International Journal of Future Generation Communication and Networking
Vol. 8, No. 2 (2015), pp. 295-306
http://dx.doi.org/10.14257/ijfgcn.2015.8.2.24

New Routing Algorithm for Hex-Cell Network

Qatawneh Mohammad and Hebatallah Khattab

Department of Computer Science, King Abdullah II School for Information Technology,
The University of Jordan
mohd.qat@ju.edu.jo
Department of Computer Science, King Abdullah II School for Information Technology,
The University of Jordan
heba_tallah@yahoo.com

Abstract

Hex-Cell network is one of modern interconnection networks in which the nodes are connected with each other in hexagonal topology. This topology gives its network an attractive characteristic represented by expandability due toits recursive structure. In this paper a new routing algorithm for Hex-Cell network is developed depending on new addressing mode.According to this mode, each node in the Hex-Cell is identified by its level and the node number in that level.Consequently, there is no need to readdress the nodes when extra level or levels are added to the topology.Several experiments were conducted to evaluate the proposed algorithm and compare it with other routing algorithms. The results showed the superiority of the new routing algorithm over the other routing algorithms for Hex-Cell in terms of execution time.

Key words: *Hex-Cell network, addressing mode, routing algorithm*

1. Introduction

In distributed systems, the overall performance of the network, basically, depends on the topology of the network and the used routing algorithm. Hex-Cell network which was proposed in [1] has a useful topology that facilitates the embedding mechanism of number of interconnection networks such as bus, ring, binary tree, mesh, and multi 3D hypercube [2-3], [7-12]. Visa versa, Hex-Cell network was embedded in other topologies such as Tree-Hypercube [4].Furthermore, in [5], a fault tolerance routing algorithm for Hex-Cell network was introduced. Despite the topology, it has a limitation of supposing that the network must be fixed, which means, wherever there is a need for adding new level or levels to the network, the nodes addresses must be readdressed because each node in the HC is identified by a pair (X.Y), where X denotes the line number in which the node exists, and Y denotes the location of the node in that line. The necessity of the readdressing, actually, comes from the idea of that the routing algorithm in [1] depends on the depth of the network in computing the nodes' addresses during the message transmission trip. So, the Hex-Cell network scalability and flexibility are considered to be low.

To solve the issue of readdressing, a new addressing mode was presented in [6]eliminates the dependability on the depth of the network by labeling each cell in the network by the ordered pair (X.Y), where X denotes the horizontal location, and Y denotes the vertical one. In fact, that labeling process was used to address the nodes. A node address consists of three parts (L, C, R) which are called (Left cell, Center cell, Right cell). Actually, this addressing mode is not being applicable for the nodes in the border level because one part of the node address will be missing. However, this problem

ISSN: 2233-7857 IJFGCN

was treated by assuming the existence of a virtual level around the last one to give the ability of completing the nodes addressing of the last actual level.

In this paper, a new routing algorithm is developed using new addressing mode that deals with levels and nodes number. The network levels (L) are labeled from inner to outer one starting from 1 up to N(number of levels).The nodes in each level are labeled starting from 0 up to (6(2L-1)-1) [1]. Indeed, the proposed algorithm could overcome the limitations in both [1] and [6].

The rest of this paper is organized as the following. Section 2 gives a brief description of the Hex-Cell network. A review of the related works is introduced in Section 3. A full explanation of the proposed algorithm can be found in Section 4. Section 5 is used to discuss the proposed algorithm using numbers of examples for the routing probable cases. In section 6 the results of the conducted experiments are presented. The conclusion and future works are left for Section 7.

2. Definition of Hex-Cell network

As it is defined in [1], HC (d) denotes a Hex-Cell network with depth d.actually; d indicates the number of levels in the network which are constructed using sequence of hexagonal cells, each of six nodes. These levels are numbered from 1 up to d, where level 1 represents the innermost level that consists of one hexagon cell. The next 6 hexagon cells which surround the innermost one refers to Level 2. As illustrated in Fig (1), Level 3 is formed by covering Level 2 with other 12 hexagon cells. The number of nodes in each level (L) can be calculated by 6(2L-1).

