AN INNOVATIVE AND EFFICIENT MULTIPATH MULTICAST MECHANISM FOR DATA STREAMS

Safiullah Faizullah¹, and Arshad M. Shaikh²
¹Hewlett-Packard, Piscataway, NJ, USA
²Isra University, Hyderabad, Pakistan
Email: safi.research@gmail.com; amshaikh@hotmail.com

ABSTRACT
With the advancements in networked applications that multicast (such as audio, video and teleconferencing) and with increased requirement for more network resources (such as bandwidth), there is a need for better ways to deliver data so that the network resources are efficiently utilized. This paper presents a multipath multicast technique that creates multiple multicast flows to provide more aggregate bandwidth for delivery of the data. Formally, the purpose of this work is to devise an algorithm, which efficiently divides the multicast flow (from one sender to many receivers) into sub-flows and construct trees corresponding to each of these sub-flows so as to maximize the total inflow at all the receivers. We refer to this network optimization problem as Multipath Multicast Problem (MMP) and establish that it is NP-Complete in general. We offer an approximation algorithm to solve MMP. Furthermore, we present simulation results for randomly generated networks to evaluate MMP.

KEY WORDS
Multipath; Multicast; Data Streams; Network Optimization; Efficient Delivery; Flows; Sub-Flows.

1. Introduction and Background
Recently, tremendous advancements have been made where many applications that use multicast (such as audio, video and teleconferencing) are becoming popular and need of everyday life activities [1-3, 5-6, 8, 10-14]. That combined with increased requirement for more network resources (such as bandwidth), there is a need for better ways to deliver data so that the network resources are efficiently utilized. Many recent and emerging applications require some quality-of-service (QoS) requirements as maximum end-to-end delay and minimum bandwidth resources. Several techniques have emerged over the past decades from those that use unicast to transmit traffic flows [10, 13], with others improved techniques which were proposed where the flows were divided to sub-flows for better results [16], to the newer techniques that use multicasting. Multicast connections are connections between one or more senders and a number of members of a group which suites very well to these applications. As the main aim of multicasting is to be able to send data from a sender to the members of a group in an efficient manner, this mode of communication is important due to the increased use of new point to multipoint applications, like web-based e-learning, web-based radio and TV, videoconferencing and other collaborative environments, and movies/video on-demand video. Traditional multicasting techniques were restricted by the one/few bottlenecked links and hence quality of the traffic was degraded for all receivers. Researchers have proposed to use multipaths of equal bandwidth to enhance the results but this approach is limited in nature. Only recently we have seen work on general multipath multicasting to achieve enhanced data delivery for QoS-aware applications [6, 11, 12, 15].

The concept of using multipath to aggregate bandwidth is not new with some work done on multipath in unicasting. To the best of our knowledge, through an extensive survey, MMP has not been extensively and widely addressed in the literature with some recent work giving the much needed attention to this important research area [6, 15]. There has been some limited work on multipath multicast for equal weighted paths and with other restrictions [1-3, 5-6, 8, 10-14] but robust techniques are still not widely proposed [15]. This paper presents a multipath multicast (MMP) technique that creates multiple multicast flows to provide more aggregate bandwidth for delivery of the data.

1.1 Motivations and Informal Description of the Proposed Approach
Consider the network shown in Fig. 1 where a source S needs to multicast a data stream to receivers R1, R2 and
R3. Routers A, B, ..., G are intermediate routers and they are not interested in getting the complete flow. The labels along each link shows the available (residual) bandwidth on that link in Kbps. Suppose that the bandwidth needed for the multicast is 4 kbps. Clearly, no single multicast tree exists which can deliver the data to receivers at the required rate. The maximum bandwidth that receivers R1, R2 and R3 can achieve is 3, 4 and 2 kbps respectively. In some multicast protocols the sender will adjust its rate to match the minimum which in our example is 2 kbps which degrades the quality of service. However, if we divide the flow of 4 kbps into sub-flows and send them along several multicast trees then it is possible to get the required bit-rate (or at least improve upon the minimum link). This is shown in Fig. 2 with three sub-flows – red, green and blue of bandwidths 2, 1 and 1 kbps respectively. Each sub-flow has a multicast tree of its own. Note that these trees may overlap on certain links.

