Adaptive routing on the Recursive Diagonal Torus

A. Funahashi
Dept. of Computer Science
Keio University
Yokohama, JAPAN 223

A. Jouraku
Dept. of Computer Science
Keio University
Yokohama, JAPAN 223

H. Amano
Dept. of Computer Science
Keio University
Yokohama, JAPAN 223

Abstract

Recursive Diagonal Torus, or RDT consisting of recursively structured tori is an interconnection network for massively parallel computers. By adding remote links to the diagonal directions of the torus network recursively, the diameter can be reduced within $log_2(Nodenumber)$ with smaller number of links than that of hypercube.

For an interconnection network for massively parallel computers, routing algorithms which can bypass a faulty or congested node are essential. Although the conventional vector routing is a simple and near-optimal method, it can only use a deterministic path. In this paper, deadlock free adaptive routing algorithms on RDT are proposed and evaluated. One is called D-RDT, and the other algorithm is called FD-RDT. These algorithms are based on Duato's necessary and sufficient condition.

With these methods, virtual channels are effectively used while paths with redundant routing steps are prohibited. From simulation results, it is shown that both the throughput and latency are much improved especially by using algorithm FD-RDT.

keywords: interconnection network, adaptive routing, deadlock avoidance, wormhole routing, RDT

1 Introduction

Communication network is one of the critical components of a highly parallel multicomputer. Recently, multicomputers providing more than a thousand computation nodes are commercially available, and efforts have been exerted to implement Massively Parallel Computers (MPCs) with tens of thousands nodes. In these systems, the connection topology often dominates the system performance.

Instead of hypercube used in first-generation multicomputers, most recent machines take the 2-D or 3-D mesh (torus) network[1]. Although the diameter of a mesh network is large ($O(\sqrt{M})$ or $O(\sqrt[3]{M})$ for M nodes), it only requires four or six links per node unlike the hypercube which requires $log_2 M$ links per node.

However, in an MPC with more than ten thousands nodes, the large diameter of the mesh network is intolerable. To address this problem, we proposed a novel extension of mesh network called Recursive Diagonal Torus (RDT) [3], which consists of recursively structured mesh (torus) connection. It supports a smaller diameter and degree than that of the hypercube if the number of nodes is 1000-10000. Through the computer simulation, the bandwidth and latency are much improved compared with 2-D/3-D tori [3]. The router chip providing the vector routing algorithm with multicasting was implemented for a massively parallel machine JUMP-1[2].

In this paper, deadlock-free adaptive routing algorithms on RDT are proposed and evaluated. In Section 2, the structure of RDT and the vector routing algorithm are briefly introduced. An adaptive routing using minimal paths based on Duato's method is proposed in Section 3. The evaluation through computer simulation is described in Section 4, and we describe the conclusion in Section 5.

2 Interconnection Network: RDT

Recursive Diagonal Torus (RDT) is a novel class of networks which consists of recursively structured mesh (torus) connections of meshes with different sizes in the diagonal directions[3].

When four links are added between node (x, y) and nodes $(x \pm n, y \pm n)$ (n: *cardinal number*) respectively, additional links result in a new torus-like network. New torus-like network is formed at an angle of 45 degrees to the original torus, and the grid size is $\sqrt{2}n$ times that of the original torus. We call this new torus-like network the rank-1 torus. On the rank-1 torus, we can form another torus-like network (rank-2 torus) by providing additional links in the same manner. Fig-

ure 1 shows rank-1 and rank-2 tori when n is 2. The RDT consists of such recursively formed tori.

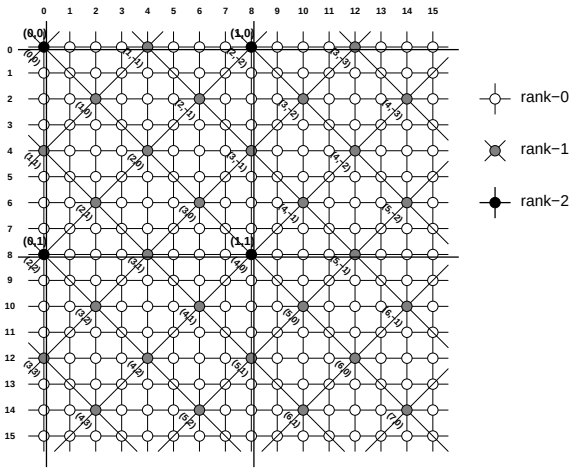


Figure 1: Upper rank tori

RDT(n,R,m) can be defined as a class of networks in which each node has links to form base (rank-0) torus and m upper tori (the maximum rank is R) with cardinal number n. Note that, each node can select different rank of upper tori from others.

