JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 25, 465-479 (2009)

Deadlock Detection and Recovery for True Fully Adaptive Routing in Regular Wormhole Networks

SOOJUNG LEE
Department of Computer Education
Gyeongin National University of Education
Anyang, Kyunggi-do, 430-804 Korea

Deadlock detection and recovery-based routing schemes for wormhole networks have gained attraction because unlike deadlock avoidance-based schemes, they do not restrict routing adaptability. In order to alleviate the overhead of running a recovery procedure, the studies on deadlock detection have focused on the accuracy of deadlock detection, trying to reduce the number of false detections. This paper proposes both deadlock detection and recovery schemes. The proposed detection scheme is based on the turn model and designed to declare only one packet per simple cycle of blocked packets as deadlocked. Our recovery scheme adjusts the time-out value flexibly according to the utilization rate of the recovery resources, rather than fixing a single time-out value as in previous schemes. As a consequence, it not only prevents saturation of the recovery resources by deadlocked packets but also reduces congestion of normal buffers at heavy loads. Simulation experiments show that the proposed deadlock detection scheme significantly reduces the number of false deadlock detections over previous schemes for low to moderate time-out thresholds. It is also found that the proposed recovery scheme prevents overloading of the recovery resources, yielding better network performance.

Keywords: multicomputer, wormhole routing, deadlock detection, deadlock recovery, adaptive routing

1. INTRODUCTION

Direct communication in multicomputer networks has been used for executing tasks in parallel to achieve better throughput. Wormhole routing has been preferred in multicomputer networks, since it requires smaller buffer requirements and the message latency is less sensitive to the distance from the source and destination, leading to lower message latency for the network with little contention [9]. All the algorithms discussed in this paper focus on wormhole routing. In wormhole routing, a packet is split into several *flits* for transmission. A header flit leads the route and the remaining flits follow in a pipelined fashion. Since a router is provided with a few flit buffers only, a packet may reside in several routers simultaneously. Therefore, wormhole routing is susceptible to deadlock, where each packet in a set of packets requests a channel resource held by another packet in the set in a circular way.

Deadlock avoidance has been a traditional approach in handling deadlock problem, where routing is restricted so that it could fundamentally prevent the occurrence of cyclic dependency between channels. For example, the turn model [4, 16] prohibits turns that may form a cycle. However, such design of routing algorithm results in low adaptivity and increased packet latency. A deadlock avoidance scheme by Duato [3] exploits virtual

Received June 7, 2007; revised May 14, 2008; accepted July 3, 2008.
Communicated by Chung-Ta King.

channels which are divided into two classes; one, referred to as the escape channel, is used for a routing free from cyclic dependencies and the other for fully adaptive minimal routing allowing cyclic dependencies among packets. A packet proceeds on the fully adaptive virtual channels until it blocks, where it moves onto the escape channels.

As reported in [11], deadlock rarely occurs if sufficient routing freedom and multiple virtual channels are provided, so it is wasteful to limit adaptivity, as in deadlock avoidance approaches. This motivated a new approach, deadlock detection and recovery [5, 9]. In contrast to deadlock avoidance schemes, this approach provides fully adaptive routing by its nature. However, current deadlock detection schemes cannot distinguish between real deadlocked and simply blocked packets, yielding many false deadlock detections. Another problem is that in principle all the packets involved in cyclic dependency can be presumed as deadlocked, although it is sufficient to select only one for recovery. This might lead to unignorable recovery overhead. The fundamental difficulty of deadlock detection and recovery schemes lies in determining the time-out value; there is no single time-out value that satisfies various network conditions.

This paper proposes a simple but efficient idea whose objective is to select only one packet in a simple cycle of blocked packets in most cases under a fully adaptive routing. The basic idea is that only the packets making certain turns are eligible for checking for the existence of any potential deadlock. It is found from simulation experiments that the proposed strategy effectively reduces the number of deadlock detections compared to previous schemes, which implies that it also reduces the number of false deadlock detections, since most detected deadlocks are in fact non-existent.

Deadlock recovery schemes are in general classified into two groups, regressive and progressive schemes [8, 10]. In order to resolve deadlock, regressive schemes simply kill deadlocked packets and re-inject them into the network after some delay. On the other hand, progressive recovery schemes do not kill but allow deadlocked packets to keep progressing toward their destinations [15]. One of the progressive recovery mechanisms, named *Disha* [10], provides buffers other than normal flit buffers, named *deadlock buffers*, which are centralized at routers. Those buffers form a dedicated recovery path for deadlocked packets to follow by preempting network bandwidth from undeadlocked packets when necessary. Another progressive technique proposed in [8] absorbs deadlocked packets at the current node and later reinjects them.

In order to minimize performance degradation, it is recommended not to saturate recovery resources such as the centralized buffers suggested in [10]. So, lower deadlock detection rates are preferable. However, we found through simulation experiments that lower deadlock detection rates do not always lead to better network throughput at saturated loads, when recovery resources are provided as in [10]. We observed that utilization of the recovery resources to some extent leads to better performance in a heavily-loaded situation, since it directs blocked packets to those resources, relieving congestion of normal flit buffers. Therefore, we propose a deadlock recovery scheme that flexibly adjusts the time-out according to the utilization rate of the recovery resources, rather than fixing a single value as in previous schemes. The scheme is proved through simulation to effectively relieve congestion of normal flit buffers as well as saturation of the recovery resources by reflecting the network status on the time-out value.

The rest of this paper is organized as follows. Section 2 introduces the proposed deadlock detection and recovery schemes. In section 3, performance of the proposed

schemes is evaluated through simulation experiments and compared with that of previous schemes. Section 4 concludes this paper.

