluates the performance indexes to be optimized. These represent the input for the second phase, which implements the exploration algorithm and produces the next mapping/s to be evaluated. The mappings evaluated are stored and can be used by the exploration algorithm to decide the next step. This iterative process is concluded when a stop criterion is met. Then the non-dominated mappings (Pareto mappings) are extracted from the mappings evaluated.

In this paper we will focus on the second phase of the framework, the one referring to the mapping space exploration algorithms.

4 Multi-Objective Exploration of the Mapping Space

The mapping problem is an instance of a constrained quadratic assignment problem which is known to be *NP-hard* [Garey and Johnson 1979b]. The search for an optimal mapping (henceforward referred to as exploration) is also complicated when the concept of optimality is not limited to a single performance index (or objective) but comprises several contrasting indexes. The traditional approach to a multi-objective optimization is to aggregate the objectives into a single one by means of a *weighting sum*. The main drawback to this approach is that it does not cover the non-convex regions of the Pareto-front and requires several instances of the optimization algorithm to be run with different weights. In this section we present: 1) an approach to multi-objective mapping space exploration that uses evolutionary algorithms as the optimization strategy; 2) multi-objective extension of an exploration algorithm based on the branch-and-bound proposed in [Hu and Marculescu 2003]; and 3) multi-objective extension of a variation of the exploration algorithm proposed in [Murali and De Micheli 2004]. Although not directly specified, optimization is constrained in all the approaches by bandwidth requirements. That is, once the bandwidth constraints for each communication flow are known a mapping will not be considered valid if even one of the constraints is not met.

4.1 Terminology and Problem Formulation

If C is the set of cores, and T the set of tiles, we will use the term *mapping* to indicate an injective and surjective function $M : C \rightarrow T$ that associates the tile $t \in T$ on which c is mapped with each $c \in C$.

Evaluating a mapping means obtaining the related performance indexes for a specific traffic scenario. If S indicates a traffic scenario, we define the *evaluation function*

$$\mathbf{V}(S,M) = (V_1(S,M), V_2(S,M), \ldots, V_n(S,M))$$

which yields the values of the n performance indexes relating to the mapping M for the traffic scenario S. In our case study, for example, the evaluation function corresponds to the simulation framework (described in [Ascia et al. 2004b]) and the performance

indexes are those the platform is capable of measuring (power, communication latency, bandwidth, throughput, etc.). Evaluation of an incomplete mapping made up of a set of cores $C' \subset C$ with a traffic scenario S is performed by evaluating the mapping on a traffic S' obtained by filtering out all communication flows in which the source or destination is a core $c \in C'$.

Given a traffic scenario S and two mappings M_1 and M_2, M_1 can be said to *dominate* M_2 (which will be indicated as $M_1 \succ M_2$) if $V_i(S,M_1) \leq V_i(S,M_2),\ i \in \{1,2,\ldots,n\}$ and there exists at least one $j \in \{1,2,\ldots,n\}$ such that $V_j(S,M_1) < V_j(S,M_2)$. The *set of Pareto mappings* is a set of mappings that do not dominate each other. The Pareto front is the image of the evaluation function for the set of Pareto mappings. If $\mathscr{M}$ is the set of all possible mappings, the Pareto-optimal set $\mathscr{P}$ is the set of Pareto mappings such that $\nexists\, M \in \mathscr{M} : M \succ M',\ M' \in \mathscr{P}$.

The aim of the approach we propose is to obtain as accurate an approximation as possible of the Pareto-optimal front by evaluating (visiting) as few mappings as possible.

4.2 GA-based Multi-Objective Exploration of the Mapping Space

The use of evolutionary algorithms (EAs) as a multi-objective optimization technique is of increasing appeal. The fields of application are numerous, including among others computer science, engineering, economics, finance, industry, physics, chemistry, and ecology. EAs have been demonstrated to be very powerful and generally applicable for solving difficult multi-objective problems [Coello et al. 2002]. Such algorithms create an interesting alternative to other approaches since they can be scaled with the problem size and can be easily run on parallel computer systems. In VLSI design, EAs have been applied to a very broad range of problems [Mazumder and Rudnick 1999]: in problems relating to layout such as partitioning [Alpert et al. 1996], placement and routing [Lienig 1997] [Valenzuela and Wang 2002]; in design problems [Schiner et al. 2001] including power estimation [Jiang et al. 1997], low-power synthesis [Bright et al. 2001], technology mapping [Kommu et al. 1993], netlist partitioning [Alpert and Kahng 1995] and in reliable chip testing through efficient test vector generation [Saab et al. 1996].

In this paper we propose the use of a heuristic technique based on EAs for multi-objective mapping space exploration. More precisely, we use the Strength Pareto Evolutionary Algorithm (SPEA) [Zitzler and Thiele 1999] which maintains an external set to preserve the nondominated solutions encountered so far besides the original population. The chromosome is a representation of the solution to the problem, which in this case is described by the mapping. Each tile in the mesh has an associated gene which encodes the identifier of the core mapped in the tile. In an $n \times m$ mesh, for example, the chromosome is formed by $n \times m$ genes. The i-th gene encodes the identifier of the core in the tile in row $\lceil i/n \rceil$ and column $i\%m$ (where the symbol % indicates the modulus operator).

