The Static and Dynamic Performance of an Adaptive Routing Algorithm of 2-D Torus Network Based on Turn Model

Yasuyuki Miura [1], Kentaro Shimozono [2], Kazuya Matoyama [3], and Shigeyoshi Watanabe [1]
[1] Department of Information Science, Shonan Institute of Technology (SIT), Fujisawa, Kanagawa, Japan
[2]TOP Engineering Inc., Minato-Ku, Tokyo, Japan
[3] Net-Research Co.Ltd., Kawasaki, Kanagawa, Japan

Abstract - *A 2-D torus network is one of the most popular networks for parallel processing. Many algorithms have been proposed based on the turn model, but most of them cannot be applied to a torus network without modification. In this paper, we propose the North-South First Routing (NSF Routing) which combined the North First method (NF) and the South First method (SF). NF and SF are algorithms yielded by the turn model. NSF Routing is applicable to 2-D Torus. The dynamic performance was evaluated by a software simulation, and the static performance was also evaluated. As a result, it was shown that a throughput improves in some communication patterns, and about the same performance of a "The Average Number of Shortest Paths (ANSP)" static property.*

Keywords: Network on Chip, Interconnection Network, Adaptive Routing, Turn model

1 Introduction

The interconnection network is an important topic in the field of parallel processing. Parallel computers have processing elements (PEs) that are directly connected to a network such as a k-ary n-cube. Parallel processing is also performed in a *Network on Chip (NoC)* between PEs located on one chip. Many different interconnection networks for parallel processing have been proposed, and the 2-D torus network is one of the most popular.

The routing algorithms of interconnection networks are classified into deterministic routing, in which paths are fixed, and adaptive routing [1]-[7], in which paths are changed to avoid failures or congestion. Because of its tolerance to failures and congestion, adaptive routing has been the topic of a lot of research. Various adaptive routing algorithms have been proposed for k-ary n-cubes [3]-[7]. However, these methods require additional hardware for virtual channels comparison with deterministic routing (its name is *Dimension Order Routing,* DOR) on a 2-D torus.

A number of adaptive routing algorithms based on the turn model [8]-[10] do not need additional virtual channels. However, most of these algorithms cannot be applied to torus networks without change because most of those methods are algorithms for mesh or hyper-cube network. If an adaptive routing algorithm for a torus network could be realized by modifying the turn model, it would be possible to realize adaptive routing without having to install additional virtual channels.

In this paper, we propose the North-South First Routing (NSF Routing) which combined the North First method (NF) and the South First method (SF) and evaluate the performance by software simulation and a static property. NF and SF are the part of the algorithms by a Turn model. NSF Routing is applicable to 2-D Torus. Moreover, performance is evaluated by a software simulation.

2 2-D Torus Network

The 2-D torus network has an $N \times N$ 2-dimensional structure, and its four edges are connected by wraparound links. It is used in many parallel computers and some interconnection networks include this.

Dimension order routing (DOR) is generally used for deterministic routing on a 2-D torus. In DOR, the packet moves on channels in the y-direction before moving to the x-direction. To avoid deadlocks on a 2-D torus, DOR needs two virtual channels (channel-L and channel-H).

The method of selecting a virtual channel in the case of DOR on a 2-D torus network is as follows:

- Choose channel-L when starting routing in the y-direction.
- When the head of the packet passes through a wraparound link, move the packet to channel-H.
- When the routing in the y-direction is completed, move the packet in the x-direction; use channel-L regardless of the current channel.
- When the head of a packet passes through a wraparound link, moves the packet to channel-H. Use channel-H until the routing finishes.

Figures 1 and 2 show the link selection function and channel selection function of DOR on an $N \times N$ torus. Here, the address of each PE of the torus is shown in terms of their coordinates (x, y). Moreover, the y-direction channels are written as Y+ and Y−, and the x-direction channels are written as X+ and X−. The four inputs of the link selection function indicate the *x* and *y* coordinates of the present PE, and the *x* and *y* coordinates of the destination PE. The function outputs the link of either X+, X−, Y+, Y− or "OUT", which is an output link to a node.

