Descending Layers Routing: A Deadlock-Free Deterministic Routing using Virtual Channels in System Area Networks with Irregular Topologies

Michihiro Koibuchi Akiya Jouraku Konosuke Watanabe Hideharu Amano

Dept. of Information and Computer Science Keio University
3-14-1 Hiyoshi, Kohoku-ku, Yokohama, 223-8522 Japan
{*koibuchi,jouraku,nosuke,hunga*}*@am.ics.keio.ac.jp*

Abstract

System Area Networks (SANs), which usually accept irregular topologies, have been used to connect nodes in PC/WS clusters or high-performance storage systems. Since wormhole or virtual cut-through transfer is used for low latency communication, deadlock-free routings are essential in SANs. In this paper, we propose a novel deadlock-free deterministic routing called descending layers (DL) routing for SANs. In order to reduce both non-minimal paths and traffic congestion, the network is divided into layers of sub-networks with the same topology using virtual channels, and a large number of paths across multiple sub-networks are established. The DL routing is implemented on a real PC cluster called RHiNET-2, and execution results show that its throughput is improved up to 33% compared with that of up/down* routing. Its execution time of a barrier synchronization is also improved 29% compared with that of up*/down* routing. Simulation results of various sizes and topologies also show that the DL routing achieves up to 266% improvement on throughput compared with up*/down* routing.*

Keywords *Deterministic routing, System Area Networks, deadlock avoidance, RHiNET, irregular topologies, virtual channels, interconnection networks, PC clusters*

1 Introduction

Network-based parallel processing using commodity components, such as personal computers, has received attention as an important parallel-computing environment [16] [21] [5] [17]. System Area Network (SAN), which consists of switches connected with point-to-point links, is one of the crucial components of such an environment. Unlike Local Area Networks (LANs), wormhole or virtual cut-through is used in SANs for low-latency direct-communication. When such methods are used, deadlock-free routings are required. On the other hand, unlike interconnection networks used in massively parallel computers, SANs usually accept irregular topologies, because connection flexibility and robustness are preferred over the uniformity of interconnection networks. The irregularity of interconnection introduces difficulty on guarantee of connectivity and deadlock-free packet transfer.

Thus, spanning tree-based routings[11] [6] [12] [2] which use the connectivity and acyclicity of tree structure have been received attention as practical solutions. Up*/down* routing used in both Autonet[11] and Myrinet[16] is the most popular and fundamental technique in such spanning tree-based routings. It allocates the direction (up or $down$) to each channel, and prohibits packet transfer from the down direction to the up direction in order to guarantee deadlock-free.

Although up*/down* routing is simple and easy to implement, it has two major problems: (1) it must accept a number of non-minimal paths in most cases, and (2) it tends to unbalance network links. To address these problems, improved routing methods, most of which make the use of additional virtual channels or buffers, have been recently proposed[18] [19] [9] [7] [13]. In the background of appearing such routing strategies, switching fabrics in recent SANs[20] [5] [17] have a sufficient amount of hardware for a limited number of virtual channels[3] or extra buffers for packets.

On the contrary, even in recent switching fabrics, deterministic routings are preferred over adaptive routings because of the following advantages: (1) deterministic routings guarantee in-order packet delivery; (2) high-speed switching fabrics can be implemented, because packet control is simple; (3) routing errors of packets can be easily detected.

In this paper, we propose a deadlock-free deterministic routing called descending layers (DL) routing, which can

be applied to networks with a limited number of virtual channels. In order to reduce both non-minimal paths and traffic congestion, the network is divided into layers of sub-networks with the same topology using virtual channels, and a large number of paths across multiple sub-networks are established. It is implemented on a real PC cluster, RHiNET-2 (Real World Computing Partnership High Performance Network)[21], and a prototype with 64 hosts is available at Keio University.

The following sections are organized as follows. In Section 2, up*/down* routing is introduced with its problems. In Section 3, the DL routings is proposed, and implementation and evaluation on the RHiNET-2 cluster are described in Section 4. In Section 5, evaluation results with a flit level simulator are shown. In Section 6, advanced related work is described and compared with the DL routing, and in Section 7, the conclusion is presented.

2 Up*/Down* Routing

Up*/down* routing is based on an assignment of direction to network channels[11]. As a basis of the assignment, a spanning tree whose node (also called vertex) corresponds to a switch in the network is built. A traditional algorithm to build it is the breadth-first search (BFS) used in Autonet[11].

After building the BFS spanning tree, the "up" end of each channel is defined as follows: (1) the end whose node is closer to the root in the spanning tree; (2) the end whose node has the lower unique identifier (UID), if both ends are at the nodes in the same tree level. The restriction on up*/down* routing is simple: a legal path must traverse zero or more channels in the up direction followed by zero or more channels in the down direction. Thus, the up*/down* rule prohibits any packet transfer from the down direction to the up direction.

Since no cycles are formed among paths with the above restriction, it guarantees deadlock-free routing while still allowing all hosts to be reached. However, up*/down* routing uses a number of non-minimal paths which consume more network resources than minimal paths, thus degrades the throughput.

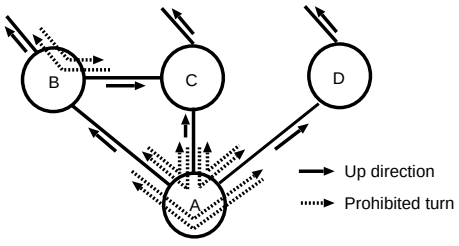


Figure 1. Pairs of prohibited packet turns in up*/down* routing

Furthermore, up*/down* routing tends to cause an unbalanced traffic because each prohibited turn always pairs with the opposite one in two links. For example, as shown in Figure 1, a pair of prohibited turns is formed at switch B and three pairs are formed at switch A. Such prohibited turns lead packets to the root direction, and the heavy traffic around the root causes a congestion which degrades the total throughput.

