# Dynamic Communication Performance Enhancement in Hierarchical Torus Network by Selection Algorithm

M.M. Hafizur Rahman*, Yukinori Sato†, and Yasushi Inoguchi†

*Dept. of Computer Science, KICT, IIUM, Jalan Gombak-53100, Malaysia
E-mail: rahmanjaist@gmail.com, hafizur@iium.edu.my
†Center for Information Science, JAIST, Nomi-Shi, Ishikawa 923-1292, Japan
E-mail: {yukinori & inoguchi}@jaist.ac.jp

***Abstract*— A Hierarchical Torus Network (HTN) is a 2D-torus network of multiple basic modules, in which the basic modules are 3D-torus networks that are hierarchically interconnected for higher-level networks. The static network performance of the HTN and its dynamic communication performance using the deterministic, dimension-order routing algorithm have already been evaluated and shown to be superior to the performance of other conventional and hierarchical interconnection networks. However, the assessment of the dynamic communication performance improvement of HTN by the efficient use of both the physical link and virtual channels has not yet been evaluated. This paper addresses three adaptive routing algorithms – link-selection, channel-selection, and a combination of link-selection and channel-selection – for the efficient use of physical links and virtual channels of an HTN to enhance dynamic communication performance. It also proves that the proposed adaptive routing algorithms are deadlock-free with 3 virtual channels. The dynamic communication performances of an HTN is evaluated by using dimension-order routing and proposed adaptive routing algorithms under various traffic patterns. It is found that the dynamic communication performance of an HTN using these adaptive routing algorithms are better than when the dimension-order routing is used, in terms of network throughput.**

***Index Terms*— Interconnection network, HTN, wormhole routing, adaptive routing algorithm, deadlock free routing, and dynamic communication performance.**

## I. Introduction

For massively parallel computer (MPC) systems with tens of thousands or millions of nodes, the large diameter of conventional topologies is infeasible. Hierarchical interconnection networks (HINs) [1] are a cost-effective way to interconnect a large number of nodes. A variety of hyper-cube based HINs have been proposed [2]–[6]. However, for large-scale MPC systems, the number of physical links becomes prohibitively large. To alleviate this problem, several $k$-ary $n$-cube based HINs: H3D-Mesh [7], H3D-torus [8], TESH [9], and Cube Connected Cycles (CCC) [10] have been proposed. However, the dynamic communication performance of these networks is still very low, especially in terms of network throughput.

It has already been shown that a torus has better dynamic communication performance than does a mesh network [11]. This is the key motivation for us to consider a HIN, in which both the basic module and the interconnection of higher levels have toroidal interconnections. A Hierarchical Torus Network (HTN) [12] has been proposed as a new interconnection network for large-scale three-dimensional (3D) multicomputers. The HTN consists of a basic module (BM) which is a 3D-torus ($m \times m \times m$). The basic modules (BMs) are hierarchically interconnected by 2D-torus ($n \times n$). To reduce the peak number of vertical links between silicon planes, we consider higher-level networks as 2D-toroidal connections instead of 3D-toroidal connections, despite the fact that a 3D-torus has better performance than a 2D-torus network. The HTN is attractive since its hierarchical architecture permits the systematic expansion of millions of nodes. We have shown that the HTN possesses several attractive features including constant node degree, small diameter, small average distance, better bisection width, small number of wires, a particularly small number of vertical links, and economic layout area [12].

A routing algorithm specifies how a message selects its network path to cross from source to destination. Efficient routing is critical to the performance of an interconnection network. Deterministic, dimension-order routing has been popular in multicomputers because it has minimal hardware requirements and allows the design of simple and fast routers [13]. Although there are numerous paths between any source and destination, dimension-order routing defines a single path from source to destination. If that selected path is congested, the traffic between that source and destination is delayed, despite the presence of uncongested alternative paths. Adaptive routing allows paths to be chosen dynamically based on router status [14], [18], [19]. Although adaptive routing increases routing freedom, potentially improving performance, it also increases the cost of deadlock prevention. This cost can reduce the network clock speed, overwhelming the benefit of adaptive routing. This is why, dimension-order routing is still used in contemporary MPC system. If the logic for implementing adaptive routing were as simple as the dimension-order routing, and the required virtual


doi:10.4304/jnw.7.3.468-479

channels (VCs) were exactly equal, then, that routing algorithm would be a good choice for the routing of messages in a MPC system, rather than dimension-order routing.

The most expensive part of an interconnection network is the wire that creates the physical links, buffers and switches. For a particular topology, the physical links cost is constant. Since we have considered wormhole routing [20], [21] for message switching, the main factor in buffer expense is the number of VCs [22]. Efficient use of the physical links and VCs significantly improves dynamic communication performance. Since the hardware cost increases as the number of VCs increases, its unconstrained use is not cost-effective in parallel computers.

Our previous studies evaluated the dynamic communication performance of the HTN with a dimension-order routing algorithm under the uniform traffic pattern with 2 VCs [23] and various non-uniform traffic patterns with 3 VCs, and it proved to be better than conventional and other hierarchical interconnection networks [24]. Only the routing performance enhancement by efficient use of physical link by link-election algorithm is carried out in [25]. However, the performance enhancement by efficient use of VCs has not been assessed. The main objective of the work reported in this paper is to effectively use physical links and VCs to improve the dynamic communication performance of the HTN.

The remainder of this paper is organized as follows. In Section II, we briefly describe the basic structure of the HTN. In Section III, we propose the selection routing algorithm along with dimension-order routing for the HTN. All the algorithms are proven to be deadlock-free in HTN with only three VCs. Dynamic communication performance evaluation and analysis are described in Section IV. Finally, in Section V, we summarize the results presented in this paper.

## II. Interconnection of the HTN

The *Hierarchical Torus Network (HTN)* [12] is a HIN consisting of BM that are hierarchically interconnected for higher level networks. The BM consists of a 3D-torus network. In this paper, unless specified otherwise, BM refers to a Level-1 network. Successive higher level networks are built by recursively interconnecting lower level subnetworks in a 2D-torus. Both the BMs and the higher level networks have a toroidal interconnection. Hence, we use the name "Hierarchical Torus Network".

### A. Architecture of HTN

The BM of the HTN is a 3D-torus network of size $(m \times m \times m)$, where $m$ is a positive integer. $m$ can be any value, however the preferable one is $m = 2^p$, where $p$ is a positive integer. The node degree of the HTN is 8 while it is 6 for the 3D-torus network. Thus, each node of the HTN has 2 more ports than a node of the 3D-torus. The BM has some free ports at the contours of the $xy$-plane. These free ports are used for higher level interconnection. A $(m \times m \times m)$ BM has $4 \times m^2$ free ports for higher level interconnection. All free ports, typically one or two, of the exterior Processing Elements (PEs) are used for inter-BM connections to form higher level networks. All ports of the interior PEs are used for intra-BM connections.

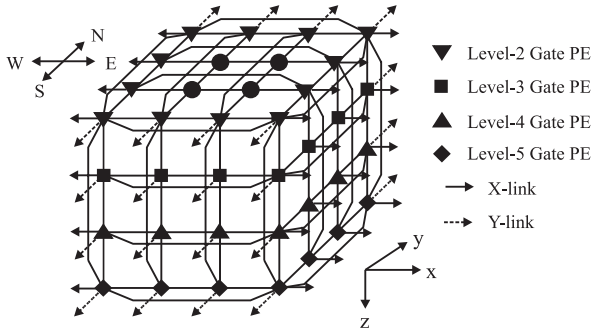
Successively higher level networks are built by recursively interconnecting lower level subnetworks in a 2D-torus network of size $(n \times n)$, where $n$ is also a positive integer. A Level-2 HTN can be formed by interconnecting $n^2$ BMs as a $(n \times n)$ 2D-torus. Each BM is connected to its logically adjacent BMs. Similarly, a Level-3 network can be formed by interconnecting $n^2$ Level-2 subnetworks, and so on. Thus, Level-$L$ is interconnected as a 2D-torus network, in which Level-$(L-1)$ is used as subnet modules. Since the Level-3 interconnection includes many BMs as subnet modules, a reasonable number of BMs is selected to make the 2D-torus of Level-2 modules. BMs with the same co-ordinate position in each Level-2 subnetwork are interconnected by a 2D-torus in a Level-3 interconnection. A similar interconnection rule is applied for higher levels.