The three inputs of the channel selection function correspond to the current direction, current channel, and direction of the next hop. The current direction and the direction of the next hop have four states, i.e., X+, X−, Y+, and Y−. The current channel has three states, i.e., channel-L (L), channel-H (H), and wraparound channel (W). Although the output has two states (L and H), it unconditionally serves as W when the selected link is a wraparound link.

```
// Link Selection Function for Dimension-Order Routing
Link_Select_DOR (cx, cy, dx, dy)
    cx, cy;          // current node      0 ≦ cx, cy ≦ N−1
    dx, dy;          // destination       0 ≦ dx, dy ≦ N−1
{
    if(cy≠dy){                            // dimension Y
      dist_y = (N+dy-cy)%N;
      if(1≦dist_y ≦N/2)                 return Y+;
      else                              return Y-;
    }
    else if(cx≠dx){                       // dimension X
      dist_x = (N+dx-cx)%N;
      if(1≦dist_x ≦N/2)                 return X+;
      else                              return X-;
    }
    else                                return OUT;
}
```

Fig.1 The Link Selection Function of the Dimension-Order Routing

```
// Channel Selection Function for DOR
Channel_Select_DOR (cd, cc, nd)
    cd;         // current direction    ∈{Y+, Y-, X+, X-}
    cc;         // current channel      ∈{L, H, W}
    nd;         // next direction       ∈{Y+, Y-, X+, X-}
{
    if(cc∈L)                  return L;   // before wrap around
    else                                  // after wrap around
      if(cd∈{X+,X-} & nd ∈{Y+,Y-})
                              return L;   // Y→X
      else                    return H;
}
```

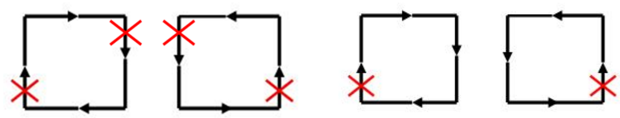
Fig.2 The Channel Selection Function of the Dimension-

3 Adaptive Routing of *k*-ary *n*-cube

3.1 Turn Model

The turn model [8] is used by some adaptive routing algorithms [9][10]. Packet cycles can be prevented by adding a restriction to a path change (turn) of a packet. In the case of a 2-D mesh, there are eight kinds of turn, and the various turn model methods put restrictions on two of the eight turns. There is essentially no difference between these methods other than the choice of turn to be restricted. In this paper, we shall incorporate the North First (NF) algorithm and South First (SF) algorithm into one (NSF) and apply it to a 2-D torus.

The turn model of DOR for a 2-D mesh is shown in Fig.3 a), and the turn model of the NF algorithm is shown in Fig.3 b). DOR restricts four out of eight turns, whereas the NF algorithm restricts only two, i.e., X− (left, west)→ Y+ (upper, north) and the X+ (right, east) → Y+ (upper, north). The South First algorithm, by which the Y− (South) direction is chosen at the beginning of a routing path, is similar.



a) Dimension -Order Routing b) North First (NF) Algorithm

Fig.3 The Turn Model for 2D-Mesh Network

3.2 Application of the Turn Model to a Torus Network

When applying a turn model such as the NF algorithm to a torus network without change, the following differences from the case of a mesh network have to be considered.

1) In a torus network, when the packet passes through a wraparound channel, a deadlock by cyclic dependency can occur. Therefore, it is necessary to impose an additional restriction.
2) At least two virtual channels are needed for routing in a torus network. As a result, adaptive routing with higher pliability is attained by applying different turn models to each channel.

An example of a cyclic dependency that occurs in the NF algorithm is shown in Fig.4. Here, packets A-D mutually block a path, causing a deadlock. By contrast, the deadlock does not happen in DOR because packets A and C do not turn in Fig.4. This problem illustrates that it is necessary to take into consideration complicated turn restrictions in adaptive routing on a torus network. Our method deals with this issue by applying the NF and SF algorithms to channel-H and channel-L.

4 North-South First Routing

If the turn model such as NF or SF algorithms is used for 2-D torus, the circulation as shown in Fig. 4 occurs by packets of wraparound channels. To avoid the sort of deadlock described above, additional restrictions have to be put on the NF and SF algorithms:

1) The SF algorithm does its routing on channel-H. However, a cycle may occur when a path is chosen in which a packet returns to channel-L through channel-H, and for this reason,

DOR is carried out instead of the adaptive routing. In DOR, the x-direction channel chosen after a vertical (y-direction) wraparound channel has to be channel-L.

2) The NF algorithm does its routing on channel-L. Because the path of channel-H→channel-L exists after a wraparound channel, the cycle shown in Fig.4 occurs. As shown in Fig.5, though, the cycle can be avoided by adding one more restriction to the other two. Here, three restrictions are put on eight turns, specifically, right→upper, left→upper, and right→lower. This algorithm was named *restricted North First (rNF)*.

