완전 적응 웜홀망에서 교착 복구 기반 다중 전송 알고리즘

(A Deadlock Recovery-based Multicast Algorithm for Fully Adaptive Wormhole Networks)

정 종 인 † 김 인 †† 김 시 관 †††
(Jongin Chung) (Ihn Kim) (Si-Gwan Kim)

요 약 최소의 다중전송 지연시간을 갖기 위하여 다중 목적지 패킷의 경로 길이와 웜의 초기화 횟수를 최소화한 다중전송 알고리즘을 제안한다. 그 알고리즘은 교착이 탈출 채널에서 복구되면서 단일 전송과 다중전송 메시지에 대하여 완전 적응하며 교착이 발생하지 않게 한다. Disha로 동시에 경로 기반 다중 전송할 때 교착이 발생하지 않기 위해서는 2개의 교착버퍼가 필요하다. 다중 전송 지연 시간은 메시지 startup 지연시간에 좌우된다. 제안된 시스템은 단지 2번 이하의 메시지 startup 횟수를 가지며 이것은 기존의 어떤 다른 시스템보다 뛰어난 성능을 갖는다.

Abstract We proposed a multicast algorithm that can perform multicast operation in minimum number of worm initiation and minimizing the path length of the multi-destination packet to achieve minimum multicast latency. The algorithm ensures deadlock freedom and achieves full adaptation for both unicast and multicast message while deadlocks can be recovered in the escape channel. 2 deadlock buffers are used for a deadlock free path-based multicast in Disha concurrent. Multicast latency is dominated by message startup latency. The proposed scheme need only 2 or less message startup, which outperform any existing schemes.

1. Introduction

Performance of multicomputers and distributed shared memory multiprocessors are critically affected by the performance of the interconnection network system. Most modern parallel processing systems use wormhole switching due to their low communication latency and high bandwidth. Wormhole switching is particularly suitable for low dimensional networks because it has communication latencies relatively insensitive to the distance between the source and destination node. Unfortunately, wormhole switching is very susceptible to deadlock because the flow control within the network may block the trailing flits, and they remain in the network occupying flit buffers along the established route. Several schemes have been proposed to cope with this problem. Most of them are based upon a strategy that provide deadlock-free routing by restricting the routing function so that worms cannot form a cyclic waiting condition. Trends have been to loosen this restriction and achieve more routing adaptation to improve network performance[1-5].

Several parallel applications require one-to-many communication which involves a group of inter-communicating nodes. Some parallel programming languages provide efficient support for these kinds

· This work was supported by postdoctorial fellowship program from Korea Science & Engineering Foundation(KOSEF)

† 종신회원 : 공주대학교 컴퓨터교육과 교수
jichung@kongju.ac.kr
†† 비 회 원 : Dept. of EE on System, University of Southern California
ihnkim@usc.edu
††† 비 회 원 : 안동대학교 멀티미디어학과 교수
sgkim99@andong.ac.kr
논문접수 : 2000년 9월 28일
심사완료 : 2001년 4월 11일

of communication patterns for simplifying the programming and enhancing code portability of multicomputers. Collective communications defined in the Message Passing Interface include broadcast, multicast, gather, scatter, all-gather, barrier synchronization, and global reduction. Among these, multicast service which sends messages to multiple arbitrary number of nodes is the most complex one and other services can be benefited from efficient implementation of multicast.

Approaches to implement multicasting can be divided into two categories. One is tree-based approach, the other is path-based approach. In tree-based approach, the source-rooted tree is formed and messages flow down the tree. In this approach, each node in the network should be able to replicate messages to send out one or more copies through different output channels. These multiple predecessors can introduce dependencies and are susceptible to deadlock.