The RDT in which every node has links to form all possible upper tori is called the perfect RDT (PRDT(n,R)) where n is the cardinal number (usually, 2) and R is the maximum rank. Although PRDT is unrealistic due to its large degree (4(R+1)), it is important as the basis for establishing routing algorithm, broadcasting/multicasting, and other message transfer algorithms.

For a system with thousand of nodes, the RDT whose degree is 8 and the maximum rank of upper tori is 4, that is, RDT(2,4,1) is suitable. In the RDT, each node can select different rank tori from others. Thus, the structure of the RDT(2,4,1) also varies with the rank of tori which are assigned to each node. This assignment is called the *torus assignment*. Various torus assignment strategies can be selected considering the traffic of the network. The major torus assignment for RDT(2,4,1) is called RDT(2,4,1)/α[3]. Torus assignment used in the RDT(2,4,1)/α is shown in Figure 2.

2.1 The vector routing

The vector routing is an assignment independent routing algorithm which represents the route of a message with a combination of unit vectors each of which corresponds to each rank of tori.

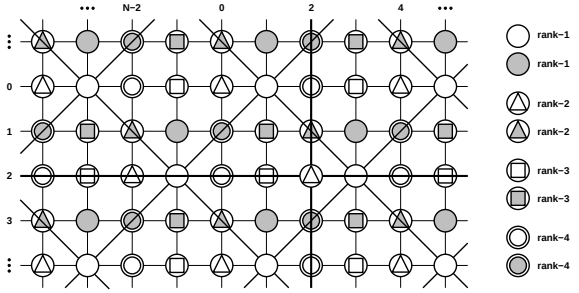


Figure 2: Torus assignment for the RDT(2,4,1)/α

On the torus structure, a vector from a source node to the destination node is represented with a vector $\vec{A} = a\vec{x_0} + b\vec{y_0}$ where $\vec{x_0}$ and $\vec{y_0}$ are unit vectors of the base (rank-0) torus. The goal of the routing algorithm is to represent the vector $\vec{A}$ with a combination of vectors each of which corresponds to a unit vector of each rank of torus.

First, the direction of the unit vector corresponding to each rank torus must be defined. Here, the direction of the unit vector for each rank torus is changed clockwise at an angle of 45 degrees. That is, the unit vectors of rank-(i+1) torus $\vec{x}_{i+1}, \vec{y}_{i+1}$ can be represented with the unit vectors of rank-i $\vec{x_i}, \vec{y_i}$ as follows:

$$\vec{x}_{i+1} = n\vec{x_i} + n\vec{y_i} \quad (1)$$

$$\vec{y}_{i+1} = -n\vec{x_i} + n\vec{y_i} \quad (2)$$

First, the target vector $a\vec{x_0}+b\vec{y_0}$ is represented with a combination of $\vec{x_1}$,$\vec{y_1}$,$\vec{x_0}$ and $\vec{y_0}$ as follows:

$$a\vec{x_0} + b\vec{y_0} = g\vec{x_1} + f\vec{y_1} + j\vec{x_0} + k\vec{y_0} \quad (3)$$

Here, we select maximum g and f in order to use the upper torus as possible. From equations (1) and (2), maximum integers for g and f are represented as follows:

$$g = \frac{a+b}{2n}, f = -\frac{a-b}{2n}$$

In order to minimize j and k corresponding to the remaining unit vectors of the rank-0 torus (thus, the required message transfers using the rank-0 torus), the integer divisor used here is rounded to the nearest whole number (If the remainder is greater than n, increment the divisor).

