ISSN 2007-9737

Adaptive Algorithm Based on Renyi's Entropy for Task Mapping in a Hierarchical Wireless Network-on-Chip Architecture

Maribell Sacanamboy[1,3], Freddy Bolaños[2], Alvaro Bernal[3]

[1] Pontificia Universidad Javeriana Cali, Electronics and Computer Sciences Department, Cali, Colombia

[2] Universidad Nacional de Colombia, Electrical Energy and Automation Department, Medellín, Colombia

[3] Universidad del Valle, Electrical and Electronics Engineering School, Cali, Colombia

msacanamboy@javerianacali.edu.co, fbolanosm@unal.edu.co,alvaro.bernal@correounivalle.edu.co

Abstract. This paper describes the use of Renyi's entropy as a way to improve the convergence time of the Population-Based Incremental Learning (PBIL) optimization algorithm. As a case study, the algorithm was used in a hierarchical wireless network-on-chip (WiNoC) for the sake of performing the optimal task mapping of applications. Two versions of Renyi's entropy are used and compared to the more traditional Shannon formulation. The obtained results are promising and suggest that Renyi's entropy may help to reduce the PBIL convergence time, without degrading the quality of the found solutions.

Keywords. Renyi's entropy, PBIL, wireless network-on-chip (WiNoC), mapping, convergence time.

1 Introduction

Renyi's entropy [35], emerges as a generalization of the Shannon proposal and was defined by the mathematician Alfred Renyi in approximately 1950. His main aim was to find a suitable metric for measuring the amount of information by preserving simultaneously the additive property of statistically independent systems [35]. Renyi's entropy has been successfully used in several applications, such as medical imaging for capturing and analysis [17], heart neuropathy detectors [14], and the DNA sequence reconstruction process, starting from a given number of needed readings [18]. In the field of security and cryptography,Renyi's entropy has been used for breaking passwords [21], and it is also a common tool in financial applications [37], as well as penalty learning [32].

The population-based incremental learning or PBIL algorithm is a heuristic algorithm aimed at optimizing multi-objective problems. PBIL mixes concepts such as population, coming from genetic algorithms, with concepts related to incremental learning, which are usually present in neural networks. This combination generates an algorithm that is more efficient than any other population-based approach in terms of speed and accuracy [2, 36]. Specifically, relative to genetic algorithms, PBIL has shown improved performance for several optimization problems [6, 12, 27] by reducing the convergence time.

The PBIL applications range from molecular biology optimization [20] and fuel consumption reduction in hybrid power cells [22] to conflict resolution in supply chains [33]. PBIL has also been used in embedded system design optimization [11, 7, 5], where the aim was to perform the static mapping of tasks for network-on-chip (NoC) systems.

The PBIL algorithm works with a population of solutions and incremental learning. It usually represents such a population by means of an array

of probabilities of occurrence for each potential solution. In the adaptive version of the algorithm, the learning process is controlled by a parameter called the learning rate, which has a strong impact on the convergence speed of the algorithm. The learning rate parameter is adjusted based on the entropy estimates of the probability matrix at any given time. The most usual way to assess such entropy is the Shannon formulation [5].

NoC systems are composed of a set of processing elements and a communication architecture based on routers. This network approach appeared as a solution to scaling problems in high-performance embedded systems. NoC systems can better resolve latency issues when the number of processing elements grows or when compared to traditional connecting approaches, such as bus topologies. However, since it is expected that the number of processing elements will continue to grow, thanks to advances in the integration scales, the NoC approach might exhibit some drawbacks. As the number of processors increases, the communication becomes a complex problem due to both network loads and the number of messages, which finally degrades the latency [13].

The hierarchical NoC is a proposal to address larger sizes of nets and is based on communication levels (hierarchies), in order to reduce the communication latency [13, 15, 9]. It is quite common that at least one of these levels will be a wireless network, and for this reason, these systems are also referred to as wireless NoCs or WiNoCs. Several technologies have been proposed for the implementation of WiNoC systems [16, 25]. Some of these approaches use antennas, and others are implemented using waveguides, which can increase the transmission bandwidth up to tens of gigahertz.

This paper describes some variations of the adaptive PBIL algorithm, such as the use of different entropies for adjusting the learning rate (Shannon, Renyi one, and Renyi two). The main reason for such variations is to improve the convergence time of the algorithm. The PBIL approach is used to find task mapping solutions. The main difference with respect to previously reported approaches relies on the fact that this work performs mapping optimization over a WiNoC architecture, instead of a classical one (NoC). Another conspicuous difference is the relationship with the modelling of the mapping solutions, in which there is no overlapping of tasks for the same node.

The focus of this work is to improve the convergence (mapping) times, which means that the obtained quality for the whole set of techniques (genetic algorithms and PBIL with Shannon's or Renyi's entropy), may be equivalent. Such improved times may serve as a way to reduce the time to market of embedded designs and may open the way to dynamic mapping or real-time approaches.

The rest of the paper is organized as follows. Section 2 describes the background of the use of NoC and WiNoC as hardware platforms and the PBIL algorithm in the problem of the optimization of task mapping for applications in embedded systems. Section 3 talks about Renyi's entropy and its relationship with Shannon's entropy. Section 4 describes the adaptive PBIL algorithm, and in Section 5, the experimental results, analysis, and discussion are presented. Finally, Section 6 summarizes our main conclusions.

2 Background

The problem of optimizing resource allocation in embedded system is currently a challenge for designers because they face constraints of performance, energy consumption, variability of applications, reliability, real-time features, parallelism, and reduction of time in the prototyping stage [19].

