CGMAP: a new approach to Network-on-Chip mapping problem

Fahime Moein-darbari[1a)], **Ahmad Khademzade**[2], **and Golnar Gharooni-fard**[1]

[1] *Department of Computer Science, Arak Azad University, Arak, Iran*

[2] *Iran Telecommunication Research Center, Tehran, Iran*

a) *fahimeh_md@iau-arak.ac.ir*

Abstract: The NoC paradigm is one, if not the only one, fit to enable the integration of an exceedingly large number of computational, logical and storage blocks in a single chip. This paper presents a novel technique called CGMAP, which finds a mapping of the vertices of a task graph to the tiles of a mesh based NoC architecture. The proposed algorithm is basically a genetic algorithm, which takes the advantages of the chaotic systems by using them instead of the random processes in the GA. Experimental results show that the proposed algorithm performs as well as the previously proposed mapping algorithms considering some performance indexes such as hop distance, energy consumption, and latency ratio.

Keywords: Network-on-Chip, mapping, chaos-genetic algorithm

Classification: Integrated circuits

References

[1] L. Benini and G. De Micheli, "Networks on chip: a new SoC paradigm," *IEEE Comput.*, vol. 35, no. 1, pp. 70–78, Jan. 2002.

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

[3] X. F. Yan, D. Z. Chen, and S. X. Hu, "Chaos-genetic algorithms for optimizing the operating conditions based on RBF-PLS model," *Elsevier Computers and Chemical Engineering*, pp. 1393–1404, Feb. 2003.

[4] P. Kaelo and M. M. Ali, "Integrated Crossover rules in real coded genetic algorithms," *European Journal of Operatioanl Research*, vol. 176, no. 1, pp. 60–76, 2007.

[5] G. Zhang, G. Zhang, J. Ma, and C. Zhou, "Three real coded Genetic algoritm with new mutation operators," ISKE 2007, Chengdu, China, in ISKE-2007 Proceedings, ed Tianrui Li, Yang Xu, Da Ruan, Atlantis Press, France, pp. 531–534, Oct. 2007.

[6] R. Caponetto, L. Fortuna, S. Fazzino, and M. G. Xibilia, "Chaotic sequences to improve the performance of evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 7, pp. 289–304, 2003.

[7] S. Murali and G. D. Micheli, "Bandwidth-Constrained Mapping of Cores onto NoC Architectures," *Proc. DATE'04*, vol. 2, pp. 896–901, Feb. 2004.

© IEICE 2009
DOI: 10.1587/elex.6.27
Received November 03, 2008
Accepted November 22, 2008
Published January 10, 2009

[8] W. T. Shen, C. H. Chao, Y. K. Lien, and A. Y. Wu, "A new binomial mapping and optimization algorithm for reduced-complexity mesh-based on-chip-network," *Proc. 1st International Symposium on Network-on-Chip (NOCS'07)*, June 2007.
[9] J. Hu and R. Marculescu, "Energy-Aware Mapping for Tile-based NoC Architectures Under Performance Constraints," *ASP-DAC 2003*, pp. 233–239, Jan. 2003.

1 Introduction

The growing complexity of embedded multi-processor architectures will soon require highly scalable communication infrastructures. There are many reasons why traditional bus-based shared medium architectures will not be suitable anymore, including high signal propagation delays, signal integrity issues, and etc. The Network-on-Chip architecture provides the communication infrastructure for the resources. In this way it is possible to develop the hardware of resources independently as stand-alone blocks and create the NoC by connecting the blocks as elements in the network [1]. There are many different ways to design a NoC, depending on the network architecture and protocol choice. We choose a two dimensional mesh interconnection, which is very simple 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 has proved to be a simple task, resulting in potentially small switches, high bandwidth, short clock cycle, and overall scalability.

In this paper, we present a new algorithm for mapping IP cores on the tiles of a mesh-based NoC. Since the mapping problem is an NP-hard one [2], we attempt to develop a technique to obtain an optimal approximation of the desired performance indexes (delay, energy consumption, etc.). The Chaos-Genetic Algorithm (CGA) presented in [3], seems to have the potential to achieve this goal.