2. THE PROPOSED DEADLOCK DETECTION AND RECOVERY MECHANISMS

2.1 Deadlock Detection

The proposed scheme detects deadlock for n-dimensional direct regular networks with true fully adaptive minimal routing that have directions associated with the channels. The basic idea comes from the observation that deadlock involves packets waiting on each other cyclically. This property is also exploited by previous methods where certain turns are prohibited to avoid deadlock [4, 16]. Regarding deadlock detection, this implies that it is enough to examine only those packets making certain turns for potential deadlock. Inactivity time of the channels requested by those packets is measured to check for potential deadlock. By restricting the eligibility of packets to be examined for deadlock, a significant reduction of the number of false deadlock detections can be achieved.

Definition 1 Let REQ_P be the set of all the next feasible output channels requested by a packet P according to the routing algorithm. A feasible output channel is one that leads a packet closer to its destination. Let OCC_P be the set of all channels occupied by a packet P. A blocked packet P is *dependent* on a packet Q if $REQ_P \cap OCC_Q \neq \varnothing$. ❑

Definition 2 A blocked packet P is said to be *transitively dependent* on a packet Q if either of the following conditions is true:
(i) there exists a packet R such that P is dependent on R and R is dependent on Q.
(ii) there exists a packet R such that P is dependent on R and R is transitively dependent on Q. ❑

A packet is either advancing or blocked. The latter is further classified as follows.

Definition 3 A packet P is *temporarily blocked* if
(i) all of its next feasible output channels are occupied by some other packets and
(ii) at least one of the packets on which P is dependent is advancing. ❑

Definition 4 A packet P is *completely blocked* if
(i) all of its next feasible output channels are occupied by some other packets and
(ii) none of the packets on which P is dependent is advancing. ❑

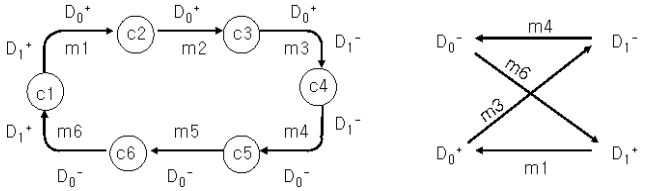
Note that a deadlocked packet is sure to be completely blocked but the opposite may not be the case.

2.1.1 *n*-dimensional meshes

To identify the types of turns in cycles of blocked packets, we focus on directions of packet movement. For *nD* meshes, the following notations are used.

Notation 1 Let D_i denote dimension i for $i = 0, \ldots, n - 1$. Then directions D_i^- and D_i^+ represent the negative and the positive directions along dimension i, respectively. ❑

Fig. 1 illustrates a cycle of blocked packets changing their directions in 2D meshes. A cycle is pictured in two different representations. Fig. 1 (a) shows a channel dependence graph [12] at a particular point of time, with arcs labeled with packet IDs depicting packet progressions or blocking relation and vertices representing physical channels. Fig. 1 (b) depicts only turns made by packets, so named *turn-based graph*. A vertex in the turn-based graph represents a direction and a directed edge ($d1$, $d2$) indicates that some packet is turning or waiting to turn from direction $d1$ to direction $d2$; the edge is labeled with that packet. In the figure, $m1$, holding channel c_1, is waiting to change its direction from D_1^+ to D_0^+ and is dependent on $m2$. In the turn-based graph, $m2$ and $m5$ are not shown, since they are not involved in any turns.



(a) A channel dependence graph. (b) A turn-based graph.

Fig. 1. An illustration of a cycle in 2D meshes with two different representations.

In general, a blocked packet in the turn-based graph of a cycle may be transitively dependent on a packet ahead of it in the graph. Note that there is no one-to-one correspondence between the two types of graphs in Fig. 1, since the latter focuses on packet turns only. Hence, a turn-based graph may represent several different network states. It is observed that the turn-based graph representation of a cycle in nD meshes has the following properties.

P1. Since 180-degree turns are not allowed in our assumption, (D_i^-, D_i^+) and (D_i^+, D_i^-) are not included in the graph.

P2. The graph includes a cycle such that

P2-1. the cycle includes at least four vertices and their associated edges and

P2-2. if the cycle includes vertex D_i^-, it also includes vertex D_i^+, and vice versa.

Now we need to select common types of turns that are included in every cycle, in order to detect deadlock. In [4], it is said that prohibition of $n(n - 1)$ turns is sufficient to prevent any deadlock in nD meshes. However, care must be taken so that the selected turns cover all possible cycles in nD meshes. In our scheme, turns from D_i^+ to D_{i+1}^-, D_{i+1}^+, …, D_{n-1}^-, and to D_{n-1}^+, for $i = 0, \ldots, n - 2$, are selected. Note that the total number of turns selected as such is $n(n - 1)$.

Theorem 1 Any cycle in nD meshes includes at least one of the turns from D_i^+ to D_{i+1}^-, D_{i+1}^+, …, D_{n-1}^-, and to D_{n-1}^+, for $i = 0, \ldots, n - 2$.

Proof: By contradiction. Consider a *m*-dimensional cycle, for $m = 2, \ldots, n$, which excludes all of the specified turns. Fig. 2 illustrates the turn-based graph representing the cycle. The *m* dimensions are renamed and listed in an ascending order, $i_1, i_2, \ldots, i_m$. As seen in the figure, the graph contains no cycle of *m* dimensions and it violates Property P2-2. ❑

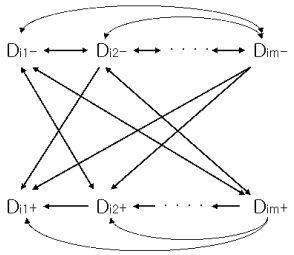


Fig. 2. A turn-based graph including all edges except those corresponding to the selected turns in *nD* meshes.

To describe the proposed deadlock detection scheme in detail, a bit of a packet header, named *turn bit*, is allocated to indicate whether the next routing direction of a packet corresponds to one of the selected turns. If the turn bit of a packet is set and all of its next feasible output channels have been inactive for time-out, the packet is presumed as deadlocked. Note that the turn bit of a packet may be set at a router but the packet may be presumed as deadlocked at another router on its way to the destination.