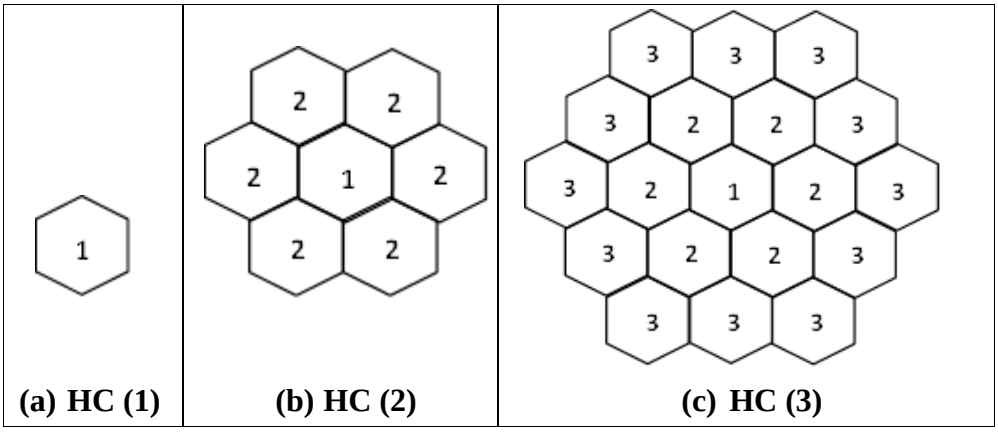

Figure 1. Hex-Cell Network with Depths (a) 1 (b) 2 (c) 3 as in [1]

3. Related Works

In [1], Hex-Cell network was developed and its nodes addressing mode was described as shown in Figure 2. The overall network with depth d was numbered as lines from 1 to 2d. Each node was addressed as ordered pair (X.Y), X was the line number, and Y was the node number in the X line. For example, node (1.1) was the first node in the first line, (3.10) was the node number 10 in the third line, and so on.

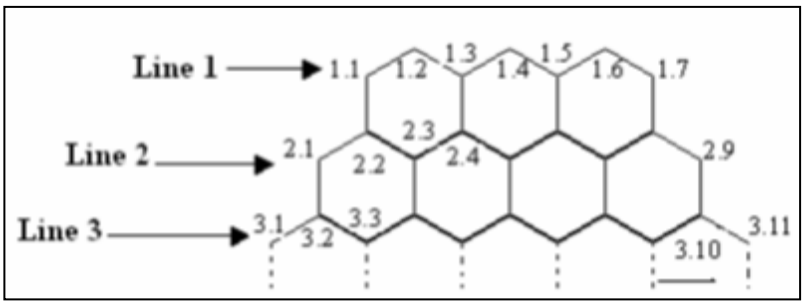

Figure 2. Addressing Mode for Hex-Cell Network as in [1]

In the introduced routing algorithm for Hex-Cell network which was built using the above addressing mode, the message from source to destination had three choices for moving, move Up, move Down, and move Horizontal. The decision of which movement the message should follow depended on three cases, the first one when (X_s>X_d) then the message followed move Up movement with one of two sub directions, move Up/Left-to-Right, and move Up/ Right-to-Left. The second case when (X_s<X_d) then move Down must be followed with also one of other two sub directions, move Down/Left-to-Right and move Down/Right-to-Left. The last one when (X_x = X_d) then the message must use move Horizontal movement. As in the previous two cases, there were two directions, move Horizontal/Left-to-Right and move Horizontal/Right-to-Left.

In [6], to eliminate the dependability of the original algorithm [1] on the network depth, an alternative routing algorithm was developed depending on new addressing mode as shown in Figure 3. In that addressing mode, the address of each node was related to the surrounded three cells which were, firstly, identified as (X.Y), where X denotes the horizontal location, and Y denotes the vertical one. Consequently, the address of each node, (L, C, R), consisted by concatenating the addresses of its Left cell (L), Center cell (C), and Right cell (R) as [(XL.YL), (XC.YC), (XR.YR)]. Figure 4 gives examples for nodes addresses. In fact, that addressing mode was not applicable for the outer level since each node in that level would miss one part of its address depending on the its location. To treat that problem, a virtual level was supposed to surround the outer level of the network.

According to the routing algorithm which was constructed using the above described addressing mode, it was designed to consist of two parts; the first part was called “Cell Routing” which routes the message from cell to cell in the network. The other part which was called “Node Routing” routes the message from node to node within the same cell.