It is useful to note that for each higher level interconnection, a BM must use $4m(2^q)$ of its free links: $2m(2^q)$ free links for $y$-direction interconnections and $2m(2^q)$ free links for $x$-direction interconnections. Here, $q \in \{0, 1, \ ... \ , p\}$, is the inter-level connectivity, where $p = \lfloor log_2^m \rfloor$. $q = 0$ leads to minimal inter-level connectivity, while $q = p$ leads to maximum inter-level connectivity. The highest level network which can be built from a $(m \times m \times m)$ BM is $L_{max} = 2^{p-q} + 1$. The total number of nodes in a network having $(m \times m \times m)$ BMs and $(n \times n)$ higher level is $N = \left[m^3 \times n^{2(L_{max}-1)}\right]$. Thus, the maximum number of nodes which can be interconnected by the HTN is $N = \left[m^3 \times n^{2(2^{p-q})}\right]$.

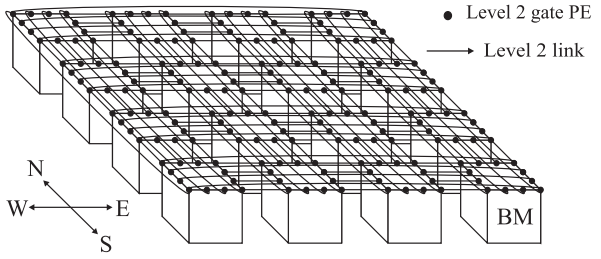
Figure 1 shows the interconnection of a HTN for $m = 4$ and $n = 4$. As depicted in Figure 1(a), for example, the $(4 \times 4 \times 4)$ BM has $4 \times 4^2 = 64$ free ports. With $q = 0$, $(4 \times 4 \times 2^0 =)$ 16 free links are used for each higher level interconnection, 8 for $y$-direction and 8 for $x$-direction interconnections. With $q = 0$, Level-5 is the highest possible level to which a $(4 \times 4 \times 4)$ BM can be interconnected. Figure 1(b) portrays a Level-2 HTN which is formed by interconnecting 16 BMs as a $(4 \times 4)$ 2D-torus with minimal inter-level connectivity. The total number of nodes in a network having $(m \times m \times m)$ BMs and $(n \times n)$ higher level is $N = \left[m^3 \times n^{2(L_{max}-1)}\right]$. Thus, the number of nodes interconnected by the HTN is $N = \left[m^3 \times n^{2(2^{p-q})}\right]$. If $m = 4$, $n = 4$, and $q = 0$, then $N = 4^3 \times 4^8 = 4194304$, i.e, about 4.2 million.

A BM with $m = 4$ and the higher levels with $n = 4$ is perhaps the most interesting network size because it has better granularity than the larger sizes. With $m = 8$, the size of the BM becomes $(8 \times 8 \times 8)$ with 512 nodes. Correspondingly, with $n = 8$, the second level would have 64 BMs. In this case, the total number of nodes in a Level-2 network is $32,768$. Clearly, the granularity of the family of networks is rather coarse.

The question may arise whether we need MPC with millions of nodes. The answer is 'yes'. Solving the grand




(a) Basic Module ($4 \times 4 \times 4$)




(b) Level-2 HTN ($4 \times 4$)

Figure 1. Interconnection of a HTN

challenge problems in many areas such as development of new materials and sources of energy, development of new medicines and improved health care, strategies for disaster prevention and mitigation, weather forecasting, and for scientific research including the origins of matter and the universe, requires teraflops performance for more than a thousand hours at a time. This is why, in the near future, we will need computer systems capable of computing at the petaflops or exaflops level. To achieve this level of performance, we need MPC with tens of thousands or millions of nodes.

### B. Addressing of Nodes

PEs in the BM are addressed by 3 base-$m$ numbers, the first representing the $x$-axis, the second representing the $y$-axis, and the last representing the $z$-axis. PEs at Level-$L$ are addressed by two base-$n$ numbers, the first representing the $x$-axis and the second representing the $y$-axis. The address of a PE at Level-$L$ is represented as shown in Eq. 1.

$$A^L = \begin{cases} (a_z)(a_y)(a_x) & \text{if } L = 1, \text{ i.e., BM} \\ (a_y^L)(a_x^L) & \text{if } L \geq 2 \end{cases} \quad (1)$$

More generally, in a Level-$L$ HTN, the node address is represented by:

$$\begin{aligned} A &= A^L A^{L-1} A^{L-2} \ ... \ ... \ ... \ A^2 A^1 \\ &= a_\alpha \ a_{\alpha-1} \ a_{\alpha-2} \ a_{\alpha-3} \ ... \ ... \ ... \ a_3 \ a_2 \ a_1 \ a_0 \\ &= a_{2L} \ a_{2L-1} \ a_{2L-2} \ a_{2L-3} \ ... \ ... \ ... \ a_3 \ a_2 \ a_1 \ a_0 \\ &= (a_{2L} \ a_{2L-1}) \ (a_{2L-2} \ a_{2L-3}) \ ... (a_2 \ a_1 \ a_0) \quad (2) \end{aligned}$$

Here, the total number of digits is $\alpha = 2L+1$, where $L$ is the level number. The first group contains three digits and the rest of the groups contain two digits. Groups of digits run from group number 1 for Level-1, i.e., the BM, to group number $L$ for the $L$-th level. In particular, $i$-th group $(a_{2i} \ a_{2i-1})$ indicates the location of a Level-$(i-1)$ subnetwork within the $i$-th group to which the node belongs; $2 \leq i \leq L$. In a two-level network, for example, the address becomes $A = (a_4 \ a_3) \ (a_2 \ a_1 \ a_0)$. The last group of digits $(a_4 \ a_3)$ identifies the BM to which the node belongs, and the first group of digits $(a_2 \ a_1 \ a_0)$ identifies the node within that basic module. The detailed architecture of the HTN was presented in [12].

## III. Routing Algorithm for HTN

Routing of messages in the HTN is performed from top to bottom. That is, it is first done at the highest level network; then, after the packet reaches its highest level sub-destination, routing continues within the subnetwork to the next lower level sub-destination. This process is repeated until the packet arrives at its final destination. When a packet is generated at a source node, the node checks its destination. If the packet's destination is the current BM, the routing is performed within the BM only. If the packet is addressed to another BM, the source node sends the packet to the outlet node which connects the BM to the BM of Level-$L$ at which the routing is performed.

Suppose a packet is to be transported from source node 0000000 to destination node 1131230. In this case, we see that routing should first be done at Level-3, therefore, the source node sends the packet to the Level-3 outlet node 0000130, whereupon the packet is routed at Level-3. After the packet reaches the $(1,1)$ Level-2 network, routing within that network is continued until the packet reaches the BM $(3,1)$. Finally, the packet is routed to its destination node $(2,3,0)$ within that BM.

### A. Dimension-Order Routing (DOR) Algorithm

In our previous studies, we considered a simple deterministic, dimension-order routing (DOR) algorithm [23], [24]. DOR routes a packet successively in each dimension until the distance in that dimension is zero, then proceeds to the next dimension. For routing messages using DOR in HTN, first find the nonzero offset in the most significant position (from left to right) by subtracting the current address from the destination. Then make a step towards nullifying the offset by sending the packet along that dimension in descending order. When the offset along a dimension is zero, then the routing message is switched over to the next dimension. The higher level networks and BM are 2D-torus and 3D-torus networks, respectively. Routing at the higher level is performed first in the $y$-direction and then in the $x$-direction. In a BM, the routing order is initially in the $z$-direction, next in the $y$-direction, and finally in the $x$-direction.

