SMRP: Fast Restoration of Multicast Sessions from Persistent Failures[*]

Jian Wu   Kang G. Shin
Real-Time Computing Laboratory
Department of Electrical Engineering and Computer Science
The University of Michigan
{wujz,kgshin}@eecs.umich.edu

Abstract

The growing reliance of networked applications on timely and reliable data transfer requires the underlying networking infrastructure to provide adequate services even in the presence of "persistent" failures (e.g., broken links/routers). It is much more difficult to meet this requirement for multicast sessions than for unicast communications because any on-tree component failure may often cause simultaneous service disruptions to multiple receivers. This paper presents a new multicast routing protocol, called the Survivable Multicast Routing Protocol *(SMRP), which facilitates fast recovery of multicast sessions in face of persistent failures via local detour paths. Our evaluation results show that SMRP trades end-to-end delay and resource usage for short, and hence fast, recovery paths. For example, under a certain set of parameter values, SMRP shortens the recovery path by 20% with only a 5% increase of end-to-end delay and resource usage. Moreover, several design enhancements have made SMRP efficient, robust, flexible and scalable.*

1 Introduction

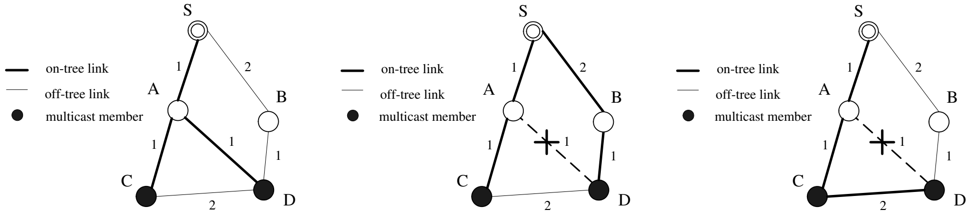
There has been a growing desire among service providers to provide their customers new revenue-generating services with some form of Quality of Service (QoS) guarantees. Unlike traditional datagram services in which average performance is of prime interest, these services impose more stringent QoS requirements in terms of packet delivery delay, jitter, error rate, and so on. It is also essential for the providers to maintain an adequate level of service even in the presence of "persistent" network failures. A persistent network failure, such as disconnection of a link or incapacitation of a node,[1] can occur for various reasons, causing service disruptions that usually last for hours. Typical events that cause persistent failures include accidental cable/fiber cuts, hardware malfunction, power outage, software errors, natural disasters (e.g., fire or earthquake), and human errors (e.g., incorrect maintenance/upgrade) [20]. Moreover, routing instability could also cause serious damage [3, 4], disrupting the original network service for an extended period of time.

Network failures could be much more destructive for multicast communication than for the unicast case. In multicast, each data packet is delivered through a tree topology to achieve efficient resource usage. A link or node failure usually results in simultaneous disconnection of multiple members which use the faulty component to receive data from the source node. A large portion of the original multicast tree might have to be reconstructed, thus imposing a heavy burden on the network. Although new scheme [16] has been proposed for fault-tolerant multicast, it requires a complicated tree construction process and assume the availability of global topology information, rendering itself impractical for large networks, such as today's Internet.

In a traditional multicast environment, once a link or node failure occurs to a receiver's path, a detour path around the faulty component has to be found. Recent studies [25] have shown that the failure recovery time for PIM-based multicast sessions [5] is found to be dominated by the underlying unicast protocol (e.g., OSPF [10]) recovery process, i.e., the time required to reconstruct consistent unicast routing tables in the affected networks. Our study, however, shows that faster service restoration could be achieved by quickly identifying a local detour instead of waiting a long time for routing re-stabilization. Since each node on the multicast tree has the same piece of information from the source, the portion of the original multicast tree, which was unaffected by the failure, can be used for service recovery. For instance, consider the multicast tree in Figure 1(a).

[*]The work reported in this paper was supported in part by the Office of Naval Research under Grant N00014-99-1-0465.

[1]This includes both the physical breakdown of the node and service unavailability under heavy network congestion.



(a) initial network (b) traditional recovery (c) recovery via local detouring

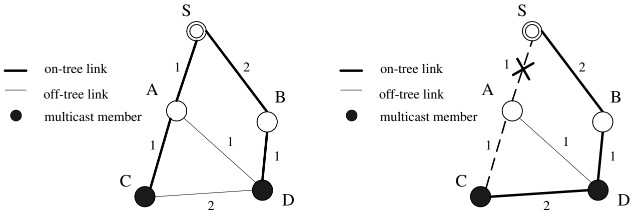
Figure 1. Multicast session recovery.

The number on each link indicates the delay between two end nodes of the link and the original multicast tree is constructed under the assumption that the underlying routing protocol uses the shortest-path-first (SPF) algorithm. Suppose the on-tree link L_{AD} fails, node D is disconnected from the multicast session and attempts to recover by locating a new non-faulty path. Existing multicast routing protocols, such as PIM and its variants [14], select the shortest path from the multicast member to the source or the rendezvous point (RP)[2] in the network. As shown in Figure 1(b), a new path $D \rightarrow B \rightarrow S$ is constructed. An alternative choice for recovery is to take path $D \rightarrow C \rightarrow A \rightarrow S$. Despite of incurring a larger end-to-end delay between the multicast member and the source, path $D \rightarrow B \rightarrow S$ has shorter recover path (i.e., path $D \rightarrow C$) and the recovery process is thus sped up.

The service restoration latency decreases when each disrupted member utilizes its non-faulty on-tree neighbor nodes. In all of the multicast routing schemes known to date, each link/node on the delivery tree is shared by as many members as possible to minimize the tree cost. When a commonly shared link/node fails, it is less likely for a member to receive any assistance from its neighbors for recovery. In Figure 1(a), if link L_{SA} became faulty, both members C and D lose their connections and have to reroute their paths to the source completely. Figure 2(a) illustrates a new tree structure in which C and D's paths are disjoint. Compared to the previous tree, the new tree has the following characteristics.

