Dual-Homing Based Scalable Partial Multicast Protection

Jianping Wang, *Member*, *IEEE*, Mei Yang, *Member*, *IEEE*,
Bin Yang, and S.Q. Zheng, *Senior Member*, *IEEE*

Abstract—In this paper, we propose a scalable multicast protection scheme based on a dual-homing architecture where each destination host is connected to two edge routers. Under such an architecture, there are two paths from the source of a multicast session to each destination host, which provides a certain level of protection for the data traffic from the source to the destination host. The protection level varies from 0 percent to 100 percent against a single link failure, depending on the number of shared links between these two paths. The major advantage of the proposed scheme lies in its scalability due to the fact that the protection is provided by constructing a dual-homing architecture at the access network while keeping the routing protocols in the core network unchanged. The selection of dual edge routers plays an important role in enhancing the protection level. Two problems arise for the proposed dual-homing partial multicast protection scheme. One is to calculate the survivability from the source to any pair of edge routers. The other is to assign a pair of edge routers for each destination host such that the total survivability is maximized for the multicast session subject to the port number constraint of each edge router. We propose an optimal algorithm to solve the first problem. We prove the decision version of the second problem is NP-complete and propose two heuristic algorithms to solve it. Simulation results show that the proposed heuristic algorithms achieve performance close to the calculated lower bound.

Index Terms—Networks, multicast, survivability, protection, complexity, NP-completeness.

1 Introduction

One important issue in networks is survivability. Survivability is the capability of a network to function in the event of node or link failures [6]. There are two approaches to provide survivability. One is called on-demand recovery, which reroutes the packets around a failed link/node when the failure is detected in the network [4]. The major disadvantage of this scheme is that the long recovery latency can be undesirable for many real-time communications. An alternative solution is protection, which provides two disjoint paths, one designated as the primary path, the other reserved as the backup path [4]. The backup path is activated when a link or node failure is detected on the primary path. This approach can provide guaranteed survivability for the network subject to a single link/node failure.

Multicast is a one-to-many communication scheme that requires simultaneous transmission from a source to a set of destination hosts. Protection is more challenging for multicast communications since a link/node failure will affect a number of multicast destination hosts. Meanwhile, the large number of destination hosts in a multicast session certainly makes it more difficult to provide efficient protection for multicast communications.

- *J. Wang is with the Department of Computer Science, University of Mississippi, University, MS 38677. E-mail: jpwang@cs.olemiss.edu.*
- *M. Yang is with the Department of Electrical and Computer Engineering, University of Nevada at Las Vegas, Las Vegas, NV 89154. E-mail: meiyang@egr.unlv.edu.*
- *B. Yang and S.Q. Zheng are with the Department of Computer Science, University of Texas at Dallas, Richardson, TX 75083. E-mail: binyang@cisco.com, sizheng@utdallas.edu.*

Manuscript received 7 May 2005; revised 12 Sept. 2005; accepted 15 Feb. 2006; published online 20 July 2006.
For information on obtaining reprints of this article, please send e-mail to: tc@computer.org, and reference IEEECS Log Number TC-0151-0505.

In the literature, several multicast protection schemes have been proposed to provide 100 percent protection against a single link failure [4], [12], [16], [17], all requiring 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 serving as the primary multicast tree and the other as the backup multicast tree [12]. However, it is hard and even impossible to find two link-disjoint multicast trees for a large-scale multicast session. Alternative ways include segment protection [17], [18] and path protection [17]. For all these solutions, to provide protection for a multicast session, certain changes to multicast routing algorithms and protocols are necessary and the core network must be aware of these changes. Hence, they are not scalable for large-scale multicast sessions.

Modern networks can no longer limit the options of protection only to the extreme cases: with 0 percent protection or with 100 percent protection. As a matter of fact, *partial protection* should be provided by setting up two paths from the source to a destination with certain shared links. If there is one link failure among the disjoint links along the two paths, protection can be provided; while, if there is one link failure among the shared links along the two paths, protection cannot be provided. Therefore, two such paths can provide partial protection against a single link failure.

In this paper, we propose a scalable scheme for partial multicast protection based on the dual-homing architecture. The proposed scheme does not require any change to the routing algorithm in the core network; therefore, it is scalable for large-scale multicast sessions. Dual-homing was

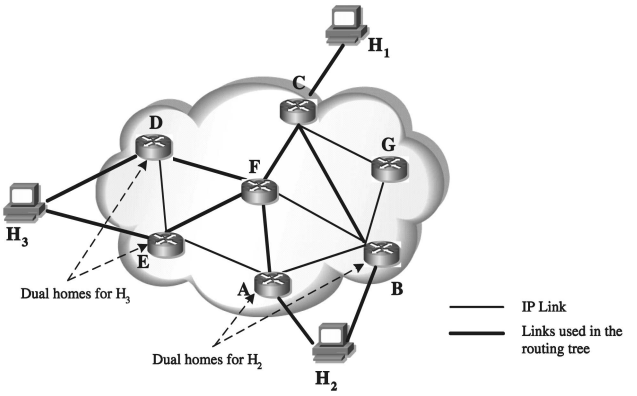

Fig. 1. An example of the dual-homing architecture.

originally proposed to enhance survivability for access networks [8]. In a dual-homing architecture, a host in an access network can be connected to two IP edge routers. The major objective of dual-homing is to provide protection against access link or access node (edge router) failure(s) caused by a system malfunction or a scheduled outage [8].

The dual-homing architecture has been widely studied in self-healing ring networks [8], [9], [11], [15]. In recent years, some research has been conducted to provide protection for a dual-homing IP-based access network over Wavelength Division Multiplexing (WDM) core networks [10], [14]. A measurement based analysis for the performance of multi-homing solutions is given in [2]. Dual-homing applications in wireless networks are also reported in [3].

We apply the concept of dual-homing to multicast protection based on the following observation: If a destination host is attached to two edge routers, there exist two paths from the source to the destination (host). If these two paths are disjoint, they can provide 100 percent protection for the destination host against a single link failure. If they are not disjoint, partial protection can be provided. Fig. 1 shows two examples. We assume that H_1 is the source of the multicast session, H_2 and H_3 are two destinations. H_2 is connected to edge routers A and B so that there are two paths from H_1 to H_2, H_1-C-F-A-H_2 and H_1-C-B-H_2. The two segments of these two paths in the core network, i.e., C-F-A and C-B, are disjoint. When any single link along the two segments fails, H_2 can still receive data through the alternative path. H_3 is attached to edge routers D and E, where the two paths from H_1 to H_3 are H_1-C-F-D-H_3 and H_1-C-F-E-H_3. The two segments of these two paths in the core network, i.e., C-F-D and C-F-E, share a common link C-F. If any link fails along the two segments except link C-F, H_3 can receive data through the alternative path. Thus, H_3 is partially protected.

It is observed that the two edge routers to which a destination host 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 introduce the concept of *vulnerability*, which is defined as the number of shared links between the two paths from the source to the two edge routers that the destination host is connected to. Obviously, each destination host should try to connect to two edge routers with the minimum vulnerability. However, since the number of ports of each router is limited in reality, it may not be feasible for each destination host to be connected to such two edge routers.

Let the total vulnerability of a multicast session be the summation of the vulnerability for all destination hosts when each of them is feasibly connected to two edge routers. The objective studied in this paper is to minimize the total vulnerability of a multicast session by determining which two edge routers each destination host should be connected to subject to the port number constraint of each edge router. This problem is called *the edge router assignment problem*. Another associated problem is how to calculate the vulnerability between any two edge routers.

The rest of the paper is organized as follows: The problem description is formally given in Section 2. The algorithm for computing the vulnerability between any two routers is given in Section 3. In Section 4, we prove the NP-completeness of the decision version of the edge router assignment problem and discuss the feasibility version of the problem. A special case of the problem with fixed primary edge routers for all destinations is also discussed in Section 4. In Section 5, we propose two heuristic algorithms to solve the general edge router assignment problem. Simulation results for both heuristic algorithms are presented and discussed in Section 6. Section 7 summarizes the paper.

2 Problem Description

A network can be modeled as an undirected graph consisting of nodes representing routers in the network and links representing connections between routers. In this paper, we will use the terminologies of routers and nodes interchangeably. We are given a multicast session denoted by $\mathcal{M} =< s, D >$, where s is the source host and $D = \{d_1, d_2, \ldots, d_{|D|}\}$ is the set of destination hosts. Let r_s be the edge router that s is connected to. Let R_i be the set of feasible edge routers that destination host d_i can be connected to. Let R be the set of all feasible edge routers that the destinations can be connected to for the multicast session, i.e., $R = \cup_{i=1}^{|D|} R_i$. Given the network topology, a multicast tree can be constructed from the root node r_s to leaf nodes of edge routers in R using any multicast routing algorithm or protocol, such as the Distance Vector Multicast Routing Protocol (DVMRP) [13] or the Protocol Independent Multicast (PIM) [13].