NoC is a suitable hardware platform approach for dealing with the high complexity and the variability of applications for embedded system design. NoC provides adequate, cost-effective solutions and allows the synchronization of several complex functions [8, 30]. These networks are made up of a set of processing elements and a communication architecture, which enables improved performance when compared to communication buses. However, if the number of processing elements grows enough, latency issues

and high communication loads appear, reducing the system performance.

A proposed solution to this problem is the use of hierarchical architectures or wireless NoCs (WiNoCs), which are composed of several levels of communication with different speeds and connection technologies (wired and wireless). Such WiNoC architectures behave better regarding latency as the number of processing elements or nodes increases.

The latter is a consequence of the availability of several communication levels, which may reduce the effective distance between nodes relative to the more traditional NoC solutions [24, 25]. Given the plethora of constraints that designers must face, such as the planning and mapping of tasks, an optimization or automation tool is very useful to cope with the challenges in meeting optimization goals and design times [8, 7].

Several algorithms have been proposed for task mapping optimization, including exact, mathematical, and search-based (heuristic and systematic), [28, 26]. Population-based techniques appear to be the most suitable strategy, since a parallel search over the solution space is performed. Among such population-based approaches, PBIL stands as an appealing solution, with very promising convergence times [34, 7]. As stated previously, the focus of this work is to present a task mapping solution over a WiNoC based on the PBIL algorithm, which uses the entropy of Renyi, for the sake of improving the optimization time.

3 Renyi's Entropy

Renyi's entropy is a mathematical generalization of Shannon entropy, proposed by Alfred Renyi in the 1950s [23]. Renyi's entropy is defined as a ratio of the likelihood, as shown in equation (1):

$$H_\alpha(P) = \frac{1}{1-\alpha} log \sum_{i=1}^{n} P_i^\alpha, \alpha \neq 1 \wedge \alpha \geq 0, \quad (1)$$

where α indicates the order of the entropy. To compute the Shannon entropy, it is only necessary to derive equation (1) and calculate the limit when α tends to one. The results of deriving $H_\alpha(P)$ are presented in Equation (2):

$$H_1(P) = \lim_{\alpha \to 1} (H_\alpha(P))' = -\sum_{i=1}^{n} (P_i \times log P_i). \quad (2)$$

When α tends to infinity in equation (1), the entropy presents low values, as shown in equation (3). This entropy is defined as the Chebyshev entropy [35]:

$$H_\infty(P) = min_i(-log P_i) = -log(max_i P_i). \quad (3)$$

The maximum entropy value is obtained when α is equal to zero in equation (1). This is known as the Hartley entropy and is presented in equation (4):

$$H_0(P) = \frac{1}{1-\alpha} log \sum_{i=1}^{n} P_i^0 = log(n). \quad (4)$$

Another aspect that is worth mentioning when working with the entropies of Renyi and Shannon is the computational complexity of the sample. In the case of the Shannon entropy, the complexity grows close to linearly with the size of the alphabet or number of symbols, while for the Renyi entropy, the complexity has a sublinear growth.

The complexity of the sample to a discrete distribution P of R symbols for the Shannon case is $(R/log(R))$ samples, whereas in Renyi's entropy with $\alpha > 1$, the complexity is $(R^{1-\frac{1}{\alpha}})$ [1]. Therefore, when the number of symbols R increases, the number of samples for Shannon's entropy grows faster than that for Renyi's, as shown in Fig. 1.

As shown in Fig. 1, by increasing the number of symbols R, the number of samples needed to compute the entropy increases. However, the Shannon's growth rate is greater than that of the Renyi's.

It is also observed that the Renyi's entropies are below the linear growth, while the Shannon's is closer to linear. Such growth behaviour influences the convergence time.

Some population-based algorithm searches use such entropies, such as the case mentioned in [32], wherein the simulation results in a learning algorithm penalized with Renyi's entropy converge

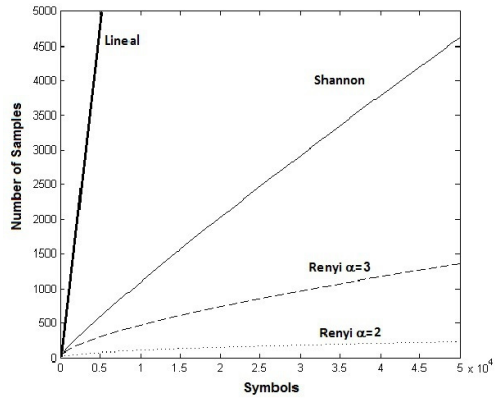

Fig. 1. Numbers of samples for Shannon's entropy and Renyi's entropy with $\alpha = 2$ and $\alpha = 3$

faster than when Shannon's entropy is used. Furthermore, the number of samples for Renyi's entropy with $\alpha = 2$ is less than with $\alpha = 3$. This is because Renyi with $\alpha = 2$ needs only ($R^{1/2}$) samples, whereas for $\alpha = 3$, more is required ($R^{2/3}$). The latter is reflected in Fig. 1, where the curve for $\alpha = 3$ grows faster than that for $\alpha = 2$.

4 Adaptive PBIL Algorithm

PBIL belongs to a group of transformative-heuristic algorithms that are based on population. It is based on stochastic estimates for searching across the solution space, avoiding local minimum issues. PBIL operates simultaneously with concepts such as population optimization and competitive learning. This population is represented as an array of probabilities of occurrence for each potential solution, which converge towards an optimum.

This research presents three modifications to the adaptive PBIL algorithm proposed in [5]. The first modification is the change in architecture, moving from a traditional NoC to a hierarchical wireless architecture (WiNoC), which has better performance when the number of nodes increases, as mentioned [24]. The second modification is related to the use of three entropies to adjust the speed of learning. In the proposed algorithm, three instances of entropy were considered: Shannon's entropy and the second- and third- order Renyi's entropy. The third change has a relationship with the modelling of the mapping solutions, in which there is no overlapping of tasks for the same node, within the constraints imposed by the application.

