A Service-Centric Multicast Architecture and Routing Protocol

Yuanyuan Yang, Jianchao Wang and Min Yang

***Abstract*—In this paper, we present a new multicast architecture and the associated multicast routing protocol for providing efficient and flexible multicast services over the Internet. Traditional multicast architectures construct and update the multicast tree in a distributed manner, which causes two problems: first, since each node has only local or partial information on the network topology and group membership, it is difficult to build an efficient multicast tree; second, due to lack of the complete information, broadcast is often used when transmitting control packets or data packets, which consumes a great deal of network bandwidth. In the newly proposed multicast architecture, a few powerful routers, called m-routers, collect multicast-related information and process multicast requests based on the information collected. m-routers handle most of multicast related tasks, while other routers only need to perform minimum functions for routing. m-routers are designed to be able to handle simultaneous many-to-many communications efficiently. The new multicast routing protocol, called Service Centric Multicast Protocol (SCMP), builds a dynamic shared multicast tree rooted at the m-router for each group. The multicast tree can satisfy the QoS constraint on maximum end-to-end delay and minimize tree cost as well. The tree construction is performed by a special type of self-routing packets to minimize protocol overhead. Our simulation results on NS-2 demonstrate that the new SCMP protocol outperforms other existing protocols and is a promising alternative for providing efficient and flexible multicast services over the Internet.**

I. Introduction and Related Work

Many emerging networking applications, such as audio/video conferencing, video on demand, e-learning, distributed interactive simulation, software upgrading and distributed database replication, require *multicast communication*, a basic type of *group communications*, over a large network such as the Internet. In multicast communication, messages from the source are delivered to all group members of a multicast group. The demand for multicast communication from networking applications has been growing at an accelerated pace. As a result, efficient support for multicast communication remains to be a critical and challenging issue in the networking area.

For multicast communication in a wide-area network (WAN), a straightforward implementation is to adopt the unicast model. However, the problem of this simple scheme is that it wastes too much network bandwidth, and causes long communication delay. A more efficient multicast routing scheme is to build a multicast tree which contains paths from a source node to a group of destination nodes via some intermediate nodes in the network, so that the source node can transmit only one copy of the packet along the tree. Clearly, the multicast tree scheme can reduce data replication and lead to shorter communication delay.

Multicast trees are usually constructed by a multicast routing protocol running on every router in the domain. The main concern of a multicast routing protocol is how to efficiently and effectively construct multicast trees and manage multicast sessions. For network-wide multicasting, a network can be modeled as a graph with the nodes representing routers and the edges representing links between routers as shown in Fig. 1(a). The internal structure of a generic router is showed in Fig. 1(b). We will use "router" and "node" interchangeably from now on.

There have been extensive research and development activities in the area of multicast routing in recent years, see, for example, [1]-[8]. Multicast routing protocols can be categorized into two types: Shortest Path Tree (SPT) based multicast routing protocols and Shared Tree (ST) based multicast routing protocols. SPT-based protocols build a separate multicast tree for each (*source*, *group*) pair rooted at the source. DVMRP (Distance-Vector Multicast Routing Protocol) [2] and MOSPF (Multicast Extensions to Open Shortest Path First Protocol) [3] are SPT-based protocols. SPT-based protocols minimize end-to-end delay because every group member is connected to the source with the shortest delay path between them. However, there are three problems in SPT-based multicast routing protocols. First, SPT-based multicast routing introduces the scalability problem for a large network in terms of routing table storage since routers need to store routing information for each (*source*, *group*) pair. Secondly, adopting DVMRP or MOSPF wastes a large portion of the network bandwidth due to flooding, although in different ways. Thirdly, the multicast trees generated in DVMRP or MOSPF are shortest path trees, which may not be the lowest cost multicast trees. ST-based multicast routing protocols have been proposed to overcome the scalability and bandwidth wasting problems. ST-based protocols create a single tree for the entire group, which is shared by all the sources. The shared tree is rooted at a core router which is publicized to all sources by some mechanism. Core-Based Tree (CBT) [5], Protocol-Independent Multicast Sparse Mode (PIM-SM) [6] and Simple Multicast (SM) [7] are ST-based protocols. However, the ST-based multicast routing protocols introduce new problems. First, the multicast communication from a source to a multicast group may not be very efficient in most cases in terms of multicast tree cost and communication delay due to the shared multicast tree. Secondly, the elected core has the same architecture as any other routers in the network, thus has limited computing and packet forwarding capability. Moreover, the ST-based approach may cause traffic jam around the core, since packets from multiple sources may reach the core simultaneously. The traffic concentration will further cause the problems of packet loss and longer communication delay. Finally, the multicast communication between a source and a multicast group cannot tolerate any failure of the core.

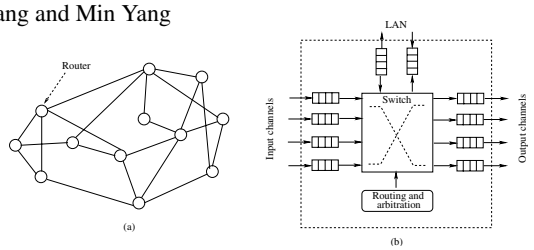
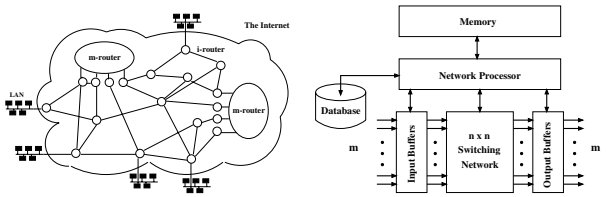


Fig. 1. (a) An example of a WAN. (b) Internal structure of a generic router.