A multicast tree is modeled as an undirected graph $T =< V, E >$, where V stands for a set of routers $\{r_1, r_2, \ldots, r_{|V|}\}$ and E stands for the set of links between routers. Note that $R \subset V$. Let P_j be the unique path from r_s to router $r_j \in V$ in the multicast tree. Let $\theta(r_s, \{r_j, r_k\})$ be the set of links shared by two paths P_j and P_k, i.e.,

$$\theta(r_s, \{r_j, r_k\}) = \{e | e \in P_j \cap P_k\}. \quad (1)$$

The vulnerability between two nodes r_j and r_k is defined as the number of links in $\theta(r_s, \{r_j, r_k\})$, i.e.,

$$\beta(r_s, \{r_j, r_k\}) = |\theta(r_s, \{r_j, r_k\})|. \quad (2)$$

Let N_j be the port number constraint of edge router r_j, i.e., the number of destination hosts that r_j can serve. The edge router assignment problem studied in this paper is to determine two edge routers for each $d_i \in D$ such that the total vulnerability of $\mathcal{M}$ is minimized and no more than N_j destination hosts are connected to edge router r_j for any $r_j \in R$.

We formally define the problem by an integer programming model with the following binary variables:

$$x_{ij} = \begin{cases} 1 & \text{if } d_i \text{ is connected to edge router } r_j, \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

$$y_{ijk} = \begin{cases} 1 & \text{if } d_i \text{ is connected to both } r_j \text{ and } r_k, \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

The objective of the problem is to minimize

$$\Delta = \sum_{d_i \in D} \sum_{r_j, r_k \in R_i} y_{ijk} \beta(r_s, \{r_j, r_k\}) \quad (5)$$

subject to

$$\sum_{r_j \in R_i} x_{ij} = 2, \forall i = 1, 2, \ldots, |D|; \quad (6)$$

$$\sum_{r_j \in R_i} x_{ij} \leq N_j, \forall j = 1, 2, \ldots, |R|; \quad (7)$$

$$y_{ijk} \geq x_{ij} + x_{ik} - 1, \forall i = 1, 2, \ldots, |D|, \forall r_j, r_k \in R_i; \quad (8)$$

$$x_{ij} \in \{0, 1\}, \forall i = 1, 2, \ldots, |D|, \forall r_j \in R_i; \quad (9)$$

$$y_{ijk} \in \{0, 1\}, \forall i = 1, 2, \ldots, |D|, \forall r_j, r_k \in R_i. \quad (10)$$

In the above problem formulation, we call Δ the total vulnerability of the edge router assignment. Then, $\frac{\Delta}{|D|}$ gives the average number of links shared by the two paths for each destination host. Clearly, minimizing the total vulnerability is equivalent to minimizing $\frac{\Delta}{|D|}$, which can be interpreted as the average risk faced by the destination hosts.

From a slightly different angle, we can see that this formulation has another interpretation regarding how an individual link failure can affect the multicast session. For a feasible router assignment $\mathcal{A}$ (that satisfies constraints (6) to (10)) and a link e in the multicast tree, we define the loss of link failure (LLF), denoted by $L_A(e)$, as the number of destination hosts that will be disconnected if e fails, i.e.,

$$L_A(e) = |\{d_i | e \in P_j \cap P_k, y_{ijk} = 1 \text{ under } \mathcal{A}\}|.$$

Clearly, larger $L_A(e)$ implies more severe disruption to the multicast session if e fails. Let $\gamma(e, \{r_j, r_k\})$ be a binary variable indicating whether link e is on $P_j \cap P_k$, i.e.,

$$\gamma(e, \{r_j, r_k\}) = \begin{cases} 1 & \text{if } e \in P_j \cap P_k, \\ 0 & \text{otherwise.} \end{cases}$$

Then, the total LLF of $\mathcal{A}$ can be derived as

$$\begin{aligned} \sum_{e \in E} L_A(e) &= \sum_{e \in E} \sum_{d_i \in D} \sum_{r_j, r_k \in R_i} y_{ijk} \gamma(e, \{r_j, r_k\}) \\ &= \sum_{d_i \in D} \sum_{r_j, r_k \in R_i} y_{ijk} \beta(r_s, \{r_j, r_k\}), \end{aligned} \quad (11)$$

which is exactly $\Delta_{\mathcal{A}}$, the total vulnerability under the edge router assignment $\mathcal{A}$. Assuming all links have the same failure probability, then the average LLF, i.e., the average number of destination hosts affected due to a single link failure, is $\frac{\Delta_{\mathcal{A}}}{|E|}$.

Therefore, we see that our edge router assignment with the minimum total vulnerability has two equivalent interpretations from different perspectives: trying to minimize the average risk faced by the destination hosts and trying to minimize the average disruption caused by the network links. Under the assumption that all links have the same failure probability, minimizing the total vulnerability provides a reasonably good protection for the multicast session in two different aspects as a unified objective.

Considering the randomness of link failures in real applications, the minimum total vulnerability is a reasonable measure for characterizing the reliability of dual-homing architecture for multicast. By the nature that the paths used by all destination hosts form a partial tree of the multicast tree, minimizing the total vulnerability forces the paths to be spread among tree branches. Consequently, as the objective function, the minimum total vulnerability exhibits an effective heuristic for avoiding the worst-case scenario from a purely combinatorial perspective.

3 Algorithm for Computing Vulnerability

Before we solve the above problem, we first introduce an efficient algorithm to compute the vulnerability of any pair of two routers in the multicast tree.

Without loss of generality, we assume that the routers in the multicast tree T are indexed by a breath-first search from the source node. Consequently, each node always has a smaller index than its child nodes in the tree. We use $level(r_j)$ to denote the level of node r_j, i.e., the number of links from r_s to r_j. Then, we have $level(r_s) = 0$. Since $\beta(r_s, \{r_j, r_k\})$ is equal to $\beta(r_s, \{r_k, r_j\})$ by definition, we only calculate $\beta(r_s, \{r_j, r_k\})$ for $j < k$.

Let $b(r_j)$ be the parent node of router r_j in the multicast tree. Two nodes r_j and r_k are called siblings if they have the same parent node. For any pair of routers r_j and r_k with $j < k$, $\beta(r_s, \{r_j, r_k\})$ can be computed recursively according to the following three mutually exclusive cases:

Case 1: $\beta(r_s, \{r_j, r_k\}) = level(b(r_j))$ if r_j and r_k are siblings.

Case 2: $\beta(r_s, \{r_j, r_k\}) = level(r_j)$ if r_j is the parent node of r_k.

Case 3: $\beta(r_s, \{r_j, r_k\}) = \beta(r_s, \{r_j, b(r_k)\})$ if r_j and r_k are neither siblings nor parent-child nodes.

We justify these three cases as follows: For Case 1: If r_j and r_k are siblings, they share all links on the path from r_s to their parent node. Therefore, by definition, $\beta(r_s, \{r_j, r_k\})$ is equal to $level(b(r_j))$. For Case 2: If r_j is the parent node of r_k, by definition, $\beta(r_s, \{r_j, r_k\})$ is the number of links from r_s

to r_j. In this case, protection is only provided under the failure of the incoming link of r_k, not under any link failure on the path from r_s to r_j. For Case 3: If r_j and r_k are neither siblings nor parent-child nodes, they do not share any incoming link, thus $\beta(r_s, \{r_j, r_k\})$ is equal to the number of shared links between r_j and r_k's parent node. Summarizing the above three cases, a top-down algorithm for computing the vulnerability of all pairs of routers is given below.

Algorithm Vulnerability Calculation (T, r_s):
begin
  //Calculating vulnerability for each edge router pair
  for $j = 1$ to $|V|$ do
    for $k = j + 1$ to $|V|$ do
      if $b(r_j) = b(r_k)$ then //case 1
        $\beta(r_s, \{r_j, r_k\}) = level(b(r_j))$
      else if $b(r_k) = r_j$ //case 2, $j < k$, so $b(r_j) \neq r_k$
        $\beta(r_s, \{r_j, r_k\}) = level(r_j)$
      else //case 3
        $\beta(r_s, \{r_j, r_k\}) = \beta(r_s, \{r_j, b(r_k)\})$
end

It only takes a constant time to compute $\beta(r_s, \{r_j, r_k\})$ for each r_j and r_k; therefore, the time complexity of computing $\beta(r_s, \{r_j, r_k\})$, for $k = j + 1, \ldots, |V|$, is $O(|V|)$. There are $|V|$ nodes in the multicast tree, therefore, the total complexity of the vulnerability calculation algorithm for all router pairs is $O(|V|^2)$. Note that this is the lowest possible computational complexity we can achieve for calculating the vulnerability for all pairs of routers because the number of such pairs is in $O(|V|^2)$.