The proposed adaptive PBIL algorithm is depicted in Algorithm 1. The algorithm has three inputs: the application represented in a task graph (TG), the hardware platform of a WiNoC architecture represented by a graph (ArG), and a population represented by a probability array (P_m). The dimensions of P_m are related to the number of tasks to be allocated, which corresponds to the columns (N), and the amount of available processing resources, which corresponds to the rows (M). For the sake of giving an equal initial probability to the whole set of resources, each entry of matrix P_m is initialized to $1/M$, thus ensuring maximum population diversity.

Algorithm 1: New Adaptive PBIL algorithm

Input: Task Graph (TG), Architecture Graph (ArG), Probability Matrix (P_m) of $M \times N$ size.

Output: Optimized solution to the task mapping.

begin
  $P_m(i,j) = 1/M \; \forall \; 1 \leq i \leq M \; and \; 1 \leq j \leq N$;
  repeat
    repeat
      $Pop = Create\_Population\,(P_m)$;
      $Val = Validate(Pop)$;
    until *Val*;
    $Fitness = Evaluation\_Sorting\,(Pop)$;
    $Best = Selection\,(Pop, Fitness)$;
    $H = Entropy\,(P_m)$;
    $LR = Learning\_Rule\,(H)$;
    $P = Update\_Array\,(P_m, Best, LR)$;
  until $H \leq Tolerance$;
  return $Best$;
end

As shown in Algorithm 1, a population is created at the beginning of each iteration, starting from the probabilities in P_m, which was initialized

previously. Such a task is performed by the $Initialization\_Population$ routine. Then, the recently created population is validated to ensure that the solutions meet the constraints imposed by the task graph (TG). Such a validation is performed by the $Validate$ routine. The set of potential solutions just created is assessed by a fitness function, which is usually related to a multi-objective problem. For this work, three objectives of optimization were considered: performance, power consumption and bandwidth. The sum and normalization of these three figures of merit corresponds to the fitness value.

The solutions of the population are sorted according to their fitness. This task is performed by the $Evaluation\_Sorting$ routine. The $Selection$ routine is devoted to extracting the best solution found in the current iteration; this solution helps to update the probabilities of the array P_m.

The features of the best solution found so far are strengthened into the population by increasing the values of its corresponding probabilities. The function of the PBIL algorithm that is responsible for updating matrix P_m is called $Update\_Array$ in Algorithm 1. The updating increases those probabilities associated with the best solution and decreases the remaining values since the sum along each column in the P_m array must be equal to one at each iteration. The updating of the probabilities is shown in equation (5), which corresponds to a modified version based on the Hebbian learning rule [31]:

$$P(i,j)_U = \begin{cases} P(i,j)_O + \left[1 - P(i,j)_O\right] \cdot LR, & \text{if } j = k \\ \frac{\left[1-P(i,k)_U\right] \cdot P(i,j)_O}{1-P(i,j)_O}, & \text{otherwise.} \end{cases} \quad (5)$$

In equation (5), k represents the best solution obtained for a given attribute j, and the suffixes O and U denote the old and new probabilities, respectively.

The adaptive feature of the algorithm is achieved by changing the learning rate in a dynamic fashion. To do so, it is necessary to make an estimate of the current status of the convergence process. This goal is accomplished by using an entropy measure of the probability matrix. The value of entropy H is computed by the $Entropy$ routine and serves as a measure of the population diversity. Such a measure is used by the Learning Rule routine for adjusting the learning rate. Three different learning rules have been used in the adjustment of the Learning Rate, namely, Linear, Exponential, and Bell-Shaped. Table 1 summarizes the form of each rule.

Table 1. Learning rules for the adaptive PBIL algorithm

Name	Learning Rule
Linear	$LR = LR_{max} - [H_N \times (LR_{max} - LR_{min})]$
Exponential	$LR = LR_{min} + [e^{-4.5H_N} \times (LR_{max} - LR_{min})]$
Bell-Shaped	$LR = LR_{min} + [e^{-\frac{(H_N-3)^2}{2}} \times \frac{(LR_{max}-LR_{min})}{\sqrt{2\times\pi}}]$

In Table 1, LR_{max} and LR_{min} are the upper and lower limits for the learning rate, respectively. H_N refers to a normalized entropy, which may be calculated as the ratio of the entropy to the maximum possible value ($H_N = H/H_{max}$). The LR adjusted value is used by the Update routine for changing the probabilities in the P_m array.

Equations (6) and (**??**), describe the computation of the entropy using the Renyi and Shannon formulations, respectively, considering that the PBIL algorithm works with a probability array P_m of (M x N) size. Renyi's entropy (H_α) is computed according to its order, i.e., α. Equation (**??**) may be viewed as a special case of Renyi's entropy, and it leads to the Shannon formulation:

$$H_\alpha(P_m) = \frac{1}{1-\alpha} log_M \sum_{i=1}^{M}\sum_{j=1}^{N} P_m(i,j)^\alpha, \alpha > 1, \quad (6)$$

$$H_1(P_m) = \lim_{\alpha\to 1}(H_\alpha(P_m))' \quad (7)$$
$$= -\sum_{i=1}^{M}\sum_{j=1}^{N}(P_m(i,j) \times log_M P_m(i,j)).$$

The results described here for Renyi's entropy were obtained using orders of $\alpha = 2$, and $\alpha = 3$, as shown in equations (8) and (9), respectively:

$$H_2(P_m) = -log_M \sum_{i=1}^{M}\sum_{j=1}^{N} P_m(i,j)^2, \quad (8)$$

$$H_3(P_m) = -0.5log_M \sum_{i=1}^{M}\sum_{j=1}^{N} P_m(i,j)^3. \quad (9)$$

The algorithm stops when the value of the entropy is less than the tolerance value. This is when the probability matrix tends to focus on individual entries of each column of the P_m matrix, i.e., when, the algorithm achieves an optimal solution.