Our work is motivated by the need of a practical, flexible multicast architecture that can minimize the overall tree cost while maintaining a relatively low overhead in bandwidth and computing and storage resources in routers. In this paper, we propose a service-centric approach for multicast communication, in which there are one or more powerful routers in each domain which handle most of multicast-related tasks. This approach can provide better efficiency and flexibility for two reasons: first, the powerful routers possess all the information on the network, such as network topology, link delay, link cost and group membership,

Y. Yang and M. Yang are with Dept. of ECE, SUNY at Stony Brook, NY 11794, USA and J. Wang is with East Isle Technologies, Inc. Setauket, NY 11733, USA.
Research supported by NSF grant numbers CCR-0073085 and CCR-0207999.



(a) (b)

Fig. 2. (a) Illustration of m-routers and i-routers in the Internet. (b) A sketch of the internal structure of an m-router.

thus it can adopt a more sophisticated algorithm to build a more efficient multicast tree without increasing network bandwidth usage (note that constructing the optimal multicast tree is a NP-hard problem [14]); second, since the multicast tree constructing algorithm is run only on the powerful routers, it is convenient to modify the algorithm if the requirements of the multicast applications change. Other routers in the domain do not need to know the change. We also propose an associated multicast routing protocol, SCMP, which takes advantage of both the shared tree and centralized processing. We conducted extensive simulations on NS-2 and compare SCMP with other protocols in terms of multicast tree optimality and network-wide performance.

II. The New Multicast Architecture

The proposed multicast architecture consists of three components: the specially designed powerful routers, the group and session management protocol and the multicast routing protocol. In this section, we first give an overview of the multicast architecture, and then describe the three components separately.

A. Overview of the new multicast architecture

The proposed multicast architecture and protocol are based on the concept of *service-centric multicast routing*. Instead of treating each router equally as in existing multicast architectures, the new multicast architecture has two different types of multicast routers, which we call *master multicast router* (or *m-router*) and *intermediate multicast router* (or *i-router*). An i-router functions as an ordinary multicast router for forwarding multicast packets; while an m-router is responsible for more complex service-related tasks such as multicast session and group membership management, routing scheme control, transmission bandwidth management, and traffic scheduling and performs some service specific functions. The m-router integrates multiple routers, each of which can serve more than one multicast groups. Each m-router should be owned by an ISP (Internet Service Provider), who provides multicast services, so that the ISP centralizes most of service-related tasks on the m-router to alleviate the burden on the Internet. An ISP may own more than one m-routers in the Internet for serving its customers in different geographic regions. Fig. 2(a) shows the m-routers and i-routers in the Internet. For simplicity, we assume that one domain owns only one m-router in this paper although our approach can be easily extended to multiple m-routers per domain. An i-router can adopt any multicast-capable switching fabric. For an m-router, we need to develop a special type of switching fabric. In general, we can consider an m-router has m input ports and m outputs ports and each of its input/output links has sufficiently high bandwidth.

A typical multicast communication is realized as follows. For each multicast group, the m-router dynamically assigns one of its output ports to the group, and a multicast tree rooted at the output port is built to reach members of the group via some i-routers which are non-root nodes of the multicast tree. The multicast tree is generated in the m-router in response to the JOIN/LEAVE messages and then distributed around the domain. When transmitting a multicast packet, if the source router is the m-router itself, or it is on the multicast tree already, the packet is sent along the multicast tree; if the source router is a router which is not on the multicast tree, the source router first sends its packet to the m-router, then the m-router forwards the data along the multicast tree.

B. Design of the m-router

The m-router plays a critical role in the multicast architecture because it handles most multicast related tasks and it is the root of the multicast trees. It is required that the m-router is capable of handling multiple multicast tasks simultaneously and forwarding multicast traffic efficiently. Fig. 2(b) shows the sketch of the internal structure of an m-router, which has m input ports and m outputs ports, and an $n \times n$ switching fabric.

Many tasks in the m-router, such as managing multicast group membership, generating multicast trees, scheduling, routing and transmission, are relatively independent, which can be performed in parallel. Thus, the m-router can adopt a multiprocessor or a cluster computer architecture. MMC Networks' NP3400 processor [16] and Motorola's C-port network processor [17] are examples of the routers with multiple processors.

Efficient hardware support from the underneath switching fabric with multicast capability is the key for the m-router to provide various multicast services and handle multicast traffic. There has been a lot of work concerning multicast switching fabric designs in the literature [9]-[13]. Recall that in various multicast applications, for a multicast connection, a source may or may not belong to the multicast group. Also, there may be several multicast connections from different sources to the same multicast group, which can be referred to as *many-to-many communication*. To support multiple such many-to-many communications in the Internet, the multicast switching fabric can be designed by adopting the concepts from the conference switching networks [12], but the switching fabric of the m-router is required to support more general communication patterns and make fully use of the multicast trees built in the Internet. An illustration of the m-router switching fabric interconnecting with the Internet is shown in Fig. 3. By adopting the sandwich network structure [11], [12], the $n \times n$ switching fabric consists of three $n \times n$ subnetworks, permutation network (PN), connection component network (CCN) and distribution network (DN). Among them, the PN and the DN are permutation networks, which have the functions of keeping inputs/outputs in some order for the CCN and performing load-balance for the m-router in the Internet. The CCN realizes the connections of multiple sources by "merging" them in a reversed tree rooted at an output, which is then linked through the DN to the root of the corresponding multicast tree in the Internet. This way, the multiple sources can share one multicast tree via the connections in the CCN. However, as it should be, sources to different multicast groups are never be connected in the switching fabric. The CCN in the switching fabric of the m-router functions as connecting

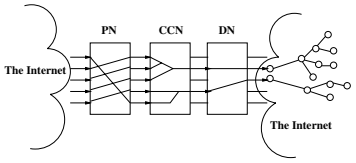


Fig. 3. An m-router switching fabric interconnected with the Internet.

links coming from different sources to a link leading to the root of the multicast tree.