3 Descending Layers (DL) Routing

In this section, we propose a novel deterministic routing called descending layers (DL) for using virtual channels to increase minimal paths and relax the traffic congestion.

The DL routing is composed of the following steps.

1. Divide the target network into layers of sub-networks with the same topology.
2. Impose conditions to avoid deadlocks.
3. Select deterministic paths.

3.1 Dividing the Network into Sub-networks

The target network is divided into layers of sub-networks with the same topology numbered 0 to $(n-1)$ by using virtual channels, where n is the number of virtual channels per physical channel. The sub-network is a virtual network whose topology is the same as that of the target network.

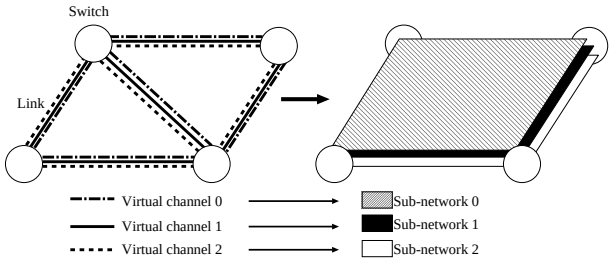


Figure 2. An example of the sub-network structure

Figure 2 shows an example of the layers of sub-networks when using three virtual channels. Since virtual channels with different numbers are assigned into different sub-networks, no virtual channels are shared with different sub-networks.

3.2 Avoiding Deadlocks

First, restrictions of routing in each sub-network are decided enough to satisfy deadlock-free as long as every packet

is routed inside the sub-network. In addition, routing restrictions in the sub-network 0 are decided under the condition to keep the connectivity of the sub-network 0 between each pair of switches.

In order to impose the routing restrictions in each sub-network, various deadlock-free routings[11] [6] [12] [2] or conditions can be used. Here, simple algorithms based on the up*/down* routing and L-turn routing[12] [2] are selected as restrictions in sub-networks.

(UD)* The simplest algorithm—(UD)*— uses up*/down* routing as long as a packet is routed inside a sub-network. Thus, a packet must not be transferred from a down channel to an up channel without switching sub-networks.

(UD-DU)* In an even-numbered sub-network, a packet must not be transferred from a down channel to an up channel without switching sub-networks. On the other hand, in an odd-numbered sub-network, a packet must not be transferred from an up channel to a down channel without switching sub-networks.

UD-(DU)* In the sub-network 0, a packet must not be transferred from a down channel to an up channel like up*/down* routing. On the other hand, in the other sub-networks, a packet must not be transferred from an up channel to a down channel without switching sub-networks.

(L-turn)* (L-turn)* is the same as the (UD)* except that L-turn routing is used for packet transfer without switching sub-networks. L-turn routing is a deadlock-free adaptive routing by identifying and analyzing cycles on a two-dimensional graph, and achieves better performance compared with up*/down* routing[12] [2].

Notice that packet transfer across sub-networks is beyond the scope of the restrictions because the restrictions work for packet transfer only within a sub-network.

Second, in order to prevent deadlocks across sub-networks, packet transfer to a sub-network with a larger number is prohibited.

3.3 Selecting Deterministic Paths

A single path between every pair of switches is searched under the above restrictions in the target network topology. Since the routing algorithm used in the sub-network 0 guarantees the connectivity, there is at least a path between each pair of switches in any topology[1].

[1]From the viewpoint of path establishment, various partitioning of sub-networks that have different topologies are also possible. However, using sub-networks of the same topology is advantageous for easy implementation and path distribution among virtual channels. Thus, in this paper, we focus on the way using only sub-networks of the same topology.

Since non-minimal paths consume larger amount of network resources than minimal paths, minimal paths should be selected as possible. Thus, the only shortest paths among alternative paths are used in the DL routing. Moreover, in order to avoid redundant changes of sub-networks, the path goes across sub-networks only when the output direction doesn't satisfy the restrictions to remain in the sub-network. A procedure of the shortest-path search is as follows, where n is the number of sub-networks, and v is initialized to $(n-1)$.

1. Search paths between each pair of switches under the rules of the DL routing, when the sub-network v is used at the source. Then, paths across all sub-networks are available in the search.
2. Search the paths, of which the same length as the paths in Step 1, under the rules of the DL routing, when the sub-network $(v-1)$ is used at the source.
3. Repeat Step 2 with $v \leftarrow (v-1)$ until no legal paths are found in Step 2 or until the search, that uses the sub-network 0 at the source, is done.
4. Select a single path by a certain policy[6] [14] [2], when multiple shortest paths are found between a pair of switches.

In this paper, such a policy, which chooses a path among alternative paths between each pair of switches, is called "path selection algorithm". A path selection algorithm is commonly required when an adaptive routing is implemented as a deterministic routing. For example, up*/down* routing originally proposed as an adaptive routing is changed into a deterministic routing in Myrinet[16] [6]. A path selection algorithm influences performance, because well-distributed paths can remove the congestion around the hot spot[14].

The simplest path selection algorithm is the random selection. Another simple one selects a path for the port with the smaller port-ID when more than two channels are available in a switch. In this paper, this policy is called "low-port first". However, the above path selection algorithms possibly select a path to congestion points even if there exist alternative paths which can avoid it. To mitigate this problem, Sancho et al. propose the algorithm using a static analysis of routing paths[6]. However, it does not consider on virtual channels, since its target is up*/down* routing for Myrinet without virtual channels. Considering the influence of virtual channels, we proposed four path selection algorithms based on such static analysis of routing paths[14]. Consequently, in order to determine paths, we select one of various path selection algorithms in the DL routing, and its impact is described in Section 5.