The concept of using multipath to aggregate bandwidth is not new. It is already being employed, in a limited form in several unicast protocols. For example, OSPF uses equal cost multi-paths by keeping all the next hops in the routing table that lead to the same destination and forward the packets in a round robin formation.

Introducing multipath in multicasting is more complex than unicasting because of the construction and manipulation of multiple multicast trees within limited table space, and processing. In particular, the hard problem is to divide the flow into sub-flows and construct their corresponding multicast trees such that the trees do not block each other as much as possible, so as to maximize the overall satisfaction of the receivers. For example, in Fig. 2 if the bandwidth of red, green and blue sub-flows be 1, 1 and 2 Kbps then the green sub-flow will be blocked on the link S-F and therefore receivers R3 will not be able to receive this sub-flow ever, while R1 will be able to receive it via A-B-D-R1 with a larger delay.

Note that there is still a possibility that some receivers do not get the complete flow, for example in Fig. 3, R4 cannot receive more than 2kbits and in Fig. 4, one of R1 and R2 will be blocked by the other and will receive 1kbits less.

However, sending data over several trees assures that the receivers not getting the complete flow will not slow down the transmission for other receivers. This is because each tree reaches only those receivers to whom it can fully deliver the corresponding sub-flow. In fact, the technique we are describing in this document construct trees in an order (iteratively from residual graph) such that the subsequently created trees cannot reach more receivers than any tree created prior to them. Hence there is an implied precedence/hierarchy in the sub-flows which we can exploit to send layered streams like mpeg video. The most important data packets (like I-frames of mpeg) should be sent over the first tree (or initial trees) and less important data be sent over the subsequent trees.

The proposed technique uses the link-state information at the sender (source router) to create multiple multicast trees and deploy them in the network using source-routing strategy like MPLS [17]. Since our approach does not target the generic multicast routing problems, rather it focuses in providing more bandwidth to non-delay-critical applications like simplex (one-way) audio and video, therefore we cannot impose it to all types of multicast. Hence the use of source-routing is justified. In addition to that, MPLS switches packets at layer 2 instead of forwarding at layer 3 hence resulting in a further improved performance.

The purpose of this work is to device an algorithm which efficiently divides the flow into sub-flows and constructs multicast trees corresponding to each of these sub-flows so as to maximize the sum of all inflows at all receivers; we will refer to this problem as “Multipath Multicast Problem”, MMP.

1.2 Aims and Objectives of the Undergoing Research

We intend to study a number of results to evaluate the proposed technique using simulation.

![Figure 1. Sample Network with Available Bandwidth](image-url)
The following is the list of characteristic and present simulation-based evaluations of the results which are compared with existing multicast schemes:

- Overall users’ satisfaction (quality of video, bandwidth) - we also show ratio for each receiver vs. overall;
- Network Throughput and traffic concentration - maximum network traffic on any links as comparison to mean throughput;
- Network Resource Usage, total number of hops for all of the sub-flows;
- End-to-end packet Delay (worst case).

We will study some other aspects of MMP and simulation-based evaluations of the results will be presented in future work.

The rest of the paper is organized as follows: Section 2 briefly presents the Multipath Multicast Problem formulation. Our proposed Multipath Multicast solution is discussed in Section 3. Section 4 presents experimental methodology and simulation results. Finally, Section 5 gives some concluding remarks.