The *crossover* and *mutation* genetic operators were have been suitably redefined. More specifically, a crossover between two mappings M_f and M_m generates a new mapping M_s constructed as follows. The dominant mapping between M_f and M_m is chosen. Its hot-spot core is remapped on a tile in a random position in the mesh, thus providing the new mapping M_s. Figure 3 describes the crossover operator. Where the function

```
1   Mapping XOver(Mapping M_f , Mapping M_m)
2   {
3       Mapping M_s;
4
5       if (M_f ≻ M_m)
6           M_s = M_f;
7       else
8           M_s = M_m;
9
10      Swap(M_s , HotSpot(M_s) , Random({1,2,...,N^2}));
11
12      return M_s;
13  }
```

Figure 3: Crossover operator

`Swap(`M`,`i`,`j`)` exchanges the i-th tile with the j-th tile from mapping M.

The mutation operator acts on a single mapping M to obtain the mutated mapping M' as follows. A tile T_s from mapping M is chosen at random. Indicating the core in the tile T_s as c_s and c_t as the core with which c_s communicates most frequently, c_s is remapped on a tile adjacent to T_s so as to reduce the distance between c_s and c_t by a hop, thus obtaining the mutated mapping M'. Figure 4 describes the mutation operator. The `RandomTile(`M`)` function gives a tile chosen at random from mapping M. The `MaxCommunication(`c`)` function gives the core with which c communicates most frequently. The `Row(`M`,`T`)` and `Col(`M`,`T`)` functions respectively give the row and column of the tile T in mapping M. Finally, the `North, South, East, West(`M`,`T`)` functions give the tile to the north, south, east and west of the tile T in mapping M.

The definition of suitable and more effective genetic operators has a great impact on the results of the optimization. This is not, however the aim of this paper and remains a topic for future research.

4.3 Pareto-based Branch-and-Bound Approach

In [Hu and Marculescu 2003] Hu and Marculescu present an approach using branch-and-bound as the mapping space exploration strategy. The approach is, however, a mono-objective one. In this subsection we will extend their approach in order to perform

```
1  Mapping Mutate(Mapping M)
2  {
3     Mapping M' = M;
4
5     Tile T_s = RandomTile(M');
6     Core c_s = M'^-1(T_s);
7     Core c_t = MaxCommunication(c_s);
8     Tile T_t = M'(c_t);
9
10    Tile T'_s;
11    if (Row(M', T_s) < Row(M', T_t))
12       T'_s = North(M', T_s);
13    else if (Row(M', T_s) > Row(M', T_t))
14       T'_s = South(M', T_s);
15    else if (Col(M', T_s) < Col(M', T_t))
16       T'_s = East(M', T_s);
17    else
18       T'_s = West(M', T_s);
19
20    Swap(M', T_s, T'_s);
21
22    return M';
23 }
```

Figure 4: Mutation operator

multi-objective exploration of the mapping space. We will call our approach *Pareto-based Branch-and-Bound* (*PBBB*).

Let $\{c_1, c_2, \ldots, c_{N^2}\}$ be the set of cores in the system in decreasing order with respect to the communication traffic. The core c_1 can be mapped on any of the N^2 tiles in the mesh. These N^2 mappings generate the first layer of a tree which is the starting point for the branch-and-bound algorithm. For each first-level mapping the core c_2 can be mapped on any of the N^2-1 free tiles, thus generating a second level $N^2 \times (N^2-1)$ mappings. This is the *branch* phase of the algorithm and is described in pseudo-code in Figure 5. Where the `MakeMappings(`M`,`c`)` function, given a mapping template M and a core c, yields a set of mappings obtained by mapping c on each free tile in M.

Each mapping at this level is evaluated (simulated) and then characterized according to the optimization objectives, which in our case are power and delay. The dominated mappings are discarded, while the branch and bound phases are reiterated on the survivors. This is the *bound* phase of the algorithm as described in pseudo-code in Figure 6. Where the `ExtractPareto(`$\mathscr{M}$`)` function extracts the non-dominated mappings from the set $\mathscr{M}$. To prevent the algorithm from degenerating the bound phase is followed by a further pruning phase. Let us indicate the set of mappings generated by the bound phase as $\mathscr{M}$. If $|\mathscr{M}| > T$ (where T is a user-defined threshold) $|\mathscr{M}| - T$ map-

```
1  Mappings Branch(Mappings M , Core c)
2  {
3     Mappings M' = ∅;
4
5     for (M ∈ M)
6        M' = M' ⋃ MakeMappings(M , c);
7
8     return M';
9  }
```

Figure 5: Branch phase of the branch-and-bound algorithm

```
1  Mappings Bound(Mappings M)
2  {
3     Mappings M' = ExtractPareto(M);
4
5     if (|M'| > T_pbbb)
6        Pruning(M' , T_pbbb);
7
8     return M';
9  }
```

Figure 6: Bound phase of the branch-and-bound algorithm

pings are eliminated at random from $\mathscr{M}$. The `Pruning`$(\mathscr{M}, T_{pbbb})$ function randomly eliminates mappings from a set $\mathscr{M}$ if the cardinality of this set exceeds a threshold T_{pbbb} in such a way as to make the cardinality of $\mathscr{M}$ equal to T_{pbbb}.