Thus, j and k are represented with g and f:

$$a\vec{x_0} + b\vec{y_0} = g(n\vec{x_0} + n\vec{y_0}) + f(-n\vec{x_0} + n\vec{y_0}) + j\vec{x_0} + k\vec{y_0}$$

$$a = ng - nf + j, b = ng + nf + k$$
$$j = a - ng + nf, k = b - ng - nf$$

.

Then, $g\vec{x_1}+f\vec{y_1}$ are represented with a combination of vector $\vec{x_2}$, $\vec{y_2}$, $\vec{x_1}$ and $\vec{y_1}$ in the same manner. By iterating this process to the maximum rank, vectors for message routing are obtained.

The routing vectors for each rank are obtained in the array $vector[rank]$.

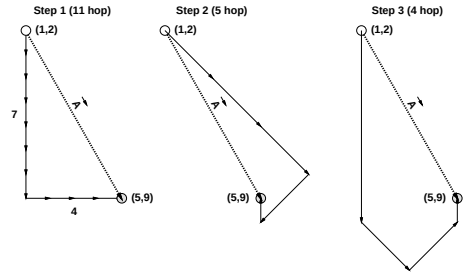


Figure 3: An example of the vector conversion

Figure 3 shows an example of a vector from (1,2) to (5,9) converted into a combination of unit vectors of rank-0, rank-1 and rank-2.

2.2 Deadlock-free routing algorithm for RDT(2,4,1)/α

The e-cube algorithm [4] is a common deadlock-free deterministic routing algorithm for k-ary n-cubes. Since the RDT consists of recursive structured k-ary 2-cubes, the e-cube routing algorithm can be applied.

In the e-cube routing algorithm, the packet is routed in order of dimension, most significant dimension first. In each dimension i, a packet is routed in the dimension until it reaches to a node whose subscript matches the destination address in the i th position.

This algorithm is easily extended for PRDT. Though PRDT or RDT has k-ary 2-cubes at each rank, using e-cube routing at each rank makes vector routing algorithm deadlock-free at each rank. If we restrict the vector routing algorithm to use the vector in fixed order(thus, use the highest rank vector first and base rank vector last) the vector routing algorithm would become deadlock-free.

For the RDT(2,4,1)/α, the above method is not available because the channel of the base torus is used when the message is transferred between upper rank tori. To cope with this problem, two virtual channels and some extension for routing algorithm are required.

Extension

1. In RDT(2,4,1)/α, nodes are ordered so that their upper tori are in the descending order to $+x$ direction, that is, 4, 3, 2 and 1 (See Figure 2).

2. Add two virtual channels $C_U(up)$ and $C_D(down)$. These two virtual channels are associated with the links for $+x$ and $-x$ direction of the base (rank 0) torus respectively.

3. According to the e-cube routing algorithm, tori (vectors) are used in descending order of ranks. When a packet is transferred to the node with the highest rank torus(r_{max}), the packet is transferred to $-x$ direction through the virtual channel C_U. After the packet is routed on the rank r_{rmax} torus, it is sent to $+x$ direction through the virtual channel C_D. Then, the packet is routed on the rank r_{max-1} torus. These steps are repeated until the packet is reached to the lowest rank used in the vector.

The proof of deadlock-free is shown in [3]. Figure 4 illustrates the extended routing algorithm.

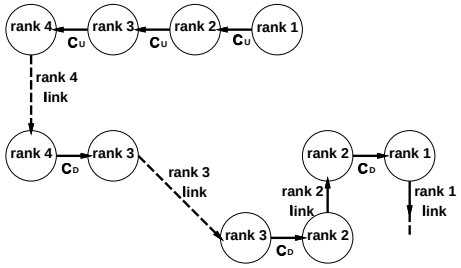


Figure 4: The e-cube routing on the RDT(2,4,1)/α

3 Adaptive routing on RDT(2,4,1)/α

Adaptive routing is a technique to select the route of packet dynamically. When a packet encounters a faulty or congested node, the packet can select another bypassing route. The vector routing is useful for a basis of an adaptive routing, since alternative routes can be easily obtained by changing the order of vectors. However, we must not forget that an adaptive routing has a possibility of deadlock. There are a lot of researches on deadlock-free adaptive routing techniques[5]. These techniques are classified into two methods: using only minimal paths, and using alternative paths with additional routing steps. The former method does not require extra routings while the latter can use alternative routes more flexibly. First, deadlock free adaptive routings with minimal routes are proposed for the RDT based on Duato's protocol. Then, another algorithm which permits redundant routing steps is proposed.

