Efficient Mapping Algorithm on Mesh-based NoCs in Terms of Cellular Learning Automata

Mohammad Keley[1], Ahmad Khademzadeh[2], and Mehdi Hosseinzadeh[1]
[1]Department of Computer, Islamic Azad University, Iran
[2]Information and Communication Technology Research Institute, IRAN Telecommunication Research Center, Iran

Abstract: *Network-on-Chip (NoC) presents the interesting approaches to organize complex communications in many systems. NoC can also be used as one of the effective solutions to cover the existing problems in System-on-Chip (SoC) such as scalability and reusability. The most common topology used in NoC is mesh topology. However, offering the mapping algorithm for mapping applications, based on weighted task graphs, onto the mesh is known as a NP-hard problem. This paper presents an effective algorithm called 'Boundary Mapping Algorithm' (BMA), in terms of decreasing the priority of low weighted edges in the task graph to improved performance in the NoCs. A low complexity mapping algorithm cannot present the optimal mapping results for all applications. Then, adding an optimization phase to mapping algorithms can have a positive impact on their performance. So, this study presents an optimization phase based on Cellular Learning Automata to achieve this goal. For the evaluation mapping algorithm and optimization phase, we compared the BMA method with Integer Linear Programming (ILP), Nmap, CastNet and Onyx methods for six real applications. The mapping results indicated that the proposed algorithm can be useful for some applications. Also, optimization phase can be useful for the proposed and other mapping algorithms.*

Keywords: *Cellular learning automata, mapping algorithm, network on chip, optimization algorithm, power consumption.*

Received May 22, 2014; accepted June 8, 2016

1. Introduction

System-on-Chip (SoC) is designed for integrated circuits of all components of electronic system into single chip [12]. Scalability and complex communications are considered as challenging issues in the SoC. Network-on-Chip (NoC) is an effective approach to overcome scalability and complex communications problems in designing the SoC. The main characteristic of NoC is the separation between computation and communication, which provides advantages such as scalability, reusability and high performance. Although there are various topologies exist for NoC architectures, the basic and well accepted topology is the mesh topology. Teraflops Research Chip of Intel used mesh-based topology which is known as multi-core architecture with commercial usages. Actually 80 connected processing cores in a two dimensional mesh network strategy (2D mesh network) can be viewed in the so-called chip. Mapping the applications in an optimal way is one of the most important challenging points which exists for mesh-based NoC architectures. In terms of [6], this would be NP-hard problem, when the applications are mapped on the mesh topologies.

An approach to map a task graph into the NoC includes selecting the neighboring node with the highest communication rate as the next node. The challenge in this regard is that, compared to the other weight of edges in the task graph, the selected neighbor's node is of a low communication need according to the highest communication rate. (It is possible that a node's highest communication need to its neighbors, compared to other graph's edges is considered as low). This situation may make some nodes not to be mapped properly with high priority at the end of the high priority list since some of these nodes are limited in selecting tile. For instance, in Video Object Plane Decoder (VOPD) task graph, for making high priority list, the next candidate after V9 is V12 while node V12 is not a proper member for the high priority list.

This paper proposes an efficient mapping algorithm through selection of edges with higher communication rate in the earlier mapping steps. In other words, in the earlier mapping steps, the edges with lower communication rates are ignored. The mapping operation begins from boundary tiles and moves on. Our proposed mapping algorithm is called Boundary Mapping Algorithm (BMA). To evaluate the performance of BMA method, we compared it to other mapping algorithms, such as Integer Linear Programming (ILP) [16], Near-optimal Mapping (Nmap) [11] , CastNet [18] and Onyx [9].

Also, studies show that a mapping algorithm with optimal solution for all applications is a very complicated one and because of its long time implementations is not effective enough. On the other hand, a complicated algorithm with algorithmic complexity and little time implementation cannot guarantee best results for all applications. So, to optimize efficiency of mapping algorithm with little time implementation, using a supplementary level can be effective. In this study, an algorithm based on Cellular Learning Automata has been proposed. The algorithm can be used by all mapping algorithms as a supplementary optimization phase.

Also, in order to evaluate the improved performance of our optimization algorithm, we have compared the results of many mapping algorithm such as Nmap [11] CastNet [18], Onyx [9] and BMA.

For this purpose, some parameters such as communication cost, Maximum Bandwidth Requirement and Power Consumption have been used.

The obtained results of the mathematical analysis have shown that this algorithm is efficient for mapping operations.

The remaining parts of this paper are organized as follows. In section 2, related work is discussed. Section 3 presents the mathematical formulation on mapping algorithms. Section 4 shows the proposed boundary mapping algorithm. Section 5, describes Cellular Learning Automata and section 6 explains the use of the cellular learning automata to improve performance in our proposed mapping algorithm and other mapping algorithms. Experiment results are shown in section 7. Finally, the conclusion is presented in section 8.

2. Related Work

In this section, we discuss different studies on mapping algorithms and some related studies about cellular learning automata.

The mapping idea proposed in [4] entitled Physical Mapping (PMAP) considers single minimum path routing in parallel with traffic routing. Also two strategies in [11, 16] and have been considered and proposed while paying attention to energy saving as a crucial factor. In [8], A novel approach has been introduced for the efficient mapping of the Directed Acyclic Graphs (DAG)-based applications. The approach that takes into account the lower and upper bounds for the start time of the tasks. The algorithm is based on list scheduling approach and has been compared with the well known list scheduling algorithms existing in the literature. The comparison results for the randomly synthesized graphs as well as the graphs from the real world elucidate that the proposed algorithm significantly outperforms the existing ones on the basis of different cost and performance metrics. The authors of [7] focused to increase speed branch-and-bound method which is capable of exploiting flexible routing and can also enhance the quality of the solution, as well. The authors of [13] present Mesh based On-Chip interconnection Architectures (MOCA) that utilizes tree-search based task mapping method have generated routes (paths) to improve mapping results. Among the ideas proposed for solving the mentioned problems that are revealed by [9, 18] and, CastNet and onyx are assumed as two heuristic methods who utilize the mesh's symmetric feature as a starting point. Chaos-Genetic-Based Algorithm (CGMAP) is the presented idea of [10] which uses chaos-genetic-based method, is able to achieve results close to other meta-heuristic methods.