C. Multicast group and session management protocol

We expect the new architecture would still maintain the user-transparency property for multicast group information, and be compatible to existing protocols. We use an existing protocol, Internet Group Management Protocol (IGMP) [1], to manage the process of a host computer joining or leaving a multicast group in the subnet. IGMP is used by hosts to register their dynamic multicast group membership. It is also used by routers to discover these group members. In IGMP, one of the routers connected to the same subnet is selected to act as the designated router (DR). The DR is responsible for sending Host Membership Query messages to discover which groups have members on their subnet. Hosts respond to a Query by generating Host Membership Reports, reporting each group they belong to on the network interface from which the Query was received. Thus, the group membership is transparent to end-users regardless of they are in the group or outside the group. However, unlike DVMRP and CBT where no routers have the complete group membership information and MOSPF where every router knows all the membership information, in the new architecture only the m-router knows all the group membership information.

The m-router acquires group membership information from i-routers as follows. Whenever an i-router finds out that a host in its resided subnet joins a new group, or all members in its subnetwork quit from an existing group, the i-router sends a unicast message to the m-router to inform the group membership change. Thus, the m-router collects group membership information passively and dynamically. Since the m-router is responsible for managing the multicast groups, it should be able to issue a multicast address for a new multicast group, revoke a multicast address from an abandoned multicast group, and publish the multicast addresses for existing multicast groups.

For managing multicast sessions, the m-router is responsible to start a new multicast session, to tear down an expired multicast session, and to check, track and record the multicast traffic in the corresponding multicast session. Since the lifetime of a multicast session depends on its multicast service requirements, multicast session management follows the service-related requirements and policies. The m-router also keeps track of all the membership on-off information for multicast scheduling/routing and for accounting/billing purposes.

Because the m-router is the sole entity for managing the multicast groups and multicast sessions, it should have abilities for outsiders to query proper information about multicast groups and sessions in the m-router. All the service-related information will be kept in a database on the m-router.

D. Multicast routing protocol (SCMP) - an overview

Having described the multicast architecture, we now give a brief overview of its associated multicast protocol SCMP. The details of this protocol will be presented in the next section.

We assume that the Internet consists of a number of autonomous systems or domains, where each domain is under the administrative control of a single entity. Besides a multicast routing protocol, each domain also runs a unicast routing protocol. SCMP is an intradomain multicast protocol that constructs the multicast tree within the domain in which a link state unicast routing protocol is running.

The multicast routing protocol for the proposed architecture is expected to satisfy the following requirements: first, allow any sophisticated network-wide routing algorithms to be used for constructing multicast trees; second, the computing effort for multicast trees is centralized at the m-router, saving the computing resource of other routers; third, multicast routing information is transmitted only through the i-routers on the multicast tree, and does not affect the rest of the Internet.

We assume that each link has two parameters: *link delay* and *link cost*. *Link delay* is defined as the sum of the perceived queueing delay, transmission delay and propagation delay over the link. *Link cost* is determined by the utilization of the link. It is used to describe the cost to use the link. The higher the utilization, the higher the link cost. The links are symmetric, which means that any link has the same delay and cost in both directions.

In SCMP, the m-router's IP address is known to all the routers in the domain in advance. This can be realized by putting the IP address of the m-router in every router's configuration file. After a router is notified by one host in its subnet that it wants to join or leave a group, the router sends a JOIN/LEAVE request message to the m-router indicating the group ID and the IP address of the router. The m-router keeps track of all the group members in the group. When the m-router receives such JOIN/LEAVE request message, it updates the multicast tree according to the change of the group membership. A network-wide routing algorithm is run on the m-router to generate a multicast tree for a given multicast group. This is achievable because the m-router has all the group membership and global network topology information.

The multicast tree is generated in the m-router based on the collected topology and membership information. As more and more applications have QoS requirements, the multicast tree should be optimized to satisfy these requirements. However, constructing an optimal multicast tree has been proved to be a NP-hard problem [14]. Kou, Markowsky and Berman proposed KMB algorithm in [19] which achieves best approximation ratio on tree cost, but it does not consider tree delay. In [20], we proposed an efficient heuristic algorithm, called Delay Constrained Dynamic Multicast (DCDM), to construct a delay constrained dynamic multicast tree with minimum cost. The basic idea of our algorithm is that when a new member joins the group, the routing algorithm tries to find a graft node which can minimize the tree cost and the tree delay at the same time; when a group member leaves, the branch leading to the leaving group member will be pruned and the rest of the tree is intact. In SCMP, we adopt the DCDM algorithm for constructing the multicast tree.

After the multicast tree is generated in the m-router, it should be physically constructed in the domain. The routers on the tree should update the routing table according to the generated multicast tree. SCMP uses a special type of packet, TREE packet, to accomplish this. Each TREE packet contains the complete information about a subtree of the multicast tree. The m-router is responsible to generate the original TREE packets. The i-router receives a TREE packet which represents a subtree rooted at the i-router itself. After receiving the TREE packet, the i-router sets the routing table according to the information in the TREE packet, and sends a new TREE packet to each i-router which is the child of the i-router in the multicast tree. The procedure is performed recursively on the multicast tree until it reaches leaf routers. The resulting multicast tree is a shared, bi-directional tree rooted at

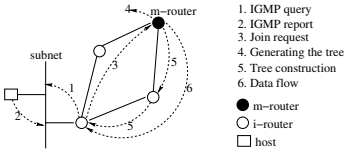


Fig. 4. Overview of the SCMP protocol.

the m-router.

When a source sends a packet to a group, if the source is not on the tree, the multicast packet is encapsulated in a unicast packet and sent to the m-router first. The m-router decapsulates the packet and forwards it along the tree as a multicast packet. If the source is on the tree, the packet can be forwarded along the tree directly because the tree is bi-directional. Fig. 4 shows an overview of the SCMP.

III. Multicast Routing Protocol (SCMP)

In this section, we give the detailed descriptions of the multicast routing protocol. We first provide some terminologies.

