A Ring Based Multicast Service over a High Speed Deflection Network

Elena Pagani Gian Paolo Rossi

Dipartimento di Scienze dell'Informazione
Università degli Studi di Milano
Via Comelico 39, 20135 Milano, Italy

Abstract

A simple ring support structure has been used to provide multicast service over a prototype of high speed network that uses deflection routing *on mesh topology. By exploiting the behaviour of Deflection Networks it is possible to ensure, with a simple protocol, a significant performance improvement with respect to n-unicast communications and to obtain results that are close to those achievable with tree based multicasting. Further, the devised multicast protocol has a simple implementation that does not affect the ongoing prototype development. The paper shows that value added services can be provided on top of datagram Deflection Networks to extend the LAN services to larger geographical areas.*

1 Introduction

Multicasting, i.e. the transmission of a packet to a group of network nodes, is a well supported service by local and metropolitan area networks. Multicast capability on a datagram subnetwork is recognized as an important facility because it can provide efficiency with respect to *n-unicast* communications and it hides multiple destination addressing from the upper entities. Further, group communications are widely used by modern multimedia and distributed applications. These applications are forcing the development of high speed communication networks that extend the LAN services to larger geographical areas. In recent studies, networks that use *deflection routing* on mesh topologies, have gained great attention [1, 2] and have been considered as being an effective and practical solution to this need.

Deflection Networks, henceforth also referred to as DNs, naturally provide datagram service on point to point communications. This paper aims at introducing a simple solution to improve the basic DN service to support multicasting without affecting the architecture of the DN prototype that is currently under development in the frame of a national project [3]. The paper shows that a simple multicast protocol, based on ring support structure and exploiting the nature of DN, guarantees a significant improvement of performances with respect to *n-unicast* communications and obtains results that are close to those achievable with tree based multicasting. The results we present are encouraging because they indicate that value added services can be provided on top of datagram Deflection Networks and that they are a practical solution to extending the LAN services to larger geographical areas. An overview of the adopted solutions is given together with the encountered problems and the first simulation results.

The paper is organized as follows: in Section 2 we describe the basic Deflection Network behaviour and the advantages of the approach. In Section 3 we outline the applicable multicast model and in Section 4 we introduce the followed approach. Section 5 describes the multicast protocol, while simulation results and compared evaluations are given in Section 6.

2 Deflection networks

In this Section, we briefly describe the behaviours of basic Deflection Networks, or DNs, that will be considered throughout this paper. Each DN node has the number of input links that equals the number of output links. Links have the same transmission speed.

For every input link, I/O operations are performed as those usually encountered in slotted rings. A packet addressed to the node is extracted. A packet in the local user queue enters the network if an empty slot is available, i.e. a slot has been received empty or a packet has been extracted.

Switching and transmission processes proceed in time slots. At a locally synchronized instant, each node switches to the preferred link the packets it has in input buffers according to addressing and local routing information. Should two or more packets be routed to the same link, they collide. Collided packets are *deflected* over nonpreferred links and they are not locally buffered. Deflections may waste the network capacity and reduce the maximum throughput attainable under ideal conditions. However, it has been shown that this waste can be very limited and that its impact decreases as the network size increases [2]. The network behaviour is stable under any load, i.e. the throughput cannot collapse due to the saturation of some resources. In fact, packets cannot be accumula-

*This work has been carried out under financial support of CNR, the National Research Council, Telecommunication Project, 1993.

0-8186-6430-4/94 $4.00

ted into nodes nor be blocked waiting for the availability of transmission buffers. Packets are kept travelling in the network until they reach their destination.

The routing function assumes that nodes know their shortest-path distances to all possible destinations and Backward Learning technique, [4], is used to dynamically estimate the required distances.

The attractive potential features of DN have been described in previous works, see for instance [1, 3], and have led the Italian National Research Council to found a project to build an experimental network able to provide an aggregate throughput of tenths of Gb/s.