4 Edge Router Assignment Problem

In this section, we study the edge router assignment problem, which is to assign a pair of two edge routers for each destination host $d_i \in D$ with the minimum total vulnerability of the multicast session subject to the port number constraint of each edge router.

The problem can be easily solved if edge routers have no port number constraints (e.g., $N_j > |D|\ \forall r_j \in R$). Given the vulnerability of each pair of edge routers, a destination host can simply choose a feasible pair of edge routers with the minimum vulnerability. In this way, the total vulnerability is minimized. However, such a greedy algorithm may not always find a feasible solution if edge routers have port number constraints.

In the remainder of this section, we first prove the NP-completeness of the decision version of the edge router assignment problem. Then, we discuss the solutions to the feasibility version of the problem and a special case of the problem with fixed primary edge routers.

4.1 NP-Completeness

In the following, we will use the 3SAT problem, a known NP-complete problem, to prove the NP-completeness of our problem.

An instance of 3SAT is an expression $C_1 \wedge C_2 \wedge \cdots \wedge C_m$ of m clauses with n Boolean variables where $C_i = (\hat{x}_{i1} \vee \hat{x}_{i2} \vee \hat{x}_{i3})$. C_i is called a clause, which contains three literals. A literal $\hat{x}_i$ is either x_i or $\overline{x_i}$ of variable x_i. A truth assignment is an assignment $\tau : x_i \rightarrow \{\text{true}, \text{false}\}$. We say that C_i is satisfied under τ if C_i contains a literal x_{ij} with $\tau(x_{ij}) =$ true or literal $\overline{x}_{ij}$ with $\tau(x_{ij}) =$ false. The 3SAT problem is to ask if there exists a truth assignment τ that satisfies all m clauses of C.

The decision version of the edge router assignment problem is stated as follows: Given a destination set D and an edge router set R for a multicast session, where each edge router has a port number constraint and every pair of edge routers has a vulnerability value, is there a feasible assignment with the total vulnerability of the multicast session being 0? Recall that an assignment is called feasible if each destination is assigned to two edge routers and the number of destinations connected to each edge router is less than or equal to its port number constraint.

Now, we show that the 3SAT problem is polynomial-time reducible to the decision version of an edge router assignment problem. Thus, the NP-completeness of the latter problem is proven. For a 3SAT instance, $C = C_1 \wedge C_2 \wedge \cdots \wedge C_m$, we construct a dual-homing architecture as follows: For each x_i in C, $1 \leq i \leq n$, let p_i be the number of its occurrences in all clauses. We use $u_i^j, \overline{u}_i^j$ to represent the jth appearance of x_i and $\overline{x}_i$.

Define the set of destination hosts as

$$D = \{u_i^j, \overline{u}_i^j \mid 1 \leq i \leq n, 1 \leq j \leq p_i\}.$$

Define the set of edge routers as

$$\begin{aligned} R = \{a_i^j, b_i^j \mid 1 \leq i \leq n, 1 \leq j \leq p_i\} \cup \\ \{s_1^k, s_2^k \mid 1 \leq k \leq m\} \cup \{g_1, g_2\}. \end{aligned} \tag{12}$$

Let the port number constraints for g_1 and g_2 be $2m$ and the port number constraints for other edge routers be 1.

Clearly, $|D| = 3m \cdot 2 = 6m$,

$$|R| = 3m \cdot 2 + 2 \cdot m + 2 = 8m + 2,$$

and the total port number constraints for all edge routers is $8m \cdot 1 + 2 \cdot 2m = 12m$. As each destination host needs two edge routers, every feasible assignment will use up all available ports of edge routers.

The feasible edge router sets for destinations are defined as:

$$\begin{cases} u_i^j\ (1 \leq i \leq n, 1 \leq j < p_i): \\ \quad \{b_i^j, a_i^{j+1}\};\ \cup \{s_1^k, s_2^k \mid u_i^j \in c_k, 1 \leq k \leq m\} \cup \{g_1, g_2\} \\ u_i^{p_i}\ (1 \leq i \leq n): \\ \quad \{b_i^{p_i}, a_1\} \cup \{s_1^k, s_2^k \mid u_i^{p_i} \in c_k, 1 \leq k \leq m\} \cup \{g_1, g_2\} \\ \overline{u}_i^j\ (1 \leq i \leq n, 1 \leq j \leq p_i): \\ \quad \{a_i^j, b_i^j\} \cup \{s_1^k, s_2^k \mid \overline{u}_i^j \in c_k, 1 \leq k \leq m\} \cup \{g_1, g_2\}. \end{cases} \tag{13}$$

Fig. 2 shows an example of such a construction. There are four "star-shape polygons," each corresponding to a literal. The three (s_1^i, s_2^i) pairs represent the three clauses. All the possible assignments are represented by the edges: Each edge is between a destination and a node, which means the destination can be assigned to the node.

Let the vulnerability of all pairs of edge routers be 1, except for the following edge router pairs, which are defined as 0:

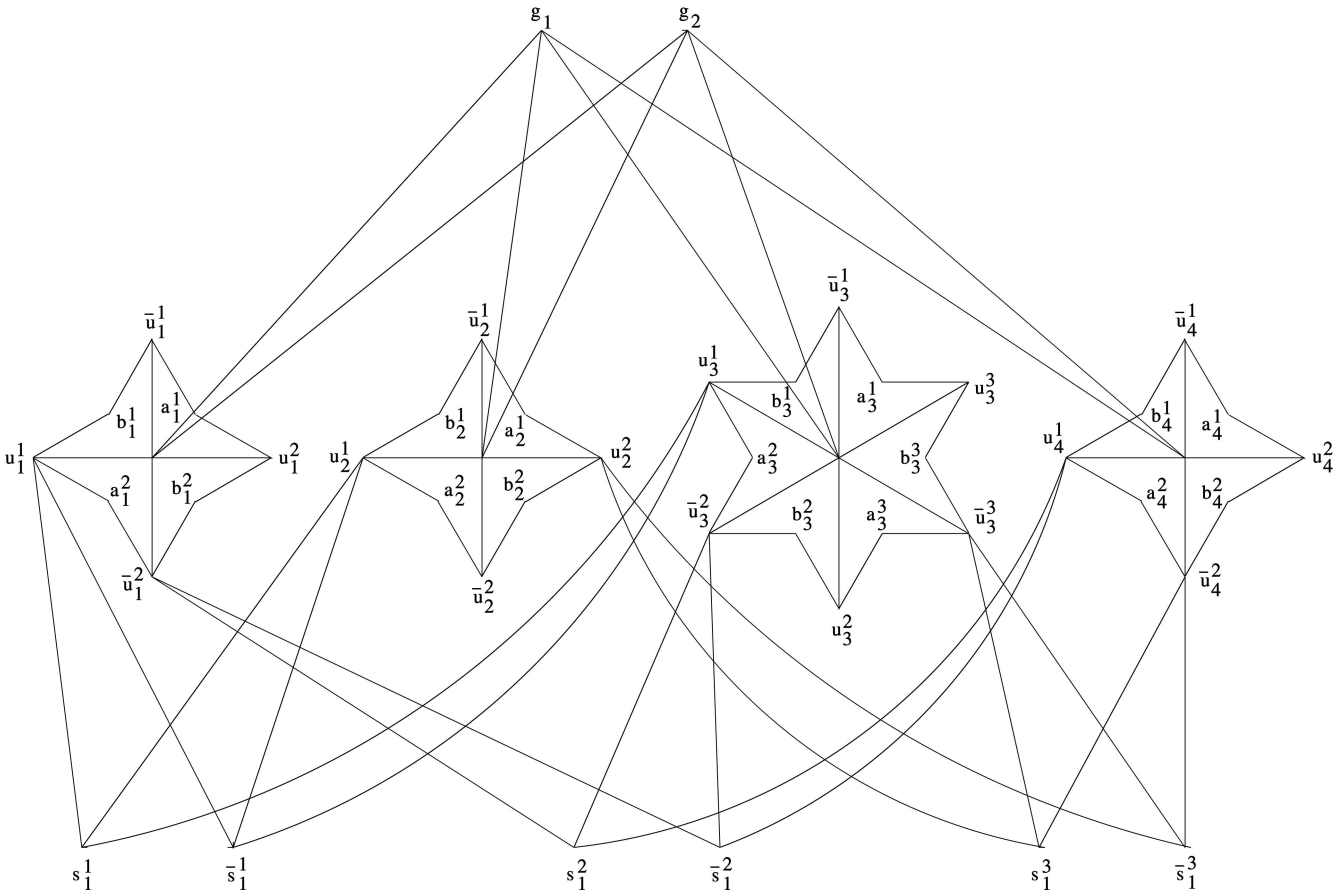

Fig. 2. The dual-homing architecture for $C = (x_1 \vee x_2 \vee x_3) \wedge (\overline{x}_1 \vee \overline{x}_3 \vee x_4) \wedge (x_2 \vee \overline{x}_3 \vee \overline{x}_4)$. For clarity, we use the centralized lines to represent that all of $u_i^j, \overline{u}_i^j$ nodes are connected to g_1, g_2.

