IP 망에서의 이동 노드를 위한 향상된 Xcast 프로토콜☆

An Enhanced Xcast Protocol for Mobile Nodes in IP Networks

남 세 현*
Sea hyeon Nam

요 약

Mobile IP를 기반으로 하는 기존의 멀티캐스트 방식은 대규모 멀티캐스트 그룹을 제한된 개수만큼만 지원할 수 있는데 반하여, Xcast는 소규모 멀티캐스트 그룹을 여러 개 지원하는 것이 가능한 방식이다. 하지만, Xcast에서는 송신 노드가 Xcast 헤더에 수신 노드들의 주소를 나열하는 방식을 사용하기 때문에 망에서의 최대 패킷 크기에 따라 하나의 Xcast 패킷이 갖는 수신 노드 주소의 개수에 제약을 받는다. 본 논문에서는 이동 노드들에게 멀티캐스트 서비스를 제공할 때 기존의 Xcast 프로토콜이 갖고 있는 멀티캐스트 그룹 크기의 제약을 해결하기 위한 향상된 Xcast 프로토콜을 제안한다. 또한, 제안된 멀티캐스트 방식은 SIP과의 통합을 통하여 응용 계층에서 이동성을 인식하도록 한다. 시뮬레이션 결과를 통하여 제안된 멀티캐스트 방식은 망에서의 패킷 전달율과 패킷 전송 효율을 높일 수 있을 뿐만 아니라 패킷의 지연도 크게 낮출 수 있음을 알 수 있다.

Abstract

Whereas the traditional multicast schemes based on Mobile IP can support a limited number of very large multicast groups, the Xcast protocol can support a very large number of small multicast groups. In the Xcast, the source node encodes the list of destinations in the Xcast header. Therefore, the maximum packet size in the network limits the number of destinations that a Xcast packet may have. In this paper, an enhanced Xcast protocol is proposed to solve the multicast group size limitation of the existing Xcast protocol in providing multicast service for mobile nodes. Moreover, the SIP (a very flexible control plane protocol) is integrated with the proposed multicast scheme to provide mobility awareness on the application layer. The simulation results verify that the proposed multicast scheme not only increases the packet delivery ratio and the data packet forwarding efficiency but also achieves low latency of packets in the network.

☞ Keyword : explicit multicast, session initiation protocol, mobile IP

1. Introduction

Recently, mobility support for Internet access has created significant interest among researchers as wireless/mobile communications and networking proliferate, especially boosted by the widespread use of laptops and handheld devices. IP multicast, the ability to efficiently send data to a group of destinations, is becoming increasingly important for applications such as IP telephony and multimedia conferencing. Providing multicast support for mobile nodes in an IP internetwork is a challenging problem and the solutions rely on the underlying mechanism of mobility support.

Currently, there are two basic approaches to support mobility in IP networks. The first one seeks to solve the mobility problem in the network layer by using Mobile IP[1] and related proposals. The multicast schemes over Mobile IP and related proposals include remote subscription, bi-directional tunneling, MoM[2], RBMoM[3], and so on. All of these schemes re-

* 정 회 원 : 대구대학교 정보통신공학부 교수
shnam@daegu.ac.kr(제 1저자)

☆ This paper was supported in part by the Daegu University Research Grant.
[2004/07/26 투고 - 2004/10/19 심사 - 2005/01/25 심사완료]

ly on two components: the host group model and the multicast routing protocol. In the host group model, a group of hosts is identified by a multicast group address, which is used both for subscriptions and forwarding. The multicast address allocation leads to quite complex procedures and introduces additional state information in the network. The multicast routing protocol is required to maintain the member state and the multicast delivery tree. Those traditional multicast schemes were designed to handle very large multicast groups. These work well if one is trying to distribute broadcast-like channels all around the world. However, they have scalability problems when there is very large number of small groups. In addition, for delay-sensitive multimedia applications, Mobile IP has some limitations, including triangle routing, triangle registration, encapsulation overhead, and the need for home addresses.