Deadlock Detection Scheme for *nD* meshes:

1. If the next routing direction of a packet corresponds to one of the turns from D_i^+ to D_{i+1}^-, D_{i+1}^+, …, D_{n-1}^-, and to D_{n-1}^+, for $i = 0, \ldots, n-2$, set its turn bit to true.
2. If a packet has been completely blocked for the time-out interval and its turn bit is set true, the packet is presumed as deadlocked. ❑

2.1.2 *k*-ary *n*-cubes

For *k*-ary *n*-cubes, in addition to cycles in meshes, there can be a cycle involving wraparound channels. This cycle may not include the selected turns mentioned previously. A simple modification is made to the scheme to detect such deadlock; when the next routing direction of a packet uses a wraparound channel, set its turn bit true. This strategy is more than enough to detect all deadlocks involving wraparound channels. More sophisticated method can be devised to reduce the number of turn bit settings. However, we choose simplicity rather than complexity not to degrade the router performance.

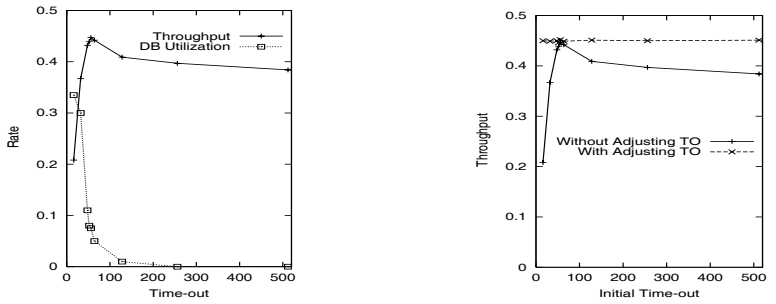
Deadlock Detection Scheme for *k*-ary *n*-cubes:

1. If the next routing direction of a packet corresponds to one of the turns from D_i^+ to D_{i+1}^-, D_{i+1}^+, …, D_{n-1}^-, and to D_{n-1}^+, for $i = 0, \ldots, n-2$, set its turn bit to true.
2. Otherwise if the next routing direction of a packet uses a wraparound channel, set its turn bit to true.

3. If a packet has been completely blocked for the time-out interval and its turn bit is set true, the packet is presumed as deadlocked. ❑

2.2 Deadlock Recovery

A time-out value is an important factor affecting the performance of a deadlock handling scheme. Especially for Disha Concurrent [10], prompt deadlock recovery depends on the number of deadlocked packets which is primarily determined by the time-out value, since deadlocked packets are routed through the path of deadlock buffers concurrently. Given too many deadlocked packets, the path is overloaded, degrading performance. To investigate this correlation in detail, we conducted simulations for a wide range of time-out values. Fig. 3 (a) shows the result. It is noted that the deadlock buffer (DB) utilization ranging from 0.05 to 0.11 yields the best throughput. Therefore, we may conjecture that it is possible to improve throughput by managing the time-out flexibly so that deadlock buffer utilization be within some range. To verify our conjecture, we conducted simulation for the same network conditions and varying initial time-out values. As shown in Fig. 3 (b), for each initial time-out value, one experiment is conducted without adjusting the time-out during simulation and another experiment with adjusting the time-out to maintain deadlock buffer utilization in the range of 0.05 and 0.11. It is seen that adjusting the time-out flexibly during simulation yields better throughput regardless of the initial time-out value.



(a) Fixed time-out values during simulation. (b) Throughput improvement by time-out values adjusted flexibly during simulation.

Fig. 3. Deadlock buffer utilization and/or throughput vs. time-out value, offered traffic rate = 0.45.

The above observation motivates us to develop an idea that adjusts the time-out value flexibly by periodically checking the deadlock buffer usage. Specifically, if the deadlock buffer usage is high, increase the time-out value. Otherwise, consider either of the two situations; the network is lightly loaded or the time-out value is too large in a heavily-loaded network. In the latter case, a shorter time-out would facilitate dissipating congestion as it decreases utilization of normal buffers. To realize this idea of managing the time-out flexibly, named FLEX-TO recovery scheme, it is required to estimate network congestion status and determine the proper rate of deadlock buffer utilization. The time-out needs to be modified only when the network is congested. There have been several approaches developed for measuring and controlling network congestion [1]. We

take one of the useful approximations of congestion in the literature that is obtained by measuring busy output channels at each node [8]. When the network is presumed as heavily loaded, our recovery scheme adjusts the time-out value, thus realizing the selected proper utilization rate of deadlock buffers. The scheme utilizes some predetermined values as follows.

1. Set a threshold for output channel utilization rate (TH_{CH}).
2. Set a threshold for deadlock buffer utilization rate (TH_{DB}).
3. Set a minimum time-out value ($TO_{\min}$) and a maximum time-out value ($TO_{\max}$).
4. Set a time-out update period (I).
5. Set a time-out increment/decrement value (VAL).

After determining the above parameters, FLEX-TO recovery scheme runs the following steps at each router at every time-out update period.

1. Calculate the current utilization rate of output channels (CUR_{CH}).
2. If $CUR_{CH} < TH_{CH}$ then exit.
3. Calculate the current utilization rate of the deadlock buffers (CUR_{DB}).
4. If $TH_{DB} - CUR_{DB} > \alpha$ for some $\alpha > 0$ and TO (current time-out value) $> TO_{\min}$ then decrease TO by VAL.
5. Otherwise if $CUR_{DB} - TH_{DB} > \alpha$ and TO $< TO_{\max}$ then increase TO by VAL.

In steps 4 and 5, if α is too small, the scheme is very sensitive to the change of deadlock buffer utilization rate, thus frequently updating the time-out value. On the other hand, a large α rarely changes the time-out value if the network is stable.