Routing in the HTN is strictly defined by the source node address and the destination node address. Let a source node address be $s_\alpha, s_{\alpha-1}, s_{\alpha-2}, ..., s_1, s_0$, a destination node address be $d_\alpha, d_{\alpha-1}, d_{\alpha-2}, ..., d_1, d_0$, and

```
Routing HTN(s,d);
source node address:s_α, s_α−1, s_α−2, ..., s_1, s_0
destination node address: d_α, d_α−1, d_α−2, ..., d_1, d_0
tag: t_α, t_α−1, t_α−2, ..., t_1, t_0
  for i = α : 3
    if (i/2 = 0 and (t_i > 0 or t_i = −(n − 1))), routedir = North; endif;
    if (i/2 = 0 and (t_i < 0 or t_i =  (n − 1))), routedir = South; endif;
    if (i%2 = 1 and (t_i > 0 or t_i = −(n − 1))), routedir = East; endif;
    if (i%2 = 1 and (t_i < 0 or t_i =  (n − 1))), routedir = West; endif;
      while (t_i ≠ 0) do
        N_z = outlet_z(s, d, L, routedir)
        N_y = outlet_y(s, d, L, routedir)
        N_x = outlet_x(s, d, L, routedir)
        BM_Routing(N_z, N_y, N_x)
        if (routedir = North or East), move packet to next BM; endif;
        if (routedir = South or West), move packet to previous BM; endif;
        if (t_i > 0), t_i = t_i − 1; endif;
        if (t_i < 0), t_i = t_i + 1; endif;
      endwhile;
  endfor;
    BM_Routing(t_z, t_y, t_x)
end
BM_Routing (t_2, t_1, t_0);
BM_tag t_2, t_1, t_0 = receiving node address (r_2, r_1, r_0) − destination (d_2, d_1, d_0)
  for i = 2 : 0
    if (t_i > 0 and t_i ≤ m/2) or (t_i < 0 and t_i = −(m − 1)), movedir = positive; endif;
    if (t_i > 0 and t_i = (m − 1)) or (t_i < 0 and t_i ≥ −m/2), movedir = negetive; endif;
    if (movedir = positive and t_i > 0), distance = t_i; endif;
    if (movedir = positive and t_i < 0), distance = m + t_i; endif;
    if (movedir = negative and t_i < 0), distance = t_i; endif;
    if (movedir = negative and t_i > 0), distance = −m + t_i; endif;
  endfor
  while(t_2 ≠ 0 or distance_2 ≠ 0) do
    if (movedir = positive), move packet to +z node; distance_2 = distance_2 − 1; endif;
    if (movedir = negetive), move packet to −z node; distance_2 = distance_2 + 1; endif;
  endwhile;
  while(t_1 ≠ 0 or distance_1 ≠ 0) do
    if (movedir = positive), move packet to +y node; distance_1 = distance_1 − 1; endif;
    if (movedir = negetive), move packet to −y node; distance_1 = distance_1 + 1; endif;
  endwhile;
  while(t_0 ≠ 0 or distance_0 ≠ 0) do
    if (movedir = positive), move packet to +x node; distance_0 = distance_0 − 1; endif;
    if (movedir = negetive), move packet to −x node; distance_0 = distance_0 + 1; endif;
  endwhile;
end
```

Figure 2. Dimension-Order Routing Algorithm of the HTN

a routing tag be $t_\alpha, t_{\alpha-1}, t_{\alpha-2}, ..., t_1, t_0$, where $t_i = d_i - s_i$. The source node address of HTN is expressed as $s = (s_{2L}, s_{2L-1}), (s_{2L-2}, s_{2L-3}), ..., (s_2, s_1, s_0)$. Similarly, the destination node address is expressed as $d = (d_{2L}, d_{2L-1}), (d_{2L-2}, d_{2L-3}), ..., (d_2, d_1, d_0)$. The DOR of the HTN is represented by $R_1$. Figure 2 shows the routing algorithm $(R_1)$ for the HTN.

Various adaptive routing algorithm has been proposed from different aspect. A shortest path adaptive routing algorithm with time complexity of $O(n^2)$ for the crossed cube-connected ring interconnection network proposed in [15]. Adaptive load balanced routing algorithm can manage traffic during congestion by sensing the traffic through channel queues in the X-Torus network [16]. Dual Adaptive Routing reduces the number of lookups by optimal utilization of cross links of X-Torus network [16]. High-radix interconnection networks require global adaptive routing to achieve optimum performance. Four indirect global adaptive routing algorithms such as credit round trip, progressive adaptive routing, piggyback routing, and reservation routing were proposed in [17] for dragonfly network.

### B. Channel-Selection (CS) Algorithm

TESH network is a combination of mesh and torus network. The VCs used in the higher-level torus network are effectively used, however, the VCs used in the BM are remain idle [18]. HTN is full of toroidal interconnection

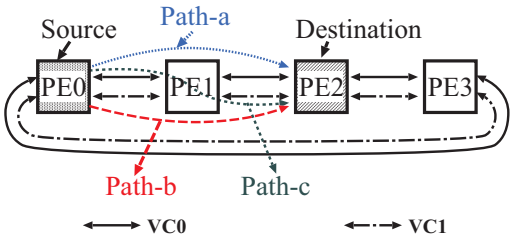



Figure 3. Selection of Virtual Channels by CS Algorithm

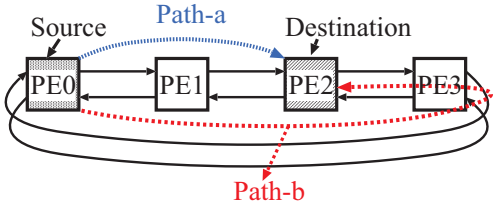



Figure 4. Selection of Physical Link by LS Algorithm

in both BM and higher level networks. Thus, the VCs are more efficiently used in the HTN.

A deadlock-free routing algorithm for a $k$-ary $n$-cube network using DOR can be obtained by using 2 VCs. The first channel is used for routing inter-node routing and the message is switched to second channel if the wrap-around link is going to be used. Only one channel is used at a time and other remains idle in DOR. Efficient use of these VCs improves dynamic communication performance. A Channel-Selection (CS) algorithm provides an efficient use of these VCs. HTN is a combination of 2D-torus and 3D-torus networks. Thus, the CS algorithm can be used in the BM and higher-level networks of the HTN.

If the first VC is congested and the second VC is not used in routing, then the message is switched to the second channel, or the second channel is selected initially. This is the main idea of the CS algorithm. Figure 3 portrays a 4-PE ring network with allocation of VC number. Two VCs are numbered as $VC_0$ and $VC_1$. When the wrap-around links are not used in routing, such as routing a message from $PE_0$ to $PE_2$, only $VC_0$ is used. In this case, because $VC_1$ is not used in dimension-order routing, it is possible to move from $VC_0$ to $VC_1$ or use $VC_1$ initially. That is, if path-$a$ is congested, then the message can initially follow either path-$b$ or switched to path-$c$, as shown in Figure 3. Similar scenario will happen for routing message from $PE_2$ to $PE_0$. VCs are efficiently used by this phenomenon.

The proposed CS algorithm breaks the restriction of using VCs in DOR and provides an efficient use of them. However, the routing order is strictly followed to route messages. The routing algorithm that applies this channel-selection principle is denoted as $R_2$.

### C. Link-Selection (LS) Algorithm

In each dimension of the wrap-around connected networks, such as $k$-ary $n$-cube, some links are used for inter-node connection and another link is used for wrap-around connection of end to end nodes; the message

will find an extra path to pass through. DOR does not make effective use of these links. However, effective use of these links significantly improves dynamic communication performance. A Link-selection (LS) algorithm provides an efficient use of these links.

Figure 4 shows an example of routing a message in a ring network using the LS algorithm. As shown in Figure 4, the number of hops from source to destination in the clockwise direction and in the counter-clockwise direction is 2. Then, the message can follow either path-a (clockwise) or path-b (counter-clockwise), as shown in Figure 4. Therefore, if the distance from source to destination is equal in both the clockwise and counter-clockwise directions, then the packet can follow either of these two directions. This is the principal idea of the LS algorithm. A 2D-torus consists of $x-y$ rings. Similarly, a 3D-torus consists of $x-y-z$ rings. Thus, LS algorithm is applicable in each direction of the HTN. If the following equation is satisfied, a packet can select from either a clockwise or counter-clockwise direction.

$$|s-d| = \begin{cases} \frac{m}{2} & \text{if } L=1, \text{ i.e., BM} \\ \frac{n}{2} & \text{if } L \geq 2 \end{cases} \quad (3)$$

Here $s$ and $d$ denotes the source and destination node addresses, respectively.

The deterministic routing algorithm also uses the wrap-around links. If the source-to-destination distance using wraparound links is less than that of not using wrap-around links, then message will follow the wrap-around links in DOR. In DOR, if the distance from source to destination is equal in both the clockwise and counter-clockwise directions, then the message will follow only the clockwise direction. In this study, however, the message can move in direction; first, it will attempt the clockwise direction and then, if the clockwise channel is busy, it will follow the counter-clockwise direction.

The proposed LS algorithm enforces some routing restrictions as DOR to make the routing algorithm simple and avoid deadlocks [13]. Routing order is strictly followed to route message. The routing algorithm that applies this link-selection principle is denoted as $R_3$.

The proposed LS algorithm can only improve the performance of the network when the number of nodes in each direction is even. According to the architecture of the HTN, a BM with $m=4$ and the higher levels with $n=4$ is perhaps the most interesting network size because it has better granularity than the larger sizes [12]. Therefore, the proposed LS algorithm is suitable for HTN.

