Alternatives of XY-Routing for Mesh

Lalit Kishore Arora
Ajay Kumar Garg Engg College, Ghaziabad, U. P., India

Raj Kumar
Gurukul Kangri University, Haridwar, U. K., India

ABSTRACT

The XY-Routing is most popular routing algorithm in the Mesh interconnection networks. This paper propose two new alternatives for XY routing algorithms, indexed-based and level-based routing, which can be easily replace by XY routing.

Keywords: Source routing, Mesh, Torus, interconnection networks, minimal path.

1. INTRODUCTION

The routing algorithm is used to find the path between any two nodes. There are several routing algorithms for interconnection network has been proposed. These routing algorithms are heavily dependent of the exact structure of the network topology, and cannot be transferred easily to other topologies.

XY routing is the deterministic source routing algorithm for the regular topologies, The majority of network topologies for XY-routing in multicomputers are meshes, particularly low-dimensional meshes. Low-dimensional meshes are generally preferable because they have low, fixed node degrees, which makes them more scalable than high-dimensional meshes and k-ary n-cubes. They also have fewer channels and higher channel bandwidth per bisection density, which results in lower communication latencies [1].

With deterministic routing algorithms, a packet's route through the network is decided as it is inserted into the network by the source [1].

In keeping with the trend toward low-dimensional meshes, this paper examines 2D meshes . A 2D mesh has m×n nodes, m along the x dimension by n along the y dimension, where $m \geq 2$ and $n \geq 2$.

Algorithms for routing message packets through a network topology should have three character- istics: low communication latency, high network throughput, and ease of implementation.

In other words, the routing algorithms should provide packet transmission that is high in quality, quantity, and practicability [1]. Features contributing to ease of implementation are little hardware for channels, buffers, and control logic. Features contributing to low latency and high throughput are freedom from deadlock, freedom from livelock, freedom from indefinite postponement, fault tolerance, routing packets along short paths, spreading packet traffic evenly, and routing packets adaptively [1]. Of these features, the most important is freedom from deadlock. Deadlock can keep many or all packets from reaching their destinations and occurs readily unless a routing algorithm includes preventive measures. Adaptiveness is important, too, because it contributes to several of the other features [1]. It provides alternative paths for packets that encounter continuously blocked channels, faulty hardware, or hotspots in traffic patterns. A minimal routing algorithm routes packets exclusively along a shortest path. Algorithms that can route packets along longer paths are called non-minimal. Unless otherwise specified, all routing algorithms proposed d in this paper are minimal.

This paper proposed two minimal routing in 2D meshes which are substitute of XY routing. Section 2 reviews XY routing algorithm for 2D meshes. Section 3, describe the indexed-based routing algorithm, and Section 4 presents the level-based routing algorithm for 2D meshes. Section 5, evaluate the performance of both the routing algorithms with the XY routing algorithm.

2. RELATED WORK

In [2] paper, author proposed a novel systematic approach for designing deadlock-free routing algorithms for torus NoCs. Using this method a new deterministic routing algorithm (called TRANC) is proposed that uses only one virtual channel per physical channel in torus NoCs. he also proposed an algorithmic mapping that enables extracting TRANC-based routing algorithms from existing routing algorithms, which can be both deterministic and adaptive. The simulation results show power consumption and performance improvements when using the proposed algorithms.

Paper [3] aimed to protect the routing path with deadlocking freedom and improve the performance drastically. Author proposed method will increase the availability and dependability of the network and reduce 100 % pocket drops ratio with deadlock freedom.

Paper [4],[5] suggested two new routing algorithms, Layered shortest-path routing [4] (LASH) and Segment-based routing [5] (SR). LASH is a deterministic and topology independent routing algorithm which requires virtual channels. The main advantage of LASH is that it guarantees shortest path routing while only using a modest number of virtual channels. SR is also a deterministic and topology independent routing algorithm, but without the need for virtual channels. SR does not guarantee shortest path routing, but as it does not require virtual channels it is applicable to a wider range of network technologies. The main strengths of SR is its locality independence property that gives us a large degree of freedom when enforcing routing restrictions. This freedom also makes it possible to exploit the regularity present in regular and semi-regular topologies.