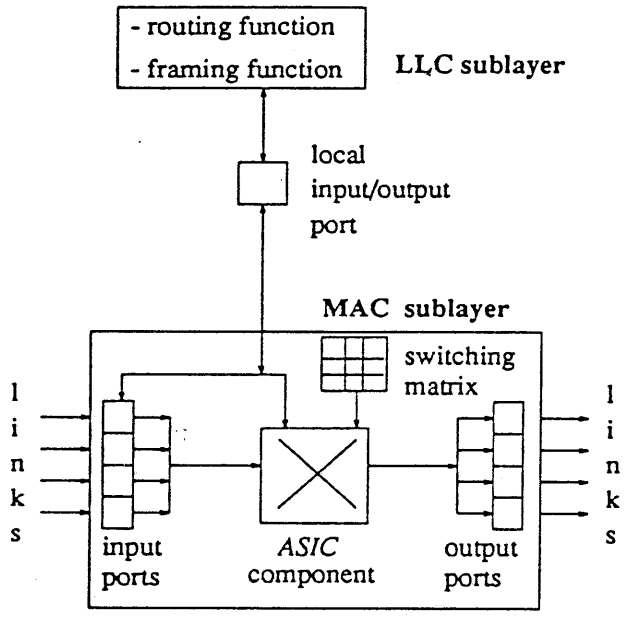

Figure 1: Architecture of a DN node.

In the design [5] the use of commonly available technology and off-the-shelf components has been assumed. As a result, the overall architecture of the prototype is described in Figure 1 and combines:

1. a single Application Specific Integrated Circuit (ASIC chip, 1.2 technology, *CMOS*, 8 × 8 mm^2 die size), that implements the access control and the switching functions according to the described deflection mechanism. Furthermore, when a packet reaches the destination this component extracts it, thus releasing the network resources, and passes the packet to the upper layer. In the frame of a functional protocol architecture, this set of operations may be ascribed to a Multiple Access Control (MAC) sublayer;

2. a microcontrolled node (based on Intel i960 microcomputer) that implements the addressing function and the routing policy, and may be functionally assimilated to a Logical Link Control (LLC) sublayer.

Four bidirectional 100 Mbit/s optical links per node are available.

The multicast protocol we are describing should fit with the described architecture.

3 Multicast model

In this Section, we define the problem to be solved and outline the applicable model. Let's consider a set of network nodes $S = \{n_1, n_2, \ldots, n_k\}$ interconnected by a mesh topology and using the protocol described in Section 2 to access the network resources; the multicast model is composed of a subset $G \subseteq S$ of nodes on which one or more processes, belonging to some distributed application that requires group organization, are executed. Nodes in G may fail crash (fail stop failure) either before or while transmitting to the output links the packets that are contained in the input ports. Crash failures produce total or partial loss of the packets passing by. Packet loss or omission is also possible due to link failures or node omissions.

We consider groups G that have members only on a subset of the network nodes, possibly separated from one another, and that can be long lived. We assume that nodes may be added to and removed from the group G according to the application requirements or to network failures. In the literature, this type of group is often referred to as *sparse* group [6].

Sparse groups fit with the requirements of many modern group applications such as cooperative work, conferencing and those operating on replicated data objects. To support these applications, the DN multicast facility we are describing aims to provide:

1. efficient delivery of multicast packets through the DN subnetwork;

2. multicast addressing facility that hides from the higher entities the mapping of group names into node addresses. We assume that each node i has a node identifier, nid_i, that uniquely identifies it and that the multicast identifier, mid_G, uniquely identifies the set $G = \{nid_1, nid_2, \ldots, nid_k\}$. We also assume that the group naming is coordinated by upper management functions according to application specific needs. As a consequence, we define that only one among the group nodes (the *initiator*) initiates the procedure to set up the group at the subnetwork level;

3. the mechanism to deal with dynamic changes of the group membership;

4. simple solution that integrates easily into the ongoing prototype development.

Further, we require that the following properties are satisfied:

Agreement. All the active, i.e. noncrashed, nodes of a group G know the same set of nodes in G. If a node crashes then the active nodes agree on the new group composition within a bounded time.

Availability. The multicasting of packets to the group continues despite the crash failures.

The above fault tolerance clauses require that the group members maintain consistent information about the group composition in spite of failures. These are strong requirements that are rarely provided at the subnetwork level. However, higher layer multicast algorithms, e.g. those providing reliable group communications [7, 8], have to cope with both crash and omission failures by adopting proper fault tolerance mechanisms. This class of protocols would yield great efficiency benefits by exploiting the service provided by an underlying, hardwarized, protocol that ensures the packet delivery by identifying and recovering crash failures. Packet loss and omission are supposed to be recovered by higher layer protocols.