The branch and bound phases are reiterated until all the cores have been mapped. For example, indicating the mappings obtained in the bound phase as $M_1, M_2, \ldots, M_n$, the core c_3 will be mapped for each of them on to the N^2-2 possible tiles. The $n \times N^2-2$ mappings will be the third level of the tree. The algorithm terminates when all the cores have been mapped and the leaves of the tree will be the Pareto mappings. A pseudo-code description of *PBBB* is given in Figure 7. Where the `SortByTraffic`(C) function orders the set of cores C according to the communication traffic.

4.4 Pareto-based NMAP Approach

Murali and De Micheli in [Murali and De Micheli 2004] propose NMAP, an algorithm that maps the cores in a mesh NoC architecture with the aim of minimizing the average communication delay. In this subsection we will extend NMAP to perform a multi-objective exploration of the mapping space. Unlike [Murali and De Micheli 2004], however, we will refer to a routing XY. We will call this approach *Pareto-based NMAP* (*PBNMAP*).

```
1   Mappings PBBB(Cores C)
2   {
3       Cores Cs = SortByTraffic(C);
4       Mappings M = MakeMappings(∅, Cs,1);
5       for (c ∈ Cs \ {Cs,1}) {
6           M = Branch(M, c);
7           M = Bound(M);
8       }
9
10      return M;
11  }
```

Figure 7: Pareto-based branch-and-bound approach

The algorithm comprises two phases. In the first the cores featuring the largest amount of communication traffic are mapped onto the central tiles in the mesh (i.e. the $(N-2)\times(N-2)$ tiles with the greatest numbers of neighbours). The remaining cores are then ordered in decreasing order with respect to the communication traffic they have with the cores mapped in the previous phase. The first, c_1, is mapped onto each of the $4\times(N-1)$ remaining tiles. The $4\times(N-1)$ are evaluated and those that are dominated are discarded. If $\mathscr{M}_1$ is the set of non-dominated mappings, the algorithm is reiterated for each $M\in\mathscr{M}_1$ with c_2 and so on until the last core $c_{4(N-1)}$ has been mapped and the set of Pareto mappings $\mathscr{M}=\mathscr{M}_{4(N-1)}$ has been obtained. Figure 8 give the pseudo-code for this first phase. Where the `Map(`M,c,row,col`)` function maps core c onto the

```
1   Mappings PBNMAP_1st(Cores C)
2   {
3       Cores Cs = SortByTraffic(C)
4       Mapping M;
5       for (i ∈ {1,2,...,(N−2)×(N−2)})
6           Map(M, Cs,i, (i−1)/(N−2)+1, (i−1)%(N−2)+1);
7
8       Cores C2s = SortByC2CTraffic({Cs,(N−2)∗(N−2)+1,...,Cs,N²},
9                                    {Cs,1,...,Cs,(N−2)∗(N−2)});
10      Mappings M = ∅;
11      for (c ∈ C2s) {
12          M = MakeMappings(M, c);
13          M = ExtractPareto(M);
14      }
15
16      return M;
17  }
```

Figure 8: First phase of the Pareto-based NMAP approach

tile in row *row* and column *col* of the mapping M. The `SortByC2CTraffic`(C_a, C_b) function sorts the cores in the set C_a according to the communication traffic they have with the cores in the set C_b.

In the second phase, the mapping of cores c_i and c_j is inverted for each mapping $M \in \mathscr{M}$ and each pair (c_i, c_j), thus obtaining the new mapping M'. The algorithm proceeds with the next pair on the mapping M or M' according to whether M dominates M' or M' dominates M. If M and M' are Pareto mappings, the algorithm proceeds with the next pair on both mappings. A pseudo-code description of this phase is given in Figure 9, while Figure 10 describes the main program.

```
1   Mappings PBNMAP_2nd(Mappings M , Cores C)
2   {
3       for (i ∈ {1,2,...,N^2 − 1})
4           for (j ∈ {i+1,i+2,...,N^2}) {
5               Mappings M_n = ∅;
6               for (M ∈ M) {
7                   Mapping M' = Swap(M , i , j);
8               if (M' ≻ M)
9                       M_n = M_n ⋃ {M'};
10                  else if (M ≻ M')
11                      M_n = M_n ⋃ {M};
12                  else
13                      M_n = M_n ⋃ {M,M'};
14              }
15              M = ExtraxtPareto(M_n);
16      }
17
18      return M;
19  }
```

