Analog Integr Circ Sig Process (2012) 73:851–860
DOI 10.1007/s10470-012-9869-9

A multi-objective adaptive immune algorithm for multi-application NoC mapping

Martha Johanna Sepúlveda · Wang Jiang Chau · Guy Gogniat · Marius Strum

Received: 22 August 2011 / Revised: 11 January 2012 / Accepted: 5 May 2012 / Published online: 10 June 2012

Abstract Current SoC design trends are characterized by the integration of larger amount of IPs targeting a wide range of application fields. Such multi-application systems are constrained by a set of requirements. In such scenario network-on-chips (NoC) are becoming more important as the on-chip communication structure. Designing an optimal NoC for satisfying the requirements of each individual application requires the specification of a large set of configuration parameters leading to a wide solution space. It has been shown that IP mapping is one of the most critical parameters in NoC design, strongly influencing the SoC performance. IP mapping has been solved for single application systems using single and multi-objective optimization algorithms. In this paper we propose the use of a multi-objective adaptive immune algorithm (M^2AIA), *an evolutionary approach* to solve the multi-application NoC mapping problem. Latency and power consumption were adopted as the target multi-objective functions. To compare the efficiency of our approach, our results are compared with those of the genetic and branch and bound multi-objective mapping algorithms. We tested 11 well-known benchmarks, including random and real applications, and combines up to 8 applications at the same *SoC*. The experimental results showed that the M^2AIA decreases in average the power consumption and the latency 27.3 and 42.1 % compared to the branch and bound approach and 29.3 and 36.1 % over the genetic approach.

Keywords Network-on-chip · Mapping · Multi-objective optimization · Immune algorithm · Power · Latency

1 Introduction

Electronics system design is being revolutionized by the widespread adoption of the system-on-chip (*SoC*) paradigm. A SoC can integrate hundreds of cores on a single die. In such a scenario, SoC designers are faced with the task of meeting the design requirements in a reduced time-to-market. To be cost effective, SoCs are often programmable and integrate several different applications on the same chip (i.e. cell-phone, personal digital assistant) [1]. Although sharing many of the hardware components on the SoC, the different applications executed on the same die may present very different communication requirements and design constraints. Such type of system is called multi-application [1]. A communication centric paradigm, network-on-chip (*NoC*), has been adopted to address the interconnection issues of current SoCs. NoC has become the heart of the SoC [2]. A NoC is an integrated network that uses routers to allow the communication among the computation structure components. Routers carry out the communication exchange by means of packets. Packets consist of a set of minimal transmission units called *flits* (*flow control digits*). The information is queued at each router until the communication through another router or HW computational core has succeeded. The NoC configuration has a great impact on the cost and on the

M. J. Sepúlveda (✉) · W. J. Chau · M. Strum
Microelectronics Laboratory-EPUSP, University of São Paulo,
São Paulo, Brazil
e-mail: jsepulveda@lme.usp.br

W. J. Chau
e-mail: jcwang@lme.usp.br

M. Strum
e-mail: strum@lme.usp.br

G. Gogniat
LabSTICC, University of South Brittany-UBS, Lorient, France
e-mail: guy.gogniat@univ-ubs.fr

performance of the SoC [2]. A NoC may be configured by a set of global parameters (topology, size and mapping) and local parameters (link width, buffer configuration, flow control, routing technique, arbitration mechanism) leading to a very large NoC design space to be explored. The final configuration of the NoC must support the requirements of all the applications of the SoC. NoCs designed for a particular application does not necessarily meet the requirements of the remaining applications. Finding an optimal global solution is not an easy task [2]. This paper addresses the *mapping problem*. It deals with the allocation of HW cores onto the network routers such that all the applications requirements of the SoC are met and a set of performance metrics is optimized.

According to Murali et al. [1], NoC mapping is one of the most critical parameters in NoC design. Previous works showed that an optimal mapping may enhance the NoC performance up to 60 % [2]. Mapping is a quadratic assignment problem that is known to be NP-hard [3]. The search space of the problem increases factorially with the system size [4]. Furthermore, the mapping solution must satisfy all the system requirements consisting of multiple desired objectives that are frequently in contrast with each other [4]. To the best of our knowledge, only Murali et al. [1] addresses the multi-application NoC mapping with the aim of minimizing communication delay by exploiting the possibility of splitting traffic among various paths. However, previous works show that mapping strategies that search for single performance index optimization may lead to unacceptable values for other performance indexes [4]. The best mapping solutions have been obtained using a multi-objective strategy [4–6]. As a result, the designer obtains a set of best mapping alternatives (*Pareto optimal set, nondominated solutions*) featuring different trade-offs among the performance indexes [4–6]. Pareto dominance is used to compare and rank the mapping solutions. A mapping belongs to the *Pareto optimal set* if there is no other mapping that can improve at least one of the objectives without degrading any other objective (*nondominance*) [4–6, 8]. Three multi-objective mapping strategies, PBBB, MGAP and multi-objective adaptive immune algorithm (MAIA) have been proposed to solve single application NoC mapping [4, 5]. Their goal was the optimization of latency and power consumption for a mesh-based NoC. PBBB uses a branch and bound algorithm [4], MGAP uses a genetics algorithm [5] and MAIA uses an adaptive immune algorithm (AIA) [6].

This paper is an evolution of the work presented at [6]. In this work we propose M^{2}AIA an improved version of our MAIA, to solve the multi-application NoC mapping problem. M^{2}AIA explores the mapping space producing a set of best mapping alternatives. We compared our solution with modified versions that we implemented for the PBBB (MA_PBBB) and MGAP (MA_MGAP) algorithms. The Pareto optimal set of all 3 algorithms were then evaluated and compared using a NoC-based TLM *(SystemC)* simulation environment. The remaining text is divided into five sections. Section 2 presents an overview of the previous multi-objective mapping works. Section 3 presents the M^{2}AIA mapping algorithm. Section 4 shows our experimental results and the comparison among M^{2}AIA, MA_PBBB and MA_MGAP. Finally we present our conclusions in Sect. 5.

