An Approach to Building Multicast Trees over MPLS

Deyan G. Razsadov[1], Peter Tz.Antonov[2]

Abstract – **MPLS is a new technology suitable for traffic engineering. In this paper we discuss some aspects of multicast switching over MPLS related to traffic engineering. Problems of building optimal multicast trees will be shown. A method to find the optimal multicast tree will be presented. The hierarchical approach of applying this method to large networks will be proposed.**

Keywords – **MPLS, Multicasting, Traffic engineering, Steiner Minimal Tree, Minimal Spanning Tree.**

I. INTRODUCTION

Implementation of traffic engineering has two main purposes: network resources usage optimization and providing Quality of Service (QoS) to users' traffic streams. It is impossible to achieve these goals without tight control over data flow, especially the path of the packets.

Best-effort protocols (e.g. IP) primary use hop-by-hop routing, where each router the packet goes through determines only its next step. Existing routing protocols base their next hop decisions mostly on shortest path algorithms, what may lead to non-optimal usage of network resources.

Multiprotocol Label Switching (MPLS) is a technology that is about to become a common core backbone transport for traffic engineering solutions. The idea of MPLS is mapping of level 3 paths to level 2 predefined Label Switched Paths (LSP) as described in [1]. When a packet enters MPLS based network, ingress Label Switching Router (LSR) analyzes its header and the packet is assigned to a Forwarding Equivalence Class (FEC). According to that FEC the packet is labeled and sent to next-hop LSR. At the end of LSP egress LSR removes the label and further routes the packet according to its level 3 header. Advantages of MPLS can be summarized:

- Simple and short label needs less processor overhead for making switching decisions;
- Predefined LSPs allows avoiding links capacity overusage through better distribution of data packets flow;
- FEC could be much more precise that destination network address, what allows us ability to provide different QoS to different types of traffic.

In [2] and [3] the framework of switching multicast traffic over MPLS and its traffic engineering issues are given. In this paper we discuss building explicit multicast trees for switching multicast traffic through MPLS.

[1]Deyan G. Razsadov is with the Computer Science and Engineering Dpt., TU, 1 Studentska Str., Varna 9010, Bulgaria, email: razsadov@yahoo.com

[2]Peter Tz. Antonov is with the Computer Science and Engineering Dpt., TU, 1 Studentska Str., Varna 9010, Bulgaria, email: peter.antonov@ieee.org

II. FORMULATION OF THE PROBLEM

In general, there are two types of multicast trees: point-to-multipoint (one host sending multicasts and one or more receiving hosts) and multipoint-to-multipoint (every end node sends multicasts to the others). Both of them must have the following features:

- All sending and/or receiving nodes must be on the tree.
- The tree must have no loops. Only one path between any two nodes on the tree must exist.
- Only host sending and/or receiving multicasts can be a terminal node on the tree (a node with only one connection to the tree).

To choose from all the possible trees we use weight of the tree, which is the sum of weights of all links involved. The most commonly used metric is weightened sum of some of the characteristics of the link:

$$M = \sum_{i=1}^{n} c_i \cdot p_i \,, \qquad (1)$$

where n is number of characteristics, c_i are weight quotients for parameters, and p_i are characteristics of the link. Most commonly used characteristics are: capacity (total and available bandwidth, free/used bandwidth percentage, etc), reliability (average packet loss, average time between failures, etc.), speed (average travel time, dispersion of travel time), cost. Varying weight quotients we can get different optimal trees as result.

Thus, the goal of optimal tree algorithm is to minimize the sum of metrics of links involved:

$$F = \sum_{i=1}^{l} M_i \rightarrow \min \,, \qquad (2)$$

where l is the number of links used.

The criterion (2) minimizes the total usage of network resources, even though the path between any two random nodes on the tree is not supposed to be optimal (shortest) according to the same criterion.

Finding an optimal multicasting tree is a Steiner Minimal Tree (SMT) problem. Its brief definition follows. Network is described by a complete graph $G = (V, E)$, consisting of vertices and edges, and a metric given by edge weights $M : E\{V_i \rightarrow V_j)$. Nodes that should form a multicast tree are subset $T \subseteq V$. Solution S is a subgraph of G that includes all the vertices in T ($T \subseteq S \subseteq V$) and has the minimal sum of the weights of its edges.