ILP [16] based methods determine the optimum mappings in very long execution times. Cluster-based mapping method [13, 17] proposes a clustering based relaxation for ILP formulations in lower execution times than ILP, but it does not have optimal mapping.

In [13] authors present a detailed survey of the former work related to mapping methods. It classifies the reported techniques into groups like dynamic and static mapping approaches. Static mapping techniques have further been categorized as exact methods, branch-and-bound, transformative, and constructive approaches. They have also presented a performance comparison between the static mapping techniques.

Using cellular learning automata can be efficient to improved performance for mapping algorithms.

In [19, 20], Cellular Automata of (CA) was proposed by Van Neumann and thereafter was proposed by a mathematician called Ulam as a model to study the behavior of complex systems. Cellular automata are, in fact, discrete dynamical systems whose behavior is based completely on local communication.

Cellular learning automata is called uniform if operations between neighbors and cell, local rule and learning automata are the same for all cells, otherwise it is called non-uniform. The other kind of cellular learning automata is the Open Cellular Learning Automata (OCLA) [3]. In OCLA, in addition to local environment a global environment is considered. In OCLA, penalizing or rewarding the selected action by a cell depends not only on the selected action of its neighbors but also on the response of the global environment. It has been proved in [3] that this model like the closed Cellular Learning Automata (CLA) [2, 5] can be convergent to the local optimized points for mobility rules. If the connections among learning automata are not regular (like one dimensional or two dimensional array), the cellular learning automata is called irregular. If cells updating is carried out synchronously, it is called a synchronous automata otherwise it is called asynchronous [1].

3. Mathematical Formulation of the Mapping

Obviously, each application is modeled by a Task Graph (TG), including some nodes equivalent to the tasks and edges which indicate the relationships among the tasks. The following definitions are used to formulate mapping algorithms [11].

- *Definition* 1: An Application Characterization Graph (APCG) is a directional graph, *G (V, E)*, in which each vertex denoted by $V_i \in V$ refers to one task and each directional edge, $e_{i,j} \in E$, indicates the communication between V_i and V_j. The weight of the edge $e_{i,j}$ represents the bandwidth of the communication between V_i and V_j [1,].

  The communication volume, namely the ratio of transmitted data from V_i to V_j, denoted by CommVolume $e_{i,j}$, is measured in Mbits/sec and is computed as shown in Equation (1).

$$\forall e_{i,j} \in E, \quad \exists CommVolume_{i,j} = volume(e_{i,j}) \tag{1}$$

The NoC topology is usually modeled by a directional graph.

- *Definition* 2: The Architecture Characteristics Graph (ARCG) is a directional graph represented by *g'* = *G (T, P)*, where each vertex, $t_i \in T$, refers to one tile in NoC topology and each directional edge, $p_{i,j} \in P$, refers to a path from t_i to t_j. A task graph, *g* = *G (Vertex, Edge)*, is mapped onto the topology graph, *g'* = *G (tile, path)*, by a one-to-one mapping function, *map( )* function, as follows [11]:

$$Size(APCG) \le Size(ARCG) \tag{2}$$

$$\forall v_i \in V, map(v_i) \in T \tag{3}$$

$$\forall v_i \ne v_j \in V, map(v_i) \ne map(v_j) \tag{4}$$

$$d\begin{cases} d^k : vl(d^k) = comm_{i,j}, k = 1,2,..|E| \ \forall e_{i,j} \in E \\ source(d^k) = map(v_i),\ dest(d^k) = map(v_j) \end{cases} \tag{5}$$

- *Communication Cost parameter*: The communication cost is one of the evaluation criteria for mapping algorithm onto NoC, which is calculated as follows [11].

$$commcost = \sum_{k=1}^{|E|} vl(d^k) dist(source(d^k), dest(d^k)) \tag{6}$$

Where $vl(d^k)$ is data transfer rate from source to destination and $|E|$ is the Number of edges.

Communication cost criteria is used to make a preliminary evaluation for mapping algorithms. To establish a complete evaluation, other parameters should be taken into account.

- *Bandwidth Requirement parameter*: The communication cost is calculated based on total traffic of links and minimum distance among task tiles in NoC. So, the communication cost is independent of routing algorithms. However, to determine the actual Bandwidth Requirement on links, the routing path between source and destination nodes should be considered [15].

  Bandwidth Requirement on links equals the amount of transferred data during implementation of the application.

$$\forall link \ \ l_{i,j} \in L \ \ B(l_{i,j}) = \sum_{k=1}^{|E|} f(l_{i,j}, path(source(d^k), dest(d^k))) \tag{7}$$

$$f(l_{i,j}, path(S,D)) = \begin{cases} vl(d^k) & if \ \ l_{i,j} \in path(S,D) \\ 0 & otherwise \end{cases} \tag{8}$$

Where $l_{i,j}$ is Bandwidth requirement on link, and *f(l*$f(l_{i,j},path(S,D))$ is data transfer rate from source to destination through $l_{i,j}$.

An efficient mapping algorithm should minimize the Bandwidth Requirement on NoC links, during the implementation process of application. Note that the mapping algorithm, dependent on the routing algorithm, must operate so that the Bandwidth requirements are less than the physical bandwidth.

$$\forall l_k; \ \ max(B(l_k)) = B_{noc} \le B_{max} \tag{9}$$

Where B_{max} is the maximum allowed bandwidth of the system and B_{noC} is the minimum required bandwidth of our design.