2 Related works

NoC Mapping has been widely explored [1, 4–8]. The purpose of these previous works is to find a NoC configuration that satisfies the requirements of the SoC. According to the number of the optimization objectives and the number of application supported by the SoC, previous works can be divided into 3 categories: (1) *Single objective and single application* [7, 8]; (2) *Multiple objective and single application* [4–6]; and (3) *Single objective and multiple applications* [1]. All works [1, 4–8] used an application characterization graph (APCG) that describes the communication requirements.

The works that belong to the first category [7, 8] presented a heuristic algorithm that selects the first NoC configuration that satisfies the single application SoC latency requirement. However, the works of [4–6] show that for many applications a single objective optimization is not enough. Moreover, the requirements of the set of applications may be contrasting.

The works of the second category [4–6] employed multi-objective algorithms to solve the mapping problem for mesh-based NoCs while optimizing latency and power indexes. In Ascia and Catania [4] PBBB, a branch and bound algorithm is proposed. PBBB maps the cores according to their communication traffic, creating a tree of mapping alternatives. At the bound phase, each mapping alternative is evaluated according to both optimization objectives through event-driven trace-based simulation (dynamical model). The best mapping alternatives are kept while the others are pruned. The branch and bound phases are repeated on the survivors. In Jena and Sharma [5] MGAP, a genetic mapping algorithm is presented. MGAP codifies different mapping alternatives in chromosomes. The mapping alternatives are evaluated through an analytical model (static model). Crossover and mutation operators are used in order to explore the mapping space (create new mapping alternatives).

Despite their good results, these strategies present some difficulties. The branch and bound strategy has two main disadvantages. First, the performance exploration time is highly dependent on the number of cores that must be mapped. Therefore, the usefulness of such strategy is limited to small systems [8]. Second, the search tree may grow exponentially without improving the solution [8]. The genetics strategy is a probabilistic heuristic that depends on the configuration of the genetic operators. Its effectiveness may be reduced in order to speed-up the algorithm's convergence. Furthermore, the lack of diversity (it progresses around the best solution) may result in suboptimal solutions [9]. AIA may overcome these drawbacks [10]. They integrate a wide set of features that improve local search while preventing the premature convergence by preserving the diversity of solutions in the population [10]. Previous works [10–12] show that AIA speeds-up the execution time and improve the search task over the genetic algorithms. Moreover, the PBBB and the MGAP use only a static or dynamic model approach to evaluate and select a mapping alternative. This characteristic can generate suboptimal solutions [6].

Our previous work of Sepulveda et al. [6] combines both; a static-dynamic model approach to find the mapping alternatives that optimizes the performance metrics. It uses an artificial immune algorithm to explore the efficiently the huge NoC design space. MAIA integrate a wide set of features that improve local search while preventing the premature convergence by preserving the diversity of solutions in the population. The pareto optimal set is then simulated through a SystemC-TLM SoC model. Despite their good results, these strategies must be modified to support the new systems requirements, characterized by supporting different applications that may have different performance requirements and design constraints.

The work of Murali et al. [1] is the only previous work that belongs to the third category. Murali et al. [1] proposes a heuristic capable of select the NoC configuration that satisfies the latency requirements of all the applications (just a single optimization objective).

For the best of our knowledge our work is the first attempt that addresses the multi-application NoC mapping while optimizing multiple performance indexes.

3 M^{2}AIA

An immune system protects the organism by producing antibodies capable of identify attackers (antigens). It constantly monitors the defense process through the evaluation of the *affinity* and *avidity*. They quantify the match between antigen–antibody pairs (for recognition) and between a single antibody and the whole antibody population (for diversity). The survival of a specific antibody depends on these values. The immune system integrates a wide set of mechanisms: pattern recognition (affinity), clonal selection (cloning the antibodies that best match the antigen), clonal suppression (killing the worst antibodies), mutation (modifying a set of antibodies), affinity maturation (creating a new set of antibodies), learning and memory (storing the successful antibodies). AIA have been successfully used in network security [11], parallel processing [13], image processing [14], robotic [15] and many challenging optimization problems [16].

M^{2}AIA uses the MAIA adaptive immune algorithm to solve the mapping problem. Table 1 shows the metaphors employed by MAIA. The algorithm is depicted on Fig. 1.

Table 1 Immune system metaphors

Immune system feature	MAIA
Antigen	Application characterization graph (APCG)
Antibody	Mapping alternative
Pattern recognition	Multi-objective quantification
Clonal selection	Top mapping alternatives selection
Clonal suppression	Mapping alternatives elimination
Mutation	Mapping alternatives modification
Maturation	Mapping alternatives creation
Learning and memory	Mapping solutions

3.1 Mapping algorithm for multiple-objectives and a single application

MAIA performs the mapping search using static evaluation (analytical model). MAIA determines the best NoC mappings (*Pareto optimal set*) which are then simulated through a TLM-based NoC performance evaluation framework. The simulation performs the dynamic evaluation of the mapping alternatives under different traffic conditions.

MAIA uses the APCG of the application and the size of the NoC to find the set of mapping alternatives that optimizes the objective functions. MAIA is composed of six phases.

Phase 1 Generating the initial set of mapping alternatives. It is composed of *M*[1] random generated and designer suggested mapping alternatives. Each mapping consists of an IP core-NoC router pair.

Phase 2 Evaluation of the *objective functions,* power consumption and latency of all the mapping alternatives. Its value will be used to rank the mapping alternatives. Note that MAIA can support any objective functions as well.

Phase 3 Ranking of mapping alternatives according to the *dominance value* of the objective functions results. A mapping is dominated by the solutions with lower

[1] All literals in italic refer to designer specified parameters.