[2]Random selection, low port first, or methods using a static analysis of paths have been developed. According to a implementation policy, one of them is used.

3.4 Properties of the DL Routing

Theorem 1 *The DL routing is deadlock-free.*

Proof:

1. No cyclic dependency is formed within each sub-network because a packet must follow the restrictions for deadlock-free as long as transferred in a sub-network.

2. No cyclic dependency is formed across sub-networks because a packet is passed between sub-networks only in the descending order.

Therefore, the DL routing is deadlock-free. ∎

The DL routing has the following advantages compared with deterministic up*/down* routing.

The Length of Paths Up*/down* routing must use a number of non-minimal paths so as not to create cycles among physical channels in most cases. On the other hand, by switching sub-networks, the DL routing breaks cycles among virtual channels and allows some cycles among physical channels. Thus, the DL routing takes shorter paths than up*/down* routing.

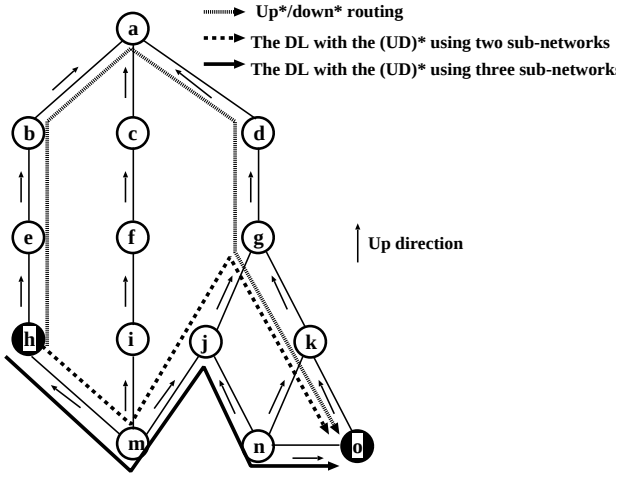


Figure 3. The routing examples of the DL routing with the (UD)* and up*/down* routing

Figure 3 is the example of deterministic routings from **h** to **o**. As shown in Figure 3, up*/down* routing requires seven hops for the packet to reach the destination (**h**→**e**→**b**→**a**→**d**→**g** →**k**→**o**). Whereas the DL routing that uses the (UD)* with two sub-networks handles the same routing in five hops (**h**→**(1)**→**m**→**(0)**→**j**→**(0)**→**g**→**(0)**→**k**→ **(0)**→**o**). The parenthesized number indicates the sub-network in which the packet is being transferred. Moreover with three sub-networks, the path is further reduced to four hops (**h**→**(2)**→**m**→**(1)** →**j**→**(1)**→**n**→**(0)**→**o**). The example shows the effectiveness of the DL routing to use sub-networks for shortening the path.

Traffic Balance In up*/down* routing, the traffic tends to concentrate around the root, since it relies on the connectivity and acyclicity of a spanning tree. On the other hand, the DL routing allows more freedom in path selection. In effect, the traffic is distributed by using an efficient path selection algorithm.

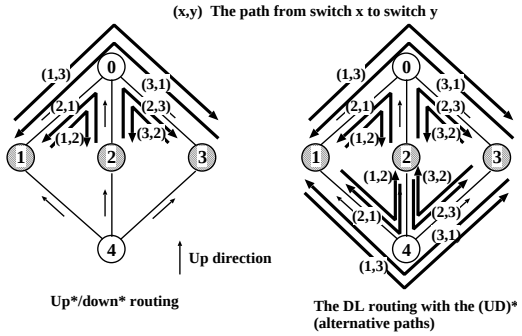


Figure 4. An example of path distribution in the DL routing with the (UD)* and up*/down* routing

Figure 4 compares the traffic distribution of the DL routing that uses the (UD)* with that of up*/down* routing. In Figure 4, bold arrows show the paths for all pairs of switches except the root and the leaf. Figure 4 illustrates that up*/down* routing concentrates the paths around the root. On the other hand, the DL routing distributes the path more uniformly by selecting paths randomly or using the static analysis of routing paths[6] [14]. The example shows the effectiveness of the DL routing to use sub-networks for relaxing the congestion.

4 Performance Evaluation Through the RHiNET-2 Cluster

Although a large of routing methods for SANs have been proposed, a few of them are implemented and evaluated on real systems. However, simulation results may differ from that of real PC clusters, since hosts, and network interfaces used in the simulation are usually simplified for achieving enough simulation speed. In this section, the DL routing and up*/down* routing are implemented and evaluated on a real PC cluster called RHiNET-2[20] [10].

4.1 Execution Environment

4.1.1 The RHiNET-2 Cluster

Real World Computing Partnership (RWCP) carried out a research project called RHiNET[21] for establishing high-end system area networks coordinate with Hitachi Co. Ltd. and Keio University. RHiNET is designed not only for dedicated clusters but also parallel computing environments using personal computers distributed within one or more floors of a building.

The RHiNET-2 cluster with 64 hosts, that is a prototype of such systems shown in Figure 5, consists of hosts with specially designed network interfaces (RHiNET-2/NI) and switches (RHiNET-2/SW) connected with 8Gbit/sec optical interconnects [10]. Table 1 shows specifications of the host consisting of commodity components.