- *Power Consumption parameter*: A model for energy consumption in routers has been proposed in [7]. The amount of consumed energy for transferring one bit from a router to its neighbor is calculated by Equation (10):

$$E_{bit} = E_{Sbit} + E_{Bbit} + E_{Wbit} + E_{Lbit} \tag{10}$$

Where E_{Sbit}, E_{Bbit}, E_{Wbit}, and E_{Lbit} represent the consumed energy in switch, buffer, interconnection wires inside the fabric and communication lines, respectively. The consumed energy in buffers and interior lines is trivial, compared to the E_{Lbit} (E_{Bbit} + E_{Wbit} < E_{Lbit}). Therefore, Equation (10) can be represented as follows:

$$E_{bit} = E_{Sbit} + E_{Lbit} \tag{11}$$

The average consumed energy in sending one bit from tile t_i to tile t_j is calculated by Equation (12) and it is as follows:

$$E_{bit}^{t_i,t_j} = \sum_{k=1}^{nhop} E_{Sbit} + \sum_{k=1}^{nhop-1} E_{Lbit} \tag{12}$$

Since a bit may be routed through 3×3, 4×4, and 5×5 routers, Equation (13) can be used to calculate the above parameter, as follows:

$$E_{bit}^{t_i,t_j} = \sum_{r=3,4,5} hops_{(r\times r)} \times E_{Sbit(r\times r)} + \sum_{k=1}^{nhop-1} E_{Lbit} \tag{13}$$

Where $hops_{(r \times r)}$ and $E_{sbit\,(r \times r)}$ are the number of routers with $r\times r$ crossbar from t_i to t_j and power consumption for $r\times r$ crossbar when it routes a bit, respectively. The total energy consumed for data transferring is determined by multiplying the obtained energy for one bit by the data amount as formulated in Equation (14):

$$E_{total}^{t_i,t_j} = DataSize_{t_i,t_j} \times E_{bit}^{t_i,t_j} \tag{14}$$

4. The Proposed Boundary Mapping Algorithm (BMA)

As it has been seen, one of the critical problems in NoC is mapping the task graph. For this purpose, so far, many mapping algorithms with different performances have been proposed. In most of these algorithms, the next node is selected based on finding a neighbor node with high communication rate. This selection method may result in using inappropriate nodes, as shown in Equation (15).

$$\begin{aligned} &Next \quad Priority(v_i) = v_j \quad \forall v_j \in V \\ &with \quad Max(vl(d^k)) \\ &vl(d^k) = comm_{i,j} \end{aligned} \tag{15}$$

The main problem of these algorithms is that selecting the next node may result in getting a node with low communication rate. Sometimes, this node is the only neighbor node connected to the last mapped node, or the other neighbors have low communication rate. One of the ideal mapping algorithms includes selecting nodes with high communication rate which have been located near each other.

Another problem that occurs in the mapping process of a loop with three nodes is having edges with high communication rate in the task graph. Three nodes can be mapped through linear or square approaches. The problem of the linear approach includes increasing the traffic on links in Network-on-Chip (NoC), as there is only one path among three nodes. The main advantage of the square approach is the distribution of this traffic on links, which results in decreasing the total traffic on link, compared to the linear approach as shown in Figure 1.

Figure 1 shows a part of VOPD task graph. It shows nodes V_8, V_9, and V_{10} compose a three-member loop of high data transition. The traffic increases critically on links while mapping three-member loop linearly. The most suitable case for mapping such a loop is using the square case, as shown in Figure 1. To implement this case, two nodes should be mapped linearly and the third node is mapped according to its closest distance to its neighbor in task graph.

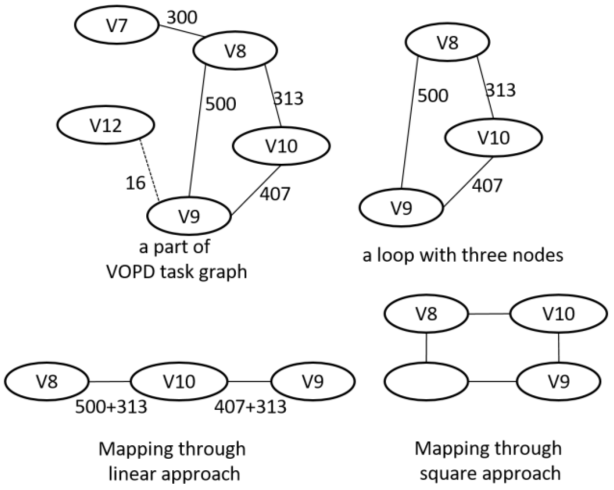

Figure 1. Mapping through linear and square approach for a part of VOPD with three member loop.

To overcome the existing problems in the previous presented mapping algorithms, a new mapping algorithm called BMA is proposed that is able to map the graph task effectively with selecting boundary tiles.

In the BMA method: Edges with lower communication rate are calculated based on the Equation (16):

$$e_{low} = e_{max}/D \tag{16}$$

Where e_{low} , e_{max} and D are the lowest value for comprising volume edges, maximum volume edges and the maximum manhattans distance between each pair of tiles in mesh topology, respectively.

The nodes with data communication rate more than e_{low} are put in the higher priority list. Other nodes are moved to the lower priority list. Also, it should be mentioned that if selecting neighbor node forms a loop with triple nodes, it will be added to lower priority list. It should be mentioned that every entity of the low priority list is formed by three parts, including the intended node number, its neighbor node number, and the highest communicative need between index and its neighbor. Before proposed mapping process, it is necessary for the lower priority list to be sorted in terms of the data communication rate.

The proposed mapping process starts with mapping the nodes from higher priority list onto boundary tiles in NoC. Next, the nodes of lower priority are mapping onto remained tiles in terms of minimum distance with neighbor's node.

This algorithm tries to map three-member loops with high data transition as square mapping as far as possible. This algorithm selects two members of this loop and the third node is included in the low-priority list. Since, based on the rate of data transition, the low-priority list is arranged in a descending order, the third node is mapped when the other previous nodes in the low priority list have already been mapped. Then, the third node is mapped in the closest tile onto its neighbor in graph. If the tile in its neighbor is empty, then the third node is included in a square case along with the two first nodes (similar to mapping on VOPD task graph) and if the selected tile is available for another task then the three nodes are no longer in a square case. (The reason is that other nodes may be related to a member of the three-member loop which has the highest data transition rate in task graph). The pseudo-code of BMA method is given in Algorithm 1.