The protocol we present can support the application processes grouped according to different organizations: client server, diffusion, hierarchical and so on [7]. In most of these cases, a mechanism to establish the path that links the external client to the group of servers, is required. In the sequel, we consider *peer* groups, i.e. groups in which processes cooperate through peer to peer communications to perform a common task. This simplifies the protocol description by eliminating the architectural aspects related to the communication among clients and servers.

4 Multicast support structure

The multicast problem has been widely investigated in the past. In [9] a good review can be found. By observing the existing literature on the topic, it can be argued that the best results are achieved when the multicast packets follow a tree support structure that links together the destination nodes within a group (see for instance [10, 11]).

Multicast communications that exploit tree support structures take advantage of the use of separated paths from source to multiple destinations thus obtaining a high parallelism in the packet delivery. This can be easily achieved by store and forward switching nodes where one multicast packet entering the node is duplicated and the copies can be enqueued to the output ports corresponding to the subtree branches. Unfortunately, this basic need is not achievable on a Deflection Network, in which packets cannot be stored in the node and the packets available at the input ports must be accommodated in output ports according to the described mechanism. A hole is created only when a packet is extracted because it reached its destination.

In this case, only linear topologies, namely ring or bus topologies, may provide the multicast support that gracefully coexists with DN characteristics. The ring topology has been used for our purposes because it is unidirectional and it can be easily adapted to a point to point subnetwork. Although the derivation of the minimum cost ring is a NP-complete problem, see for instance [12], several heuristics may be followed to obtain practical solutions. The approach we followed has the advantage of simplicity and allowed quick investigation of the capability of satisfying the fixed requirements and the evaluation of the behaviours of the ring based approach.

We used simulations to compare and to observe the behaviours of three multicast policies: *n-unicast*, *tree* and *ring* based. We analyzed the mean delay D, i.e. the average elapsed time that is computed from the time a multicast packet is generated by the user-entity to the time it is delivered to the group members, and the number of hops required to perform the delivery process. While the former is a performance index, the latter is a measure of the amount of network resources the service utilizes; this describes the effects that the multicast traffic has on other types of traffic, i.e. datagram and circuit [13, 14], that are competing to access the network resources (remind that in a DN the longer a packet keeps travelling in the network the longer it steals resources from other packets that wait in the input queues).

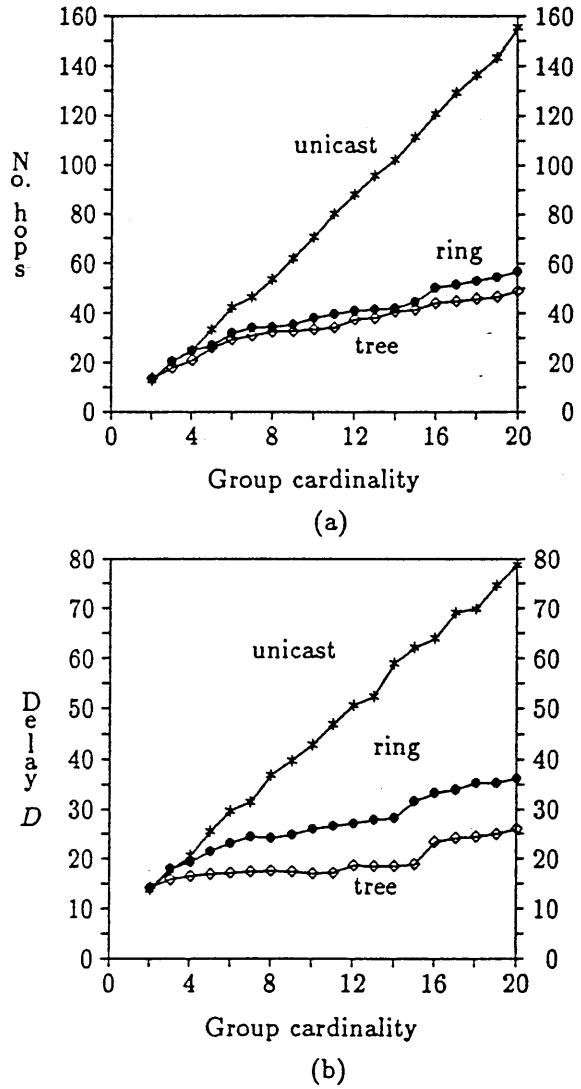

Figure 2: (a) No. hops vs. group cardinality. (b) Delay D (in network slots) vs. group cardinality.