Table 1. Specifications of the host

CPU	Intel Pentium III 933MHz $\times$ 2 (SMP)
Chipset	Serverworks ServerSet III HE-SL
Memory	PC133 SDRAM 1GByte
PCI	64bit/66MHz
OS	RedHat Linux 7.2 (kernel 2.4.18)

RHiNET-2/NI A network interface, RHiNET-2/NI, carries a network controller chip, Martini, 256 MByte SDRAM, O/E and E/O interfaces for 800 MHz optical interconnects. It is put into a common personal computer with 64bit/66MHz PCI bus. Martini is an ASIC chip which manages fundamental zero-copy communications only with a hard-wired logic including complicated processing for address conversion and memory protection. It also provides a core processor compatible to MIPS3000 for exceptional processing. Martini is fabricated by Hitachi Device Development Center using 0.14μm CMOS embedded array technology.

Martini provides two types of communication primitives. One is a remote DMA transfer for high-bandwidth communication: PUSH (remote write)/PULL (remote read). It is initiated when a data item is written into the kick address. The other is a PIO-based transfer for low-latency communication. Since latency of address conversion or DMA setup cannot be ignored for a small-sized DMA transfer, the PIO-based transfer is suitable for sending a small-sized data.

RHiNET-2/SW A network switch, RHiNET-2/SW[20], provides eight input/output ports each of which is connected to an 8Gbit/sec optical link. The core of RHiNET-2/SW is one-chip switching fabric with 0.18μ CMOS embedded array technology. It provides 800 Mbit/sec-per-signal high-speed low-voltage differential signaling (LVDS) I/O. Its aggregate throughput is 64Gbit/sec. Sixteen virtual channels at each port are provided, and each virtual channel has a 4 KByte buffer on a chip so that Go & Stop flow control supports 200 m link length. In the RHiNET-2 cluster, the optical interconnection modules have the order of 10^{-20} at bit-error-rate (BER). In addition, the error detection and correction are done with ECC attached to each flit. Thus, a reliable communication with 200 m cable is guaranteed in the RHiNET-2 cluster. The detail of the RHiNET-2 cluster and its performance are shown in [10].

4.1.2 Routing Implementation

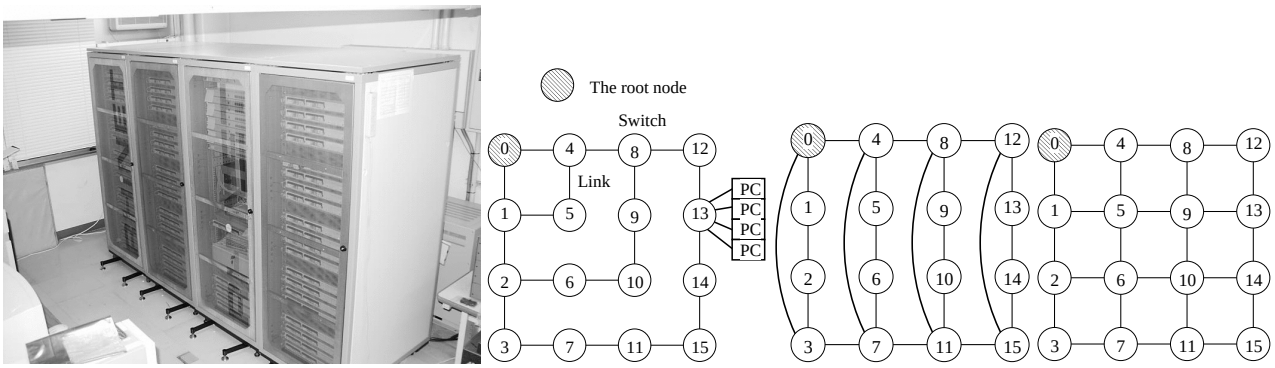
RHiNET-2/SW originally supports only a simple deadlock-free deterministic routing called the modified structured channel[20], which is one of algorithms based on structured buffer pools[15] [8]. In this method, a packet stored in the virtual network i is transferred to the virtual network $(i+1)$ when the switch provides branch links.

RHiNET-2/SW uses table look-up manner, in which a packet finds its output channel by referring a routing table provided at each switch. At table reference, only destination tag is indexed to get output port (physical channel). On the other hand, the difference between a number of output and input virtual channels indexes with both input port and output port. As mentioned before, sixteen virtual channels are provided in RHiNET-2/SW. Eight of them numbered from 0 to 7 are used for data-transfer packets, while the rest numbered from 8 to 15 are used for system control packets. Both types of channels use the same routing table.

Since the routing mechanism in RHiNET-2/SW is simple, it can accepts various topologies and routing algorithms by rewriting the routing table. Each switch is equipped with a simple control processor, and the data of routing table is loaded from the processor at initialization. Since the RHiNET-2 cluster uses optical cables, it is possible to change its connection topology manually, and we tried three topologies shown in Figure 5. Four ports of each switch are connected with hosts, and remaining ports are connected to other switches if required.

Here, the DL routing with the (UD)* restriction, that uses the BFS spanning tree with the root 0, is implemented using two virtual channels [3] shown in Figure 5. Then, it takes minimal paths when there are two virtual channels in three topologies shown in Figure 5. For both distribution of paths and easy packet tracing, we use a path selection algorithm that allocates different paths to different hosts when multiple paths are found between a pair of switches. The influence of path selection algorithms is minutely evaluated with the simulation and shown in Section 5. For the comparison, a deterministic up*/down* routing with the same

[3] "the number of virtual channels" in this evaluation represents that the number of virtual channels which are used by packets for data transmission. Notice that packets for system control will also use the same number of virtual channels.



(a) The RHiNET-2 cluster with 64 hosts (b) Topology A (c) Topology B (d) Mesh

Figure 5. The RHiNET-2 cluster and its three switches topologies considered in executions

path selection algorithm is also implemented.