2. Multipath Multicast Problem Formulation

We formally define the multipath multicast problem as the following network optimization problem, Given a directed graph \( G(V, E) \), where \( V \) is the set of nodes with a designated node as sender and a subset \( R = \{r_1, r_2, \ldots, r_n\} \) of receiver nodes and \( E \) is the set of edges. An edge \( e \in E \) connecting nodes \( x \) and \( y \) will be denoted by \( (x, y) \). Let \( B(e) \) denote the available bandwidth on each edge \( e \) in \( E \). Further, let \( B(F) \) be the bandwidth required by the multicast flow \( F \) from \( S \) to \( R \). Now the objective is to find a flow-partition \( P = \{F_1, F_2, \ldots, F_m\} \) of sub-flows of \( F \) together with corresponding set of multicast tree \( T = \{T_1, T_2, \ldots, T_m\} \); each tree \( T_i \) being sub-graph of \( G \); such that for every edge \( e \in E \), the combined bandwidth of all the sub-flows whose trees contain \( e \) does not exceed the available bandwidth of link \( e \). That is,

\[
\text{For all } e \in E, B(e) \geq \sum_{n=1}^{m} B(F_i) \text{if } e \in T_i
\]
Let $\text{InFlow}(v, P:T)$ be the total inflow for node $v$ with respect to partitioning $P$ with corresponding set of trees $T$, calculated as:

$$\text{InFlow}(v,P:T) = \sum_{i=1}^{n} \sum_{j=1}^{m} B(F_i) \times \begin{cases} 0 & \text{if edge } (v_j, v) \text{ not in } T_i \\ 1 & \text{if edge } (v_j, v) \text{ in } T_i \end{cases}$$

The desired solution $(P:T)$ is subject to the optimality condition that the sum of all inflows at all the receivers should not be less than any other partitioning $P'$ with any other corresponding set of multicast trees $T'$. That is, for all possible $P':T'$

$$\sum_{i=1}^{n} \text{Inflow}(r_i, P:T) \geq \sum_{i=1}^{n} \text{Inflow}(r_i, P':T').$$

### 2.1 MMP is an NP-Complete Problem

It has been established that determining an optimal multicast tree for a static multicast group can be modeled as the NP-complete Steiner problem in networks [7]. It is easy to see that finding multiple optimal multicast trees of MMP can be reduced to finding a single optimal multicast tree for a static multicast, which is a special case where there is only one workflow and no other sub-flows.

Hence, it follows that MMP is a NP-complete problem. In addition, we strongly believe and conjecture that our problem of finding multiple multicast trees is even harder problem. An approximate solution is therefore needed to solve MMP.

### 3. The Proposed Multipath Multicast Solution

Before we present our algorithm to solve MMP, we would like to define few terms.

*Deferred FanOut Directed Steiner Tree (DF-DST):* A directed spanning tree rooted at node $S$ is termed as Deferred FanOut if the branches are made as away from $S$ as possible. Ideally, there are no branches in such a tree as all the nodes lie on a single path starting at $S$. Note that this is nothing but the depth-first traversal tree of the graph. If there is only a subset $R$ (of the set of nodes $V$) to be spanned, then the tree is not just the depth-first traversal tree but it is the Steiner tree for $R$ in $G$ with minimum number of nodes outside $R$. We refer to such tree as $S$ FanOut Directed Steiner Tree.

The advantage of constructing Deferred FanOut DST over normal DST is that it tries not to block other paths from the root $S$ to the vertices in $R$.

#### 3.1 MMP Algorithm

Now we present our algorithm which will be executed periodically at the sender.

**Step 1:** $I := 0$; $BW_{sofar} := 0$; residual graph

**Step 2:** $I := I + 1$

**Step 3:** Construct a Deferred-FanOut Directed Steiner Tree $T_i$ in $G'$ rooted at $S$ that spans all those nodes in $R$ which are connected to $S$ in $G'$

**Step 4:** $B(F_i) :=$ the bandwidth on the minimum capacity branches of tree $T_i$.

**Step 5:** $BW_{sofar} := BW_{sofar} + B(F_i)$

**Step 6:** Update residual graph $G' := G' - T_i$

**Step 7:** If (any node in $R$ is connected to $S$ and $BW_{sofar} < B(F)$) then goto step 2

**Step 8:** End.
3.2 NP-Completeness and an Approximate Solution

The construction of DST in Step 3 is a well-known NP-complete problem; therefore an approximate solution is needed. We propose to use the following heuristics:

Do a best-first traversal of G' starting from S with cost of each alternative as follows:

- Cost = 0 if without branching proceed to a node in R
- Cost = -1 if without branching proceed to node outside R
- Cost = -C if branching.