Algorithm1: The pseudo-code of the BMA

```
procedure BMA(input APCG, ARCG)
// APCG; Characteristic Task Graph
// ARCG; NoC Architecture
//G: Set of Vertexes of ARCG
//k; the number of rows in ARCG
//l; the number of columns in ARCG
begin
D = (k - 1) + (l - 1);
elow = max(volume of edges)/D;
Index←First_Member;
State=1; // show that index is the first node of sub graph
while (G≠0) do
     Max_volume = Max(comm(Index,Neighbor's));
     while  (Max_volume < elow And G ≠ 0) do
          if (state == 1)
                Low_PL =  Low_PL + {(Index,null,0)};
                G = G - {index};
                Index←First_Member;
                State = 1;
                Max_volume = Max(comm(Index,Neighbor's));
          else if
                High_PL = High_PL + {Index};
                G = G - {Index};
                Index←First_Member;
                State = 1;
                Max_volume = Max(comm(Index,Neighbor's));
          end if
     end while
     for all neighbor's
          if (comm(Index,Neighbor) = Max_volume And
                         comm(Index,Neighbor) > elow)
              if(loop(index,neighbor) = True)
                 Low_PL =  Low_PL + {(neighbor,Index,
                       comm(Index,Neighbor))};
             G = G - {Neighbor};
             Max_volume = Max(comm(Index,Neighbor's));
             Start the For loop again for index neighbors;
               else
             High_PL = High_PL + {Index};
             G = G - {Index};
             Index = Neighbor;
         State = 0;  //state == 0 show that index is not  first node
             Max_volume = Max(comm(Index,Neighbor's));
               end if
          end if
     end for
 end while
Low_PL ←SORT Low_PL;
ARCG ←Map_High(ARCG, High_PL);
ARCG ←MAP_low(ARCG, Low_PL);
end procedure;
```

- *Example* 1: The BMA method for VOPD application is explained as follows. As shown in Figure 3, the highest communication rate is 500 Mbits/s. Also, the maximum manhattans distance between tiles is 6.

  $e_{low} = 500/6 = 83$

Figures 2 shows selection edges of VOPD task graph in terms of e_{low} for high and low priority lists.

G={V_1, V_2, V_3, ………, V_{15}, V_{16}}

- High_ PL= {V_2, V_3, V_4, V_5, V_6, V_7, V_8, V_9, V_{13}, V_{14}}

- Low_PL= {(V_1,V_2,70),(V_9,V_{10},407),(V_{11},V_{12},16), (V_{12},V_{13},16),(V_{15},V_{14},16), (V_{16},V_4,49)}

- Sort(Low_PL) ={(V_9,V_{10},407), (V_1,V_2,70), (V_{16},V_4,49), (V_{11},V_{12},16), (V_{12},V_{13},16), (V_{15},V_{14},16)}

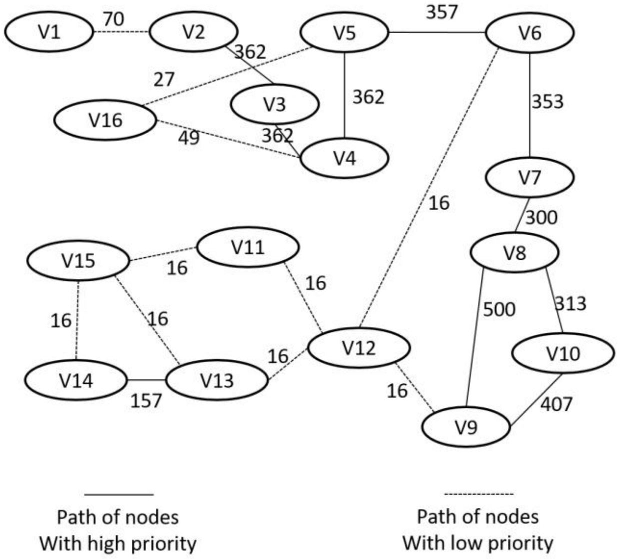

Figure 2. Edge selection based on priority for VOPD.

Figure 3 shows map high and low priority lists onto NoC tiles.

V 5	V 6	V 7	V 8
V 4			V 10
V 3			V 13
V 2			V 14

Map
High_PL

V 5	V 6	V 7	V 8
V 4	V 16	V 9	V 10
V 3	V 11	V 12	V 13
V 2	V 1	V 15	V 14

Map
LOW_PL

Figure 3. Boundary mapping algorithm for VOPD in 2D NoC.

The priority survey for nodes is started from node V_1. The nodes with high priority are specified with red color. Despite having higher communication rate than

83 between V_9 and V_{10}, the node V_9 is placed in low priority list, because it makes a three member loop with V_8 and V_{10}. After sorting out low priority list in term of communication rate, all members of high and low priority lists are mapped onto NoC tiles, respectively.

In mapping task graph, nodes having the IO (input/output) function should be mapped on bordering tiles. So, we can say that the rate of data transition is higher than e_{max} for nodes with IO functions. Therefore, these nodes are included in the high-priority list and then mapped on the border tiles.

5. Cellular Learning Automata

In this section, we briefly describe Learning Automata, Cellular Automata and Cellular Learning Automata.

5.1. Cellular Automata

A cellular automaton consists of a regular grid of cells, each in one of a finite number of states such as on and off (in contrast to a coupled map lattice). The grid can be in any finite number of dimensions. For each cell, a set of cells called its neighborhood (usually including the cell itself) is defined relative to the specified cell. Time proceeds discretely and its rules are as general through which in each cell, considering its neighbors. It obtains a new state in each stage. The rules of cellular automata determine the way a cell is influenced by neighboring cells. We call a cell a neighboring cell when it can influence its neighbor in a stage based on the dominant rule. To obtain the current cell state, one can take into account the previous state of cell in addition to the previous state of neighbor's cell.