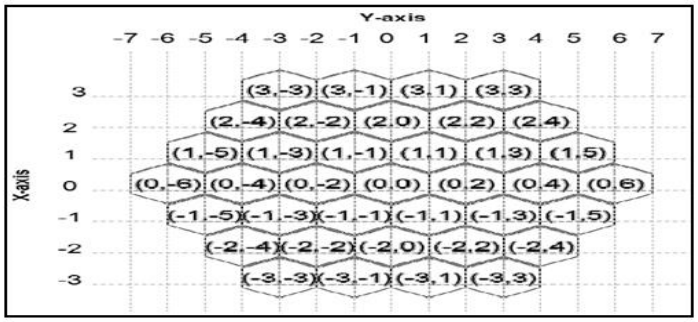

Figure 3. Addressing Mode for Hex-Cell Network as in [6]

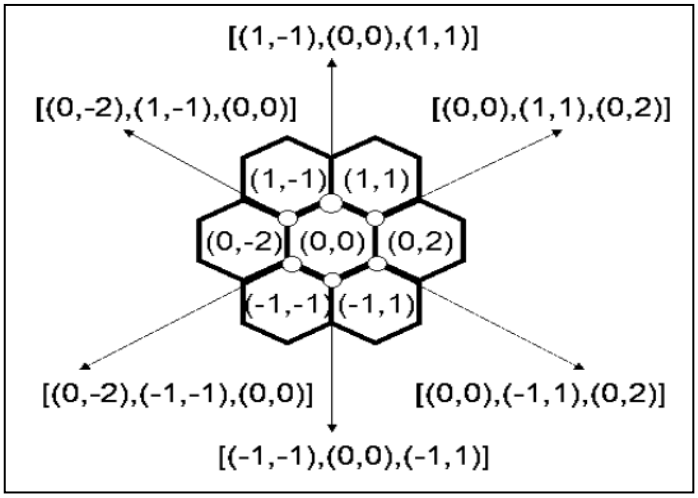

Figure 4. Example for Node Addressing as in [6]

4. Proposed Routing Algorithm for Hex-Cell Network

The original routing algorithm which depends on the depth of the network to compute the addresses of the nodes during message transmission is preferable when the network is guaranteed to be fixed. However, this is not the usual case in real life, where the networks always increase in their size. To increase the size of the network by adding extra levels will, actually, need extra cost which can be dispensable by using the proposed addressing mode with its routing algorithm. On the other hand, the alternative routing algorithm which depends on the addresses of the network cells to identify the addresses of nodes during message delivering tour, by comparison study, needs more calculations and comparisons which severely increases the delivering time.

In contrary, the proposed routing algorithm does not need the depth of the network to calculate the addresses of the nodes and, by the same comparison study, it needs less calculations and comparisons than the alternative one which consequently gives less running time.

Section 4.1 explains the proposed addressing mode. In section 4.2, the proposed routing algorithm is introduced.

4.1. Proposed Addressing Mode

According to the proposed addressing mode, the levels of a network with depth d are numbered from 1 for the innermost level to d for the outermost level. Each node in each level is identified by the ordered pair (L, N) where L denotes the levelnumber, and N denotes the node number in the L level.The total number of nodes in each level$L(N_L)$ is equal to 6(2L-1)[1]. Thus, in each level, the node numbering is started from 0 up to N_L-1. For example, node (1, 0) is the first node in level 1, node (3, 10) is the node number 11 in the third level, and so on. In Fig (5), a sketch for the new addressing mode is shown.

4.2. Proposed Routing Algorithm

First of all, and after describing how the levels and nodes are numbered, to understand how the movements from source node to destination node are executed, it is important to be known that there is a hidden grouping for the nodesin the Hex-Cell network in a total of six sections as shown in Figure 6.