1. $W_i = \{(a_i^j, b_i^j) \mid 1 \le j \le p_i\}, i = 1, \cdots, n$.
2. $X_i = \{(b_i^l, a_i^{j+1}) \cup (b_i^{p_i}, a_1) | 1 \le j < p_i\}, i = 1, \cdots, n$.
3. $Y = \{(s_1^k, s_2^k) \mid 1 \le k \le m\}$.
4. $Z = \{(g_1, g_2)\}$.

Let $S = (\cup_{i=1}^n W_i) \cup (\cup_{i=1}^n X_i) \cup Y \cup Z$. Then, S includes all the edge router pairs which have vulnerability equal to 0.

Lemma 1. *A 3SAT instance $C_1 \wedge C_2 \wedge \cdots \wedge C_m$ is feasible if and only if the constructed edge router assignment problem has a feasible assignment with the total vulnerability equal to 0.*

Proof. The proof consists of two parts. Part I: If there exists a truth assignment τ that satisfies all m clauses, then we obtain an assignment with the total vulnerability equal to 0 for the corresponding edge router assignment problem as follows:

1. For each $i = 1, \cdots, n$, if $\tau(x_i) =$ true, assign edge routers a_i^j and b_i^j to $\overline{u}_i^j$, for every $1 \le j \le p_i$; if $\tau(x_i) =$ false, assign edge routers b_i^j and a_i^{j+1} to u_i^j, for every $1 \le j \le p_i - 1$ and assign edge routers $b_i^{p_i}$ and a_i^1 to $u_i^{p_i}$.
2. For each $i = 1, \cdots, m$, find a literal $\hat{x}_k$ in C_i such that $\hat{x}_k = x_k$ and $\tau(x_k) =$ true or $\hat{x}_k = \overline{x}_k$ and $\tau(x_k) =$ false. Since C_i is satisfied, there must exist one such $\hat{x}_k$ in C_i. Let the literal be the lth occurrences of $\hat{x}_k$ in clauses. There are two cases:
   a. $\hat{x}_k = x_k$: Then, $\tau(x_k) =$ true. From 1, all $\overline{u}_k^j$, $1 \le j \le p_k$, are assigned and none of u_k^j, $1 \le j \le p_k$, is assigned. Assign edge routers s_1^i and s_2^i to u_k^l, and assign edge routers g_1 and g_2 to u_k^j for $j \ne l$.
   b. $\hat{x}_k = \overline{x}_k$: Then, $\tau(x_k) =$ false. From 1, all u_k^j, $1 \le j \le p_k$, are assigned and none of $\overline{u}_k^j$, $1 \le j \le p_k$, is assigned. Assign edge routers s_1^i and s_2^i to $\overline{u}_k^l$ and assign edge routers g_1 and g_2 to $\overline{u}_k^j$ for $j \ne l$.

It is easy to verify that the above assignment is feasible and the total vulnerability is 0.

Part II: We show that if the corresponding edge router assignment problem has an assignment with the total vulnerability equal to 0, then there exists a truth assignment τ that satisfies all m clauses. Since the assignment is feasible, all ports are used. Meanwhile, as the total vulnerability is 0, each destination is assigned a pair of edge routers from S. Then, we have:

- For each $i = 1, \cdots, n$, all edge routers $\{a_i^j, b_i^j \mid 1 \le j \le p_i\}$ are assigned. The pairs in S involving a_i^j and b_i^j are the pairs in W_i, X_i. There are two cases:
  1. There exists a pair of edge routers from W_i assigned to a destination. Assuming the pair is (a_i^k, b_i^k) for some k, then it is assigned to destination $\overline{u}_i^k$. Considering edge router a_i^{k+1},

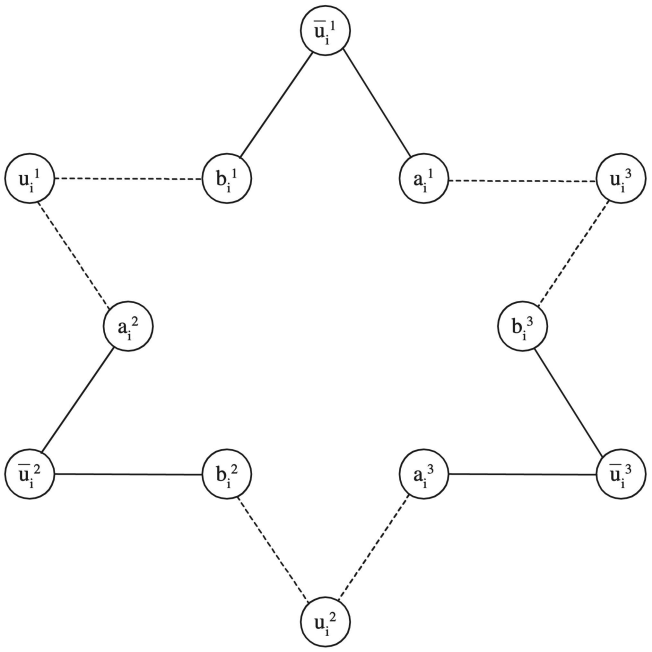

Fig. 3. Assignments of edge routers to destinations. Two sets of assignments are indicated by solid lines and dotted lines, respectively.

since it cannot be assigned together with b_i^k (which is already assigned), it has to be assigned together with b_i^{k+1}. Similarly, a_i^{k-1} has to be assigned together with b_i^{k-1}. Therefore, all edge routers a_i^j, b_i^j, $1 \le j \le p_i$, are assigned as (a_i^j, b_i^j) pairs. That is, all pairs in W_i are assigned to destinations $\overline{u}_i^j$, $1 \le j \le p_i$. And, clearly, no pair from X_i is assigned to any destination.

2. No pair of edge routers from W_i is assigned to any destination. Then, all edge routers in $\{a_i^j, b_i^j \mid 1 \le i \le n,\ 1 \le j \le p_i\}$ are assigned as pairs in X_i. That is, every (b_i^j, a_i^{j+1}) is assigned to destination u_i^j for $1 \le j \le p_i - 1$ and $(b_i^{p_i}, a_1)$ is assigned to $u_i^{p_i}$.

Therefore, edge routers $\{a_i^j, b_i^j\}$ are either all assigned to u_i^js or all assigned to $\overline{u}_i^j$s. Fig. 3 shows the two possible assignments.

- For each $k = 1, \cdots, m$, all edge routers s_1^k and s_2^k are assigned. In S, (s_1^k, s_2^k) is the only pair that involves s_1^k and s_2^k. Let (s_1^k, s_2^k) be assigned to d_l. Then, from the assignment rule, shown in (13), we have $d_l \in C_k$.

We set function $\tau(x_i)$ as follows: If a pair of edge routers from W_i is assigned to a destination, set $\tau(x_i) =$ true; if no pair of edge routers from W_i is assigned to a destination, set $\tau(x_i) =$ false.

We will show such an assignment satisfies C. Let us consider clause C_k. According to the above discussion, there is a $d_l \in C_k$ which is assigned to edge routers s_1^k and s_2^k. Then d_l is one of $\{u_i^j, \overline{u}_i^j\}$ for some i and j. If $d_l = u_i^j$, then, by Case 1, all $\{\overline{u}_i^j \mid 1 \le j \le p_i\}$ are assigned with edge router pairs from W_i. Hence by the function definition, $\tau(x_i) =$ true. Because $u_i^j \in C_k$ means that $x_i \in C_k$, C_k is true. Otherwise, $d_l = \overline{u}_i^j$, then, by Case 2, all $\{u_i^j \mid 1 \le j \le p_i\}$ are assigned with edge router pairs from X_i and no pair from W_i is assigned to a destination. Hence, by the function definition, $\tau(x_i) =$ false. Because $\overline{u}_i^j \in C_k$ means that $\overline{x}_i \in C_k$, C_k is true. As the assignment makes every clause true, the assignment τ is a satisfied assignment for C. □

Theorem 1. *The decision version of the edge router assignment problem is NP-complete.*

Proof. Clearly, the transformation from an instance of the 3SAT problem to the corresponding instance of the decision version of the edge router assignment problem can be done in polynomial time. By Lemma 1, the decision version of the edge router assignment problem is NP-complete. □