The other approach is to solve the mobility problem in the application layer by using Session Initiation Protocol (SIP) [4]. The SIP is an application layer protocol used for establishing and tearing down multimedia sessions, both unicast and multicast. It has been standardized within the Internet Engineering Task Force (IETF) for the invitation to multimedia conferences and Internet telephone calls. Three types of multiparty sessions can be supported by the SIP: full mesh, mixer, and network-layer multicast. Both full mesh and mixer are not true multicast schemes. They deliver datagrams by using multi-unicasting. The SIP should rely on the host group model and the traditional multicast routing protocols to support network-layer multicast. Thus, this solution shares the same disadvantages as the Mobile IP in providing multicast service for mobile nodes.

Explicit multicast (Xcast)[5] is a new scheme for Internet multicast that complements the traditional multicast schemes. In the Xcast, the source node keeps track of the destinations in the multicast session, encodes the list of destinations in the Xcast header, and then sends the packet to a router. Each router along the way parses the header, partitions the destinations based on each destination's next hop, and forwards a packet with an appropriate Xcast header to each of the next hops. Whereas the traditional multicast schemes can support a limited number of very large multicast sessions, the Xcast can support a very large number of small multicast sessions. Unlike traditional multicast schemes, the Xcast does not specify a "control plane". It relies on neither IGMP nor multicast routing protocols to support multicast. With Xcast, the means by which multicast sessions are defined is an application level issue. In general, xcasting is more efficient than unicasting because a single packet can hold data intended for multiple destinations, but it is less efficient than traditional multicasting because the addresses of these multiple destinations must be encoded in each packet, and the maximum packet size in the network limits the number of destinations that a packet may have. Therefore, xcasting was considered as an alternative solution to traditional multicasting that excels at handling data being sent to small number of destinations.

In this paper, an Enhanced Xcast (EXcast)

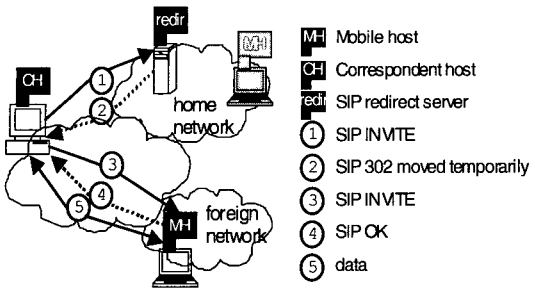
protocol is proposed to solve the multicast group size limitation of the existing Xcast protocol. In addition, the SIP (a very flexible control plane protocol) is integrated with the proposed multicast scheme to provide mobility awareness on the application layer. In the proposed EXcast scheme, as data packets with explicitly encoded destination addresses are routed through the network, each node along the forwarding path remembers the destinations to which it forwarded the last time and how the data was forwarded (i.e., which next hop was used for each destination). By caching this information, the proposed EXcast scheme no longer needs to list all the destinations in every data packet header. When changes occur in the underlying unicast routing or destination list, an upstream node only needs to inform its downstream neighbors (i.e., its next hops) regarding the differences in destination forwarding since the last packet. Reporting only these differences significantly reduces Xcast header sizes. Ideally, in a stable network where the topology and membership remain unchanged, only the first data packet needs to contain destination addresses and all subsequent packets would contain no destination information. In practice, the state kept at each node along the forwarding path is "soft". Each time data forwarding occurs, this state is refreshed.

The remainder of this paper is divided as follows. A brief description of the previous work on the mobility support by the SIP is presented in Section 2. The proposed multicast scheme for mobile nodes is presented in Section 3. The discrete-event simulation model to evaluate the performance of the proposed scheme and the comparative simulation results and discussions are presented in Section 4. Finally, Section 5 presents the conclusions.