In Topology A and B, up*/down* routing must accept some non-minimal paths, while the DL routing takes minimal paths. On the other hand, both take minimal paths in the mesh topology. Other complicated topologies and larger network sizes are evaluated with a simulation and shown in Section 5.

4.1.3 Measures

We use the following measures for evaluations: latency of barrier synchronization on 64 hosts, and the average bandwidth of each routing algorithm, which are crucial primitives for supporting parallel processing. The former is the average execution time of NIC-based barrier synchronization with all hosts using PIO-based communication on 100,000 trials. The latter is the average bandwidth of each routing algorithm under the condition that each host injects repeatedly 2KByte-data packets using PUSH primitive as short interval as possible. Thus, the bandwidth of each routing algorithm is equal to its throughput. The typical traffic patterns — *bit reversal*, *matrix transpose*, *butterfly*, and *complement*[8]— are used.

4.2 Execution Results

Table 2. Latency of barrier synchronization (μsec)

	Topology A	Topology B	The Mesh
Up*/Down*	52.32	44.42	39.42
DL	40.50	40.12	39.38

Table 2 shows the average execution time of barrier synchronization of two routing algorithms. In the synchronization, the PIO-based communication generates a large number of small packets that cause a network congestion. As shown in Table 2, the DL routing achieves up to 29% improvement on the average execution time on Topology A and B, in which the DL routing and up*/down* routing take different path hops. On the other hand, on the mesh in which both take minimal paths, the execution time of DL routing is close to that of up*/down* routing. Therefore, reducing non-minimal paths, that consume larger network resources, is a main factor of the latency improvement of the DL routing.

Table 3 shows bandwidth of two routing algorithms under the typical traffic patterns. When data size is 2 KByte, the RHiNET-2 cluster provides 206.55 MByte maximum bandwidth between two hosts without congestion[10]. As shown in Table 3, the DL routing achieves up to 137.26 MByte, that is smaller than the above bandwidth because of the packet congestion. Nevertheless, the DL routing improves up to 33% on bandwidth compared with up*/down* routing. Thus, the DL routing is also efficient for increasing bandwidth in the RHiNET-2 cluster.

5 Performance Evaluation Through the Flit Level Simulator

In order to analyze the impact of the DL routing in complicated topologies and large network sizes, the DL routing and up*/down* routing are evaluated on a flit-level simulation.

5.1 Simulation Environment

5.1.1 Parameter

A flit-level simulator written in C++ was developed for analysis. Topology, network size, and packet length are selected just by changing parameters. Like real switches with eight ports, such as, RHiNET-2/SW[20] and Myrinet M3F-SW8/M3F-SW8M[1], a switching fabric is assumed to provide eight bi-directional ports. Four ports are used

Table 3. Bandwidth of Routing Algorithms(MByte)

	Topology A				Topology B			The Mesh			
	bit rev.	matrix.	butfly	comp.	bit rev.	matrix.	butfly	bit rev.	matrix.	butfly	comp.
Up*/Down*	95.12	101.99	90.15	87.69	91.98	110.49	86.28	130.51	138.09	127.43	112.29
DL	118.38	121.14	120.26	104.73	107.56	110.50	108.30	135.28	137.26	132.14	121.81

for hosts and remaining ports are connected to other switches. Here, a simple model consisting of channel buffers, crossbar, link controller and control circuits is used for the switching fabric. Two classes of network topologies, irregular and regular, are used as follows. As irregular topologies, ten different ones are randomly generated under the condition that only one link is connected between two different switches. Two network sizes, small (16 switches) and large (64 switches), are used in irregular topologies. On the other hand, 64 switches two-dimensional torus is used as a regular topology. A destination of a packet is determined by the traffic patterns, *uniform* or *bit-reversal*[8], and hosts inject a packet independently of each other.

A header flit transfer requires at least three clocks, that is, one for routing, one for transferring a flit from an input channel to output channel through a crossbar, and the rest for transferring the flit to the next switch or host. The model is simple compared with the operation in RHiNET-2/SW. Nevertheless, it is useful for larger systems and complicated topologies because more exact modeling of modern switching fabrics consumes a huge simulation time.

Other parameters are set as shown in Table 4.

Table 4. Simulation parameters

Simulation time	1,000,000 clocks (ignore the first 50,000 clocks)
The number of vchs	3
Packet length	128 flits
Switching tech.	Virtual cut-through

5.1.2 Deterministic Routings

The DL routing uses the four types of restrictions ((UD)*, (UD-DU)*, DU-(UD)*, and (L-turn)*) to investigate the impact of deadlock-free algorithms in each sub-network. When multiple paths between a pair of switches are found in the DL routing or up*/down* routing, a path selection algorithm—low-port first, Sancho's algorithm, or high physical-channel first[14]— determines one.

Advanced deterministic routings [19] [9] [7] are not included in our evaluation, since they may require buffer at hosts, more than three virtual channels to complete implementation on any irregular topology, or different network architecture. They will be qualitatively compared in Section 6. Adaptive routings are also out of our focus, since targets of the DL routing are networks like RHiNET[20] which do not support adaptive routings.

In the simulation, up*/down* routing, which originally does not require virtual channels, uses the same number of virtual channels as the DL routing as follows. At intermediate switches, up*/down* routing doesn't change virtual channels. On the other hand, at the source, alternative paths with different virtual channels are available, and each path selection algorithm selects a single path at initialization.

Both the DL routing and up*/down* routing need to build a spanning tree to assign the up or down direction to each network channel. Here, we use the Autonet algorithm[11] to build the spanning tree. In the Autonet algorithm, the switch with unique identifier (UID) $zero$ is chosen as the root and the order of the BFS is used to add links into a tree[11].