Because of the NP-completeness, we have to rely on heuristic algorithms to solve the problem. Ideally, we expect an approximation algorithm with a known error bound. Let β^* be the total vulnerability given by the optimal solution and β^A be the total vulnerability given by an approximation algorithm. The error bound of the approximation algorithm is defined as β^A/β^*. For the edge router assignment, we have the following theorem:

Theorem 2. *For the edge router assignment problem, unless $P = NP$, there is no polynomial-time approximation algorithm with fixed error bound M for any positive integer M.*

Proof. Suppose, for the sake of contradiction, that there is a polynomial-time algorithm $\mathcal{A}$ with fixed error bound M for some positive integer M. For a 3SAT instance C, we make the same transformation to get the corresponding edge router assignment problem, except for setting the vulnerability to 1 and $12m \cdot M + 1$ for pairs in S and pairs outside of S, respectively. Similarly to the NP-completeness proof, it is easy to see that C is feasible if and only if the corresponding edge router assignment problem has an assignment that all destinations are assigned with the edge router pairs in S. If all destinations are assigned by edge routers pairs in S, then the total vulnerability is $12m$. If there is an assigned pair which is not in S, then the total vulnerability is at least $12m \cdot M + 1$. Thus, for a feasible assignment, its vulnerability is either $12m$ or at least $12m \cdot M + 1$. As algorithm $\mathcal{A}$ has fixed error bound M, if the optimal result is $12m$, it will give a result with the total vulnerability less than $12m \cdot M + 1$, which has to be $12m$. Thus, a 3SAT instance C is feasible if and only if the corresponding edge router assignment problem can be solved by $\mathcal{A}$ with the total vulnerability equal to $12m$. It is impossible unless $P = NP$. □

4.2 Feasibility Problem

If we simply want to find a pair of edge routers for each destination subject to the port number constraint of each edge router, regardless of the objective of minimizing the total vulnerability, then the problem, referred to as the *feasibility problem*, can be solved by being formulated as a maximum flow problem ([1]) as follows:

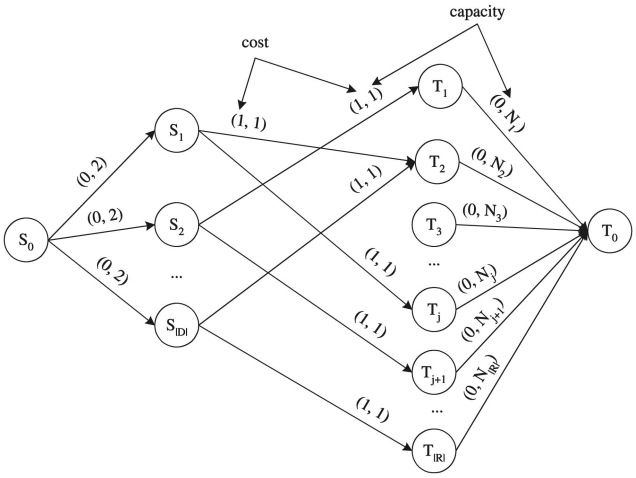

Fig. 4. The maximum flow model for the feasibility problem.

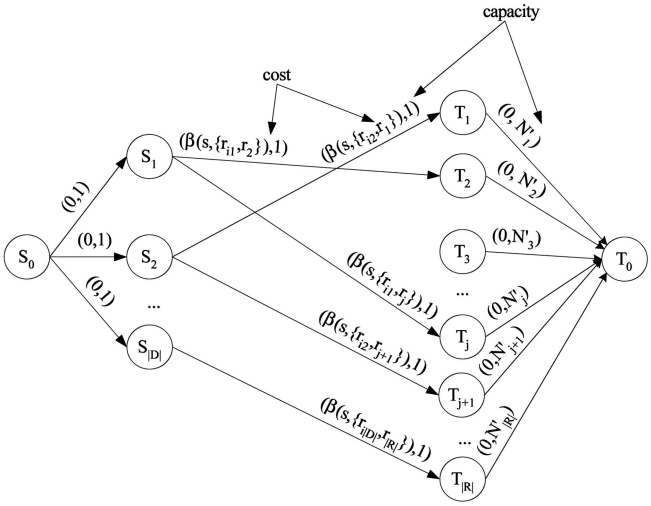

Fig. 5. The network flow model for the special case of the edge router assignment problem.

1. For each destination host d_i, we introduce a node S_i.
2. For each edge router r_j, we introduce an intermediate node, denoted by T_j, which is connected to a common sink node T_0 by a link with the capacity set as the port number constraint of edge router r_j.
3. Let the common supply node be S_0 and connect S_0 to each S_i, $i = 1, 2 \ldots, |D|$, by a link with a capacity of 2.
4. For each node S_i and intermediate node T_j, if destination d_i is allowed to connect to edge router r_j, we add a link from S_i to T_j with a unit capacity.

We then need to solve a maximum flow problem from node S_0 to node T_0. Fig. 4 shows an example of the constructed maximum flow model. It can be easily shown that the edge router assignment problem has a feasible solution if and only if the corresponding maximum flow problem has a solution of $2|D|$ units of flow. Therefore, the feasibility problem can be solved in $O(|D|^2|R|)$ since a maximum flow problem can be solved in $O(|D|^2|R|)$ time as in [1].

4.3 A Special Case of the Edge Router Assignment Problem

We now consider a special case for the edge router assignment problem in which the primary edge router for each destination host is predetermined. Under this special case, we only need to decide the backup edge router for each destination host such that the total vulnerability for the multicast session is minimized subject to the port number constraint of each edge router. This problem can be solved by the minimum cost network flow algorithms on a constructed network flow model. Let r_{i_k} be the primary edge router that destination host d_k is connected to. For each edge router r_j, let N'_j be the number of available ports except for the ports used by the destination hosts as primary edge routers. The network flow model is constructed as follows:

1. For each destination host d_k, we introduce a node S_k.
2. For each edge router r_j, we introduce an intermediate node, denoted by T_j, which is connected to a common sink node T_0 by a link with the capacity set as the remaining capacity of edge router r_j and the cost set as 0. Let the demand of node T_0 be $|D|$.
3. Let the common supply node be S_0 and connect S_0 to each S_k for $k = 1, 2 \ldots |D|$ by a link with the capacity set as 1 and the cost set as 0. Let the supply of node S_0 be $|D|$.
4. For each node S_k and intermediate node T_j ($r_j \neq r_{i_k}$), if destination host d_k is allowed to connect to edge router r_j, we add a link from S_k to T_j with a unit capacity and the cost set as $\beta(r_s, \{r_{i_k}, r_j\})$.

Fig. 5 illustrates the construction of such a network flow model. The solution to the minimum cost network flow problem corresponds to an optimal solution to the edge router assignment problem with fixed primary edge router. This problem can be solved in $O(|D|(|D||R| + (|D| + |R|)log(|D| + |R|)))$ time since we need to find $|D|$ successive shortest paths when solving the minimum network flow problem as in [1] and each takes $O(|D||R| + (|D| + |R|)log(|D| + |R|))$ time.

5 Heuristic Algorithms

In this section, we present two heuristic algorithms for solving the edge router assignment problem. One is a greedy algorithm, the other is a heuristic algorithm based on the special case of the edge router assignment problem with fixed primary edge routers.

5.1 Greedy Algorithm

The basic idea of the greedy algorithm is to repeatedly assign the best edge router pair to an unassigned destination (i.e., a destination which has no edge router pair assigned). The best edge router pair is the one that has the minimum vulnerability and does not cause the infeasibility of the assignment of other unassigned destinations after assigning this pair to the destination. Hence, the greedy algorithm always finds a solution as long as a feasible solution exists. Let D' represent the set of unassigned destinations, which is initialized as D. For each router r_j, we use p_j to represent the number of remaining available ports of r_j. Initially, $p_j = N_j$ for any $r_j \in |R|$. The greedy algorithm is listed below.

Algorithm Greedy algorithm (T, D):
begin
  sort all pairs of edge routers in the
  nondecreasing order of their vulnerability,
  let $Q = \{q_1, \ldots, q_{|Q|}\}$ be the sorted list,
  where $|Q| = |R| * (|R| - 1)/2$
  let $D' = D$
  for $r = 1$ to$|Q|$ do
    let $q_r = (r_j, r_k)$
    if $p_j = 0$ or$p_k = 0$ continue
    for $i = 1$ to $|D'|$ do
      if $r_j \notin R_i$ or $r_k \notin R_i$ continue
      if feasible solution still exists
      after assigning r_j and r_k to d_i then
        assign r_j and r_k to d_i
        $p_j = p_j - 1$
        $p_k = p_k - 1$
        $D' = D' - \{d_i\}$
        if $D' = \emptyset$ then break
end

In the greedy algorithm, we first sort edge router pairs in the nondecreasing order of their vulnerability and assign them to destinations one by one. Each edge router pair will be assigned to as many destinations as possible as long as it is still feasible to get other unassigned destinations assigned. It takes $O(|R|^2 \log(|R|))$ time for the sorting. It takes $O(|D|)$ time to initialize D'. The feasibility test takes $O(|D|^2|R|)$ time using the algorithm proposed for the feasibility problem. There are at most $O(|D||R|^2)$ feasibility tests; therefore, the total complexity is $O(|D|^3|R|^3)$.