The disjoint spanning-tree multicast(DSTM) algorithm[4] provides deadlock-free multicast routing that is fully compatible unicast. The hardware supported tree-based multicast in wormhole-routed network[5] uses up-down routing with a globally-agreed spanning tree to avoid deadlock. Since up-down routing uses links in a strictly monotonic order, circular waiting is prevented and deadlock freedom is achieved for unicast. The tree-based algorithm using up-down routing produces a potential congestion problem, especially at the tree links around the root. This may cause a message to hold many channels for an extended period of time thereby increasing channel contention. In addition, contention may arise between concurrent multicast operations initiated by different sources.

Alternatively, in path-based approach, the header of each multicasting message consists of multiple ordered list of destinations. When the header packet reaches the first destination, the address of the current node is removed off by the router and messages are sent to the next address. While the flits are forwarded to the next destination, they are also copied flit by flit to the message buffer to that node. However, although they can send a message to multiple destinations in single startup, path must be selected carefully because it can cause deadlock during the path to the final destination even if it follows legal path between all the inter-destination path.

Dual-path and Multi-path scheme restricts routing function of a multidestination worm to avoid this situation. Though this scheme improves multicast performance over multiple unicast routing, using different routing algorithm for multicast and unicast complicates router design and should lead to longer clock cycle. Furthermore, they are based on the Hamiltonian path and by strictly restricting routing function of a multidestination worm, overall communication latency could be dominated by path length and blocking time.

D.K.Panda proposed *Base-Routing Conformed Path(BRCP)* model which supports multidestination messages without introducing additional channel dependencies with respect to the base routing algorithm used for unicast routing[6]. In his Hierarchical Leader scheme (HL) multicast operation can be achieved in (m+1) phase using $\lceil \log_2 \lfloor L_m \rfloor + 1 \rceil + m$ communication startups where m is the number of grouping levels.

We can extend this model for those routing algorithms which allow cyclic dependency between channels and some virtual channels are used exclusively for the recovery or preventing of deadlocks while allowing fully adaptive routing on other channels, by letting the hierarchical grouping be done based on the routing algorithm of the escape channel. The routing algorithm in the escape channel should not allow cyclic dependency between channels so that a message drained to this channel class is guaranteed to be consumed at the final destination node. However, since a multicast message drained to the escape channel is not consumed at the intermediate destination, there should be some ordering for destinations of a multi-destination packet so that next destination can be reached at any node during the path to the last destination through the escape channel.

The transition from tree-based approach to path-based approach is basically based on the fact that multicast latency is dominated by message startup latency. In current generation machines, communication startup time(ts) is around 1.0-35 microseconds while propagation time per flit per hop (tp) is in the range of 5.0-15.0 nanoseconds[6]. Thus, communication latency is dominated by ts. And furthermore, tree-based approach is not suitable for wormhole switching which is very susceptible for deadlock as mentioned earlier. So, one may be interested in obtaining reduced communication latency primarily by delivering to as many destinations as possible with a single packet assuming path-based approach.

D.K. Panda's HL and Multiphase Greedy scheme (MG) showed less average number of packet initiation compared to the U-mesh algorithm. However, the worst-case scenario of these schemes are still order of $\lceil \log_2(k) \rceil$ for e-cube routing of unicast message. This is because of the fact that depending on the base-routing algorithm, a new packet initiation may be needed for those destination that lie outside the path range that can be covered with a single packet. This effect is more pronounced for smaller system size in which number of participating nodes are less than 100[6]. In that case, the Dual-Path scheme (DP) which always require only 2 packet initiations outperform HL scheme. For bigger system size, HL or MG schemes showed shorter latency than DP. DP restricts routing function based on Hamiltonian path to avoid deadlock. Because of this strong restriction of routing function, the packet path length of a multi-destination worm should follow quickly grows as the system size gets bigger. In this case, the trip length can dominate the multicast latency hence grouping of destinations and having more worm-initiation is profitable due to the large channel propagation time and channel contention time. This problem of elongated packet length of a multi-destination worm should be minimized for full benefit of the path-based scheme.