3. PERFORMANCE

3.1 Simulation Model

Performance comparison is mainly made with a previous deadlock detection scheme in [7] in combination with two deadlock recovery schemes, Disha Concurrent and Disha Sequential [10]. Disha Sequential requires sequential access to the recovery path; only one deadlocked packet at a time can proceed through the recovery path after capturing a token. To our knowledge, the scheme in [7] is most sophisticated and efficient in terms of the number of deadlock detections. In order to view performance difference between various paradigms of routing strategies, we also experimented on those routing schemes most studied in the literature through simulation. Namely, we evaluated the fully adaptive Duato's avoidance-based scheme [3], the Negative-First routing algorithm, and Compressionless Routing [6], a true fully adaptive regressive recovery routing scheme. Duato's scheme has been widely studied as a typical example of deadlock avoidance algorithms and shown to exhibit superior performance over other existing avoidance routing algorithms [5, 14]. As a result, it has been practically accepted in real systems such as the Cray T3E [13] and Reliable Router [2]. The Negative-First algorithm is partially adaptive and shown to yield the best results among the Turn Model schemes dis-

cussed previously [4]. Compressionless routing recovers from deadlock by simply killing the packets involved and later reinjecting them into the network after some delay.

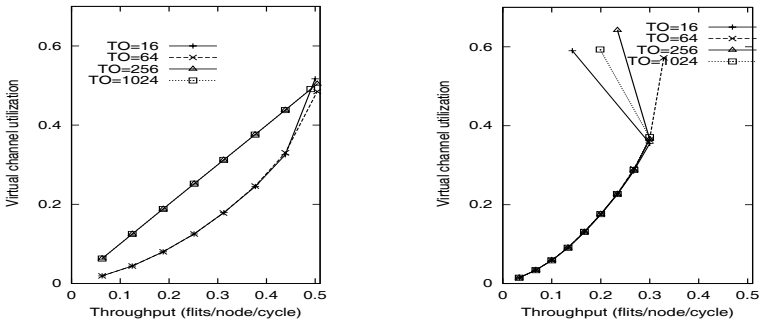
All simulations are conducted at the flit level. The routing algorithm can use any minimal path to forward a message toward its destination. Performance comparison is made on two important metrics, the ratio of packets presumed as deadlocked (simply, the ratio of deadlocked packets) and throughput. Throughput is measured as normalized accepted traffic in flits per node per cycle. Considering the sizes of current multicomputers and widely-used topologies in the simulation study, we simulated the schemes on 8x8x8 meshes and 8-ary 3-cubes. A node is equipped with one injection and reception channel at the network interface. A physical channel between nodes is shared by three virtual channels of buffer depth of two flits. It is assigned to a virtual channel in a demand-slotted round robin fashion. We used the true fully adaptive routing except for Duato's scheme [3], which requires one virtual channel for avoiding deadlock in mesh networks, thus two virtual channels remaining for true fully adaptive routing. For 8-ary 3-cubes, the scheme requires one more channel for deadlock avoidance, leaving only one channel for true fully adaptive routing. It is assumed that both routing time and transmission time of a flit across a channel equal one clock cycle. Also it takes one cycle to transfer a flit from an input buffer to an output buffer unless there is congestion. The crossbar switch allows multiple messages to traverse it simultaneously.

Each simulation result was obtained after running the program sufficiently long enough for the network to operate in a steady state or become saturated. We discarded the data collected during the initial transient period, *i.e.*, before the network enters into a steady state. We simulated the packet length of 32 flits and the uniform, perfect-shuffle, and bit-reversal distributions of packet destinations. Packets are generated with the injection rate varying from low load to saturation and exponentially distributed, where the same rate is applied to all nodes.

For the simulation of Disha Sequential, the token propagates at twice the router clock speed with no impact on router delay, as also assumed in [10]. The central deadlock buffers for Disha Sequential have the size of two flits. For Disha Concurrent, the router is provided with two types of deadlock buffers, both of two-flit size, using the Hamiltonian-path with shortcuts [17]; one in the increasing order for those packets with the destinations of higher ids and the other in the decreasing order for those packets with lower destinations.

3.2 Tuning the Parameters

In this section, we select appropriate values for those parameters required by FLEX-TO recovery scheme discussed in section 2.2 for best performance. Fig. 4 shows output channel utilization rates vs. throughput for different time-outs for the uniform traffic patterns. Output channel utilization is measured as the ratio of the number of virtual channels occupied by packets. It is noted that the network begins to be saturated when the utilization rate is over 0.5 for 8-ary 3-cubes. For meshes, the rate is approximately 0.4, regardless of the time-out. Similar results are obtained in [8] where nine out of eighteen virtual output channels are busy at saturation regardless of message length for a $8 \times 8 \times 8$ torus. Therefore, we selected 0.5 as the threshold for output virtual channel utilization rate (TH_{CH}) to activate FLEX-TO recovery scheme for torus networks. For meshes, the selected threshold rate is 0.4.



(a) 8-ary 3-cubes. (b) $8 \times 8 \times 8$ meshes.

Fig. 4. Output channel utilization for varying time-out value of the proposed deadlock detection scheme with Disha Concurrent recovery scheme.

Next we determine a proper threshold for deadlock buffer utilization rate (TH_{DB}). As mentioned in section 2.2, the utilization rate is calculated as the mean ratio of the number of the deadlock buffers occupied by deadlocked packets per cycle. We used various threshold values ranging from 0.05 to 0.11 for 8-ary 3-cubes, as obtained from the simulation results in section 2.2, and found that throughput differences are almost negligible even at high network load. For meshes, the range of thresholds was chosen from 0.03 to 0.11, which was found to yield better throughput than with the other thresholds. Although any threshold within this range produces almost the same throughput, the thresholds from 0.07 to 0.10 yield slightly higher throughput. Among these values, we selected 0.08 as a threshold for deadlock buffer utilization rate in the results presented next. As discussed in section 2.2, the time-out is adjusted when CUR_{DB} (current deadlock buffer utilization rate) differs from the threshold by α. After examining the threshold range yielding better throughput, we chose 0.02 for α which is approximated difference between the threshold (0.08) and each limit value of the range.