### D. Combination of LS and CS (LS+CS) Algorithm

The channel-selection algorithm is used to select a VC in a physical link and the link-selection algorithm is used to select a physical link in a network. Therefore, both the CS and LS algorithms can be applied at the same time. The routing algorithm that applies both the channel-selection and link-selection principles is denoted as $R$.

### E. Deadlock-Free Routing

Buffers and switches are expensive elements in an interconnection network. Since we have considered wormhole-routed HTN, the main factor in buffer expense is the number of VCs. VCs [22] reduce the effect of blocking; they are used widely in MPC systems, to improve dynamic communication performance by relieving contention in the multicomputer network and to design deadlock-free routing algorithms [13], [26]. Since the hardware costs increase as the number of VCs increases, its unconstrained use is prohibited. Again, efficient use of these VCs also improves dynamic communication performance.

A useful parallel computer must be both efficient and reliable. A key component of reliability is freedom from deadlock. A deadlock-free routing algorithm for an interconnection network with a minimum number of VCs is preferred. However, there is a trade-off between dynamic communication performance and the number of VCs [27]–[29]. We have shown that using 3 VCs in the HTN with DOR yields better dynamic communication performance than other approaches [24]. We presented a deadlock-free dynamic routing algorithm using 3 VCs in [24]. In this paper, we recall the dynamic routing algorithm and the proof of a deadlock-free routing for HTN using DOR presented in [24] and present a proof for deadlock-free routing that applies the CS and LS principles using 3 VCs. We have applied the CS and LS algorithms in each level of the HTN like hierarchical routing algorithms (HiRA) [30]. To prove that the proposed routing selection algorithm for the HTN is deadlock free, we divide the routing path into three phases, as follows:

- *Phase 1:* Intra-BM transfer path from source PE to the face of the BM.
- *Phase 2:* Higher-level transfer path.
  **sub-phase** $2.i.1$ : Intra-BM transfer to the outlet PE of Level $(L-i)$ through the $y$-link.
  **sub-phase** $2.i.2$ : Inter-BM transfer of Level $(L-i)$ through the $y$-link.
  **sub-phase** $2.i.3$ : Intra-BM transfer to the outlet PE of Level $(L-i)$ through the $x$-link.
  **sub-phase** $2.i.4$ : Inter-BM transfer of Level $(L-i)$ through the $x$-link.
- *Phase 3:* Intra-BM transfer path from the outlet of the inter-BM transfer path to the destination PE.

Each direction of the HTN is a ring network. Therefore, the proposed CS and LS algorithms can be applied in each direction of the HTN. To make the routing algorithm simple and avoid deadlocks, we enforce some routing restrictions as DOR [13]. As the interconnection of the BM and the higher-level network of HTN is toroidal, a lemma and a corollary are stated below without proof, which was presented in [23]. Using the following lemma and corollary, we will prove that the proposed LS and CS algorithm for the HTN is deadlock-free using 3 VCs.

*Lemma 1:* If a message is routed in the order $z \rightarrow y \rightarrow x$ in a 3D-torus network, then the network is deadlock free with 2 VCs. [23]

*Corollary 1:* If the message is routed in the $y \rightarrow x$ direction in a 2D-torus network, then the network is deadlock free with 2 VCs. [23]

*Theorem 1:* The routing algorithm $(R_1)$ for the HTN with 3 VCs is deadlock-free.

*Proof:* Both the BM and the higher-levels of the HTN have a toroidal interconnection. In phase-1 and phase-3 routing, packets are routed in the source-BM and destination-BM, respectively. The BM of the HTN is a 3D-torus network. According to Lemma 1, the number of necessary VCs for phase-1 and phase-3 is 2. Intra-BM links between inter-BM links on the $xy$-plane of the BM are used in sub-phases $2.i.1$ and $2.i.3$. These sub-phases utilize channels over intra-BM links, sharing either the channels of phase-1 or phase-3. PEs at the contours of the $xy$-plane are assigned to each high level as gate nodes. The exterior links of the BM are used in sub-phase $2.i.2$ and sub-phase $2.i.4$, and these links form a 2D-torus network, which is the higher-level interconnection of the HTN. According to Corollary 1, the number of necessary VCs for this 2D-torus network is also 2. The mesh connection of the higher-level 2D-torus network shares the VC of either sub-phase $2.i.1$ or sub-phase $2.i.3$. The wrap-around connection of the higher-level 2D-torus networks requires 1 more VC.

Therefore, the total number of necessary VCs for the whole network is 3.

As mentioned earlier, the CS algorithm effectively uses the VCs of a wrap-around connected networks. Now, using theorem 1 and following lemma, we will prove that the proposed CS algorithm $(R_2)$ for the HTN is deadlock-free using 3 VCs.

*Lemma 2:* In a $k$-ary $n$-cube network, if 2 VCs are used according to condition 1 or condition 2, and the links are used according to condition 3, then the network is deadlock free .

**Condition 1:** Initially use VC 0.
**Condition 2:** When the packet is going to use wrap-around links, use VC 1.
**Condition 3:** If the wrap-around links are not used in routing and packet is in VC 0, then the packet can select VC 1.

*Proof:* The channels are allocated according to Theorem 1. It is proven that channel circulation will not occur during message flow. Thus, the network is deadlock-free.

*Theorem 2:* Suppose routing algorithm $R_1$ of the HTN is deadlock free. The routing algorithm $R_2$ which applies the CS algorithm, is also deadlock free.

*Proof:* Each direction of the HTN is a ring network. Routing algorithm $R_1$ is deadlock-free with 3 VCs. According to the Lemma 3, the routing algorithm $R_2$ for the HTN is also deadlock-free with 3 VCs.

Now, using theorem 1 and the following lemma, we will prove that the proposed LS algorithm $(R_3)$ for the HTN is deadlock-free using 3 VCs.

*Lemma 3:* In a $k$-ary $n$-cube network, if 2 VCs are used according to condition 1 or condition 2, and the links are used according to condition 3, then the network is deadlock free .

**Condition 1:** Initially use VC 0.
**Condition 2:** When the packet is going to use wrap-around links, use VC 1.
**Condition 3:** Packets can move either in the clockwise direction or the counter-clockwise direction if Eq. 3 is satisfied. Otherwise, move to a link nearer to the destination.

*Proof:* The physical links and the VCs are allocated in the BM according to lemma 1 and in the higher-level network according to corollary 1. Each direction of the network is a ring network, and LS algorithm is applicable in each direction. According to lemma 1 and corollary 1, no cyclic dependencies will occur in the BM and in the higher-level network, respectively. And according to theorem 1, the whole network is deadlock free.

*Theorem 3:* Suppose routing algorithm $R_1$ of the HTN is deadlock free. The routing algorithm $R_3$ which applies the LS algorithm is also deadlock free.

*Proof:* If Eq. 3 is not satisfied, then the routing of the message is carried out using routing algorithm $R_1$. According to theorem 1, routing $R_1$ for the HTN is deadlock free. If Eq. 3 is satisfied, i.e., if the LS algorithm is used to route the message, then according to lemma 3, the LS algorithm is also deadlock free. Therefore, the proposed LS algorithm $R_3$ is deadlock free.

*Theorem 4:* The routing algorithm $R$ for the HTN is deadlock free with 3 VCs.

*Proof:* According to theorem 2, the algorithm $R_2$ for the HTN which applies the CS algorithm, is deadlock free with 3 VCs. Similarly according to theorem 3, the routing algorithm $R_3$ for the HTN which applies the LS algorithm, is deadlock-free with 3 VCs. Therefore, the routing algorithm $R$ for the HTN which applies both the CS and LS algorithms,is deadlock free with 3 VC.

The main idea of these adaptive routing algorithms are presented in [31]. To show the superiority of these algorithms, we have evaluated the dynamic communication performance of the HTN using these algorithms under different traffic patterns such as uniform, hot-spot, and bit-permutation and combination traffic patterns.

## IV. Dynamic Communication Performance

The overall performance of a multicomputer system is affected by the performance of the interconnection network, as well as by the performance of the nodes. Continuing advances in VLSI/WSI technology promise to deliver more power to the individual nodes. On the other hand, low performance of the communication network will severely limit the speed of the entire multicomputer system. Therefore, the success of massively parallel computers is highly dependent on the efficiency of their underlying interconnection networks.

### A. Performance Metrics

The dynamic communication performance of a multicomputer is characterized by message latency and network throughput. Message latency refers to the time

elapsed from the instant when the first flit is injected into the network from the source to the instant when the last flit of the message is received at the destination. Network throughput refers to the maximum amount of information delivered per unit of time through the network. For the network to have good performance, low latency and high throughput must be achieved.