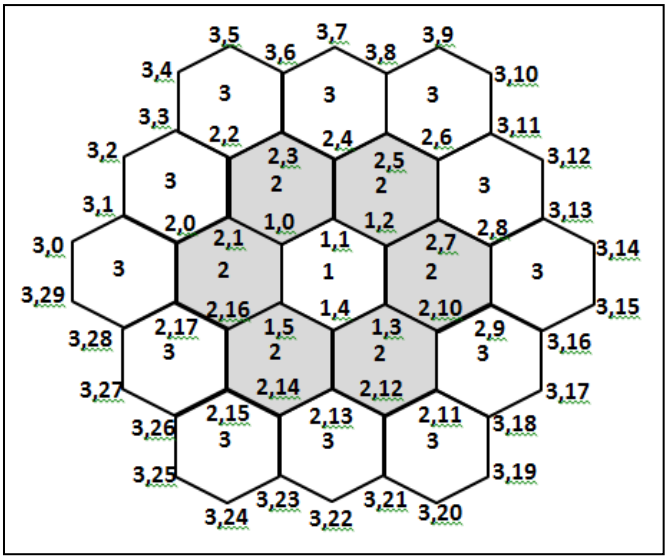

Figure 5. Proposed Addressing Mode

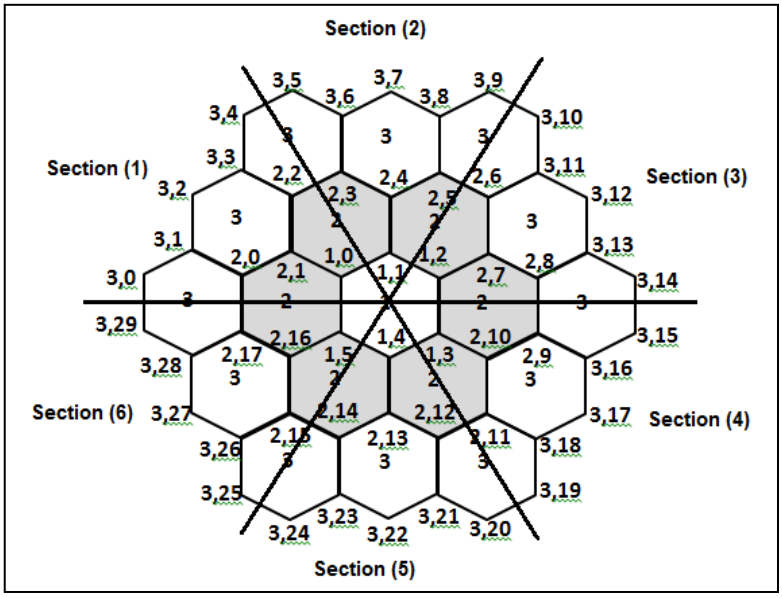

Figure 6. Proposed Addressing Mode with Sections

Now, as it is shown in Figure 7, the movement from source node to destination node can be done using one of the following three main cases:

CASE 1: if (SL = DL) then move Same Level (SL, SN, DL, DN), Figure 8.

CASE 2: if (SL > DL) then move In (SL, SN, DL, DN), Figure 9.

CASE 3: if (SL < DL) then move Out (SL, SN, DL, DN), Figure 10.

The mentioned three cases also have subcases when the source and destination are in same section or in different sections. These cases will be explained by examples in Section 5.

```
Route( S_L, S_N, D_L, D_N)  //Main
    if (S_L= D_L&& S_N =D_N) then
       Destination-Reached
    else
       if (S_L=D_L) then
moveSameLevel(S_L, S_N,  D_L, D_N)
       else
           if (S_L<D_L) then
Dsec = D_N/(2D_L-1)+1
HalfD_N= Dsec(2D_L-1)-(2D_L-1)/2 -1
              If (D_N= HalfD_N) then
tempD_N= Dsec(2S_L-1)-(2S_L-1)/2 -1
              else
                  If (D_N>HalfD_N) then
tempD_N= Dsec(2S_L-1) -1
                  else
tempD_N= Dsec(2S_L-1) -(2S_L-1)
moveSameLevel(S_L, S_N, S_L, tempD_N)
moveOut(S_L, tempD_N, D_L, D_N)
          else  // S_L>D_L
Ssec = S_N/(2S_L-1)+1
HalfS_N= Ssec(2S_L-1)-(2S_L-1)/2 -1
              If (S_N= HalfS_N) then
tempD_N= Ssec(2D_L-1)-(2D_L-1)/2 -1
              else
                  If (S_N>HalfS_N) then
tempD_N= Ssec(2D_L-1) -1
                  else
tempD_N= Ssec(2D_L-1) -(2D_L-1)
moveIn(S_L, S_N, D_L, tempD_N)
moveSameLevel(D_L, tempD_N, D_L, D_N)
```