5.2 Heuristic Algorithm Based on the Special Case of the Edge Router Assignment Problem

The second algorithm is based on the special case discussed in the previous section. We propose a heuristic algorithm for the general case (shortened as a heuristic algorithm). In this heuristic algorithm, we use a greedy approach to determine the primary edge router for each destination host and then use the solution given for the special case to determine the secondary edge router for each destination.

For each destination d_i and any $r_j \in R_i$, let

$$c_{ij} = \sum_{r_k \in R_i, k \neq j} \beta(r_s, \{r_j, r_k\}), \tag{14}$$

and

$$a_i = \sum_{r_j \in R_i} c_{ij}/|R_i|. \tag{15}$$

Intuitively, we should allow destination hosts with larger a_is to select their primary edge routers before destination hosts with smaller a_is. The heuristic algorithm is listed as follows:

Algorithm Heuristic algorithm (T, D):
begin
  for $d_i \in D$ do
    for $r_j \in R_i$ do
      calculate c_{ij} using (14)
    calculate a_i using (15)
  sort $d_i \in D$ in the nonincreasing order of a_i,
  let $d_{i_1}, d_{i_2}, \ldots d_{i_{|D|}}$ be the sorted list
  for $l = 1$ to $|D|$ do
    select $r_{j'}$ or $r_{k'}$ with
    $\beta(r_s, \{r_{j'}, r_{k'}\}) = \min\{\beta(r_s, \{r_j, r_k\}) | \forall r_j, r_k \in R_{i_l}\}$,
    and $p_{j'} > 0$, $p_{k'} > 0$
    let $k^* = j'$ or k'
    assign r_{k^*} as the primary edge router for d_{i_l}
    $p_{k^*} = p_{k^*} - 1$
  call the algorithm for the special case
  (discussed in Section 4.3) to assign a secondary edge
  router for each destination
end

It takes $O(|D||R|^2)$ time to calculate c_{ij}s and a_is for $i = 1, 2 \ldots, |D|$ and $j = 1, 2 \ldots, |R|$. It takes $O(|D| \log |D|)$ time for sorting. It takes $O(|D||R|^2)$ time to assign a primary edge router for each destination. It takes $O(|D|(|D||R| + (|D| + |R|)log(|D| + |R|)))$ time as the algorithm for the special case of edge router assignment needs. Hence, the time complexity of this heuristic algorithm is $O(|D||R|(|D| + |R|) + |D|^2 log(|D| + |R|))$, which is faster than the greedy algorithm.

5.3 Lower Bound

We now develop a lower bound in order to evaluate the effectiveness of our proposed heuristic algorithms. The lower bound is calculated based on the following network flow model:

1. For each destination d_i, we introduce a node S_i.
2. Let the common supply node be S_0 and connect S_0 to each S_i for $i = 1, 2 \ldots |D|$ by a link with the capacity being 1 and the cost being 0. Let the supply of node S_0 be $|D|$.
3. For each edge router pair (r_j, r_k), $j < k$, we introduce an intermediate node T_{jk}. If destination d_i can be connected to both edge routers r_j and r_k, we add a link from S_i to T_{jk} and set its capacity as 1 and its cost as $\beta(r_s, \{r_j, r_k\})$.
4. We also introduce $|R|$ sink nodes U_js, each corresponding to a router r_j. For each node T_{jk}, we add a link to node U_j and set its capacity as N_j, i.e., the port number constraint of router r_j, and its cost as 0. Note that, because $j < k$, there will be $|R| - 1$ such links from nodes T_{1k}s to node U_1, $|R| - 2$ such links from nodes T_{2k}s to node U_2, and so on.
5. Finally, we add a link from each node U_j to a common sink node U_0 and set its capacity as N_j as its cost is 0. Let the demand of node U_0 be $|D|$.

Fig. 6 illustrates the construction of such a network flow model. We claim that the optimal solution to a minimum cost network flow problem for the above constructed network is a lower bound of the total vulnerability. We denote it as LB_1. The justification is as follows: Dropping all port number constraints, it is clear that the minimum cost network flow solution generates a trivial lower bound for the minimum vulnerability. In the above network, the port number constraint of edge router r_1 is strictly enforced, while the port number constraints for other edge routers are partially enforced. For example, the port number constraint

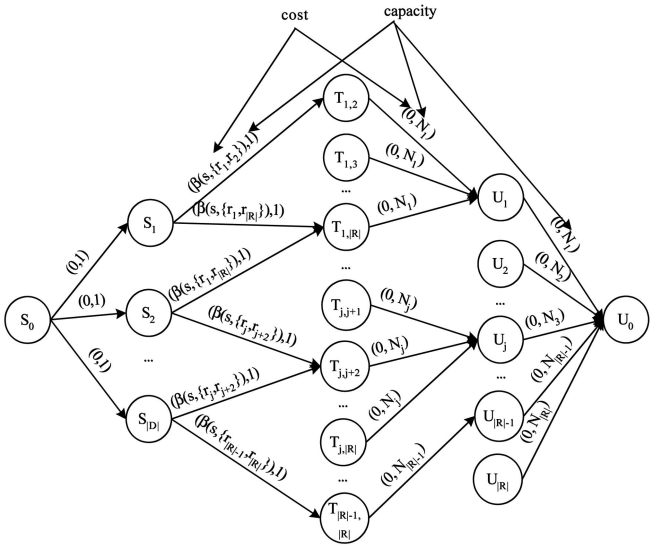

Fig. 6. The network flow model for calculating the lower bound.

of edge router r_2 can be violated only if some destination hosts use the pair (r_1, r_2) since we do not count those destination hosts in calculating the number of available ports on r_2. The port number constraint of edge router r_3 can be violated only if some destination hosts use the pairs (r_1, r_2), (r_1, r_3), (r_2, r_3), etc.

More lower bounds can be derived based on the above idea. In general, lower bound LB_i can be obtained if we reindex the edge routers from $(r_1, r_2, \ldots, r_{|R|})$ to $(r_i, r_{i+1}, \ldots, r_{|R|}, r_1, \ldots, r_{i-1})$ and repeat the above approach. We use the lower bound $LB = \max\{LB_i\}$ for all $i = 1, 2, \cdots, |R|$.

6 Simulation Results

In this section, we present the simulation results for the proposed heuristic algorithms and the comparison of their performance with the lower bound.

6.1 Simulation Settings

In the simulation, we use two parameters to characterize the network topology, the number of available edge routers $|R|$, and the maximum vulnerability value between any pair of edge routers, U.

The range of $|R|$ is set in $[100, 200]$, which can serve fairly large scale multicast sessions. To determine a reasonable value of U, we notice that the Internet diameter is tested to be around 10 hops [19], a value which can be used as a practical upper bound of the number of shared links between any two paths. For each pair of edge routers, we randomly generate a vulnerability value which is uniformly distributed in $\{0, 1, 2, \cdots, U\}$.

We use N to denote the maximum port number constraint for a router, which is determined by the routers placed in the network. For example, Cisco SOHO router series support 4-port and Cisco 3700 router series support up to 36 ports [20]. Based on such information, we choose the value of N from the set $\{8, 16, 32, 56, 64\}$. For each edge router r_j, its maximum available port number to a specific multicast session, N_j, is randomly generated to be uniformly distributed in $\{4, 5, \cdots, N\}$.

We also generate $|D|$ destination hosts. Note that $|D|$ is determined by the size of the multicast session, which is independent of $|R|$ or U. Let M be a parameter indicating the maximum number of feasible routers a destination host can be connected to. For each destination host, the number of feasible routers it can be connected to is randomly generated to be uniformly distributed in $\{2, 3, \cdots, M\}$. Note that M depends on the location of the destination set. As discussed in [2], the available number of ISPs at different cities varies from five to nine, which is a good indicator for the available edge routers of each destination host. As such, the range of M is set in $\{5, \ldots, 9\}$.