- **Mitigated service disruption:** since no link/node is shared between two multicast members, at most one member suffers the service disruption due to one network component failure.

- **Faster failure restoration:** the possibility that both members simultaneously lose their connections is reduced, and therefore, fast failure recovery becomes more likely with assistance from neighboring members. In Figure 2(b), when L_{SA} fails, C can quickly restore its service by connecting to its non-faulty neighbor node D.



(a) initial network (b) recovery

Figure 2. Multicast recovery in a new tree.

- **Increased tree cost and/or end-to-end delay:** Because the degree of link/node sharing is decreased, the total tree cost or end-to-end delay might be increased.

In this paper, we propose a new multicast tree construction algorithm called *Survivable Multicast Routing Protocol* (SMRP), which incorporates into tree construction the need for speeding up the service recovery from persistent network failures. Specifically, SMRP builds a multicast tree with less path sharing so as to increase the likelihood that the disrupted members can restore their service with the assistance from their on-tree neighbors. Inevitably, thus-built trees cannot guarantee optimal paths with regard to the end-to-end delay and tree cost, which have been traded away for better fault-tolerance. In one simulation case, SMRP achieves an average of 20% shorter recovery paths at the expense of a 5% increase in the average end-to-end delay or tree cost. In particular, by introducing a configurable parameter D_{thresh}, SMRP can adaptively make a good trade-off between the recovery latency and the transmission efficiency.

The rest of the paper is organized as follows. Section 2 summarizes the related work on fault-tolerant multicast. In Section 3, we formulate the basic problem and propose the new multicast routing protocol. The merits of the proposed technique are evaluated via in-depth simulations and the results are analyzed in Section 4. Finally, Section 5 makes concluding remarks.

[2] For ease of presentation, we ignore the RP case and assume the root of the multicast tree is the actual multicast source in the rest of discussions.

2 Related Work

In general, fault-tolerance can be realized *reactively* or *proactively*. Under the reactive approach, upon failure of an active routing path, its replacement path is constructed for service restoration. Searching for a replacement path is usually time-consuming and hence causes a long service disruption. In contrast, Han and Shin [22] proposed the concept of a dependable real-time connection that consists of *primary* and *backup* channels. If a primary channel is disabled by a persistent failure, its backup channel is activated to become the new primary. The recovery is fast because there is no need to search a new path. In multicast, Medard *et al.* [16] developed an algorithm to construct two multicast trees such that any single failure leaves each member connected to the source by at least one of the two trees. Such redundant trees guarantee the continuity of multicast sessions in face of a network failure. Its complexity, however, makes it difficult, if not impossible, to be applied to large networks.

In order to achieve reliable, in-order delivery of multicast packets, many reliable multicast protocols [6, 17, 21, 23] have been proposed. One widely used technique is *local recovery*: designating one or more hosts other than the source to relay or retransmit packets. In recent years, so-called *gossip-based* protocols became a popular solution to the reliable transmission problem. The common idea of this family of probabilistic protocols [12, 15, 19] is to have each node in a multicast group periodically "talk" to a random set of other nodes in the group about its knowledge of the state of the group. Missing packets can then be recovered by the nodes in a peer-to-peer style. These types of mechanisms assume the occurrence of infrequent, transient packet losses and hence are inefficient in handling *persistent* failures like cable cuts or node crashes, especially for applications that have stringent QoS requirements.

In this paper, we focus on how to restore multicast services efficiently from persistent network failures, which has not yet been well addressed in the literature. We adopted the idea of *local recovery* to accelerate the recovery process. Meanwhile, noticing that current cost-minimized multicast protocols make it difficult to realize local recovery, we devised a new multicast routing protocol which can adaptively adjust the multicast tree structure so as to make a good tradeoff between transmission efficiency and service recovery latency.

3 The Proposed Multicast Routing Protocol

In this section, we first formulate the problem and state our design goals. With the objective of decreasing the length of recovery path, we develop a new multicast routing algorithm and describe the details of the algorithm, including the basic tree construction during the join and departure of multicast members, the maintenance of data structures, and the subsequent tree reshaping procedure for efficiency. Finally, we will discuss the important aspects in the proposed protocol and make several enhancements thereof.

3.1 Problem Formulation and Design Goals

There is an increasing need for communication service with a guaranteed level of fault-tolerance in many multicast QoS-sensitive applications, such as video conferencing, remote monitoring and control of safety-critical assets, distant learning, and medical services. These applications are characterized by the stringent QoS requirement of delay, delay jitter and bandwidth. They usually cannot tolerate a large service restoration latency in the face of significant packet losses. Although a number of reliable multicast protocols have been proposed to ensure reliable in-order packet delivery, they are mainly targeted at infrequent, transient packet losses (e.g., bit errors caused by transmission signal noise) and are unable to handle service disruptions due to persistent link/node failures.

The main intent of this paper is to design a scheme which enhances multicast applications with their required fault-tolerance. Specifically, in our service recovery architecture, the restoration path selected by each disconnected multicast receiver has the following properties: (1) no faulty link/node is involved; and (2) a non-faulty *local neighbor*'s on-tree path is utilized to decrease the length of recovery path. As shown in Figure 1, when D attempts to recover from the failure of link L_{AD}, two detour paths are available. Path $D \rightarrow B \rightarrow S$ is shorter in terms of end-to-end delay between S and D, and would have been chosen by the underlying SPF-based routing protocols. However, path $D \rightarrow C \rightarrow A \rightarrow S$ is preferred in the terms of the required recovery effort because only link L_{CD} needs to be brought into the multicast tree. We define a new metric RD_R which represents the recovery distance (i.e., the length of restoration path) for member R. Here the restoration path only accounts for the new links that need to be brought into the multicast tree. For example, if D chooses $D \rightarrow C \rightarrow A \rightarrow S$ as its new path, the restoration path is $D \rightarrow C$ and hence $RD_D = 2$. Obviously, the restoration path with a small RD_R is preferred for fault-tolerance purposes.