Our simulations consider a 10×10 torus DN, a single multicast packet to send and uniformly distributed background datagram traffic sufficient to saturate the

network (rate 0.6 *pck/slot*). As shown in Figure 2(a), the ring based topology uses almost the same amount of network resources (in terms of network links) as the tree based one (simulations are based on optimal *ring* and *tree* topologies, i.e. they provided the shortest paths to the destinations). Both waste significantly less resources than unicast, for the same offered load. On the other hand, the tree topology performs better (Figure 2(b)) because it exploits parallel paths, while the ring has intermediate figures. The performance difference between ring and unicast grows together with the growing of the group cardinality. This latter aspect will be confirmed by other measurements and will be discussed in the next sections.

5 Multicast protocol

The purpose of this section is to describe the simple multicast protocol that has been devised to provide datagram multicast service by satisfying the requirements given in Section 3. The multicast protocol combines two sets of functionalities: *i*) management of the multicast virtual ring and *ii*) data transfer on the virtual ring.

The management of the ring includes the establishment and the maintaining of the multicast group ensuring the defined fault tolerance clauses. This is performed by the LLC entities which are responsible for addressing and routing, and have a microcontrolled implementation. Once the virtual ring has been established, the highest performances are achieved during the data transfer phase if the multicast packets are routed to sequentially visit the nodes of the group without being extracted from the DN, i.e. without releasing the accessed network resources. Lightweight operations should be performed by the MAC entities to switch multicast together with datagram packets. As a consequence, the following further functions are performed at the MAC sublayer; they allow that both virtual (ring) and real (mesh) topologies are handled at the same functional layer on the base of switching tables that are properly packaged by the upper entities. In particular:

1. forward multicast packets over the virtual ring topology,
2. perform local copy without extraction of a multicast packet when a node belonging to the group is visited,
3. extract the packet when all the active group members have been visited.

The resulting structure for a DN node is given in Figure 3 a), where the use of the *Multicast Packet Preprocessor* (MPP) allows to implement new multicast functions without affecting operations of the *ASIC* component. In Figure 3 b) the multicast packet format is also given. In the sequel, we provide a brief description of the solutions we adopted to perform the functions of both LLC and MAC sublayers.

A totally distributed mechanism allows to establish virtual ring topologies. Initially, only the *initiator* node knows the group composition and uses the mid_G