So, the first objective of this study is to develop a multicast algorithm that can perform multicast operation in minimum number of worm initiation and minimizing the path length of the multi-destination packet to achieve multicast latency. Of course, deadlock freedom of this algorithm also should be ensured.

Secondly, we want to allow adaptation for multi-destination packets so that any minimal paths between two intermediate destinations could be used. By allowing adaptation channel contention problem could be relieved compared to the strong Hamiltonian-based schemes. There are proposed schemes that are aimed to provide adaptation for multi-destination packets. Partially Adaptive Minimal Multicast Routing (PM) scheme allows partial adaptation based on turn model[7]. Label-Based Dual Path (LD) scheme can provide non-minimal adaptive routing of a multi-destination packet but also limited to partial adaptation[7]. So, the second objective of this study is to develop an multicast algorithm that can allow true-fully adaptation of multi-destination packets. This is important because as the multicast-zone size becomes larger and load rate of the network becomes higher, the portion of multicast latency due to channel contention time could become dominant especially for longer packets like multi-destination worm.

Finally, it should be pointed out that using different hardware for multicast and unicast complicates router design and could lead to longer clock cycle. Restricting routing function or slowing router degrades unicast performance which represents most of the network traffic. It makes no sense to sacrifice unicast performance for relatively rare multicast performance. Some of the recent deadlock recovery schemes claim improved unicast performance by applying Duato's theory[8,9]. So, the third objective is to develop a multicast algorithm that is completely compatible with high performance unicast algorithm.

In this paper, we'd like to propose a routing algorithm that can achieve the full adaptation for both unicast and multicast while deadlocks can be

recovered in the escape channel class. By splitting destinations according to their geometrical destinations only when it is expected to be profitable, it could achieve reduced multicast latency. It provides a comprehensive routing solution without requiring excessive additional hardware complexity.

2. Proposed Multicast Scheme

This scheme corresponds to the Disha[8,9] concurrent unicast algorithm which uses a deadlock-recovery mechanism. By using recovery strategy instead of avoidance one based on the observation that deadlocks are very rare given reasonable degree of routing freedom, this routing algorithm shows the most efficient and highest performance using minimal dedicated resources. The networks are composed of two virtual channel classes. One is full adaptive channel (FC) and the other is escape channel (EC). Deadlock freedom is guaranteed although cyclic waiting condition can be formed by allowing full adaptation in the FC, because it can be resolved by breaking one of the deadlocked packet to be drained in EC. The routing function in the EC should be restricted to ensure that there should be no deadlock. EC is constructed according to the Hamiltonian path. Shortcuts are allowed in the EC. If a packet is detected to be deadlocked, one of the associated packet is drained into the EC and follow those path provided by this Hamiltonian path until it is consumed at the last destination.

Different from the original Disha concurrent, 2 deadlock buffers are essential for a deadlock free path-based multicast. If there is only one deadlock buffer, indirect multicast channel dependency could create cyclic dependency in its extended multicast channel dependency graph. Deadlock freedom of the proposed scheme will be given later. The routing algorithm in a deadlock buffer(R_{db1}) is the Hamiltonian path constructed in opposite direction to the Hamiltonian path of the other deadlock buffer(R_{db2}).

Given the deadlock-free routing algorithm for unicasting, multicasting routing is done as follows. First, the source node of a multicasting first defines the Multicasting Zone which includes all the destination nodes of the multicasting. ((1) Zone Defining)

Algorithm 1: Zone Defining
Input: Destination set DS[(d_{ix}, d_{iy})]
Output: The coordinates of the upper and lower corner of the multicast zone (1_x,1_y), (u_x, u_y)
Procedure:
begin
1_x=min{DS[d_{ix}]}, 1_y = min{DS[d_{iy}]}
u_x=max{DS[d_{ix}]}, u_y = max{(DS[d_{iy}]}
end

Then, the multicasting zone is divided into groups according to their geometrical location to reduce path length of the multi-destination packet and distribute network traffic. ((2) Grouping) By grouping destinations according to their geometrical distribution, destination splitting algorithm becomes simpler and faster since there is no need to incorporate the base-routing algorithm in splitting destinations compared to other multicast algorithms. While grouping the destinations, we can determine the optimal group size considering system parameters such as network size, channel propagation time, message startup latency, and router delay. This is possible because the grouping based on geometrical location and because of the fact that any destination group can be covered with a single multi-destination packet. When the multicast zone size is so large that more fine grouping is profitable, the decision is made at this time.