The key factor in realizing local recovery via an on-tree neighbor node is that the node is not affected by the current network failure, i.e., its multicast path is disjoint from the faulty path segment of the disconnected member. Unfortunately, this scheme is hindered by the current multicast routing protocols, which are either based on SPF algorithm or cost-minimizing algorithms. In typical multicast tree, neighbor multicast nodes tend to share a common sub-path to receive packets from the source. If one of the shared components fails, all of these members are likely to be disconnected simultaneously, and it becomes impossible to find a "connected" neighbor for recovery. In this paper, we propose a new multicast routing protocol called Survivable Multicast Routing Protocol (SMRP), which constructs the multicast tree to reduce the likelihood of simultaneously disconnecting neighboring nodes in the tree.

In order to increase the chance in finding a neighbor whose multicast path is disjoint from the faulty segment taken by the disconnected member, it is natural to construct the multicast tree with less link/node sharing among members' multicast paths. In SMRP, when each member joins the multicast session, it always selects a path to the multicast source which is least shared by the other members subject to some constraints. Specifically, we define a new metric $SHR_{S,R}$ that measures the degree of link/node sharing along the on-tree path between source S and node R and is calculated by:

$$SHR_{S,R} = \sum_{all\ L_{i,j} \subset P_T(S,R)} N_{L_{i,j}}, \qquad (1)$$

where $P_T(S,R)$ is the on-tree path between S and R[3], and $N_{L_{i,j}}$ is the number of multicast members whose paths include link $L_{i,j}$. The larger the value of $N_{L_{i,j}}$, the more multicast members share the link $L_{i,j}$. For instance, consider the multicast tree in Figure 1(a). The value of $SHR_{S,C}$ is computed as $SHR_{S,C} = N_{L_{S,A}} + N_{L_{A,C}} = 2 + 1 = 3$. $SHR_{S,R}$ is thus defined to account for link/node utilization by all multicast members in the subtree rooted at R. When a new receiver joins the multicast group, it selects a multicast path via the on-tree node R that has the smallest $SHR_{S,R}$. More details are presented in the following sections.

3.2 Survivable Multicast Routing Protocol

We now describe the main features of SMRP that meet the design goal of increasing the disjointness of the multicast paths between a pair of neighbor nodes. SMRP builds a multicast tree *incrementally* with explicit join or leave requests from members, and consumes only a small amount of network bandwidth for tree construction. Moreover, SMRP adopts the soft-state mechanism to maintain each constructed multicast tree for robustness. Finally, SMRP dynamically reshapes the multicast tree for better overall performance.

We present the proposed protocol in three components: the data structure, the basic tree construction algorithm when member joins or departs, and the tree reshaping procedure.

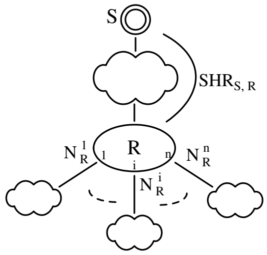


Figure 3. Data structure in SMRP

3.2.1 Data Structure

We have defined a new metric, $SHR_{S,R}$, to quantify the degree of link/node sharing in the path between S and R. The new path should be merged into the current multicast tree at the node R that has the smallest value of $SHR_{S,R}$. Additional data structure is maintained at each on-tree node to enable the path selection process. Listed below are the symbols used.

- S: multicast source. R: on-tree node.
- R_u: upstream node of R in the multicast tree.
- N_R: number of members in the subtree rooted at R.
- N_R^i: number of members in the subtree rooted at the i-th downstream interface of R.
- $SHR_{S,R}$: the measure of link/node sharing along the on-tree path between S and R; $SHR_{S,S} = 0$.

As illustrated in Figure 3, SMRP maintains the following data structure in each on-tree node R: N_R, N_R^i, and $SHR_{S,R}$. The state of N_R is kept implicitly at R since the condition $N_R = \sum_{1 \le i \le n} N_R^i$ holds. In particular, $N_{L_{R,R_u}} = N_R$ since all of the multicast members in the subtree rooted at R use link L_{R,R_u} to reach the source. Hence, Eq. (1) can be transformed to:

$$SHR_{S,R} = \sum_{all\ R' \in P_T(S,R)\ except\ S} N_{R'} = SHR_{S,R_u} + N_R. \qquad (2)$$

The value of $SHR_{S,R}$ can thus be iteratively calculated by exchanging information between each pair of direct on-tree neighbor nodes, R and its ancestor R_u.

3.2.2 Member Join/Leave

Suppose a new member NR prepares to join the multicast session. Instead of using the shortest path provided by the underlying unicast routing protocol, it attempts to locate a multicast path that is merged into the current multicast tree via node R that has the smallest value of $SHR_{S,R}$. For ease

[3] The "$\subset$" operation in Eq. (1) indicates that the link is in the path while the "$\in$" operation is used in Eq. (2) to indicate that the node is in the path.

of exposition, we assume that NR has knowledge of the network topology and can generate all possible paths connecting to the current tree. Relaxation of the assumption is discussed in Section 3.3.1. NR obtains a set of available path options $\{P_T^{R_i}(S,NR)\}$, where $P_T^{R_i}(S,NR)$ indicates a multicast path between S and NR which is merged into the tree at node R_i[4]. For each candidate path $P_T^{R_i}(S,NR)$, there are two states. One is the path length denoted by $D_{S,NR}^{R_i}$, and the other is SHR_{S,R_i}, indicating the degree of link/node sharing along the on-tree path between the source and the node R_i. NR determines its multicast path according to the following criterion.