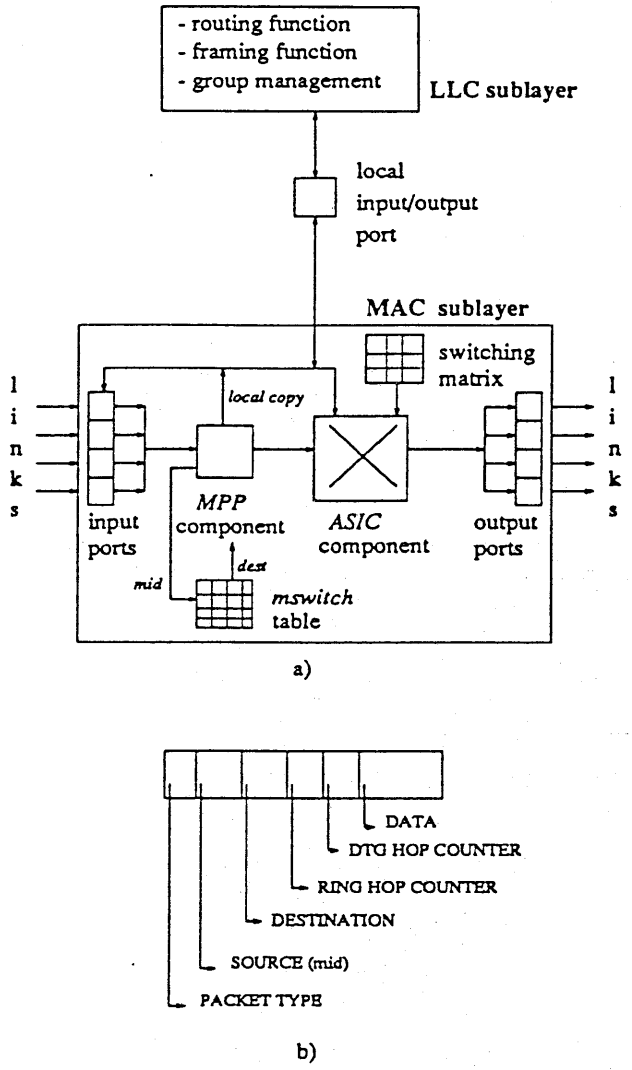

Figure 3: (a) Architecture of the DN prototype with multicast service, and (b) multicast packet format.

name to identify the group. A $SET\ UP$ control packet is passed among the group nodes. It contains the group composition, the mid_G and the set of the already *visited* nodes (initially none). Upon receiving the $SET\ UP$ packet a node decides, among the *non-visited* nodes, the downward node in the ring by selecting the closest one (in terms of amount of hops) according to local routing information. It constructs the local multicast switching table, called *mswitch* table and containing both the group cardinality and the network address of the downward node. Then, it forwards the $SET\ UP$ packet through the newly created virtual link of the ring. When the control packet returns to the *initiator* node the algorithm terminates.

By following this algorithm each node in G has a local and partial knowledge about the group state; indeed, a global knowledge might be required, e.g. to

ascertain whether or not the set-up procedure has properly terminated. A commitment mechanism allows the nodes to atomically agree on a common *group view* that describes the state of the nodes in G. The copies of the *group view* must be updated to consistently reflect the changes in the group composition, due to crashes or new entries. The next section describes the way the commitment is activated to cope with crash failures.

Once committed, a node provides the underlying entity with a copy of the *mswitch* table and activates the data transfer phase whose functionalities mainly involve operations of *MAC* entities.

Upon receiving a multicast packet (the *TYPE* field is inspected), the MPP of an entity belonging to the group uses the *SOURCE* field (that contains the mid_G name) to access the *mswitch* table from which the *nid* of the downward node in the ring is extracted. This address is put into the *DESTINATION* field, a local copy is performed and the packet is then passed to the ASIC component to be switched according to the DN rules. Since the virtual topology is *mounted* over the meshed one, between two consecutive nodes of a virtual ring a multicast packet is considered as being a datagram one. The *RING-HOP-COUNTER* field of the multicast packet allows one to count the amount of virtual hops the packet run on the ring. After $n-1$ virtual hops the packet is removed from the ring.

5.1 Tolerance to omission and crash failures

The multicast protocol we are describing can be affected by both omission and crash failures. Since the protocol provides basic datagram service, the omission failures involving multicast data packets are supposed to be recovered by the higher layer protocols. However, the loss or the corruption of control packets, e.g. those sent during the set up phase, may damage the protocol. In the sequel, we consider only the omission failures that hit control packets. Furthermore, crashes interrupt the normal circulation of packets on the ring. Since a crash failure modifies the *group view* that the nodes in G are considering, whenever a crash occurs the faulty node should be identified and the remaining active nodes should recover by agreeing on a new *group view*. The problem is to properly distinguish among omission and crash failures.

The most common way to recover from omission failures is the use of acknowledgements and retries. Each control packet is acknowledged and after k consecutive faulty attempts the sender is authorized to suspect a crash. In this case, the commitment protocol can be executed to atomically decide on the new group composition and restart the operations.

This is the simple schema that we used to handle the multicast group on the ring topology. In fact, acknowledgements are automatically obtained through the complete circulation of the control packets. They are routed through the ring according to the described mechanism and are extracted by the source node. Recalling that multicast data packets are extracted after $(n-1)$ hops for efficiency purposes, the complete circulation is simply achieved by initializing the hop counter field, within the header of multicast control packets, to 0 instead of 1.