5.2 Simulation Results

5.2.1 Preliminaries

In order to compare simulation results with execution results in the RHiNET-2 cluster, we preliminarily evaluate the DL routing and up*/down* routing, that use the same set–the BFS spanning tree, two virtual channels, and path distribution— as ones in Section 4, on Topology A, B, and the mesh shown in Figure 5. Table 3 and 5 show that the DL routing improves the throughput/bandwidth compared with up*/down* routing.

As shown in Table 3 and 5, the DL routing in the simulation achieves larger improvement than that in the RHiNET-2 cluster. The packet interval of the RHiNET-2 cluster includes the processing time (e.g. the DMA transfer to memory and the generation of an acknowledge packet) at each host. Thus, we consider that the effect of routing algorithms is enhanced compared with one in the RHiNET-2 cluster. The absolute values are different from those in Table 3, however, the tendency of simulation results is similar to the execution results in the RHiNET-2 cluster.

5.2.2 Irregular Topologies

Table 6 shows the average throughput in 10 different irregular topologies, and its standard deviation (SD). Throughput, that is the maximum amount of accepted

Table 5. Throughput of Routing Algorithms in Topology A, B and the Mesh (flits/host/clock)

	Topology A				Topology B			The Mesh			
	bit rev.	matrix.	butfly	comp.	bit rev.	matrix.	butfly	bit rev.	matrix.	butfly	comp.
Up*/Down*	0.056	0.062	0.039	0.031	0.075	0.062	0.046	0.124	0.080	0.085	0.049
DL	0.103	0.080	0.085	0.065	0.096	0.087	0.053	0.139	0.102	0.089	0.073

Table 6. Throughput on SANs with irregular topologies (flits/host/clock)

	16 switches				64 switches			
	Uniform		Bit reversal		Uniform		Bit reversal	
	Ave.	SD	Ave.	SD	Ave.	SD	Ave.	SD
Up*/Down* (low port)	0.161	0.032	0.269	0.048	0.033	0.004	0.052	0.010
Up*/Down* (Sancho's one)	0.177	0.033	0.322	0.056	0.037	0.005	0.059	0.013
Up*/Down* (high p-ch first)	0.176	0.032	0.328	0.053	0.037	0.005	0.062	0.013
DL ((UD)*, low port)	0.169	0.008	0.277	0.052	0.105	0.002	0.123	0.012
DL ((UD)*, Sancho's one)	0.289	0.022	0.353	0.038	0.162	0.004	0.192	0.010
DL ((UD)*, high p-ch first)	0.290	0.024	0.350	0.041	0.164	0.006	0.188	0.010
DL ((UD-DU)*, low port)	0.169	0.008	0.277	0.052	0.105	0.002	0.120	0.012
DL ((UD-DU)*, Sancho's one)	0.286	0.023	0.354	0.052	0.159	0.004	0.199	0.010
DL ((UD-DU)*, high p-ch first)	0.282	0.020	0.360	0.049	0.158	0.006	0.193	0.006
DL (UD-(DU)*, low port)	0.180	0.016	0.274	0.050	0.109	0.004	0.124	0.010
DL (UD-(DU)*, Sancho's one)	0.295	0.022	0.349	0.041	0.169	0.006	0.196	0.008
DL (UD-(DU)*, high p-ch first)	0.289	0.025	0.358	0.048	0.170	0.007	0.192	0.012
DL ((L-turn)*, low port)	0.181	0.009	0.284	0.049	0.103	0.003	0.112	0.010
DL ((L-turn)*, Sancho's one)	0.292	0.020	0.347	0.044	0.156	0.007	0.186	0.005
DL ((L-turn)*, high p-ch first)	0.283	0.019	0.338	0.042	0.159	0.009	0.185	0.009

traffic[8], is the most important metric of routing algorithm in SANs. In Table 6, "Up*/Down*(Sancho's one)" represents up*/down* routing with the Sancho's algorithm, and "DL ((UD)*, high p-ch first)" represents the DL routing with the (UD)* that selects a single path between each pair of switches using the policy of high physical-channel first. Table 7 shows the analysis result of the average path hops.

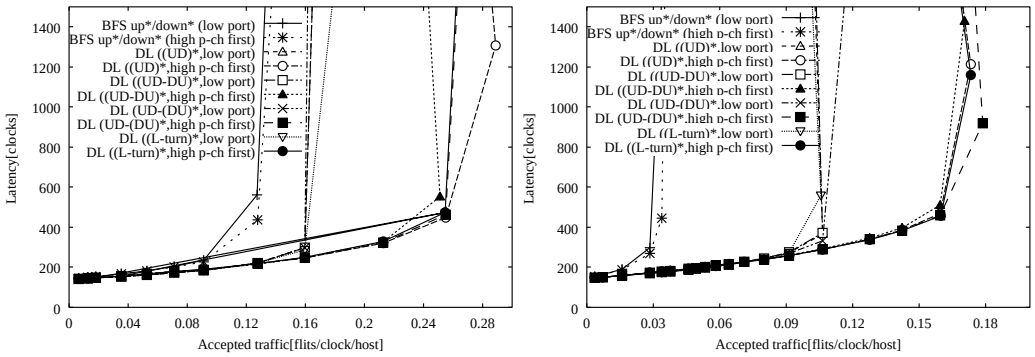
Table 7. The average hops of packets

	16 switches		64 switches		2D torus	
	Uni.	Bit.	Uni.	Bit.	Uni.	Bit.
Up*/Down*	2.01	1.98	3.72	3.48	4.36	3.80
DL,(UD)*	1.89	1.94	3.14	3.14	4.02	3.50
,(UD-DU)*	1.89	1.94	3.15	3.13	4.02	3.50
,UD-(DU)*	1.89	1.93	3.14	3.13	4.02	3.50
,(L-turn)*	1.88	1.94	3.14	3.13	4.02	3.50