Figure 7. Main Routing Algorithm

```
moveSameLevel(S_L, S_N, D_L, D_N)
if (S_N=D_N) Destination-reached
else
Ssec = S_N/(2S_L-1)+1, Dsec = D_N/(2D_L-1)+1, Sdiff = Dsec-Ssec
if (S_L=1)
        if ((1 <= diff <=3)||(-5<=diff <=-4))
           if (S_N =5) S_N =0  else S_N= S_N+1
        else
           if (S_N=0)  S_N=5 else S_N= S_N-1
     moveSameLevel(S_L, S_N, D_L, D_N)
else
if (Sdiff=0)
if (diff>0) S_N= S_N+1  else S_N= S_N-1
          moveSameLevel(S_L, S_N, D_L, D_N)
else
if (Sdiff=3||Sdiff=-3)
             tempDN1 = S_N/(2S_L-1), tempDN2=D_N/(2D_L-1)
          moveIn(S_L, S_N, 1, tempDN1)
          moveSameLevel (1, tempDN1, 1, tempDN2)
          moveOut(1, tempDN2,D_L, D_N)
```



```
        else
            if (Sdiff=2||Sdiff=-4)
        SS=Ssec,Dec=2Ssec-1,temp_L =S_L,temp_N =S_N
         while (SS=Ssec)
        if (((Ssecis even) && (SNis even)) || ((Ssecis odd) && (SNis
  odd)))
                temp_L =temp_L-1,  temp_N= temp_N -Dec
        else
  if (temp_N=6 (2temp_L -1)-1) temp_N=0 else temp_N = temp_N +1
        SS =temp_N/(2temp_L-1)+1
        N1_L=temp_L  N1_N= SS (2temp_L-1)-1)
        if(N1_N=6(2temp_L-1)-1)
        N2_L=temp_L, N2_N= 0 elseN2_L=temp_L, N2_N= N2_N+1
                moveSameLevel(temp_L, temp_N, N1_L, N1_N)
  moveOut(N2_L, N2_N, D_L, D_N)
        else
        if (Sdiff=-2||Sdiff=4)
        SS=Ssec,Dec=2Ssec-1, temp_L =S_L, temp_N =S_N
         while (SS=Ssec)
        if (((Ssecis even) && (S_Nis even)) || ((Ssecis odd) && (S_Nis odd)))
                temp_L =temp_L-1, temp_N= temp_N -Dec
                                else
                if (temp_N=0) temp_N=6 (2temp_L -1)-1 else temp_N = temp_N-1
        SS =temp_N/(2temp_L-1)+1
                N1_L= temp_L, N1_N=SS(2temp_L-1)-(2temp_L-1)
                if(N1_N=0)
                N2_L=temp_L, N2_N=6(2temp_L-1)-1
                  else
N2_L=temp_L, N2_N= N2_N -1
        moveSameLevel(temp_L,temp_N,N1_L, N1_N)
moveOut(N2_L, N2_N, D_L, D_N)
        else
           if(Sdiff=1||Sdiff=-5)
if (S_N =6(2S_L-1)-1) S_N=0 else   S_N= S_N +1
             else
if (S_N=0) S_N=6 (2S_L-1)-1 elseS_N= S_N-1
        moveSameLevel(S_L, S_N, D_L, D_N)
```

Figure 8. Move Same Level Function

```
move In(S_L, S_N, D_L, D_N)
   if (S_L= D_L&& S_N =D_N)Destination-Reached
   else
Ssec=S_N/(2S_L-1)+1, Dec=2Ssec-1, Sym= S_N -Dec × (S_L-D_L),  PlusMinus = -1
      if ([(Ssec is even) && (S_N is even)] || [(Ssec is odd) && (S_N is odd)]) then
moveIn(S_L-1, S_N-Dec, D_L, D_N)
      else
          if (D_N>= Sym) then  PlusMinus = 1
moveIn(S_L, S_N+ PlusMinus, D_L, D_N)
```

Figure 9. MoveInfunction

moveOut(S_L, S_N, D_L, D_N)
if (S_L= D_L&& S_N =D_N)Destination-Reached
else
Ssec=S_N/(2S_L-1)+1, Inc=2 Ssec -1, Sym=Inc × (D_L-S_L)+S_NPlusMinus = -1
if ([(Ssec is even) && (S_N is odd)] || [(Ssec is odd) && (S_N is even)]) then
moveOut(S_L+1, S_N+Inc, D_L, D_N)
else
if (D_N>= Sym) then PlusMinus = 1
moveOut(S_L, S_N+ PlusMinus, D_L, D_N)

Figure 10. Move Out Function

5. Discussion via Examples

Several cases for node movements will be described by the following examples. All examples will use the Hex-Cell network that is represented in Figure 6.

Move Same Level /Same Section

Let (3, 0) be a source node, and (3, 4) is a destination node. By applying the proposed algorithm, the routing path will be:(3,0)→(3,1)→(3,2)→(3,3)→(3,4), Figure 11.

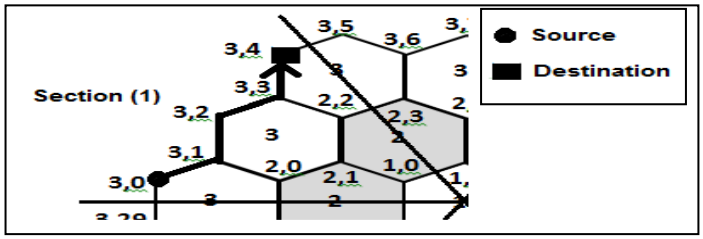

Figure 11. Move Same Level / Same Section

Move Same Level / Different Sections

To move from source node (3, 3) to destination one (3,12), the proposed routing algorithm output will be as follows after running move Same Level() function.
(3,3)→(2,2)→(2,3)→(2,4)→(2,5)→(2,6)→(3,11)→(3,12), Figure 12.