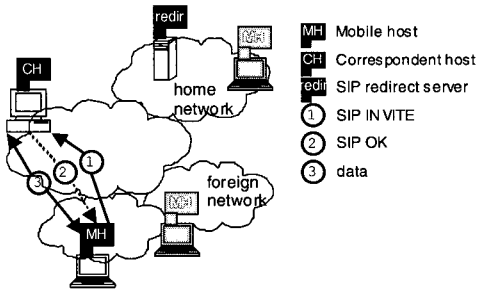
2. SIP Mobility Support

Entities in the SIP are user agents, proxy servers, and redirect servers. The SIP user agent has two basic functions: listening to the incoming SIP messages, and sending SIP messages upon user actions or incoming messages. The SIP proxy server relays SIP messages, so that it is possible to use a domain name to find a user, rather than knowing the IP address or name of the host. A SIP proxy can thereby also be used to hide the location of the user. On the other hand, the SIP redirect server returns the location of the host rather than relaying the SIP messages. This makes it possible to build highly scalable servers, since it only has to send back a response with the correct location, instead of participating in the whole transaction. The SIP redirect server has properties resembling those of the home agent (HA) in Mobile IP with route optimization, in that it tells the caller where to send the invitation. Both the redirect and proxy servers accept registrations from users, in which the current location of the user is given. The location can be stored either locally at the SIP server, or in a dedicated registrar.

One of the central tasks of the SIP is to locate one or more IP addresses where a user can receive media streams, given only a generic, location-independent address identifying a domain. This mechanism makes it easy to offer pre-call mobility. The mobile host (MH) simply re-registers with its home registrar each time it obtains a new IP address. When the co



〈Fig 1〉 SIP pre-call mobility.



〈Fig 2〉 SIP mid-call mobility.

rrespondent host (CH) sends an INVITE message to the MH, the SIP server can redirect (or relay) the INVITE message (see Fig. 1).

If the MH moves during a session (mid-call mobility), the moving MH sends another INVITE request to the CH using the same call identifier as in the original call setup (see Fig. 2). It should put the new IP address in the "Contact" field of the SIP message, which tells the CH where it wants to receive future SIP messages. Finally, the MH should update its registration at the home SIP server, so that new calls can be correctly redirected.

3. Proposed Multicast Scheme

Since the traditional multicast routing protocols impose limitations on the number of groups and the size of the network in which they are deployed, they have scalability problems. By contrast, the Xcast eliminates the membership management and routing information exchange in the intermediate routers, so that it can support very large numbers of small multicast groups. However, since the destination addresses must be explicitly encoded in each Xcast data packet, the maximum packet size in the network limits the number of destinations that a Xcast packet may have. In this section, the EXcast protocol which solves the maximum group size limitation of the existing Xcast protocol is described. For real-time communications such as IP telephony, multimedia conferencing, collaborative applications, and networked games, there are typically very large numbers of small to medium size multicast groups. For real-time traffic, it is more common to use the Real-Time Transport Protocol (RTP) over UDP, and important issues are fast handoff, low latency, and high bandwidth utilization especially for wireless networks. Therefore, it is desirable to introduce mobility awareness on a higher layer, where we can utilize knowledge about the traffic to make decisions on how to handle mobility in different situations.

3.1 Membership management

For membership management, the SIP (a very flexible control plane protocol) is integrated with the proposed multicast scheme to provide mobility awareness on the application layer. Specifically, by using the SIP, a full or partial mesh of RTP sessions is established to provide connectivity among multicast members. If a full mesh of sessions is established, then

every hosts can send their messages to every other participants. Thus, every hosts can be a multicast sender of the group. This is useful for applications such as multimedia conferences and multi-player games. When a multicast application requires only one sender, a partial mesh that connects one sender to all receivers is established. By using the SIP, a host takes the initiative to set up sessions for multicast. When a MH (multicast sender or receiver) moves during a session, we do not need to reconfigure the multicast delivery tree or rely on the tunneling service between HA and foreign agent (FA). With the assistance of the SIP servers, sessions for multicast are created and maintained to support pre-call and mid-call mobility. The session states are kept in the hosts.