2 Problem description

The input of the CGMAP algorithm is a directed task graph $G(V, E)$ in which every $v_i \in V$ denotes a processing element or a memory unit (generally an IP core), and a directed edge $e_k = (v_i, v_j)$ denotes a communication trace from the source node v_i to the destination node v_j. The weight of the edges $w(e_k)$ usually refers to the communication cost between two corresponding nodes.

A mesh-based topology of NoC is defined by $T(U, L)$ where each vertex $u_i \in U$ denotes a node in the topology and each $l_i \in L$ denotes a physical link between two vertices. The weight of a link $w(l_k)$ represents the bandwidth available across the link l_k.

The problem of mapping IPs concerns selecting IPs from the task graph and assigning them to tiles of the NoC architecture. CGMAP combines the two ideas of genetic algorithms and chaotic systems. Genetic Algorithms can

© IEICE 2009
DOI: 10.1587/elex.6.27
Received November 03, 2008
Accepted November 22, 2008
Published January 10, 2009

find feasible solutions in a very short time and using chaotic sequences leads to more effective search.

3 CGMAP algorithm

In this section, we present our chaos-genetic-based algorithm CGMAP. Representing the solution is simplified by the fact that the genes in our problem can only take real values between 1 and n. The value of n is proportional to the number of tiles in the mesh. The length of each chromosome, however, depends on the number of nodes in the communication task graph. Fig. 1 exhibits the process of mapping a task graph into a tile based 2-D mesh.

In a 3×3 mesh, for instance, a chromosome is formed by 9 genes, each taking a value between 1 and 9. When numbering the mesh tiles as it is shown in Fig. 1, the value of the i-th gene in a solution represents the place of the i-th IP core in mesh.

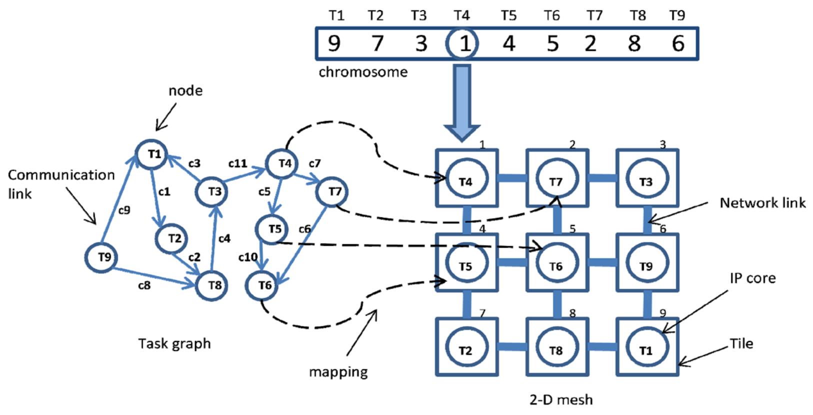


Fig. 1. The process of mapping a task graph into a 2-D mesh

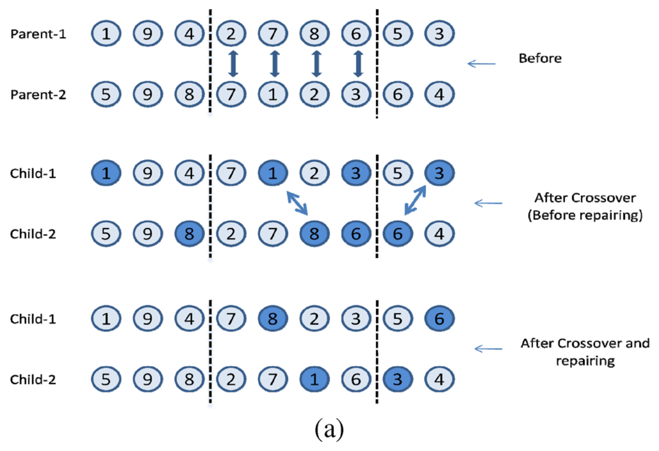
Population size is another important parameter. In an actual application, it would be common to have somewhere between a few dozen and a few hundred individuals. For the purposes of this problem, however, we assume that our first population consists of 100 individuals.