Now consider the time-out update period (I) and update value (VAL). Intuitively, with a large I, the scheme slowly reflects the status on the time-out, thus rather insensitive to the change of network status. A large VAL will fluctuate the deadlock buffer utilization rate sharply, as well as the time-out. To study their effects on performance, we simulated our schemes for heavy traffic load; recall that FLEX-TO recovery scheme is triggered only when the network load is over the predetermined threshold. The experiments were conducted for (10, 5), (50, 5), and (100, 10), where the first value of the pair indicates I and the second one VAL. It was noted that the scheme reflected the network status most promptly with (10, 5), whereas the scheme tended to make relatively infrequent time-out change with the larger I. It turned out that throughput difference after applying each of these parameter values was minimal. We selected (50, 5) as values for I and VAL, respectively. For $TO_{\min}$ and $TO_{\max}$, we selected 4 and 1024 cycles, respectively, considering the range of time-out values, 16 to 1024 cycles, used in our experiments.

3.3 Simulation Results

This section presents simulation results of several schemes under various network conditions: the scheme in [7] together with Disha Sequential [10] (LPZ-DS), that with Disha Concurrent [10] (LPZ-DC), our proposed deadlock detection scheme with Disha

Concurrent (PRP-DC), our detection scheme with FLEX-TO recovery scheme (PRP-FLX), the fully adaptive Duato's avoidance-based scheme (DUATO) [3], the Negative-First routing algorithm (TURN), and Compressionless Routing (COMPL) [6]. The performance of PRP-DC is studied to fairly investigate the efficiency of the proposed deadlock detection scheme in terms of the number of deadlock detections as compared with the scheme in [7] when the same recovery scheme, Disha Concurrent, is applied.

We measured the ratio of packets presumed as deadlocked by each deadlock detection scheme for varying packet injection rate (offered traffic) and time-out value (TO). Offered traffic loads are normalized with respect to the network's maximum wire capacity, defined as all of the network channels transmitting simultaneously. Fig. 5 shows the results for $8 \times 8 \times 8$ meshes for uniform traffic patterns. For PRP-FLX, the specified time-out values are given initially to the simulator and adjusted during simulation according to FLEX-TO recovery scheme. The schemes behave similarly in that the ratios are almost zeros for normal network loads but increase substantially for heavy loads. In general, the ratio is inversely related with the time-out in every scheme except PRP-FLX.

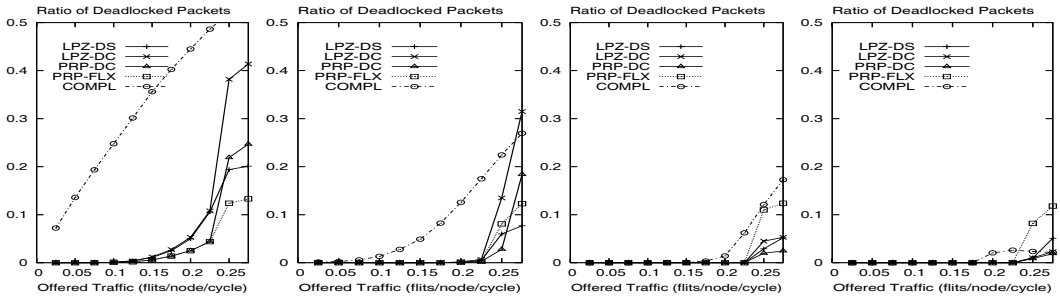


Fig. 5. Deadlocked packet ratio in $8 \times 8 \times 8$ meshes for uniform traffic patterns when (a) TO = 16; (b) TO = 64; (c) TO = 256; and (d) TO = 1024 cycles.

It is noted that PRP-FLX yields the lowest ratio for TO of 16 cycles, but the highest ratio for the large TO of 1024 cycles at high loads. This is because of the property of FLEX-TO recovery scheme, as it flexibly adjusts the time-out based on the deadlock buffer utilization. That is, for a small time-out in congested networks, the deadlock buffers tend to be heavily loaded, thus having the time-out increased. Consequently, packets would be presumed as deadlocked less frequently than the other schemes. On the other hand, for a large time-out in congested networks, the opposite situation occurs.

COMPL performance varies most depending on the time-out, as shown in Fig. 5. The obvious reason is that it adopts the loosest condition for deadlock detection, *i.e.*, simply killing packets that have not progressed for time-out at the sources. Among deadlock recovery schemes, LPZ-DC seems more sensitive to the time-out for saturated networks, as its range of ratios for different time-outs is the largest. Specifically, at saturation (rate of 0.225), the ranges are 0.10, 0.11, and 0.04, for LPZ-DC, LPZ-DS, and PRP-DC, respectively. For highly saturated situation (rate of 0.25), they are 0.37, 0.18, and 0.21 in that order. In terms of the number of deadlock detections, PRP turns out to be more efficient than LPZ, as the ratio for LPZ is as much as 2.5 times higher for TO of

16 cycles at saturation. Even for the large TO of 1024 cycles, LPZ detects approximately 1.2 times more deadlocks than PRP.