The value of parameter C determines the length of linear paths (in hops) to be considered before any branch is taken. As indicated above, longer linear paths will help reduce the chance of blocking other paths from the source S to the receivers in R. Network administrator at the multicast source may set the value of parameter C that gives the required quality (delay vs. bandwidth). We will be studying different delay vs. bandwidth scenarios for several network topologies for a wide range of network applications.

3.3 Complexity Analysis

In this section we will present only a brief discussion on the complexity of our algorithm. The complete analysis will be given in a later report.

The complexity of performing a Best-First search on G'(V,E') is $O(|V'|+|E'|) \leq O(|V|+|E|)$.

Further, since the construction of residual graph in each iteration will remove at least one edge (the edge of minimum capacity in the tree $T_i$) therefore, the main algorithm will iterate at most $O(|E|)$ times. Hence the worst case complexity of the complete algorithm is $O(|E||V'|+|E|^2)$.

In a dense network where each node is connected to almost every other node, $|E|$ is estimated as $|V|^2$. And hence the worst case performance of the proposed algorithm is $O(|V|^4)$. However, since real networks are not normally dense and $|V| = O(|E|)$, therefore, the average running time of our algorithm comes down to $O(|V|^2)$. 
4. Simulation Setup

To conduct the simulation studies, we will be using randomly generated networks of varying complexities and sizes. This ensures that the simulation results are independent of the characteristics of any particular network topology. Using randomly generated network topologies also provide the necessary flexibility to tune various network parameters such as average degree, number of nodes, number of edges, and to study the effect of these parameters on the performance of the technique. To generate random graphs, we will use a method similar to [9]. Sparse graphs, containing a small percentage of the total number of possible edges, with low average degrees are more representatives of real networks and pose a tougher problem, in general, to Steiner heuristics. To generate node addition and node deletion requests to multicast groups, we will employ probabilistic model similar to [9] that allows control of the relative frequencies of add and delete requests.

The results presented in this paper are based on averages for 10-20 iterations of same network setup (topology, traffic, groups, etc.) Implementation details including overhead for multicast protocols are removed from results, the effect of which will be studied in future. We also impose a bound on the number of sub-flows that we create.

We have incorporated a traditional Multicast algorithm, referred to as MP, and our MMP algorithm into a simulator that we will simply refer to as SIM in this paper. We also added multipath multicast algorithm which supports only equal weight paths, referred to as MMeP, to further study the strength of our MMP algorithm.

The following is the list of results which we compare with existing multicast schemes:

A. Overall users’ satisfaction (quality of video received) - we also show ratio for each receiver vs. overall. For this we used a utility function that we developed in [4]. The utility function is simple where the utility function takes the following form:

\[ U(\sigma_{QoS}) = L(\sigma_{QoS}) - C(\sigma_{QoS}) \]

Where \( L(\sigma_{QoS}) \) is the apparent quality degradation level (in favor of other users) to a user while \( C(\sigma_{QoS}) \) is the price one is charged for. \( U(\sigma_{QoS}) \rightarrow +R \) is a mapping from a non-negative real number to non-positive real number showing the worth/quality perceived by a user (for a usage request with quality given by \( \sigma_{QoS} \) and for which cost \( C(\sigma_{QoS}) \) was paid). For this study we adopted the following definition:

\[ L(\sigma_{QoS}) = -\text{number of packets that did not meet.} \]

Fig. 5 shows users’ satisfaction as a function of load for the first configuration (utilizing the utility function shown above.) The figure shows that users’ satisfaction (y-axes) deteriorates (normalized for all users), as we increase the offered load (x-axes), in case multicast algorithm (MP) without sub-flows capabilities is employed. This is because packet drops as network becomes congested (due to high load). The proposed MMP scheme, as the offered load reaches the capacity of the bottleneck link, still maintains high percentage (23% over that of simple MP) of satisfied users.