5.2. Learning Automata

A learning automata is a machine which can perform a finite number of actions. Each selected action is evaluated by a probable environment and the evaluation result is either a positive or negative response. The automata, then, is influenced by this response in its next action. The final goal for the automata is to learn to choose the best action from among its actions. The best action is the action which maximizes the probability of getting a reward. Learning automata in interacting with the environment is shown in Figure 4 below.

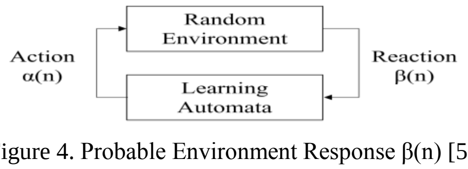

Figure 4. Probable Environment Response β(n) [5].

Environment can be shown as triple $E \equiv \{\alpha, \beta, c\}$ in which α is input sets, β is output sets, and c is probable penalty set. If β is a double-member set, the environment is of p type. In such an environment, $\beta_1 = 1$ considered as penalty and $\beta_2 = 0$ as reward. In a Q-type environment $\beta\ (n)$ can be a discrete value in [0, 1] interval and in s-type environment it can be random variable in [0, 1] interval. c_i is a probability showing that α_i performance was unfavorable. In a stationary environment c_i amount remains unchanged, while in a stationary-static environment, these amounts change as time passes.

Learning automata with variable structure can be shown by a quadruple set of $\{\alpha, \beta, p, T\}$ in which $\alpha \equiv \{\alpha_1, \ldots, \alpha_r\}$ is automata's performance set, $\beta \equiv \{\beta_1, \ldots, \beta_m\}$ is automata's input set, $p \equiv \{p_1, \ldots, p_r\}$ is probable vector of selecting each action and $p(n+1) = T[\alpha(n), \beta(n), p(n)]$ is learning algorithm. The algorithm below is a type of linear learning algorithm. Suppose α_i action in nth step is selected:

-The favorable response

$$p_i(n+1) = p_i(n) + a[1 - p_i(n)] \tag{17}$$

$$p_j(n+1) = (1-a)p_j(n) \qquad \forall j \quad j \neq i \tag{18}$$

-the unfavorable response

$$p_i(n+1) = (1-b)p_i(n) \tag{19}$$

$$p_j(n+1) = (b/r-1) + (1-b)p_j(n) \qquad \forall j \quad j \neq i \tag{20}$$

In the relations above, P_i refers to probability of choosing a_i and P_j refers to probability of choosing the other actions. Also, a is the reward parameter and b is the penalty parameter. Considering a and b values, we can consider the three following states. When a and b are equal, we call the algorithm L_{RP} when b is smaller than a, we call it $L_{R\varepsilon P}$ and when b equals 0, we call it L_{RI}.

5.3. Cellular Learning Automata

Many problems cannot be solved by learning automata alone. The main power of learning automata appears when they are used collectively. Regarding this issue, a new model was developed by combining learning automata and cellular automata [5]. The following formal definition has been given for cellular learning automata [2].

The performance of cellular learning automata is described as the following. The learning automata select an action from among its own actions in any iteration. This action can be selected randomly or based on previous observations. The selected action is penalized or rewarded based on selected actions of the neighboring cells and the dominating rule of the learning automata.

Then, based on whether the selected action is rewarded or penalized, the automata modifies its behavior and the internal structure of the automata is updated. Updating for all automata is normally done simultaneously. After updating, each automata in cellular learning automata selects an action from among its actions set again and performs it. The process of selection, rewarding, and penalizing continues until the system reaches a stable state or the previously defined criteria are established. The updating process of existing automata structure in cellular learning automata is carried out by a learning algorithm.

6. The Proposed Optimization Algorithm

In this section, based on cellular learning automata, an approximate algorithm is proposed to optimize the efficiency of mapping algorithms. In this algorithm, first, a regular synchronous uniform cellular learning automata is considered based on NOC topology. For example in Figure 5, for an NOC topology with 3×3 dimensions, systematic cellular learning automata is considered with the same dimensions. The assumed neighbor type for this automata is of Van Neumann type. In other words, in this automata, each cell is neighboring the four cells which are above, below, right to, and left to it. For the bordering cells, neighborhood is defined as wrap-round.

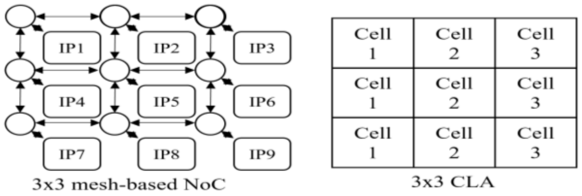

Figure 5. 3×3 mesh-based NoC [20] and 3×3 Cellular Learning Automata.

Each of the learning automata in the cellular learning automat has five actions of 'Exchange with the above neighbor', 'Exchange with the below neighbor', 'Exchange with the right neighbor', 'Exchange with the left neighbor' and 'no exchange'.

$$\alpha \equiv \{\alpha 1, \alpha 2, \alpha 2, \alpha 2, \alpha 2\} = \{\text{'Ex\_above ', 'Ex\_ below', 'Ex\_ right', 'Ex\_ left ', 'No exchange'}\} \quad (21)$$

The number of each cell in cellular learning automata is the number of the tile in NOC topology of the problem (from 1 to $n \times m$).

Learning automata in each cell of L_{RP} type has the rewarding and penalizing rate of 0.01. The possibility of selecting actions of every learning automata equals 0.2 (1/5) in the beginning. The proposed algorithm called *CLA*-mapping-optimization is as the following: Each cell selects simultaneously an action from the existing actions. Then the *comm_cost*$_x$ for the influenced cells by the selected action is calculated. If the calculated *comm_cost*$_x$ is less or equal than *comm_cost*$_x$ in before action selection, the action gives reward, or else the action gives penalty. The procedure continues to the time when the selected action by cells in a stage is 'not changing'. Finally, the obtained arrangement is presented and tested as the optimized NOC topology. The *comm_cost*$_x$ parameter is calculated in term of Equation (22).

$$\text{comm\_cost}_x = \sum_{k=1}^{\deg(v_x)} \text{volume}(e_{x,k}) \times (\text{dist}(\text{map}(v_x), \text{map}(v_k))) \quad (22)$$

The pseudo-code of this algorithm is shown in Algorithm 2.