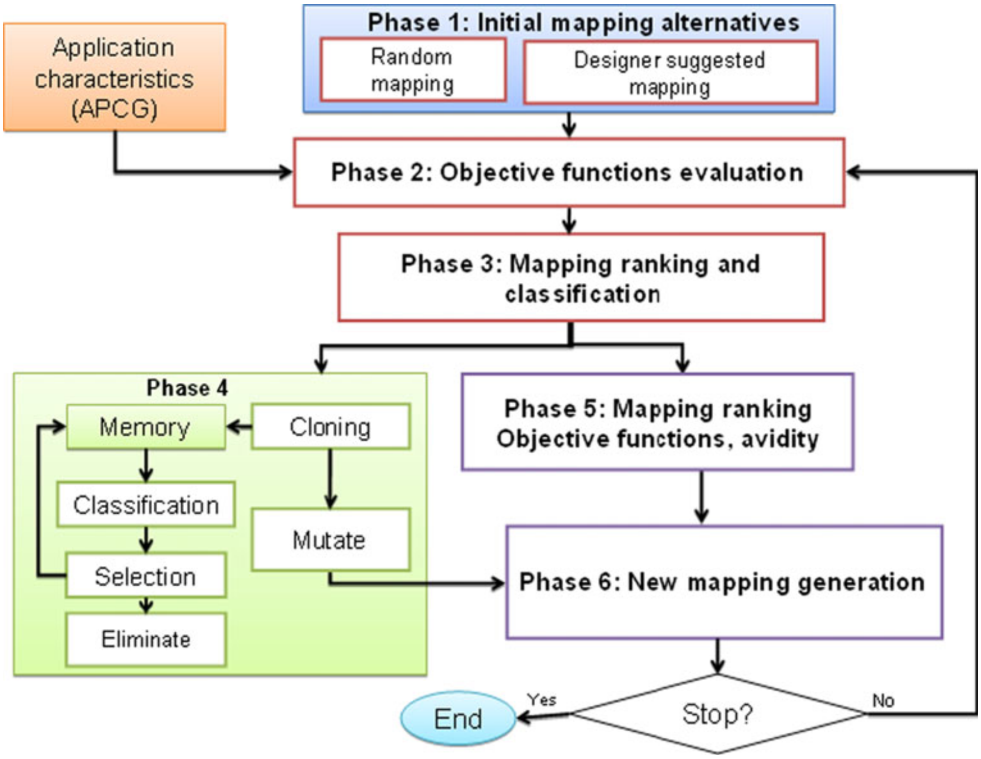
Application characteristics (APCG)
Phase 1: Initial mapping alternatives
Random mapping
Designer suggested mapping
Phase 2: Objective functions evaluation
Phase 3: Mapping ranking and classification
Phase 4
Memory
Cloning
Classification
Mutate
Selection
Eliminate
Phase 5: Mapping ranking Objective functions, avidity
Phase 6: New mapping generation
End
Yes
Stop?
No

Fig. 1 MAIA algorithm

objective function values [4, 5]. Each *m* mapping is assigned a fitness value *r*(*m, i*) based on its rank *d*(*m, i*) at *i* iteration, as in (1).

$$r(m,i) = 1 + d(m,i) \tag{1}$$

Phase 4 Refining the *Pareto optimal set*. The *N* non-dominated mapping alternatives ($r(m, i) = 1$) are copied and stored in memory (Pareto optimal set). The non-dominance characteristic of all the stored mapping solutions is verified. The new non-dominated alternatives are kept as part of the set of solutions. The remaining are erased from the memory. The copied mapping alternatives are modified using two mutation operators: *shift* (random shift of IPs) and *somatic point* (random swap of two IP). Figure 2 shows the M2 resulting mapping alternative after a shift of 2 positions of M1. Figure 3 shows the 2-positions somatic pint application.

Phase 5 Ranking the remaining (*M–N*) dominated mapping alternatives ($r(m, i) \neq 1$) according to two parameters: (1) the objective functions and (2) avidity (normalized sum of euclidean distances between every solution pair). The purpose is to identify and penalize mapping solutions in densely populated areas.

Phase 6 Generating *M* new mapping alternatives. The new set of mapping alternatives comes from the crossover of the *N* mutated mappings (step 4) and the (*M–N*) mapping alternatives (step 5). The crossover operator (Fig. 4) creates a new mapping (*M3*) from the combination of two different mappings (*M1, M2*).

