EMBEDDING BUS AND RING INTO HEX-CELL INTERCONNECTION NETWORK

Qatawneh Mohammad[1], Ahmad Alamoush[1], Sawsan Basem[1] , Maen M. Al Assaf[1], and Mohammad Sh. Daoud[2]

[1]Department of Computer Science, King Abdullah II School for Information Technology, The University of Jordan.
[2]Al Ain University of Science and Technology, UAE.

ABSTRACT

Hex-Cell is an interconnection network that has attractive features like the embedding capability of topological structures; such as; bus, ring, tree and mesh topologies. In this paper, we present two algorithms for embedding bus and ring topologies onto Hex-Cell interconnection network. We use three metrics to evaluate our proposed algorithms: dilation, congestion, and expansion. Our evaluation results show that the congestion of our two proposed algorithms is equal to one; and the dilation is equal to 2d-1 for the first algorithm and 1 for the second.

KEYWORDS

Embedding, Hex-Cell network, Dilation, Expansion

1.INTRODUCTION

Network topology shows the way in which a set of nodes are connected to each other by edges. A variety of network topologies for parallel architecture have been proposed due to the importance of interconnection networks in parallel systems and their performance. Consequently, some features in parallel system network are highly desirable such as: communication cost, efficient routing, minimum diameter, partitioning, and the capability of embedding topological structures (e.g. ring, star, linear array, tree, mesh, and hypercube) [3, 4, 5]. Graph embedding (mapping) has many applications in parallel processing and is considered one of the most important issues in network selection and evaluation. Hex-Cell is one of the various proposed interconnection networks structures for a parallel system. It receives much attention due to its attractive property of embedding ability and other features [1, 2, 5, 11]. In this paper, we propose an algorithm for embedding bus and ring onto a Hex-Cell.

Let G be (guest graph) and H be (host graph). An embedding of a graph G= (V_G, E_G) into a graph H= (V_H, E_H) is represented by a mapping function f: G → H that consists of two mappings: f_V: V_G → V_H and f_E: E_G → P_H, where P_H denotes the set of paths in the graph H. The mapping function f_E maps edge (u,v) ∈ E_G to a path p ∈ P_G connecting f_V (u) and f_V(u) [9, 10].
Three criteria can be used to evaluate embedding results. The first criterion is dilation, which is the maximum distance in H between adjacent vertices in graph G. The second one is congestion,

which is the maximum load on the edge of graph H. The third criterion is expansion, which is the efficiency of node utilization for an embedding scheme [6, 7, 8].

In this study, we propose two algorithms for embedding bus and ring onto Hex-Cell. The rest of this paper is organized as follows: Section 2 introduces Hex-Cell network. Section 3 illustrates the proposed embedding algorithms for bus and ring onto Hex-Cell Network. Section 4 presents an analytical model for the proposed embedding algorithm using several performance metrics like: dilation, congestion and expansion. Section 6 provides our research conclusions.

2. RELATED WORK

In this section, we illustrate the current research that is related to our research. In particular, we will discuss Hex-cell network, bus topology and ring topology.

2.1 Hex-Cell Network

Parallel processing is used for solving problems in major application areas such as image processing and scientific computing as it provides high processing power. Parallel machines performance is affected by the underlying communication network and use of suitable algorithm. Hence, there exist several desired features in parallel machine network such as minimal communication cost, efficient routing, and the capability of embedding topological structures such as tree, ring, linear array, and mesh. Hypercube is one of the most important interconnection networks structures proposed for parallel computing in which it has much attention due to the its important properties like: embeddability, symmetry, regularity, strong resilience, and simple But in hypercube structure, it was noticed that the number of communication ports and channels per processor increases as the network sizes increases. This forms a drawback for this approach. In our research, we aim to construct a new network that combines the following good features: less communication, efficient routing and capability of embedding other static topologies. This motivates us to "Hex-Cell" topology which forms a new interconnection network that meets the demands of highly parallel systems. Despite the facts that the hexagon topology was addressed as a 3D hexagonal network, the node degree and number of links are still considered high. Hence, our "Hex-Cell" approach has an enhanced use of node degree and links count when compared to the existing topologies.