Latency is measured in time units. However, when comparing several design choices, the absolute value is not important; because the comparison is performed by computer simulation, latency is measured in simulator clock cycles. Throughput depends on message length and network size. Therefore, throughput is usually normalized, dividing it by the product of message length and network size. When throughput is compared by simulation and wormhole routing is used for switching, throughput can be measured in flits per node and cycle.

### *B. Simulation Environment*

We have developed a wormhole routing simulator to evaluate dynamic communication performance. In our simulation, we use DOR, LS, CS, and a combination of LS and CS algorithms. The DOR algorithm, which is exceedingly simple, provides the only route for the source-destination pair. The routing restriction of the LS, CS, and LS+CS algorithms is similar to the DOR, and it provides efficient use of the network's physical links and VCs. Extensive simulations for the HTN have been carried out for five different traffic patterns: uniform [32], hot spot [33], bit-reversal [34], bit-flip [35], and perfect shuffle [37]. Three VCs per physical channel are simulated, and the VCs are arbitrated by a round-robin algorithm. In the evaluation of dynamic communication performance, flocks of messages are sent in the network to compete for the output channels. For each simulation run, we have considered that the message generation rate is constant and the same for all nodes. The packet size is 16 flit and flits are transmitted at $20,000$ cycles. In each clock cycle, one flit is transferred from the input buffer to the output buffer, or from output to input if the corresponding buffer in the next node is empty. Therefore, transferring data between two nodes takes 2 clock cycles.

### *C. Traffic Patterns*

In an interconnection network, sources and destinations for messages form the traffic pattern. Traffic characteristics such as message length, message arrival times at the sources, and destination distribution have significant performance implications. Message destination distributions vary a great deal depending on the network topology and the application's mapping onto different nodes. In the uniform traffic pattern the source and the destination are randomly selected. However, depending on the characteristics of the application, some nodes may communicate with each other more frequently than with others. Consequently, non-uniform traffic patterns are frequent and cause uneven usage of traffic resources, significantly degrading the performance of the network. We have considered five different traffic patterns as follows:

**Uniform** – In the uniform traffic pattern, every node sends messages to every other node with equal probability in the network. That is, source and destination are randomly selected.

**Hot spot** – For generating hot spot traffic, we used Pfister and Norton's model [33]. In this model, each node first generates a random number. If that number is less than a predefined threshold, the message will be sent to the hot spot node. Otherwise, the message will be sent to other nodes with a uniform distribution.

**Bit-reversal** – The binary representation of the node address is $a_{\beta-1}, a_{\beta-2}$ ... ... $a_1, a_0$. In bit-reversal traffic, the $\text{node}\,(a_{\beta-1}, a_{\beta-2}$ ... ... $a_1, a_0)$ communicates with the $\text{node}\,(a_0, a_1,$ ... ... $a_{\beta-2}, a_{\beta-1})$.

**Bit-flip** – The node with binary coordinates $a_{\beta-1}, a_{\beta-2}$ ..... $a_1, a_0$ communicates with the $\text{node}(\overline{a_0}, \overline{a_1},$ ... ... $\overline{a_{\beta-2}}, \overline{a_{\beta-1}})$. That is, complement of bit-reversal traffic.

**Perfect shuffle** – The node with binary coordinates $a_{\beta-1}, a_{\beta-2}$ ... ... $a_1, a_0$ communicates with the $\text{node}(a_{\beta-2},$ ... ... $a_1, a_0, a_{\beta-1}$, i.e., rotate left 1 bit.

The uniform traffic pattern has been used to evaluate the network's dynamic communication performance in many previous studies and therefore, provides an important point of reference. When a hot spot occurs, a particular link experiences a much greater number of requests than the rest of the links – more than it can service. In a remarkably short period of time, the entire network may become congested. Hot spots are particularly troublesome because they may result from the cumulative effects of very small traffic imbalances. Hot spots often occur because of the burst nature of program communication and data requirements and, therefore can provide a benchmark for interconnection networks. Bit Permutation and Computation (BPC) is a class of non-uniform traffic patterns, which are very common in scientific applications. These include bit-reversal, bit-flip, and perfect shuffle, etc. BPC patterns take into account the permutations that are usually performed in parallel numerical algorithms [36]. These distributions achieve the maximum degree of temporal locality and are also considered as benchmarks. While these patterns are by no means exhaustive, they represent some interesting extremes of possible patterns.

### *D. Dynamic Communication Performance Evaluation*

A network can not accept more traffic than is supplied, and limitations in routing and collisions cause saturation before throughput reaches unity. However, efficient use of physical links and virtual channels using the proposed routing algorithms will improve the network throughput, causes saturation after the DOR. We have evaluated the dynamic communication performance of a Level-2 HTN the using DOR, LS, CS, and LS+CS algorithms under five different traffic patterns: uniform, hot spot, bit-reversal, bit-flipped, and perfect shuffle. To make a fair comparison, we allocated 3 VCs to the router for

performance evaluation. We will present the improvement of the dynamic communication performance of the HTN using our LS, CS, and LS+CS algorithms over DOR.

In a uniform traffic, destinations are chosen randomly with equal probability. Figure 5 depict the result of simulations under uniform traffic for the HTN using the DOR, LS, CS, and LS+CS algorithms. The figure shows the average transfer time as a function of network throughput. Each curve stands for a particular algorithm. From Figure 5, it is seen that the maximum throughput of HTN using LS, CS, and LS+CS algorithms are higher than when the DOR algorithm is used. Also the maximum throughput of HTN using the LS+CS algorithm is higher than when the LS and CS algorithms are individually used. The average transfer time of HTN using the LS+CS algorithm is lower than when the DOR, LS, and CS algorithms are used, but the difference among them is trivial. Therefore, with the uniform traffic, the LS, CS, and LS+CS algorithms achieve better dynamic communication performance than the DOR algorithm. And the LS+CS algorithm yields better dynamic communication performance than individual use of the LS and CS algorithms.

The hot-spot flit generation probability is assumed to be $P_h = 0.05$, i.e., 5% hot-spot traffic. In the HTN, the centered $4$ nodes that connect the sub-network module are considered as hot spot nodes. Figure 6 shows the message latency versus network throughput curve for hot spot traffic. From Figure 6, it is also seen that the maximum throughput of HTN using the LS, CS, and LS+CS algorithms is higher than when the DOR algorithm is used. Also the maximum throughput of HTN using the LS+CS algorithm is higher than when the LS and CS algorithms are individually used. The average transfer time of HTN using the LS+CS algorithm is lower than when the DOR, LS, and CS algorithms are used. Therefore, with the hot-spot traffic pattern, the LS, CS, and LS+CS algorithms achieve better dynamic communication performance than the DOR algorithm. And the LS+CS algorithm achieves better dynamic communication performance than either LS or CS algorithm.

Figure 7 depicts the result of simulations under bit-reversal traffic. From Figure 7, it is seen that the maximum throughput of HTN using the LS, CS, and LS+CS algorithms are higher than when the DOR algorithm is used. Also the maximum throughput of HTN using the LS+CS algorithm is higher than when the LS and CS algorithms are individually used. The average transfer time of HTN using the LS+CS algorithm is lower than when the DOR, LS, and CS algorithms are used. Therefore, with the bit-reversal traffic, the LS, CS, and LS+CS algorithms achieve better dynamic communication performance than the DOR. And the LS+CS algorithm yields better dynamic communication performance than the LS and CS algorithms used individually.

Figure 8 shows the latency versus throughput curve for bit-flip traffic. From Figure 8, it is seen that maximum throughput of HTN using the LS, CS, and LS+CS algorithms is higher than when the DOR algorithm is used.

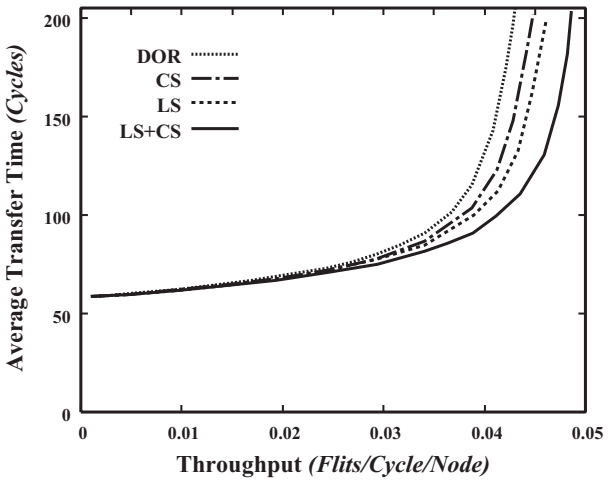



Figure 5. Comparison of dynamic communication performance of the HTN between DOR, LS, CS, and LS+CS algorithms with uniform traffic pattern: 1024 nodes, 3 VCs, 16 flits, and $q = 1$.

