An Energy-aware Hybrid Particle Swarm Optimization Algorithm for Spiking Neural Network Mapping

Junxiu Liu[1], Xingyue Huang[1], Yuling Luo[1]✉, and Yi Cao[2]

[1] Guangxi Key Lab of Multi-source Information Mining & Security,
Faculty of Electronic Engineering, Guangxi Normal University, Guilin, China, 541004
[2] Department of Business Transformation and Sustainable Enterprise,
Surrey Business School, University of Surrey, Surrey, UK, GU2 7XH
yuling0616@mailbox.gxnu.edu.cn

Abstract. Recent approaches to improving the scalability of Spiking Neural Networks (SNNs) have looked to use custom architectures to implement and interconnect the neurons in the hardware. The Networks-on-Chip (NoC) interconnection strategy has been used for the hardware SNNs and has achieved a good performance. However, the mapping between a SNN and the NoC system becomes one of the most urgent challenges. In this paper, an energy-aware hybrid Particle Swarm Optimization (PSO) algorithm for SNN mapping is proposed, which combines the basic PSO and Genetic Algorithm (GA). A Star-Subnet-Based-2D Mesh (2D-SSBM) NoC system is used for the testing. Results show that the proposed hybrid PSO algorithm can avoid the premature convergence to local optimum, and effectively reduce the energy consumption of the hardware NoC systems.

Keywords: Particle Swarm Algorithm, Genetic Algorithm, Spiking Neural Networks, Networks-on-Chip

1 Introduction

As an artificial neural network with the biological details, Spiking Neural Network (SNN) emulates information processing and communication capabilities of the mammalian brain [1]. Due to its biological properties and computational power, researchers aim to investigate custom hardware architectures to simulate information processing mechanisms of mammalian brain [2, 3]. Recently, researchers used the Networks-on-Chip (NoC) strategy to interconnect the large-scale hardware SNNs [1,4,5]. A typical NoC architecture includes computing and communication subsystems. The computing subsystem consists of a number of processing elements (PEs). The communication subsystem is composed of the routers and the channels, which are responsible for the communication between the PEs [6]. For realizing SNNs, PEs are used to implement the basic functions of spike neurons. In order to achieve this, spike neurons should be assigned to the appropriate PEs. This process of assignment is defined as SNN mapping. The correspondence between the SNN neurons and the NoC PEs is a one-to-one correspondence, thus the mapping problem of the SNN belongs to a typical quadratic assignment problem. The heuristic algorithms, e.g. ant colony optimization [7], Genetic Algorithm (GA) [8] and simulated annealing [9] etc., can be used to solve the quadratic assignment problems. Compared with Ant Colony Optimization and GA, particle swarm optimization (PSO) [10,11] does not have complex parameters, which is easy for the implementation. Thus the PSO has been applied to the NoC mapping [12,13]. In this paper, the PSO is applied to solve the problem of SNN mapping. Meanwhile, in order to avoid the fast convergence and local optimum, this approach combines the mutation operations of GA. For the SNN hardware systems, energy consumption affects the system performance, e.g. high energy consumption reduces the lifetime of the system and also affect the reliability. Therefore, this paper proposes an energy-aware mapping algorithm for the hardware SNN. The rest is organized as follows. Section 2 presents the basic concepts and definitions of SNN mapping and Section 3 presents the proposed hybrid PSO algorithm. Section 4 reports the experimental results and performance analysis. Section 5 provides a summary.

2 SNN Mapping Problem

This section introduces the SNN mapping problem, which includes the basic concepts and definitions of SNN mapping. In this paper, the SNN mapping aims to assign the spiking neurons to the hardware NoC for specific applications under the optimization rules. The target is normally to minimize the cost, e.g., energy consumption or latency, to allow the SNN to achieve a good performance under the mapping. In this approach, the target is to minimize the energy consumption of the NoC system. It is assumed that one neuron in the SNN corresponds to one PE in the NoC, i.e. the function of each neuron can be designed using one PE. This process can be described by Fig. 1, where the neurons of the SNN are mapped to the hardware NoC system. A SNN communication graph (SNNCG) and a NoC architecture graph (NoCAG) [14] are employed for the mapping.