5 Results and Discussion

The proposed adaptive PBIL algorithm was written and tested in a Matlab tool (R2016), and two types of target applications to be mapped were tested. The first application tested is an MPEG2 video decoder with 12 tasks [29], and the second test corresponds to a synthetic application of 16 tasks selected from the database Power-Struggles [4]. Fig. 2 presents the task graphs for the two target applications.

As shown in Fig. 2, for each task graph, there are several tasks that could be executed at the same time. These applications were selected because they allow assessing one of the improvements proposed for the PBIL algorithm, which implies avoiding the overlapping of tasks on the same node. The technical specifications of the network are taken from [10].

For testing, a net of 16 nodes distributed in four 2D mesh subnets in the first level was used. Each subnet was interconnected to the second hierarchical level through wireless connections with a star topology. The nodes in each subnet are formed by a processor with different technological characteristics and a conventional router (R) for wired communication. The wireless communication in each subnet used a wireless router (WR) with four ports and two wired ports. Communication in the second level is handled by a wireless router (WR) that has 16 wireless ports, divided into groups of 4 for each subnet, Fig. 3 shows this communication architecture.

Fig. 3, shows the communication architecture for a WiNoC with 16 nodes, where wireless channels are dotted arrows and wired channels are continuous arrows. The WiNoC then has 16 channels of each of the wireless and wired types.

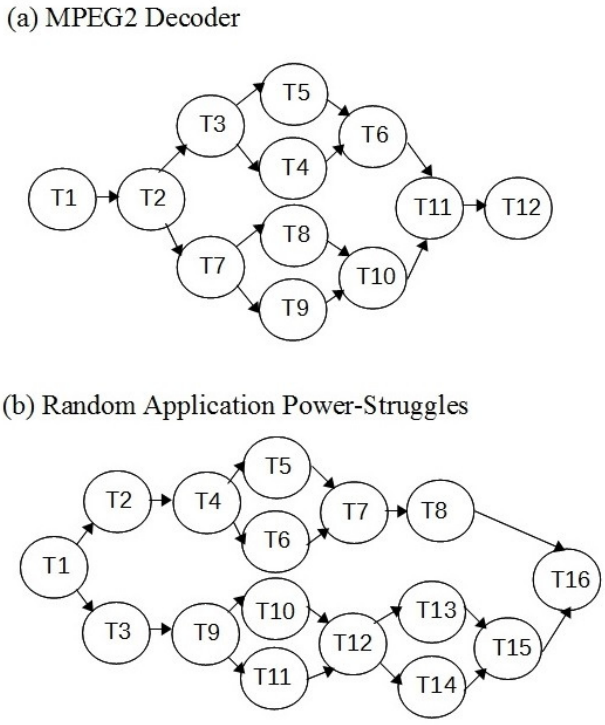

Fig. 2. Task graphs for the two target applications

Each wireless link has a transfer rate of 32 Gbps for a wireless bandwidth of 512 GHz, and each wired link is 64 bits wide with a frequency of 1 GHz, whereby it has a transfer rate of 64 Gbps.

The tests consisted of measuring the convergence times of the adaptive PBIL algorithm under different parameters. The first test measured the convergence time for each learning rule (linear, exponential and bell-shaped) using three different entropies (Shannon and Renyi with $\alpha = 2$ and $\alpha = 3$) over a total of 2000 runs of the algorithm, and the second measured the convergence time of the adaptive PBIL algorithm for each type of entropy using the different learning rules reported above.

The results of the convergence times of the adaptive PBIL algorithm for implementing the MPEG2 decoder are presented in Figs.4 and 5.The convergence times for a synthetic application based on Power-Struggles are presented in Figs.6

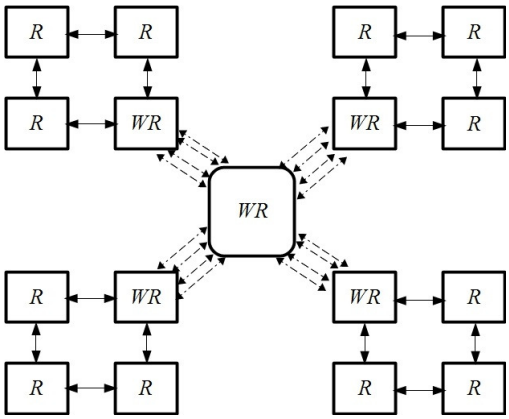

Fig. 3. Communication architecture for WiNoC

and 7. Figs. 8 and 9 depict the algorithm performance for the two proposed test applications.

Fig. 4 shows that for the three learning rules, the convergence time is always greater for the Shannon's entropy, followed by Renyi's entropy with $\alpha = 2$ and $\alpha = 3$. However, in some of the two thousand executions of the algorithm, there may be some counterexamples to the general behavior, due to the random nature of the PBIL algorithm. The convergence times for the three learning rules using each form of entropy are shown in Fig. 5.

Fig. 5 also shows that when the proposed adaptive PBIL algorithm is executed along with the bell learning rule, the convergence time is much greater than that for the exponential rule or linear rule. The average convergence time for this instance and the bell learning rule was two times that of the exponential rule and three times that of the linear case.

In Fig. 6, the results for the second target application are presented, consisting of a synthetic application composed of 16 tasks. The profiling time, power consumption and bandwidth of each task were extracted from the test bench named Power-Struggles [3].

