A Proactive Tree Recovery Mechanism for Resilient Overlay Multicast

Supriya Chaluvadi
priya.544@gmail.com

Poorna Satyanarayana B.
poornasatyanarayana@gmail.com

Abstract - Overlay multicast constructs a multicast delivery tree among end hosts. An important problem for making overlay multicast more dependable is how to recover from node departures in order to minimize the disruption of service to those affected nodes. In this paper, we propose a proactive tree recovery mechanism to make the overlay multicast resilient to these failures and unexpected events. The salient feature of the approach is that rescue plans for multiple non-leaf nodes can work together for their respective children when they fail or leave at the same time. Extensive simulations demonstrate that our proactive approach can recover from node departures much faster than reactive methods, while the quality of trees restored and the cost of recovery are reasonable.

Keywords - Failure recovery; overlay multicast; tree restoration.

I. INTRODUCTION

Overlay multicast (also known as application-layer multicast) implements the multicast functionality at end hosts rather than routers. It uses a virtual overlay network topology based on the underlying unicast mechanism to transport data between end hosts. This may require more overall bandwidth than IP multicast because duplicate packets travel the same physical links multiple times, but it provides an inexpensive, deployable method of providing point-to-multi point communication. One example application using overlay multicast is live media streaming, which constructs a delivery tree at the application layer from the source to all participating end hosts. Usually it lasts for a long period of time and needs significant bandwidth constantly. We consider making overlay multicast dependable as a key factor that will determine whether it will be accepted as a fundamental infrastructure for group communications. The specific aspect of dependability we will study in this paper is resilience to failures and unexpected events. In overlay multicast, a node may fail or simply leave the multicast session voluntarily. The key issue is how to reconstruct the overlay tree after these unexpected events. The time to resume the data flow after a node departure (failure or leave) is important for multicast applications such as live media streaming. In traditional network-layer multicast, the non-leaf nodes in the delivery tree are routers, which are relatively stable and do not leave the multicast tree unless there are no downstream nodes. End hosts join the multicast tree only as leaves, and their departures do not require restoring the delivery tree. However, this is not the case for overlay multicast. When a non-leaf end host leaves the multicast session, all the nodes in the sub tree rooted at it are affected.

In this paper, we propose a proactive approach to the recovery of overlay multicast tree. The basic idea is that each non-leaf node in the overlay multicast tree pre-computes a rescue plan before it fails or leaves. The rescue plan for node is to pre-calculate a parent-to-be for each of its children. Once the departure of really happens, all of its children can contact their respective “parents-to-be” immediately. The challenges with this approach and their solutions are given below.

First, we need to consider the degree constraints in overlay multicast. They can be easily observed in streaming applications. With the degree constraints, finding the rescue plan for node becomes more complicated. We cannot use the parent of (i.e., the grandparent of x’s children) as the parent-to-be for all of x’s children as used in the network-layer fault-tolerant multicast routing protocol, because this may violate the degree constraint of the grandparent.

Second, we need to be able to deal with multiple departures that happen at the same time. The interference between their rescue plans should be minimized. Siblings may have their independent rescue plans, such as letting their children contact the grandparent. When they both fail, the collective effect may result in the traffic concentration at the grandparent and violation of its degree constraint. The idea of our scheme is to refrain from increasing the number of children of x’s parent after x leaves.

Third, we need to be able to deal with the cases in which the rescue plan is not available at a child. In these situations, the ancestor list will be used for these nodes to find a new parent as described in the Recovery protocol. Our approach also tries to use the degree information about downstream nodes and redirect JOIN request to a subtree with a high probability of success.

II. THE PROBLEM OF RESTORING OVERLAY MULTICAST TREES

The focus of this paper is on the problem of restoration of the overlay multicast tree after nodes leave or fail, especially those non-leaf nodes in the tree. When a node leaves the multicast tree gracefully, it may send a message to inform affected nodes. Usually the node will leave immediately after sending the notification message. We do not expect that they will continue forwarding packets to downstream nodes during the process these affected nodes try to reconnect to the tree. So fast recovery is still important in this case to minimize disruption of service to the nodes involved. When a node fails, we assume that other nodes can detect the failure by some heartbeat mechanisms. We have the same problem of restoring the
overlay multicast tree after failures or voluntary leaves. Therefore, we use “fail” and “leave” interchangeably in the rest of the paper to mean either or both cases.

In Fig. 1, we show an example of tree containing 24 nodes. By taking this tree as an example we will analyse what happens when a node leaves the tree. Suppose node 5 fails, then how to recover is shown below.