3.2 Forwarding computation

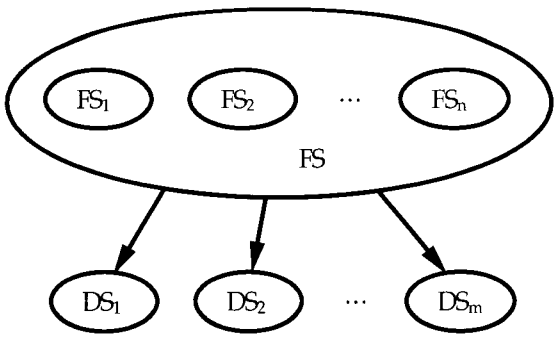
After a full or partial mesh of RTP sessions is established, EXcast protocol is used to deliver identical RTP datagrams sent from a sender to multiple receivers. As long as the current SIP/SDP syntax and semantics are used [6], one has to rely on the UDP-enhanced version of the Xcast with 18 bytes overhead (16 bytes for IPv6 address and 2 bytes for UDP port number) for each destination. When an application decides to use EXcast forwarding, it does not affect its interface to the SIP agent and it can use the same SIP messages as it would for multi-unicasting. Since the application in the sender host keeps track of the participants' unicast addresses, it is a simple matter to replace multi-unicast code of the SIP with EXcast code. All that the developer has to do is to replace a loop that sends a unicast to each of the participants by a single "EXcast_send" that sends the data to the participants.

EXcast packets contain a payload and an EXcast header. Three types of EXcast headers are defined: Empty (E) header, Refresh (R) header, and Difference (D) header. A D type can be either an incremental (D_i), or decremental (D_d). When there are no changes in the destination list, an E header is used. The E header does not include any destination address. When a new destination node is added to the forwarding path, a R header is used. The R header contains a destination list which is intended to replace the destination list on the downstream node. Each D header contains a list of destinations which describes the destination change since the last forwarding. Usually only one EXcast header is needed to describe the change in destination list for a downstream neighbor. However, when both D_d and D_i headers are needed to describe a difference, both headers are constructed and put into the header (i.e., put the D_d header immediately ahead of the D_i header).

At each node, there is one Forwarding Set (FS) for each active multicast session. It records to which destinations this node needs to forward multicast data. At the source node, the FS is the same as the all multicast members. At other nodes, this FS is actually the union of several smaller sets. Each small FS_k set records the destinations included in data packets received from upstream neighbor k. After receiving a data packet from a neighbor k, the receiving node will update the corresponding subset FS_k and then the un-

ioned overall set FS.

The overall FS set contains all the destinations to which this node need to forward. However, these destinations may be reached via different paths (i.e., next hops). Therefore, the FS needs to be partitioned into subsets according to the next hops. The destinations in the FS who use the same downstream neighbor (next hop) will be put into the same subset. These subsets resulted from partitioning the FS are called Direction Sets (a DS_l exists for each downstream neighbor l). For a pair of multicast tree neighbor u and d, the DS_d set on the upstream node u should contain the same list of destinations as the FS_u set on the downstream neighbor d. The data structures involved in the forwarding computation is shown in Fig. 3.



〈Fig 3〉 Forwarding computation data structure

When a node receives an EXcast packet from an upstream neighbor k, the receiving node accesses the FS_k set. If the received EXcast packet is an E type, there is no change since the last data forwarding. Therefore, the states left by last forwarding computation can be reused: the receiving node only needs to forward one copy for the data packet to each downstream neighbor l whose DS_l is not empty. If the received EXcast packet is a R type, the node replaces its FS_k by the destinations in the EXcast header. If a node receives a D_i or D_d type, it updates the corresponding FS_k according to the D type: removing the addresses in the D_d header from its FS_k and adding the addresses in the D_i header into its FS_k set.

After updating the FS_k, the receiving node updates the union FS. Then it partitions the new overall FS set into DS'_l sets according to the next hop l used by the unicast routes towards the destinations in the FS. All destinations in the same DS'_l share a common next hop l.