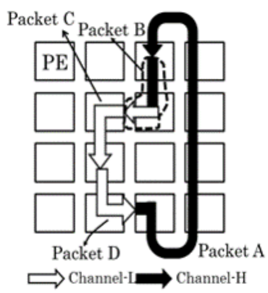


Fig.4 The Cyclic Dependency by the Application of the NF Algorithm in Torus Network

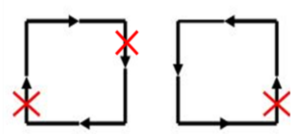


Fig.5 The Restricted North First Routing

4.1 Definitions

From here on, all channels will be described in terms of their dimension $d \in \{\mathrm{X}, \mathrm{Y}\}$, direction $\delta \in \{+, -\}$, channel type $c \in \{\mathrm{L}, \mathrm{W}, \mathrm{H}\}$, i.e., $(d\delta, c)$. X means X dimension, Y means Y dimension, and L, W, and H means channel-L, wraparound channel, and channel-H. $(d+, c)$ and $(d-, c)$ will be shown as a set, written as $(d\pm, c)$.

4.2 Routing Algorithm

In our method, the restricted NF algorithm is carried out in channel-L and the SF algorithm is carried out in channel-H. Since $(\mathrm{Y}-, \mathrm{L})$ and $(\mathrm{Y}+, \mathrm{H})$ are respectively used in the restricted NF algorithm and SF algorithm, we will study cases in which $(\mathrm{Y}+, c)$ is used and not used, and cases in which the horizontal and vertical wraparound channels are used and not used.

Figures 6 and 7 show the link selection function and channel selection function of the proposed method on a $N \times N$ torus. As in the case of DOR in Fig.1, the link selection function outputs X+, X−, Y+, Y−, or "OUT" (an output link to a node). The proposed method needs the "current channel" as an input in addition to the inputs of DOR.

The channel selection policy varies depending on whether $(\mathrm{Y}+, c)$ is used or not. If it is used, adaptive routing is carried out only when the wraparound channels is not to be used from that point on. If $(\mathrm{Y}+, c)$ is not used, the restricted NF algorithm is carried out from the source PE until the first wraparound channel (or destination PE) is reached.

The algorithm of Fig.6, in ①, first determines whether the wraparound links of X and Y are used. In this case, the determination is based on the X and Y coordinates of the source and destination PEs as follows:

- When the difference between the X coordinates of the current PE and destination PE is less than *N*/2, h_wrap is set to 0 because the wraparound channel of the x-direction is not straddled. If not, h_wrap is set to 1.
- When the difference between the Y coordinates of the current PE and destination PE is less than *N*/2, v_wrap is set to 0. If not, v_wrap is set as 1.

Next, the link is chosen on the basis of whether the Y+ channel (channel $(\mathrm{Y}+, c)$) is used or not, as follows:

- When $(\mathrm{Y}+, c)$ is used, the procedure ② is carried out. In this case, since the restricted NF in channel-L is equivalent to DOR, only the adaptive routing of the SF method in channel-H is carried out. If neither wraparound channel is used in going from the current PE to the destination, the packet can be sent over channel-H and routing can be continued. Thus, adaptive routing can be carried out with the SF method. The only other case in which channel-H may be used is after the packet has passed through a vertical wraparound channel (Y+, W) and is due to pass through a horizontal wraparound channel (X±, W). Even in this case, it is thought that adaptation routing using the SF method is possible. However, since it is difficult to prove that is deadlock-free, only the X-directional routing is carried out at first and SF is applied after the packet has passed through channel (X±, W). In the other case, DOR is carried out because only the channel-L is used.
- When $(\mathrm{Y}+, c)$ is not used, the procedure ③ is carried out. Since the SF method in channel-H is equivalent to DOR, only the adaptive routing of the restricted NF method in channel-L is carried out. In this case, the following restriction is added in order to make the order of passage in a wraparound channel into (Y−, W) → (X±, W).
  - ➢ Restricted NF is carried out only when (Y−, W) is not be passed from the current PE to the destination or the next channel is not (X±, W). DOR is carried out otherwise.