III. THE PROACTIVE RECONSTRUCTION OF OVERLAY MULTICAST TREES

Pre-computation Process

This is the internal process invoking rescue plan. Using this module the node will collect the information about the parent and child node. The information will be concerned with the degree of nodes, as every tree contains some ingress and egress degree. So the target of this module is to collect the degree information of the nodes.

The difference between the multicast tree in use and the spanning tree to be calculated. We do not actually add the edges in the calculated tree to the multicast tree in use. The only purpose of the result is to find the parent-to-be for each child. The difference from the problem in restoration process is that we cannot use the total degree of each node as its degree constraint in the tree to be calculated, because these nodes already connect with other nodes. So we define the used degree of a node as the number of (in and out) edges connected to the node in the current multicast tree, and the residual degree as the difference between the total degree and the used degree.

The residual degree reflects how many more edges that a node can be connected to. The total degree, the used degree and the residual degree of node are represented by \( d_t(x), d_u(x) \) and \( d_r(x) \), respectively. Obviously \( d_t(x) = d_u(x) + d_r(x) \). Also, we expect that \( d_u(x) \geq 0 \) and \( d_r(x) \geq 0 \).

In the example in Fig. 1, the used degrees of nodes 8, 9, and 10 are \( d_u(8)=4, d_u(9)=3, d_u(10)=4 \). If we assume that the total degree of each node is 4, the residual degrees will be \( d_r(8)=0, d_r(9)=1, d_r(10)=0 \). We now describe the algorithm.

Node leaving

This will show the process of where the nodes will plan to leave and it will show the process of node leaving the tree after pre-computing to-be-parent.
Proactive Approach

This will show the process of proactive approach. The idea of the proactive approach is to find a rescue plan before the failure happens. For each non-leaf node, it must find parents-to-be for all of its children. Once it fails, each of its children will know exactly who should be its new parent. One single message from each child will complete the recovery process. This is much faster than the reactive approach, where all children will go through a process of finding an appropriate node to be their parents. Two decisions have been made during the formation of the problem.

First, we want to make use of existing parent-child relationships as much as possible, to reduce the impact of the node failure. Therefore, we only find the parents-to-be for the children, instead of all descendants. For example, in Fig. 1, node 5 only plans the parents-to-be for nodes 8, 9, and 10. All other descendants (15–22) will keep the current parent.

Second, we want to keep the impact on the grandparent low. A naive method is to let all the children of the failed node contact the grandparent. However, this may quickly exhaust the degree available at the grandparent, and then the grandparent has to redirect them to other nodes. The affected nodes will experience a longer period of service disruption. Moreover, when siblings, such as nodes 5 and 6 in Fig. 1, fail at the same time, their rescue plans may no longer work for their respective children because of their interference.

For example, if the rescue plan is generated using the current available degree at the grandparent, it may work in case of a single node failure. However, when two sibling nodes fail at the same time, their children will compete with each other for the available degree at the grandparent, and the rescue plans may not be able to work together. This point will be further explained at the end of this section. Therefore, we make the decision to have only one child contact the grandparent so that the degree of the grandparent will be the same before and after the node failure. The idea is to have one child use the degree freed by its parent, and therefore the degree requirement at the grandparent will not be increased. If a non-leaf node (say) leaves the tree, the subtree rooted at it will become a forest of smaller subtrees, with each of ‘s children being a root of one subtree. A simple way to find a parent-to-be for each child is to form a spanning tree among the roots of these subtrees.

Our strategy in this case is to find descendants of nodes 8, 9, and 10. Those with a non-zero residual degree can contribute to the construction of the spanning tree. For example, we can find node 16, which has a residual degree. We then form a spanning tree among node 2 and the three subtrees, by adding edges between nodes 2, 8, 9, 10, and 16. We give an example spanning tree in Fig. 3. Node 5 will derive that the parent-to-be of node 8 is node 9, the parent-to-be of node 9 is node 2, and the parent-to-be of node 10 is node 16.

An observation about the protocol is that it can deal with most multiple failures cases efficiently. For example, in Fig. 1, if both nodes 5 and 7 fail, their respective sets of children will be able to recover independently. Even if two siblings fail at the same time, their children can still find their parents-to-be immediately without interference. Consider a case in which nodes 5 and 6 in Fig. 1 both fail. Fig. 6 shows the rescue plans for node 5 and node 6. Once nodes 5 and 6 fail, their children will be able to find their respective parents-to-be. We notice the importance of keeping the degree requirement for the grandparent the same before and after the node failure. The children of node 5 and the children of node 6 will not compete for the resources at their grandparent in this case. Every node can find their respective parents-to-be after node failures successfully.