Table 6 shows that each DL routing achieves higher throughput than up*/down* routing with the same path selection algorithm, because it reduces the average hops of packets compared with up*/down* routing as shown in Table 7. Notice that, as shown in Table 7, the average hops of packets are not related to the path selection algorithm because it selects a single path among alternative paths which have the same hops. In particular, the DL routing with Sancho's algorithm or high physical-channel first increases its advantage on throughput. Since each DL routing without a path selection algorithm has the larger number of alternative paths between each pair of switches than up*/down* routing, Sancho's algorithm and high physical-channel first, which consider the path balance, distribute paths more uniformly in the DL routing. Table 6 also shows that each DL routing has the highest stability of throughput on irregular topologies since the SD of its throughput is small. As shown in Table 6, we can see that the deadlock-free algorithms in each sub-network give the small impact in the DL routing.

Table 6 shows that each routing algorithm in bit reversal traffic achieves higher throughput than one in uniform traffic. This comes from that, in uniform traffic, packets whose source hosts are different have possibility to collide at a consumption channel on the destination host. Such collisions drastically degrade the performance especially in smaller network sizes. On the other hand, in bit reversal traffic, such collisions on the destination host are not occurred.

Figure 6 shows the latency on irregular topologies, in which the improvement ratio of the DL routing with high physical-channel first against up*/down* routing with one is the average shown in Table 6. Latency is the second important metric of routing algorithm in SANs. As shown in Figure 6, every DL routing decreases the latency. This is because each DL routing decreases the packet hops and relaxes packet congestion by distributing paths. Consequently, each DL routing improves both throughput and latency in SANs



(a) 16 switches (d) 64 switches

Figure 6. Accepted traffic versus latency on typical irregular topologies (uniform traffic)

with irregular topologies.

5.2.3 Two-Dimensional Torus (8 ×8)

Figure 7 shows the relation between the average latency and the accepted traffic of ten algorithms on 8×8 two-dimensional torus. As shown in Figure 7, each DL routing achieves up to 266% improvement on throughput compared with up*/down* routing with the same path selection algorithm, and it also reduces the latency. Table 7 shows the analysis result of the average path hops on two-dimensional torus. Like irregular topologies, each DL routing has the smallest value of path hops, which influences the throughput.

The evaluation on regular topologies is important, since topologies of most SANs are not completely irregular but have some regularity in practice. At two-dimensional torus, the performance of up*/down* routing is quite poor, while each DL routing achieves high performance as well as at most cases of irregular topologies.

6 Related Work

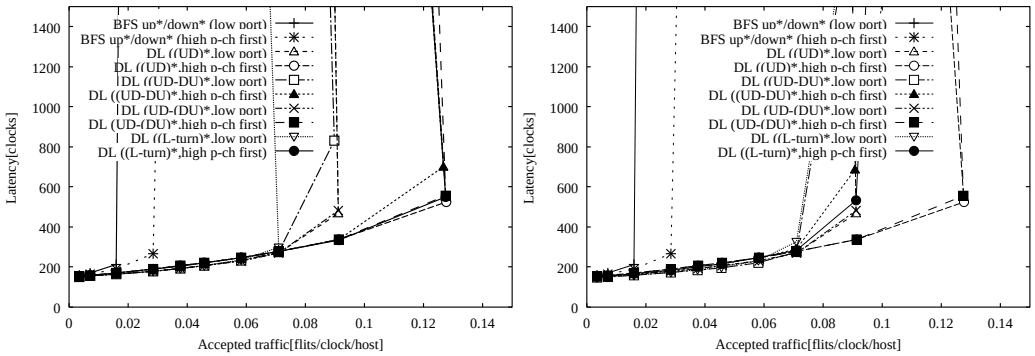
As mentioned in Section 2, the common problems with up*/down* routing are that (1) it must accepts a number of non-minimal paths in most cases, and (2) it tends to unbalance network links. To aggregate the problems, different approaches have been investigated as well as the DL routing.

Layered shortest path (LASH) routing proposed by Skeie et al. guarantees minimal paths by dividing the physical network into a set of virtual layers. The virtual layer is a virtual network like the sub-network in the DL routing. The LASH routing is safe from deadlocks by making acyclic virtual layers[19], however, it needs the number of virtual channels enough to guarantee both minimal paths and deadlock-free. A minimal routing proposed by Sancho et al.[7] for InfiniBand is similar to the LASH routing. It adopts up*/down* routing to make acyclic virtual networks (layers). On the other hand, an adaptive escape-path routing proposed by Silla and Duato doesn't always guarantee minimal paths, however, it guarantees deadlock-free routing while still allowing cycles[18]. Since each packet in channels out of escape path is forwarded along a minimal path, most of packets take minimal paths in the Silla's routing. It is difficult to be implemented as a deterministic routing using a path selection algorithm. This is because it guarantees deadlock-free through dynamically selecting a path between the original channel (escape path) and the new channel(fully adaptive path). The other approach that uses buffers at intermediate hosts is proposed. A true minimal routing proposed by Flich et al. is intended to be a source routing in Myrinet[9]. The true minimal routing breaks all cycles by storing and later re-injecting packets at some intermediate hosts.

Consequently, they have different conditions — adaptive/deterministic routings, the required number of virtual channels, and the use of buffers on intermediate hosts— to apply in SANs, that is, their target networks are different.