Besides the three inputs of the channel selection function of DOR in Fig.2, the channel selection function needs four inputs that indicate the x and y coordinates of the source and destination PEs. These new inputs can be used to judge the possibility of the packet passing through a wraparound channel. Based on the judgment, channel-H is chosen only when the wraparound channel is not to be used and $(\mathrm{Y}+, c)$ is to be used. DOR is carried out otherwise. As in the case of DOR, the output has two states, L and H. However, an output unconditionally serves as W when the selected link is a wraparound link.

```
// Link Selection Function for Proposed Algorithm
Link_Select_Prop (cx, cy, cc, dx, dy)
    cx, cy;          // current node       0 ≦ cx, cy ≦ N－1
    cc;              // current channel    ∈{L, H, W}
    dx, dy;          // destination        0 ≦ dx, dy ≦ N－1
{
    if(|dx-cx|≧N/2)           h_wrap = 1;
    else                      h_wrap = 0;
    if(|dy-cy|≧N/2)           v_wrap = 1;          ①
    else                      v_wrap = 0;

    dist_y = (N+dy-cy)%N;
    if(1≦dist_y ≦N/2)                   // Y+ direction
        if(h_wrap=0 & v_wrap=0)
                              return adaptive_SF(cx, dx);
        else if(h_wrap=1 & v_wrap=0)                    ②
                              return DOR(cx, cy, dx, cy);
        else                  return DOR(cx, cy, dx, dy);
    else if(cy≠dy)      // Y- direction
        if((cc∈L) and ((cx≠0) or (v_wrap=0))
                              return adaptive_NF(cx, dx);   ③
        else                  return DOR(cx, cy, dx, dy);
        else if(cx≠dx)        return x_route(cx, dx);
    else return OUT;
}

adaptive_SF(cx, dx){       //adaptive routing of SF algorithm
    if(cx=dx)                 return Y+;
    else if(buffer_is_full(Y+, H)=TRUE)
                              return x_route(cx, dx);
    else                      return Y+;
}

adaptive_NF(cx, dx){          //adaptive routing of NF algorithm
    dist_x = (N+dx-cx)%N;
    if(cx=dx)                 return Y-;
    else if(N/2＜dist_x)                  // X- direction
                              return X-;
    else if(buffer_is_full(Y-, L)=TRUE)   // X+ direction
                              return X+;
    else                      return Y-;
}

x_route(cx, dx){
    dist_x = (N+dx-cx)%N;
    if(1≦dist_x ≦N/2)         return X+;
    else                      return X-;
}

DOR (cx, cy, dx, dy){
    return Link_Select_DOR (cx, cy, dx, dy);
}
```

Fig.6 The Link Selection Function of the Proposed Algorithm

4.3 Deadlock Avoidance

A channel dependency graph is drawn in order to prove that the routing algorithm described in the previous section does not cause a deadlock[11][12]. The channel dependency graph is a directed graph in which nodes (channels) with dependencies are connected by an arrow. Specifically, nodes (channels) with dependencies are pairs of nodes (channels) in which a packet may be directly transmitted and received while routing.

```
// Channel Selection Function for Proposed Algorithm
Channel_Select (cx, cy, dx, dy , cd, cc, nd)
    cx, cy;          // current node        0 ≦ cx, cy ≦ N－1
    dx, dy;          // destination         0 ≦ dx, dy ≦ N－1
    cd;              // current direction   ∈{Y+, Y-, X+, X-}
    cc;              // current channel     ∈{L, H, W}
    nd;              // next direction      ∈{Y+, Y-, X+, X-}
{
    if(dx-cx≧N/2)                  h_wrap = 1;
    else                           h_wrap = 0;
    if(dy-cy≧N/2)                  v_wrap = 1;
    else                           v_wrap = 0;

    dist_y = (N+dy-cy)%N;
    if((1≦dist_y ≦N/2)                      // Y+ direction
   and (h_wrap=0 & v_wrap=0))      return H;
    else                                      // Others
                         return DOR_Channel (cd, cc, nd);
}

DOR_Channel (cd, cc, nd){
                    return Channel_Select_DOR (cd, cc, nd);
}
```

Fig.7 The Channel Selection Function of the Proposed Algorithm

First, the channel dependency graph is drawn. Then, each channel is numbered. If it is proved that the channel numbers are in ascending order (or descending order) in the direction of the arrows of the channel dependency graph, deadlock does not happen. In such case, the channels are said to have an ordered relation and the corresponding channel will not cause a cyclic dependency.