In [1], we proposed a new class of interconnection networks topologies called "Hex-Cell". The most important feature of our proposed topology is the embedding capability of topological structures such as linear array, ring, tree and mesh. Our approach uses an efficient routing algorithm that does not require that much knowledge of the network interconnections and is capable to do communication in a less cost. Those features of our proposed solution make it a good candidate for general purpose applications. Due to its recursive structure, Hex-Cell is expandable in an incremental manner with the minimal cost. The metrics in which Hex-Cell is compared with other related networks are: degree, diameter, and network-cost.

In [1], we presented a complete modeling, topological properties, and a routing algorithm for our "Hex-Cell" solution. At the performance evaluation, we considered a Hex-Cell network topology with four levels (see: figure 1), and we illustrated six cases of the "Hex-Cell" routing algorithms:

move up/ left to right, move up/ right to left, move down/ left to right, move down/ right to left, horizontal/ left to right, horizontal / right to left (see: figures 2,3,4,5,6, and 7 respectively).

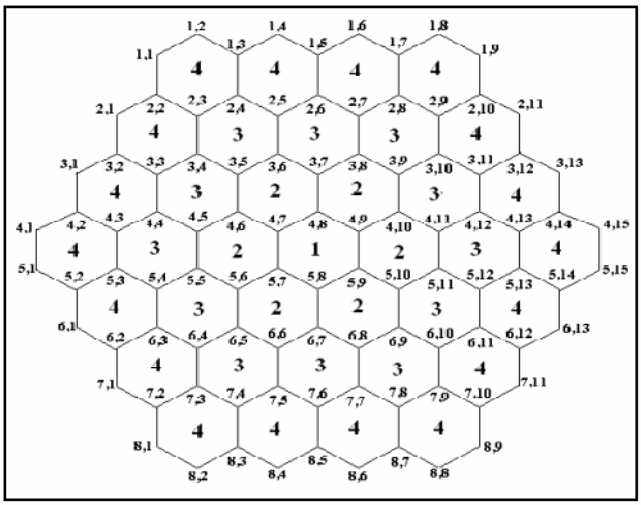

Figure 1: (Quoted from [1]) HC(4) topology with labeled nodes and levels

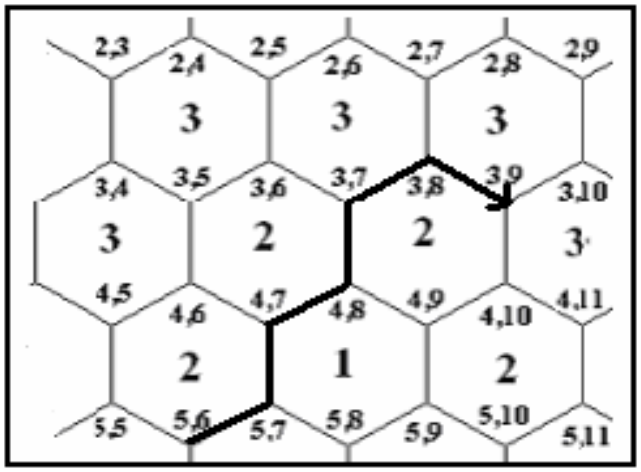

Figure 2: (Quoted from [1]) MoveUp/ Left-to-right

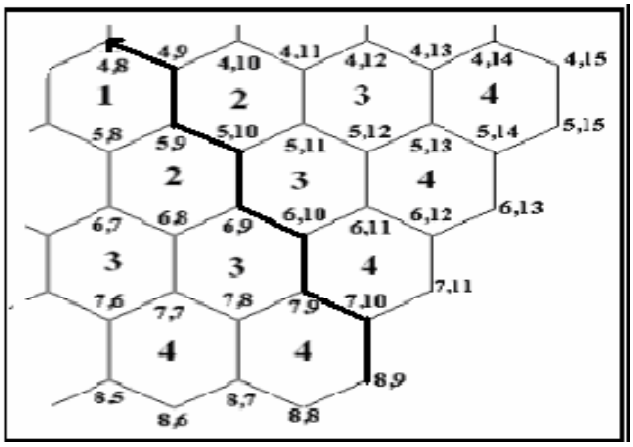

Figure 3: (Quoted from [1]) MoveUp/ Right-to-left

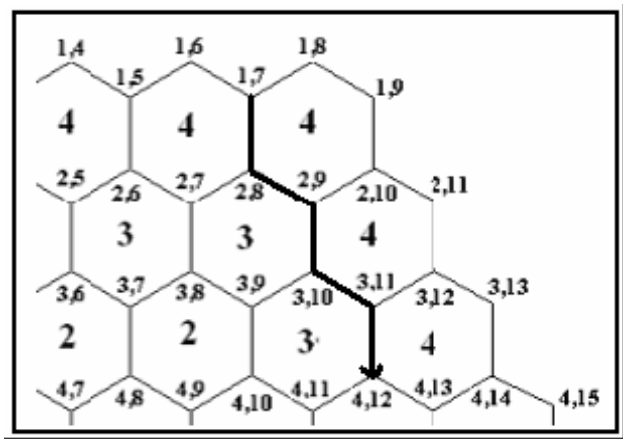

Figure 4: (Quoted from [1]) moveDown / Left-to-right.