- **Path Selection Criterion:** for each new multicast member NR, its selected multicast path $P_T^{R^*}(S,NR)$ satisfies the following two conditions:

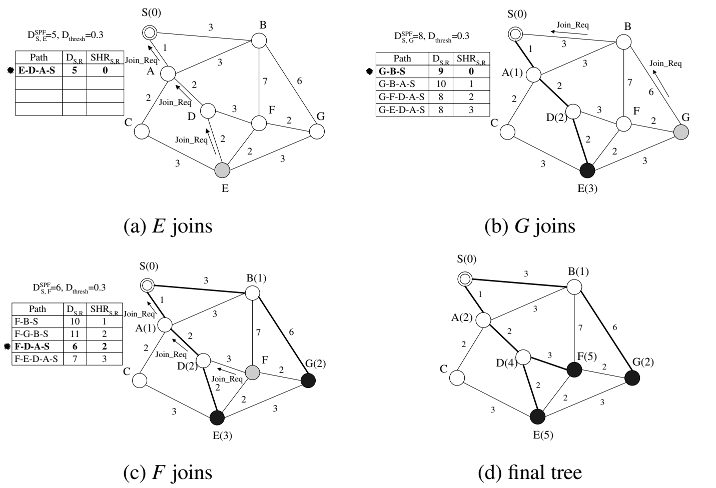
$$\begin{aligned} SHR_{S,R^*} &= \min\{SHR_{S,R_i}\} \\ D_{S,NR}^{R^*} &\leq (1+D_{thresh}) \cdot D_{S,NR}^{SPF} \end{aligned}$$

  where $D_{S,NR}^{SPF}$ is the shortest path between S and NR computed by the underlying unicast routing algorithm, and D_{thresh} is the parameter used to prevent the selection of a path that has an arbitrarily large end-to-end delay. The first condition requires the selected path to have a merger node with a minimum $SHR_{S,R}$ value, while the second condition guarantees the path length to be bounded. If there are multiple candidate paths that satisfy both conditions, the shortest path among them will be chosen.

This criterion is fairly straightforward because the selected path is expected to have the fewest overlapping nodes or links with the current multicast tree. The parameter D_{thresh} is designed to make a controlled tradeoff between reduced degree of sharing and increased end-to-end delay as well as increased tree cost.

After the path selection, NR issues an explicit *Join_Req* message towards S along the selected path. Each intermediate node the message traverses sets up the soft-state multicast routing information in its local database and updates the data structure, if necessary.

The procedure for a member's departure is simple. When one member prepares to leave the multicast group, it issues an explicit *Leave_Req* toward the source along its on-tree path. Each node this request traverses checks if there are still members underneath other than the departing member. If not, the soft-state routing information for this multicast session is cleared and the resource is released. This procedure continues until a router, which has a non-null set of members underneath, is reached. Similar member join/leave procedures can also be seen in the existing multicast protocols such as PIM [5].



$D_{S,E}^{SPF}$=5, D_{thresh}=0.3

	Path	$D_{S,R}$	$SHR_{S,R}$
✹	**E-D-A-S**	**5**	**0**

(a) *E* joins

$D_{S,G}^{SPF}$=8, D_{thresh}=0.3

	Path	$D_{S,R}$	$SHR_{S,R}$
✹	**G-B-S**	**9**	**0**
	G-B-A-S	10	1
	G-F-D-A-S	8	2
	G-E-D-A-S	8	3

(b) *G* joins

$D_{S,F}^{SPF}$=6, D_{thresh}=0.3

	Path	$D_{S,R}$	$SHR_{S,R}$
	F-B-S	10	1
	F-G-B-S	11	2
✹	**F-D-A-S**	**6**	**2**
	F-E-D-A-S	7	3

(c) *F* joins

(d) final tree

Figure 4. Basic tree construction in SMRP.

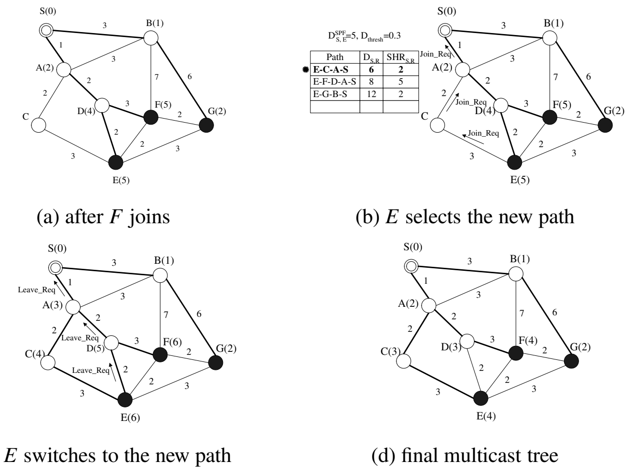
Figure 4 illustrates the path selection process during the consecutive arrival of multicast members E, G, and F. D_{thresh} is set to 0.3. The tables in Figure 4(a-c) show the set of paths available to each new member. The join procedure of E is trivial, and it selects the shortest path as in the traditional routing protocols. In what follows, we focus on how member G determines its multicast path in Figure 4(b). After E completes its join process, each on-tree node updates its $SHR_{S,R}$. For example, node D has $SHR_{S,D} = 2$ as shown in the parenthesis near node D. When G joins, it first generates a set of available paths connecting to the current multicast tree. The table lists four such paths each of which uses a different merger node. According to the path selection criterion described in Section 3.2.2, G chooses path $G \rightarrow B \rightarrow S$ even though path $G \rightarrow F \rightarrow D \rightarrow A \rightarrow S$ has shorter end-to-end delay. Similarly, receiver F in Figure 4(c) selects path $F \rightarrow D \rightarrow A \rightarrow S$. F does not choose path $F \rightarrow B \rightarrow S$ and path $F \rightarrow G \rightarrow B \rightarrow S$ because their path lengths exceed the specified bound in parameter D_{thresh}. Figure 4(d) shows the multicast tree that is eventually constructed.