This grouping profitability test is the major difference and advantage of this multicast scheme and grouping is done to the optimal number of groups only when it is expected to be profitable. However, grouping profitability test is not simple because of the dynamic nature of network traffic. Probabilistic or empirical approach is needed to incorporate all the effect of arbitrarily distributed multicast destinations and portion of multicast latency due to channel contention and possible deadlocks. Basically, it should compare the multicast latency between *before* grouping and *after*

grouping to number of groups. Once the grouping profitability is determined as a function of multicast zone size based on given system parameters including network load rate, the source node can quickly decide whether to go grouping or not and to how many number of groups. This group size optimization should be the next step of this study.

Algorithm 2: Grouping
Input: Destination set $DS[(d_{ix}, d_{iy})]$ $1 \le i \le k$, multicast zone (l_x, l_y), (u_x, u_y)
Output: Divided destination set. Q_1, Q_2, ···, Q_m
Procedure:
begin
  $w=(u_x-l_x)$; $h=(u_y-l_y)$
  m=optimal number of destination group
  Add each destination into the group list Q_m to which it belongs.
end

Next, the leader node at each group which is nearest to the source node among its group members is selected. ((3) Leader Selection) Those leaders will be the destinations of the first Phase multi-destination worm. And then, those leaders will be sorted according the node label of the Hamiltonian path of R_{db2} and put into the multi-destination packet header. ((4) Header Encapsulation) By ordering destinations like this, the multi-destination worm is guaranteed to find a path to the final destination even if it is caught in a cyclic waiting situation on FC.

Algorithm3: Leader Selection
Input: Divided destination set Q_m, Source node (s_x, s_y)
Output: Leader node of each group L_m,
Procedure:
begin
  1 Calculate the distance between a node in a group and the source node.
  2. Sort them using the distance as the key.
  3. Select the node nearest to the source node as a Leader of that group.
end

Algorithm 4: Header Encapsulation
Input: Destination set $L_m[(L_{ix}, L_{iy})]$, source node (s_x, s_y).
Output: Ordered list of destinations, H, placed in the message header.
Procedure:
begin
  Sort the destinations in increasing order using their labels in R_{db2} as key. ; call the sorted list S.
  Append S to H.
  Place H in the message header.
end

Now, the source node can send the multi-destination worm to those leaders consuming 1 startup latency. As mentioned above, at intermediate destination the router strips off its address from the header and forward it to the next destination while copying the message flit by flit to the message buffer. At non-destination intermediate nodes, the multi-destination worm is treated exactly the same as the unicast message. The router does not even need to know whether the packet is uni-destination or multi-destination message. It can take all the minimal paths while routed to the intermediate destination like a unicast message. But once it is determined to be drained into the EC, it is forwarded to either DB1 or DB2 based on the node label of next destination in its header and it follows the base-routing algorithm of the EC all the way to the final destination. ((5) Message routing)

Algorithm5:Message Routing for multicasting
Input: A message with ordered destination list H $[\{d_k\}]$, a local address u(x, y)
Procedure:
begin
  1. If $u=d_1$, then $H=H-\{d_1\}$ and the message is copied to the local host ; otherwise, H=H.
  2. If $H=\varnothing$, terminate the message forwarding(the last destination).
  3. The message is forwarded to the first destination in the header $d(x_d, y_d)$ like a unicast message with its destination d (x_d, y_d).
end

An illustration of this algorithm till to this step is depicted on Fig. 1.