The next step is to initialize the population, which is usually done randomly. Therefore, a random number generator is utilized to assign values between 1 and n, to each of the n positions in each of the 100 individuals. Then the chaotic mapping operator is applied to each individual in the initial population in order to create a chaotic population. The chaotic mapping operator will be discussed in detail, in the next section.

At this stage, the fitness of all 200 individuals is evaluated. The fitness value is often proportional to the output value of the function being optimized. Since data always take the shortest distance in the network and there often exist more than one such path for data going from node $v_i = (x_i, y_i)$ to $v_j = (x_j, y_j)$, we estimated this distance as:

$$hd(e_k) = (|x_i - x_j| + |y_i - y_j|) \tag{1}$$

© IEICE 2009
DOI: 10.1587/elex.6.27
Received November 03, 2008
Accepted November 22, 2008
Published January 10, 2009


Parent-1
Parent-2
Before
Child-1
Child-2
After Crossover (Before repairing)
Child-1
Child-2
After Crossover and repairing
(a)

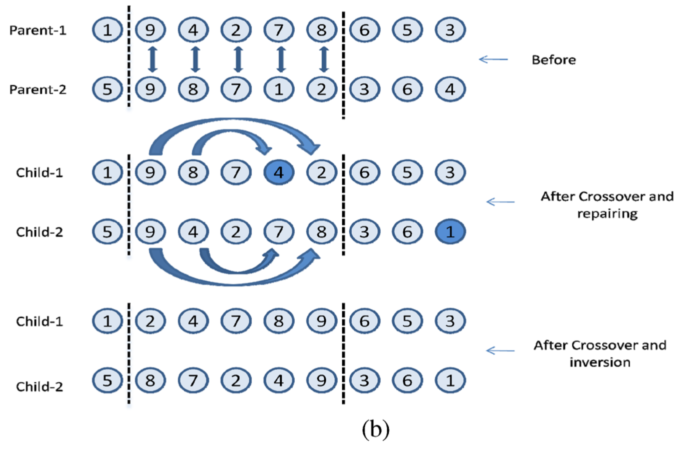


Fig. 2. (a) Two-point crossover (b) Crossover and inversion

and defined the fitness function as follows:

$$F = \sum_{\forall e_k} w(e_k)hd(e_k) \tag{2}$$

Elitism is incorporated into the algorithm by transferring the single fittest individual directly to the next generation.

Crossover is performed on randomly selected individuals according to a varying probability. In order to improve the performance of CGMAP different types of crossover are integrated into the algorithm. Fig. 2 demonstrates two examples of the crossover operations used in CGMAP. Finally, for mutation, as in the case of crossover, the probability can vary widely according to the application. However, in our proposed algorithm a constant mutation rate (0.01) is applied. The new population is now ready for another round of crossover, and mutation, producing yet another generation. So the initial population is replaced by these newly generated individuals. Obviously, more generations are produced until the stopping condition (a maximum number of generations k) is met. The fittest chromosome is thus returned as a solution.

3.1 Chaotic mapping operator

Recently, the idea of using chaotic systems instead of random processes has been noticed in several fields. One of these fields is optimization theory.

© IEICE 2009
DOI: 10.1587/elex.6.27
Received November 03, 2008
Accepted November 22, 2008
Published January 10, 2009

In random-based optimization algorithms, the role of randomness can be played by chaotic dynamics. Experimental studies assert that the benefits of using chaotic instead of random signals are often evident although it is not mathematically proved yet [6]. For example in genetic algorithms, chaotic sequences increase the value of some measured algorithm-performance indexes with respect to random sequences.

The evolution process of the chaotic variables could be defined through the following equation:

$$cx_i^{(k+1)} = 4cx_i^{(k)}\left(1 - cx_i^{(k)}\right), \qquad i = 1, 2, \ldots, n \tag{3}$$

In which cx_i is the i-th chaotic variable and k and $k+1$ denote the number of iterations. Note that values of cx_i, are distributed in the range of (0,1). In order to perform the chaotic mapping, the following procedure is proposed:

1. Divide the interval (0,1) to n equal sub-intervals (n denotes the number of tiles in mesh), of which the lower limit $[a_1, a_2, \ldots, a_n]$ is represented by vector a, and the upper limit $[b_1, b_2, \ldots, b_n]$ by vector b.