A routing algorithm based on the turn model generally assigns numbers to the output channels from the PE on the basis of the PE address. As mentioned above, a 2-D torus network has two virtual channels. Accordingly, the following 4-dimensional channel numbers *CN* are given to the 4 links ×2 channels (=8 channels) in each PE of an $N \times N$ torus.

$$CN(x, y, d, ch) = (g_m, c_1, g_s, c_2) \quad (1)$$

Here, x $(0 \le x \le N-1)$ and y $(0 \le y \le N-1)$ is x and y coordinates of PE address, $d \in \{Y+, Y-, X+, X-\}$ is the direction of the channel, and $ch \in$ {L,H,W} is the type of channel. Also, g_m, c_1, g_s, and c_2 are named as *Main Group*, *First Coordinate*, *Sub Group*, and *Second Coordinate*, respectively.
The channel number in each channel is as Fig.8. When a channel number is set as Fig.8, deadlock occurrence can be avoided because channel numbers will become an ascending order through a routing path[13].

5 Dynamic Performance Evaluation

5.1 Environment

We used a wormhole routing simulator to evaluate the dynamic communication performance of our algorithm on a

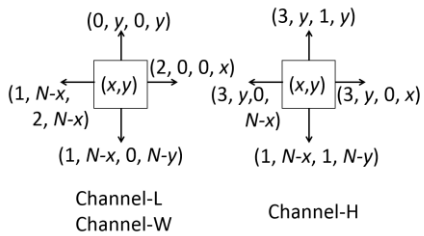


Fig.8 The Channel Number

16×16 2-D torus/mesh network with 256 PEs. Dynamic communication performances are simulated for dimension-order routing algorithm and proposed algorithm (North South First Routing). Extensive simulations have been carried out for uniform, matrix-transpose, and bit-reversal.

The dynamic communication performance of an interconnection network was characterized by the average transfer time and throughput. The average transfer time was the average value of the latency for all packets. Latency was the time between the injection time of the first flit and the reception time of the last flit at the destination. Throughput was the average value of the number of flits which a PE receives in each clock cycle. In the evaluation of dynamic communication performance, flocks of messages were sent in the network so that they competed for the output channels. Packets were transmitted with a request probability *r* during *T* clock cycles and the number of flits which reached the destination PE and their transfer times were recorded. The average transfer time and throughput were then calculated and plotted. The request probability *r* was varied. The packet size was 16 flits, and flits were transmitted for 50,000 cycles, i.e., *T*=50000. Two virtual channels per physical link were simulated. The buffer length of each channel was 8 flits.

5.2 Uniform Traffic

In uniform traffic, destinations are randomly chosen with equal probability among the nodes in the network. The result of the uniform traffic pattern is shown in Fig.9. As shown in Fig.9, the throughput of the proposed method is slightly higher than DOR. In the communication pattern such as uniform traffic, the whole network is equally crowded. So the effect of avoidance from crowded links is limited. However in our method, since some packets are directly sent into the channel-H, the load of the channel is distributed. So the throughput is improved.

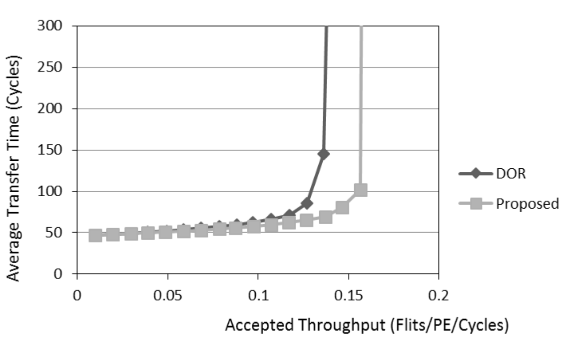


Fig.9 The Result of Uniform Traffic Pattern

5.3 Matrix-Transpose

The matrix-transpose is a traffic pattern based on the transposition of matrix. In this pattern, packets are transmitted between PEs over a diagonal line. In this research, it was assumed that the number of PEs and data are same. About each element of the matrix $A = \{a_{ij}\}$, a_{ij} was assigned to PE (i, j) and the communication of transposition was carried out. Therefore, the traffic pattern of the matrix-transpose is the communication between PE(i, j) and PE(j, i). The result of the matrix-transpose traffic pattern is shown in Fig.10 As shown in Fig.10, the throughput of DOR is a limit by the throughput of 0.1. On the other hand, it turns out that the throughput is extended to 0.14 by the proposed method.