As observed in Fig. 6, there is a consistent behaviour of the test performed for the MPEG-2 decoder. For both target applications, the convergence times are higher when Shannon's entropy is used. In Fig. 7, the convergence times for the three learning rules are shown.

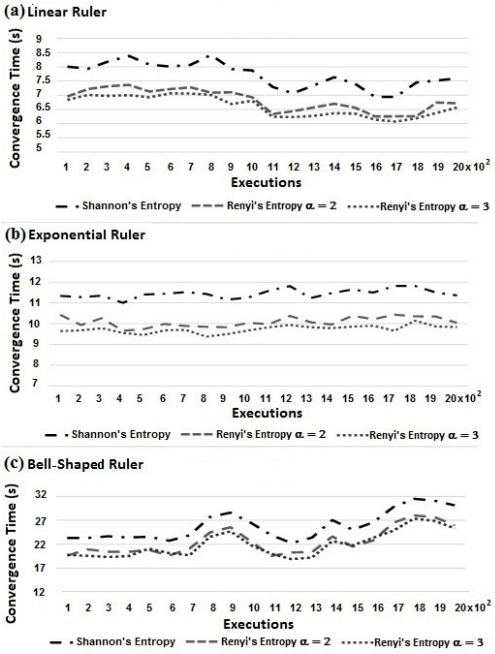

Fig. 4. Convergence times for adaptive PBIL in the application MPEG2 Decoder

As shown in Fig. 7, the convergence times of the PBIL algorithm when the bell rule is used are higher than those of the exponential and linear rules. The average convergence time for the bell learning rule is close to double those for the exponential and linear cases.

Fig. 8 compares the obtained performances when the proposed adaptive PBIL algorithm is executed using the linear rule Shannon's entropy and Renyi's entropy with each application. For this figure, the performance must be viewed as the quality of the found solutions.

As shown in Fig. 8, there is no significant performance difference in the proposed adaptive PBIL algorithm when Renyi's entropy or Shannon's entropy is used, i.e., there is no performance degradation when Renyi's entropy is used.

Fig. 9 shows a performance comparison for WiNoC and a conventional NoC architecture for the adaptive PBIL algorithm using the linear rule Shannon's entropy and Renyi's entropy. The number of nodes for both nets was 16. The topology for the NoC was a 2D mesh with a size

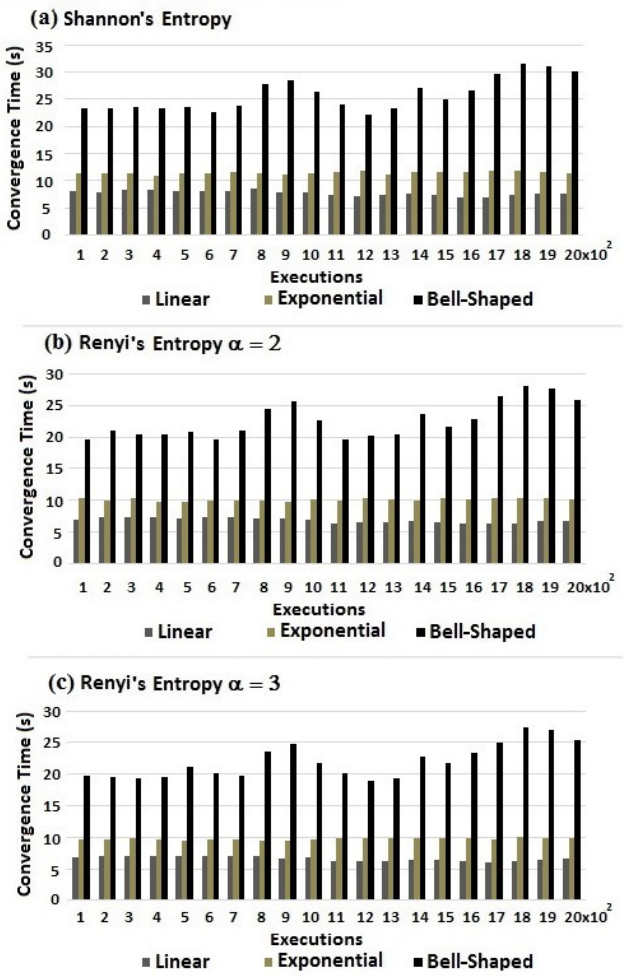

Fig. 5. Convergence times for adaptive PBIL in all three learning rules on the application MPEG2 Decoder

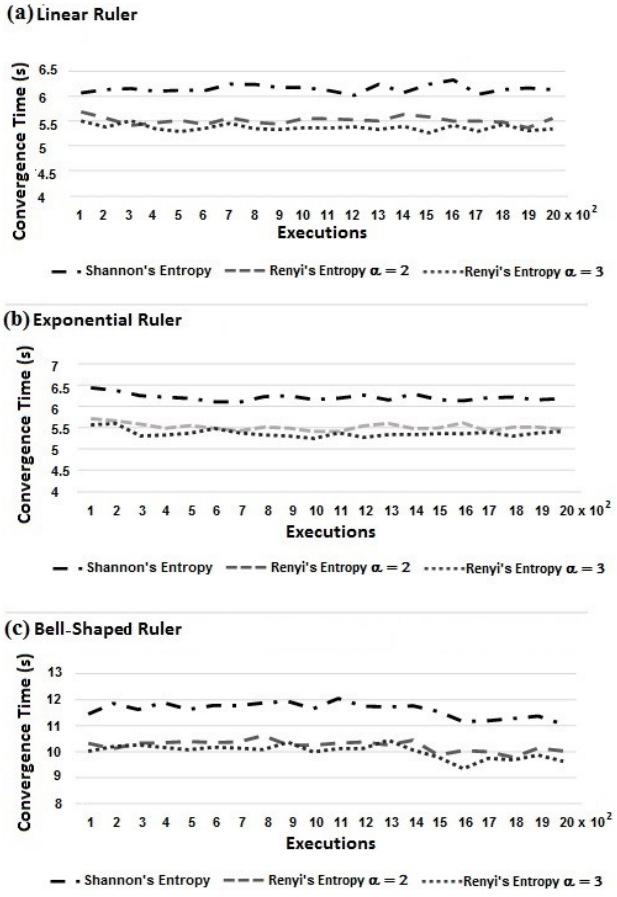

Fig. 6. Convergence times for adaptive PBIL in the synthetic application 16 Tasks Power-Struggles

of 4x4 and wired connections, a transfer rate of 64 bps, a frequency of 1 GHz and the basic routing algorithm XY.