2. The value of each gene in the first randomly produced population is mapped to new values of x_i in the range of (0,1), so that $x_i \in (a_i, b_i)$.

3. These values of $x_i^{(1)}$, $i = 1, 2, \ldots, n$ are linearly mapped using the operator
$$cx_i = \frac{1}{b_i - a_i}(x_i - a_i) \tag{4}$$

4. The next iteration chaotic variables $cx_i^{(2)}$, will be produced through applying Eq. 4 to $cx_i^{(1)}$, values, generated in the previous step.

5. The chaotic variables $cx_i^{(2)}$, are then used to produce $x_i^{(2)}$, using
$$x_i^{(2)} = a_i + cx_i^{(2)}(b_i - a_i), \qquad i = 1, 2, \ldots, n \tag{5}$$

Note that, $x_i^{(2)}$ has a value between 0 and 1, and it should be mapped to a value between 0 and n in order to be used in our algorithm. Thus, we can continue to produce the values of $x_i^{(2)}$ for each chromosome, through the operators defined in Eqs. (4–6).

4 Experimental results

In this section, we present the results of the execution of CGMAP on two benchmark applications, Video Object Plane Decoder (VOPD) with 16 IP-cores and 20 links and MPEG-4 decoder with 12 nodes and 13 links. Afterwards we compare these results with those of previous mapping algorithms such as NMAP [7], BMAP [8], PBB [9], etc. using the same routing and scheduling characteristics.

Table I. (a) shows that, our proposed algorithm performs as well as NMAP, PBB and BMAP considering their communication costs. In fact, in case of MPEG-4 decoder, CGMAP achieves lower communication cost than the

© IEICE 2009
DOI: 10.1587/elex.6.27
Received November 03, 2008
Accepted November 22, 2008
Published January 10, 2009

other algorithms. The obtained results have been modeled and simulated by NS-2, and parameters such as energy consumption, hop distance and latency rate, have been calculated for NMAP and CGMAP. The *distance vector* routing algorithm is used to detect the path in NS simulator and the network traffic is created by *Exponential* and *Pareto* distributions. A queue with the length of 2 with a FIFO timing mechanism is specified for each external port of a switch. The amount of the injected traffic is gradually increased from 27% to 100%. Table I. (b), demonstrates a comparison between the hop counts of the three most efficient mapping algorithms for two benchmark applications. The table shows that the hop number is decreased to 0.98 in the first application (MPEG-4). But there is a very slight rise (0.008) in the hop number in the case of the second application (VOPD) compared to NMAP.

Because NMAP has a great performance in [7], it is adopted as a reference point to judge our approach. Table I. (c), (d) shows the simulation results of CGMAP compared with NMAP. Since CGMAP is mainly based on genetic algorithms, therefore different results may be obtained each time the algorithm is executed. We use the average result obtained after 20 times of executing the algorithm on a specific application. As is shown in Table I. (c), (d) using CGMAP reduced the energy consumption and latency for MPEG-4, especially in higher traffic loads.

In order to evaluate the time complexity of CGMAP, let n be the number of nodes in the task graph, e be the number of edges in the task graph, c and k be the constants related to the GA's design defining the population size and the number of generations. Since CGMAP is based on Genetic Algorithm therefore, its complexity depends on the complexity of the fitness function which is described in section 3. The complexity of the fitness function is calculated as $\theta(c \times e \times n^2)$.

Besides, the fitness function will be iterated k times, equal to the maximum number of generations, to reach the optimal solution. Therefore the complexity of the algorithm is estimated as:

$$O(k(c \times e \times n^2)) \tag{6}$$

NMAP costs more than $O(n^4 \log(n))$ computation time to do its shortest path optimization and get the best result [8], which is greater than the complexity of CGMAP.