3.2.3 Tree Reshaping

The shape of the multicast tree determines the disjointness of multicast paths among on-tree neighbor nodes, i.e., the efficiency of local recovery. In real networks, a member might dynamically join or leave the multicast group. As described earlier, the tree structure is incrementally updated during a member's join or departure, e.g., a new branch is created when a new member joins, and it may be trimmed once the associated receiver leaves. After a series of join and departure events, the multicast tree may become skewed and undesirable to certain receivers for fast failure recovery. Hence, we examine how to reshape the tree structure so as to improve the overall performance.

[4] There might be a variety of ways connecting to node R_i from NR. Here we only consider the shortest one.

0-7695-2282-3/05 $20.00 © 2005 IEEE



	Path	$D_{S,R}$	$SHR_{S,R}$
✹	**E-C-A-S**	**6**	**2**
	E-F-D-A-S	8	5
	E-G-B-S	12	2

(a) after F joins

(b) E selects the new path

(c) E switches to the new path

(d) final multicast tree

Figure 5. Tree reshaping in SMRP

For each on-tree node R, it is selected as the merger point by the new member NR because, at the time when NR joins, R has the minimum $SHR_{S,R}$ among all available nodes. If the subsequent new members keep choosing R as the merger point, the value of $SHR_{S,R}$ will be increased, making R unsuitable for accommodating the previous NR. Similarly, with the departure of certain underneath members, R may become a good choice for other members which did not initially select R when they joined the group. Based on these observations, the tree-reshaping operation can be triggered by the following two conditions:

- **Condition I:** For each on-tree node R, every time a new member NR joins the tree through a merger node on the path $P_T(S,R)$, the value of $SHR_{S,R}$ is incremented by the number of links shared by both the new member and the current multicast tree. For example, in Figure 4(d), $SHR_{S,D}$ is increased from 2 to 4 after F joined the group because the two links, L_{SA} and L_{AD}, are used in F's path. We maintain a data structure SHR^{old}_{S,R_u} at R, which records the value of SHR_{S,R_u} received after last reshaping process. Once the difference between SHR_{S,R_u} and SHR^{old}_{S,R_u} is larger than a threshold value, possibly meaning that the accommodation of new members in the sibling subtrees makes it inappropriate for R to use the current on-tree path, a tree-reshaping operation is triggered at R.

- **Condition II:** This condition deals with the case when the departure of members from other on-tree nodes makes them become candidates for a new path. In order to detect such a condition, each on-tree R sets up a periodic timer. Once the timer expires, the node initiates another path selection process as it does when it first joins the multicast group.

The tree-reshaping procedure consists of two steps. First, the reshaping node determines the new multicast path by performing the same procedure as described in Section 3.2.2. If the new path is inferior to the current one, the reshaping process is unnecessary. Otherwise, in the second step, after the new path is set up, the reshaping node switches all its communication to the new path and releases the resources on the old path as in the member departure procedure. Figure 5 illustrates the tree reshaping triggered by E after F is admitted. The increase of $SHR_{S,D}$ by F's sharing of link $L_{S,A}$ and $L_{A,D}$ triggers the reshaping process at E. As shown in Figure 5(b), E completes another path selection process by selecting path $E \rightarrow C \rightarrow A \rightarrow S$. The merger point of the new path, i.e., A, has a smaller value ($SHR_{S,A} = 2$) than the merger point of the current path, D ($SHR_{S,D} = 4$). Note that since the current path still exists when the new path is located, the value of $SHR_{S,R}$ may be inaccurate and should be adjusted before the path comparison is made. Figure 5(c) shows the path switching process after which a better tree structure for fault-tolerance is obtained in Figure 5(d).

3.3 Discussion

In this section, we discuss the key aspects of SMRP and describe potential extensions to make it more efficient, robust, and scalable.

3.3.1 Knowledge of Topology

In Section 3.2.2, we assumed that each member has a full knowledge of the current network topology. This might hinder the deployment of the protocol when the topology information is not available. In what follows, we develop a query scheme to obtain the required information.

In the query scheme, each new member relies on its neighbor nodes to relay its query messages to on-tree nodes. Specifically, the neighbor node sends the query message along its shortest path to the multicast source. Once the first on-tree node R is met, a response message is generated and sent back to the neighbor node, carrying the value of $SHR_{S,R}$. The disadvantage of this query scheme is that it does not guarantee to obtain $SHR_{S,R}$ for all on-tree nodes and the selected multicast path may not be optimal, thus degrading the protocol performance.

3.3.2 Protocol Overhead

SMRP meets its design goals by inducing a certain amount of computation overhead in the maintenance of $SHR_{S,R}$. In order to keep the value up-to-date, any change in the current tree (e.g., node joins or leaves) might trigger a new tree-wide update process. One solution is to defer the calculation of the new $SHR_{S,R}$ value until it is really needed. In the modified protocol, each node initiates the re-calculation of

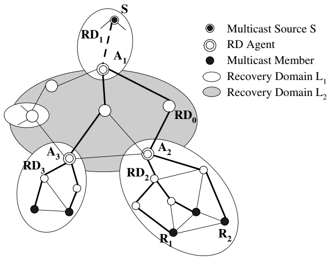


Figure 6. Hierarchical recovery architecture

its $SHR_{S,R}$ only when a query message from a certain new member is received. With the adoption of this technique, the maintenance overhead is amortized into each member's join process. Nowadays, with fast expansion of Internet services, more and more functionalities (e.g., the identification of an authorized member for billing or auditing reasons) have been emerging in the new multicast applications [8]. In view of these complicated functionalities, the fairly small overhead introduced by SMRP should be acceptable, especially when fast service recovery is required or desired.

3.3.3 Hierarchical Recovery Architecture

All of the previous descriptions of SMRP assume a flat network topology. We now extend the basic protocol into an N-level hierarchical network model and explore how this structure improves the scalability of SMRP.