The value of k is fixed according to the observed error rate. Two approaches are commonly followed:

1. Optimistic: a node sequentially uses the k attempts to circulate a control packet, i.e. only if the first attempt fails the protocol resorts to the second and so on;

2. Pessimistic: the concept of redundancy is used. k control packets are sent one after another. Extra packets are ignored.

The first approach is more conservative and may generate long waiting times. The second may load the network with useless traffic, while it accelerates the decision process. We adopted this latter approach to grant a privilege to efficiency since the number of control packets is very limited.

The described mechanism is performed at *LLC* layer; during the set-up phase it is performed by the *initiator* node and during the data transfer by all the active members of the group that autonomously execute periodical (e.g. after a certain amount of data packets of after a given time) ring monitoring.

If a node has a high multicast traffic to generate it can also piggyback control information on data packets. When a suspect is raised, the commitment should be activated.

The description of the commitment protocol is outside the scope of this paper. We followed a simple centralized approach; when a node suspects a crash it starts a voting algorithm (the *bully* algorithm has been used in our simulation program [15]) and then the elected node executes a three way handshake protocol.

6 Analysis of the multicast protocol

As soon as we realized that the ring was the only affordable support structure for multicast service, we suspected poor ring performances with respect to *n-unicast* delivery. In fact, the lack of parallelism can affect performances much more than the simple algorithm we used to establish a non-optimal ring. Because of the nature of DN, this contrasts with the results we observed by means of simulations. In fact, a node using unicast to perform multicast communications generates as many packets as many nodes are in the group, while a single packet is required to circulate over the ring. Given that DN does not provide internal buffering, the n-unicast packets are likely to spend their time in the input queue waiting for an empty slot to pass by. The higher the network load is and the larger the multicast message (that requires fragmentation), the longer is the delay the unicasts suffer, thus wasting the benefits of parallel paths.

We used simulations to observe the multicast delay D for both n-unicast and ring based protocols. Simulations consider a 10×10 torus DN with background datagram uniformly distributed traffic and user multicast messages that are fragmented into 50 multicast

packets. In Figure 4 the Delay versus the group cardinality is reported for different background traffic rates: 0.4 *pck/slot*, (a), and 0.2 *pck/slot*, (b). In line with the above considerations, unicast suffers from both the network load and the group size; the ring based multicast generates a single block of 50 packets. The multicast flow may be affected by the number of ring hops to perform, but has limited problems to access network resources.

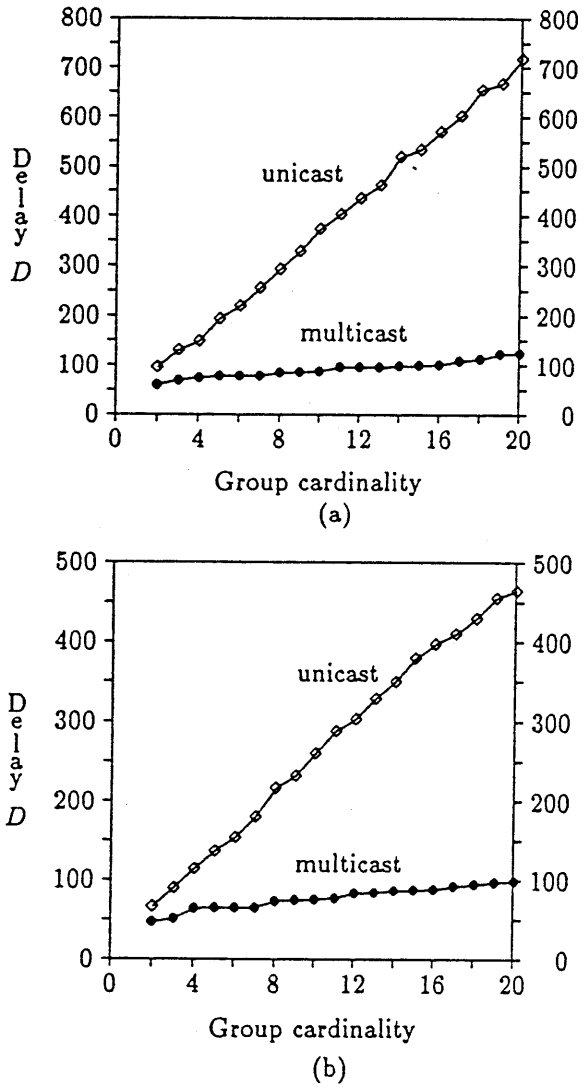

Figure 4: Delay vs. group cardinality. (a) datagram traffic rate of 0.4 *pck/slot* and (b) 0.2 *pck/slot*. 10×10 torus DN, 50 packets per multicast message.