As shown in Fig. 9, for the case of the three entropies, the WiNoC architecture improves the performance compared to the traditional NoC, with a performance improvement of 1.3% by MPEG and 1.1% by Power-Struggles. The performance of a mesh network is affected by the distance traveled by the package from source to destination.

The worst case of this distance corresponds to the diameter of the network (2 x ($n^{1/2}$-1)), where n is the number of nodes. For the NoC used in these tests, the maximum distance was six hops.

Another test was implemented by a network of 36 nodes, and for this case, the WiNoC architecture achieved an improved performance of 3.6% by MPEG and 3.0% by Power-Struggles compared to the traditional NoC.

Then, when the size of the network increases, the performance of the traditional NoC is affected. It is in these particular cases where the WiNoC network has better performance compared to the traditional NoC.

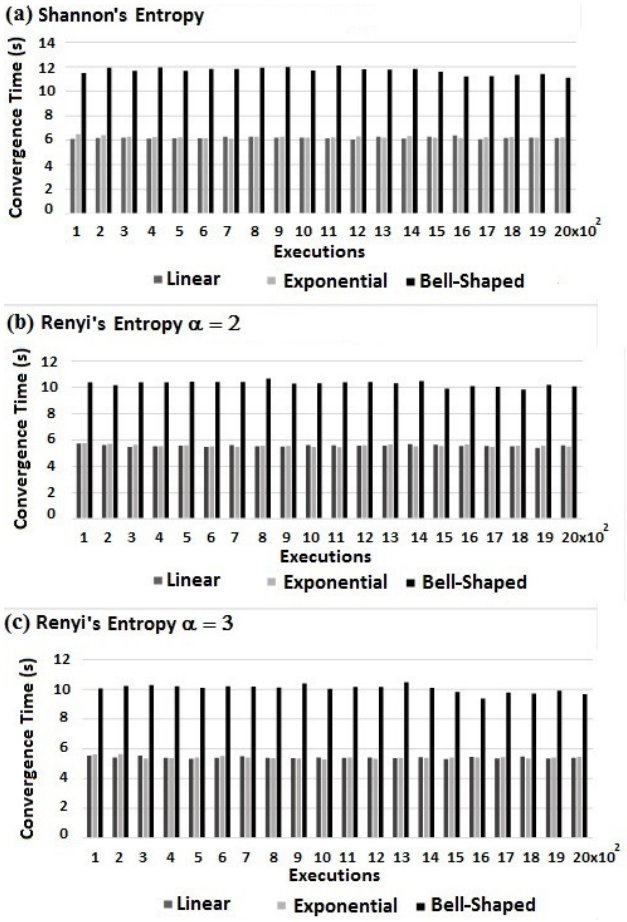

Fig. 7. Convergence times for adaptive PBIL in all three learning rules for the synthetic application 16 Tasks Power-Struggles

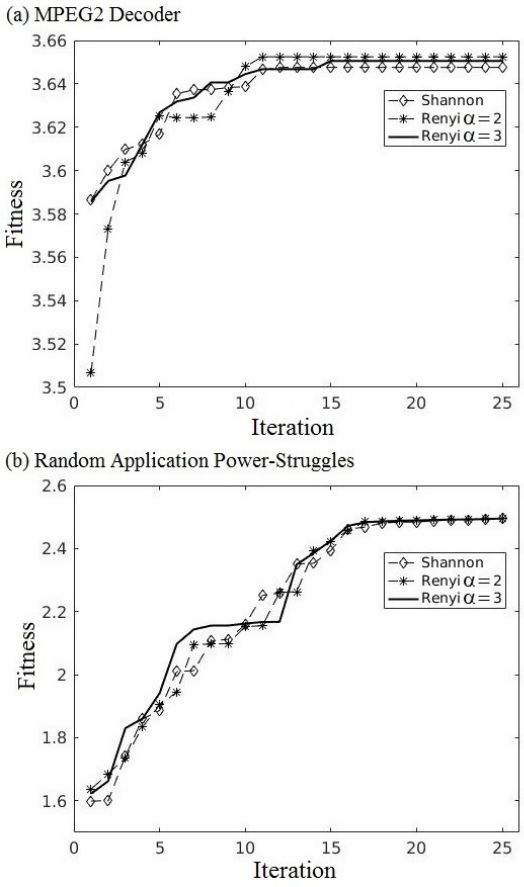

Fig. 8. Performance for adaptive PBIL algorithm using Renyi's and Shannon's entropy by WiNoC

6 Conclusion and Future Work

The convergence time for the adaptive PBIL algorithm when Renyi's entropy is used along with $\alpha = 2$ and $\alpha = 3$ are lower than those obtained when Shannon's entropy is used. This confirms using Renyi's entropy whit α greater than one, produces fewer samples than Shannon's entropy.

The convergence times for the adaptive PBIL algorithm using the bell learning rule are superior to the linear and exponential rules for the three cases of entropy. From these tests, it can be concluded that the best convergence times for the PBIL algorithm are obtained using a linear learning rule, combined with an estimate of Renyi's entropy with $\alpha = 3$ or $\alpha = 2$.