Figure 6 shows a 2-level hierarchical recovery architecture. We choose a 2-level hierarchy because it can be easily mapped onto the current transit-stub Internet structure. Such a 2-level hierarchical structure can be easily generalized into an N-level architecture. The recovery architecture is created by constructing sub-multicast trees in different levels. Each sub-multicast tree represents a *recovery domain* and any node/link failure inside a recovery domain is handled by that domain. As shown in the figure, members are usually clustered into the lowest level (i.e., recovery domain L_1) based on their proximity in the network. In each recovery domain, there is an *agent* acting as the multicast source only for the members inside this domain. The only exception is the recovery domain of the actual multicast source in which the agent acts as a multicast member (e.g., A_1 in Figure 6) relaying packets from the source to the rest of the tree. All $(n-1)$-level agents are clustered so as to create a certain number of n-level recovery domains.

In what follows, we examine how the multicast session recovers from any link/node failure in the network. Consider the multicast path between S and R_1 which runs through recovery domains RD_1, RD_0, and RD_2 in order. As long as the domain in which the failure occurs is identified [1], a fast recovery via local detour path is possible by deploying SMRP in that domain. For example, if the failure occurs in RD_0, agent A_2 then uses its neighbor node A_3 to reconnect to the multicast tree and all tree reconfigurations are confined inside RD_0. The accommodation of SMRP into a hierarchical structure indicates the feasibility of its incorporation into other hierarchical architectures, like NICE [18], for better fault-tolerance.

4 The Simulation Results

In this section, we present the simulation results that demonstrate the merits of the proposed protocol. We first describe the simulation setup and evaluation metrics, then present the simulation results.

4.1 Simulation Setup

We use ns2 [24] to simulate the operations of SMRP. The network topologies are generated by GT-ITM [11], adopting the most common random graph model proposed by Waxman [2] to reflect the structure of real internetworks. With this model, the nodes are distributed randomly in the plane, and for an edge between pairs of nodes (u,v), the edge probability is given by:

$$P(u,v) = \alpha \cdot e^{-\frac{d(u,v)}{\beta \cdot L}},$$

where $0 \leq \alpha, \beta \leq 1$ and $d(u,v)$ is the Euclidean distance from u to v. An increase in α increases the edge density, while an increase in β increases the number of connections of distant nodes. Since we are only interested in the effects of the average node degree on the performance of SMRP, and Zegura *et al.* [7] showed that a targeted node degree can be achieved by different combinations of α and β, we fix the value of β and only change α for our purpose.

In order to explore the characteristics of SMRP more thoroughly, we introduce the following parameters to configure the network scenarios and the proposed protocol.

- N: the number of nodes in the network.
- N_G: the number of multicast members.
- α: the parameter to decide the average node degree.
- D_{thresh}: the parameter to bound the length of the path as described in Section 3.2.2.

4.2 Evaluation Metrics

The simulations were conducted to compare SMRP against the traditional SPF-based multiple routing protocols in terms of the following performance metrics.

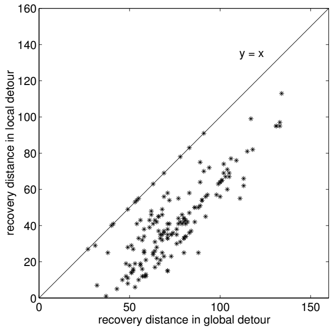


Figure 7. Local detour vs global detour

- **Recovery Distance** (RD_R)**:** the distance between the disconnected member R and its local recovery on-tree node.
- **End-to-End Delay** ($D_{S,R}$)**:** the distance between the source S and each multicast member R.
- **Tree Cost** ($Cost_T$)**:** the sum of the link costs in the multicast tree. Although SPF-based multicast routing protocols do not yield optimal paths in terms of tree cost, we expect that the results presented in this paper are also applicable to the cost-minimizing multicast routing protocols using the study in [13].

The absolute value of the metrics varies arbitrarily, depending on the specific network topology or multicast member selection. Instead, we compute their relative values as follows.

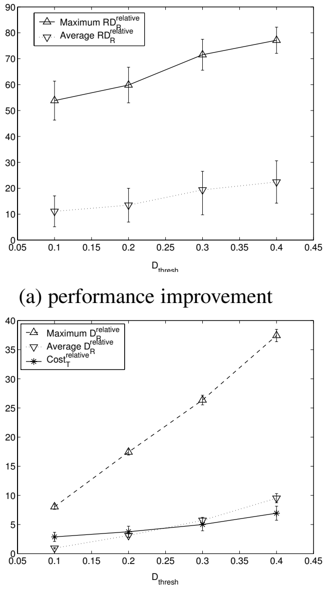
$$
\begin{aligned}
RD_R^{relative} &= \frac{RD_R^{SPF} - RD_R^{SMRP}}{RD_R^{SPF}} \\
D_{S,R}^{relative} &= \frac{D_{S,R}^{SMRP} - D_{S,R}^{SPF}}{D_{S,R}^{SPF}} \\
Cost_T^{relative} &= \frac{Cost_T^{SMRP} - Cost_T^{SPF}}{Cost_T^{SPF}}
\end{aligned}
$$

4.3 Analysis of Simulation Results

We conducted the simulation while varying the configuration of three major parameters, D_{thresh} in protocol design, α and N_G in topology setup. Here we present and analyze the main results, demonstrating the salient features of SMRP.

4.3.1 Global Detour vs. Local Detour

In this section, we verify the correctness of the argument that motivates the design of SMRP, that is, a local detour recovery path is superior to the path established automatically by SPF-based multicast routing protocols (e.g., MOSPF [9], PIM), which we call *global detour* for ease of exposition. In the simulation, we we set the value of N, N_G, α, and D_{thresh} to 100, 30, 0.2, and 0.3, respectively. Five network topologies are randomly generated by GT-ITM and in each topology, the group of multicast members is also randomly selected. In Figure 7, the x-axis and the y-axis represent the recovery distance via global detour and local detour. For each multicast member R, we consider the worst case for R's recovery in which the link closest to the source node on R's multicast path (i.e., the incident link of S towards R) fails. This situation represents the worst situation for R since the failure disables the largest portion of the multicast tree Each asterisk point in the figure indicates the simulation result for one member in each randomly-generated topology. As shown in the figure, most points are below the line $y = x$, meaning that the recovery path via local detour is shorter than the recovery path via global detour. Overall, we observe that the length of the recovery path via local detour is reduced by an average of 33%.