3. XY ROUTING ALGORITHM

The XY-routing is the most popular routing algorithm for mesh like networks. In order to understand XY-routing, consider the two-dimensional nxn Mesh. Every node in this mesh has a location in the form of (x, y) where x represents its position in the x-dimension and y represents its position in the y-dimension, Figure-1.

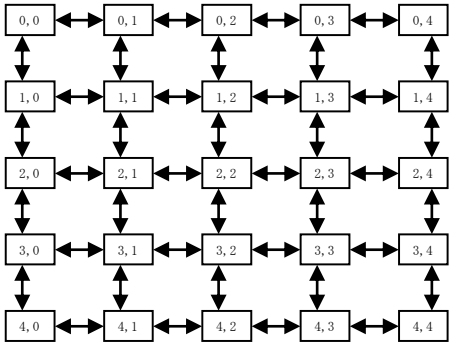

Fig. 1. Mesh with 2D addressing

The XY-routing algorithm works as follows:

Given a packet with source-destination pair ((x1, y1) to (x2,y2)), first route the packet along the x-dimension from (x1, y1) to (x2, y1) and then along the y-dimension from (x2, y1) to (x2, y2).

This is certainly an oblivious routing algorithm, since the path of a packet only depends on its source and destination locations.

The pseudo-code for the XY-routing is given below, which found in several literature:

```
int XY_routing(int xcurrent,int ycurrent,int xdest,int ydest)
    {
     int xoffset=xdest-xcurrent;
     int yoffset=ydest-ycurrent;
     if(xoffset<0) return(ms[xcurrent-1][ycurrent]);
     if(xoffset>0) return(ms[xcurrent+1][ycurrent]);
     if(xoffset==0 && yoffset<0)
          return(ms[xcurrent][ycurrent-1]);
     if(xoffset==0 && yoffset>0)
           return(ms[xcurrent][ycurrent+1]);
     if(xoffset==0 && yoffset==0)
          return(ms[xcurrent][ycurrent]);
    }
```

In above XY-routing, ms[x][y] is taken as 2D array for mesh where node numbers are stored on its x, y positions. As shown in the algorithm, it finds the offset of both node locations, and compare them to move forward and backward directions. This routing algorithm returns next node of the path form source to destination node.

4. INDEXED-BASED ROUTING ALGORITHM

The most popular XY-routing, find offsets for both direction and move accordingly. Here we present a new routing technique to achieve same path. In this method we directly checks the x and y directions to move along the in x or y-dimension.

The pseudo-code for the Indexed- based routing (IB-routing) is given below:

```
int IB_routing (int xcurrent,int ycurrent,int xdest,int ydest)
    {
      if (xcurrent < xdest) return (ms[xcurrent+1][ycurrent]);
      if (xcurrent > xdest) return (ms[xcurrent-1][ycurrent]);
      if (ycurrent < ydest) return (ms[xdest][ycurrent+1]);
      if (ycurrent > ydest) return (ms[xdest][ycurrent-1]);
    }
```

The Level-based routing algorithm save the time to calculate offset values and also save the time for checking these values with multiple conditions.

5. LEVEL-BASED ROUTING ALGORITHM

Above both routing algorithms based on the (x,y) position of the source and destination nodes. But here we proposes a new routing algorithm where we directly consider the node numbers from 0 to onwards, which assigned to each node according to its position, figure-2.

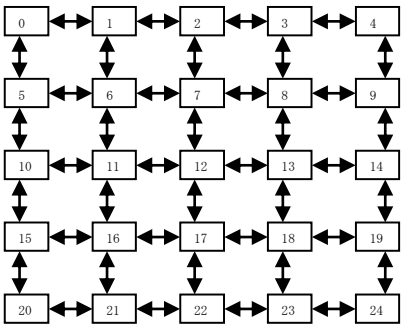

Fig. 2. Mesh with node numbering

In this routing algorithm, first we find the level of both source and destination nodes, and compare them to move forward or backward along the x or y-dimension.

The pseudo-code for the Level-based routing (LB-routing) for nxn Mesh is given below:

```
int LB_routing(int curr_node,int dest_node)
    {
      int level1=curr_node/n;
      int level2=dest_node/n;
          if(level1==level2)
           {
            if(curr_node<dest_node)
                        return(curr_node+1);
             else
                        return(curr_node-1);
           }
          if(level1<level2)
                        return(curr_node+n);
          else
```

```
        return(curr_node-n);
    }
```

The above Level-based routing algorithm works for nxn Mesh network, here n is the length of Mesh in y-dimension. LB-routing not uses any array to fetch the next node number, it directly calculates the next node number after comparing the level and node numbers.