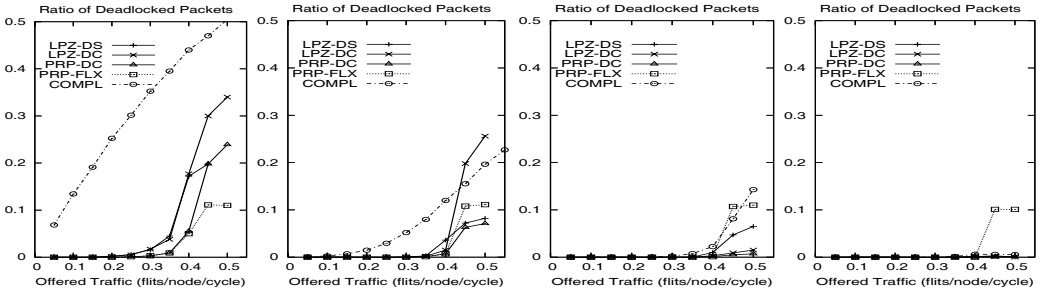
Note in Fig. 5 that LPZ-DS yields lower ratio than LPZ-DC and PRP-DC at highly saturated networks (rate of over 0.25) for relatively short TOs of 16 and 64 cycles. The reason is as follows. DC enables concurrent recovery of deadlocked packets proceeding along the deadlock buffers. On the contrary, DS allows only one packet to use the recovery path, since a packet for DS must wait for the token for recovery. During the waiting time, there is a higher chance that the packet presumed as deadlocked may obtain a next feasible channel and become unblocked, since a network with a fully adaptive routing rarely enters into deadlock. In case of larger TOs of 256 and 1024 cycles, such effect is diminished, since the time-out is sufficiently long enough for blocked packets to proceed, substantially decreasing the number of deadlock detections for LPZ and PRP.

Fig. 6 plots the ratio of deadlocks for 8-ary 3-cubes. The schemes behave almost similarly as in meshes. However, the ratios are lower in general than those for meshes, due to the higher adaptiveness property of tori. Note that the ratios for PRP-DC are still lower than those for LPZ-DC in every result, even though PRP enforces loose criteria for deadlock declaration for tori than for meshes. Specifically, the ratios for LPZ-DC are 2.1 to 3.2 times higher than those for PRP-DC for the TO less than 1024 cycles at saturation.

Simulation experiments are also conducted with non-uniform traffic patterns for 3D tori. Fig. 7 depicts the deadlocked packet ratios for perfect-shuffle and bit-reversal traffic patterns for the TOs of 16 and 256 cycles. They tend to increase at lower offered traffic loads, but are in general lower than for uniform traffic patterns, except for PRP-FLX. The ratios quickly level off at high loads regardless of time-outs and traffic patterns. For the TO of 256 cycles, LPZ and PRP-DC yield almost no deadlocks. Different from the uniform traffic pattern case for the TO of 16 cycles, PRP-DC results in the lowest ratio for both non-uniform traffic patterns. It is noted that PRP-FLX consistently yields the ratio of 0.1 to 0.15 at heavy loads for all traffic patterns and all time-outs presented.

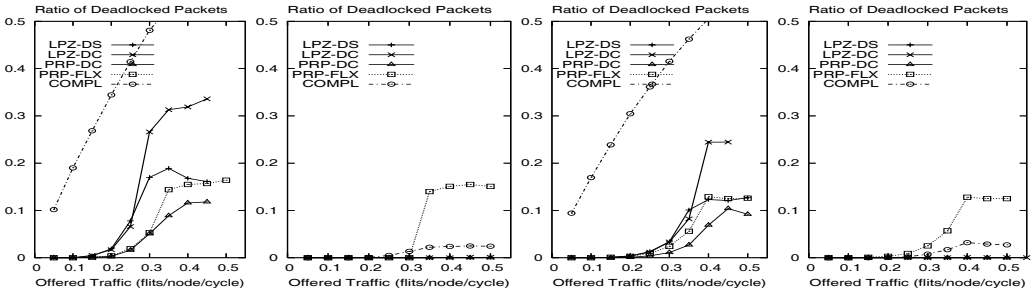
Fig. 8 plots normalized throughput of each scheme in 3D meshes. It is observed that the schemes except TURN and COMPL perform almost the same for low and moderate loads but differ at high loads. In particular, PRP-FLX generally performs better than the other schemes for all time-outs. This is because PRP-FLX manages to allocate the channel bandwidth properly by not overloading the recovery path. Especially, such management takes effect for the TO of 16 cycles, as PRP-FLX significantly outperforms all the other schemes, yielding from 1.9 to 6.68 times higher throughput at highly offered traffic of 0.25. One exception is COMPL which tends to sustain its performance at high loads for low TOs of 16 and 64 cycles. This is mainly because it kills sufficient number of packets for the network to return to normal condition. However, for larger time-outs, COMPL performance degrades more significantly than the other schemes, demonstrating its degree of sensitivity to time-out, which is also addressed in [6, 10].

It is noticed in Fig. 8 that LPZ-DS performs worst at heavy loads. This is attributed to its sequential recovery procedure. That is, although it results in lower ratio of deadlocked packets than LPZ-DC or PRP-DC in some cases as shown in Fig. 5, only one deadlocked packet at a time is recovered, thus taking much longer time for recovery. For instance, the recovery time of a deadlocked packet for LPZ-DS is approximately 1.98 times longer than that for LPZ-DC, for the TO of 64 cycles at the offered traffic of 0.25. Unless the network is significantly heavy, TURN yields the lowest throughput, due to its restriction on routing adaptiveness.