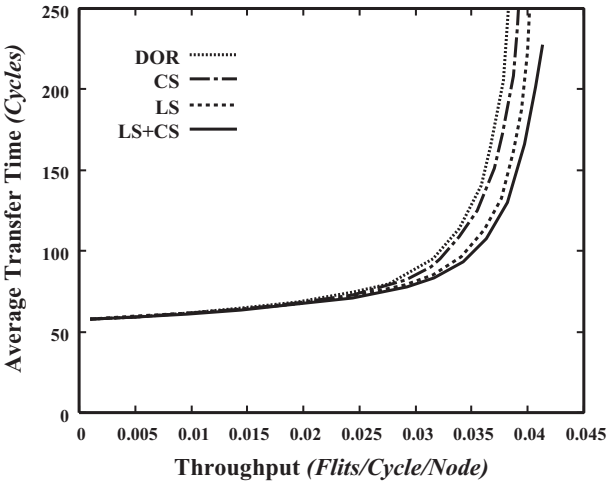



Figure 6. Comparison of dynamic communication performance of the HTN between DOR, LS, CS, and LS+CS algorithms with hot-spot traffic pattern: 1024 nodes, 3 VCs, 16 flits, $q = 1$, and 5% hot-spot traffic.

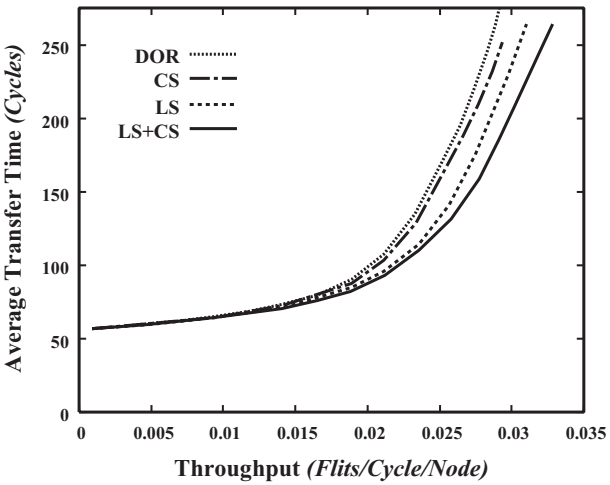



Figure 7. Comparison of dynamic communication performance of the HTN between DOR, LS, CS, and LS+CS algorithms with bit-reversal traffic pattern: 1024 nodes, 3 VCs, 16 flits, and $q = 1$.

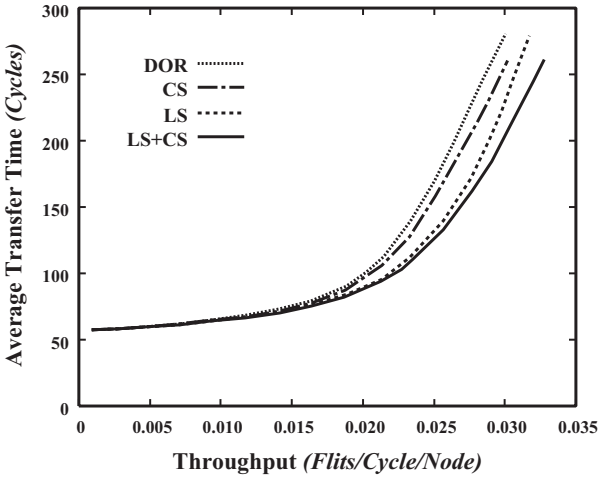



Figure 8. Comparison of dynamic communication performance of the HTN between DOR, LS, CS, and LS+CS algorithms with bit-flip traffic pattern: 1024 nodes, 3 VCs, 16 flits, and $q = 1$.

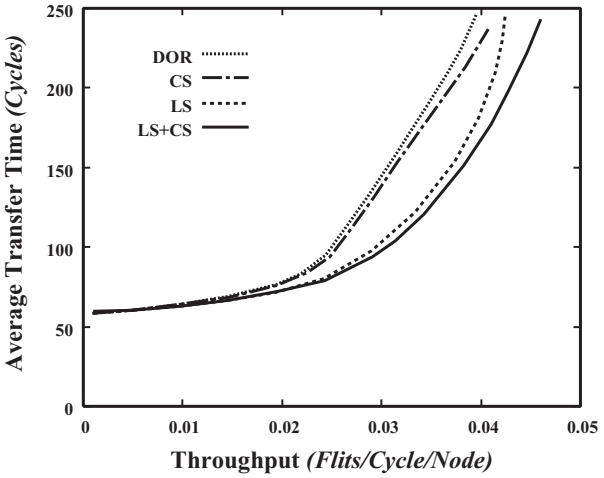



Figure 9. Comparison of dynamic communication performance of the HTN between DOR, LS, CS, and LS+CS algorithms with perfect shuffle traffic: 1024 nodes, 3 VCs, 16 flits, and $q = 1$.

Also, the maximum throughput of HTN using the LS+CS algorithm is higher than when the LS and CS algorithms are individually used. The average transfer time of HTN using the LS+CS algorithm is lower than when the DOR, LS, and CS algorithms are used. Therefore, with the bit-flip traffic pattern, the LS, CS, and LS+CS algorithms achieve better dynamic communication performance than the DOR algorithm. And the LS+CS algorithm yields better dynamic communication performance than the individual use of the LS and CS algorithms.

Figure 9 shows simulation result under perfect shuffle traffic. It is seen that maximum throughput of HTN using the LS, CS, and LS+CS algorithms is higher than when the DOR algorithm is used. Among them the LS+CS algorithm results highest throughput than that of others. The average transfer time of HTN using the LS+CS algorithm is lower than when the DOR, LS, and CS algorithms are used. Therefore, with the perfect shuffle traffic, the LS, CS, and LS+CS algorithms achieve better dynamic communication performance than the DOR. And the LS+CS algorithm yields better performance than the individual use of the LS and CS algorithms.

From these results, as shown in Figures 5, 6, 7, 8, and 9, it is clear that the selection algorithms (LS, CS, and LS+CS) outperform the DOR, especially in terms of network throughput. Using the LS+CS algorithm on HTN achieves better network throughput than the individual use of either the LS or the CS algorithms.

### *E. Performance Improvement*

The LS+CS algorithm efficiently routes the packet from source to its destination avoiding congestion by using the alternative paths. If the desired path is busy, another path leading towards the destination may be chosen. A number of pairs of nodes transmit packets simultaneously without blocking, which in turn increase the network throughput.

The maximum throughput and the average transfer time to achieve this maximum throughput, zero-load latency increasing of the HTN under different traffic patterns using LS+CS algorithm over DOR is presented in Figure 10. These parameters of LS+CS are normalized by the values of DOR and are shown in percentage. And their corresponding numerical values are presented in Table I. It is shown that in all traffic patterns the maximum throughput using the LS+CS algorithm is significantly higher than when the DOR algorithm is used. Also the average transfer time to achieve the maximum throughput is significantly reduced. Therefore, LS+CS algorithm significantly improves the dynamic communication performance of the HTN over the DOR algorithm is used.

In wormhole routing, the header flit contains the routing information. As the header flit advances along the specified route according to the routing algorithm, the remaining data flits of the packet follow the header flit in a pipelined fashion. The delay experienced by a packet in an interconnection network is its routing delay and queuing delay. A packet never contends for network resources with other packets in zero load. In this zero load situation, the queuing delay is zero. Thus, the delay experienced by a packet is the routing delay for header selection. Due to routing complexity, header selection in adaptive routing is complex to that of DOR, which results large zero load latency for the adaptive routing to that of DOR. The message latency is increasing with the increase of load in the network due to increase of queuing delay. The header selection overhead of the LS+CS algorithm is diminished with the increase of load, since it routes more packets by providing multiple path options.