Figure 9: Second phase of the Pareto-based NMAP approach

5 Experiments

In this section we will describe the experiments performed and the results obtained by applying the approaches described above to two different traffic scenarios. The approaches will initially be evaluated in *synthesized* traffic scenarios, i.e. ones in which the traffic is generated statistically without particular reference to a specific application. Although in many cases synthesized traffic is quite different from real traffic, we use it to identify the optimal mapping[1] (in some cases) and compare it with the sub-optimal

[1] With "optimal mapping" we intend the mapping in which communicating cores are placed at minimum distance each from the others.

```
1  Mappings PBNMAP(Cores C)
2  {
3     Mappings M;
4
5     M = PBNMAP_1st(C);
6     M = PBNMAP_2nd(M , C);
7
8     return M;
9  }
```

Figure 10: Pareto-based NMAP approach

solutions obtained by the various approaches. The second traffic scenario analyzed, on the other hand, is generated by running real application (an MPEG-4 codec).

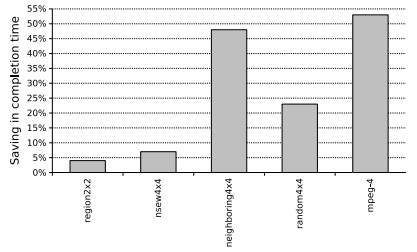


Figure 11: Saving in completion time

The optimization objectives are the total amount of energy consumed and completion time. We consider completion time to be a more representative performance index than the communication cost used in [Hu and Marculescu 2003] (evaluated as the sum of the products of the bandwidth required for each communication and the number of hops needed). In general, a mapping that optimizes communication cost does not necessarily optimize completion time. Figure 11, for example, shows for some of the benchmarks discussed in the following subsections the percent reduction in completion time with a mapping that optimizes completion time rather than communication cost. As can be seen, the saving is on average 20% , and in some cases as much as 50%.

5.1 Synthetic Traffic Scenarios

To evaluate the effectiveness of the approaches discussed in the previous section, 5 different traffic scenarios were defined, in four of which the optimal mapping is intuitive.

The following values were used for the free parameters of the exploration algorithm. For *GAMAP* we chose a population of 50 mappings, a crossover probability of 0.7 and a

mutation probability of 0.1. These values were chosen after numerous simulations and were the values that on average led to better solutions or shorter convergence times. The number of generations was set runtime by means of a stop criterion based on analysis of the convergence of the Pareto-front [Ascia et al. 2004a]. For *PBBB*, the parameter T_{pbbb} was set to 100.

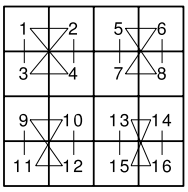
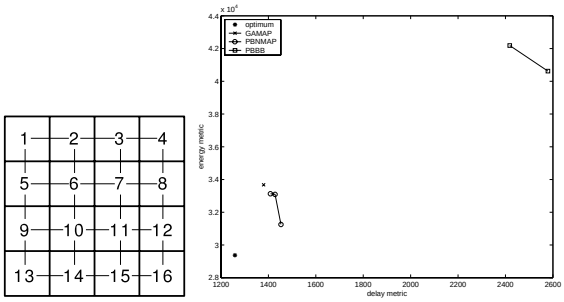


Figure 12: Optimal mapping for the *region2x2* scenario

The first scenario (*region2x2*) consists of 4 traces of concurrent communications, each of which defines the flow of communication between 4 cores. The sets of cores involved in each trace are disjoint. More specifically, the generic trace $trace_i, i = 1, \ldots, 4$ describes communication between the cores in the set $C_i = \{c_j | c_j = 4(i-1) + j, j = 1,2,3,4\}$. The traffic generated by each core in a set is defined as follows. The destination address is uniformly distributed between the cores belonging to the same set as the current core, and the quantity of traffic exchanged has a Gaussian distribution with an average of 128 bytes and a variance of 64 bytes. In this case, all the approaches obtain the optimal mapping, which comprises 4 square-shaped regions (where the number of hops between two cores is minimal) as shown in Figure 12. In this representation (which will also be used later on) a segment connecting two cores indicates a possible communication between them. Of course a generic approach does not obtain only one solution but a set of Pareto mappings, the only difference being the arrangement of the cores in a region. In this case, however, the arrangement has a marginal impact on the performance indexes considered (less than 0.2% for completion time and less than 0.5% for energy consumption).