By comparing the contents of the new DS'_l set and the existing DS_l set (result of the forwarding of the last data packet) for the same next hop l, the node assembles new EXcast packet for next hop l. If there is one DS'_l but there is no matching DS_l, a R type EXcast packet is constructed. If the DS'_l set is the same as the DS_l set, an E type EXcast packet is used since there is no change. Otherwise, D type EXcast packet is constructed. All destination addresses that are in the DS'_l but not in the DS_l are included in the D_i header. All addresses that are in the DS_l but no longer in the DS'_l are included in the D_d header. Addresses that are common in both sets appear in neither header. The EXcast packet forwarding process is described in Fig. 4.

```
ProcessEXcastPacket(P) {
  header = P.header;
  if (header == E_type)
    New_FSk = Old_FSk;
  else if (header == R_type)
    create New_FSk;
  else if (header == Dd_type)
    New_FSk = remove addresses in Dd header from Old_FSk;
  else
    New_FSk = add addresses in Di header into Old_FSk;

  Update overall ForwardingSet;

  for each destination D in overall ForwardingSet {
    NextHop = FindNextHop(D);
    add D into NewDirectionSet[NextHop];
  }

  for each NewDirectionSet[i] {
    if (exists DirectionSet[i])
      create D or E header for nexthop i based on the difference between
      NewDirectionSet[i] and DirectionSet[i];
    else
      create R header for nexthop i;
    send away EXcastPacket;
  }
}
```

〈Fig 4〉 EXcast packet forwarding process.

4. Performance Evaluation

To evaluate the performance of the proposed EXcast protocol, relative to the existing unicast and the Xcast protocol, a series of simulations have been performed by using a dis crete-event simulation package AweSim v2.0 (Pritsker Corporation). The simulations involve a network with a simple tree topology (Fig. 5). With a tree topology, only one shortest path exists between any two nodes, so routing issues do not affect the simulation results. The tree consists of 85 network nodes where each internal node has four children. The 64 leaf nodes are considered as networks (or subnets) where stationary and mobile hosts are connec-ted. In the simulation, it is assumed that a multicast sender is a stationary host located at randomly selected network and multicast receivers are all mobile hosts. At the beginning of a simulation, each MH is connected to its home network. After receiving 10 multicast packets in a network, the MH moves to other network with probability 0.1. The foreign net-

works to visit are chosen equiprobably at random. Thus, the residency time for each visit to a home or foreign network is geometrically distributed. When a MH moves to a foreign network, it can receive the multicast packets by mid-call mobility mechanism of the SIP without the services of HA and FA. In the simulation, the multicast group size was varied from 1 to 35. The Xcast packets in the simulation can be addressed to at most 20 destinations, and they use 18 bytes to encode each destination address (16 bytes for an IPv6 address and 2 bytes for an UDP port). However, EXcast packets can be addressed to more than 20 destinations because an upstream node only needs to inform its downstream neighbors regarding the differences in destination since the last packet. Table 1 summarizes the main parameters used in the simulation experiments.

During a simulation run, one host on the network attempts to send a stream of data simultaneously to different destinations. Depending on the scenario being tested, the stream of data is sent using one of three distribution methods: multi-unicasting, xcasting, or enhanced xcasting (excasting). The data stream that is sent simultaneously to multiple destinations is a CBR stream of 200 byte UDP packets sent every 5 ms. The group of destinations to which the stream is sent does not change during the course of a simulation run. All links in the network have transfer rates of 1 Mbps and queues that can hold 50 packets. The simulation time for a run was selected as the time for a multicast sender to generate 20000 CBR packets for the multicast group.

The simulation measures the data packet delivery ratio, the latency of delivered packets, and the data forwarding efficiency in the network. Initially a single mobile destination for the data stream is randomly chosen, and the simulation is run. Then, an additional mobile destination for the stream is randomly selected, and the simulator is run again. This cycle continues until the stream is being sent to 35 mobile destinations. The measurements from ten such simulation sequences are averaged together to produce the final results.