Figure 10(c) depicts the increase of zero load latency due to the router delay for header selection in LS+CS algorithm over the DOR algorithm. In all traffic patterns, the zero load latency of the LS+CS algorithm is higher than that of DOR algorithm. Table I shows the numerical value of zero load latency of DOR and LS+CS algorithms and its increase in LS+CS algorithm over DOR algorithm. Due to limited adaptivity and simple routing principle, the zero load latency increase in LS+CS algorithm is very

TABLE I.
PERFORMANCE IMPROVEMENT USING SELECTION ALGORITHM OVER DIMENSION-ORDER ROUTING

| Traffic Patterns | Maximum Throughput | | Latency for Max. Throughput | | Zero Load Latency (ZLL) | | Throughput Enhancement | Latency Reduction | ZLL Increase |
|---|---|---|---|---|---|---|---|---|---|
| | DOR | LS+CS | DOR | LS+CS | DOR | LS+CS | % | % | % |
| Uniform | 0.04317 | 0.04861 | 216.01 | 203.23 | 58.40 | 58.61 | 12.60 | 6.29 | 0.36 |
| Hot-Spot | 0.03833 | 0.04130 | 251.11 | 227.41 | 57.59 | 57.84 | 7.75 | 10.42 | 0.43 |
| Bit-Reversal | 0.02951 | 0.03281 | 287.09 | 264.11 | 57.11 | 57.39 | 11.18 | 8.70 | 0.49 |
| Bit-Flip | 0.03005 | 0.03279 | 280.49 | 261.38 | 57.16 | 57.25 | 9.12 | 7.31 | 0.16 |
| Perfect Shuffle | 0.03958 | 0.04600 | 246.42 | 243.19 | 58.31 | 59.50 | 16.22 | 1.33 | 0.33 |

low in all traffic patterns, less than 0.5%. With this 0.5% routing delay increase in the selection algorithm enhance the satuarion throughput upto 16% as shown in Table I.

## V. CONCLUSION

In this paper, we have proposed three adaptive routing algorithms, CS, LS, and LS+CS with DOR. The proposed adaptive routing algorithms – CS, LS, and LS+CS – are simple and efficient for using the physical links and virtual channels of an HTN to improve dynamic communication performance. The freedom from deadlock of the proposed CS, LS, and LS+CS algorithms using 3 VCs has been proved. Using the routing algorithms described in this paper and several traffic patterns, we have evaluated the dynamic communication performance of the HTN. The average transfer time of the HTN using the CS, LS, and LS+CS algorithms is lower than when the DOR algorithm is used, but the differences are not impressive. On the other hand, maximum throughput using the CS, LS, and LS+CS algorithms is higher than when the DOR algorithm is used. Efficient use of physical links and virtual channels improves the dynamic communication performance significantly. A comparison of dynamic communication performance reveals that the LS+CS algorithm outperforms all other algorithms; an HTN using the LS+CS algorithm yields low latency and high throughput, which are indispensable for the high performance of MPC system.

This paper focused on 3 simple adaptive routings for the efficient use of physical links and VCs to improve the dynamic communication performance of the HTN. Issues for future work include the replacement of the long length electronic links by optical links, i.e., to study the architecture and performance of opto-electronic (hybrid)-HTN [39].

## ACKNOWLEDGMENT

The author was a Post Doctoral researcher at JAIST supported by JSPS. This work is supported in part by JSPS fellowship program and Grand-in-Aid for Scientific Research (B), 21-09058, JSPS, Japan. The authors are grateful to the anonymous reviewers for their constructive comments which helped to greatly improve the clarity of this paper. The preliminary version of this paper has been published in the proceedings of the $13^{th}$ ICCIT, pp. 204 - 209, Dhaka, Bangladesh, 2010 [31].

## REFERENCES

[1] Y.R. Potlapalli, Trends in Interconnection Network Topologies: Hierarchical Networks, *Int. Conf. on Parallel Processing Workshop*, pp. 24-29, 1995.

[2] A. El-Amawy and S. Latifi, "Properties and Performance of Folded Hypercube," IEEE Trans. Parallel and Distrib. Syst., vol. 2, no. 1, pp. 31-42, 1991.

[3] A. Esfahanian, L.M. Ni, and B.E. Sagan, "The Twisted $n$-Cube with Application to Multiprocessing," IEEE Trans. Comput., vol. 40, no. 1, pp. 88-93, 1991.

[4] J.M. Kumar and L.M. Patnaik, "Extended Hypercube: A Hierarchical Interconnection Network of Hypercube," IEEE Trans. Parallel Distrib. Syst., vol. 3, no. 1, pp. 45-57, 1992.

[5] N.F. Tzeng and S. Wei, "Enhanced Hypercube," IEEE Trans. Comput., vol. 40, no. 3, pp. 284-294, 1991.

[6] S.G. Ziavras, "A Versatile Family of Reduced Hypercube Interconnection Network," IEEE Trans. Parallel Distrib. Syst., vol. 5, no. 11, pp. 1210-1220, 1994.

[7] S. Horiguchi, "New Interconnection for Massively Parallel and Distributed System," Research Report, Grant-in-Aid Scientific Resh, pro. no.09044150, JAIST, pp. 1-72, 1999.

[8] S. Horiguchi and T. Ooki, Hierarchical 3D-torus interconnection network, *Proc. of ISPAN'00, Texas, USA*, pp.50-56, 2000.

[9] V.K. Jain and S. Horiguchi, "VLSI Considerations for TESH: A New Hierarchical Interconnection Network for 3-D Integration," IEEE Trans VLSI Syst., vol.6, no. 3, pp. 346-353, 1998.

[10] F.P. Preparata and J. Vuillemin, The Cube-Connected Cycles: A Versatile Network for Parallel Computation, *Journal of ACM*, vol.24, no.5, pp.300-309, May 1981.

[11] W.J. Dally, Performance Analysis of $k$-ary $n$-cube Interconnection Networks, *IEEE Trans. on Computers*, vol. 39, no. 6, pp. 775-785, 1990.

[12] M.M. Hafizur Rahman and S. Horiguchi, HTN: A New Hierarchical Interconnection Network for Massively Parallel Computers, *IEICE Trans. on Inf. & Syst.*, vol.E86-D, no.9, pp. 1479-1486, 2003.

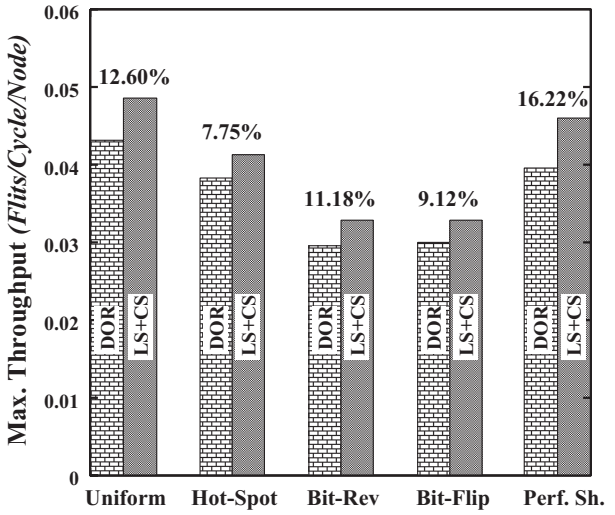
[13] W.J. Dally and C.L. Seitz, "Deadlock Free Message Routing in Multiprocessor Interconnection Networks," IEEE Trans. on Computers, C36-5, pp. 547-553, 1987.

[14] Jose Duato, A New Theory of Deadlock Free Adaptive Routing in Wormhole Networks, *IEEE Trans. Parallel and Distrib. Syst.*, vol.4., no.12, pp. 1320-1331, 1993.

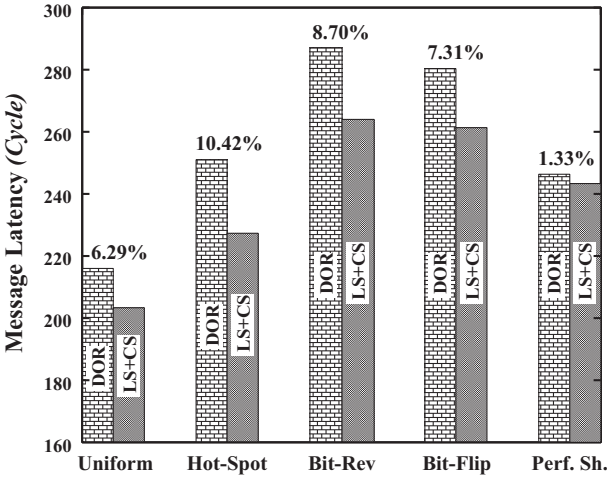
[15] Xin Yu and T.S. Li, On shortest path routing algorithm in crossed cube-connected ring networks, *Proc. of the CyberC*, pp. 348 - 354, 2009.

[16] N.R. Vaish and U. Shrivastava, On a deadlock and performance analysis of ALBR and DAR algorithm on X-Torus topology by optimal utilization of Cross Links and minimal lookups, *The Journal of Supercomputing, Springer*, pp. 1-37, Dec. 2010.