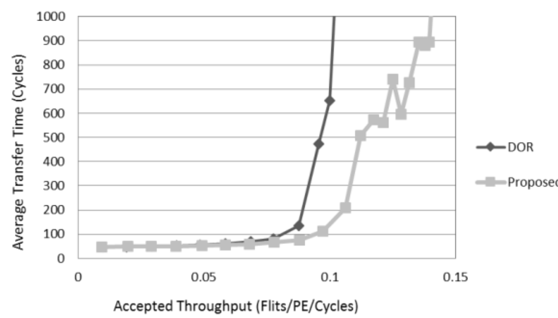


Fig.10 The Result of matrix-transpose

5.4 Bit-reversal

Bit-reversal is traffic pattern to other PE that the bit of the address by binary expression becomes reverse. Since the number of PEs is 256 in this experiment, this pattern is the communication from PE (x, y)=PE $(x_3x_2x_1x_0, y_3y_2y_1y_0)$ to PE$(y_0y_1y_2y_3, x_0x_1x_2x_3)$. The result of the bit-reversal traffic pattern is shown in Fig.13. As the result of matrix transpose, it turns out that the limit of network load improves by the proposed method.

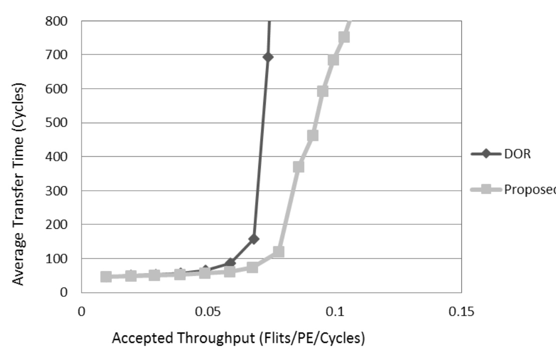


Fig.11 The Result of Bit-reversal

6 Static Performance

6.1 Derivation

In this section, *The Average Number of Shortest Paths (ANSP)* is derived in order to evaluate what pliability the proposed method has. Here, ANSP is defined as "the average number of shortest paths in all the combinations from source to destination". The ANSP of interconnection network with $M = N \times N$ nodes is defined as follows:

$$\text{ANSP} = \sum_{i=0}^{M-1} \sum_{j=0}^{M-1} Np(i,j)/M \times M \quad (2)$$

Here, $Np(i,j)$ is the number of the shortest paths from node i (the coordinate of node i is $(x_i,\ y_i)$) to node j (the coordinate of node j is $(x_j,\ y_j)$).

Based on (2), ANSP is derived as follows:

- Deterministic Routing

In the Deterministic Routing (DOR), $Np(i,j) = 1$ constantly. Thus, $\sum_{i=0}^{M-1} \sum_{j=0}^{M-1} Np(i,j)$ becomes $M \times M$. So ANSP becomes 1.

- South First Algorithm

In the South First Algorithm of the mesh network with $M = N \times N$ nodes, $Np(i,j)$ is derived as follows:

- When y-coordinate of the node j is lower than or equal to the node i ($y_j \leq y_i$), $Np(i,j)$ is 1 because the packet moves as the DOR.
- When y-coordinate of the node j is higher than the node i ($y_j > y_i$), the adaptive routing is carried out. When the minimum number of hops from i to j is n and the distance of y-coordinate is v and the distance of x-coordinate is $h = n - v$, Np becomes $Np(i,j) = {}_nC_v$.

Since $Np(i,j)$ depends on n and v, the table of $Np(i,j)$ from the combination of n and v (from here, it is described as (n, v)) can be obtained. Also, since one (n, v) is obtained from 1 set of combinations of i and j, the number of counts of (i,j) can be obtained from (n, v). From the above tables, the value of ANSP can be derived. In the case of 4×4 mesh, the number of counts of (i,j) (num.of (i,j)) based on n and v is obtained as table 1 (when $v = 0$ or $h = 0$, $Np(i,j) = 1$ obviously. So these cases were excepted from the table). For example, since n and v become $n = 6$ and $v = 3$, "the number of counts of (i,j)" becomes 2 when i, j are upper-left and lower-right, or upper-right and lower-left respectively.

