On Finding the Best Partial Multicast Protection Tree under Dual-Homing Architecture

Mei Yang[†], Jianping Wang[‡], Xiangtong Qi[⋆], Yingtao Jiang[†]

[†] Department of Electrical and Computer Engineering, University of Nevada Las Vegas, NV 89154, USA

[‡] Department of Computer Science, Georgia Southern University, Statesboro, GA 30460, USA

[⋆] Department of IEEM, The Hong Kong University of Science & Technology, Hong Kong, China

E-mail: [†]{meiyang,yingtao}@egr.unlv.edu,[†]jpwang@georgiasouthern.edu,[⋆]ieemqi@ust.hk

***Abstract*—In this paper, we introduce the concept of partial protection and propose an efficient solution for providing partial multicast protection given the dual-homing architecture in the access network. In the dual-homing architecture, each destination is connected to two edge routers to enhance the survivability in the access network. The routing algorithm which finds a path from the source to each edge router holds the key for the multicast protection. We study the problem of finding the best partial multicast protection tree for the multicast session given the dual-homing architecture assuming that the hop count on each path is limited. We show the NP-completeness of the problem and propose the Partition and Sharing (PAS) algorithm to solve the problem efficiently. Simulation results show that the PAS algorithm achieves performance very close to the computed lower bounds. The solution proposed in this paper fills the gap between traditional 100% protection and non-protection subject to single link failure.**

I. Introduction

Multicast is a means of one-to-many or many-to-many communication scheme. Many bandwidth-intensive multicast applications, such as high-definition television, video and video conferencing, distance learning, etc., become widely popular with the advances in optical transmission technology. It is vital to efficiently protect critical multicast sessions against link or node failures. Yet protection is more challenging for multicast communications since one node/link failure will affect a number of multicast destinations. On the other hand, the large number of destinations in a multicast session certainly makes it harder to provide protection for multicast communications.

In the literature, several multicast protection schemes have been proposed to provide 100% protection against single link failure [3], [9], [10], [11], which requires disjoint paths from the source to each destination. Classified by the granularity of disjointedness, three general approaches can be applied. A straightforward way is to compute two link-disjoint multicast trees. One serves as the primary multicast tree, and the other serves as the backup multicast tree [9]. However, it is hard and even impossible to find two link-disjoint multicast trees for a large scale multicast tree. Alternative ways include segment protection [11], [12] and path protection [11].

Modern networks can no longer limit the options of providing protection only to the extreme cases: with 0% protection or with 100% protection against single link failure. Instead, partial protection should be provided against single link failure, which is to find two paths from the source to the destination with minimum shared links. If there is one link failure on the disjoint links along the two paths, protection can be provided, while if there is one link failure on the shared links along the two paths, protection can not be provided. Therefore, such two paths can provide partial protection again single link failure.

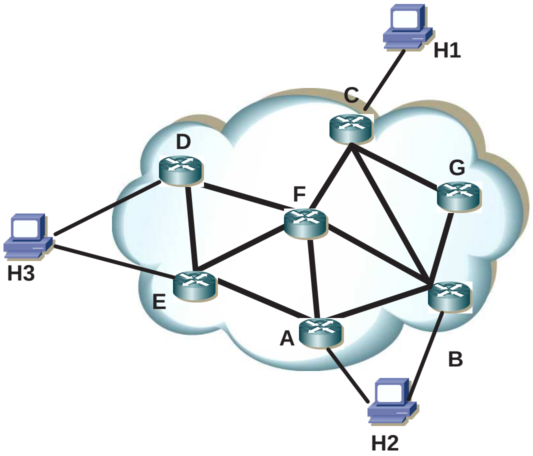

Fig. 1. An example of the dual-homing architecture.