3.1 Duato's protocol in k-ary n-cube

Duato states a general theorem defining a criterion for deadlock freedom and then uses the theorem to propose a fully adaptive, profitable, progressive protocol[6], called Duato's protocol. The theorem states that by separating virtual channels on a link into restricted and unrestricted partitions, a fully adaptive routing can be performed and yet be deadlock-free. This is not restricted to a particular topology or routing algorithm. Cyclic dependencies between channels are allowed, provided that there exists a connected channel subset free of cyclic dependencies.

Simple description of Duato's protocol is as follows.

a. Provide that every packet can always find a path toward its destination whose channels are not involved in cyclic dependencies(escape path).

b. Guarantee that every packet can send to any destination node using an escape path and the other path on which cyclic dependency is broken by the escape path.

By selecting these two routes a. and b. adaptively, deadlock can be prevented. Duato applied this method to k-ary n-cube[6].

3.2 Applying Duato's protocol on RDT

Here, we apply routing algorithm called "D-RDT" for RDT. D-RDT is already proposed by us on [7]. Please refer the paper for more information.

3.2.1 D-RDT

Definition 1 : Algorithm: D-RDT

1. *Provide an escape path C_1 on a torus of RDT as same as the case of Duato's protocol applied on k-ary 2-cube. Two virtual channels are required to provide escape path C_1.*

2. *Next, the order of rank usage is restricted. Let X_i and Y_i be channel of each dimension in the rank i torus. Use the channel in the X first and descending order of the rank. That is, for RDT(2,4), the channel is used in the following order*
   $X_3 \rightarrow Y_3 \rightarrow X_2 \rightarrow Y_2 \rightarrow X_1 \rightarrow Y_1$
   We refer this escape path C_1'.

3. *Provide virtual channel C_{Fn} not for C_1' but for C_1 in each rank. C_{Fn} channels can cross dimensions in any order following a minimal path, but must cross ranks in descending order.*

□

Figure 5 illustrates the virtual channel C_{Fn} in this algorithm. The channel C_{Fn} is assigned to the escape path C_1 of each rank. Therefore, the order of ranks is restricted.

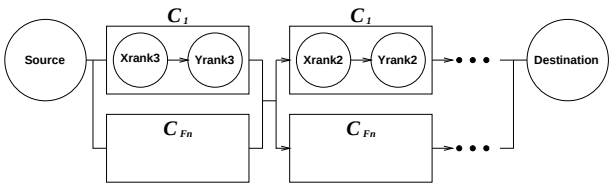


Figure 5: An arrangement of adaptive virtual channel C_{Fn}

Figure 6 illustrates the possible path and impossible path for algorithm D-RDT. The path (b) which uses rank 2 before rank 3 is prohibited in the algorithm D-RDT, since the rank is not used in the descending order. On the contrary, path (c) in which the unit vectors of rank 1 and rank 3 are directed opposite to each other (thus, it doesn't satisfy minimal routing) is allowed in the algorithm D-RDT.

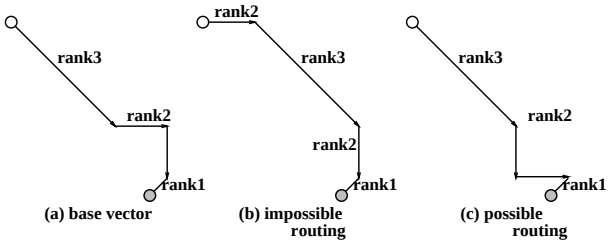


Figure 6: Examples of vectors in algorithm D-RDT

Theorem 1 *Algorithm D-RDT is deadlock-free.* □

Proof C_1 is the same escape path used in Duato's protocol, and is deadlock free. From Duato's theorem[6], algorithm D-RDT is deadlock-free in each rank of torus. Since the order of used rank is the same as the e-cube routing[4], C_1 nor C_{Fn} in any rank does not cause a cycle each other. Therefore, Algorithm D-RDT is deadlock-free.□