Figure 5 shows the data packet delivery ratio which is defined as the ratio of the number of data packets actually received by group members to the number of data packets that should be received. Since the membership is static during the entire simulation, the number of data packets that should be received equals

〈Table 1〉 Simulation Parameters

Parameters	Value
Network nodes	85 (nodes)
Multicast sender	1 (stationary host)
Multicast receivers	1...35 (mobile hosts)
Moving probability to other networks	0.1 (after receiving 10 packets)
Xcast header limitation	20 (destinations)
Xcast header overhead	18 (bytes per destination)
Multicast data stream	200 (bytes per 5msec)
Link speed	1 Mbps
Queue size at the link	50 (packets)

the product of the number of group members and total number of data packets generated by the multicast source. Since the amount of data sent into the network is the same for the three different distribution methods, if the multi-unicasting and xcasting distribution methods result in lower data packet delivery ratio than excasting, the differences are due to lost packets. The excasting clearly dominates in its ability to deliver the stream data to a large number of destinations. By contrast, unicasting can not handle the number of packets generated by a data stream sent to more than three destinations. This sudden degrade of performance occurs because the majority of data stream packets must travel from the stream source to the center of the network in order to reach their destinations. If any link along this path becomes saturated, that link will act as a bottleneck and prevent any additional data stream packets from reaching their destinations. For xcasting, the network is able to achieve performance equal to that of excasting until the stream source needs to be sent to more than 20 destinations. At that point, the data packet delivery ratio abruptly decreases. This phenomenon occurs because Xcast packets cannot be addressed to more than 20 destinations, so a second Xcast packet needs to be used for the additional destinations. Unfortunately, the links along the critical path from the stream source to the backbone are already near saturation, and the extra Xcast packets can not be accommodated and are discarded.

Figure 6 shows the average latency of delivered packets in the network. Again, excasting exhibits the best performance. For xcasting, the delay increases slightly as the multicast group size increases. However, once the data stream is sent to more than 20 destinations, latency shoots up dramatically in networks with xcasting. This is because the extra Xcast packets needed when a stream is sent to more than 20 destinations clog queues and significantly increase delay. Although unicasting also experiences queuing delay when a stream is sent to more than 20 destinations, it shows lower overall latency than xcasting. This is because unicasting is only handling and thus delivering much smaller number of packets than xcasting.

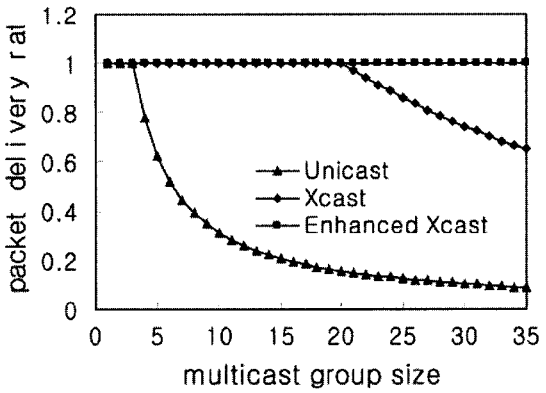


Fig. 5 Data packet delivery ratio.

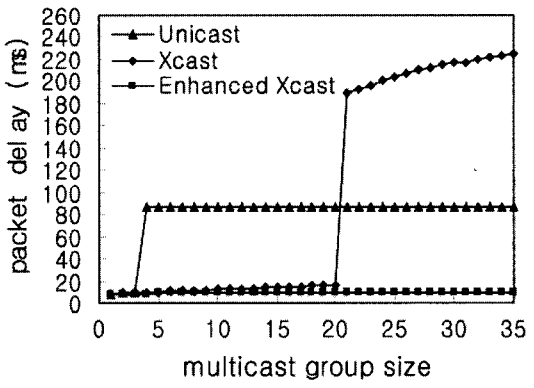


Fig. 6 Packet delay.