In [13], we proposed a partial multicast protection scheme based on the *dual-homing* architecture, which was originally proposed to enhance survivability for the access network [5], [8]. In a dual-homing architecture, a host in the access network can be connected to two IP edge routers. Under such an architecture, the two paths from the source of the multicast session to the two edge routers provide certain degree of protection for the data traffic from the source to the destination. Figure 1 illustrates one example of dual-homing protection for a multicast session composed of source H1 and destinations of H2 and H3. H2 is connected to edge routers A and B. There are two paths from H1 to H2, H1-C-F-A-H2 and H1-C-B-H2. Since these two paths are disjoint in the core network, H2 can receive data from H1 irrespective of any link failure in the core network. H3 is attached to edge routers D and E. The two paths from H1 to H3 are H1-C-F-D-H3 and H1-C-F-E-H3. These two paths share a link C-F in the core network. If any link fails along the two paths except link C-F, H3 can receive data through the alternative path.

Clearly, the two edge routers to which a destination is attached determine the level of protection from the source to the destination. To quantify the protection level from the source to a destination, we introduced the concept of *vulnerability* [13], which was defined as the number of shared links between the two paths from the source to the two edge routers the destination can connect to. The overall optimization objective is thus to minimize the total vulnerability of the multicast session.

To achieve such an objective, two problems are involved subject to different network scenarios. One is to keep the routing algorithm unchanged in the core network and assign two edge routers to each destination such that the total vulnerability of the multicast session is minimized, named as the *edge router assignment problem*. The other is to determine a multicast routing tree such that the sum of the vulnerability between every edge router pair of the multicast tree is minimized assuming that the edge router pair each destination can connect to is predetermined, named as the *best partial multicast protection tree problem*.

We studied the first subproblem in [13]. In this paper, we focus our study on the second problem. Considering the total cost of the multicast tree, we set a constraint of the number of the hops on the path from the source to each destination assuming that the cost of each edge is a constant. Hence, the key to solving the problem is to solve the 2-best paths problem with hop limit.

The best path pairs problem was generally defined as the K-best paths problem which finds K paths as diverse as possible and with the lowest total cost. The problem of finding K-best paths has been studied in [1], [7], [4]. In [4], an optimal solution is given for finding K-best paths without hop limits using minimum cost network flow (MCNF) algorithms. However, we show that the 2-best paths problem with hop limit is NP-complete by showing its special case, the 2-disjoint paths problem with hop limit is NP-complete. Therefore, the best partial multicast tree problem is NP-complete. We derive a lower bound for the problem based on the optimal solution to the problem of minimizing the vulnerability between every edge router pair without hop limit. We then propose the Partition and Sharing (PAS) algorithm to solve the problem. The efficiency of the PAS algorithm is evaluated by simulations and compared with the computed lower bound.

The rest of the paper is organized as follows. Section II presents the problem statement and proves its NP-completeness. Section III presents the PAS algorithm and derives a lower bound. Simulation results are presented and discussed in Section IV. Section V concludes the paper.

II. Problem Description

We model the network as an directed graph $G =< V, E >$, where V stands for the set of nodes, including the source, destinations and routers, and E stands for the set of links between nodes. For simplicity, we assume the cost of each link is unity. A multicast session is denoted as $\mathcal{M} =< s, D >$, where $D = \{d_1, d_2, \ldots, d_n\}$ is the destination set with $n = |D|$. Each destination d_k is connected to two edge routers $R_k = \{r_{k1}, r_{k2}\}$, where $1 \leq k \leq n$.

Let P_{k1} be the path from s to destination d_k through edge router r_{k1}, and P_{k2} be the path from s to destination d_k through edge router r_{k2}. Let $\theta(s, d_k)$ be the set of links shared between path P_{k1} and path P_{k2}, which is defined as:

$$\theta(s, d_k) = \{e | e \in P_{k1} \cap P_{k2}\}$$

The vulnerability from s to d_k, $\beta(s, d_k)$, is defined as the number of links in $\theta(s, d_k)$, i.e.,

$$\beta(s, d_k) = |\theta(s, d_k)|$$

Let $R = \cup_{d_k \in D} R_k$ be the set of edge routers which will participate in the multicast tree. Our objective is to find a multicast routing tree from s to R such that the total vulnerability for the destination set, which is denoted by $\sum_{d_k \in D} \beta(s, d_k)$, is minimized subject to the constraint that any path from s to any destination will be no more than H hops. For those edge routers shared by different destinations, the path from the source to the edge router is also shared by those destinations. This problem can be formulated by an integer programming model. We use the following notations:

$$c_i(l, m) = \begin{cases} 1 & \text{if link } (l, m) \text{ is on path } P_i \text{ from } s \text{ to } r_i, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

$$y_k(l, m) = \begin{cases} 1 & \text{if link } (l, m) \text{ is on path } P_{k1} \text{ and} \\ & \text{path } P_{k2} \text{ for destination } d_k, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Then our problem can be modeled as :

$$\min \sum_{d_k \in D} \sum_{(l,m) \in E} y_k(l, m) \tag{3}$$

subject to:

$$\sum_l c_i(s, l) = 1, \forall r_i \in R \tag{4}$$

$$\sum_l c_i(l, s) = 0, \forall r_i \in R \tag{5}$$

$$\sum_l c_i(r_i, l) = 0, \forall r_i \in R \tag{6}$$

$$\sum_l c_i(l, r_i) = 1, \forall r_i \in R \tag{7}$$

$$\sum_l c_i(l, m) = \sum_j c_i(m, j), \forall r_i \in R, \forall m \neq s, r_i \tag{8}$$

$$\sum_l c_i(l, m) \leq 1, \forall r_i \in R, \forall m \neq s, r_i \tag{9}$$

$$\sum_m c_i(m, l) \leq 1, \forall r_i \in R, \forall m \neq s, r_i \tag{10}$$

$$\sum_{(l,m) \in E} c_i(l, m) \leq H - 1, \forall r_i \in R \tag{11}$$

$$y_k(l, m) \geq c_{k1}(l, m) + c_{k2}(l, m) - 1, \forall (l, m) \in E, d_k \in D \tag{12}$$

$$c_i(l, m) \in \{0, 1\}, \forall (l, m) \in E, r_i \in R \tag{13}$$

$$y_k(l, m) \in \{0, 1\}, \forall (l, m) \in E, d_k \in D \tag{14}$$

The constraints are explained as follows. Equations (4) and (5) ensure that the source node has one-unit outgoing flow on the path to each edge router and has zero incoming flow, respectively. Equations (6) and (7) ensure that each edge router has one-unit incoming flow on the path from source to each edge router and has zero outgoing flow, respectively. Equation (8) guarantees for each intermediate node the incoming flow equals the outgoing flow if it is on the path to each edge router. Equation (9) ensures that the outgoing flow from each intermediate node on the path from the source to each edge router is at most 1. Equation (10) ensures that the incoming flow at any intermediate node on the path from the source to each edge router is at most 1. These two constraints guarantee no loop exists on the path. Equation (11) ensures that the path from the source to each edge router satisfies hop limit $H-1$. Equation (12) gives the formula for calculating the vulnerability for the two paths to each destination. Equations (13) to (14) are self-explainable.

Since the edge router pair that each destination can connect to is predetermined, the two paths to each edge router pair are actually the two paths to the destination. Hence, the problem is equivalent to finding n 2-best paths with hop limit from the source to each destination. In the following, we show the NP-completeness of the 2-best paths problem with hop limit is NP-complete since its special case, the 2-disjoint paths problem with hop limit, is NP-complete.

The 2-disjoint paths problem with hop limit is a special case of the min-max 2-disjoint paths problem (with unit edge cost) which was proved in [6] by a polynomial reduction from the maximum 2-satisfiability problem. In their proof, an undirected graph is constructed with edge costs varying in different positive and integral values. Since the maximum 2-satisfiability problem is strongly NP-complete, we can split an edge in the constructed graph with cost l into l unit-cost edges in series such that the proof is still valid for the min-max 2-disjoint paths problem with unit edge cost. Hence, the 2-disjoint paths problem with hop limit is also NP-complete.

Lemma 1: The decision version of the 2-best paths problem with hop limit is NP-complete.

We have the the following theorem.

Theorem 1: The decision version of the best partial multicast protection tree problem is NP-complete.

Because of the NP-completeness, the problem of finding the best partial multicast protection tree is unlikely to be solved in polynomial time unless $P = NP$. We instead consider efficient heuristic algorithms.

III. Heuristic Algorithm

In this section, we first propose a heuristic algorithm to solve the best partial multicast protection tree problem. We then derive a lower bound for the total vulnerability.