Applicability of the Proactive Approach

The failure recovery problem is formulated for multimedia streaming applications, which impose strict degree constraints on nodes in the multicast tree. However, the proactive approach can potentially be extended to other overlay multicast mechanisms for tree recovery. The applications without the high bandwidth requirement may have very loose or no degree constraints on nodes. The proactive approach can still be used to improve the performance (e.g., the quality of the tree recovered) by finding a good contact point for a node.
without sacrificing the recovery time. The formulation of the recovery problem has to be modified to reflect the new situation, and possibly new optimization criteria, such as the average delay from the root and the total cost of the tree.

IV. PERFORMANCE EVALUATIONS

We evaluate the performance of the proactive approach using simulations. We are mainly interested in the responsiveness of various approaches, i.e., how fast the delivery tree can be restored. Other interesting measures are the quality of the tree reconstructed, and the overhead of the recovery process. We compare our proactive scheme with several reactive methods, including grandfather, grandfather-all, root, and root-all.

V. RELATED WORK

Overlay multicast networks have been studied extensively in recent years. Degree constraints are considered in establishing multicast trees. Various degree-constrained spanning tree problems are defined and usually they are NP-complete. The proactive approach has been used in recovering link or node failures in multicast tree in the context of the traditional network-layer multicast. The fault-tolerant multicast routing proposed to use backup paths from the grandparent to deal with link or path failures. More recently, a dual-tree scheme [30] proposed that in addition to the primary tree normally used, a secondary tree is calculated that connects the leaf nodes in the primary tree together. Once a failure occurs, it will activate a path in the secondary tree. These schemes differ from our work in that they dealt with the network-layer fault-tolerant multicast routing problem and did not consider the important degree constraints on the nodes in overlay multicast.

The problem of dealing with node failures in overlay multicast has been recognized in more recent work. Spread It proposed reactive strategies to deal with node leave or failure in overlay multicast. They find appropriate places in the subtree of the grandparent or the root for the affected nodes after failure happens. Because it uses the reactive approach, the time to find an appropriate place may be long and those affected nodes may even compete with each other. Also, it did not mention the degree limit of the nodes.

Resilient multicast using overlays uses a proactive method for overlay multicast. It is complementary to our work because it works on the data plane while our scheme works on the control plane. Our scheme generates less extra traffic, and we also discuss the restoration of the multicast tree. We can make use of their idea of probabilistic forwarding to improve the transient behavior of our scheme.

Establishing multiple multicast trees is another approach to dealing with failures in overlay multicast. Coop Net used a simple, centralized tree management algorithm to construct and maintain a diverse set of trees. A node is usually in multiple trees, and the sets of its ancestors in different trees are made as disjoint as possible. The streaming content is encoded using multiple description coding (MDC), and the descriptions are distributed over different trees. Therefore, when a non-leaf node fails or the congestion occurs in a part of a tree, those affected nodes will not be able to get the descriptions sent from this tree, but they will still be able to receive the descriptions delivered from other trees because of the disjointness of ancestor sets for each node. Therefore, the multiple tree approach can deal with not only the “hard” failure (node failure or leaving), but the “soft” failure (congestion) as well. Instead, the proactive approach proposed in this paper focuses on dealing with the “hard” failure, i.e., how to reconstruct the tree.

The affected nodes will be able to contact with the parent-to-be directly in most cases and get data delivered to them. This will reduce the time of finding an appropriate new parent by reactive approaches. For example, we can save the round-trip time to the central server for getting information about its new parent. Besides, in formulating the recovery problem, we make an effort to optimize performance measures (cost, or delay) of interest to the applications. It is easier to keep these measures up to date using the distributed solution proposed in the paper. Finally, the multiple tree approach and the proactive approach are complementary to each other in that the former enable users to get media streams even in the case of failures or congestions, while the latter speeds up the actual tree reconstruction process in the case of failures.

VI. CONCLUSION

Overlay multicast differs from traditional IP multicast in that the problem of degree constraints is more prominent and nonleaf nodes in the multicast tree are unstable. This makes the problem of restoring multicast tree after node failures or leaves quite different. This paper proposed a proactive approach in which each non-leaf node precomputes the recovery plan for its children. We developed a protocol for nodes to communicate with each other and deal with various failure situations. The recovery process is much faster than the reactive approaches while the quality of the tree and the amortized cost is comparable to those methods.

REFERENCES


AUTHOR’S PROFILE

Supriya Chaluvadi
Dept. of Computer Science and Engineering,
Chaitanya Engg. College, Vishakapatnam,
Andhra Pradesh, India

B. Poorna Satyanarayana
Professor,
Dept. of Computer Science and Engineering,
Chaitanya Engg College, Vishakapatnam,
Jawaharlal Nehru Technological University, Kakinada,
Andhra Pradesh, India