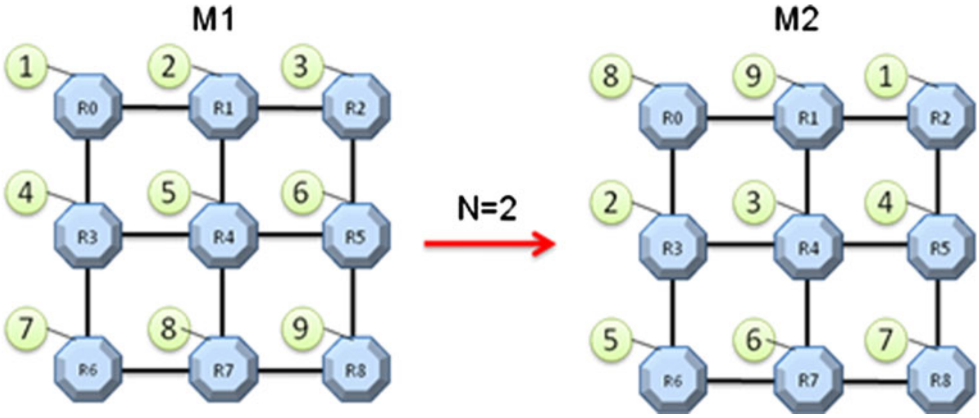

Fig. 2 Shift of M1 mapping alternative

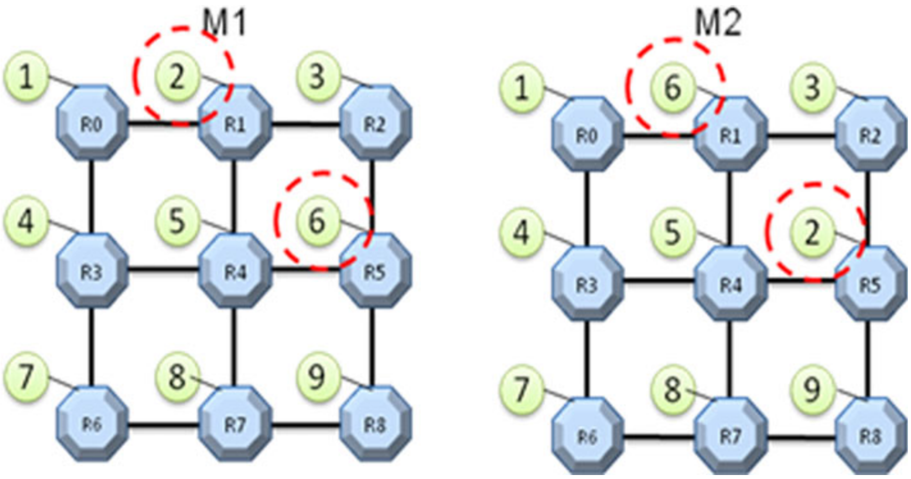

Fig. 3 Somatic point to a mapping alternative M1

MAIA stops when no more significant improvement can be expected.

3.2 Mapping algorithm for multi-application *SoC*

In order to solve the multi-application SoC mapping, M^{2}AIA adopts the combination of the APCGs of each application of the SoC in order to generate a synthetic APCG, used as an entry of the optimization process. The new APCG is called of Worst-Case APCG (WC-APCG). It includes all the IP cores integrated at the SoC. Figure 5 shows the mapping technique M^{2}AIA. For the communication flow between every pair of IP cores, the tightest communication requirements across all the applications are selected as the requirements of the WC-APCG. Thus the design constraints of all the individual applications are subsumed in the WC-APCG and any NoC mapping that satisfies the constraints in the WC-APCG will satisfy the constraints of each individual application. The WC-APCG is then used for the mapping process. Figure 6 shows an

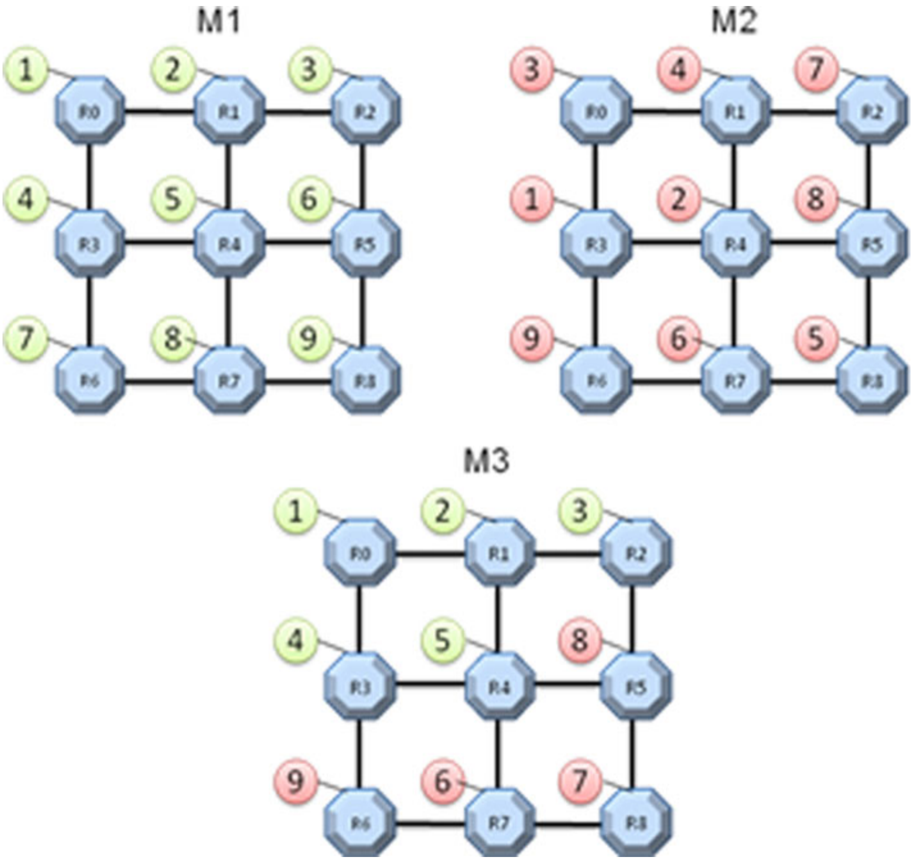

Fig. 4 Crossover of M1 and M2 mapping alternatives

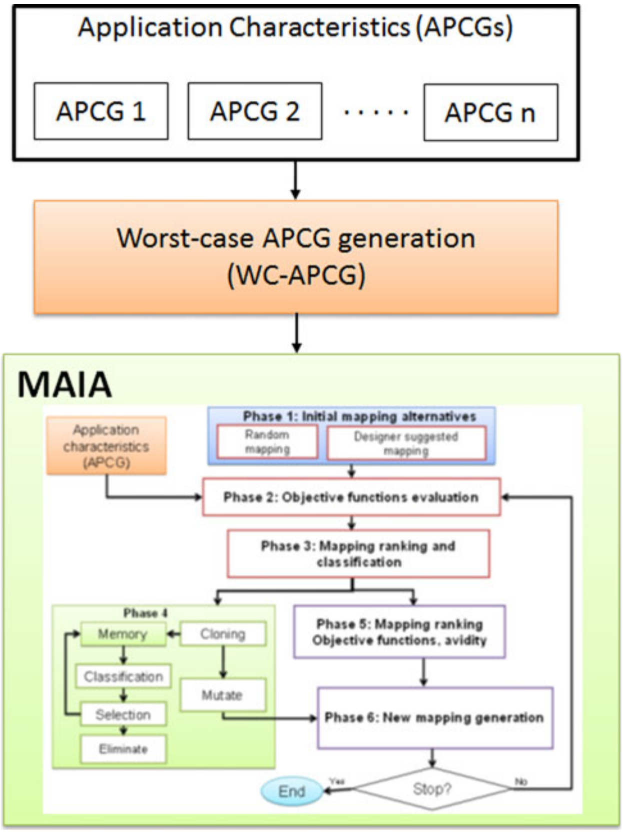
Application Characteristics (APCGs)
APCG 1
APCG 2
APCG n
Worst-case APCG generation (WC-APCG)
MAIA
Application characteristics (APCG)
Phase 1: Initial mapping alternatives
Random mapping
Designer suggested mapping
Phase 2: Objective functions evaluation
Phase 3: Mapping ranking and classification
Phase 4
Memory
Cloning
Classification
Mutate
Selection
Eliminate
Phase 5: Mapping ranking Objective functions, avidity
Phase 6: New mapping generation
End
Yes
Stop?
No