(a) performance improvement

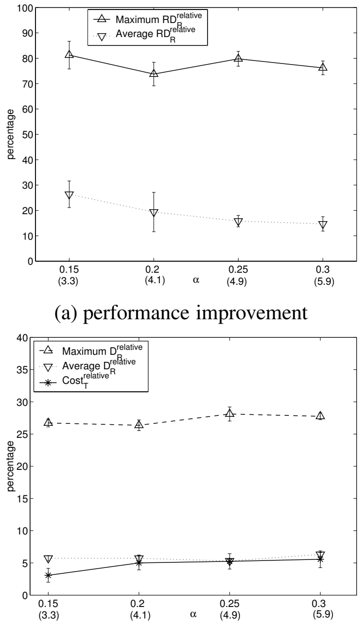
(b) performance penalty

Figure 8. The effect of D_{thresh}

4.3.2 Threshold D_{thresh}

Next, we explore how parameter D_{thresh} affects the protocol performance with respect to the evaluation metrics. All parameters except D_{thresh} are fixed and the values of N, N_G and α are 100, 30, and 0.2, respectively. Four values of D_{thresh} are tested. Under each test, ten network topologies are randomly generated by GT-ITM and in each topology, ten different sets of multicast members are also randomly selected. Each of these 100 simulation scenarios is tested for SMRP

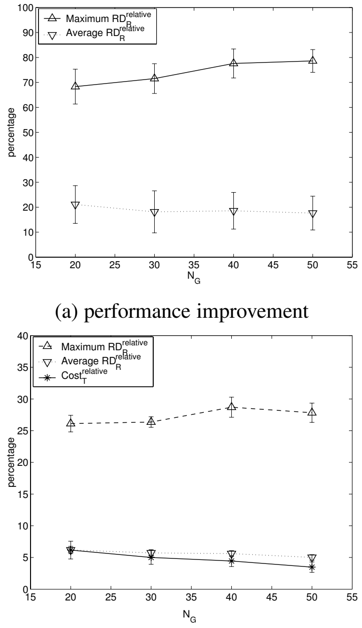
IEEE COMPUTER SOCIETY



(a) performance improvement

(b) performance penalty

Figure 9. The effect of α



(a) performance improvement

(b) performance penalty

Figure 10. The effect of N_G

and SPF protocols separately, and the performance comparison is plotted in Figure 8. The error bars in the figure represent the 95% confidence intervals with the associated metrics. Similarly as Section 4.3.1, the worst case for each member's recovery is examined. According to the definition of the three evaluation metrics, $RD_R^{relative}$ indicates how much SMRP further accelerates the service recovery while $D_R^{relative}$ and $Cost_T^{relative}$ measure the performance penalty in terms of increased end-to-end delay and tree cost, respectively. The following characteristics of SMRP can be observed in Figure 8.

- A fairly large improvement is made by SMRP with a moderate amount of overhead. For example, when D_{thresh} is 0.3, the length of the recovery path is reduced by an average of 20% in SMRP with only 5% performance penalty in terms of increased end-to-end delay or tree cost.

- The performance improvement increases linearly with the parameter D_{thresh} while more penalties are induced, illustrating the basic property of the new protocol. SMRP trades away the communication efficiency (e.g., end-to-end delay) for the decreased path sharing in the multicast tree (i.e., the increased possibility of fast service recovery via a local assistant). The introduction of parameter D_{thresh} enables a fine control of the protocol so that it can be applied to a variety of applications with different fault-tolerance preferences.

4.3.3 Average Node Degree α

In this simulation, we explore the impact of the average node degree in the network on the performance achieved by SMRP. As described earlier, the average node degree of the topology can be tuned by α. We fix the values of N, N_G, and D_{thresh} to 100, 30, and 0.3, respectively, and compare the results under four different α values, 0.15, 0.2, 0.25, and 0.3. During each test, one hundred scenarios are generated in which SMRP and SPF-based protocols are examined. Figure 9 shows SMRP's relative performance against the SPF algorithms. The number under each α value indicates the corresponding average node degree in the network.

- In Figure 9, we observe that the performance improvement diminishes slightly as α (i.e., the node degree) increases. In a network with low connectivity, the multicast tree established by SPF-based algorithms tends to have serious link/node concentration, and hence, the deployment of SMRP makes more performance improvement by lowering path sharing in the multicast tree.

- An acceptable performance improvement can still be achieved in networks with high connectivity. Further study shows, even when average node degree goes up to 10, SMRP achieves 12% path length reduction at the expense of 5% performance penalty.

4.3.4 Group Size N_G

The effect of group size N_G on SMRP's performance is examined using a similar procedure. All tunable parameters except for N_G are fixed ($N = 100, \alpha = 0.2, D_{thresh} = 0.3$) and the value of N_G is varied to 20, 30, 40, and 50. The simulation results are plotted in Figure 10 and summarized as follows.

- The performance is maintained steadily with the change of N_G. SMRP outperforms SPF-based algorithms with respect to the recovery distance, and the path is shortened by an average of 20%. The overhead incurred by SMRP remains at about 5%.
- With the increase of group size, we observe a slight decrease of the performance improvement in terms of average $RD_R^{relative}$. It is possibly because in the same network topology, a larger group makes each member have more close neighbors and SMRP's advantage diminishes.