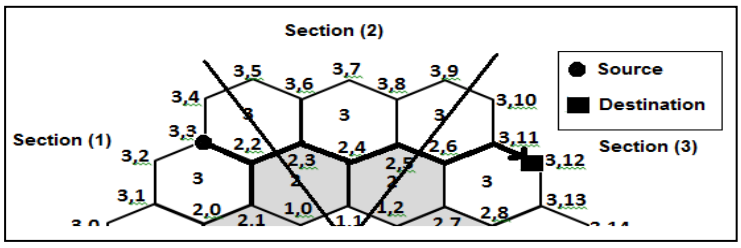

Figure 12. Move Same Level / Different Sections

Move In / Same Section

If a message is wanted to be sent from source node (3, 27) to destination node (1, 5), a move In() function will be applied resulting the following trip path.(3,27)→(3,28)→(2,17)→(2,16)→(1,5), Figure 13.

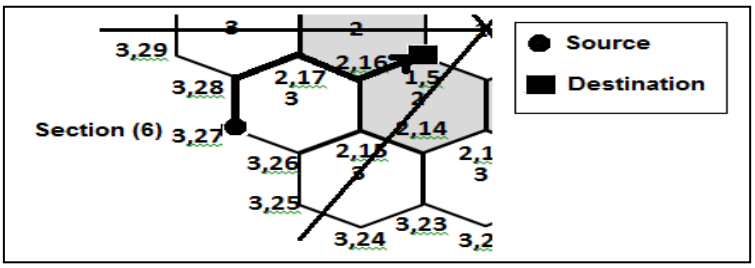

Figure 13. Move In / Same Section

Move In/ Different Sections

If nodes (3, 4) and (2, 7) are the source and destination nodes respectively, CASE 2 will be trigger because 3<2. The route in which the message will travel through is as follows:(3,4)→ (3,3)→ (2,2)→ (2,1)→ (1,0)→(1,1)→(1,2)→(2,7), Figure 14.

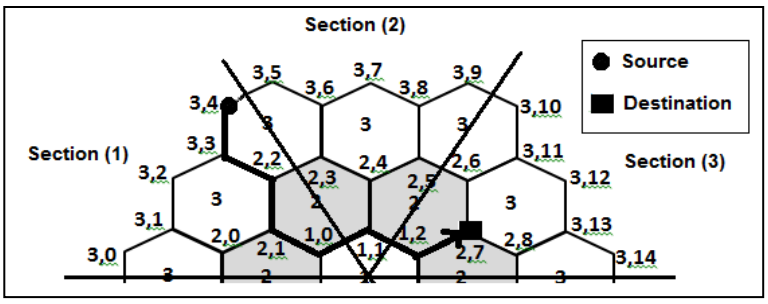

Figure 14. Move In / Different Sections

Move Out/ Same Section

Because node (1,2) and node (3, 10) locate in the same section, to transmit a massage from (1,2) to (3,10), it has to follow the next path.(1,2)→ (2,7)→ (2,6)→ (3,11)→ (3,10), Figure 15.

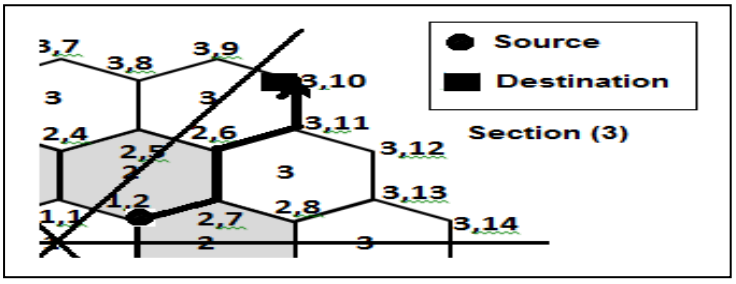

Figure 15. Move Out / Same Section

Move Out/ different sections

The following listed nodes are, indeed, the sequence of nodes that any message must go through when it starts from node (2,6) to reach node (3,25) according to the proposed routing algorithm.
2,6)→(2,7)→(1,2)→(1,3)→(1,4)→(1,5)→(2,16)→(2,15)→(3,26)→(3,25), Figure 16.