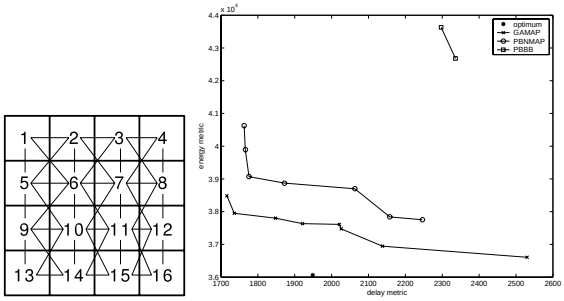
The second scenario (*nsew4x4*) was constructed in such a way that it is possible to find a mapping in which all communications can take place with a single hop. In this case none of the three approaches obtains the optimal mapping shown in Figure 13(a). The Pareto fronts they obtain are given in Figure 13(b). The only mapping obtained by *GAMAP* is 9% worse than the optimal mapping as regards completion time and 12% worse for energy consumption. Of the three mappings obtained by *PBNMAP*, the one that minimizes completion time is 11% worse than the optimal mapping and 13% worse for energy consumption. The mapping that optimizes energy consumption is 15% worse for completion time and 6% worse for energy consumption. *PBBB* obtains two mappings but the accuracy is much worse than that of the previous approaches: about



(a) (b)

Figure 13: Traffic scenario *nsew4x4*. (a) Optimal mapping. (b) Pareto fronts

50% for completion time and about 30% worse for energy consumptione. Analysis of the mappings obtained by the various approaches shows that even though the average number of hops for the mapping obtained by *GAMAP* is greater than that obtained by *PBNMAP* (1.33 as compared with 1.23), the former is better in terms of performance. That is, even though the communication paths are on average longer, the completion time is shorter due to better use of the information about the timing features of the communication flows. *PBNMAP*, in fact, neglects this information as the optimization process is only guided by information concerning bandwidth and the amount of traffic exchanged, with no reference to the overlapping of communication flows.

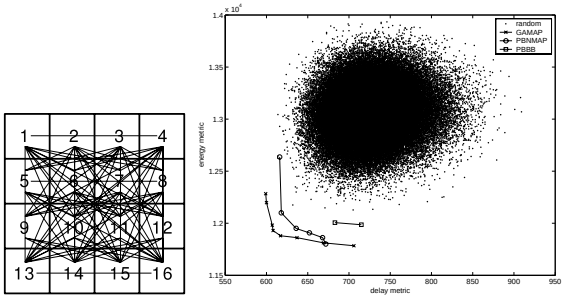


(a) (b)

Figure 14: Traffic scenario *neighboring4x4*. (a) Optimal mapping. (b) Pareto fronts

The third scenario (*neighboring4x4*), is similar to *nsew4x4* with the exception that each core communicates not only with adjacent cores but also with those in a north-east, south-east, north-west and south-east direction. The optimal mapping is shown in Figure 14(a) , while Figure 14(b) gives the Pareto fronts obtained by the three approaches.

It is interesting to note that the mapping thought to be optimal is only so for energy consumption (respectively 1.5%, 4.5% and 16% better than *GAMAP*, *PBNMAP*, and *PBBB*). From the performance point of view, *GAMAP* and *PBNMAP* obtain better mappings than the optimal one which minimizes the number of hops. The mappings they obtain offer completion times that are 12% and 9% shorter than the optimal time. Once again, the solutions obtained by *PBBB* are worse than those obtained by the other approaches. The two mappings obtained give completion times that are 15% longer than the solution which minimizes the number of hops and 25% longer than the one obtained by *GAMAP*.



(a) (b)

Figure 15: Traffic scenario *random4x4*. (a) Communication flows. (b) Pareto fronts obtained by the different approaches and evaluation of 100,000 random mappings

In the fourth and last scenario the four concurrent traffic flows describe communications between 16 cores with randomly selected origin and destination [Figure 15(a)]. As the Pareto-optimal mappings cannot be obtained in this case (it would require exhaustive evaluation of $16! \approx 10^{13}$ mappings), the results obtained by the various approaches are compared with the best ones obtained by sampling the mapping space in 100,000 randomly chosen points. Figure 15(a) shows the evaluations of the 100,000 random mappings and the Pareto fronts obtained by *GAMAP*, *PBNMAP* and *PBBB*. The solutions obtained by *GAMAP* and *PBNMAP* dominate the 100,000 random mappings, whereas those obtained by *PBBB* only give an improvement in energy consumption. In any case, the solutions obtained by *GAMAP* dominate those obtained by the other approaches.

Finally, it is necessary to evaluate the computational complexity of the various approaches. The metric used is the number of mappings that need to be evaluated to complete the exploration. This is proportional to the CPU time needed to complete the exploration, as the overhead due to management of the various algorithms is negligible as compared with the time required to evaluate a mapping (i.e. to run a simulation). Figure 16 shows the number of simulations each approach requires to complete an explo-