Algorithm 2: The CLA-mapping-optimization

```
Algorithm CLA-mapping-optimization
Input: Graph G(V,E), mapped task graph
Output: Optimized mapped task graph
Construct a regular CLA for mapped task graph
Repeat
  For all cells do in parallel
     Old_Comm ← Comm_costx
     Select an action
     Compute cell's Comm_costx
     If(the cell's, Comm_costx ≤ Old_ Comm )
            Reward the cell's action
        else
            Penalize the cell's action
     End If
  End For
Until all cell's action are no change action
Return the mapped task graph
```

7. Results and Discussions

To evaluate the performance of proposed mapping algorithm, BMA, we compared it with other mapping algorithms, such as ILP [16], Nmap [11], CastNet [18] and Onyx [9] from mapping results viewpoints. Also, we used these mapping methods for the impact of the optimization algorithm from optimization result view point.

For this purpose, we selected five video applications namely Video Object Plane Decoder (VOPD) [11], Moving Picture Experts Group (MPEG-4) decoder [12], 263EncMp3Dec (263Enc) [14], 263DecMp3Dec (263Dec) [14], and Mp3EncMp3Dec (Mp3Enc) [14], Double Video Object Plane Decoder (DVOPD) [13] .

7.1. Mapping Results

In the mapping results, some parameters such as Communication Cost, Maximum Bandwidth Requirement and Power consumption were used.

Table 1. Comparisons among BMA, Onyx, CastNet, NMAP and ILP in terms of Communication Cost.

Task Graph	**Communication Cost of mapping algorithms (Mbits/s)**				
	ILP	Nmap	CastNet	Onyx	BMA
VOPD	4119	4275	4135	4119	4135
MPEG-4	3567	3672	3852.5	3817.5	3843.5
263Enc	230.407	230.432	230.417	230.407	230.417
263Dec	19.823	17.948	17.948	18.361	18.101
Mp3Enc	17.021	17.521	17.046	17.046	17.046
DVOPD	-	10253	9370	9759	9238

In Table 1, columns two to six give the communication costs for the mappings generated by ILP, Nmap, CastNet, Onyx and BMA, respectively. In the worst case, BMA has a higher communication cost for MPEG-4 compared to Onyx, Nmap and ILP. BMA has lowest communication cost for DVOP compared to other mapping algorithms.

The high communication cost in using MPEG4 is included in node Synchronous Dynamic Random-Access Memory (SDRAM) in task graph because of it has high degree. The proposed algorithm does not have priority for nodes with high degree.

Table 2 shows the execution time of ILP, Nmap, Castnet, BMA algorithms for mapping different applications. In applications VOPD, MPEG4, 263EncMp3Dec, Mp3EncMp3Dec and DVOPD, the algorithm BMA has performed better; however, for 263EncMp3Dec application, BMA mapping algorithms places second after CastNet. For applications task graph with higher number of nodes such as DVOPD, BMA produces better results. Time Out (TO) has been considered as 6 hours for the execution of algorithm. In ILP algorithm for executing of DVOPD, time has lasted longer than TO.

Table 2 . Comparisons among BMA, CastNet, Nmap and ILP in terms of CPU Time.

Task Graph	CPU Time of Mapping Algorithms (s)			
	ILP	Nmap	CastNet	BMA
VOPD	6927.36	0.045	0.0478	0.0435
MPEG-4	35.46	0.32	0.1836	0.174
263Enc	402.62	0.023	0.0168	0.0192
263Dec	6994.23	0.34	0.2272	0.184
Mp3Enc	2915.89	0.0305	0.02734	0.02575
DVOPD	T.O.	0.67	0.7086	0.5608

Table 3 shows the maximum bandwidth requirement and comparisons among mapping algorithms, regarding the XY routing method. The BMA method has the best result for VOPD and 263DecMp3Dec task graph. The BMA method has the best result with B_{NoC}=516 Mbits/sec, compared to the other mapping algorithms such as Onyx with B_{NoC}= 613 Mbits/sec, CastNet with B_{NoC}=813 Mbits/sec and Nmap with B_{NoC}=829 Mbits/sec. Also, it has one of the best results for 263DecMp3dec application, with B_{NoC} =4.060 Mbits/sec. Based on these results, the BMA method has provided results close to the other algorithms, for other applications. At worst, the results obtained for our method are within the 1% limit of the best solution for MPEG-4 compared to Onyx.

Table 3. Comparisons among BMA, CastNet, Nmap and ILP in terms of maximum bandwidth requirement.

Task Graph	Maximum Bandwidth requirement (Mbits/s)				
	ILP	**Nmap**	**CastNet**	**Onyx**	**BMA**
VOPD	613	829	813	613	516
MPEG-4	942	926.25	942	910	926.25
263Enc	46.733	46.733	46.733	46.733	46.733
263Dec	4.06	4.06	4.06	4.1720	4.06
Mp3Enc	4.06	4.06	4.06	4.06	4.085
DVOPD	-	813	877	662	813

We use the power/energy model presented in [7] to estimate the energy consumption of each mapping algorithm, in terms of Equations (13) and (14). In [15], the results of applying energy model over routers, topology switches and links are shown in Table 4. It is mentioned that a packet contains a 96 bits of data.

Table 4. Characterization of the routers, topology switches and link [15].

Module	Energy per packet (pJ)
Link,1 mm	21
5x5 Router **Topology switch**	32 0.6/0.8
4x4 Router **Topology switch**	31 0.6/1.1
3x3 Router **Topology switch**	30 0.6/1.3

Table 5. Comparisons among BMA, CastNet, Nmap and ILP in terms of Power consumption.