A. The PAS Algorithm

In order to minimize the vulnerability on the two paths to each destination, we should avoid using common links on the two path. On the other hand, to minimize the total cost of the multicast session, we should increase sharing of links among paths for different destinations.

We propose the partition and sharing (PAS) algorithm which consists of four major stages. In Stage I, we construct the graph composed of nodes representing edge routes and edges representing edge router pairs. In Stage II, we partition the edge router R into up to $\mid R \mid$ disjoint subsets such that each subset contains at most one edge router in each edge router pair. This stage can be done using the approximation algorithm proposed for k-coloring problem [2]. In Stage III, we find a multicast tree to edge routers in each subset following in the descending order of the average node degree of all the nodes (in G). Since each path needs to satisfy the hop limit, we employ the minimum-cost path heuristic (MPH) to find a minimum-cost Steiner tree [11]. In the MPH algorithm, the shortest path to the router closest to the source is picked and added to the partially built tree. To increase the sharing among these routers, once a path is found, we reduce the cost to zero for those links on the path. After we find the first tree, we increase the cost for those links on the first tree to a large number greater than the total cost of all the links in the graph. We then find the second multicast tree to edge routers in the second subset using the MPH algorithm. This process continues until all the subset has been processed. In Stage IV, we compute the total vulnerability of the multicast session. The detailed steps of the PAS algorithms are described as follows.

```
Algorithm PAS($G, H$);
begin
    //I: Graph construction.
    1. Construct the graph composed of nodes representing edge routers
       and edges representing edge router pairs.
    //II: Set partition.
    2. Partition nodes in the constructed graph into $m$ disjoint subsets,
       $2 \leq m \leq\mid R \mid$, using the approximation algorithm for the
       $k$-coloring problem.
    3. Sort these subsets in the descending order of the average node degree
       of all the nodes (in $G$) in the each subset as $R_1$, $R_2$, $\cdots$, $R_m$.
    //III: Find multicast trees for subsets $R_1$, $R_2$, $\cdots$, $R_m$.
    4. For each $R_i$, $1 \leq i \leq m$, repeat Steps 5 to 8.
    5. For every edge router in $R_i$, $1 \leq i \leq m$, repeat Steps 6 and 7.
    6. In $G$, find the shortest paths with hop limit $H$ from $s$ to edge routers
       in the subset using Bellman-Ford algorithm. If such a path tree
       does not exist, return "no feasible solution".
    7. Find the shortest path among the remaining paths and update
       cost = 0 for links on the already-found path.
    8. Update cost = $|E|$ for links on the found tree.
    //IV: Compute the total vulnerability.
    9. Compute the total vulnerability of the multicast session.
    10. Return the multicast trees and the total vulnerability.
end
```

Stage I takes $O(\mid D \mid + \mid R \mid)$ time, where $\mid D \mid$ equals to the number of edges the constructed graph. Stage II takes $O(\mid D \mid + \mid R \mid) \log \mid R \mid$ using the approximation algorithm for k-coloring problem [2] plus $O(\mid R \mid\mid V \mid)$. In Stage III, the Bellman-Ford algorithm (with $O(\mid V \mid\mid E \mid)$ time) dominates the running time. There will be $O(\mid R \mid)$ iterations. Stage IV takes $O(\mid V \mid^2)$ time using the algorithm proposed in [13]. Hence, the time complexity of the PAS algorithm is $O(\mid V \mid^2 \mid E \mid)$.

Figure 2 shows an example of the PAS algorithm. A multicast session is composed of the source H1, connected to the edge router C, and three destinations H2, H3, and H4, each connected to the edge router pair $\{A, B\}$, $\{D, E\}$, and $\{A, E\}$ respectively. After Stages I and II, two subsets of edge routers are obtained, i.e., $\{B, E\}$ and $\{A, D\}$. In Stage III, $\{B, E\}$ is selected first because it has larger average node degree in G than $\{A, D\}$. Assuming $H = 4$ (the actual hop limit from C