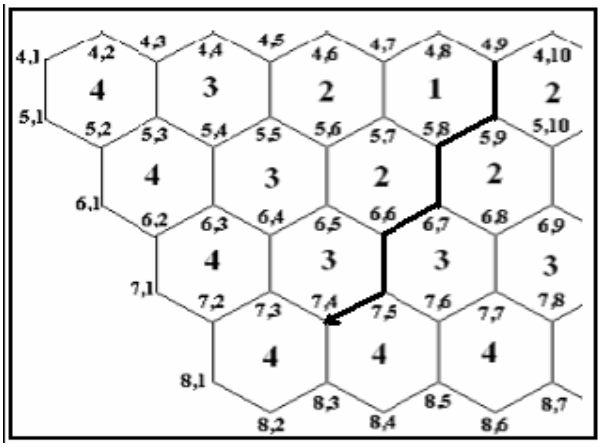

Figure 5: (Quoted from [1]) moveDown / Right-to-left

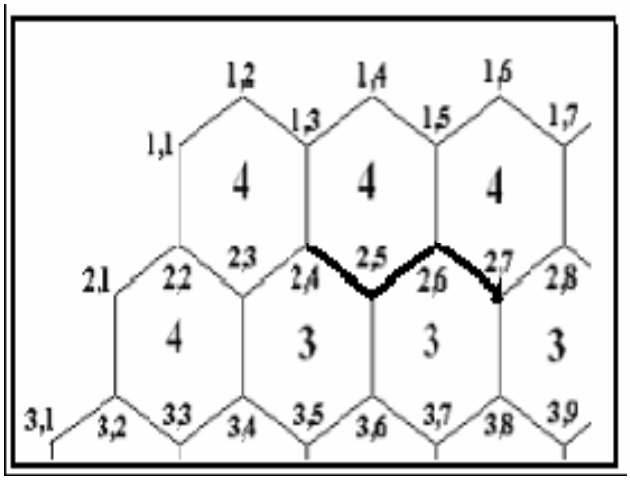

Figure 6: (Quoted from [1]) Horizontal / Left-to-right

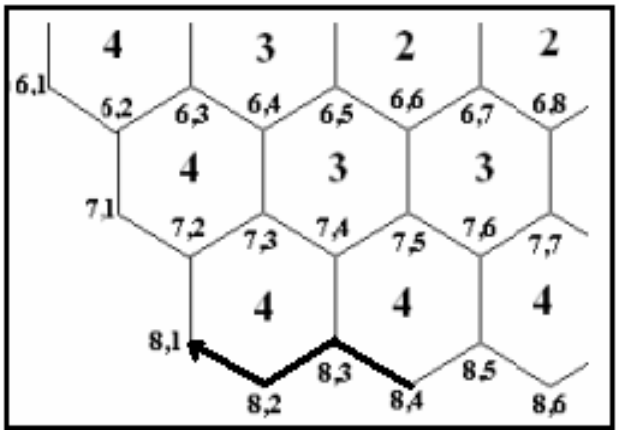

Figure 7: (Quoted from [1]) Horizontal / Right-to-left

In this research, we compare the proposed structure with other know topologies such as 3D hexagonal network, linear array, ring, binary tree, and hypercube as shown in Table 1. A Hex-Cell network with depth d is denoted by HC (d). It can be constructed using units of hexagon cells, each of six nodes [1]. A Hex-Cell network with depth d has d levels numbered from 1 to d, where level 1 represents the innermost one corresponding to one hexagon cell. Level 2 corresponds to the six hexagon cells that surround the hexagon at level 1. Level 3 corresponds to the 12 hexagon cells that surround the six hexagons at level 2 (see: Figure 8). The Hex-Cell network is divided into six sections labeled from left to right (clockwise) and numbered from 1 to 6. While each node is addressed by section number, both of; level number and the node number on that level are labeled from $X_1,\ldots\ldots,X_n$; where $n = ((2\times L) - 1)$ [1].

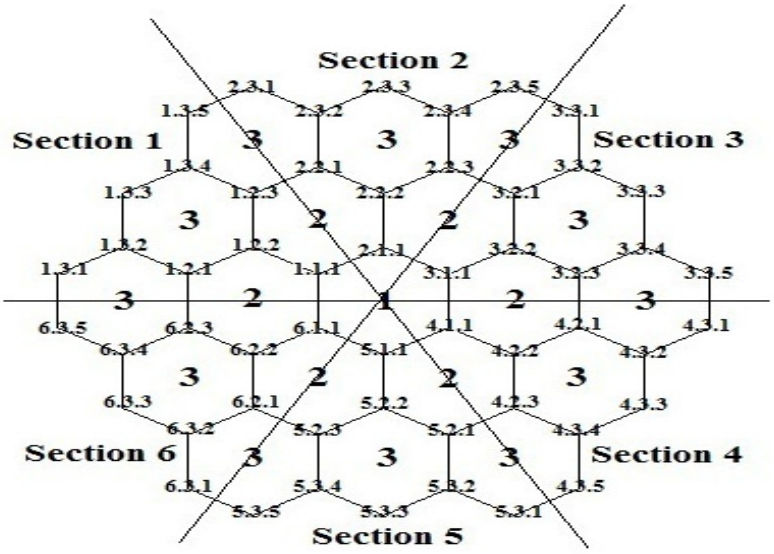