Fig. 5 Proposed approach M^2AIA

example of the generation of a WC-APCG from a SoC that executes 2 applications. Each application is described by an APCG. The applications share 5 of the 6 cores of the SoC. Each IP pair is characterized by the latency (cycles) and power (mW) requirements. The WC-APCG is composed of 6 cores, whose IP-pairs are characterized by the tightest requirements for both characteristics: latency and power. Once M^2AIA obtains the *Pareto optimal set* from the WC-APCG, a dynamical evaluation is performed using the SystemC-TLM SoC simulation and evaluation framework.

4 Analytical model

MAIA uses an analytical NoC model built from the queueing theory. In this approach, the NoC routers are represented as a collection of *service centers,* compose of queues and servers, which purpose is to attend the communication of packets. Figure 7 illustrates a single *service center*. Four processes can be distinguished at the model: (1) arrive of packets at the service center; (2) wait in the queue if necessary, when the server is busy; (3) receive service from the server; and (4) depart of packets.

Each *service center* is characterized by 3 parameters: (1) workload intensity, determined by the traffic conditions

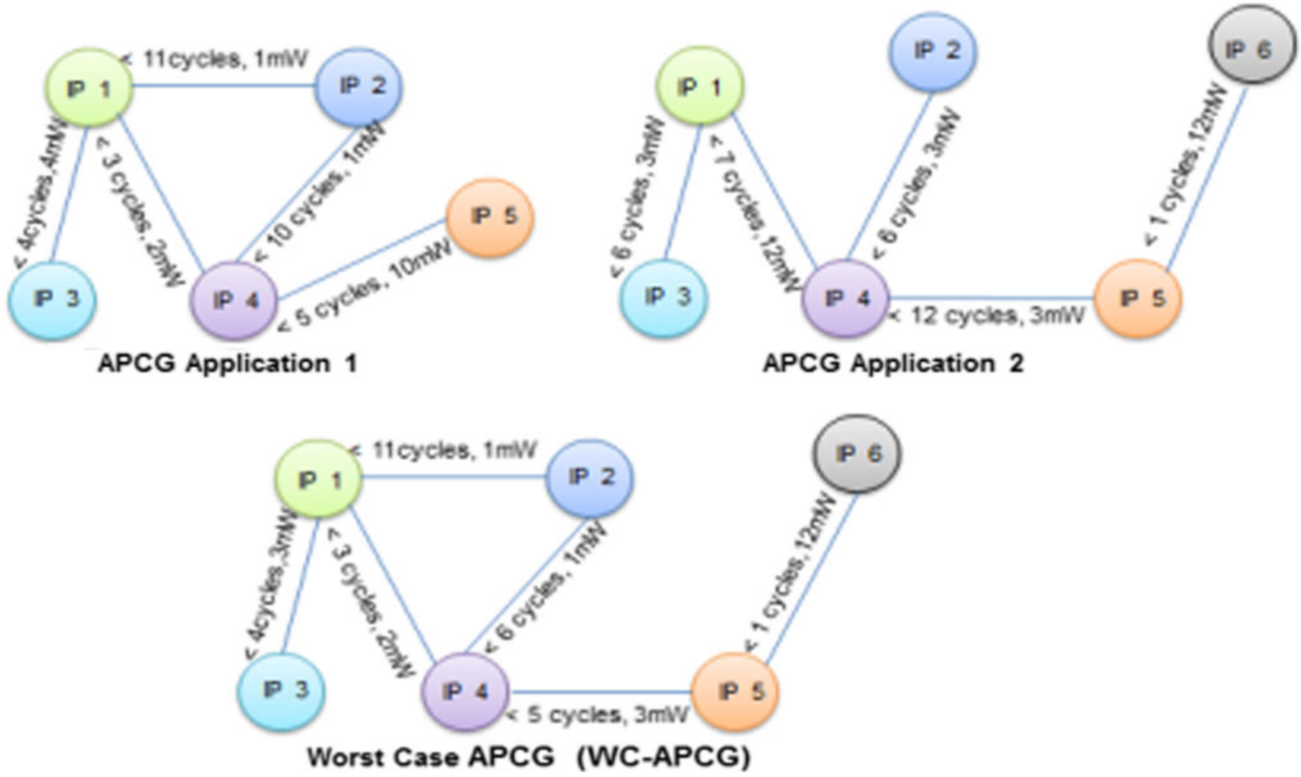

Fig. 6 Example of WC-APCG generation

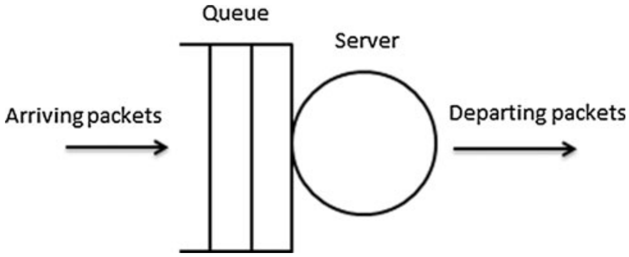

Fig. 7 Model of the router at the NoC

and expressed as the packet interarrival time; (2) service, determined by the service time; and (3) capacity, expressed as the queue size. The variation of the parameters of the model allows the description of different NoC configurations and traffic conditions.