Thus, from the 256 combinations of source and destination in 4×4 mesh with 16 nodes, $Np(i,j) = 20$ is 2 patterns, $Np(i,j) = 10$ is 8 patterns, $Np(i,j) = 6$ is 8, $Np(i,j) = 4$ is 12, $Np(i,j) = 3$ is 24, $Np(i,j) = 2$ is 18, and $Np(i,j) = 1$ is 184 patterns. So $\sum_{i=0}^{M-1} \sum_{j=0}^{M-1} Np(i,j)$ becomes 508, then ANSP≒1.98 by (2).

Table 1 The Number of Counts of (i,j) And the Value of $Np(i,j)$ from n n, v, h in the South First Routing of 4×4 Mesh Network (The Case of ($y_j > y_i$))

n	v	h	num. of (i,j)	$Np(i,j)$
6	3	3	2	20
5	3	2	4	10
	2	3	4	10
4	3	1	6	4
	2	2	8	6
	1	3	6	4
3	2	1	12	3
	1	2	12	3
2	1	1	18	2

- North-South First Algorithm

For the analysis about North-South First Algorithm of 4×4 torus, the table 1 is modified as follows:

- When ($y_j > y_i$), the SF method is carried out when $v \neq 3$, and the RNF method via wraparound link is carried out when $v = 3$. Furthermore, some algorithm of routing via horizontal wraparound link may be carried out when $h = 2{,}3$. Thus the rows of $v = 3$ and $h = 2, 3$ are modified from table 1. Since the row of $v = 3$ and $h = 3$ are changed to $v = 1$ and $h = 1$, the value of n and $Np(i,j)$ are changed. In the case of $h = 2$, since the horizontal wraparound link may be used, $Np(i,j)$ is modified in this case.
- When ($y_j < y_i$), the RNF is carried out when $v = 1$, and the SF may be carried out when $v \neq 1$. as same as above, some algorithm of routing via horizontal wraparound link may be carried out when $h = 2{,}3$. So, all rows except $h = v = 1$ are modified based on table 1. Also, the half of (i,j) patterns in RNF become $Np(i,j) = 1$ in all cases, and those values are modified.

The modified tables by above procedure are shown in table 2 and table 3. Those tables show the value of n, v, h, the number of counts of (i,j), and the value of $Np(i,j)$ in 4×4 torus with 16 nodes. n',v', and h' in tables are the value of n, v, and h when the 4×4 torus is assumed as 4×4 mesh (when the routing process is carried out without wraparound channels). The positions in tables of them are same as table 1. (n), (v), and (h) in those tables are the true values of n, v, and h as routing paths of 4×4 torus.

Thus, from the 256 combinations of source and destination in 4×4 torus with 16 nodes, $Np(i,j) = 6$ is 8, $p(i,j) = 3$ is 38, $Np(i,j) = 2$ is 40, $Np(i,j) = 1$ is 170 patterns. So $\sum_{i=0}^{M-1} \sum_{j=0}^{M-1} Np(i,j)$ becomes 412, then ANSP≒1.61 by (2).

6.2 Evaluation Result

The same evaluation as the previous section was carried out to some network topologies. The evaluation results are shown in table 4. As shown in the table, The ANSP of proposed method is a few lower than SF algorithm of mesh of same size. This may be caused by that the number of hop of mesh network

Table 2 The Number of Counts of (i,j) And the Value of $Np(i,j)$ from n n, v, h in the North-South First Routing of 4×4 Torus Network
(The Case of ($y_j > y_i$))

n' (n)	v' (v)	h' (h)	num. of (i,j)	$Np(i,j)$
6(2)	3(1)	3(1)	2	1
5(3)	3(1)	2	4	3,2,3,1
	2	3(1)	4	3
4 (4 or 2)	3(1)	1	6	2 or 1
	2	2	8	6,6,6,6, 6,6,1,1
	1	3(1)	6	2
3	2	1	12	3
	1	2	12	3,3,3,3,3,3, 3,1,3,1,3,1
2	1	1	18	2

Table 3 The Number of Counts of (i,j) And the Value of $Np(i,j)$ from n n, v, h in the North-South First Routing of 4×4 Torus Network
(The Case of ($y_j < y_i$))

n' (n)	v' (v)	h' (h)	num. of (i,j)	$Np(i,j)$
6(2)	3(1)	3(1)	2	1
5(3)	3(1)	2	4	1
	2	3(1)	4	1
4 (4 or 2)	3(1)	1	6	1
	2	2	8	1,1,1,1, 3,1,3,1
	1	3(1)	6	1
3	2	1	12	1,1,1,1,1,1, 2,1,2,1,2,1
	1	2	12	3 or 1
2	1	1	18	2 or 1