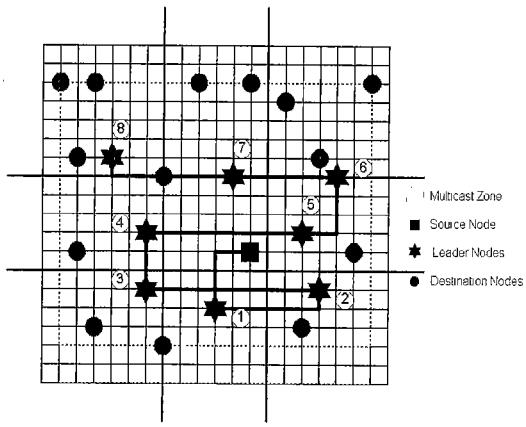

Fig. 1 An Example of proposed algorithm after phase 1

Now every leader node of each group has the message and ready to multicast to its member nodes. At phase 2, each leader node encapsulates for the second multi-destination worm using the algorithm 4 for those nodes belonging to that group and route those messages using algorithm 5 concurrently. The routing at phase 2 is illustrated in Fig.2.

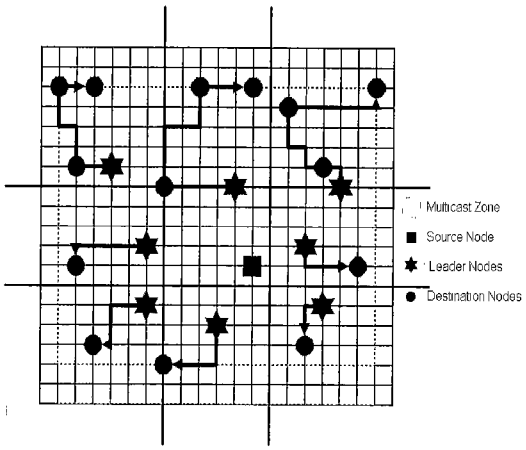

Fig. 2 An example of proposed algorithm after phase 2

3. Deadlock-freedom of the proposed scheme

According to J. Duato[3], a compatible pair (SS, R) for an interconnection network I, where SS is split and sort function and R is connected and adaptive routing function, is deadlock-free if there exists a subset of channels $C1 \subseteq C$ that defines a connected routing subfunction R1 and the pair (SS, R1) has no cycles in its extended multicast channel dependency graph D_{EM}. In the proposed algorithm, channels are divided into 2 classes. On one of the channel classes, minimal fully adaptive routing is allowed. On the other channel class, which is composed of 2 deadlock buffers, routing is restricted to follow the opposite direction Hamiltonian paths respectively including shortcuts. This routing scheme is deadlock free for unicast as shown in[4] for one deadlock buffer. It can be easily extended for 2 deadlock buffers since there is no dependency from the channels form DB2 to DB1.

In this scheme, destinations for multicast are split according to their geometrical location and sorted using the node label of DB2 as the key. (SS, R1) is compatible since for all the source nodes and for every valid destination set Q, when a message destined for the set Q is being routed, the destination subset containing the destinations which have not been reached yet is a valid destination set for the node that contains the message header. Suppose we have a destination set $Q[(d_{ix}, d_{iy})]$, $1 \leq i \leq k$, sorted according to the node label of DB2. If deadlock occurs at node e during visiting intermediate destinations, the destination set which has not been visited yet can be divided into 2 subsets.

One is Q1 which has lower node labels than the node e and the other is Q2 which has higher node labels than the node e. Those destinations of Q2 can be reached sequentially using DB2 after reaching those destinations of Q1 through DB1 since they are sorted according to the node label of DB2. This means whenever the multi-destination

packet caught in a deadlock during the path to the intermediate destinations it can find a way to the final destination covering all the intermediate destinations in the deadlock free lane. A recovery from possible deadlock is illustrated in Fig. 3.