For each combination of parameters $|R|$, M, $|D|$, and N, we generate 200 instances. For each instance, we solve it using the greedy algorithm and the heuristic algorithm, respectively, and compute the lower bound of the total vulnerability. The performance of the two algorithms is evaluated by their relative errors compared with the lower bound. Let β_g denote the total vulnerability obtained by the greedy algorithm and β_l denote the total vulnerability given by the lower bound. The relative error of the greedy algorithm is defined as:

$$\epsilon_g = \frac{\beta_g - \beta_l}{\beta_l}.$$

Correspondingly, let β_h denote the total vulnerability obtained by the heuristic algorithm, the relative error of the heuristic algorithm is defined as:

$$\epsilon_h = \frac{\beta_h - \beta_l}{\beta_l}.$$

6.2 Effects of Number of Destinations and Number of Edge Routers

We first evaluate the performance of the proposed algorithms by fixing $M = 8$ and $N = 16$, while varying the number of destination hosts $|D|$ in $\{100, 120, \cdots, 200\}$ for $|R| = 100$ and varying the number of edge routers $|R|$ in $\{100, 120, \cdots, 200\}$ for $|D| = 200$, respectively. Fig. 7 and Fig. 8 show the relative errors of the two algorithms versus the number of destination hosts and versus the number of edge routers, respectively. As shown in Fig. 7, the relative errors of both the greedy and heuristic algorithms increase as the number of destination hosts increases. This result is consistent with our intuition. When there is a fixed total number of edge routers and more destination hosts, one edge router can be in the feasible edge router set of more destinations. Therefore, the assignment of one edge router pair to one destination tends to have a higher impact on the assignments of edge router pairs to other destinations, which explains the increasing trend of the relative errors.

As shown in Fig. 8, the relative errors of both the greedy and heuristic algorithms decrease as the number of edge routers increases. When there is a fixed number of destinations and more edge routers available, the feasible edge router sets for different destinations are less overlapped since the size of the feasible edge router set for each destination is the same. The assignment of one edge router pair to one

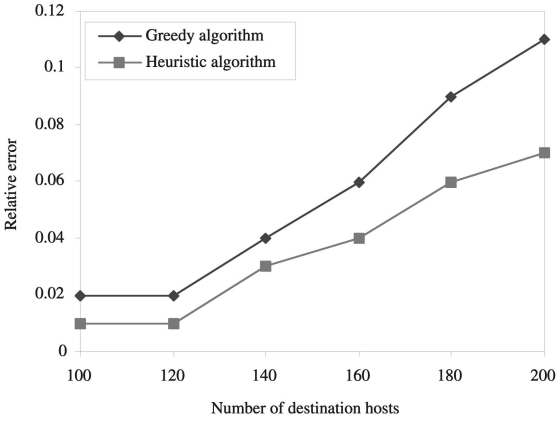

Fig. 7. Relative error of the heuristic algorithms versus the number of destination hosts.

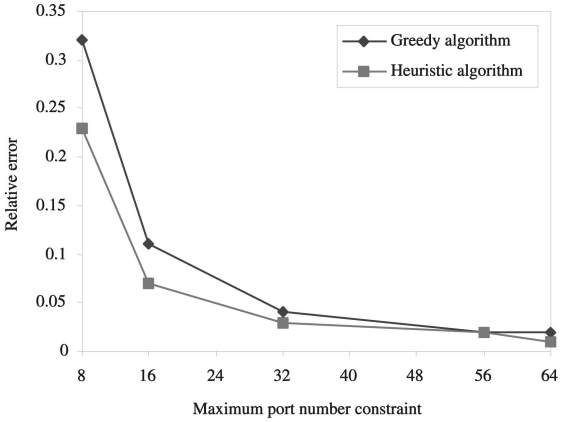

Fig. 9. Relative error of the heuristic algorithms versus the maximum port number constraint.

destination tends to have less impact on the assignments of edge router pairs to other destinations, which explains the decreasing trend of relative errors in Fig. 8.

6.3 Effects of Maximum Port Number Constraint and Number of Feasible Routers

In reality, the number of edge routers tends to be limited. We then evaluate the performance of the two algorithms by setting $|R| = 100$ and $|D| = 200$, while varying N in $\{4, 8, 16, 32, 56, 64\}$ for $M = 8$ and varying M in $\{5, 6, 7, 8, 9\}$ for $N = 16$. Fig. 9 and Fig. 10 show the relative errors of the two algorithms versus the maximum port number constraint and versus the maximum number of feasible edge routers each destination can be connected to, respectively. As shown in Fig. 9, the relative errors of both algorithms decrease as the maximum port number constraint increases. When there are more available ports for each edge router, more destinations can choose the same edge router pair with small vulnerability. Hence, the relative errors are smaller.

As shown in Fig. 10, the relative errors of both algorithms increase as the maximum number of feasible edge routers for each destination increases. When there are a fixed total number of edge routers and there are more feasible edge routers for each destination, there are more shared feasible edge routers among destinations. The assignment of one edge router pair to one destination tends to have a higher impact on the assignments of edge router pairs to other destinations. Therefore, the relative error tends to be larger.

As shown in Fig. 7, Fig. 8, Fig. 9, and Fig. 10, the relative error of the heuristic algorithm is smaller than that of the greedy algorithm. However, the heuristic algorithm cannot provide the guarantee of finding a solution whenever a feasible solution exists as the greedy algorithm does. Fig. 11 and Fig. 12 show the percentage of feasible instances with heuristic solution versus the number of destinations for $|R| = 100$ and versus the number of edge routers for $|D| = 200$, respectively. As shown in Fig. 11, when the number of destinations is large, it is less possible that the heuristic algorithm finds a solution. As shown in Fig. 12, when the number of edge routers is large, it is more possible that the heuristic algorithm finds a solution. These results are consistent with our expectation.

7 Discussion

In this paper, we proposed a scalable partial multicast protection scheme where no change to the routing algorithms and protocols in the core network is needed.

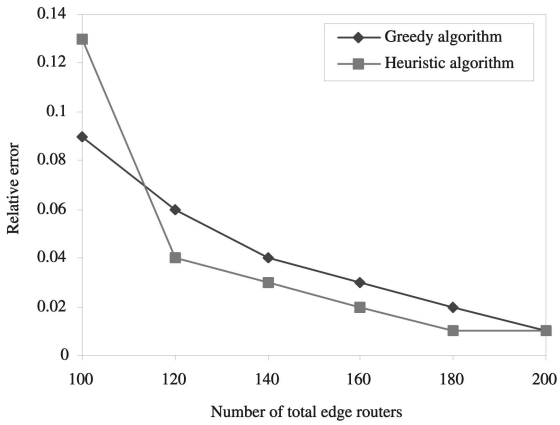

Fig. 8. Relative error of the heuristic algorithms versus the total number of edge routers.

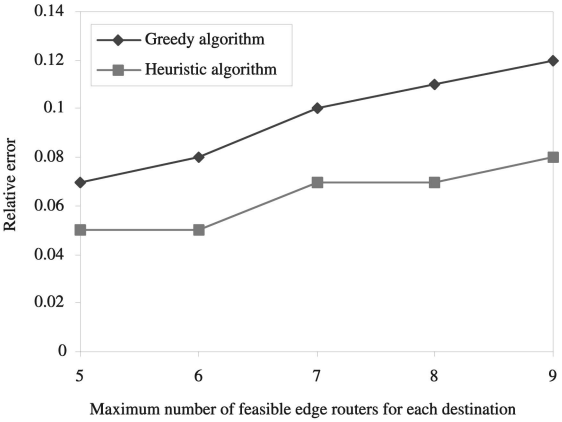

Fig. 10. Relative error of the heuristic algorithms versus the maximum number of feasible edge routers for each destination.

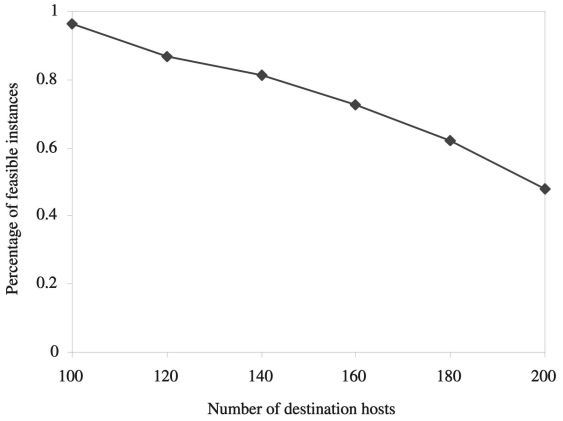

Fig. 11. Percentage of feasible instances with heuristic solution versus the number of destinations.