Let λ_i be the packet arrival rate at the router *i*. It is compose of the communication flows that request commutation service from all the ports of the router. Let S_i the service time expend by the router to commutate a packet. Then the service rate μ_i is given by (2).

$$\mu_i = \frac{1}{S_i} \tag{2}$$

The traffic intensity at the router i is given by (3)

$$\rho_i = \frac{\lambda_i}{\mu_i}. \tag{3}$$

The bottleneck of the system will be the router with the higher ρ value [17]. The residence time T_i at each router can be calculated by (4). T_i represents the average time spent by the packet at the router.

$$T_i = \frac{S_i}{1-\rho_i} \tag{4}$$

The commutation time T_{ci} of the router, given by (5), is the result of the product of the number of commutation performed by the router η_i and the residence time T_i.

$$T_{ci} = n_i T_i \tag{5}$$

The NoC is modeled as an open network of *service centers*, that is, packets come from sources to be commuted by the NoC routers until being deliver to a specific sink, target of the communication. Network of queues have proven to be useful models to analyze the performance of complex systems [17]. For specific model parameter values, it is possible to calculate several performance metrics by solving (2–5), yielding performance measures such as latency, router utilization (the proportion of time the router is busy), residence time, queue length (the average number of packets at the router) and router throughput (the rate at which packet pass through the router.

5 Experimental results

M^{2}AIA was tested and compared with the MA_PBBB (branch and bound) [4] and MA_MGAP (genetic-based) [5] multi-objective algorithms. MA_PBBB and MA_MGAP were modified in order to support multi-application mapping. We implemented M^{2}AIA, MA_PBBB and MA_MGAP in C++. All were executed on a single Pentium IV–1.73 GHz personal computer. For comparison purposes, all 3 algorithms perform the mapping exploration using the same analytical model. The purpose of our experiments was to minimize the latency L$_{NoC}$ and the power consumption P$_{NoC}$. The objective functions are given by (6) and (7) respectively.

$$L_{\mathrm{NoC}} = \frac{\sum t_{\mathrm{acc}} + (h_{s,d} - 1)t_{\mathrm{c}} + t_{\mathrm{lea}}}{\#\ \mathrm{flits}}. \tag{6}$$

$$P_{\mathrm{NoC}} = \frac{\sum (h_{s,d} + 1)P_{\mathrm{R}} + h_{s,d}P_{\mathrm{L}}}{\#\ \mathrm{flits}} \tag{7}$$

L_{NoC} is determined by three components: (i) the time t_{acc} to access the NoC and insert the flit; (ii) the commutation time t_{c}, spent by the intermediate *h* routers from *s* source to *d* destination and; (iii) the time t_{lea} required to leave the NoC. P_{NoC} is determined by P_{R} and P_{L}, the power consumed in the routers and links respectively. P_{R} and P_{L} are proportional to the *channel utilization rate* and *router utilization rate* respectively.

All the tests were performed on a homogeneous wormhole 2D mesh-based NoC, a XY routing algorithm, 4-flits sized buffers and a round-robin arbitration technique. Table 2 shows the adopted M^{2}AIA parameters. The multi-objective mapping algorithms were used to solve the mapping problem of 11 multi-application benchmarks. Table 3 shows the characteristics of the experimental work. The APCG values of each benchmark were randomly selected. Benchmarks T1-T7 combine different well-known embedded communication patterns [8].

Benchmarks T8-T11 are part of MiBench tool suite. Each benchmark is composed of a set of applications targeting a specific area of the embedded market: automotive, consumer devices and security [10]. T8 represents embedded processors in network devices like switches and routers. It involves the shortest path calculation, tree and table lookups and data input/output. T9 represents typical applications of the automotive systems like air bag controllers, engine performance monitors and sensor systems. The processors require performance in basic math abilities, bit manipulation, data input/output and simple data organization. T10 focuses mainly on multimedia applications. It includes encoding/decoding algorithms, image color format conversion, image dithering and color palette reduction. T11 includes several common algorithms for data encryption, decryption and hashing.

Table 2 M^2AIA parameters

Parameter	Value
Initial population *M* (phase 2)	600
Latency objective function	L_{NoC}
Power objective function	P_{NoC}
Mutation probability (phase 5)	0.1
Crossover (phase 7)	40 %
Stop criterion	0.1

Table 3 Characteristics of the experimental work

Benchmark	#IP's	#APCGs	Type
T1	9	5	Hot spot/transpose
T2	16		Hot spot/transpose/random
T3	25		
T4	36		
T5	49		
T6	100		
T7	12		3-node rooted (1, 2, 3) forest
T8	7	3	Networking
T9	16	8	Automotive
T10		5	Costumer devices
T11		4	Security

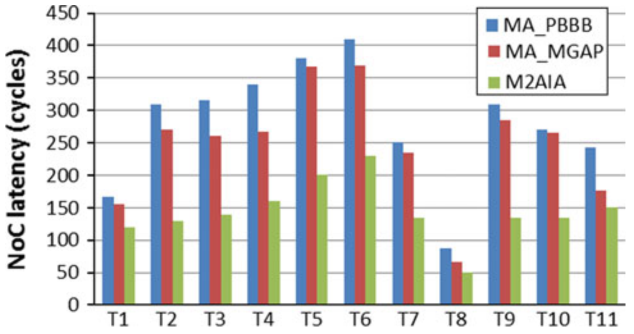

Fig. 8 NoC latency static evaluation results

Figures 8 and 9 show the analytical L_{NOC} and P_{NOC} results for the pareto optimal set of the three approaches. Figures 10 and 11 show the TLM-based simulation results for the NoC power and NoC latency respectively of the Pareto optimal set of the 3 algorithms. The comparisons among these results show the fidelity of the prediction of our model. However the lack of precisions is because the difficulty of the analytical (static) model to represent the dynamic behavior of the NoC. The M^2AIA algorithm produced better results for all benchmarks. M^2AIA found NoC mappings that satisfy the requirements of the set of applications and also achieves higher optimizations in power and latency when compared with the MA_PBBB and MA_MGAP. Our algorithm decreases the power consumption in average 27.3 and 42.1 % and the latency 29.3 and 36.1 % over the MA_PBBB and MA_MGAP,