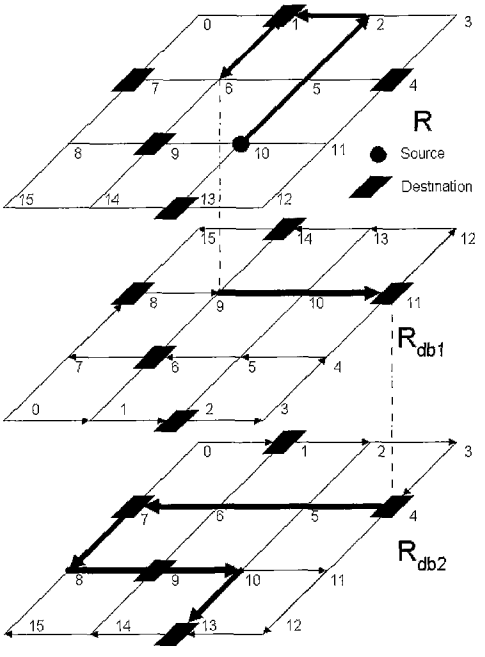

Fig. 3 Recovery from possible deadlock

The routing function of deadlock free lane R1= R_{db1}+R_{db2} is obviously connected. Any node can be reached from any node using this deadlock free lane only. Furthermore, it has no cycles in its extended multicast channel dependency graph D_{EM} since there is no dependency from DB1 to edge buffers and from DB2 to DB1 and R_{db} uses deadlock buffer in monotonic order.

Multi-destination packets can form a channel dependency that is not present in the unicast since the packet is routed again after reaching an intermediate destination. If there is only one deadlock buffer, this indirect multicast dependency can form cycles in its extended channel dependency graph. This situation is illustrated in Fig. 4.

In the case of one deadlock buffer, those channels shown in Fig. 4 are needed to make the R1 connected. Suppose a multi-destination packet is initiated from source node 10 to destinations 1 and 4. After reaching the first destination through $C_{10,5}$, $C_{5,2}$, $C_{2,1}$, the multi-destination packet can be routed to next destination 4 using $C_{1,6}$, $C_{6,5}$, $C_{5,4}$. However, while it is holding $C_{1,6}$, it could be determined to be deadlocked and follow the path $C_{6,1}$ and $C_{1,2}$, $C_{2,3}$, $C_{3,4}$ in the deadlock free lane.

The reason this path is chosen is to route in Hamitonian path. So, there could be a indirect dependency from $C_{5,2}$ to $C_{6,1}$. Likewise, if the source node is 9, there can be a indirect dependency from $C_{6,1}$ to $C_{5,2}$ forming a cyclic dependency. In this case, before reaching the lower label node than the next destination and drained into the deadlock buffer, it can be caught in a deadlock again. If there is 2 deadlock buffers, $C_{5,2}$ in the first case belongs to the FC class and there is no cyclic dependency in the extended multicast channel dependency graph of R1.

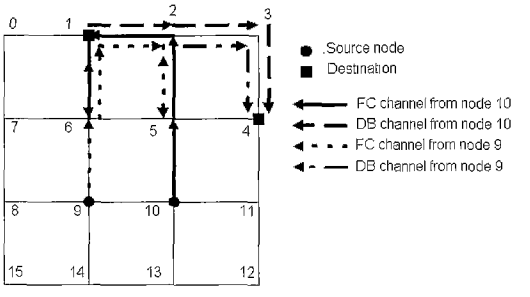

Fig. 4 Cyclic dependency in extended multicast channel dependency graph in the case of one deadlock buffer

4. Comparison with other schemes

4.1 Splitting and sorting complexity

Splitting and sorting of destination is inevitable in path-based multicast to guarantee deadlock freedom. Either the routing function of the multi-destination packet or split-and-sort function need to be precisely defined to avoid cyclic dependencies in its extended channel dependency

graph. DP limits the routing function of a packet to strictly follow the Hamiltonian paths to ensure deadlock freedom. HL and MG limits the split-and-sort function such that the path of a multi-destination packet conforms to the base routing algorithm so it does not create other dependencies than that of unicast messages (BRCP model). Since we want to restrict the routing function as minimal as possible, BRCP model seems to be the right approach.