5 Conclusion and Future Work

With the objective of increasing the likelihood of successful local multicast service recovery, we have proposed a new multicast routing protocol, called SMRP, to construct a multicast tree with less node/link sharing. During the path selection, parameter D_{thresh} can be used to make the controlled tradeoff between the recovery distance and communication overhead in terms of end-to-end delay and tree cost. Our in-depth simulation demonstrates the merits of the proposed protocol. For example, in one simulation setup, the recovery path for each receiver is reduced, on average, by 20% with about 5% overhead. SMRP provides a good option for the multicast applications with different quality-of-service (especially fault-tolerance) preferences.

In our ongoing work, we are conducting more comprehensive evaluation of the protocol by comparing it against other multiples algorithms proposed recently. Meanwhile, we are collecting Internet's topology to evaluate SMRP's applicability to real networks.

References

[1] A. Reddy, R. Govindan, and D. Estrin. Fault Isolation in Multicast Trees. In *Proc. ACM SIGCOMM*, pages 29–40, Stockholm, Sweden, Aug. 2000.

[2] B. M. Waxman. Routing of Multipoint Connections. *IEEE Journal on Selected Areas in Communications*, 6(9):1617–1622, 1988.

[3] C. Labovitz, R. Malan, and F. Jahanian. Internet Routing Instability. *IEEE/ACM Transactions on Networking*, 6(5):515–558, Oct. 1998.

[4] C. Labovitz, R. Malan, and F. Jahanian. Origins of Internet Routing Instability. In *Proc. IEEE INFOCOM '99*, volume 1, pages 21–25, Mar. 1999.

[5] D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jabobson, C. Liu, P. Sharma, and L. Wei. Protocol Independent Mutlicast-Sparse Mode (PIM-SM): Procotol Specification. *IETF RFC 2362*, Jun. 1998.

[6] D. Li and D. R. Cheriton. OTERS (On-Tree Efficient Recovery using Subcasting): A Reliable Multicast Protocol. In *Proc. IEEE International Conference on Network Protocols (ICNP'98)*, pages 237–245, Oct. 1998.

[7] E. W. Zegura, K. Calvert and M. J. Donahoo. A quantitative comparison of graph-based models for internet topology. *IEEE/ACM Transactions on Networking*, 5(6), Dec. 1997.

[8] H. W. Holbrook and D. R. Cheriton. IP Multicast Channels: EXPRESS Support for Large-scale Single-source Applications. In *Proc. ACM SIGCOMM '99*, pages 65–78, Cambridge, MA, Aug. 1999.

[9] J. Moy. Multicast Extensions to OSPF. *IETF RFC 1584*, Mar. 1994.

[10] J. Moy. OSPF Version 2. *IETF RFC 2328*, Apr. 1998.

[11] K. Calvert and E. Zegura. GT-ITM: Georgia Tech internetwork topology models. *http://www.cc.gatech.edu/fac/Ellen.Zegura/gt-itm/gt-itm.tar.gz*, 1996.

[12] K. P. Birman, M. Hayden, O. Ozkasap, Z. Xiao, M. Budiu, and Y. Minsky. Bimodal Multicast. *ACM Transactions on Computer Systems*, 17(2):41–88, 1999.

[13] L. Wei and D. Estrin. The Trade-offs of Multicast Trees and Algorithms. In *Proc. ICCCN '94*, San Francisco, CA, Sep. 1994.

[14] M. Handley, I. Kouvelas, T. Speakman, and L. Vicisano. Bi-directional Protocol Independent Multicast (BIDIR-PIM). *IETF Draft*, Jun. 2003.

[15] M. J. Lin and K. Marzullo. Directional Gossip: Gossip in a Wide Area Network. In *Proc. of European Dependable Computing Conference (EDCC-3)*, 2000.

[16] M. Medard, S. G. Finn, R. A. Barry, and R. G. Gallager. Redundant Trees for Preplanned Recovery in Arbitrary Vertex-Redundant or Edge-Redundant Graphs. *IEEE/ACM Transactions on Networking*, 7(5):641–652, Oct. 1999.

[17] R. Yavatkar, J. Griffioen, and M. Sudan. A Reliable Dissemination Protocol for Interactive Collaborative Applications. In *Proc. ACM MULTIMEDIA'95*, pages 333–344, Nov. 1995.

[18] S. Banerjee, B. Bhattacharjee, and C. Kommareddy. Scalable Application Layer Multicast. In *Proc. ACM SIGCOMM*, pages 205–220, Pittsburgh, PA, Sep. 2002.

[19] S. Banerjee, S. Lee, B. Bhattacharjee, and A. Srinivasan. Resilient Multicast using Overlays. *Proc. ACM SIGMETRICS '03*, pages 102–113, Jun. 2003.

[20] S. D. Nikolopoulos, A. Pitsillides and D. Tipper. Addressing Network Survivability Issues by Finding the *K*-best Paths through a Trellis Graph. In *Proc. IEEE INFOCOM '97*, volume 1, pages 370–377, Kobe, Japan, Jun. 1997.

[21] S. Floyd, V. Jacobson, C. Liu, S. McCanne and L. Zhang. A Reliable Multicast Framework for Light-Weight Sessions and Application Level Framing. *IEEE/ACM Transactions on Networking*, 5(6):784–803, Dec. 1997.

[22] S. Han and K. G. Shin. Fast Restoration of Real-Time Communication Service From Component Failures in Multihop Networks. In *Proc. ACM SIGCOMM '97*, pages 77–88, Cannes, France, Sep. 1997.

[23] S. Paul, K. Sabnani, J. Lin, and S. Bhattacharyya. Reliable Multicast Transport Protocol (RMTP). *IEEE Journal on Selected Areas in Communications*, 15(3):407–421, Apr. 1997.

[24] UCB/LBNL/VINT. Network Simulator – ns2. *http://www.isi.edu/nsnam/ns/index.html*, Mar. 2002.

[25] X. Wang, C. Yu, H. Schulzrinne, P. Stirpe, and W. Wu. IP Multicast Fault Recovery in PIM over OSPF. In *8th International Conference on Network Protocols (ICNP'2000)*, Osaka, Japan, Nov. 2000.