Figure 8: 3-dimensional Hex-Cell Network

2.2 Bus Topology

Is a network design that can transmit the data in one direction and can form a single network segment that can communicate with each other in the level of TCP/IP data link layer. Such network can host several workstations. The major concern of such network is the transmission coordination to avoid collision. Carrier sense multiple access (CSMA) technique is used for avoiding collisions. Bus topology is mainly used for small networks and requires less cable length. Figure 9 shows a typical buss topology network.

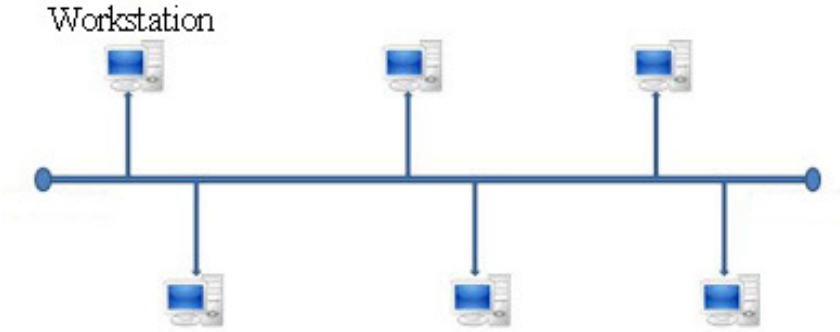

Figure 9: Typical Bus Topology

2.3 Ring Topology

Is a network design in which each node is connected with only two other nodes in a form of ring (see: figure 10). Data travels from the source to the destination through all the nodes located on the path. In order to avoid collision between the sending nodes, only the node who currently owns

the token can transmit over the network. When a node finishes transmission, it passes the token to the adjacent nodes, and so on. The problem with this approach is that the token is passed to the nodes that need it and to those that has nothing to transmit over the network.