Since the order of ranks is restricted in D-RDT, there are some problems for typical structure of RDT(such as RDT(2,4,1)/α). The vector is required to be used in descending order of the rank, but RDT(2,4,1)/α doesn't have all rank torus at each

node. If routing requires moving across a torus for which there are no links at the current node, the local routing is used to reach the node which has the required torus. To achieve this local routing, two extra virtual channels (C_U, C_D) are required(as shown in Section 2). Duato's protocol requires at least two virtual channels for k-ary 2-cube, and one for adaptive routing(C_{Fn}), so consider the local routing, D-RDT requires three (y dimension) or four (x dimension) virtual channels to achieve adaptive routing.

3.2.2 FD-RDT

D-RDT is restricted because of the complicated structure of RDT(2,4,1)/α. If the structure of RDT is not so complicated like PRDT(because there is no local routing in PRDT), fully adaptive routing algorithm using Duato's protocol is easily implemented. Though algorithm D-RDT is restricted that it can only use the vectors in descending order, we provided a new routing algorithm for any structure of RDT called "FD-RDT".

Definition 2 : Algorithm: FD-RDT

1. *Provide an escape path C_1 and C_1' as same as the case of D-RDT.*
2. *Provide virtual channel C_{Fn} not for C_1' but for C_1 in each rank(This is as same as D-RDT). C_{Fn} channels can cross dimensions in any order following a minimal path, but must cross ranks in descending order.*
3. *Add a virtual channel C_R for base (rank 0) torus. This virtual channel is used to achieve the rank order independent routing on any structure of RDT. C_U and C_D are added only to one direction($+x$ or $-x$) at each node, but C_R is added to each direction(i.e. 4 direction) at each node of base torus.*
4. *When using D-RDT, the order of rank(vector) usage is restricted (i.e. D-RDT cannot change the order of C_U and C_D). New virtual channel C_R is used to prevent this restriction while C_R can be used when FD-RDT wants to use the rank which is not descending order. Using C_R between different ranks and C_{Fn} at each rank, fully adaptive routing is achieved.*
5. *The restriction for C_R is as follows:*
   When a packet is routed to upper rank torus by C_U, C_R cannot be used when the packet wants to move to previous node's upper rank(r) torus and lower rank ($\leq r$) torus (i.e. the packet must use C_D to move to such rank torus).

□

The reason for restriction of C_R (i.e. the order of C_U and C_D is restricted even using the channel C_R) is represented as follows:

If the packet was traversed along C_U and C_D and there exists a vector whose rank is higher than that of current node's upper rank torus(ex. current node's upper rank torus is 3 and there exist rank 4 vector to be routed), it is impossible to route these vector to the destination node by e-cube routing for RDT, and it will cause deadlock.

Proof C_1' is the same escape path used in D-RDT and is deadlock free, and algorithm D-RDT is deadlock-free. D-RDT guarantees that using each rank torus in descending order, the routing algorithm is deadlock free. However C_R can use in any order of rank, C_R won't prevent the order of C_U and C_D. Thus, there exists at least one deadlock-free path(escape path) to a destination node for each path. From Duato's theorem, Algorithm FD-RDT is proved to be deadlock-free.□

Unlike D-RDT, FD-RDT can route any path presented in Figure 6, and this is the benefit of the algorithm.

Providing one more virtual channel to each direction, FD-RDT requires four (y dimension) or five (x dimension) virtual channels to achieve adaptive routing.

4 Performance evaluation and comparison

Our previous paper[7] has no comparison between e-cube routing and adaptive routing. In this section, performance of the above two algorithms are evaluated by computer simulation, and compared with the simple deterministic e-cube routing.