A. Terminologies

Every router on the multicast tree has an *upstream* which is the parent router of the router. The root of the tree (the m-router) has no *upstream*. Every router on the tree has a *downstream* which is a set of routers and interfaces. The routers in the *downstream* are the child routers of the router. The interfaces in the *downstream* are the interfaces of the router that are connected to the subnets in which there is at least one host belonging to the group. A multicast routing entry is a triple with three fields: (group id, upstream, downstream). Suppose the router is on the multicast tree of a group, "group id" is the identification of the group; "upstream" is the *upstream* of the router and "downstream" is the *downstream* of the router. The multicast routing table is composed of one or more multicast routing entries. The *tree cost* is defined as the sum of the link cost on the tree. It is used to evaluate the cost to deliver packets along the multicast tree. For every pair of nodes in the network, there exists a path that has the least cost among all the paths connecting these two nodes, and we use P_{lc} to denote this path. Similarly, for every pair of nodes in the network, there exists a path that has the shortest delay among all the paths connecting these two nodes, and we use P_{sl} to denote this path. The *unicast delay* between two nodes is the delay of path P_{sl}. The unicast delay of a group member is the unicast delay between the group member and the m-router which is denoted as ul. For any group member, there is a unique path on the tree connecting the group member to the m-router. The *multicast delay* of the group member is the delay of the unique path and is denoted as ml. The longest multicast delay of all group members is the *tree delay*.

B. Member joining

When a host wants to join a group G, it sends an IGMP report message identifying the group id, gid, in response to the designated router DR's IGMP query message. When the DR receives an IGMP report for group gid, it checks whether it is on the multicast tree of group gid first. This is completed by checking whether there exists a multicast routing entry whose "group id" is gid. If there exists such a multicast tree, then it checks whether the interface connected to the host is included in the "downstream" of the multicast routing entry. If not, add it to the "downstream." If the interface is the first interface added to the "downstream," the DR will send a JOIN message to the m-router. Although the multicast tree does not need to be updated, the m-router needs this information for possible accounting and billing purposes.

If the DR is not on the multicast tree, it sends a JOIN message to the m-router indicating the gid and the IP address of the DR. At the same time, the interface from which the IGMP report message is received is marked so that it will be added to the "downstream" of the multicast routing entry which will be set up when the DR receives the TREE packet later. The pseudo-code of the member joining procedure is shown as follows.

Member Joining Procedure:
//suppose DR receives IGMP join report containing gid from interface inf.
if there exists a multicast routing entry whose group id field is gid
  if inf is not in the multicast routing entry
    add inf to downstream of the routing entry;
  if inf is the only interface element in downstream
    send JOIN message to m-router;
else
  store (gid, inf) for creating the multicast routing entry in the future;
  send JOIN message to m-router;

C. Member leaving

When the last group member of a subnet sends an IGMP leave report to the DR, the DR removes the interface from the "downstream" of the routing entry first. After that, the DR checks whether it becomes the leaf node of the multicast tree. A router is a leaf node of the multicast tree when the *downstream* of the router is null. If the DR is not a leaf node, there are two cases: (1) There is at least one interface element in the *downstream*. In this case, no action is needed; (2) All the elements in the *downstream* are routers. In this case, although the multicast tree remains the same, the DR should send a LEAVE message to the m-router for possible accounting and billing purposes.

If the DR is a leaf node after receiving the IGMP leave report, in addition to sending a LEAVE message to the m-router, it also sends a PRUNE message to the *upstream* router so that the *upstream* router will no longer forward the multicast packet to it. Similarly, if the *upstream* router finds that itself is a leaf node, it triggers another PRUNE message to its *upstream* router. This PRUNE message will continue until it reaches a non-leaf router. Following is the pseudo-code of the member leaving procedure.

Member Leaving Procedure:
//suppose DR receives IGMP leave report containing gid from interface inf.
remove inf from downstream of the multicast routing entry;
if downstream becomes NULL
  send a PRUNE message to upstream router;
  send a LEAVE message to m-router;
else if all the elements in downstream are routers
  send a LEAVE message to m-router;

D. Constructing the multicast tree at the m-router

The multicast tree is constructed in the m-router before it is physically formed in the domain. As discussed earlier, whenever a host wants to join or leave a group, a JOIN or LEAVE message will be sent to the m-router. These JOIN/LEAVE messages make the m-router update the multicast tree.

SCMP adopts a heuristic algorithm DCDM we proposed in [20], to construct a delay constrained dynamic multicast tree with minimum tree cost. Given a network topology, the m-router, the old multicast tree and the new member s, the new multicast tree can be constructed as follows. Suppose l is the old tree delay and m is the number of routers on the old tree. For each router on the tree, there are two paths, P_{lc} and P_{sl}, connecting s to the router which were computed in advance. Thus, there are a total of $2m$ paths connecting s to the old tree. If the unicast delay of s is greater than l, the shortest delay path from s to the m-router

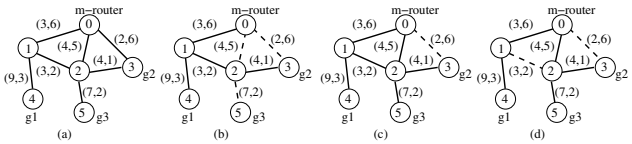


Fig. 5. Example of using the DCDM algorithm. (a) Network topology. (b) Multicast tree after g_1 and g_2 join the group. (c) A loop is formed after g_3 joins. (d) Multicast tree after g_3 joins.

is added to the tree and l is changed to the unicast delay of s. If the unicast delay of s is shorter than or equal to l, check the $2m$ paths one by one, choose the path with the least cost which makes the multicast delay of s no longer than l, and add this path to the tree. There are two parameters we need to minimize in the algorithm: the tree cost and the tree delay. On one hand, in order to minimize the tree delay, the longest unicast delay of all group members is used as the upper bound. On the other hand, in order to minimize the tree cost, the new member with shorter unicast delay than the upper bound will graft on an ontree node that can lead to the minimum tree cost as long as its multicast delay is less than the upper bound. Readers may refer to [20] for more details of this algorithm.