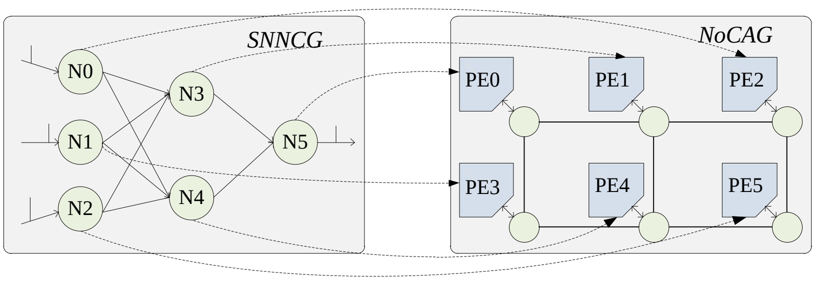

Fig. 1. SNN mapping process.

Definition 1: In the directed graph SNNCG (N, E), each vertex $n_i \in N$ represents a neuron, and each edge $e_{ij} \in E$ represents the communication path from the neuron n_i to n_j. The value t_{ij} of each edge e_{ij} represents the communication traffic between the neuron n_i to n_j, and the L_{ij} represents the maximum communication delay from the neuron n_i to n_j.

Equation (1) gives an example of communication matrix $T = [t_{ij}]$ between n neurons ($0 \le i \le n-1, 0 \le j \le n-1$). If there is no communication between the neuron n_i and n_j, then $t_{ij} = 0$.

$$T = \begin{bmatrix} 0 & t_{01} & t_{02} & \cdots & t_{0,n-1} \\ t_{10} & 0 & t_{12} & \cdots & t_{1,n-1} \\ t_{20} & t_{21} & 0 & \cdots & t_{2,n-1} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ t_{n-1,0} & t_{n-1,1} & t_{n-1,2} & \cdots & 0 \end{bmatrix} \tag{1}$$

Definition 2: In the directed graph NoCAG (V, P), each vertex $v_i \in V$ denotes a PE on the NoC. The edge $p_{ij} \in P$ denotes the communication path between the PE v_i and v_j. The f_{ij} on the edge p_{ij} represents the communication traffic from the PE v_i to v_j. The B_{ij} is the maximum bandwidth that the path p_{ij} can provide. The h_{ij} represents the distance from v_i to v_j, i.e. the number of hops in this approach.

In summary, given the SNNCG (N, E) and NoCAG (V, P), the SNN mapping problem is defined by

$$Mapping : N \to V \Rightarrow v_j = map(n_j) \in N, \exists v_j \in V. \tag{2}$$

The next section will introduce the details of the hybrid PSO algorithm for the SNN mapping problems.

3 Hybrid PSO Algorithm

3.1 Objective Function

According to the NoC communication energy model given in the approach of [15] and combined with the definition of Section 2, the energy consumption of 1 bit data from PE v_i to v_j is defined as

$$E_{ij} = (h_{ij} + 1) \times Es + h_{ij} \times E_l, \tag{3}$$

where E_s and E_l are the energy consumption of 1 bit transmission through the router and the adjacent channels, respectively. Then the NoC system communication energy E is given by

$$\begin{aligned} E &= \sum_{0 \le i \le n-1}^{0 \le j \le n-1} [(h_{ij} + 1) \times E_s + h_{ij} \times E_l] \times t_{ij} \\ &= \sum_{0 \le i \le n-1}^{0 \le j \le n-1} t_{ij} \times h_{ij} \times (E_s + E_l) + \sum_{0 \le i \le n-1}^{0 \le j \le n-1} E_s \times t_{ij}. \end{aligned} \tag{4}$$