Task Graph	Power consumption (pJ)			
	Nmap	**CastNet**	**Onyx**	**BMA**
VOPD	3621.508	3516.631	3520.753	3511.785
MPEG-4	3193.608	3282.226	3262.781	3268.991
263Enc	203.8280	203.8199	203.8199	202.6042
263Dec	15.7475	15.7425	16.0631	15.8412
Mp3Enc	15.0866	14.8910	14.8910	14.6514
DVOPD	8151188	7657458	7866104	7576156

The best mapping algorithm shows minimum power consumption. Based on the values reported in Table 5, the BMA mapping algorithm shows the best results for VOPD, 263EncMp3Dec, MP3EncMp3Dec and DVOPD. In other words, it has power consumption for VOPD is 3511.785 pJ which compared to the other mapping algorithms, such as Onyx with 3520.753 pJ, CastNet with 3516.631 pJ and Nmap with 3621.508 pJ, is minimum. The BMA method has provided results close to the other algorithms for other applications. At worst, in this mapping algorithm, our result is within the 2% limit of the best solution, for MPEG-4 compared to Nmap.

For DVOPD, BMA has 4% better result than Onyx, 1% better than Castnet and 7% better than Nmap from power consumption view point.

7.2. Optimization Result

In Optimization result, the communication cost of applications for mentioned mapping algorithms after optimization phase were calculated and presented in Table 6.

Optimization phase behaves in a way so that communication cost is reduced as far as possible and communication cost parameter can affect maximum bandwidth requirement and power consumption parameter. Then, the improved ratio for these parameters is calculated in term of Equation (23).

For Improved Ratio parameter, a positive sign shows the success of optimization phase. If it has the N.C (No Change), it means that the optimization phase has had no effect, and if the sign is negative, it means the optimization phase has had a negative effect.

$$Im\,proved \quad Ratio = \frac{Value \quad of \quad parameter_{Before \quad Im} - Value \quad of \quad parameter_{After \quad Im}}{Value \quad of \quad parameter_{Before \quad Im}} \tag{23}$$

Table 6. Improved value of communication cost for mapping algorithms.

Task Graph	After Improved (Mbits/s)			
	Nmap	CastNet	Onyx	BMA
VOPD	4265	4135	4119	4119
MPEG-4	3672	3772.5	3772.5	3793
263Enc	230.417	230.417	230.407	230.417
263Dec	17.948	17.948	18.148	17.948
Mp3Enc	17.521	17.046	17.046	17.046
DVOPD	9759	9370	9759	9238

In Table 6, columns two to five give the communication cost after optimization phase for the mapping algorithms generated by Nmap, CastNet, Onyx and BMA, respectively. The communication cost before optimization phase for mapping algorithms is shown in Table 1.

The result shows that the optimization phase could reduce the communication cost for some applications in different mapping algorithms. We should notice that compared to other mapping algorithms for mapping task graph of different applications, the mapping algorithms of NMAP, CastNet, Onyx and BMA have less communication cost. The optimization phase for the algorithm of the increased mapping has less time and can optimize communication cost to 4.8%, for Nmap in DVOPD. The optimization could reduce communication cost parameter 0.5% on average for the mapping algorithms of Nmap, CastNet, Onyx and BMA for the mentioned benchmarks. Mapping algorithms of Nmap, CastNet and Onyx cannot guarantee the best results for all applications and each of them is useful for some applications. Then, if a mapping algorithm gives the best result for an application, the optimization phase is not useful. For example, Onyx mapping for VOPD could give the best mapping with communication cost of 4119 and no other mapping algorithm could give a communication cost less than 4119. Therefore, the optimization phase was not effective; however, the same algorithm could not give the best result for MPEG-4 and 263DecMp3Dec. Then, the optimization phase was very effective and could give the best result for the output of Onyx mapping algorithm for MPEG-4 and 263DecMp3Dec applications. Other similar cases have been generated for the other mapping algorithms. The important thing is that the optimization algorithm could give positive effect for all mapping algorithms.

Table 7 shows the maximum bandwidth requirement and comparisons among mapping algorithms regarding the XY routing method. As the improved value shows, the optimization phase improved maximum bandwidth requirement in mapping algorithms for some applications such as VOPD and DVOPD for Nmap, MPEG-4 for CastNet, 263DecMp3Dec for Onyx, Mp3EncMp3Dec for BMA. Our optimization algorithm has optimized the limitation of maximum bandwidth requirement to 18.5%, for Nmap in DVOPD.

Table 7. Improved value of maximum bandwidth requirement for mapping algorithms.

Task Graph	After Improved (Mbits/s)			
	Nmap	CastNet	Onyx	BMA
VOPD	813	813	613	516
MPEG-4	926.25	910	910	926.25
263Enc	46.733	46.733	46.733	46.733
263Dec	4.06	4.06	4.06	4.06
Mp3Enc	4.06	4.06	4.06	4.06
DVOPD	662	877	662	813

The result of energy consumption in Table 8 shows that our optimization algorithm could improve power consumption to 4.5%, for BMA in 263DecMp3Dec.

Table 8. Improved value of power consumption for mapping algorithms.

Task Graph	After Improved(pJ)			
	Nmap	CastNet	Onyx	BMA
VOPD	3540.256	3516.631	3520.753	3511.384
MPEG-4	3193.608	3205.477	3205.477	3257.361
263Enc	203.8199	203.8199	203.8199	202.6042
263Dec	15.7475	15.7425	15.9396	15.1149
Mp3Enc	14.8025	14.891	14.891	14.6514
DVOPD	7849161	7657458	7866104	7576156

8. Conclusions

In this paper, we have proposed a new effective method for mapping applications onto NoC architectures with low complexity. We have also compared the presented approach with the four previous methods, ILP, Nmap, CastNet and Onyx. The result shows our mapping algorithm is better than other mapping algorithms in terms of power consumption for some application such as VOPD, 263EncMp3Dec and DVOPD.

A low complexity mapping algorithm cannot guarantee optimal mapping for different applications in designing NOC. Then, using a complementary phase for optimization in mapping operation can prove useful. In this study, we have used Cellular Learning Automata for optimization of the efficiency of mapping algorithms in designing NOC. In this method, a new algorithm using a set of five actions is used which uses the aspects of learning CLA and tries to select better states for each cell based on reduction of communication cost in mapping algorithms and the increase of NoCs performance. The obtained results of the mathematical analysis have shown that this algorithm is efficient for mapping operations.