Fig. 5 gives an example of the multicast tree construction by using the DCDM algorithm. Fig. 5(a) is the network topology. The numbers on the link represent (*link delay*, *link cost*). Node 0 is the m-router. Nodes 4, 3 and 5 represent three group members g_1, g_2 and g_3 respectively. Suppose g_1 is the first group member to join. The algorithm finds a shortest delay path connecting the m-router and g_1, which is $0 \rightarrow 1 \rightarrow 4$, and the current tree delay is $3 + 9 = 12$. Now g_2 wants to join the group. The unicast delay of g_2 is 2 which is less than the current tree delay 12. Thus, there are two nodes for g_2 to graft on, node 0 and node 1. If choosing node 0, the multicast delay of g_2 is 2 and the tree cost is increased by 6. If choosing node 1, the multicast delay of g_2 is $3 + 3 + 4 = 10$, the tree cost is increased by 3. Therefore, choosing node 1 will not increase the tree delay while the tree cost is minimized at the same time. The final tree is $0 \rightarrow 1 \rightarrow 4$ and $1 \rightarrow 2 \rightarrow 3$, as shown by the solid lines in Fig. 5(b).

One issue we need to deal with in the multicast tree construction is to eliminate loops. Suppose g_3 wants to join the group after g_1 and g_2 join the group. The unicast delay of g_3 is $4 + 7 = 11$, which is less than the current tree delay 12. If choosing node 2 as the graft node, the multicast delay will be $3 + 3 + 7 = 13$, which is greater than 12. Thus, the graft node should be node 0. As shown in Fig. 5(c), after path $0 \rightarrow 2 \rightarrow 5$ is added to the tree, a loop $0 \rightarrow 1 \rightarrow 2 \rightarrow 0$ is formed. In order to break the loop, the algorithm will search the path between the joining node and the graft node. Once finding a node which is already on the tree, the algorithm will prune the branch from the node towards upstream in a similar way to that the prune operation is done in the group leaving procedure. In this case, the algorithm prunes the tree upstream from node 2 until it reaches node 1. The final tree is $0 \rightarrow 1 \rightarrow 4$, $0 \rightarrow 2 \rightarrow 5$ and $2 \rightarrow 3$, as shown by the solid lines in Fig. 5(d).

When the m-router receives a LEAVE message, it removes the node from the group. If the node becomes a leaf node, the tree is pruned towards the upstream router until it reaches a group member or a node that has more than one downstream routers. This guarantees that the tree in the m-router is consistent with the actual tree in the network. The "real" prune operation is accomplished by the leaving member sending the PRUNE message upstream hop by hop.

E. Forming the multicast tree in the network

After the m-router constructs the multicast tree, it should distribute the tree in the domain so that the i-routers on the tree can update their routing tables and forward the multicast packet correctly. This is completed by the multicast tree forming process.

In order to minimize the protocol overhead, we adopt the self-routing scheme used in [10], in which multicast routing is realized by the tag attached to the packet. Similarly, in a random topology network, we can still use such self-routing packets to construct the multicast tree, which is called TREE packet. The TREE packet includes all the information on a tree. The length of the TREE packet is a variable depending on the size of the tree. The TREE packet format is described in the following table.

TREE Packet Format

Number of the downstream routers
IP address of the downstream router 1
Length of subpacket 1
Subpacket 1
IP address of the downstream router 2
Length of subpacket 2
Subpacket 2
...

The format of the subpacket is the same as the format of TREE packet. This recursive packet structure reflects the recursive structure of the tree. "Subpacket i" includes all the information on the tree rooted at "downstream router i."

The TREE packet is a self-routing packet, which means that the routers forward the TREE packet according to the information in the TREE packet itself. When a router receives a TREE packet from its upstream router, it updates its routing table based on the information in the packet and sends new TREE packets to its downstream routers if any. The first TREE packet is generated in the m-router based on the multicast tree. Since the m-router is the root of the multicast tree, each downstream router of the m-router is the root of a subtree. The m-router builds a TREE packet for each subtree. Then the m-router sends these TREE packets to the corresponding downstream routers and the downstream routers are added to the "downstream" of the routing entries. When an i-router receives a TREE packet, the TREE packet should include the information of the subtree which is rooted at the i-router itself. The "upstream" of the route entry is set to be the router from which the TREE packet is received. The TREE packet is split into several smaller TREE packets each of which represents a subtree rooted at one of the downstream routers. Then each of the smaller TREE packets is sent to the corresponding downstream router after the downstream router is added to "downstream" of the route entry. After all the TREE packets reach leaf routers, the tree is formed.

The pseudo-code of the TREE packet processing algorithm in i-routers is showed in the following table.

TREE Packet Processing Algorithm
$upstream$ = IP address of the router the TREE packet is received from;
$childnum$ = "the number of downstream routers" in TREE packet;
for each downstream router ds in the TREE packet
  add ds to the $downstream$ of the routing entry;
  extract the subpacket corresponding to ds from the TREE packet;
  send the subpacket as a new TREE packet to the ds;

Now let's look at an example. Suppose the m-router has three downstream routers. Fig. 6 shows the multicast subtree rooted at node 2. Here we use the node id to represent the ad-

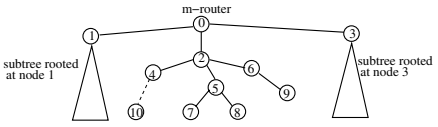


Fig. 6. Forming the multicast tree using TREE and BRANCH packets.

dress of the router. The m-router generates three TREE packets for its three downstream routers respectively. The TREE packet for node 2 is $(3; 4, 1, 0; 5, 7, 2, 7, 1, 0, 8, 1, 0; 6, 4, 1, 9, 1, 0)$, where 3 means node 2 has three downstream routers; $(4, 1, 0)$ means the first downstream router is node 4, the length of subpacket representing the subtree rooted at node 4 is 1, the subpacket is (0); Similarly, the second downstream router is 5, the length of subpacket representing the subtree rooted at node 5 is 7, the subpacket is $(2, 7, 1, 0, 8, 1, 0)$, and so on. When node 2 receives this TREE packet, it sets the "upstream" to the m-router and splits this TREE packet into three TREE packets: $(0), (2, 7, 1, 0, 8, 1, 0)$ and $(1, 9, 1, 0)$. Then sends them to nodes 4, 5 and 6 and adds nodes 4, 5 and 6 into the "downstream" of the routing entry. The multicast routing entry in node 2 after processing the TREE packet is (gid;1;4,5,6). When receiving the TREE packets, node 4 will add the interface marked after receiving the IGMP report message into the "downstream" of the routing entry, then it sets "upstream" to node 2; node 5 will continue to split the TREE packet into two TREE packets which are both (0) after updating the routing entry; node 6 will send TREE packet (0) to node 9 after updating the routing entry. After nodes 7, 8 and 9 receive the TREE packets and update the routing entries, the tree forming process is completed.