It can be seen that except $\sum_{0\leq i\leq n-1}^{0\leq j\leq n-1} t_{ij} \times h_{ij}$, the rest are constants. Thus the hybrid PSO algorithm objective function can be defined as

$$\min\{\sum_{0\leq i\leq n-1}^{0\leq j\leq n-1} t_{ij} \times h_{ij}\}, \tag{5}$$

s.t.

$$\forall n_i \in N \Rightarrow map(n_i) \in V \tag{6}$$

$$\forall n_i \neq n_j \Rightarrow map(n_i) \neq map(n_j) \tag{7}$$

$$\text{size(SNNCG)} \leq \text{size(NoCAG)} \tag{8}$$

$$\forall t_{ij} \leq B_{ij} \tag{9}$$

It can be seen that equations (6) and (7) ensure that the neuron and the PE satisfy the one-to-one mapping requirement, and (8) and (9) ensure that the network size and bandwidth meet the requirements. The optimization goal of energy consumption is to minimize the sum of weighted distance between PEs.

3.2 Hybrid PSO

In the PSO, a solution for the optimization problem is a particle. Each particle has its own position, velocity, and a fitness value. A particle swarm consists of many particles. For each iteration, there is an optimal particle which has the best fitness. Other particles memorize and follow this optimal particle. Therefore, iteration process of the PSO is not completely random. According to the optimal particles at each iteration, the PSO can find the best solution using the algorithm update rule. In this paper, in order to apply PSO to the SNN mapping problem, several aspects should be considered, including a). the particle position representation, b). algorithm update rule, and c). the velocity representation. They are discussed as follows.

(a). ***Particle position representation***. The particles are expressed by the D-dimensional position vector $X = (x_0, x_1, x_2, \cdots, x_{n-1})$, where x_i denotes the neuron number in the SNN and $0 \leq i \leq n-1$, n is the total number of neurons, the index i represents the PE where the neuron is to be placed in NoC. For example, the particle $X = (2, 1, 3, 4, 0)$ indicate that there is 5 neurons in the SNN, neuron #2 is placed on the first PE of NoC, neuron #1 is placed on the second PE, and same for the others.

(b). ***Algorithm update rule***. The traditional PSO is not suitable for optimizations in the discrete space, e.g. the mapping problems [16]. In order to overcome this drawback, this paper updates the positions of particles by "jump" operations. That is, for a multidimensional particle, each update makes at least one dimension equal to one dimension of global optimal particle. Fig. 2(a)

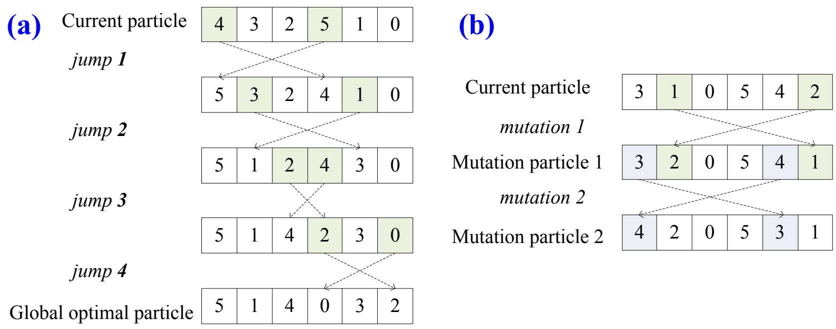

Fig. 2. Particle update process (a) and mutation operations (b).

shows an example. During the iterations of the hybrid PSO, the algorithm firstly selects the global optimal particle $X_g = (5,1,4,0,3,2)$, then the particle $X_p = (4,3,2,5,1,0)$ follows it to update. Each update makes at least one dimension to be the same as one dimension of global optimal particle. After four iterations, the current particle X_p is the same as the particle X_g.