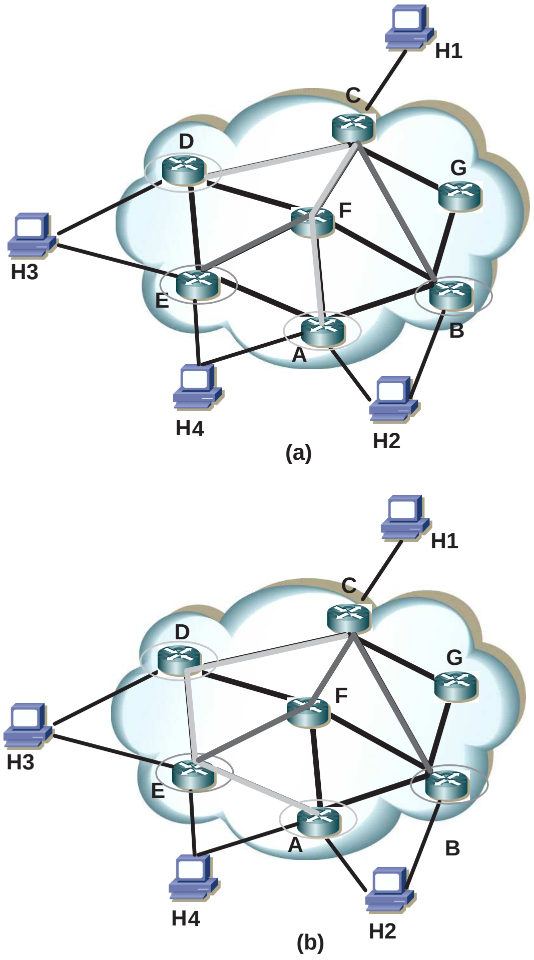

Fig. 2. Example of the PAS algorithm. (a) Multicast trees in the core network for $H = 4$. (b) Multicast trees in the core network for $H = 5$. Paths for $\{B, E\}$ are in dark grey lines and paths for $\{A, D\}$ are in light grey lines.

to the edge router pairs is 2), a multicast tree (in the core network) is found for $\{B, E\}$ consisting of paths C-B and C-F-E, as shown in dark grey lines in Figure 2(a). The edges on the found paths are put back to the graph with their costs updated as $\mid E \mid$. Then the multicast tree for $\{A, D\}$ is found consisting of paths C-F-A and C-D, as shown in light grey lines in Figure 2(a). Thus, the total vulnerability for the multicast session is 1. As one can understand, with the hop limit increasing, the search space for edges is enlarged, which tends to yield a better solution with decreased total vulnerability. For $H = 5$ (the actual hop limit from C to the edge router pairs is 3), the multicast tree found for $\{B, E\}$ consists of paths C-B and C-F-E, while the multicast tree found for $\{A, D\}$ consists of paths C-D-E-A and C-D, as shown in Figure 2(b). The total vulnerability for $H = 3$ is only 0.

B. Lower Bound

We derive a lower bound by finding the the best path pairs without hop limit for each edge router pair and summing up the vulnerability of these path pairs. We construct an auxiliary graph G_1 by adding nodes and edges from G. For each edge router pair $\{r_{i1}, r_{i2}\}$, we add one common node d_i' and links from r_{i1} to d_i' and from r_{i2} to d_i'. We then solve the 2-best paths problem from s to d_i' with no hop limit using the modified K-best path (KBP) algorithm proposed in [4], where $K = 2$.

Before we call the modified KBP algorithm, we need construct another auxiliary graph G_1' by letting link capacity to be unit and adding additional nodes and links as follows. For each link (i, j) in G_1, we add a dummy node and two artificial links from node i to the dummy node and from the dummy node to node j. The link cost and capacity of the two added links are $|E'|/2$ and $\min\{in\_degree, out\_degree, K\} - 1$, respectively. We define a mapping from paths obtained from G_1' to G_1, denoted as $P_{G_1} \leftarrow P_{G_1'}$ by replacing paths through artificial links to link (i, j).