Split function and sort function corresponds to algorithm 2 and 4 respectively in this scheme. In BRCP model, when splitting destinations, the base routing algorithm such as Hamiltonian path or e-cube routing algorithm should be incorporated to find the destinations that fall into the trajectory of a single packet. In the proposed scheme, there is no need to incorporate the base routing algorithm when grouping destinations. Instead, the number of optimal groups is calculated as a function of multicast zone size and they are grouped according to their geometrical distribution. This can be done much faster once it is predetermined given the system parameters such as router delay, channel propagation time, startup latency, and network load rate, etc. The complexity of sorting function is the same as the BRCP model and it is inevitable to guarantee deadlock freedom.

This grouping flexibility stems from using the recovery-based strategy in that it renders deadlocks to occur and provide a way the deadlocked packets can be drained to the DB to the rest of destinations. Grouping flexibility is like another knob to fine tune the multicast performance. We can just not force grouping if it is expected not to be profitable such as the case of multicasting in which number of participating nodes is less than 100. And we can consider hierarchical grouping based on geometrical proximity for far bigger multicast zones to further reduce channel occupation by the multi-destination packet.

4.2 Comparison of multicast latency

Table 1 shows the average startups of HL and MG schemes for ecube grouping for randomly generated destination sets in a 32×32 mesh[6]. It can be notified that more than 3 startups are needed to perform multicast for destination set size up to 1000. However, in the proposed scheme, only 2 or less packet startup is needed. Though more number of startups may be needed if hierarchical grouping is profitable as the destination set size grows, the proposed scheme require less number of startups in most of the reasonable destination set sizes. This is especially true for smaller destination set sizes. In Fig. 5, HL and proposed scheme is contrasted to compare number of startups necessary to perform the same multicast operation. HL scheme requires 4 startups whereas proposed scheme requires 2 startups. Although precise simulation is needed to compare the actual multicast latency, providing that startup latency dominates to a certain destination set size, the proposed scheme is expected to perform better than the HL scheme. It should be reminded that in the proposed scheme, group size can be adjusted without increasing the number of startups. In addition, fully adaptive routing is allowed in FC

Table 1 Average number of startups for HL scheme for ecube grouping in 32×32 mesh

No of Dests	64	192	320	448	576	704	768	1024
HL	7	6	6	5	5	5	4	3
MG	3.95	3.00	3.25	3.78	3.95	3.90	3.90	2.90

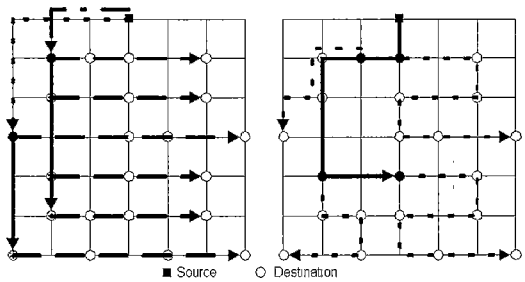

Fig. 5 Comparison of 2 multicast algorithm (a) HL using ecube dimensional order grouping and (b) proposed scheme

between destinations and Hamiltonian paths of DBs Fig. 5 Comparison of 2 multicast algorithm (a) HL using ecube dimensional order grouping and (b) proposed scheme are followed only in rare case of deadlock. So, the problem of elongated path length that existed in DP scheme is reduced to minimal in the proposed scheme.