(c). ***Velocity representation***. The velocity update rule changes the particle position in the traditional PSO where the velocity changes with the iterations. However, the hybrid PSO does not require the specific velocity update rule. Fig. 2(a) shows that in this approach two dimensions will change after each jump. Therefore, the velocity is defined as 2.

The basic PSO has the disadvantages of being convergent too quickly and falling into local optimum easily. In order to overcome them, this paper defines the concept of similarity of particle swarm and makes the hybrid PSO combined with mutation operations of GA. When the similarity of particle swarm exceeds the threshold, all particles are mutated except the global optimal particle. The mutation operation can timely and effectively reduce the similarity of particle swarm, which enables the hybrid PSO to have the ability to search for global optimal solution.

The similarity of particle swarm (S_{ps}) can be calculated by

$$S_{ps} = \frac{\sum_{i=0}^{N-2} S_{p^i}}{N-1}, \tag{10}$$

where N denotes the total number of particles and S_{p^i} denotes the individual similarity of particle i which is given by

$$S_{p^i} = \frac{S_{dims}}{N_{dims}}, \tag{11}$$

where S_{dims} denotes the number of dimensions that the particle i is the same as the global optimal particle, N_{dims} denotes the number of total dimensions of particles.

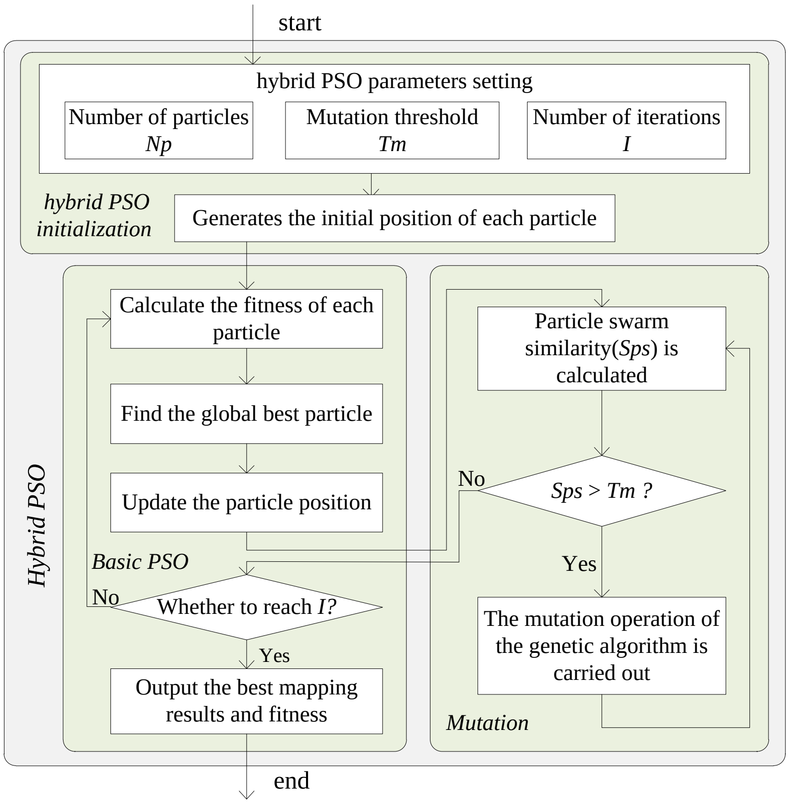

Fig. 3. The hybrid PSO algorithm.

When the similarity of particle swarm exceeds the threshold, each particle randomly swaps its own two dimensions until the similarity of particle swarm below the threshold. This process is the mutation process. Fig. 2(b) is an example. When the similarity of particle swarm exceeds the threshold, the particle $X_c = (3, 1, 0, 5, 4, 2)$ has two mutation operations, and becomes $X_{m2} = (4, 2, 0, 5, 3, 1)$ eventually whose similarity is below the threshold. This process also applies to other particles in the particle swarm.