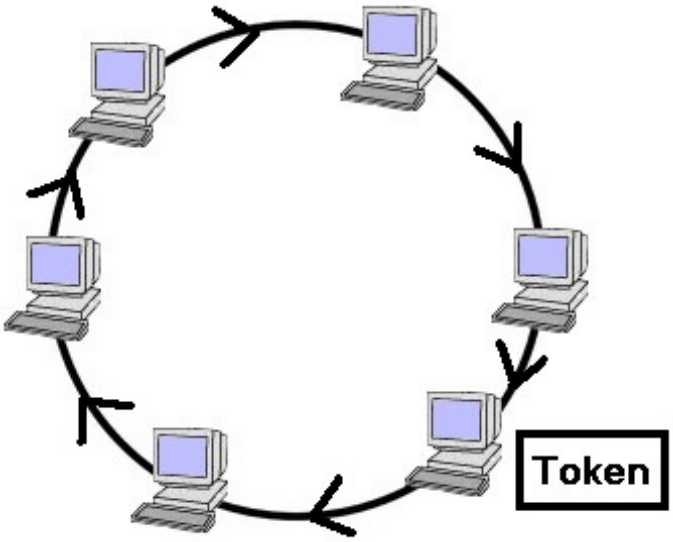

Figure 10: Typical Ring Topology

3. PROPOSED EMBEDDING ALGORITHMS

3.1 Algorithm (A)

The first proposed approach of embedding bus into Hex-Cell topology is based on two phases: First, clockwise move from the starting node at the same level. Second, move to the upper level (steps 1 to 5). In case we want to convert the Bus embedding into Ring one, one extra phase is required in which a clockwise move takes place from the last node in the Bus embedding to the next node then a move continues to lower level until it reaches to the starting node (steps 6 and 7). (see: Figure 11).

Step1 : The start node is 2.1.1
Step2 : Add the address of the start node to Node_Queue and mark it as checked node.
Step3 : Repeat Step2 by moving clockwise until the last node of that level is reached.
Step4 : Move to upper level adds the encounter node to the Node_Queue and mark it as checked, Repeat Step 3.
Step5 : Repeat step 4 until no more upper levels.
Step6 : Move to next node clockwise then move to the lower level.
Step7 : Repeat Step 6 until more level = 1 and then move right.

3.2 Algorithm (A) Discussion via Example

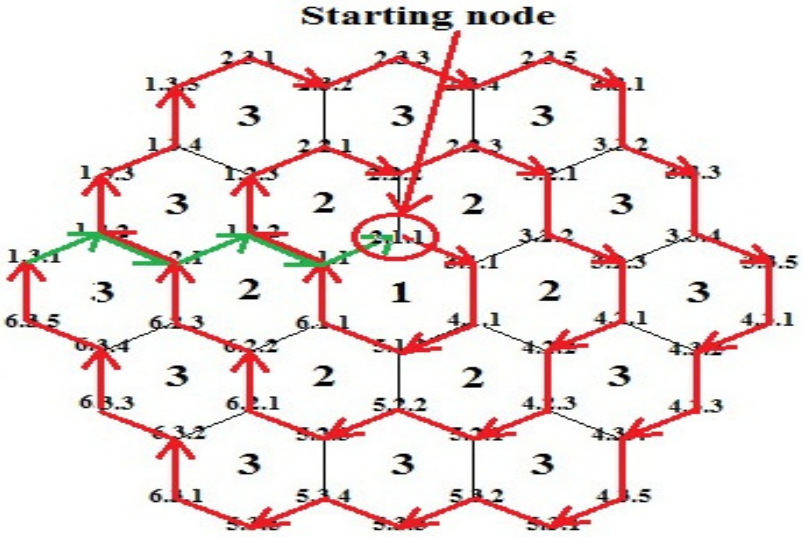

Figure 11: Algorithm A- embedding bus and ring into Hex-Cell topology

When applying the algorithm, let the starting node be (2.1.1). A clockwise move takes place through the first level until all the nodes in that level are selected; so the move ends at node (1.1.1). Then, a move to the upper level node (1.2.2) takes place. Then, a clockwise move takes place until all nodes in that level are selected; so the move ends at node (1.2.1). This pattern is continued until no more upper levels. In order to convert to Ring embedding after we finish the last upper level, Starting from the end, a clockwise move to the next node then another move to lower level take place. This is repeated until the starting node is reached. The following sequence describes the mentioned example. (see: Figure 12).