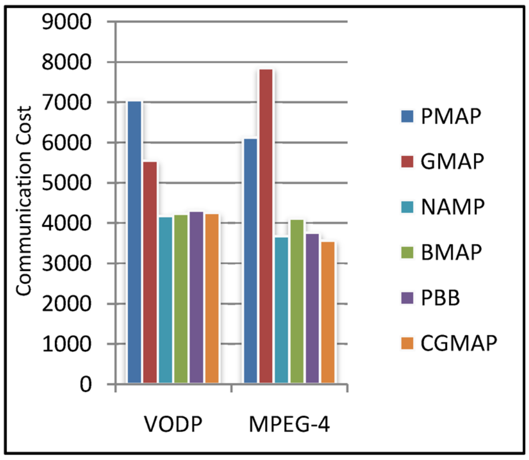
5 Conclusion

In this work we introduce the novel idea of combining the two concepts of genetic algorithms and chaotic sequences to solve the mapping problem in NoC. The proposed algorithm CGMAP, appears to work as well as the most efficient mapping algorithms introduced in the previous works. However the results are highly dependent on the encoded scheme and the crossover and mutation operators being used. CGMAP utilizes the characteristics of the chaotic variable to distribute the individuals of subgenerations ergodically in the defined space and thus avoid the premature convergence of the individuals in the subgenerations.

© IEICE 2009
DOI: 10.1587/elex.6.27
Received November 03, 2008
Accepted November 22, 2008
Published January 10, 2009

Table I. (a) Comparison between the communication costs of six mapping algorithms (b) Hop count table (c), (d) Simulation results of NMAP and CGMAP for MPEG-4

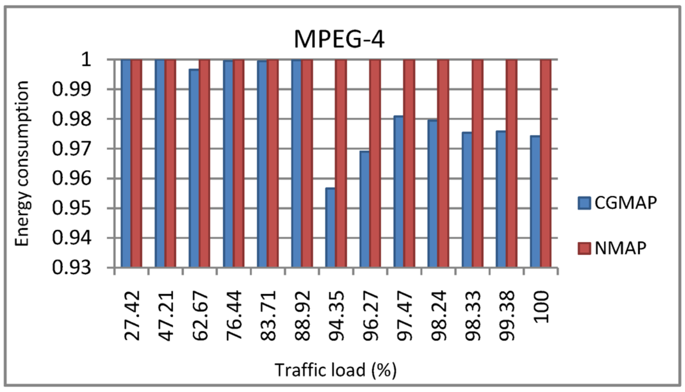
(a)



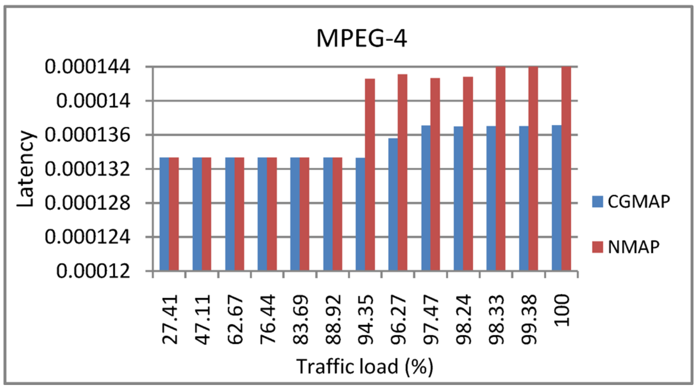
(b)

	Communication Cost (Hop)		
	NMAP	BMAP	CGMAP
MPEG-4	1	1.02	0.98
VOPD	1	1.71	1.008

(c)



(d)



© IEICE 2009
DOI: 10.1587/elex.6.27
Received November 03, 2008
Accepted November 22, 2008
Published January 10, 2009

Experiments carried out on real applications (an MPEG-4 encoder/decoder system and VOPD application) confirm the efficiency, lower time complexity and scalability of the proposed approach.

The idea of using the advantages of chaotic sequences combined with other heuristic algorithms such as Particle Swarm (PSO), Branch and Bound, and etc. to gain more efficient results in similar problems, can be a good implication for the future studies in this area.

DOI: 10.1587/elex.6.27
Received November 03, 2008
Accepted November 22, 2008
Published January 10, 2009