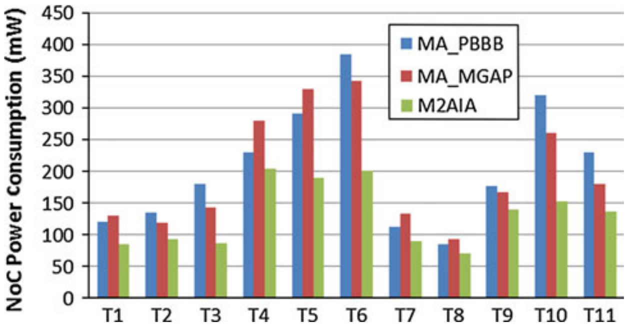

Fig. 9 NoC power static evaluation results

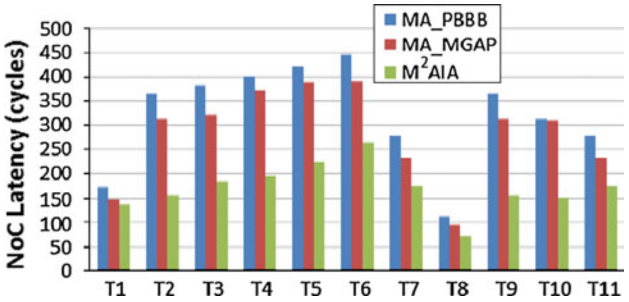

Fig. 10 NoC latency dynamic results

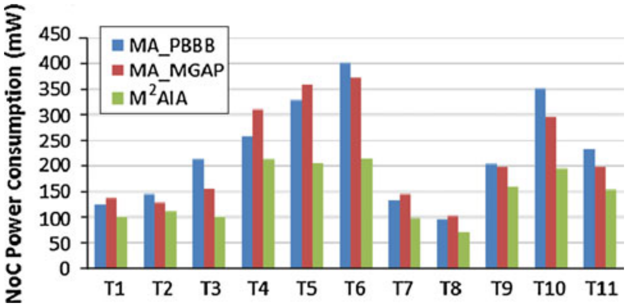

Fig. 11 NoC power dynamic results

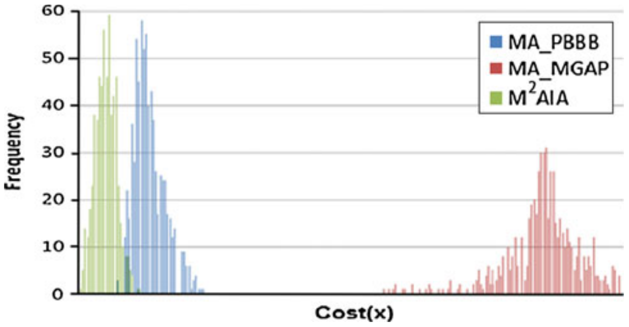

Fig. 12 Average frequency distribution of results

Table 4 Spacing and spread metrics

	Spacing	Spread	Execution time				
			#IP = 7 (%)	#IP = 16 (%)	#IP = 25 (%)	#IP = 49 (%)	#IP = 100 (%)
MA_PBBB	0.0101	0.9275	0	127	289	312	529
MA_MGAP	0.0096	0.8891	0	54	130	184	213
M^2AIA	0.0078	0.8505	0.0	0.1	0.4	0.6	0.8

respectively. The best improvement over MA_PBBB was for T3 (52 and 51 % for power and NoC latency respectively). The best improvement over MA_MGAP was for T10 (34 and 50 % respectively).

Figure 12 shows frequency distributions for the function *cost(x)* that is the addition of both objective functions values L_{NOC} and P_{NOC}. Although the lack of physical sense of *cost(x)*, it gives the designer an idea of the quality of the obtained mapping solutions of each approach. The results correspond to the last 600 mapping alternatives for T6, the benchmark that achieves the higher values of latencies. Figure 12 shows that M2AIA has a higher probability of finding better solutions than MA_MGAP and MA_PBBB. This behavior is repeated for all the benchmarks.

We evaluated the performance of the 3 algorithms. We use three performance metrics: 1-Spacing P (distance between the mapping solutions); 2-Spread A (distance among all the mapping alternatives); and 3-Execution time T (time spent to reach the stop criterion). Table 4 shows the result of the 3 metrics. The execution time results are expressed as a percentage of time spent to find the solution of the smaller system (7 IPs).

The results show that M^2AIA achieves a lower spacing and spread values, so that it performs a uniform exploration. Moreover, M^2AIA speedups the mapping search almost 100,000 times when compared to the MA_PBBB and MA_MGAP techniques. These are desirable characteristics for a search algorithm [6]. The results show the M^2AIA independence of the number of IP cores. Also M^2AIA obtained better results than all other reported algorithms in a shorter time.

6 Conclusions

In this paper we propose the use of a M^2AIA, an evolutionary approach to solve the multi-application NoC mapping problem. The contributions of our work include the adoption of an adaptive immune algorithm combined with the use of both, static and dynamic mapping evaluation techniques in order to improve the efficiency of the exploration of the multi-application mapping space. Previous multi-objective algorithms were modified in order to support the multi-application mapping. M^2AIA was tested for a mesh-based NoC targeting the minimization of the total amount of power consumption and latency. M^2AIA may use others objective functions and NoC topology.

As future work, we plan to refine our analytical model in order to include a wider set of traffic characteristics: topologies, natures (long-range-dependence) and types (QoS). We also plan to use M^2AIA to define the NoC sizing and NoC topology parameters.