7 Conclusion

A novel deadlock-free deterministic routing called descending layers (DL) is proposed for SANs with irregular topologies and implemented on the real PC cluster called RHiNET-2. It divides the network into sub-networks with the same topology consisting of layers of virtual channels, and it establishes a large number of paths across sub-networks in order to reduce the path length and path congestion. Through the evaluation of the RHiNET-2 cluster, its throughput is improved up to 33% compared with that of up*/down* routing. Its execution time of a barrier synchronization is also improved 29% compared with that of up*/down* routing. The performance of the DL routing in larger network sizes and various topologies are evaluated with the flit level simulation, and it achieves up to 266% improvement on throughput compared with up*/down* routing. Simulation results also show that, the choice of a path



(a) Uniform traffic (b) Bit reversal traffic

Figure 7. Accepted traffic versus latency on 8×8 2D torus

selection algorithm has a large influence on the DL routing, while deadlock-free algorithms to remain a packet in the sub-network have small impact.

We are planning to evaluate the DL routing through various benchmarks on the RHiNET-2 cluster with SCore system software[4].

8 Acknowledgments

The authors thank Dr. Hiroaki Nishi, at Hitachi Co. Ltd., Central Research Laboratory, for his comments on the RHiNET-2/SW environment.

References

[1] http://www.myri.com/.

[2] A.Jouraku, M.Koibuchi, A.Jouraku, and H.Amano. Routing Algorithms Based on 2D Turn Model for Irregular Networks. In *Proc. of I-SPAN*, pages 289–294, June 2002.

[3] W. J. Dally. Virtual-channel flow control. *IEEE Transaction on Parallel and Distributed Systems*, 3(2):194–205, 1992.

[4] Y. Ishikawa, H. Tezuka, A. Hori, S.Sumimoto, T. Takahashi, F. O'Carroll, and H. Harada. RWC PC Cluster II and SCore Cluster System Software – High Performance Linux Cluster. In *5th Annual Linux Expo*, pages 55–62, May 1999.

[5] I.T.Association. Infiniband architecture. specification volumen 1,release 1.0.a. *available from the InfiniBand Trade Association, http://www.infinibandta.com*, June 2001.

[6] J.C.Sancho and A.Robles. Improving the Up*/Down* Routing Scheme for Networks of Workstations. In *Proc. of the European Conference on Parallel Computing*, pages 882–889, Aug. 2000.

[7] J.C.Sancho, A.Robles, J.Flich, , P.Lopez, and J.Duato. Effective methodology for deadlock-free minimal routing in infiniband. In *Proc. of ICPP*, pages 409–418, Aug. 2002.

[8] J.Duato, S.Yalamanchili, and L.Ni. *Interconnection Networks: an engineering approach*. Morgan Kaufmann, 2002.

[9] J.Flich, P.Lopez, M.P.Malumbres, and J.Duato. Boosting the Performance of Myrinet Networks. *IEEE Trans. on Parallel and Distributed Systems*, 13(7):693–709, July 2002.

[10] K. Watanabe and et al. Performance Evaluation of RHiNET-2/NI: A Network Interface for Distributed Parallel Computing Systems. In *Proc. of International Symposium on Cluster Computing and the Grid*, May 2003.

[11] M.D.Schroeder and al et. Autonet: a high-speed, self-configuring local area network using point-to-point links. *IEEE Journal on Selected Areas in Communications*, 9:1318–1335, 1991.

[12] M.Koibuchi, A.Funahashi, A.Jouraku, and H.Amano. L-turn routing: An adaptive routing in irregular networks. In *Proc. of ICPP*, pages 374–383, Sept. 2001.

[13] M.Koibuchi, A.Jouraku, and H.Amano. Deterministic routing techniques by dividing into sub-networks in irregular networks. In *Proc. of the IASTED NPDPA*, pages 143–148, Oct. 2002.

[14] M.Koibuchi, A.Jouraku, and H.Amano. The impact of path selection algorithm of adaptive routing for implementing deterministic routing. In *Proc. of PDPTA*, June 2002.

[15] M.P.Merlin and J.P.Schweitzer. Deadlock Avoidance in Store-and-Forward Networks. *IEEE Trans. Comput.*, COM-28(3):345–354, 1980.

[16] N.J.Boden and et al. Myrinet: A Gigabit-per-Second Local Area Network. *IEEE Micro*, 15(1):29–35, 1995.

[17] F. Petrini, W. Feng, and A. Hoisie. The Quadrics network (QsNet): high-performance clustering technology. In *Proc. of Hot Interconnects*, pages 125–130, Aug. 2001.

[18] F. Silla and J. Duato. High-Performance Routing in Networks of Workstations with Irregular Topology. *IEEE Trans. on parallel and distributed systems*, 11(7):699–719, 2000.

[19] T. Skeie, O. Lysne, and I. Theiss. Layered Shortest Path (LASH) Routing in Irregular System Area Networks. In *Proc. of IPDPS*, pages 162–169, Apr. 2002.

[20] S.Nishimura, T.Kudoh, H.Nishi, J.Yamamoto, K.Harasawa, N.Matsudaira, S.Akutsu, K.Tasho, and H.Amano. High-speed network switch RHiNET-2/SW and its implementation with optical interconnections. In *Hot Intercoonect 8*, pages 31–38, Aug. 2000.

[21] T.Kudoh, S.Nishimura, J.Yamamoto, H.Nishi, O.Tatebe, and H.Amano. RHiNET: A network for high performance parallel computing using locally distributed computing. In *Proc. of IWIA*, pages 69–73, Nov. 1999.