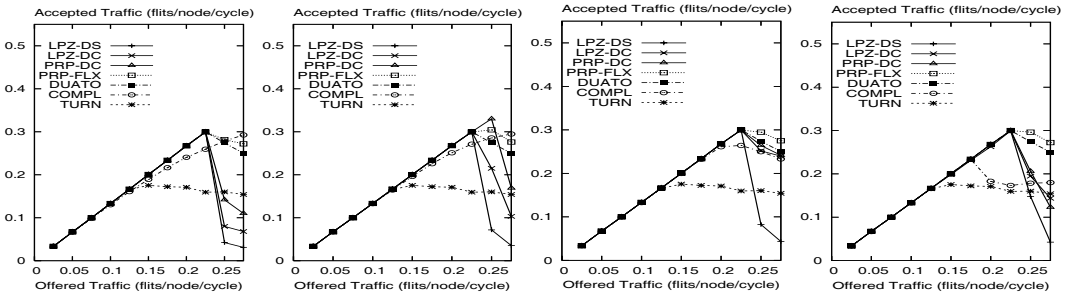
(a) (b) (c) (d)

Fig. 6. Deadlocked packet ratio in 8-ary 3-cubes for uniform traffic patterns when (a) TO = 16; (b) TO = 64; (c) TO = 256; and (d) TO = 1024 cycles.



(a) (b) (c) (d)

Fig. 7. Deadlocked packet ratio in 8-ary 3-cubes. Perfect-shuffle traffic patterns when (a) TO = 16 and (b) TO = 256 cycles. Bit-reversal traffic patterns when (c) TO = 16 and (d) TO = 256 cycles.



(a) (b) (c) (d)

Fig. 8. Throughput in 8 × 8 × 8 meshes for uniform traffic patterns when (a) TO = 16; (b) TO = 64; (c) TO = 256; and (d) TO = 1024 cycles.

Interestingly, throughput for the TO of 1024 cycles is lower than that for the TO of 256 cycles for LPZ-DC and PRP-DC at high loads. This is because those schemes redirect blocked packets to the deadlock buffers less frequently for the longer time-out, thus being unable to quickly relieve normal flit buffers from congestion. For the lower TOs of 16 and 64 cycles, PRP-DC outperforms LPZ-DC, although they use the same recovery scheme. This is attributed to the lower ratio of deadlocked packets by PRP-DC, as seen in Fig. 5. If the ratios are too high, deadlocked packets would obviously encounter considerable delay when routed along the deadlock buffers. Fig. 8 shows that DUATO sur-

prisingly outperforms all the progressive deadlock recovery schemes except PRP-FLX for all time-outs. Its behavior is similar to that of PRP-FLX, as its throughput tends to drop slightly after saturation. This indicates that efficient use of the deadlock buffers is critical to network performance.

Fig. 9 plots the performance of each scheme in terms of throughput in 3D tori. The schemes behave almost similarly as in meshes, although they show higher peak throughput. In general, PRP-FLX continues to outperform all the other schemes, especially for large time-outs and high loads. One of the main differences from the results shown in Fig. 8 is that DUATO performs relatively worse. The obvious reason is that DUATO requires two virtual channels to avoid deadlock for tori, leaving only one channel for fully adaptive routing, whereas it uses two channels for fully adaptive routing for meshes. TURN also performs worse than in meshes because it provides non-minimal routes in tori, as discussed in [6]. Another difference is that LPZ-DS performs comparably to LPZ-DC and PRP-DC for the TO of 1024 cycles. This is due to its ratio of deadlocked packets as shown in Fig. 6 (d), where all the three schemes generate almost no deadlocked packet, obviously leading to virtually no performance difference. In the figure, COMPL performs better than DUATO for the TO less than 1024 cycles, while DUATO is slightly better for the TO of 1024 cycles at high loads over 0.35. This result matches with that in [5, 10]. Moreover, DUATO significantly outperforms TURN for all the configurations, which is also demonstrated in [10].

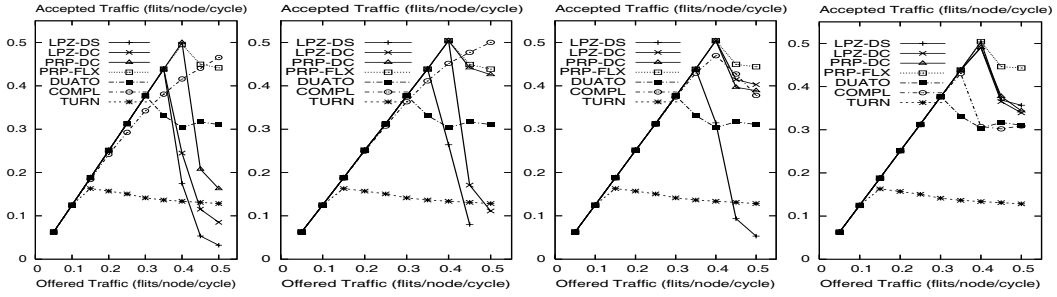


Fig. 9. Throughput in 8-ary 3-cubes for uniform traffic patterns when (a) TO = 16; (b) TO = 64; (c) TO = 256; and (d) TO = 1024 cycles.

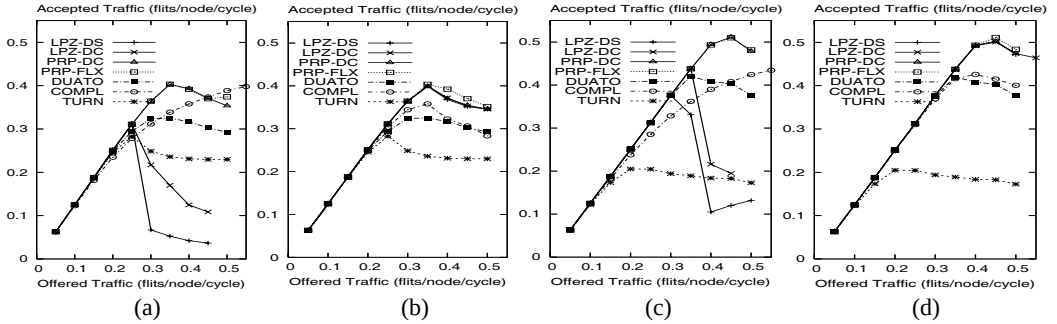


Fig. 10. Throughput in 8-ary 3-cubes. Perfect-shuffle traffic patterns when (a) TO = 16 and (b) TO = 256 cycles. Bit-reversal traffic patterns when (c) TO = 16 and (d) TO = 256 cycles.