*Starting node is (2.1.1) → (3.1.1) → (4.1.1) → (5.1.1) → (6.1.1) → (1.1.1) → (1.2.2) → (1.2.3) → (2.2.1) → (2.2.2) → (2.2.3) → (3.2.1) →(3.2.2) → (3.2.3) → (4.2.1) → (4.2.2) → (4.2.3) → (5.2.1) → (5.2.2)→ (5.2.3) → (6.2.1) → (6.2.2) → (6.2.3) → (1.2.1) → (1.3.2) →(1.3.3) → (1.3.4) → (1.3.5) → (2.3.1)→ (2.3.2) → (2.3.3) → (2.3.4) → (2.3.5) → (3.3.1) → (3.3.2) →(3.3.3) → (3.3.4) → (3.3.5) → (4.3.1) → (4.3.2) → (4.3.3) → (4.3.4) → (4.3.5) → (5.3.1) → (5.3.2) → (5.3.3) → (5.3.4) → (5.3.5) → (6.3.1) → (6.3.2) → (6.3.3) → (6.3.4) → (6.3.5) → (1.3.1) →**(1.3.2) → (1.2.1) → (1.2.2) → (1.1.1) →(2.1.1)**

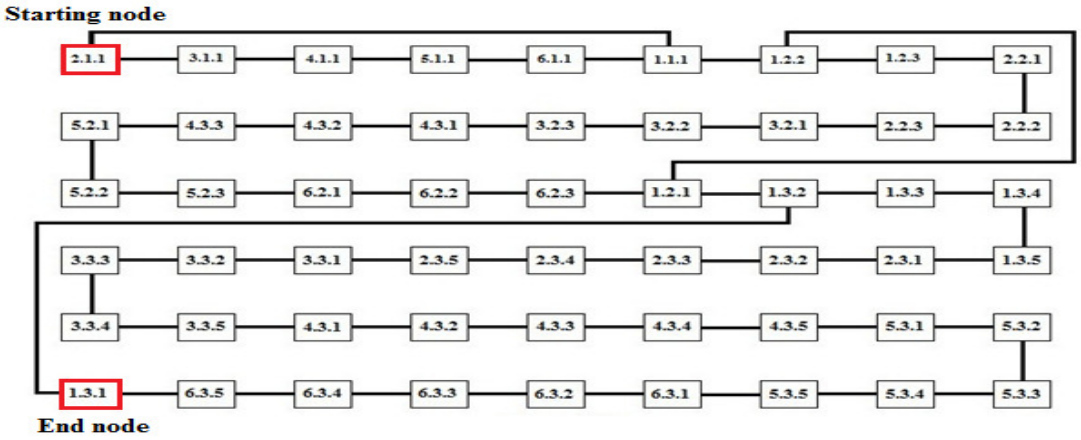

Figure 12: sequence of applying Algorithm A

3. 3 Algorithm (B)

The second algorithm for embedding Bus into Hex-Cell topology is based on three phases: First, moving from the starting node to the one located at the next upper level. Second, moving clockwise until reaching the last node in that level where a direct link to the lower level should be observed. Third, move either clockwise or counter clockwise depending on the level number. This process continues until the last node in that level is reached. This also should produce a direct link to the lower level. To embed Ring topology into Hex-Cell, we just link the endpoints together. (see: Figure 13).