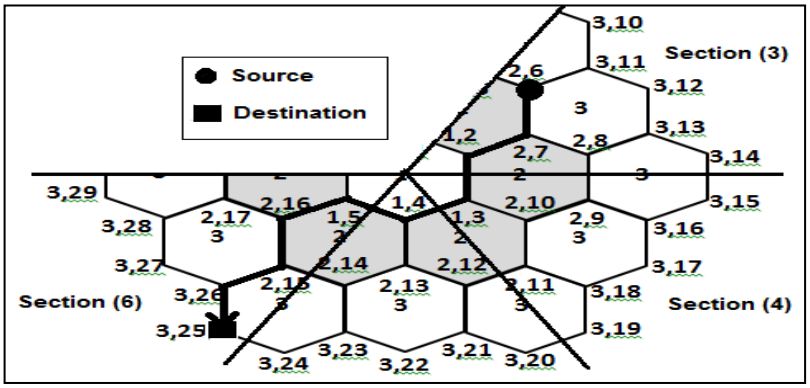

Figure 16. Move Out / Different Sections

6. Experimental Results

The execution of the proposed algorithm is investigated in this section. Its performance is compared with both the original algorithm [1] and the alternative one [2]. Basically, the execution time is taken as a performance measure because all of them give the shortest path (i.e. number of hops that the message is transferred through). All algorithms were programmed in JAVA language on a 2.40 GHz PC. Five different depths for Hex-Cell network were taken as a changeable variable in the conducted experiments. These depths were 100, 150, 200, 250, and 300. For each depth, the maximum number of hops that can a message go through is taken in the consideration when the source and destination nodes were selected. The experimental results which are represented graphically in Figure 17 are shown in Table 1. Furthermore, each result listed in Table (1), is the average of 10 tries in that case.

Table 1. Experiments Results for the Proposed Algorithm and other Algorithms

		Original Alg.[1]	Alternative Alg.[6]	Proposed Alg.
Network Depth	**Number of Hops**	**Execution Time (ms)**		
100	398	15	315	15
150	598	83	543	58
200	798	128	926	77
250	998	164	1116	108
300	1198	197	1320	135

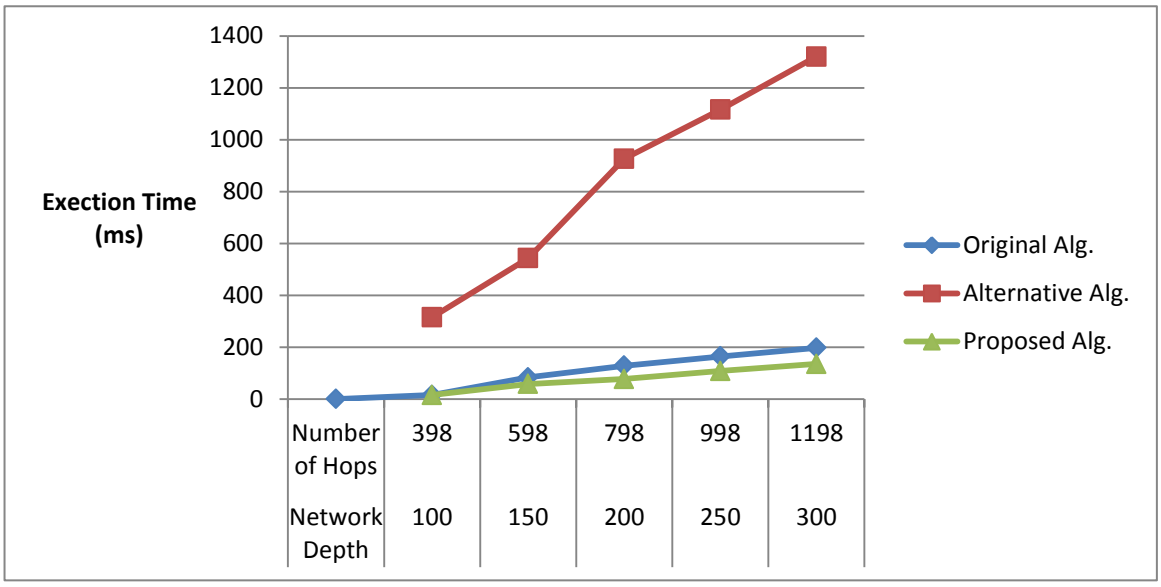

Figure 17. Execution Time for Proposed, Original [1], and Alternative [6] Algorithms

6.1. Results Discussion

In fact, it is obvious from Table 1and Figure 17 that the proposed algorithm has the superiority over the other two algorithms. It is clear that the proposed algorithm and the original one have closed results in all cases. On the other hand, there is clear difference with the alternative one.