Although the convergence time is lower when Renyi's entropy is used, there is no degradation in the performance caused by the proposed adaptive PBIL algorithm.

The performance improves when the WiNoC architecture is used rather that the traditional NoC. This is because in WiNoC, the distance between a source and destination node could decrease, thanks to the combination of wired and wireless links, which impacts the performance positively.

The results obtained show that the proposed PBIL technique has good performance when used in task mapping off-line. The next step is to

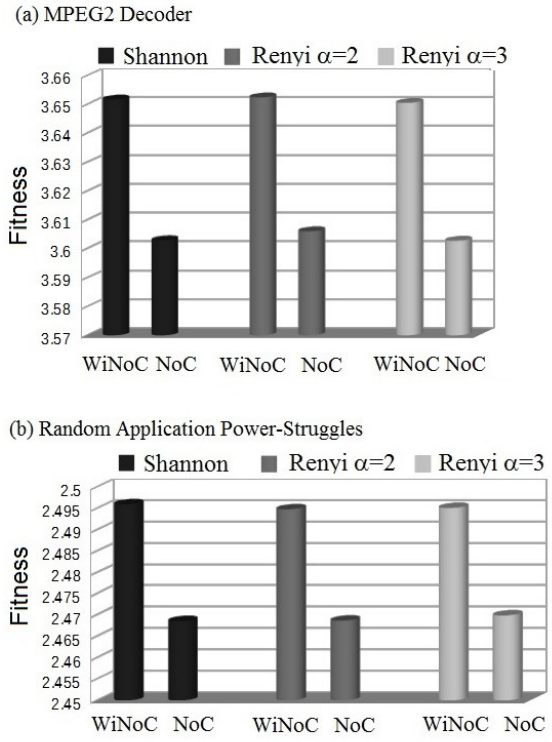

Fig. 9. Performance for adaptive PBIL algorithm using Renyi's and Shannon's entropy by WiNoC and NoC

implement the PBIL technique for task mapping on-line (i.e., at the execution time).

Acknowledgements

The authors would like to thank Pontificia Universidad Javeriana Cali, Universidad Nacional de Colombia, and Universidad del Valle for their support in the development of the current project.

References

1. **Acharya, J., Orlitsky, A., Suresh, A. T., & Tyagi, H. (2015).** The Complexity of Estimating Rényi Entropy. *Proceedings of the 26th Annual ACM-SIAM Symposium on Discrete Algorithms.*, pp. 1855–1869.
2. **Baluja, S. (1994).** Population-Based Incremental Learning : A Method for Integrating Genetic Search Based Function Optimization and Competitive Learning. Technical report, Carnegie Mellon University, Pittsburgh, Pennsylvania.
3. **Blem, E., Menon, J., & Sankaralingam, K. (2013).** Data to accompany a detailed analysis of contemporary arm and x86 architectures. available at www.cs.wisc.edu/vertical/isa-power-struggles.
4. **Blem, E., Menon, J., & Sankaralingam, K. (2013).** Power struggles: Revisiting the RISC vs. CISC debate. *IEEE 19th International Symposium on High Performance Computer Architecture (HPCA2013)*, pp. 1–12.
5. **Bolaños, F., Aedo, J., & Rivera, F. (2014).** Static and dynamic task mapping onto network on chip multiprocessors. *Dyna*, Vol. 81, No. 185, pp. 28–35.
6. **Bolanos, F., Aedo, J. E., & Rivera, F. (2012).** Comparison of Learning Rules for Adaptive Population-Based Incremental Learning Algorithms. *International Conference on Artificial Intelligence, ICAI*, pp. 1–8.
7. **Bolanos, F., Aedo, J. E., Rivera, F., & Bagherzadeh, N. (2012).** Mapping and Scheduling in Heterogeneous NoC through Population-Based Incremental Learning. *Journal of Universal Computer Science*, Vol. 18, No. 7, pp. 901–916.
8. **Chen, S.-J., Lan, Y.-C., Tsai, W.-C., & Hu, Y.-H. (2012).** *Reconfigurable Networks-on-Chip*. Springer US.
9. **Dehghani, A. & Jamshidi, K. (2015).** A fault-tolerant hierarchical hybrid mesh-based wireless network-on-chip architecture for multicore. *The Journal of Supercomputing*, pp. 3116–3148.
10. **Ditomaso, D. & Kodi, A. (2015).** A-WiNoC: Adaptive wireless network-on-chip architecture for chip multiprocessors. *IEEE Transactions on Parallel and Distributed Systems*, Vol. 26, No. 12, pp. 3289–3302.
11. **Fan, L. J., Li, B., Zhuang, Z. Q., & Fu, Z. Q. (2007).** An approach for dynamic hardware /software partitioning based on DPBIL. *Proceedings - Third International Conference on Natural Computation, ICNC 2007*, Vol. 5, No. Icnc, pp. 581–585.
12. **Gosling, T. & Tsang, P. E. (2004).** Population Based Incremental Learning Versus Genetic Algorithms : Iterated Prisoners Dilemma. Technical report, University of Essex, England.