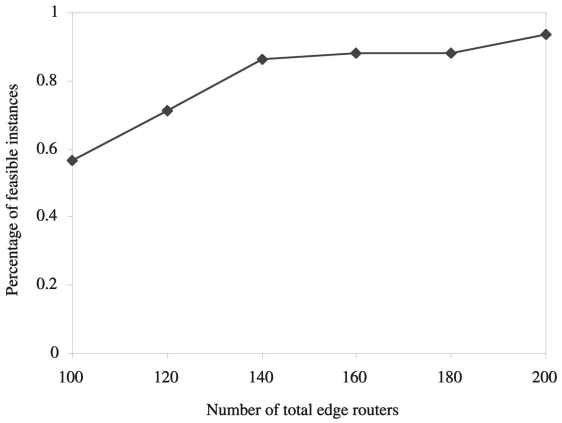

Fig. 12. Percentage of feasible instances with heuristic solution versus the total number of edge routers.

The proposed scheme is based on the dual-homing architecture, where each destination host is connected to two edge routers. Under such an architecture, the two paths from the source of a multicast session to the two edge routers provide protection for the traffic from the source to the destination. We introduced the concept of vulnerability to quantify the degree of protection level for a pair of edge routers and defined the edge router assignment problem whose objective is to determine the two edge routers for each destination such that the total vulnerability for the multicast session is minimized subject to the port number constraint of each edge router. We first proposed an optimal algorithm to calculate the vulnerability of any router pair. We then proved the NP-completeness of the decision version of the edge router assignment problem. We proposed two heuristic algorithms to solve the problem. Through simulations, we showed that the heuristic algorithm based on the special case is closer to the derived lower bound than the greedy algorithm. The trade-off is that the former one cannot always guarantee to obtain a feasible solution, while the latter one can.

Multicast is an important communication paradigm for IP networks. In the following, we discuss two typical application scenarios for the proposed research. In both cases, the protection to multicast can be implemented without any change in the core network. First, our proposed work can be used to determine the two edge routers to be connected to for a relatively static multicast session, for example, in a global enterprise, a multicast session from the headquarter to its worldwide branch offices. Each branch office can subscribe to two local different ISPs (edge routers) in order to provide protection for the multicast session. Our proposed work can help the branch offices make such decisions.

Second, our proposed work can also be used to dynamically determine the two edge routers for a multicast user. For example, according to the emerging *pay-per-use* business model for utility computing [7], each user (destination) can be physically connected to several available ISPs (edge routers) and pay the ISPs based on the traffic volume going through the ISPs. Under such an infrastructure, our proposed work can be used for each user to dynamically select a pair of edge routers to receive data from different multicast sessions at different time periods.

In summary, we believe that the proposed dual-homing partial multicast scheme provides a promising solution for achieving survivability for multicast communications.

ACKNOWLEDGMENTS

The authors would like to thank the Associate Editor, Dr. Sitharama Iyengar, and the anonymous reviewers for their comments that helped improve the paper. Part of the results were presented at IEEE GlobeCom 2004. This work was supported in part by the US National Science Foundation (NSF) under Grant number CNS-0435095.

REFERENCES

[1] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin, *Network Flows: Theory, Algorithms, and Applications.* Upper Saddle River, N.J.: Prentice-Hall, 1993.

[2] A. Akella, A. Shaikh, and R. Sitaraman, "A Measurement-Based Analysis of Multihoming," *Proc. ACM SIGCOMM,* pp. 353-364, 2003.

[3] D. Din and S. Tseng, "A Genetic Algorithm for Solving Dual-Homing Cell Assignment Problem of the Two-Level Wireless ATM Network," *Computer Comm.,* vol. 25, no. 17, pp. 1536-1547, Nov. 2002.

[4] A. Fei, J. Cui, M. Gerla, and D. Cavendish, "A Dual-Tree Scheme for Fault-Tolerant Multicast," *Proc. Int'l Computer Conf.,* vol. 3, pp. 690-694, 2001.

[5] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Bell Labs, 1979.

[6] M. Hiltunen, R. Schlichting, and C. Ugarte, "Building Survivable Services Using Redundancy and Adaptation," *IEEE Trans. Computers,* vol. 52, no. 2, pp. 181-194, Feb. 2003.

[7] M. Kallahalla, M. Uysal, R. Swaminathan, D.E. Lowell, M. Wray, T. Christian, N. Edwards, C.I. Dalton, and F. Gittler, "SoftUDC: A Software-Based Data Center for Utility Computing," *Computer,* vol. 37, no. 11, pp. 38-46, Nov. 2004.

[8] C. Lee and S. Koh, "A Design of the Minimum Cost Ring-Chain Network with Dual-Homing Survivability: A Tabu Search Approach," *Computer Operations Research,* vol. 24, no. 9, pp. 883-897, 1997.

[9] A. Orda and R. Rom, "Multihoming in Computer Networks: A Topology-Design Approach," *Computer Networks ISDN,* vol. 18, no. 2, pp. 133-141, 28 Feb. 1990,

[10] A. Phillips, J. Senior, R. Mercinelli, M. Valvo, P. Vetter, C. Martin, M. Deventer, P. Vaes, and X. Qiu, "Redundancy Strategies for a High Splitting Optically Amplified Passive Optical Network," *J. Lightwave Technology,* vol. 19, no. 2, pp. 137-149, Feb. 2001.

[11] A. Proestaki and M. Sinclair, "Design and Dimensioning of Dual-Homing Hierarchical Multi-Ring Networks," *IEE Proc.-Comm.*, vol. 147, no. 2, pp. 96-104, Apr. 2000.
[12] S. Ramamurthy and B. Mukherjee, "Survivable WDM Mesh Networks, Part I—Protection," *Proc. IEEE INFOCOM*, vol. 2, pp. 744-751, 2003.
[13] P. Sanjoy, *Multicast on the Internet and Its Applications.* Kluwer Academic, June 1998.
[14] G. Sasaki and C. Su, "The Interface between IP and WDM and Its Effect on the Cost of Survivability," *IEEE Comm. Magazine*, vol. 41, no. 1, pp. 74-79, Jan. 2003.
[15] J. Shi and J. Fonseka, "Analysis and Design of Survivable Telecommunications Networks," *IEE Proc.-Comm.*, vol. 144, no. 5, pp. 322-330, Oct. 1997.
[16] N. Singhal, L. Sahasrabuddhe, and B. Mukherjee, "Dynamic Provisioning of Survivable Multicast Sessions in Optical WDM Mesh Networks," *Proc. Optical Fiber Comm. Conf. (OFC)*, vol. 1, pp. 207-209, 2003.
[17] N. Singhal, L. Sahasrabuddhe, and B. Mukherjee, "Provisioning of Survivable Multicast Sessions against Single Link Failures in Optical WDM Mesh Networks," *J. Lightwave Technology*, vol. 21, no. 11, pp. 2587-2594, Nov. 2003.
[18] D. Xu, Y. Xiong, and C. Qiao, "Novel Algorithms for Shared Segment Protection," *IEEE J. Selected Areas in Comm.*, vol. 21, no. 8, pp. 1320-1331, Oct. 2003.
[19] http://www.msblabs.org/as-attack/report-100.txt, 2006.
[20] Cisco Systems, Cisco Router Guide, http://www.cisco.com/application/pdf/en/us/guest/products/ps5855/c1031/cdccont_0900aecd8019dc1f.pdf, 2005.

Jianping Wang received the BSc and MSc degrees in computer science from Nankai University, Tianjin, China, in 1996 and 1999, respectively, and the PhD degree in computer science from the University of Texas at Dallas in 2003. She is currently with the Department of Computer Science at the University of Mississippi. Previously, she was an assistant professor in the Department of Computer Science at Georgia Southern University. Her research interests include optical networks and wireless networks. She is a member of the IEEE.

Mei Yang received the PhD degree in computer science from the University of Texas at Dallas in August 2003. She is currently an assistant professor in the Department of Electrical and Computer Engineering, University of Nevada, Las Vegas (UNLV). Before she joined UNLV, she worked as an assistant professor in the Department of Computer Science, Columbus State University, from August 2003 to August 2004. Her research interests include computer networks, wireless sensor networks, computer architecture, and embedded systems. She is a member of the IEEE.

Bin Yang received the BS degree in computer science from Nankai University, China, in 1993, the MS degree in mathematics from Southern Illinois University in 1996, the MS degree in computer science from Texas A&M University in 1998, and the PhD degree from the University of Texas at Dallas in 2005. In 1998, he joined the telecommunication industry and worked at Ericsson Inc. and Cisco Systems. Currently, he is a software engineer in the Optical Network Group at Cisco Systems. His primary research interest is in networking algorithms.

S.Q. Zheng received the PhD degree from the University of California, Santa Barbara, in 1987. After being on the faculty of Louisiana State University for 11 years, he joined the University of Texas at Dallas in 1998, where he is currently a professor of computer science, computer engineering, and telecommunications engineering. His research interests include algorithms, computer architectures, networks, parallel and distributed processing, telecommunications, and VLSI design. He has published approximately 200 papers in these areas. He served as the program committee chairman of numerous international conferences and the editor of several professional journals. He is a senior member of the IEEE.