In the SNN mapping process, neurons and PEs are one-to-one correspondence. However, the number of neurons (N_n) and PEs (N_{pe}) is not always equal. If $N_n < N_{pe}$, some virtual neurons can be added to make them equal. But they are unused (i.e. only for solving the mapping) and can be removed after the hybrid PSO completes.

3.3 The Hybrid PSO Running Process

Fig. 3 describes the hybrid PSO running process, which includes three processes:

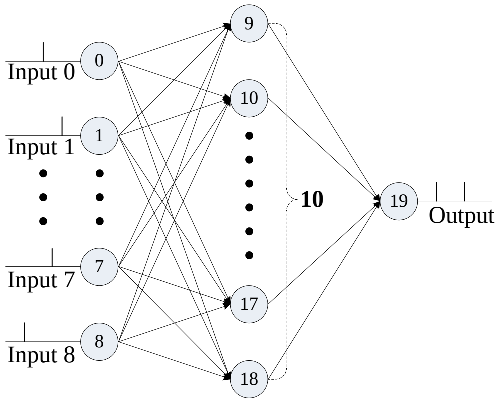

Fig. 4. Feedforward SNN architecture.

(a). ***The hybrid PSO initialization***. The main function of this process is to set the parameters of the hybrid PSO, and to add virtual neurons to let N_n equal to N_{pe} (if needed). The initial particle swarm is generated in this process.

(b). ***The basic PSO operation***. The basic operations of PSO are completed in this process. The PSO iterates and searches for the best mapping solution.

(c). ***The particle mutation operation***. The main function of this process is to avoid rapid convergence and reduce the possibility of falling into the local optimum, which enable the hybrid PSO to have the ability to search for global optimal solution.

4 Experimental Results

This section provides the experimental results for the hybrid PSO. In this paper, the SNN mapping algorithm is implemented in C++. The SNN used in this paper is shown in Fig. 4, which is a feedforward network and has an input layer, a hidden layer, and an output layer. The traffic of each communication path in the SNN is set to be the same in this experiment.

The hybrid PSO algorithm is validated on a Star-Subnet-Based-2D Mesh (2D-SSBM) NoC architecture which is based on our previous works [2, 17], see Fig. 5. The 2D-SSBM NoC architecture has two hierarchical levels, i.e. the top layer is 2D mesh and the bottom is a star. For the local communication in the star topology, the packet can be transmitted via the node router, e.g. the path between the source node #14 to the destination node #8 is shown by the dash brown in Fig. 5. The tile router is employed to forward the global data transmission and it uses the XY routing algorithm, see the dash blue as an example.

The parameters of this experiment are shown in Table. 1. In order to evaluate the hybrid PSO algorithm, the basic PSO and random mapping method are employed for benchmarking. The basic PSO does not have the mutation op-

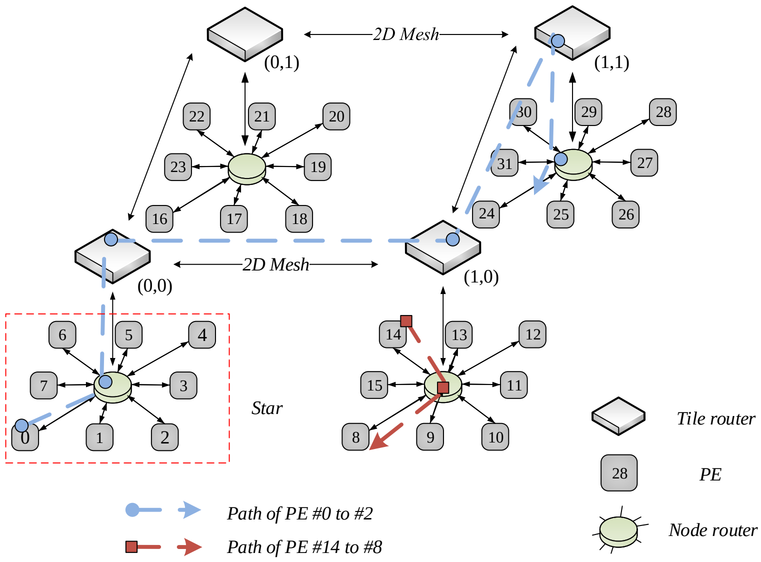

Fig. 5. Star-Subnet-Based-2D Mesh NoC architecture.