5. Conclusion

Based on deadlock recovery strategy, an efficient multicast algorithm is proposed. The proposed multicast scheme can perform multicast operation in a minimum number of packet startups and minimizing the path length of the multi-destination packet to achieve reduced multicast latency. The most unique feature of the proposed multicast scheme is grouping flexibility. Destination grouping can be done to the optimal group size considering the system parameters such as destination set size, channel propagation time, message startup latency, and router delay. It allows minimal full adaptation during inter-destination routing. The same routing function is multi-destination packet as the unicast packet. Deadlock freedom of the proposed scheme is discussed. Two deadlock buffers are required for deadlock free multicasting in Disha concurrent. The proposed scheme could easily be extended to higher dimensional networks having Hamiltonian path. Future research is to simulate exact multicast latency of the proposed algorithm using the simulator, Flitsim which Dr. Pinkston's SMART group has updated.

References

[1] X. Lin, P. McKinley and L.M.Ni, "Deadlock-Free Multicast Wormhole Routing in 2D-Mesh Multicomputers," *IEEE Trans. on Parallel and Distributed Systems,* Vol. 5 pp.793-804, Aug. 1994.

[2] R. Kesavan, D. K. Panda, "Multiple Multicast with Minimized Node Contention on Wormhole k-ary n-cube Networks," *IEEE Trans. on Parallel and Distributed Systems*, Vol. 10, No.4, April 1999.

[3] J. Duato, "A Necessary and Sufficient Condition for Deadlock-Free Adaptive Routing in Wormhole Networks," *Int'l Conf. on Parallel Processing,* Vol.1, pp.142-149, 1994.

[4] H. Wang, D. M. Blough, "Construction of Edge-Disjoint Spanning Trees in the Torus and Application to Multicast in the Wormhole-Routed Networks," *Proc. of Int'l Conf. on Parallel and Distributed Computing Systems*, pp. 178-184, 1999.

[5] R. Libeskind-Hadas, D. Mazzoni and R. Rajagopalan, "Tree-Based Multicasting in Wormhole-Routed Irregular Topologies," *Proc. of Int'l Conf. on Parallel and Distributed Computing Systems*, pp. 244-249. 1998.

[6] D. K. Panda, S. Singal, and P. Prabhakaran, "Multidestination Message Passing Mechanism Conforming to Base Wormhole Routing Scheme," *Proc. of Parallel Routing and Communication Workshop,* pp. 131-145, May 1994 or *Ohio State University Technical Paper OSU-CISRC-6/94-TR33*, 1994.

[7] X. Lin, P. K. McKinley, and A. Esfahanian, "Adaptive Multicast wormhole routing in 2D Mesh Multicomputers," *Proc. Parallel Architectures and Languages Europe* pp. 228-241, 1993.

[8] Anjan K.V., T. M. Pinkston, J. Duato, "Generalized Theory for Deadlock-Free wormhole routing and its Application to Disha Cuncurrent, " *10th Int'l Parallel Processing Symposium.* April 1996.

[9] T. M. Pinkston, "Flexible and Efficient Routing Based on Progressive Deadlock recovery," *IEEE Trans. on Computer* Vol. 48, No7, pp. 649-669, July 1999.

정 종 인

1981년 경북대학교 전자공학과(전산전공) 학사. 1985년 경북대학교 전자공학과(전산전공) 석사. 1995년 서강대학교 전자계산학과 박사. 1999년 ~ 2000년 USC Dept. of EE post-doc. 1985년 ~ 1997년 우송공업대학 전산과 부교수. 1997년 ~ 현재 공주대학교 컴퓨터교육과 부교수. 관심분야는 병렬처리구조, 웹프로그래밍, 네트워킹, 신경회로망

김 인

1990년, 1992년 KAIST 학, 석사. 1993년 ~ 1995년 LG반도체 근무. 1995년 ~ 현재 USC Dept. of EE 박사과정.

김 시 관

1982년 경북대학교 전자공학과(전산전공) 학사. 1984년 KAIST 전산학부 석사. 2000년 KAIST 전산학부 박사. 1984년 ~ 1995년 삼성전자, LG정보통신 근무. 2001년 ~ 현재 안동대학교 멀티미디어학과 전임강사. 관심분야는 멀티미디어 시스템, 무선 컴퓨팅 및 라우팅 알고리즘