References

1. Murali, S., Coenen, M., Radulescu, A., Goosens, K., & De Micheli, G. (2006). Mapping and configuration methods for multi-use case NoCs. In *Proceedings of Asia and South Pacific DAC conference*. Yokohama, Japan.
2. Benini, L., & Bertozzi, D. (2005). Network-on-chip architectures and design methods. *IEEE Proceedings Computers and Digital Techniques, 152* (6), 261–272.
3. Garey, R., & Johnson, D. S. (1979). *Computers and intractability: A guide to the theory of NP-completeness*. San Francisco, CA: Freeman.
4. Ascia, G., & Catania, V. (2005). An Evolutionary approach to network-on-chip mapping problem. In *proceedings of the IEEE congress on evolutionary computations*. Edinburgh, UK.
5. Jena, R., & Sharma, G. (2007). A Multi-objective evolutionary algorithm based optimization model for network-on-chip synthesis. In *proceedings of international conference on information technology*. Matshushima, Japan.
6. Sepulveda, J., Pires, R., Strum, M., & Chau, W. J. (2009). An evolutive approach for multi-objective NoC mapping. In *proceedings of 17th IFIP/IEEE international conference on very large scale integration, VLSI-SoC*. Florianopolis, Brazil.
7. Hu, J., & Marculescu, R. (2005). Energy and performance aware mapping for regular NoC architectures. In *proceedings of computer aided design integration circuits and systems*. Anaheim, CA.
8. Bolotin, E., Cidon, I., Ginosar, R., & Kolodny, A. (2004). QNoC: QoS architecture and design process for network on chip. *Journal of System Architecture, 50*(2–3), 105–128.
9. Pasricha, S., Dutt, N., & Kurdahi, F. (2009). Dynamically reconfigurable on-chip communication architecture. In *proceedings of Asia and South Pacific DAC conference*. Yokohama, Japan.
10. Guthaus, M., Ringenberg, J., & Ernst, D. (2001). MiBench: A free, commercially representative embedded benchmark suite. In *proceedings of international workshop on workload characterization*. Austin, TX.
11. Hofmeyr, S. A., & Forrest, S. (1999). Immunity by design: An artificial immune system. In *proceedings of 1999 GECCO conference*. Lake Como, Italy.
12. Deb, K. (1999). Multi-objective algorithms: Problem difficulties and construction of test problems. *Evolutionary Computation, 7*(3), 205–230.

13. King, L. R., Russ, H., Lambert, B., & Reese, S. (2000). An artificial immune system model for intelligent agents. *Future Generation Computer Systems, 17*(4), 335–343.
14. Rodin, V., Benzinou, A., Guillaud, A., Ballet, A., Harrouet, P., Tisseau, J., et al. (2004). An immune oriented multi-agent system for biological image processing. *Pattern Recognition, 37*, 631–645.
15. Watanabe Y., Ishiguro A., & Uchkawa Y. (2011) Decentralized behavior arbitration mechanism for autonomous mobile robot using immune system. In D. Dasgupta (Ed.) Books artificial immune systems and their applications (2nd ed.). Berlin: Springer-Verlag.
16. Bakhouya, M., & Gaber, J. (2007). An immune inspired-based optimization algorithm: Application to the traveling salesman problem. *Advanced Modeling and Optimization, 9*(1), 105–116.
17. Gross, D., & Harris, C. (2011). *Fundamentals of queueing theory* (4th ed.). New York: Wiley series in probability and statistics.

Martha Johanna Sepúlveda received the B.S. degree (summa cum laude) in electronics engineering from the Colombia National University, Bogota, Colombia, in 2004. The M.S. degree in electrical engineering in 2006 and the Ph.D. degree in electrical engineering from the Sao Paulo University, Sao Paulo, Brazil, in 2011. Currently, she holds a Postdoctoral position in the microelectronics laboratory at the Sao Paulo University, Sao Paulo, Brazil. Her current research interests include the design of high performance SoC communication structures and the inclusion of Quality-of-Service and security in embedded systems.

Wang Jiang Chau is an associate professor of the Electrical engineering department at the University of Sao Paulo. He obtained his B.Sc. and M.Sc. degrees in Electrical Engineering from the University of São Paulo in 1981 and 1988, respectively. Ph.D. in Electrical engineering from the Syracuse University in 1993. Prof. Wang has experience in the area of electrical engineering and computation, acting mainly in the fields of functional verification of digital systems, dynamic reconfiguration, and design and system level system modeling.

Guy Gogniat received the M.S.E.E. degree from the University of Paris Sud, Orsay, France, in 1994, and the Ph.D. degree in electrical and computer engineering from the University of Nice-Sophia, Antipolis, France, in 1997. He is currently a Professor in electrical and computer engineering with the University of Bretagne-Sud, Lorient, France, where he has been since 1998. In 2005, he spent one year as an invited Researcher with the University of Massachusetts, Amherst, USA, where he worked on embedded system security using reconfigurable technologies. His work focuses on embedded systems design methodologies and tools. He also conducts research in the domain of reconfigurable and adaptive computing and embedded system security.

Marius Strum received from the University of Sao Paulo the Mathematician degree from the Institute of Mathematics and Statistics and the Electrical Engineering degree from the Polytechnical School in 1971. In 1974 Prof. Strum worked in the Microelectronics laboratory of the University of Sao Paulo EPUSP (LME) where he obtained the M.Sc. and Ph.D. degrees in 1977 and 1983 respectively. In 1986 and 1987 he handled a Postdoctoral position at the IMEC in Leuven-Belgium. From 1990 he is an associated professor of the Department of Electrical Engineering of the University of São Paulo. Currently, Prof. Strum heads the GSEIS group in the LME-EPUSP whose research topics are the design of (ASICs) and development of tools of CAD applied to the automatic synthesis of ASICs and, of integrated digital systems (SoCs). His current interest is the design of high performance SoC communication structures.