erations. For the random mapping method, the average fitness of the random mapping of 30 particles is taken as the result.

The experimental results are shown in Fig. 6. In the Fig. 6(a), the fitness is the energy consumption of the NoC. It shows that the best fitness of basic PSO does not change after 24 iterations, and it falls into the local optimum. However, the proposed hybrid PSO finds the global optimization after a number of iterations (i.e. 130 in this experiment) where its fitness (i.e. energy consumption of the hardware NoC) is further reduced. Fig. 6(b) shows that the particle swarm similarity of the basic PSO reaches 1 and particle swarm convergence completes early (i.e. at the iteration #46). However by using the hybrid PSO, the similarity of the particle swarm is always around the threshold which is due to the thresholding and the mutation operations. Therefore the hybrid PSO is capable to search the global optimal solution.

5 Conclusions

An energy-aware hybrid PSO algorithm for SNN mapping is proposed in this paper. The relationship between energy consumption and communication path is analysed. The proposed algorithm is able to search for the optimal solution through the basic PSO. In the meantime, the mutation operations via the GA is employed to avoid the premature convergence. Experimental results show that the proposed hybrid PSO algorithm can avoid the local optimum and achieve a lower energy consumption compared to the benchmarking algorithms. Future works include the optimization and further improvement of the proposed algorithm.

Table 1. Experiment parameter setting

	Parameters	Basic PSO	Hybrid PSO	Random
2D-SSBM NoC	2D mesh size	3×3		
	Number of nodes in the star subnetwork	4		
	Total number of nodes	36		
Feedforward SNN	Total number of neurons	20		
	Number of layers	3		
	Number of communication paths	100		
The hybrid PSO	Number of Particles	30		
	Mutation threshold	N/A	0.5	N/A
	Total iterations	200	200	N/A

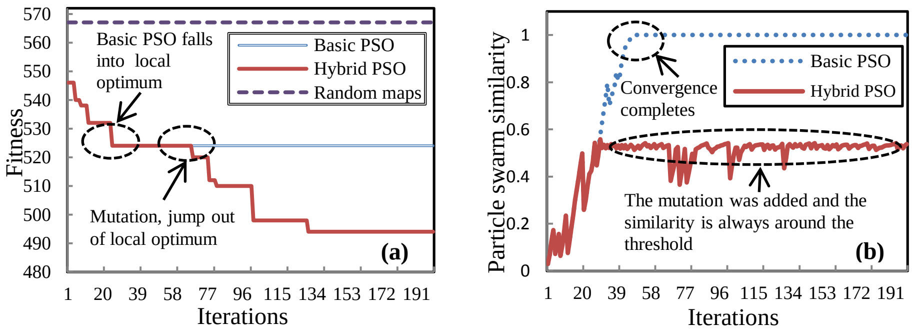

Fig. 6. The change of Fitness (a) and Particle swarm similarity (b) with iterations.

Acknowledgments. This research was supported by the National Natural Science Foundation of China under Grants 61603104 and 61661008, the Guangxi Natural Science Foundation under Grant 2015GXNSFBA139256 and 2016GXNSFCA380017, the funding of Overseas 100 Talents Program of Guangxi Higher Education, the Research Project of Guangxi University of China under Grant KY2016YB059, Guangxi Key Lab of Multi-source Information Mining & Security under Grant MIMS15-07, and the Doctoral Research Foundation of Guangxi Normal University.