Clearly, whenever a new router joins a group, a TREE packet will be triggered in the m-router if the tree is changed. If the change is small, using a TREE packet containing the whole tree structure is too expensive. Thus, we use a new type of packet, called BRANCH packet, to update a minor tree change. A BRANCH packet consists of a branch of the tree from the m-router to the new group member. The BRANCH packet is composed of a sequence of routers that are on the path from the current router to the new group member in order. When an i-router receives a BRANCH packet, it deletes itself from the head of the path, sets the "upstream" as the router from which the packet is received, adds the next router on the path to the "downstream" of the routing entry and forwards it to the downstream router. For example, in Fig. 6, if node 10 wants to join the group, the m-router generates a BRANCH packet $(2, 4, 10)$ and sends it to node 2. Node 2 then updates the routing entry, deletes itself from the packet and sends a BRANCH packet $(4, 10)$ to node $4, \ldots$, until node 10 receives the BRANCH packet (10), it adds the marked interface to the "downstream" and sets the "upstream" to node 4.

F. Forwarding multicast packets

The multicast tree constructed is a bi-directional tree. It means that the multicast packet can be transmitted in both directions on the tree. The multicast packet is not only forwarded to the downstream routers, but also forwarded to the upstream router when necessary. Since we assume the link is symmetric and the delay of a path in both directions is the same, the optimality of the multicast tree will not be impaired.

All the i-routers are configured to know the IP address of the m-router. When a source has a multicast packet to send, it first checks whether it is on the multicast tree. If it is on the multicast tree, it sends the packet to the upstream router and all the downstream routers. If the source is not on the tree, it encapsulates the packet into a unicast packet and sends it to the m-router. The m-router decapsulates the data, puts it in a multicast packet and forwards the packet according to the routing entry.

When forwarding the multicast packet, the i-router considers both the upstream router and the set of downstream routers as a single set, say, F. If an i-router receives a packet, it checks whether the packet comes from a router in F. If yes, then forwards the packet to other routes in F. If no, it drops the packet. The packet forwarding procedure is described as follows.

Multicast Packet Forwarding Procedure:
$incoming$ = the router from which the multicast packet is received;
if $incoming$ is the upstream router
  forward the packet to all the downstream routers;
else if $incoming$ is one of the downstream routers
  forward the packet to upstream router;
  forward the packet to all the downstream routers except $incoming$;
else drop the packet;

IV. Performance Evaluations

We have implemented the newly proposed multicast protocol along with three existing protocols on the NS-2 simulator and conducted extensive simulations to evaluate the protocol. In this section, we compare the multicast trees constructed by our protocol and other protocols, and the maximum end-to-end delay, protocol overhead and data overhead of the protocols.

A. Multicast trees

A random network topology is generated with each link assigned a random link delay value and a random link cost value. A set of group members are randomly picked from the nodes in the topology. We use different algorithms to construct the multicast tree for the same set of group members and compare the multicast tree afterwards. The comparison metrics are *tree delay* and *tree cost*.

We compare our multicast tree construction algorithm used in SCMP, DCDM algorithm, with other four existing algorithms, KMB, DVMRP, MOSPF and CBT. Since the multicast tree constructed by DVMRP or MOSPF is determined not only by the set of group members but also by the source node, we assume that the source node is the same node as the core in CBT. Under this assumption, the multicast trees constructed by these three algorithms (DVMRP, MOSPF and CBT) are identical because all of the trees are composed of the shortest delay paths between the core/source and the group members. Therefore, in the simulation we only implemented CBT, KMB and DCDM.

The simulation model we used is similar to that used in [18]. Nodes in the graph are placed randomly in a rectangular coordinate grid by generating uniformly distributed values for their x and y coordinates. The size of the rectangular is $32,767$ by $32,767$. x and y are random integers between 0 and $32,767$. For every pair of nodes u and v, the probability that there exists an edge connecting u and v is $P(u, v) = \beta * e^{\frac{-d(u,v)}{\alpha L}}$, where $d(u, v)$ is the Manhattan distance between u and v. Let (x_u, y_u) be the x and y coordinates of node u and (x_v, y_v) be the x and y coordinates of node v, then $d(u, v) = |x_u - x_v| + |y_u - y_v|$. L is the maximum Manhattan distance between any two nodes, which is $2 * 32,767$. α and β are two tunable parameters. Increasing α increases the number of edges between far away nodes and increasing β increases the degree of each node. The link cost value of an edge is equal to the Manhattan distance between the two nodes, and the

link delay value of an edge is equal to an uniformly distributed random variable between 0 and the link cost value of the edge.

In our simulations, the total number of the nodes in the topology is 100, the group size increases from 10 to 90 by 10 at each step, $\alpha = 0.25$, and $\beta = 0.2$. Each simulation was conducted 10 times with different random generator seeds. We plot the figure based on the average of the 10 values from the 10 simulations.

Fig. 7 shows the simulation results. We set the delay constraint to three levels: tightest, moderate and loosest. The tightest level means that the delay constraint cannot be tighter, or there is no multicast tree satisfying the delay constraint. The loosest level means that all possible multicast trees can satisfy the delay constraint. Fig. 7(a), (b) and (c) show the tree delay comparison under the three levels respectively. Fig. 7(d), (e) and (f) show the tree cost comparison under the three levels respectively.