Step 1 : The start node can be any node in level number one.
Step 2 : Add the address of the start node to Node_Queue and mark it as checked node.
Step 3 : Select the node in the upper level and add it to the Node_Queue and mark it as checked node.
Step 4 : Repeat this step until max level is reached
If node has a direct link to upper level, then do step 3.
Else If (Max level - Current Level) is odd select next node clockwise and add it to the Node_Queue and marked it as checked node.
Else select next node counter clockwise and add it to the Node_Queue and marked it as checked node.
Step 5 : Add the address of the next node clockwise to Node_Queue and marked it as checked node.
Step 6 : Repeat step 5 until all nodes in that level are checked and the last node should have a direct link to the lower level.
Step 7 : Select the node in the lower level and add it to the Node_Queue and marked it as checked node.
Step 8: If (Max level - Current Level) is odd select next node counter clockwise add it to the Node_Queue and marked it as checked node.
Else select next node clockwise and add it to the Node_Queue and marked it as checked node.
Step 9 : Repeat step 8 until all nodes in that level are checked and the last node should have a direct link to the lower level.

Step 10: If we did not reach first level go back to step 7.

3.4 Algorithm (B) Discussion via Example

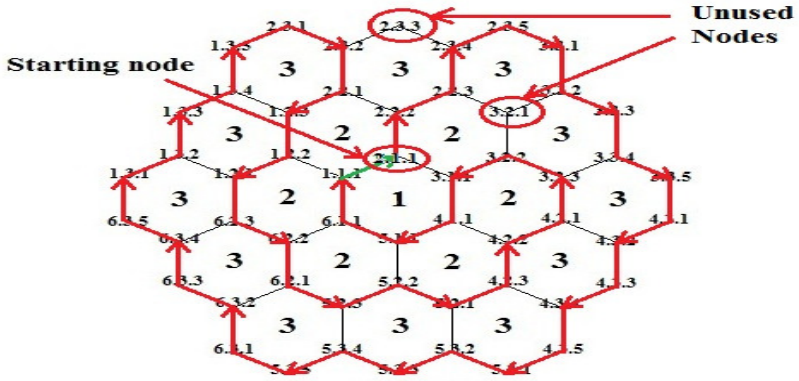

Figure 13: Algorithm B- embedding bus and ring into Hex-Cell topology

Let node (2. 1. 1) be the starting node. An up move to the node (2.2.2) takes place. Then, the (Max level - Current Level) is calculated to determine the direction. Since (Max level - Current Level) = (3 – 2) is odd, a clockwise move to next node (2.2.3) takes place. Then a move to node (2.3.4) at the upper level is performed. Then, a continuous clockwise move is performed until reaching the last node. At that point, a direct link should be produced to the lower level that ends at node (2.3.2). This pattern is continued until no more lower level. Ring embedding is performed just by linking the end points (1.1.1) and (2.1.1). (see: Figure 14).

*Starting node is (2.1.1) → (2.2.2) → (2.2.3) → (2.3.4) → (2.3.5) → (3.3.1) → (3.3.2) → (3.3.3) → (3.3.4) → (3.3.5) → (4.3.1) → (4.3.2) →(4.3.3) → (4.3.4) → (4.3.5) → (5.3.1) → (5.3.2) → (5.3.3) → (5.3.4)→ (5.3.5) → (6.3.1) → (6.3.2) → (6.3.3) → (6.3.4) → (6.3.5) →(1.3.1) → (1.3.2) → (1.3.3) → (1.3.4)→ (1.3.5) → (2.3.1) → (2.3.2) → (2.2.1) → (1.2.3) → (1.2.2) →(1.2.1) → (6.2.3) → (6.2.2) → (6.2.1) → (5.2.3) → (5.2.2) → (5.2.1) → (4.2.3) → (4.2.2) → (4.2.1) → (3.2.3) → (3.2.2) → (3.1.1) → (4.1.1) → (5.1.1) → (6.1.1) → (1.1.1) →(2.1.1)