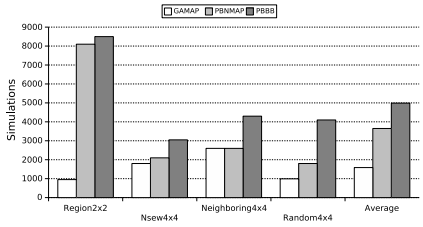
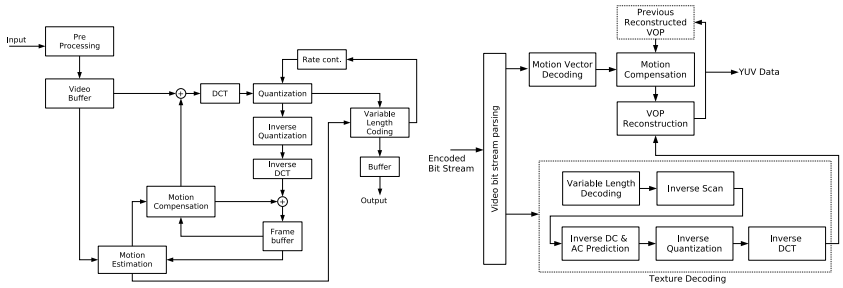


Figure 16: Number of simulations each approach requires to complete an exploration in the various traffic scenarios

ration in the various traffic scenarios. On average, *GAMAP* executes about 55% fewer simulations than *PBNMAP* and over 70% fewer than *PBBB*.

5.2 Real Traffic Scenarios

In order to evaluate the various approaches in real traffic scenarios, an MPEG-4 simple profile @ level 2 codec was used as a case study [Sikora 1997]. A general block diagram of the encoder and decoder is shown in Figure 17.



(a) (b)

Figure 17: Block diagram of the MPEG-4 codec. (a) Encoder. (b) Decoder

For the hardware/software partitioning reference was made to the MoVa architecture described in [Kim et al. 2003]. It adopts a macroblock-based pipeline with 4 stages for the encoder and 3 for the decoder. More specifically, the encoding section performs coarse motion estimation in the first stage, fine motion estimation fine and motion compensation in the second stage, discrete cosine transform and quantization in the third stage, and finally reconstruction and production of the stream in the fourth stage. In the decoding section, the first stage involves variable length decoding of each data stream; in the second stage it performs sequential inverse cosine transformation, inverse quantization and motion compensation; the third and final stage is reconstruction.

To obtain the traffic traces the C application implementing the codec [XVID] was modified with the addition of a monitor code to record the volume of incoming and outgoing traffic in the various functional blocks into which the application is partitioned. The cores implementing the codec has been characterized in terms of timing by using the clock cycle data in [Kim et al. 2003] for the execution of each operation (DCT, MC, etc.). For power characterization, we used the mean values given in the datasheets [Mentor Graphics, Philips IP]. For the interconnection system we used an approach similar to the one presented in [Hu and Marculescu 2003]. To characterize the switches, a 5x5 switch was implemented in VHDL following the architecture described in [Banerjee et al. 2004]. It was synthesized with a Synopsys Design Compiler using the Virtual Silicon $0.13\mu m$, $1.2V$ technological library and analyzed using Synopsys Design Power using different random input data streams for the inputs of the switch. The amount of power consumed by a flit for a hop switch was estimated as being $0.181nJ$. We assumed the tile size to be $2mm \times 2mm$ and that the tiles were arranged in a regular fashion on the floorplan. The load wire capacitance was set to $0.50fF$ per micron, so considering an average of 25% switching activity the amount of power consumed by a flit for a hop interconnect is $0.384nJ$.

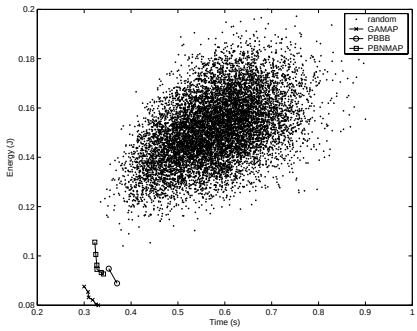
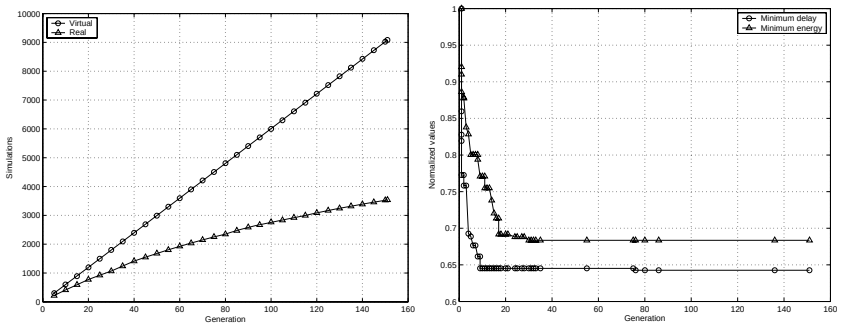


Figure 18: Evaluation of 10,000 random mappings and Pareto fronts obtained by *GAMAP*, *PBNMAP*, and *PBBB* for a 4×4 NoC and MPEG-4 codec application

Figure 18 gives the power values and traffic clearing times for 10,000 random mappings. It also shows the Pareto fronts obtained by *GAMAP*, *PBNMAP*, and *PBBB*. As can be seen, the solutions obtained by *GAMAP* dominate those obtained by the other approaches. The figure also shows the good trade-off between delay and power (respectively equal to a factor of 3 for delay and 2.5 for power).