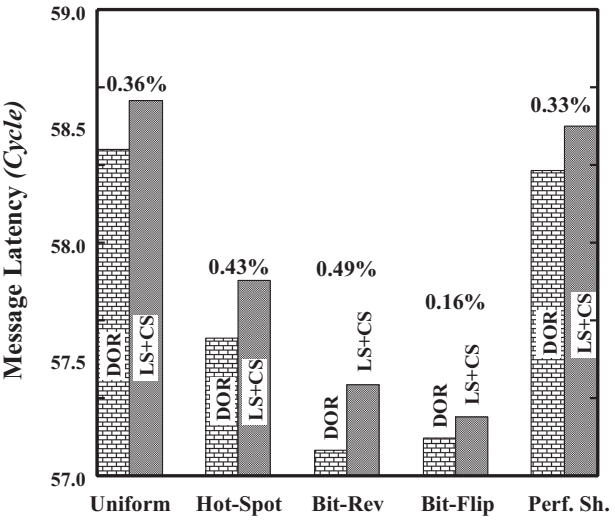
[17] N. Jiang, J. Kim, and W.J. Dally, Indirect adaptive routing on large scale interconnection networks, *Proc. of the* $36^{th}$ *ISCA*, pp. 220 - 231, 2009.




(a)




(b)




(c)

Figure 10. Dynamic communication performance enhancement by LS+CS algorithm over DOR algorithm (a) Maximum throughput enhancement, (b) Message latency reduction to achieve this maximum throughput enhancement, and (c) Zero load latency increasing.

[18] Y. Miura, S. Horiguchi, M. Fukushi, Adaptive Routing Algorithms of the Interconnection Networks TESH for Fine-Graded Parallel Processing, *IEICE Trans. on Inf. & Syst.*, vol.J91-D, no.5, pp. 1202-1215, 2008 (Japanese).

[19] Andrew A. Chien and Jae H. Kim, Planer-Adaptive Routing: Low-cost Adaptive Networks for Multiprocessors, *Journal of the ACM*, vol.42, no.1, pp.91-123, 1995.

[20] L.M. Ni and P.K. McKinley, A Survey of Wormhole Routing Techniques in Direct Networks, *IEEE Computer*, vol.26, no.2, pp. 62-76, 1993.

[21] W.J. Dally and C.L. Seitz, The Torus Routing Chip, *Journal of Distrib. Computing*, vol.1, no.3, pp. 187-196, 1986.

[22] W.J.Dally, "Virtual-Channel Flow Control," IEEE Trans. Parallel and Distrib. Syst., vol.3, no.2, pp.194-205, 1992.

[23] M.M. Hafizur Rahman and S. Horiguchi, A deadlock-free routing algorithm using minimum number of virtual channels and application mappings for Hierarchical Torus Network, *IJHPCN*, vol. 4, no. 3/4, pp. 174-187, 2006.

[24] M.M. Hafizur Rahman and S. Horiguchi, Dynamic Communication Performance of a Hierarchical Torus Network under Non-uniform Traffic Patterns, *IEICE Trans. on Inf. & Syst.*, vol.E87-D, no.7, pp.1887-1896, 2004.

[25] M.M. Hafizur Rahman and S. Horiguchi, Routing performance enhancement in hierarchical torus network by link-selection algorithm, *JPDC*, vol.65, no.11, pp. 1453 1461, 2005.

[26] L. Schwiebert and D. N. Jayasimha, A Necessary and Sufficient Condition for Deadlock-Free Wormhole Routing, *Journal of Parallel and Distributed Computing*, vol. 32, no. 1, pp. 103-117, 1996.

[27] W. Feng and K.G Shin, The effect of virtual channels on the performance of wormhole algorithms in multicomputer networks, *UM directed Study Report*, May 1994.

[28] H.Sarbazi-Azad, L.M. Mackenzie, and M.O. Khaoua, The Effect of the Number of Virtual Channels on the Performance of Wormhole-Routed Mesh Interconnection Networks, *Proc. of the 16th UKPEW*, Glasgow, 2000.

[29] Xiaoding Zhang, System Effects of Interprocessor Communication Latency in Multicomputers, *IEEE Micro*, vol.11, no.2, pp. 12-55, 1991.

[30] R. Holsmark, S. Kumar, M. Palesi, and A. Mekia, HiRA: A Methodology for Deadlock Free Routing in Hierarchical Networks on Chip, *Proc. of the* $3^{rd}$ *ACM/IEEE NOCS*, pp. 2-11, 2009.

[31] M M Hafizur Rahman, Y. Sato, Y. Inoguchi, Dynamic Communication Performance Enhancement in Hierarchical Torus Network by Selection Algorithm, *Proc. of the* $13^{th}$ *ICCIT*, pp. 204-209, Dhaka, Bangladesh, 2010.

[32] L. Schwiebert, A Performance Evaluation of Fully Adaptive Wormhole Routing including Selection Function Choice, *IEEE Int'l. Performance, Computing, and Communications Conference*, pp. 117-123, 2000.

[33] G.F. Pfister and V.A. Norton, "Hot Spot Contention and Combining in Multistage Interconnection Networks," IEEE Trans. Comput., vol. 34, no. 10, pp. 943-948, 1985.

[34] F. Petrini and M. Vanneschi, "$k$-ary $n$-trees: High Performance Networks for Massively Parallel Architectures," Tech. Report TR-95-18, Universita di Pisa, Dec. 1995.

[35] H.H. Najaf-abadi and H. Sarbazi Azad, The Effects of Adaptivity on the Performance of the OTIS-Hypercube Under Different Traffic Patterns, *Proc. of IFIP Int'l. Conf. NPC2004,*, LNCS, pp.390-398, 2004.

[36] J.H. Kim and A.A. Chien, An evaluation of planar-adaptive routing (PAR), *Proc. 4th IEEE Symp. Parallel Distrib. Processing*, pp.470–478, New-York, 1992.

[37] P.R. Miller, Efficient Communications for Fine-Grain Distributed Computers, *Ph.D. Dissertation, Southampton University*, U.K., 1991.

[38] Jose Duato, Sudhakar Yalamanchili, and Lionel Ni, *Interconnection Network: an Engineering Approach*, IEEE CS Press, Los Alamitos, California, USA, 1997.
[39] L. Xiao and K. Wang, Reliable Opto-Electronic Hybrid Interconnection Network, *Proc. of the* $9^{th}$ *I-SPAN*, pp. 239-244, 2008.

**M.M. Hafizur Rahman** received his B.Sc. degree in Electrical and Electronic Engineering from Khulna University of Engineering and Technology (KUET), Khulna, Bangladesh, in 1996. He received his M.Sc. and Ph.D. degree in Information Science from the Japan Advanced Institute of Science and Technology (JAIST) in 2003 and 2006, respectively. Dr. Rahman is now an assistant professor in the Dept. of Computer Science, Kulliyyah of Information and Communication Technology (KICT), International Islamic University, Malaysia (IIUM), Malaysia. Prior to join in the IIUM, he was an associate professor in the Dept. of CSE, KUET, Khulna, Bangladesh. He was also a visiting researcher in the School of Information Science at JAIST and a JSPS postdoctoral research fellow at Graduate School of Information Science (GSIS), Tohoku University, Japan & Center for Information Science, JAIST , Japan in 2008 and 2009 & 2010-2011, respectively. His current research include parallel and distributed computer architecture, hierarchical interconnection networks and optical switching networks. Dr. Rahman is member of the IEB of Bangladesh.

**Yukinori Sato** received the BS, MS, and Ph.D. degree in Information Science from Tohoku University in 2001, 2003, 2006, respectively. From 2006, he engaged in embedded processor system design in Sendai Software Development center of FineArch Inc. and also became a joint research member at Tohoku University. From 2007, he has been working at Japan Advanced Institute of Science and Technology (JAIST) as an assistant professor. His research interests include high-speed and low-power computer architectures and reconfigurable computing. Dr. Sato is a member of the IEEE, ACM, IEICE and IPS of Japan.

**Yasushi Inoguchi** received his B.E. degree from Department of Mechanical Engineering, Tohoku University in 1991, and received MS degree and Ph.D. from Japan Advanced Institute of Science and Technology (JAIST) in 1994 and 1997, respectively. He is currently an Associate Professor of Center for Information Science at JAIST. He was a research fellow of the Japan Society for the promotion of Science from 1994 to 1997. He is also a researcher of PRESTO program of Japan Science and Technology Agency from 2002 to 2006. His research interest has been mainly concerned with parallel computer architecture, interconnection networks, GRID architecture, and high performance computing on parallel machines. Dr. Inoguchi is a member of IEEE and IPS of Japan.