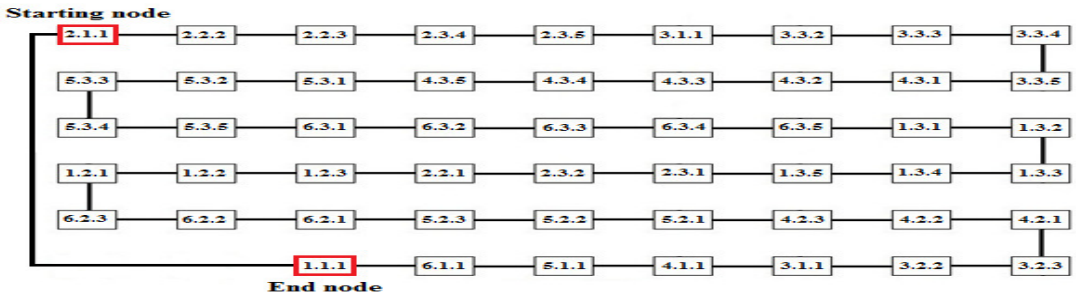

Figure 14: sequence of applying Algorithm B

4. EVALUATION FOR EMBEDDING ALGORITHMS

There are three metrics to evaluate embedding algorithm: Dilation, Congestion and Expansion. Dilation is the maximum number of links in the embedding that is mapped in one link in the original network. In algorithm A, the dilation is equal to {(2 × d) – 1 where d is the network depth}. In algorithm B, it equals to 1. Congestion is the maximum number of edges mapped into one edge in the embedding; here it equals to 1 in both algorithms. Expansion is the ratio of the number of nodes in the embedding algorithm to that in the original network. In algorithm A, the expansion is equal to 1. In algorithm B, it equals to {(N – 2 / N) Where N is the number of nodes in the original network}. (see: Table 1)

Table 1. Algorithms A and B performance evaluation.

Performance Metrics	**Algorithm A**	**Algorithm B**
Dilation	(2 × d) – 1 d : network depth	1
Congestion	1	1
Expansion	1	N-2 / N; N : Number of node in the original network

5. CONCLUSION

In this research, we proposed two algorithms for embedding bus and ring into Hex-Cell topology. The metrics used to evaluate the proposed algorithms are dilation, congestion and expansion. Evaluation results show that the congestion of both proposed algorithms is equal to one; dilation is equal to 2d-1 for the first algorithm and 1 for the second, and the expansion equals to 1 for the first algorithm and equals to (Number of nodes - 2) for the second one.

REFERENCES

[1] Ahmad Sharia, Mohammad Qatawneh, Wesam Almobaideen, and Azam Sleit, (2008) "Hex-Cell: Modeling, Topological Properties and Routing Algorithm", European Journal of Scientific Research, 22(2), 457-468.

[2] Qatawneh M, (2011) "Embedding Binary Tree and Bus into Hex-Cell Interconnection Network", Journal of American Science, 7(12).

[3] Wesam Almobaideen, Mohammad Qatawneh, Azzam Sleit, Imad salah, and Saleh Al-Sharaeh, (2007) "Efficient mapping scheme of ring topology onto tree-hypercubes", Journal of. Applied Sci, 7(18), 2666-2670.

[4] Qatawneh M, (2005) "Embedding linear array network into the tree- hypercube network", European Journal of scientific Research, 10(2): 72-76.

[5] Mohammad Qatawneh, (2011) "Multilayer Hex-Cells: A New Class of Hex-Cell Interconnection Networks for Massively Parallel Systems", Int. J. Communications, Network and System Sciences, 4, 704-708.

[6] Yang M., Jimmy J, and Hsu L. (2007) "Hamiltonian circuit and linear array embeddings in faulty k-ary n-cubes", Journal of Parallel Distributed Computing, 67: 362- 368.

[7] Tsai C. (2004) "Linear array and ring embeddings in conditional faulty hypercubes", Elsevier. Theoretical Computer Science, Vol 314, issue 3, Pages 431–443.

[8] Day K. (2004) "The conditional node connectivity of the k- ary n-cube", Journal of Interconnection Networks, 5 (1): 13–26.

[9] Baril J. and Vajnovzki V, (2005) "Minimal change list for Lucas strings and some graph theoretic consequences", Elsevier. Theoretical Computer Science, 346:189-199.

[10] V. heun and E. W. Mayr, (2002) "Efficient dynamic embedding of binary trees into hypercubes", Journal of Algorithms, vol. 43, no. 1, pp. 51-84.

[11] Mohammad Qatawneh, Hamed Bdour, Shrouq Sabah, Rana Samhan, Azzam Sliet, Ja'far Alqatawna, Wesam al-Mobaideen. (2011) "An alternative Routing algorithm for hex-Cell network", Information – An International Interdisciplinary Journal, Vol. 14, no. 10, pp. 3499-351.