The capability of performing multicasting at DN level has clear side effects on the end to end throughput S_u that higher layer multicast entities may observe. We have constructed a protocol stack by installing on top of DN a datagram transport protocol, which is responsible for packet segmentation and assembling, and a reliable multicast protocol, that provides atomic and ordered group communications [8]. By considering a group of user entities of this protocol stack, we measured the achievable end to end throughput S_u for different group cardinalities and for both multicast policies. As shown in Figure 5, the ring based multicast provides higher values of S_u because it is less sensitive to the message size (analogous results have been obtained by varying the group cardinality).

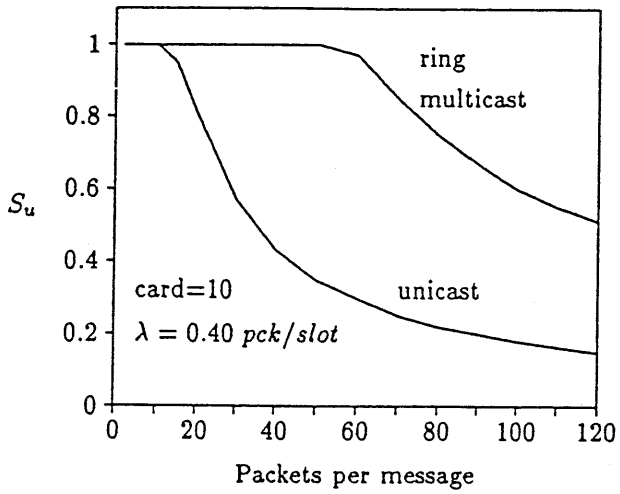

Figure 5: High level multicast throughput S_u vs. size of multicast message.

The ring based multicasting provides encouraging results. However, they are obtained against a reduced fairness degree of the system. The capability of accessing a DN under saturation traffic conditions depends on the availability of free slots that a node obtains by extracting received packets. Possible node lock out has been observed and discussed in [1], where hot spot traffic was considered. The problem is not related to multicasting but it rather depends on the traffic composition over a DN. When multicast traffic is activated over a network with uniformly distributed datagram traffic two effects can be observed:

1. the nodes belonging to the group extract on average more packets than nodes outside the group;
2. datagram packets are likely to measure a longer path to get to destinations because packets are scattered apart from the whirl created by the multicast flow. As a consequence, multicast packets have a certain implicit priority when colliding with datagram packets because of their shorter mean path to destination.

Both events favour the nodes on the ring and those close to it, while the other nodes observe some unfairness as shown in Figure 6, where the datagram throughput S_D of each of the 10×10 torus DN nodes is reported.

A booking mechanism, capable to drain empty slots from surrounding nodes, has been proposed in [1] to cope with unfairness conditions. A simple evolution of that protocol is currently under experimentation.