6. PERFORMANCE ANALYSIS

All the above algorithms are used to find the path in 2D Mesh between any two nodes. To analyze the performance of these algorithms, we have to emphasis on the coding part of the algorithms. The number of calls of each routing algorithm depends on the number of nodes exists in the path between the source and destination nodes.

In the XY-routing algorithm, the number of comparison for finding next node are 5 times in the worst case while in the Indexed-based routing the comparisons are 4 and in Level-based routing are only 2 times. In other words, we can say, in the worst case, the running cost of XY-routing is 5 units of time, Indexed-based routing is 4 units of time and Level-based routing is 2 units of time. Therefore Level-based routing is better then Indexed-based routing, and Indexed-based routing is better then XY-routing.

We can see in the table-1, where the performance of each routing algorithm are shown, performance is based on the number of nodes exist in path from source to destination node. Let assume the number of nodes are n in the path, then each algorithm will execute n times.

TABLE 1

COMPARISON IN WORST CASE

Routing Algorithms	Running Cost in Worst Case (in units of time)
XY-routing	5n
Indexed-based routing	4n
Level-based routing	2n

The Figure-3, shows the running cost performance based on the number of nodes exists in the path, here we checks the performance by increasing the number of nodes in path finding by the algorithm.

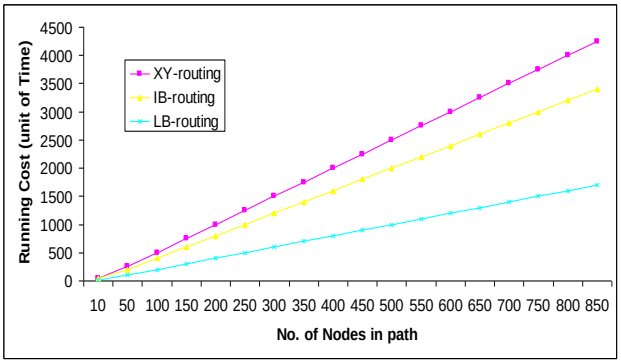

Fig 3. Comparison in worst case

Figure-3 shows that the LB-routing (Level-based) perform better then IB-routing(Indexed-based), and IB-routing perform better then XY-routing when number of nodes are increases in the path.

7. CONCLUSION

In this paper we proposed two new routing algorithms to provide shortest path between any two nodes. These proposed algorithms are basically different versions of XY routing, where we are trying to attract your attention on different point of view to find the shortest path between any two nodes of the Mesh network. The proposed both algorithms can be implements in m×n Mesh too. The result shows that LB-routing is best version of XY routing. In the next step we are going to implement these in Torus network for evaluate the performance.

8. REFERENCES

[1] Christopher J. Glass, Lionel M. Ni, "Maximally Fully adaptive routing in 2D Meshes", Technical Report , MSU-CPS-ACS-51, January 12, 1992.

[2] Dara Rahmati, Hamid Sarbazi-Azad, Shaahin Hessabi and Abbas Eslami Kiasari, "Power-efficient Deterministic and Adaptive Routing in Torus Networks-on-Chip", Microprocessors and Microsystems: Embedded Hardware Design

[3] S. Mohiadeen Abdul Kadhar , T. Revathi, "An Efficient Monarchic Reconfiguration Protocol with Deadlock Freedom on Interconnection Networks", International Journal of Computer Applications, vol-43, issue-9, page 1-6, 2012.

[4] O. Lysne, T. Skeie, S.-A. Reinemo, and I. Theiss. Layered routing inirregular networks. IEEE Transactions on Parallel and Distributed Systems,17(1):51–65, 2006.

[5] Andres Mej´ıa, Jos´e Flich, Jos´e Duato, Sven-Arne Reinemo, and Tor Skeie. Segment-based routing: An efficient fault-tolerant routing algorithm for meshes and tori. In Proceedings of the 20th IEEE International Parallel and Distributed Processing Symposium (IPDPS 2006).IEEE Computer Society, April 2006