Fig. 10 presents throughput of each scheme for non-uniform traffic patterns for 3D tori. The schemes behave very similarly as in the uniform traffic patterns. TURN shows relatively better performance in perfect-shuffle patterns than in the other patterns. This is also the case for DUATO, since its performance difference from PRP-FLX is much less than in the uniform patterns. In particular, DUATO performs comparably to COMPL for the TO of 256 cycles in the non-uniform patterns, while it is outperformed by COMPL in the uniform patterns. It is noticed that PRP-DC does not degrade significantly for the TO of 16 cycles, but yields performance competitive with PRP-FLX in the figure. In fact, for large time-outs, the progressive deadlock recovery schemes exhibit little difference in throughput.

4. CONCLUSIONS

This paper proposed efficient deadlock detection and recovery schemes for true fully adaptive routing in wormhole networks. Deadlock detection is based on the turn model and intended to mark only one packet as deadlocked in a simple cycle. Our deadlock recovery scheme operates by adjusting the time-out value flexibly according to the utilization rate of the recovery resources, rather than fixing a single time-out value as in previous schemes. Simulation experiments were conducted to show that the proposed deadlock detection scheme significantly outperforms previous schemes in terms of the number of packets detected as deadlocked for low to moderate time-out intervals. Performance study also shows that the proposed deadlock recovery scheme yields better throughput over previous ones and deadlock avoidance-based routing schemes. Although simulation results for only $8 \times 8 \times 8$ meshes and tori are presented in this paper, similar results were obtained for larger and smaller networks such as 16×16 and $4 \times 4 \times 4$ meshes. The main advantage of the proposed scheme is that it virtually removes the need for determining a best time-out value satisfying various network conditions.

REFERENCES

1. E. Baydal, P. Lopez, and J. Duato, "A family of mechanisms for congestion control in wormhole networks," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 16, 2005, pp. 772-784.
2. W. J. Dally, L. R. Dennison, D. Harris, K. Kan, and T. Xanthopoulos, "The reliable router: A reliable and high-performance communication substrate for parallel computers," in *Proceedings of the 1st Workshop on Parallel Computer Routing and Communication*, 1994, pp. 241-255.
3. J. Duato, "A new theory of deadlock-free adaptive routing in wormhole networks," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 4, 1993, pp. 1320-1331.
4. C. J. Glass and L. M. Ni, "The turn model for adaptive routing," *Journal of the ACM*, Vol. 41, 1994, pp. 874-902.
5. A. Khonsari, A. Shahrabi, M. Ould-Khaoua, and H. Sarbazi-Azad, "Performance comparison of deadlock recovery and deadlock avoidance routing algorithms in wormhole-switched networks," *IEE Proceedings of Computers and Digital Techniques*, Vol. 150, 2003, pp. 97-106.
6. J. Kim, Z. Liu, and A. Chien, "Compressionless routing: Aframework for adaptive

and fault-tolerant routing," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 8, 1997, pp. 229-244.
7. J. M. Martinez, P. Lopez, and J. Duato, "FC3D: Flow control-based distributed deadlock detection mechanism for true fully adaptive routing in wormhole networks," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 15, 2003, pp. 765-779.
8. J. M. Martinez, P. Lopez, and J. Duato, "A cost-effective approach to deadlock handling in wormhole networks," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 12, 2001, pp. 716-729.
9. P. Mohapatra, "Wormhole routing techniques for directly connected multicomputer systems," *ACM Computing Surveys*, Vol. 30, 1998, pp. 374-410.
10. T. M. Pinkston, "Flexible and efficient routing based on progressive deadlock recovery," *IEEE Transactions on Computers*, Vol. 48, 1999, pp. 649-669.
11. T. M. Pinkston and S. Warnakulasuriya, "Characterization of deadlocks in *k*-ary *n*-cube networks," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 10, 1999, pp. 904-921.
12. L. Schwiebert, "Deadlock-free oblivious wormhole routing with cyclic dependencies," *IEEE Transactions on Computers*, Vol. 50, 2001, pp. 865-876.
13. S. L. Scott and G. M. Thorson, "The cray T3E network: Adaptive routing in a high performance 3D torus," in *Proceedings of Symposium on Hot Interconnects IV*, 1996, pp. 147-156.
14. A. Shahrabi and M. Ould-Khaoua, "On the performance of routing algorithms in wormhole-switched multicomputer networks," in *Proceedings of the 11th International Conference on Parallel and Distributed Systems*, 2005, pp. 515-519.
15. Y. H. Song and T. M. Pinkston, "A progressive approach to handling message-dependent deadlock in parallel computer systems," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 14, 2003, pp. 259-275.
16. Y. M. Sun, C. H. Yang, Y. C. Chung, and T. Y. Huang, "An efficient deadlock-free tree-based routing algorithm for irregular wormhole-routed networks based on the turn model," in *Proceedings of International Conference on Parallel Processing*, 2004, pp. 343-352.
17. S. C. Wang, H. Y. Lin, and S. Y. Kuo, "A simple and efficient deadlock recovery scheme for wormhole routed 2-dimensional meshes," in *Proceedings of Pacific Rim International Symposium on Dependable Computing*, 1999, pp. 210-217.

Soojung Lee (李秀貞) received the B.S. degree in Mathematics from Ewha University, Seoul, Korea in 1985. She got the M.S. and Ph.D. degrees in Computer Science from Texas A&M University, in 1990 and 1994, respectively. She had been a senior engineer at the Telecommunication Research Center, Samsung Electronics Co. from 1994 through 1998. She is currently a professor in the Department of Computer Education, Gyeongin National University of Education. Her research interests include data mining, information retrieval, distributed computing, computer networks, and computer education.