References

1. Liu, J., Harkin, J., Maguire, L.P., McDaid, L.J., Wade, J.J., Martin, G.: Scalable Networks-on-Chip interconnected architecture for astrocyte-neuron networks. IEEE Transactions on Circuits and Systems I: Regular Papers 63(12), 2290–2303 (2016)
2. Liu, J., Harkin, J., Maguire, L.P., Mcdaid, L.J., Wade, J.J.: SPANNER: A self-repairing spiking neural network hardware architecture. IEEE Transactions on Neural Networks and Learning Systems pp. 1–14 (2017)

3. Akopyan, F., Sawada, J., Cassidy, A., Alvarez-Icaza, R., Arthur, J., Merolla, P., Imam, N., Nakamura, Y., Datta, P., Nam, G.J., Taba, B., Beakes, M., Brezzo, B., Kuang, J.B., Manohar, R., Risk, W.P., Jackson, B., Modha, D.S.: TrueNorth: Design and tool flow of a 65 mW 1 million neuron programmable neurosynaptic chip. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 34(10), 1537–1557 (2015)
4. Carrillo, S., Harkin, J., McDaid, L.J., Morgan, F., Pande, S., Cawley, S., McGinley, B.: Scalable hierarchical network-on-chip architecture for spiking neural network hardware implementations. IEEE Transactions on Parallel and Distributed Systems 24(12), 2451–2461 (2013)
5. Firuzan, A., Modarressi, M., Daneshtalab, M.: Reconfigurable communication fabric for efficient implementation of neural networks. In: International Symposium on Reconfigurable and Communication-centric Systems-on-Chip (ReCoSoC). pp. 1–8 (2015)
6. Benini, L., De Micheli, G.: Networks on chips: a new SoC paradigm. Computer 35(1), 70–78 (2002)
7. Gambardella, L.M., Taillard, É., Dorigo, M.: Ant colonies for the quadratic assignment problem. The Journal of the Operational Research Society 50(2), 167–176 (1999)
8. Misevicius, A.: An improved hybrid genetic algorithm: new results for the quadratic assignment problem. Knowledge-Based Systems 17(2-4), 65–73 (2004)
9. Paul, G.: Comparative performance of tabu search and simulated annealing heuristics for the quadratic assignment problem. Operations Research Letters 38(6), 577–581 (2010)
10. Eberhart, R., Kennedy, J.: A new optimizer using particle swarm theory. In: International Symposium on MICRO Machine and Human Science. pp. 39–43 (1995)
11. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks (ICNN). pp. 1942–1948 (1995)
12. Bahirat, S., Pasricha, S.: A particle swarm optimization approach for synthesizing application-specific hybrid photonic networks-on-chip. In: International Symposium on Quality Electronic Design (ISQED). pp. 78–83 (2012)
13. Sahu, P.K., Shah, T., Manna, K., Chattopadhyay, S.: Application mapping onto mesh-based network-on-chip using discrete particle swarm optimization. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 22(2), 300–312 (2014)
14. Singh, A.K., Srikanthan, T., Kumar, A., Jigang, W.: Communication-aware heuristics for run-time task mapping on NoC-based MPSoC platforms. Journal of Systems Architecture 56(7), 242–255 (2010)
15. Hu, J., Marculescu, R.: Energy-aware mapping for tile-based NoC architectures under performance constraints. In: Proceedings of Asia South Pacific Design Automation Conference (ASP-DAC). pp. 233–239 (2003)
16. Kennedy, J., Eberhart, R.: A discrete binary version of the particle swarm algorithm. In: Proceedings of IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation. pp. 4104–4108 (1997)
17. Liu, J., Harkin, J., McDaid, L., Martin, G.: Hierarchical Networks-on-Chip interconnect for astrocyte-neuron network hardware. In: International Conference on Artificial Neural Networks (ICANN). pp. 382–390 (2016)