4.1 The RDT simulator

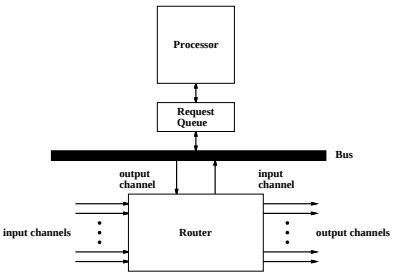


Figure 7: The construction of each node

The RDT simulator used here is a flit-level simulator written in C++. Network size, the number of virtual channel, and packet length are selected just by changing parameters. As shown in Figure 7, each node consists of a processor, request queue and the router which provides bidirectional channels. Each node is connected with neighboring nodes by using bidirectional channels attached to the router.

4.2 Router model

As shown in Figure 8, a simple router model consisting of channel buffers, crossbar, link controller (LC), virtual channel controller (VC) and control circuits is used.

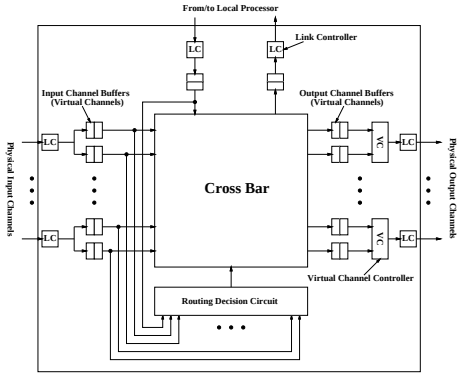


Figure 8: Router model

Two channel selection policies: *input selection policy*, and *physical channel transmission policy* are chosen in this simulator.

4.3 Simulation parameters

The destination node of a packet is determined by the traffic pattern in the simulator. Three traffic patterns are used here:

- *uniform*
  All destination nodes are selected randomly, and so distributed uniformly.
- *nonuniform*
  - *bit-reversal:*
    In the bit reversal traffic, a node with the identifier $(a_0, a_1, \cdots, a_{n-2}, a_{n-1})$ sends a packet to the node whose identifier is the bit reversal $(a_{n-1}, a_{n-2}, \cdots, a_1, a_0)$ of the source node.
  - *matrix transpose:*
    The node (x, y) sends a packet to the node $(k-y-1, k-x-1)$(k is the number of nodes in each dimension), or send to the node $(k-x-1, k-y-1)$ when $x+y = k-1$.

The following two measures are used for evaluations.

Network latency: Let the time when a node p inserts the first flit of a packet into the input buffer be t_0, and the time when the tail flit of the packet is send to the processor at destination node q be t_1. Here, we call $T_{lat}(p, q) = t_1 - t_0$ the *network latency*, and use the measure of the network performance.

Throughput: Throughput is the maximum amount of information delivered per time unit. Here, throughput could be measured in flits per node in each clock cycle.

When the network is saturated, the execution of simulation is aborted.

Here, the simulation parameters are set as Table 1.

Table 1: Simulation parameters

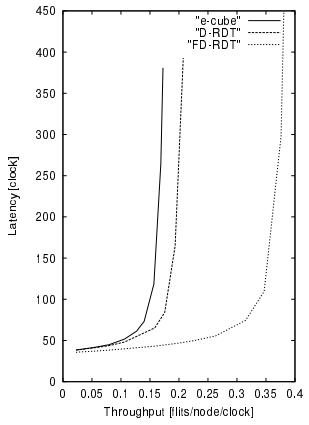
Network size	32 × 32 (i.e. 1024 node)
Packet length	16 flits (fixed)
Packet header length	2 flits (fixed)
Routing method	wormhole
Request queue size	100 (packets)
Packet generation time	1 clock
Routing and cross bar setup time	1 clock
Flit transfer time (from input buffer to output buffer)	1 clock
Flit transfer time (at physical link)	1 clock
Simulation time	10000 clock (ignore the first 1000 clock)
Input selection policy	distance traveled
Physical channel transmission policy	round robin
The number of virtual channels	2 or 3 (e-cube) 3 or 4 (D-RDT) 4 or 5 (FD-RDT)

4.4 Simulation results

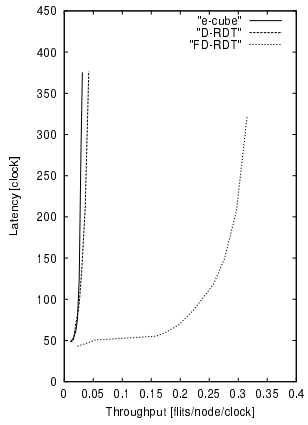
From Figure 9 (a), adaptive routing shows higher performance than deterministic routing with the uniform traffic. The latency of the FD-RDT is better than that of D-RDT. It demonstrates that high flexibility of the FD-RDT improves the performance even under the uniform traffic.