We list the modified KBP algorithm as follows, where MaxFlow(G, s, d) refers to maximum flow algorithm on G between s and d and MCNF(G, K, s, d) refers to minimum cost network flow for input graph G and K unit flow supplied between s to d.

```
Algorithm Modified KBP($G, K, s, D$);
begin
    Get $G_1$ from $G$;
    Get $G_1'$ from $G_1$;
    for each node $d_i \in D$ do
    begin
        Add node $d_i'$ and connect $r_{i1}$ to $d_i'$ and $r_{i2}$ to $d_i'$;
        Get $P_{G_1'}$ by running MCNF($G_1', K, s, d_i'$);
        Get $P_{G_1} \leftarrow P_{G_1'}$;
        Compute the vulnerability for paths on $P_{G_1}$;
        Sum to the total vulnerability.
    end-for
    Return the total vulnerability.
end
```

Following the proof of the KBP algorithm, we can prove that the Modified KBP algorithm finds the optimal solution to best path pairs from s to each edge router pair associated with the destination in D. The computational complexity of the Modified KBP is $O(|V|)$ times the complexity of the MCNF algorithm [4].

IV. Simulation Results

In the following, we present the simulation results of the PAS algorithm and compare them with the results given by the lower bound.

Simulations have been conducted for the PAS algorithm with randomly generated instances. The core network topology G is defined by two parameters N and U, where $N =\mid V \mid$ is the number of nodes and U is the maximum out degree of a node. For each node v_i, we randomly assign its out degree u_i which is uniformly distributed in $\{1, 2, \cdots, U\}$ and randomly generate d_i links originating from node v_i to other nodes. We then randomly assign two edge routers to each of the n destinations assuming that each node in the graph can be an edge router.

Preliminary tests show that the values of N and U do not have much impact of the interest of the simulations. In our simulations, we fix $N = 100$, $U = 8$, and vary two parameters, D and H, which represent the number of destinations and the hop limit for each path from the source to each destination respectively. For each combination of parameters n and H, we generate 1000 instances. For each instance, we solve it using the PAS algorithm and compute the lower bound of the total vulnerability. The performance of the PAS is evaluated by its

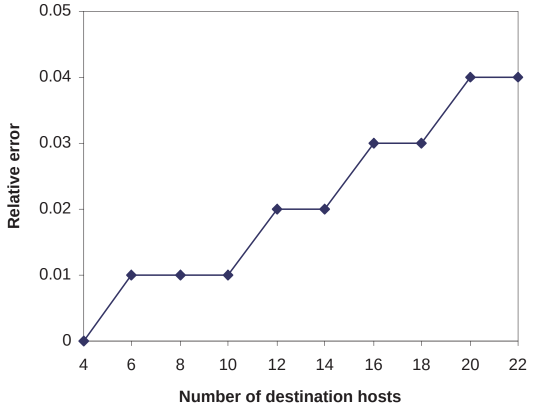

Fig. 3. Relative error for the PAS algorithm with different number of destinations.

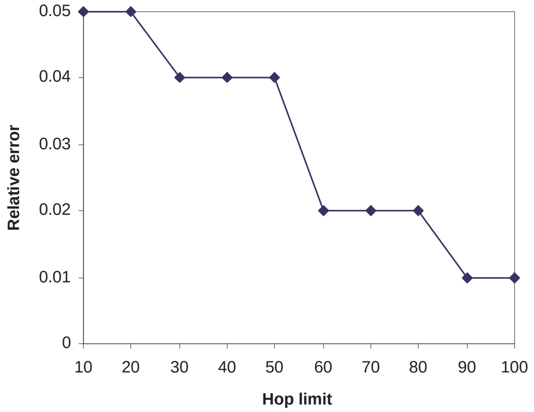

Fig. 4. Relative error for the PAS algorithm with different hop limit.

relative error compared with the lower bound. Let β_p denote the total vulnerability obtained by PAS and β_l denote the total vulnerability given by the lower bound. The relative error of the PAS algorithm is defined as:

$$\epsilon = \frac{\beta_p - \beta_l}{\beta_l}.$$

We first evaluate the performance of the PAS algorithm by fixing H at 50 and varying the number of destinations n in $\{4, 6, \cdots, 22\}$. Figure 3 shows the relative error vs. the number of destinations. As shown in the figure, the relative error of the heuristic algorithm increases with the number of destination increasing, which is consistent with our expectation.