Figure 19(a) gives the number of simulations (i.e. mappings evaluated by *GAMAP*) for varying numbers of generations. It gives the number of simulations actually performed and those virtually performed if no caching mechanism had been used. Figure 19(b) gives the normalized delay and energy values for varying numbers of gener-



(a) (b)

Figure 19: Number of (virtual and real) mappings evaluated by *GAMAP* in varying numbers of generations (a). Normalized minimum delay and power consumption values obtained by the *GAMAP* in varying numbers of generations (b)

ations. As can be seen, in both cases no mappings that determine appreciable improvements in delay and energy consumption are found after the 20th generation. At the 20th generation *GAMAP* had only performed 840 simulations as compared with 2,670 by *PBNMAP* and 7,238 by *PBBB*, thus providing an exploration time speed-up of 3.2 and 8.6 respectively.

Encoder Decoder Shared

VOM	SP	MEME	VIM
DB	REC	IQIDCT	VLD
MEC	RISC	MC	MEMD
MEF	VLC	DCTQ	ISC

Figure 20: Pareto mapping obtained by *GAMAP* for the MPEG-4 codec

Finally, Figure 20 shows a point in the Pareto set obtained by *GAMAP*. The cores specific to the encoding section are shown against a dark gray background, whereas those specific to the decoding are against a white background. The cores shared by the encoder and decoder are shown against a light gray background and have been mapped (in this case) in the centre of the NoC. In the decoding section, the cores VOM and DB are topologically separated from VLD, MEMD and ISC as there is no direct communication flow between these sets: they communicate by means of a ring represented by the core REC. In the encoding section there are also two separate parts which do not communicate directly but through the set of shared cores.

6 Conclusions

In this paper we have proposed a strategy for topological mapping of IPs/cores in a mesh-based NoC architecture. The approach uses heuristics based on multi-objective genetic algorithms to explore the mapping space and find the Pareto mappings that optimize performance and power consumption. The experiments carried out on both synthesized traffic and real applications (an MPEG-2 encoder/decoder system) confirm the efficiency, accuracy and scalability of the approach. Future developments will mainly address the definition of more efficient genetic operators to improve the precision and convergence speed of the algorithm. Evaluation will also be made of the possibility of optimizing mapping by acting on other architectural parameters such as routing strategies, switch buffer sizes, etc.

References

[Alpert et al. 1996] C. J. Alpert, L. W. Hagen, and A. B. Kahng. A hybrid multilevel/genetic approach for circuit partitioning. In *Fifth ACM/SIGDA Physical Design Workshop*, pages 100–105, Apr. 1996.

[Alpert and Kahng 1995] C. J. Alpert and A. B. Kahng. Recent developments in netlist partitioning: A survey. *VLSI Journal*, 19(1–2):1–81, 1995.

[AMBA Spec.] ARM. AMBA specification. `http://www.arm.com/`, May 1999.

[Ascia et al. 2004a] G. Ascia, V. Catania, and M. Palesi. A GA based design space exploration framework for parameterized system-on-a-chip platforms. *IEEE Transactions on Evolutionary Computation*, 8(4):329–346, Aug. 2004.

[Ascia et al. 2004b] 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.

[Banerjee et al. 2004] N. Banerjee, P. Vellanki, and K. S. Chatha. A power and performance model for network-on-chip architectures. In *Design, Automation and Test in Europe*, pages 1250–1255, Feb. 16–20 2004.

[Bertozzi et al. 2003] D. Bertozzi, L. Benini, and G. D. Micheli. Energy-reliability trade-off for NoCs. pages 107–129, 2003.

[Bright et al. 2001] M. S. Bright and T. Arslan. Synthesis of low-power DSP systems using a genetic algorithm. *IEEE Transactions on Evolutionary Computation*, 5(1):27–40, 2001.

[Coello et al. 2002] C. A. C. Coello, D. A. V. Veldhuizen, and G. B. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*, volume 5. Kluwer Academic Publishers, 2002.

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

[Gajski et al. 1994] D. Gajski, F. Vahid, S. Narayan, and J. Gong. *Specification and Design of Embedded Systems*. Prentice Hall, 1994.

[Garey and Johnson 1979a] M. R. Garey and D. S. Johnson. *Computers and intractability: a guide to the theory of NP-completeness*. Freeman and Company, 1979.

[Garey and Johnson 1979b] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W H Freeman & Co, 1979.

[Gasteier and Glesner 1999] M. Gasteier and M. Glesner. Bus-based communication synthesis on system level. *ACM Transactions on Design Automation of Electronic Systems*, 4(1):1–11, Jan. 1999.

[Glass and Ni 1998] C. J. Glass and L. M. Ni. The turn model for adaptive routing. In *25 Years ISCA: Retrospectives and Reprints*, pages 441–450, 1998.