We first compare the tree delay. From the figures we can see that no matter which level the delay constraint is, the tree delay of DCDM is much shorter than that of KMB. As the group size increases, the tree delay of DCDM is relatively stable and increases a little, while the tree delay of KMB oscillates intensely. This is because KMB tries to minimize only the cost and does not consider the delay at all. CBT always achieves the shortest tree delay. When the delay constraint is at the tightest level, DCDM can achieve the same tree delay as CBT. When the delay constraint relaxes, DCDM will try to minimize the tree cost without violating the delay constraint. Thus, the tree delay of DCDM is a little longer than that of CBT, but as will be seen shortly, its tree cost is much lower.

Now we compare the tree cost. We can see that as the group size increases, the tree cost of the three algorithms increases too. The tree cost of CBT is the highest, while the tree cost of KMB is the lowest. DCDM achieves the tree cost between CBT and KMB, and it is closer to that of KMB. The differences between the tree cost of any two algorithms increases when the group size increases. When the delay constraint is looser, the gap between DCDM and KMB is obviously smaller than that when the delay constraint is tighter. When the delay constraint is in the loosest level and the group size is small, the tree costs of DCDM and KMB are almost the same.

In our simulations, we also change the location of the m-router to see how it affects the tree cost. We observe that since the group member set and the join order of group members are changing, there is no such location of the m-router that it has the best performance under all conditions. However, we find that there are some heuristics for placing the m-router to achieve good performance in most cases. Rule 1: for each node, calculate the average delay between the node and all the other nodes, and choose the node with less average delay; rule 2: choose the node with a larger node degree; rule 3: choose the node lying on the path whose delay is equal to the diameter of the graph.

B. Network-wide performance

Besides comparing the multicast trees, we implemented SCMP and other three protocols (DVMRP, MOSPF and CBT) on the NS-2 simulator to compare their network-wide performance. The following metrics are compared.

Data overhead: The network bandwidth used by the data packets. A data packet going through one link contributes lc units to the data overhead, where lc is the link cost of the link.

Protocol overhead: The network bandwidth used by the protocol packets. A protocol packet going through one link contributes lc units to the protocol overhead, where lc is the link cost of the link.

Maximum end-to-end delay: The maximum delay experienced by the packets from the source to the group members.

Three different network topologies are used for performance comparison. One is the ARPANET, and the other two are random topologies generated by GT-ITM software[15]. The network size of each random topology is 50 and the average node degrees of the two random topologies are 3 and 5 respectively. There is a source node sending one multicast packet per second. The group size varies from 8 to 40 and the group members are picked randomly. The total simulation time is 30 seconds.

B.1 Data overhead and Protocol overhead

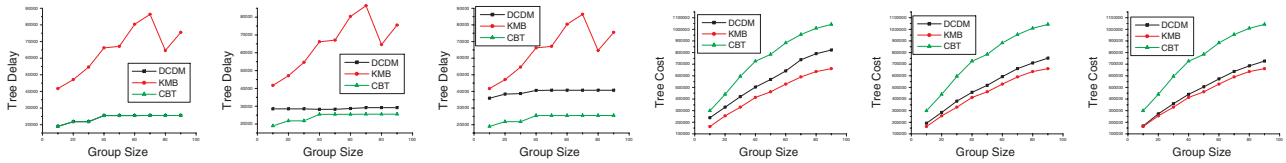
We compare the data overhead of the four protocols and show the results on the left of Fig. 8. We can see that SCMP always has the lowest data overhead among all four protocols, and CBT and MOSPF have higher but relatively closer overhead to SCMP, while DVMRP has much higher data overhead. This is caused by the fact that DVMRP floods the packets frequently when it starts to construct the tree or the timer in a leaf router is expired. The data overhead of the other three protocols is close to each other. Of the three protocol, MOSPF has the most data overhead, and SCMP has the least data overhead. As the group size increases, the difference between SCMP and MOSPF and CBT increases too. This superiority is more obvious when the average node degree is around 3. As can be seen, the data overhead is strongly correlated to the multicast tree cost. The trends of the curves of the two metrics are very similar.

The figures on the right of Fig. 8 show that the protocol overhead increases as the group size increases except for DVMRP, which shows a decrease when the group size increases. If the X axis is extended longer enough, DVMRP would be the one with the least protocol overhead. Since DVMRP adopts "flooding and pruning" algorithm, the more the group members, the less the prune messages. It is a dense mode multicast protocol, which means it is only suitable for the domain in which most nodes are group members. MOSPF has the steepest curve. This is because whenever a group member wants to join or leave the group, the DR will flood a group-membership-LSA packet throughout the domain to make all the routers updated, which generates a great deal of protocol packets. SCMP and CBT have the least protocol overhead. The difference between them is so small that we have to replace the Y axis with log(Y) to separate the two curves, which is shown in Fig. 8(e) and (f).

When the group size increases, the protocol overhead of SCMP is a little bit more than that of CBT. In our simulation, we did not simulate the core selection process of CBT which is a sophisticated and relatively open problem. However, whatever the selection mechanism CBT chooses, it will certainly induce some protocol overhead. When we do not take the core selection into consideration, CBT has a little less protocol overhead than SCMP. This is because CBT only needs to send an acknowledgement packet from the graft node to the newly joining node when performing the join operation, while SCMP always needs to send a BRANCH packet from the m-router all the way to the newly joining node.

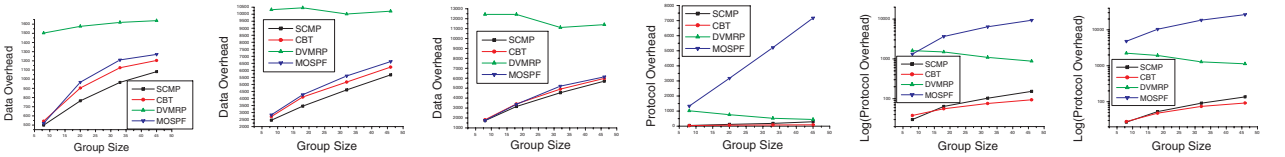
B.2 Maximum end-to-end delay

We now consider the maximum end-to-end delay with different group sizes. Fig. 9 shows the maximum end-to-end delay of the four protocols for the three network topologies. We can



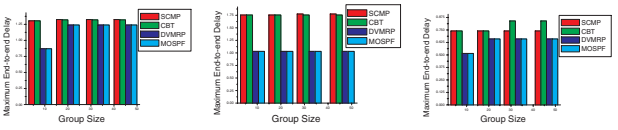
(a) (b) (c) (d) (e) (f)

Fig. 7. (a), (b) and (c): Tree delay comparison when delay constraint is tightest, moderate and loosest, respectively; (d), (e) and (f): Tree cost comparison when delay constraint is tightest, moderate and loosest, respectively.