We then evaluate the performance of the PAS algorithm by fixing n at 20 and varying H in $\{10, 20, \cdots 100\}$. Figure 4 shows the relative error of the PAS algorithm vs. the hop limit. As shown in the figure, the relative error of the PAS algorithm decreases with the hop limit increasing. When the hop limit is increased for each path, the PAS algorithm tends to find a better solution, hence the relative error is smaller.

The effectiveness of the PAS algorithm is evidenced by the relative errors, which are less than 5% for all the instances we tested. The performance can be further improved by finding a better lower bound.

V. Conclusion

Our contributions in this paper are in three folds. First, we introduced the concept of partial protection, which fills the gap between traditional 100% protection and non-protection subject to single link failure. Second, we showed the NP-completeness of the problem of finding the best partial multicast protection tree. Third, we proposed an efficient solution, the PAS algorithm, to solve the problem. Partial protection points out a more practical direction in network protection. The proposed PAS algorithm can also be applied to other multicasting problems.

VI. Acknowledgement

This work is in part supported by the University of Nevada, Las Vegas under the start-up fund provided to Dr. Mei Yang, and the National Science Foundation (NSF) under the grant NeTS-NR 0435095 provided to Dr. Jianping Wang.

References

[1] D. Castanon, "Efficient algorithms for finding the K best paths through a trellis," *IEEE Trans. AES*, vol. 26, no. 2, pp. 405-410. 1990.

[2] J.D. Cho, S. Raje, and M. Sarrafzadeh, "Fast approximation algorithms on maxcut, k-coloring, and k-color ordering for VLSI applications," *IEEE Trans. Computers*, vol. 47, no. 11, pp. 1253-1267. 1998.

[3] A. Fei, J. Cui, M. Gerla, and D. Cavendish, "A "dual-tree" scheme for fault-tolerant multicast", in *Proc. ICC 2001*, vol. 3, pp. 690-694.

[4] S. Li and C. Wu, "A k-best paths algorithm for highly reliable communication networks," *IEICE Trans. Commun.*, vol. E82-B, no. 4, Apr. 1999.

[5] C. Lee and S. Koh, "A design of the minimum cost ring-chain network with dual-homing survivability: a tabu search approach," *Computers Operations Research*, vol. 24, no. 9, pp. 883-897, 1997.

[6] C. Li, S. McCormick, D. Simchi-Levi, "The complexity of finding two disjoint paths with min-max objective function," *Discrete Applied Mathematics*, vol. 26, no. 1, pp. 105-115, 1990.

[7] S. Nikolopoulos, A. Pitsillides, and D. Tipper, "Addressing network survivability issues by finding the K-best paths through a trellis graph," in *Proc. IEEE INFOCOM*, pp. 370-377, Apr. 1997.

[8] A. Proestaki and M. Sinclair, "Design and dimensioning of dual-homing hierarchical multi-ring networks," *IEE Proceedings-Communications*, vol. 147, no. 2, pp. 96-104, Apr. 2000.

[9] S. Ramamurthy and B. Mukherjee, "Survivable WDM mesh networks, part I - protection," in *Proc. IEEE INFOCOM*, vol. 2, Marc. 2003, pp. 744-751.

[10] N. Singhal, L. Sahasrabuddhe, B. Mukherjee, "Dynamic provisioning of survivable multicast sessions in optical WDM mesh networks," *Proc. OFC*, vol. 1, pp. 207-209, Mar. 2003.

[11] N. Singhal, L. Sahasrabuddhe, B. Mukherjee, "Provisioning of survivable multicast sessions against single link failures in optical WDM mesh networks," *J. Lightwave Technol.*, vol. 21,no. 11, pp. 2587-2594, Nov. 2003.

[12] D. Xu, Y. Xiong, and C. Qiao, "Novel algorithms for shared segment protection," *IEEE J. Select. Areas in Commun.*, vol. 21, no. 8, pp. 1320-1331, Oct. 2003.

[13] J. Wang, M. Yang, X. Qi, and R. Cook, "Dual-home based multicast protection," in *Proc. IEEE GLOBECOM*, 2004, pp. 1123-1127.