Using optimization algorithm for mapping algorithms for all applications in mathematical analysis has improved to 4.8% for communication cost, 18.5% for maximum bandwidth requirement, and 4.5% for power consumption.

References

[1] Beigy H. and Meybodi M., "Asynchronous Cellular Learning Automata," *Automatica*, vol. 44, no. 5, pp. 1350-1357, 2008.

[2] Beigy H. and Meybodi M., "A Mathematical Framework for Cellular Learning Automata," *Advances in Complex Systems*, vol. 07, no. 3-4, 2004.

[3] Beigy H. and Meybodi M., "Open Synchronous Cellular Learning Automata," *Advances in Complex Systems*, vol. 10, no. 4, pp. 527-556, 2007.

[4] Benini L. and Micheli G., "Networks on Chips: a New Soc Paradigm," *Computer*, vol. 35, no. 1, pp. 70-78, 2002.

[5] FathyNavid A. and Bagheriaghababa A., *Cellular Learning Automata and its Applications*, IntechOpen, 2013.

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

[7] Hu J. and Marculescu R., "Exploiting The Routing Flexibility for Energy/Performance Aware Mapping of Regular Noc Architectures," *in Proceedings of the Design, Automation and Test in Europe Conference and Exhibition*, Munich, pp. 688-693, 2003.

[8] Ijaz S., Munir E., Anwar W., and Nasir W., "Efficient Scheduling Strategy for Task Graphs in Heterogeneous Computing Environment," *The International Arab Journal of Information Technology*, vol. 10, no. 5, pp. 486-492, 2013.

[9] Janidarmian M., Khademzadeh A., and Tvanpour M., "Onyx: a New Heuristic Bandwidth Constrained Mapping of Cores onto Tile-Based Network-on-Chip," *IEICE Electron Express*, vol. 6, no. 1, pp. 1-7, 2009.

[10] Moein-darbari F., Khademzade A., and Gharooni-fard G., "CGMAP: A New Approach to Network-on-Chip Mapping Problem," *IEICE Electron Express*, vol. 6, no. 1, pp. 27-34, 2009.

[11] Murali S. and Micheli G., "Bandwidth-Constrained Mapping of Cores onto Noc Architectures," *in Proceedings Design, Automation and Test in Europe Conference and Exhibition*, Paris, pp. 896-901, 2004.

[12] Pasricha S. and Dutt N., *On-Chip Communication Architectures: System on Chip Interconnect*, Morgan Kaufmann, 2008.

[13] Sahu P. and Chattopadhyay S., "A Survey on Application Mapping Strategies for Network-on-Chip Design," *Journal Systems Architecture,* vol. 59, no.1, pp. 60-76, 2013.

[14] Srinivasan K., Chatha K., and Konjevod G., "Linear-Programming-Based Techniques for Synthesis of Network-on-Chip Architectures," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 14, no. 4, pp. 407-420, 2006.

[15] Stensgaard M. Sparso J., "ReNoC: A Network-on-Chip Architecture with Reconfigurable Topology," *in Proceedings of the 2nd ACM/IEEE International Symposium on Networks-on-Chip,* Newcastle upon Tyne, pp. 55-64, 2008.

[16] Tosun S., Ozturk O., and Ozen M., "An ILP Formulation for Application Mapping onto Network-on-Chip," *in Proceedings of International Conference on Application of Information and Communication Technologies*, Baku, pp . 1-5, 2009.

[17] Tosun S., "Cluster-Based Application Mapping Method for Network-on-Chip," *Advances in Engineering Software*, vol. 42, no. 10, pp. 868-874, 2011.

[18] Tosun S., "New Heuristic Algorithms for Energy Aware Application Mapping and Routing on Mesh-Based Nocs," *Journal of Systems Architecture*, vol. 57, no. 1, pp. 69-78, 2011.

[19] Wolfram S., "Cellular Automata," *Los Alamos Science*, vol. 9, pp. 2-21, 1983.

[20] Wolfram S., "Universality and Complexity in Cellular Automata," *Physica D: Nonlinear Phenomena*, vol. 10, no. 1-2, pp. 1-35, 1984.

Mohammad Keley was born in Dezful. He received hir B.Sc. degree in Computer Engineering from Islamic Azad University, Dezfoul branch, Iran in 2003, and his M.Sc. degree in Computer Architecture Engineering from Science and Research Islamic Azad University, Tehran, Iran, in 2006. His research interests include design and analysis of reliable, fault tolerant and mapping in Network-on-Chips and 3-Dimentional Network-on-Chip interconnections.

Ahmad Khademzadeh was born in Mashhad, Iran, in 1943. He received the B.Sc. degree in applied physics from Ferdowsi University, Mashhad, Iran, in 1969 and the M.Sc., Ph.D. degrees respectively in Digital Communication and Information Theory & Error Control Coding from the University of Kent, Canterbury, UK. He is currently the Head of Education & National Scientific and International Scientific Cooperation Department at ICT Research Institute (ITRC). He was the head of Test Engineering Group and the director of Computer and Communication Department at ITRC. He is also a lecturer at Tehran Universities & he is a committee member of the Iranian Electrical Engineering Conference Permanent Committee. Dr. Khadem-Zadeh has been received four distinguished national and international awards including Kharazmi International Award, and has been selected as the National outstanding researcher of the Iran Ministry of Information and Communication Technology.

Mehdi Hosseinzadeh was born in Dezful in 1981. He received his B.Sc. degree in Computer Hardware Engineering from Islamic Azad University, Dezful branch, Iran in 2003. He also received his M.Sc. and Ph.D. degrees in Computer System Architecture from the Science and Research Branch, Islamic Azad University, Tehran, Iran in 2005 and 2008, respectively. He is currently an assistant professor in Department of Computer Engineering of Science and Research Branch of Islamic Azad University, Tehran, Iran. His research interests are computer arithmetic with emphasis on residue number system, cryptography, network security and e-commerce.