III. BUILDING AN OPTIMAL TREE

If we have vertices of S determined, then the minimal spanning tree for S will be the optimal solution of the SMT problem. Unfortunately, at this moment there are no methods or algorithms that allow analytically determine the subset S. The only way to find an exact solution is calculating all the possible combinations. If N is the number of vertices in G, and K – number of vertices in T, then there are 2^{N-K} possible combinations to try.

In order to find a valid solution for given combination of vertices $S'(T \subseteq S' \subseteq V)$, a spanning tree algorithm should be applied. S' leads to a valid solution if a spanning tree connecting all the vertices of T exists. Steps in finding a solution are:

- Choose some S'.
- Start with a node that belongs to T.
- Apply Prim's minimal spanning tree algorithm rules until all vertices of T are added to the tree.
- Remove all vertices that were not connected to the tree.
- Remove all terminal vertices that do not belong to T until there are no such vertices.
- Calculate the weight of remaining subgraph S. If it is less than minimal weight calculated on previous iterations, it becomes the minimal weight and S becomes the currently optimal solution
- Repeat previous steps until all possible S' are checked.

This algorithm always leads to an optimal solution, but needs a lot of processor resources. The following features could be used to make checking the combination faster:

If it is not possible to complete the minimal spanning tree algorithm, there is no a valid solution for S' and for any S^*, such that $S^* \subseteq S'$.

If a valid solution S ($T \subseteq S \subseteq S' \subseteq V$) is found, it will be the solution for any S^*, such that $S \subseteq S^* \subseteq S'$.

IV. HIERARCHICAL APPROACH

Trying of all the possible S' for a large graph needs unaffordable amount of time even after some optimization. We propose using a different approach hierarchical approach in this case.

To be able to try all possible solutions for reasonable time, the number of nodes in graph G should be reduced. Combining nodes connected with low-weight links into one virtual node is one of the ways to do it. All outgoing links of joining nodes are combined too. If more than one link between two nodes exist, only the link with lowest weight should remain. If we combine a node belonging to T, the new node will belong to T too. After reducing the number of nodes to reasonable amount, the algorithm described in section III is applied. For every combined node used in calculated solution decomposition should be made by using the same algorithm until no combined nodes remain.

The multicast tree calculated by this approach may not be the optimal one, but will be near to optimal enough for the most practical applications.

Let's show how this method applies to the network of a typical company, which used MPLS as a core backbone transport. Such network might be distributed between several buildings, cities, or even countries. Part of network consisting of computer connected with fast links we will call "site". Usually those links are 10/100/1000 Mb LAN connections, which are privately owned by the company. They could be easily upgraded if they do not provide enough capacity. Users' computers at every site are connected to the local backbone through routers. All sites are usually interconnected by low speed WAN links through public networks. In hierarchical approach first calculation is made on graph where vertices represent sites and edges represent links between them. After that, calculation is repeated for routers of every site included in SMT. As much is the difference between capacity of intrasite and intersite links, as near to optimum calculated solution will be.

V. CONCLUSION

MPLS is an advanced forwarding scheme which extends routing with respect to packet forwarding and path controlling. The features of MPLS make it a good choice for traffic engineering for both unicast and multicast traffic. We introduced a hierarchical approach to calculation of an optimal multicast tree over MPLS.

A Steiner minimal tree is optimal only for that set of end nodes calculation is based on. Adding and removing nodes would enforce recalculation of entire tree. Such recalculation needs processor resources on offline traffic engineering server; causes more network traffic for reconfiguration of LSRs; during rebuilding the tree could become inoperative. This makes presented approach suitable mostly in case when multicast tree is expected to be relatively static. Although, it can still be used for background recalculation and reoptimization of the tree after new node is added by a fast shortest path algorithm.

Another alternative to hierarchical approach is using approximation methods that give a non-optimal solution, but near to theoretical optimum [4].

REFERENCES

[1] E.Rosen et al. "Multiprotocol Label Switching Architecture," RFC 3031, January 2001.

[2] D.Ooms et al. "Framework for IP Multicast in MPLS," Internet draft <draft-ietf-mpls-multicast-07.txt>, January 2002.

[3] D.Ooms at al. "MPLS Multicast Traffic Engineering," Internet draft <draft-ooms-mpls-multicast-te-01.txt>, February 2002

[4] Gabriel Robins, Alexander Zelikovski, "Improved Steiner Tree Approximation in Graphs".