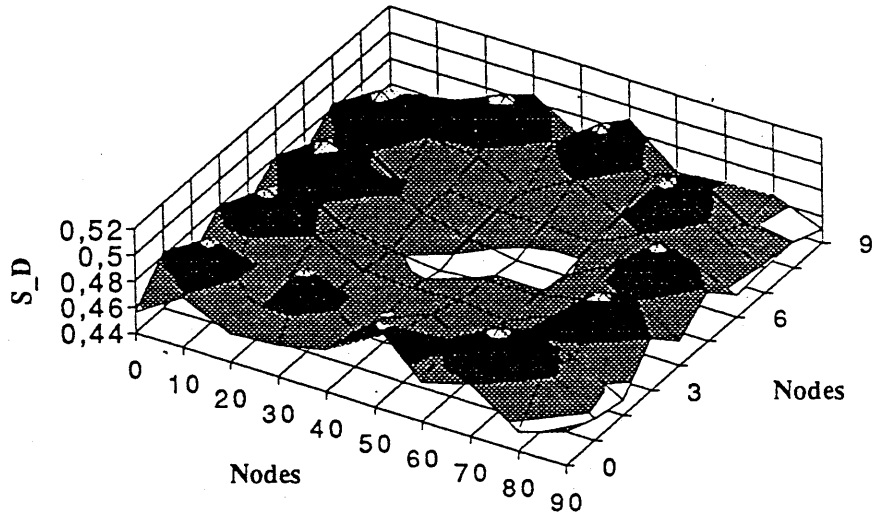

Figure 6: Datagram throughput S_D for each node of a 10 × 10 torus DN.

7 Concluding remarks

This paper provides a simple solution to improve the point to point nature of Deflection Networks, thus allowing one to support multicast communications without affecting the architecture of the prototype that is currently under development. It has been shown that a simple multicast protocol, based on a ring support structure, guarantees a significant improvement of performances with respect to *n-unicast* communications and obtains results that are close to those achievable with tree based multicasting. The results we present are encouraging to continue the activity with the aim to improve some adopted algorithms, namely the ring Set Up procedure and the algorithms that have been used to support the commitment and the election of coordinator. Further work is also needed to cope with the unfairness problems that have been observed through simulations.

References

[1] F. Borgonovo, L. Fratta, *Deflection Networks: Architectures for Metropolitan and Wide Area Networks*, Computer Networks and ISDN Systems, Vol. 24, 1992, pp. 171-183.

[2] F. Borgonovo, E. Cadorin, *Locally-Optimal Deflection Routing in the Bidirectional Manhattan Network*, IEEE INFOCOM '90, S. Francisco, CA, June 1990.

[3] G. Albertengo, F. Borgonovo, L. Fratta, G.P. Rossi, *Deflection Network: design, implementation and performance issues*, CNR-PFT Symp. on High Speed Networks, Roma, March 1993.

[4] P. Baran, *On Distributed Communications Networks*, IEEE Transactions on Communication Systems, Vol. CS.12, March 1964, pp. 1-9.

[5] G. Albertengo et al.,*Deflection Network: principles, implementation and services*, European Transaction on Telecommunications and Related Technologies (ETT), Vol. 24, No. 2, April 1992, pp.14-17.

[6] S.E. Deering, D.R. Cheriton, *Multicast Routing in Datagram Internetworks and Extended LANs*, ACM Transactions on Computer Systems, Vol. 8, No. 2, April 1990, pp. 85-110.

[7] K. Birman, A. Schiper, P. Stephenson, *Lightweight Causal and Atomic Group Multicast*, ACM Transaction on Computer Systems, Vol.9, N.3, August 1991, pp. 272-314.

[8] R. Aiello, E. Pagani, G.P. Rossi, *Design and Implementation of a Reliable Multicast Protocol*, Proc. of IEEE INFOCOM '93, San Francisco, March 1993, pp. 75-81.

[9] A.J. Frank, L.D. Wittie, A.J. Bernstein, *Multicast Communications in Computer Networks*, IEEE Software, Vol. 2, No. 3, May 1985, pp. 49-61.

[10] N.E. Belkeir, M. Ahamad, *Low Cost Algorithms for Message Delivery in Dynamic Multicast Groups*, Proc. 9th IEEE Int. Conf. on Distributed Computing Systems, 1989, pp. 110-117.

[11] P. Winter, *Steiner Problem in Networks*, Networks, Vol. 17, 1987, pp. 129-167.

[12] K. Mehlhorn, Data structures and algorithms 2. Graph algorithms and NP-completeness, Springer Verlag, 1984.

[13] F. Borgonovo, L. Fratta, F. Tonelli, *Circuit Service in Deflection Networks*, Proc. of IEEE INFOCOM '91, Bal Harbour, Miami, FL, USA, April 9-11, 1991.

[14] E. Pagani, G.P. Rossi, *Providing Circuit Service over a High Speed Deflection Network*, Proc. EUROMICRO'93, Barcelona, Sept. 1993.

[15] H. Garcia Molina, *Elections in Distributed Computing Systems*, IEEE Trans. on Computers, C-31, No.1, January 1982.