Table 4 The ANSP of Some Types of Networks

Topology/ Routing Algorithm	Maximum Number of Hops	$\sum_{i=0}^{M-1}\sum_{j=0}^{M-1} Np(i,j$	ANSP
Mesh, Torus / Deterministic	6/4	256	1
4×4 Mesh / SF Algorithm	6	508	1.98
4×2 Mesh / SF Algorithm	4	84	1.31
3×3 Mesh / SF Algorithm	4	115	1.42
4×4 Torus/ NSF Routing	4	347	1.36

is higher than torus network. Then, the comparison with SF algorithm of the mesh network with the same hops as torus was carried out. As the result, it was shown that the ANSP of the proposed method is a little higher than SF algorithm of mesh. From those results, it is thought that the proposed method has about the same performance as SF algorithm of mesh.

7 Conclusions

In this paper, we proposed the North-South First Routing (NSF Routing) which combined the North First method (NF) and the South First method (SF). Also, the communication performance was evaluated by the software simulation. As a result, it was shown that a throughput improves in some communication patterns, and about the same performance of a ANSP static property. From now on, theoretical analysis of the other properties and evaluation about the fault tolerance are remaining as future work.

References

[1] M.M. Hafizur Rahman, Yukinori Sato, Yasuyuki Miura, and Yasushi Inoguchi, Dynamic Communication Performance of a Hierarchical 3D-Torus Network, IASTED, In 10th International Conference Parallel and Distributed Computing and Networks (PDCN 2011), 2011.2.

[2] Yasuyuki Miura, Masahiro Kaneko, M.M.Hafizur Rahman and Shigeyoshi Watanabe, Adaptive Routing Algorithms and Implementation for TESH Network, Communications and Network (CN), Vol.5, No.1, pp.34-49, 2013.02.

[3] W.J. Dally and Hiromichi Aoki, Deadlock-Free Adaptive Routing in Multicomputer Networks Using Virtual Channels, IEEE Trans. On Parallel and Distributed Systems, Vol,4, pp. 466-475, 1993

[4] M.P.Merlin and J.P.Schweitzer, Deadlock Avoidance in Store-and-Forward Networks-1: Store and Forward Deadlock, IEEE Trans. On Comm., Vol.COM-28, No.3, pp.345-354, 1980

[5] W.J.Dally and C.L.Seitz. Deadlock-Free Message Rouring in Multiprocessor interconnection Networks. IEEE Trans. On Computers, Vol. C-36, No.5, pp.547-553, 1987.

[6] C. S. Yang and Y. M. Tsai, ”Adaptive Routing in k-ary ncube Multicomputers,” Proc. of ICPADS ’96, pp. 404-411, 1996.

[7] C. J. Glass and L. M. Ni, ”Maximally Fully Adaptive Routing in 2D Meshes,” ISCA92, pp. 278-287, 1992.

[8] C.J.Glass, L.M.Ni, The Turn Model for Adaptive Routing, The 25th Annual International Symposium on Computer Architecture, pp.441-450, 1998.

[9] Jie Wu, A fault-tolerant and deadlock-free routing protocol in 2D meshes based on odd-even turn model, IEEE Trans. on Computers, Vol.52, No.9, pp.1154-1169, 2003.

[10] A.Jouraku, M.Koibuchi, H.Amano, An Effective Design of Deadlock-Free Routing Algorithms Based on 2D Turn Model for Irregular Networks, IEEE Trans. on Parallel and Distributed Systems, Vol.18, No.3, pp.320-333, 2007.

[11] J.Duato, A New Theory of Deadlock-Free Adaptive Routing in Wormhole Networks, IEEE Trans. on Parallel and Distributed Systems, Vol.4, No.12, pp.1320-1331, 1993.

[12] E. Fleury and P.Fraigniaud, A General Theory for Deadlock Avoidance in Wormhole-Routing Networks, IEEE Trans. Parallel and Distributed Systems, Vol.9, No.7, pp.626-638, 1998.

[13] Yasuyuki Miura, Kentaro Shimozono, Kazuya Matoyama, Shigeyoshi Watanabe, An Adaptive Routing of the 2-D Torus Network Based on Turn Model, Proc. of 4th International Workshop on Advances in Networking and Computing, pp.587-591, 2013.12.