7. Conclusion and Future Work

In this paper, a new routing algorithm has been proposed for Hex-Cell network. It provides the shortestpath from source to destination based on a new addressing mode for the network nodes. The proposed algorithm could overcome the limitation of the original algorithm which uses the network depth to compute the nodes addresses during the message trip from source to destination. Actually, the original one is suitable for the networks which have fixed size because adding new level to the network will need extra cost for reconfiguring the network by readdressing its nodes. On the other hand, the proposed algorithm allows continuous adding for the network levels without any need for reconfiguration. That means the proposed algorithm increases the network scalability in a good manner. An alternative routing algorithm also compared with the proposed one. Although the alternative algorithm could eliminate the dependability on the network depth for node address calculation, it has a huge gap in the execution time between it and the proposed algorithm which is proved experimentally. In future, the next steps are to develop other algorithms for embedding some other network topologies as ring and bus in Hex-Cell network using the proposed algorithm. Step further; a fault tolerance algorithm must be implemented to gain a complete proficiency of Hex-Cell network.

References

[1] A. Sharieh, M. Qatawneh, W. Almobaideen, and A. Sleit, “Hex-Cell: Modeling, Topological Properties and Routing Algorithm, European Journal of Scientific Research, vol. 22, no. 2, **(2008)**.
[2] M. B. Younes, “A Study of Mapping Some Popular Schemes into the Hex-Cell.Journal of Applied Sciences Research” vol. 5, no. 8, **(2009)**.
[3] M. Qatawneh, “Embedding Binary Tree and Bus into Hex-Cell Interconnection Network”, Journal of American Science, vol. 7, no. 12, **(2011)**.

[4] A. Qatawneh, "Embedding Hex-Cells into Tree-Hypercube Networks", IJCSI International Journal of Computer Science Issues, **(2013)**.
[5] M. Qatawneh, B. Hamed, W. Almobaideen, A. Sleit, A. Oudat, W. Qutechat, and R. Al-soub, "FTRH: Fault Tolerance Routing Algorithm for Hex-Cell Networks", IJCSNS International Journal of Computer science and Network security, vol. 9, no. 12, **(2009)**.
[6] M. Qatawneh, H. Bdour, S. Sabah, R. Samhan, A. Sliet, J. Alqatawna, and W. Al-Mobaideen, "An alternative Routing algorithm for hex-Cell network", Information-An International Interdisciplinary Journal, vol. 14, no. 10, **(2011).**
[7] M. Qatawneh, "Adaptive fault tolerant routing algorithm for Tree-Hypercube multicomputer", Journal of computer Science, vol. 2, no. 2, **(2006)**.
[8] J. H. Bahn, J. Yang and N. Bagherzadah, "Parallel FFT algorithms on network-on-chips", Proceedings of the 5th International Conference on Information Technology, New Generation, **(2008)** April.
[9] J. H. Youn, B. Bose, and S. Park, "Fault-Tolerant routing algorithm in meshes with solid faults", Journal of supercomputing, vol. 3, no. 7, **(2006)**.
[10] J. Zhou and C. M. Francis, C. M. Lau, "Multi-phase minimal fault-tolerant wormhole routing in meshes.Parallel Computing. 30, no. 3, **(2004)**.
[11] C. Decayeux and D. Seme, "3D hexagonal network: modeling, topological properties, addressing scheme, and optimal routing algorithm", IEEE Transactions on Parallel and Systems, vol. 16, no. 9, **(2005)**.
[12] M. Qatawneh, A. Alamous and J. Alqatawna, "Section Based Hex-Cell Routing Algorithm (SBHCR)", International Journal of Computer Networks and Communications, vol. 7, no. 1, **(2015)**.

Authors

Prof. Mohammad Qatawneh, is the dean of the King Abdullah II School for Information Technology (KASIT) at the University of Jordan. He received his Ph.D. in computer from KievUniversity in 1996. Prof. Qatawneh published thirty papers in the areas of Networks, Parallel algorithms and Embedding systems. His research interests include Parallel Computing, Interconnection Networks and Routing Algorithms.

Hebatallah Khattab, is a PhD Student in the Computer Science Department at the King Abdullah II School for Information Technology (KASIT) in The University of Jordan, She received her Master and Bachelor degrees from The University of Jordan in 2006 and 2001 respectively.