In Fig. 6, users’ satisfaction is shown for our scheme vs. MMeP (multipath multicasting with equal weight paths.) The figure shows that users’ satisfaction deteriorates (normalized for all users), as we increase the offered load, particularly when fixed MMeP is employed. This is due to the lack of this scheme to take advantage of paths of different weights. As such when several flow traffics are in active mode, and network becomes congested (due to high load), MMeP perform poorly. The proposed MMP scheme, on the other hand, retains 17% more satisfied users’ base, which is a huge performance advantage in comparison to MMeP. Note that MMeP performs better than MP.
Fig. 7 shows the end-to-end delay (in seconds) for each of the three algorithms (y-axes) as the number of receivers increase (x-axes.) It is apparent that MMP does very well as compared to MMeP and MP. We can note that MMeP does worse than MP as it loses time trying to find multiple paths of equal weight only.

5. Conclusion

In this document we have introduced the notion of a Multipath Multicast Problem (MMP) and indicated that it is NP-complete. We have proposed and investigated an approximate algorithm to solve MMP. Based on our simulation results, better overall users satisfaction was achieved while end-to-end delay was not increased- in fact MMP did better than both MP and MMeP.) The results obtained in this paper are equally significant in the graph theory. Our undergoing research includes studying the scalability and efficiency of the algorithm. The proof of NP-completeness and the proposed approximate algorithm has a general theoretical importance. There are a number of problems which can be reduced to this MMP and hence the proposed work is of general nature. We are exploring this aspect of our research as well.

5.1 Open Issues Under Investigation

We would like to conduct additional simulations and study the following:

- Network Resource Usage, total number of hops for all of the sub-flows.
Network Throughput and traffic concentration – maximum network traffic on any links as comparison to mean throughput.

In addition, the following are some of the open issues that we would also be studying in the undergoing research:

**Upper bound on the end-to-end delay**: The proposed algorithm tries to improve the bandwidth while ignoring the end-to-end delay. We need to study the upper bound of this delay and to explore ways to control or limit this increment.

**Upper bound on the number of sub-flows (number of trees)**: the proposed algorithm in the worst case divides the flow into $|E|$ sub-flows, which is a huge number. We need to derive conditions (for instance discarding edges from the residual graphs having bandwidth less than a threshold) so as to limit the number of sub-flows without compromising on the efficiency and usability of the algorithm.

**Online modification of trees**: we would like to study the possibility of a cost-effective algorithm that updates (add and delete nodes from) the multipath multicast trees online. This will save the invocation of full algorithm each time the group membership is changed. The online changes of the trees are generally small and hence cost little when deployed in the network (compared to the deployment of the complete trees). It is an established fact that online modification of Steiner trees is NP-complete.

**Triggering the execution of algorithm**: In this work, we have proposed that the algorithm is to be executed periodically at the source, so to accommodate for any change in the network topology and in the group membership. It is our goal to investigate other techniques of invoking this algorithm in order to avoid unnecessary executions. Such techniques may need some kind of threshold of changes to be reached before the algorithm is invoked.

**Effects of changing live MPLS paths**: we are also studying the effects of deploying the new set of trees (generated by MMP) during a live multicast session. Especially we need to investigate the chances of any packet loss due to this change.

**Ordering of packets received**: each alternative path may have a different latency involved. Having packets via alternative paths will arrive out of order. Packet recording could cause Reliable-Multicast-Mechanism to believe that loss has taken place hence invoking the loss-recovery mechanism. We need to explore techniques to overcome these issues.

**Scalability and Deployment issues**: Our approach is scalable as its complexity is comparable with other link-state routing algorithm- as it was indicated in the complexity analysis of the algorithm. The only deployment issue is that, it requires MPLS (or some other source-routing mechanism) to be implemented in the network.

**References**