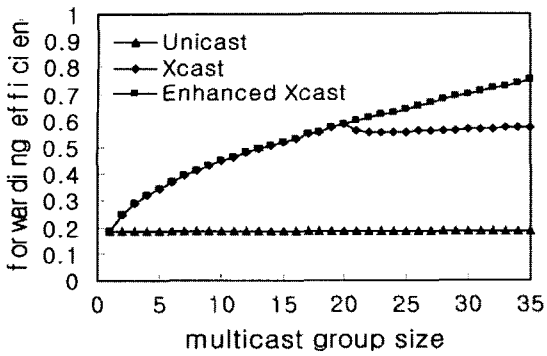


Fig. 7 Data packet forwarding efficiency

Figure 7 shows the data forwarding efficiency which is defined as the ratio of the number of data packets delivered to the number of data packet transmissions in the network links. The smaller this number is, the less efficient a protocol is. For example, when the multicast group size is three, the multicast source generates 20000 packets during the simulation time and three mobile hosts totally receive 60000 packets. If unicasting is used to deliver the multicast packets, about 326200 packet transmissions are required in the network links (data forwarding efficiency = 0.18). However, for xcasting and excasting, about 207800 packet transmissions occur in the network links to deliver the same packets (data forwarding efficiency = 0.29). In unicasting, the data forwarding efficiency is saturated when a stream is sent to more than three destinations. By contrast, in xcasting, the saturation occurs when a stream is sent to more than 20 destinations. However, for excasting, the data forwarding efficiency increases almost linearly as the multicast group size increases.

5. Conclusions

In this paper, an enhanced Xcast protocol was proposed to solve the multicast group size limitation of the existing Xcast protocol. In addition, the SIP (a very flexible control plane protocol) was integrated with the proposed multicast scheme to provide mobility awareness on the application layer. By using the SIP, a full or partial mesh of RTP sessions is established to provide connectivity between multicast source and members. After establishing connections for multicast, enhanced Xcast is used to deliver identical RTP datagrams sent from a sender to multiple receivers. In the proposed multicast scheme, as data packets with explicitly encoded destination addresses are routed through the network, each node along the forwarding path remembers the destinations to which it forwarded the last time and how the data was forwarded (i.e., which next hop was used for each destination). By caching this information, the proposed multicast scheme no longer needs to list all the destinations in every data packet header. Through the simulation study, it was verified that the proposed multicast scheme not only increases the packet delivery ratio and the data forwarding efficiency but also achieves low latency of packets in the network.

References

[1] C. Perkins, "IP mobility support," RFC 2002 (Proposed Standard), IETF, Oct. 1996.

[2] T. G. Harrison, C. L. Williamson, W. L. Mackrell, and R. B. Bunt, "Mobile Multicast (MoM) Protocol: Multicast Support for Mobile Hosts," *Proc. of ACM/IEEE Mobicom'97*, pp. 151-160, Sept. 1997.

[3] R. Lin and K. M. Wang, "Mobile Mult-

icast Support in IP Networks," *Proc. of IEEE Infocom 2000*, pp. 1664-1672, March 2000.

[4] M. Handley, H. Schulzrinne, E. Schooler, and J. Rosenberg, "SIP: session initiation protocol," RFC 2543 (Proposed Standard), IETF, March 1999.

[5] R. Boivie, Y. Imai, W. Livens, D. Ooms, and O. Paridaens, "Explicit Multicast Basic Specification," draft-ooms-xcast-basic-spec-04.txt, Jan. 2003.

[6] B. Doorselaer and D. Ooms, "SIP for the establishment of xcast-based multiparty conferences," draft-van-doorselaer-sip-xcast- 00. txt, July 2000.

◐ 저 자 소 개 ◑

남 세 현(Nam Sea hyeon)
1985년 연세대학교 전자공학과 졸업(학사)
1987년 한국과학기술원 전기및전자공학과 졸업(석사)
1991년 한국과학기술원 전기및전자공학과 졸업(박사)
1994년~현재 대구대학교 정보통신공학부 교수
관심분야 : 컴퓨터네트워크, 광통신망, 멀티캐스트 프로토콜
E-mail : shnam@daegu.ac.kr