[Hemani et al. 1999] A. Hemani, T. Meincke, S. Kumar, A. Postula, T. Olsson, P. Nilsson, J. Oberg, P. Ellervee, and D. Lundqvist. Lowering power consumption in clock by using globally asynchronous locally synchronous design style. In *ACM IEEE Design Automation Conference*, pages 873–878. ACM Press, 1999.

[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.

[CoreConnect] IBM Corporation. The CoreConnect bus architecture. `http://www.ibm.com/`.

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

[Jiang et al. 1997] Y.-M. Jiang, K.-T. Cheng, and A. Krstic. Estimation of maximum power and instantaneous current using a genetic algorithm. In *Proceedings of IEEE Custom Integrated Circuits Conference*, pages 135–138, May 1997.

[Kim et al. 2003] S.-M. Kim, J.-H. Park, S.-M. Park, B.-T. Koo, K.-S. Shin, K.-B. Suh, I.-K. Kim, N.-W. Eum, , and K.-S. Kim. Hardware-software implementation of MPEG-4 video codec. *ETRI Journal*, 25(6):489–502, Dec. 2003.

[Kommu et al. 1993] V. Kommu and I. Pomenraz. GAFAP: Genetic algorithm for FPGA technology mapping. In *European Design Automation Conference*, pages 300–305, 1993.

[Lahiri et al. 2001] K. Lahiri, A. Raghunathan, and S. Dey. System-level performance analysis for designing on-chip communication architectures. *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, 20(6):768–783, June 2001.

[Lahiri et al. 2004] K. Lahiri, A. Raghunathan, and S. Dey. Design space exploration for optimizing on-chip communication architectures. *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, 23(6):952–961, June 2004.

[XVID] C. Lampert, M. Militzer, and P. Ross. XviD MPEG4 core library. `http://www.xvid.org/`.

[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.

[Lienig 1997] J. Lienig. A parallel genetic algorithm for performance-driven VLSI routing. *IEEE Transactions on Evolutionary Computation*, 1(1):29–39, 1997.

[Mazumder and Rudnick 1999] P. Mazumder and E. M. Rudnick. *Genetic Algorithms for VLSI Design, Layout & Test Automation*. Prentice Hall, Inc., 1999.

[Mentor Graphics] Mentor Graphics. Inventra intellectual property cores. `http://www.mentor.com/inventra/cores/`.

[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.

[PalmChip] Palmchip. SoC bus architecutre. `http://www.palmchip.com/`.

[Pestana et al. 2004] S. G. Pestana, E. Rijpkema, A. Radulescu, K. Goossens, and O. P. Gangwal. Cost-performance trade-offs in networks on chip: A simulation-based approach. In *Design, Automation, and Test in Europe*, pages 896–901. IEEE Computer Society, Feb. 16–20 2004.

[Philips IP] Philips Electronics. Philips' IP portfolio. `http://www.semiconductors.philips.com`.

[Rowson and Sangiovanni-Vincentelli 1997] J. A. Rowson and A. Sangiovanni-Vincentelli. Interface-based design. In *Design Automation Conference*, pages 178–183, June 1997.

[Saab et al. 1996] D. Saab, Y. Saab, and J. Abraham. Automatic test vector cultivation for sequential VLSI circuits using genetic algorithms. *IEEE Transactions on Computer-Aided Design*, 15(10):1278–1285, Oct. 1996.

[Schiner et al. 2001] T. Schnier, X. Yao, and P. Liu. Digital filter design using multiple pareto fronts. In *The Third NASA/DoD Workshop on Evolvable Hardware*, pages 136–145, Long Beach, Cailfornia, July 12–14 2001. IEEE Computer Society.

[ITRS] International thechnology roadmap for semiconductors. Semiconductor Industry Association, 2003.

[Sikora 1997] T. Sikora. The MPEG-4 video standard verification model. *IEEE Transactions on Circuits and Systems for Video Technology*, 7(1):19–31, Feb. 1997.

[WishBone] Silicore. WISHBONE system-on-chip (SoC) interconnection architecture for portable IP cores. `http://www.silicore.net/`, Sept. 2002.

[BackPlane] Sonics. SiliconBackplane III MicroNetwork IP. `http://www.sonicsinc.com/`.

[STBus] STMicroelectronics. Stbus functional specs. `http://www.stmcu.com/`, Apr. 2003.

[Sylvester and Keutzer. 1998] D. Sylvester and K. Keutzer. Getting to the bottom of deep submicron. In *IEEE/ACM International Conference on Computer-aided design*, pages 203–211. ACM Press, 1998.

[Sylvester and Keutzer. 2000] D. Sylvester and K. Keutzer. A global wiring paradigm for deep submicron design. *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, 19(2):242–252, Feb. 2000.

[Valenzuela and Wang 2002] C. L. Valenzuela and P. Y. Wang. VLSI placement and area optimization using a genetic algorithm to breed normalized postfix expressions. *IEEE Transactions on Evolutionary Computation*, 6(4):390–401, 2002.

[VSI] VSI Alliance. On-chip bus attributes specification version 1. `http://www.vsi.org/`, Sept. 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.