13. **Guerre, A., Ventroux, N., David, R., & Merigot, A. (2010).** Hierarchical network-on-chip for embedded many-core architectures. *NOCS 2010 - The 4th ACM/IEEE International Symposium on Networks-on-Chip*, pp. 189–196.
14. **Jelinek, H. F., Cornforth, D. J., Tarvainen, M. P., & Miloevic, N. T. (2015).** Multiscale Renyi Entropy and Cardiac Autonomic Neuropathy. *2015 20th International Conference on Control Systems and Computer Science*, pp. 545–547.
15. **Lankes, A., Wild, T., & Herkersdorf, A. (2009).** Hierarchical NoCs for optimized access to shared memory and IO resources. *12th Euromicro Conference on Digital System Design: Architectures, Methods and Tools, DSD 2009*, pp. 255–262.
16. **Li, X. (2012).** *Survey of Wireless Network-on-Chip Systems*. Ph.D. thesis, Auburn University.
17. **Marsh, J. N., Wallace, K. D., McCarthy, J. E., Wickerhauser, M. V., Maurizi, B. N., Lanza, G. M., Wickline, S. A., & Hughes, M. S. (2010).** Application of a real-time, calculable limiting form of the Renyi entropy for molecular imaging of tumors. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, Vol. 57, No. 8, pp. 1890–1895.
18. **Motahari, A., Bresler, G., & Tse, D. (2013).** Information Theory of DNA Shotgun Sequencing. *IEEE Transaction on Information Theory*, Vol. 59, No. 10, pp. 1–33.
19. **Munir, A., Gordon-Ross, A., & Ranka, S. (2016).** High-Performance Energy-Efficient Multi-core-Based Parallel Embedded Computing*. In *Modeling and Optimization of Parallel and Distributed Embedded Systems*, chapter 7: High-Pe. John Wiley & Sons, Ltd, Chichester, UK., pp. 159–190.
20. **Palafox, L. (2012).** Gene Regulatory Network Reverse Engineering using Population Based Incremental Learning and K-means. *Proceedings of the 14th annual conference companion on Genetic and Evolutionary Computation, GECCO'12.*, pp. 1423–1424.
21. **Pfister, C. E. & Sullivan, W. G. (2004).** Renyi entropy, guesswork moments, and large deviations. *IEEE Transactions on Information Theory*, Vol. 50, No. 11, pp. 2794–2800.
22. **Ramos-Paja, C. A., Bolanos, F., Gonzalez, D., Ramirez, F., & Camarillo, J. R. (2015).** Reducing the Fuel Consumption of Hybrid Fuel Cell/Photovoltaic Power Systems Using PBIL-Based Reconfiguration. *2015 Asia-Pacific Conference on Computer Aided System Engineering*, pp. 90–95.
23. **Rényi, a. (1961).** On measures of entropy and information. *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability*, Vol. 1, No. c, pp. 547–561.
24. **Rezaei, A., Daneshtalab, M., Safaei, F., & Zhao, D. (2016).** Hierarchical approach for hybrid wireless network-on-chip in manycore era. *Computers and Electrical Engineering*, Vol. 51, pp. 225–234.
25. **Rezaei, A., Safaei, F., Daneshtalab, M., & Tenhunen, H. (2014).** HiWA: A hierarchical wireless network-on-chip architecture. *International Conference on High Performance Computing & Simulation, HPCS*, pp. 499–505.
26. **Sacanamboy, M., Bolaños, F., & Nieto, R. (2014).** A Primer for Mapping Techniques on NoC Systems. *ESA (2014) Proceedings 12th International Conference on Embedded Systems and Applications*, pp. 70–75.
27. **Sacanamboy, M., Quesada, L., Bolanos, F., Bernal, A., & O'Sullivan, B. (2016).** A comparison between two optimisation alternatives for mapping in wireless network on chip. *2016 IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 938–945.
28. **Sahu, P. K. & Chattopadhyay, S. (2013).** A survey on application mapping strategies for Network-on-Chip design. *Journal of Systems Architecture*, Vol. 59, No. 1, pp. 60–76.
29. **Thiele, L., Bacivarov, I., Haid, W., & Huang, K. (2007).** Mapping applications to tiled multiprocessor embedded systems. *Proceedings of 7th International Conference on Application of Concurrency to System Design, ACSD*, pp. 29–40.
30. **Tosun, S., Ozturk, O., Ozkan, E., & Ozen, M. (2015).** Application mapping algorithms for mesh-based network-on-chip architectures. *The Journal of Supercomputing*, Vol. 71, No. 3, pp. 995–1017.
31. **White, R. H. (1992).** Competitive hebbian learning: Algorithm and demonstrations. *Neural Networks*, Vol. 5, No. 2, pp. 261 – 275.
32. **Wu, J. & Ma, J. (2008).** Renyi entropy penalized learning algorithm for Gaussian mixture with automated model selection. *International Conference on Signal Processing Proceedings, ICSP*, pp. 1561–1564.

33. Wu, Y. Y., Li, J., & Jiang, G. R. (2014). Argument-based negotiation strategy based on adaptive PBIL for resolving conflicts in supply chain collaboration. *International Conference on Management Science and Engineering - Annual Conference Proceedings*, pp. 315–320.

34. Xing, H. & Qu, R. (2011). A population based incremental learning for network coding resources minimization. *IEEE Communication Letters*, Vol. 15, No. 7, pp. 698–700.

35. Xu, D. & Erdogmuns, D. (2010). Renyi's entropy, divergence and their nonparametric estimators. In *Information Theoretic Learning: Renyi's Entropy and Kernel Perspectives*. Springer New York, New York, *Kernel Perspectives*. Springer New York, New York, NY, pp. 47–102.

36. Yang, S. & Yao, X. (2005). Experimental Study on Population-Based Incremental Learning Algorithms for Dynamic Optimization Problems. *Soft Computing*, Vol. 9, No. 11, pp. 815–834.

37. Zhao, G.-h. & Hao, M. (2009). Incremental learning algorithm of least squares support vector machines based on Renyi entropy. *2009 International Conference on Management Science and Engineering, ICMSE*, pp. 95–100.

Article received on 05/03/2017; accepted on 10/11/2017.
Corresponding author is Maribell Sacanamboy.