Figure 9 (b) shows the case of bit-reversal traffic. In this case, a little performance improvement is obtained by D-RDT compared with the e-cube (deterministic) routing, while FD-RDT achieves a certain improvement like the case of uniform traffic. With the bit-reversal traffic, each node always sends a packet to the same destination node, and a blocking packet continuously interferes sending packets one after another. Since the deterministic routing can only select

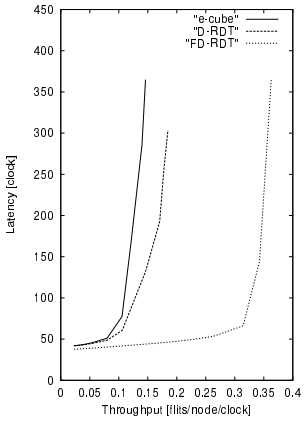
one fixed route for each packet, such packet blocking frequently occurs. Although D-RDT is an adaptive routing, it can select usually only two different paths in each rank. Therefore, it does not work well for bypassing the blocking packets unlike FD-RDT which can select a number of alternative paths.



(a) uniform traffic



(b) bit reversal



(c) matrix transpose

Figure 9: Simulation result (Latency vs. Throughput)

Also with the matrix transpose traffic, adaptive routing shows better performance than the deterministic routing as shown in Figure 9(c). FD-RDT shows much improvement compared with the rest two algorithms. With the matrix transpose traffic, the node (x, y) sends a packet to the node $(k-y-1, k-x-1)$ (k is the node number of each dimension). In this case, two neighboring nodes frequently communicates, and there is no alternative paths between such nodes for e-cube and D-RDT. For FD-RDT, adaptive local routing channel C_R can use for such communication and it improves the performance.

FD-RDT requires four or five virtual channels at each physical link while D-RDT requires three or four. Considering the performance improvement, FD-RDT is suitable for the RDT(2,4,1)/α.

5 Conclusion

Two adaptive routing algorithms on RDT(2,4,1)/α are proposed and proved to be deadlock-free. A simulation study which demonstrates the effect of the proposed routing algorithm is presented. From the simulation result, adaptive routing shows improvement at packet latency in three traffic patterns. Especially, FD-RDT is suitable for the RDT(2,4,1)/α.

References

[1] W. J. Dally A. Chien S. Fiske W. Horwat J. Kenn M. Larivee R. Lethin P. Nuth and S. Wills, The J-machine: A Fine-Grain Concurrent Computer, *IFIP 11th Computer Congress*, pp. 1147-1153, 1989.

[2] H. Nishi, K. Nishimura, K. Anjo, H. Amano and T. Kudoh, The JUMP-1 router chip: The versatile router for supporting distributed shared memory, *Proceedings of IPCCC*, 1996.

[3] Y. Yang H. Amano, Message Transfer Algorithms on the RDT, *IEICE Transaction on Information and Systems*, Vol. 79, 1996.

[4] W. J. Dally C. L. Seitz, Deadlock-Free Message Routing in Multiprocessor Interconnection Networks, *IEEE Transactions on Computers*, Vol. 36, pp. 547-553, 1987.

[5] L. M. Ni P. K. McKinley, A Survey of Wormhole Routing Techniques in Direct Networks, *IEEE Transactions on Computers*, 1993.

[6] J. Duato, A Necessary And Sufficient Condition For Deadlock-Free Adaptive Routing In Wormhole Networks, *IEEE Transaction on Parallel and Distributed Systems*, Vol. 6, 1995.

[7] A. Funahashi T. Hanawa T. Kudoh and H. Amano, Adaptive routing on the Recursive Diagonal Torus, *Proceedings of the 1997 ISHPC*, pp. 171-182, 1997.