(a) (b) (c) (d) (e) (f)

Fig. 8. (a), (b) and (c): Group size versus data overhead, (a) ARPANET, (b) Random topology network with average node degree 3, (c) Random topology network with average node degree 5; (d), (e) and (f): Group size versus protocol overhead, (d) ARPANET, (e) Random topology network with average node degree 3, (f) Random topology network with average node degree 5.



(a) (b) (c)

Fig. 9. Group size versus maximum end-to-end delay in seconds. (a) ARPANET, (b) Random topology network with average node degree 3, (c) Random topology network with average node degree 5.

see that the delay of SCMP and CBT is very close and is slightly longer than the SPT-based protocols. This is because that in SPT-based protocols, data packets are delivered from the source node to group members directly, while in SCMP or CBT, packets are sent to the core router first if the source node is not on the tree. We can also observe that the difference in delay becomes smaller when the group size or the node degree increases. However, as pointed out earlier, SPT-based protocols have the scalability and bandwidth wasting problems, which have been seen clearly in their data overhead and protocol overhead.

V. Concluding Remarks

In this paper, we have proposed a service-centric multicast architecture and an efficient and flexible multicast routing protocol SCMP. Compared to existing multicast architectures, the new architecture has following advantages: (1) By concentrating most multicast routing and service-related tasks in the m-router, routing efforts on other routers can be greatly reduced and the bandwidth wasting in the rest of the Internet can be avoided; (2) The new architecture can adopt any sophisticated network-wide routing algorithms in the m-router to construct a near optimal multicast tree, which makes it easier to be adapted to various applications with different QoS requirements; (3) ST-based protocols suffer the traffic concentration problem around the core, while the m-routers in the new architecture are specially designed powerful routers to efficiently handle heavy network traffic, which can greatly alleviate the problem; (4) Another common problem of ST-based protocols is the single core failure problem. In the new architecture, since the m-router is owned and administrated by the ISP, it is easy to install a hot standby system, in which there is a secondary m-router concurrently running with the primary m-router. When the primary m-router fails, the secondary m-router will take over the job automatically. Our simulation results demonstrate that the new SCMP protocol outperforms other existing protocols. In particular, SCMP has the least amount of data overhead among the four protocols. The protocol overhead of DVMRP and MOSPF is much higher than SCMP and CBT. Although the protocol overhead of CBT is a little bit less than SCMP, this is partially because of the simplification of the core selection process of CBT in the simulation. As for tree delay, we can see that SCMP and CBT are very close and their delay is slightly longer than the SPT-based protocols due to the fact that they use shared multicast trees. However, SPT-based protocols have the scalability and bandwidth wasting problems, which can be clearly seen in their data and protocol overhead. Therefore, overall we believe the newly proposed SCMP protocol is a promising alternative for providing efficient and flexible multicast services over the Internet.

References

[1] "Internet group management protocol (IGMPv2)," Internet draft, 1996.

[2] D. Waitzman and C. Partridge, "Distance vector multicast routing protocol," RFC 1075, 1988.

[3] J. Moy, "Multicast extension to OSPF," Internet draft, 1998.

[4] J. Moy, "OSPF version 2," RFC 2328, 1998.

[5] A. Ballardie, B. Cain and Z. Zhang, "Core based trees (CBT version 3) multicast routing," Internet draft, 1998.

[6] S. Deering et al., "Protocol independent multicast-sparse mode (PIM-SM): motivation and architecture," Internet draft, 1998.

[7] R. Perlman et al., "Simple multicast: a design for simple, low-overhead multicast," Internet draft, 1999.

[8] A. Adams, J. Nicholas and W. Siadak, "Protocol independent multicast - dense mode (PIM-DM): protocol specification", Internet draft, 2004.

[9] Y. Yang and G.M. Masson, "Nonblocking broadcast switching networks," *Trans. Computers*, vol. 40, no. 9, 1991, pp. 1005-1015.

[10] Y. Yang and J. Wang, "A new self-routing multicast network," *IEEE Trans. Parallel and Distributed Systems*, vol. 10, no. 12, 1999, pp. 1299-1316.

[11] Y. Yang and G.M. Masson, "Broadcast ring sandwich networks," *IEEE Trans. Computers*, vol. 44, no. 10, 1995, pp. 1169-1180.

[12] Y. Yang, "A new conference network for group communication," *IEEE Trans. Computers*, vol. 51, no. 9, 2002, pp. 995-1010.

[13] J. Duato, S. Yalamanchili, and L.M. Ni, *Interconnection Networks: An Engineering Approach,* Morgan Kaufmann Publishers, 2002.

[14] E. Aharoni and R. Cohen, "Restricted dynamic steiner trees for scalable multicast in datagram networks", *IEEE Infocom'97*.

[15] http://www.cc.gatech.edu/projects/gtitm/.

[16] MMC Networks, Inc. *NP3400*, 2000. http://www.mmcnet.com/.

[17] Motorola Inc. *C-Port Network Processors,* 2002. http://e-www.motorola.com.

[18] B. Waxman, "Routing of multipoint connections," *JSAC*, Dec. 1988, pp. 1617-1622.

[19] L. Kou, G. Markowsky and L. Berman, "A fast algorithm for steiner trees," *Acta Informatica*, vol. 15, 1981, pp. 141-145.

[20] M. Yang and Y. Yang, "Constructing minimum cost dynamic multicast trees under delay constraint," *14th IEEE ICCCN*, pp. 133-138, 2005.