ISP and Egress Path Selection for Multihomed Networks

Amogh Dhamdhere, Constantine Dovrolis
{*amogh, dovrolis*}*@cc.gatech.edu*
Networking and Telecommunications Group
Georgia Institute of Technology

***Abstract*— Multihoming has been used by stub networks for several years as a form of redundancy, improving the availability of Internet access. More recently, Intelligent Route Control (IRC) products allow multihomed networks to dynamically switch parts of their egress or ingress traffic between ISPs, also improving cost and performance. IRC products assume that the set of upstream ISPs is given and fixed. Typically, however, a multihomed network has several ISP choices and the actual selection of ISPs can significantly affect cost, availability, and performance. In the first part of this work, we develop a methodology to select the best set of upstream ISPs, optimizing monetary cost and availability. Our results, based on measurements of actual Internet traffic and topology, show that the proposed algorithm selects the best possible set of ISPs in terms of resiliency to inter-AS single-link failures. The algorithm also performs well in the presence of double or triple link failures. In the second part of this work, we focus on the egress path selection problem. Specifically, we propose a stochastic search algorithm, based on simulated annealing, to allocate the network's egress traffic between upstream ISPs. The objectives are to minimize cost, also ensuring that the selected paths to the major destinations of egress traffic are congestion-free. Simulation results show that the proposed algorithm performs very well in meeting the previous objectives, when congestion-free paths exist.**

Keywords: Multihoming, Intelligent Routing, Path Diversity, Network Measurements, Simulated Annealing.

I. Introduction

Multihoming refers to the connection of a stub/edge network to more than one Internet Service Providers (ISP) [1]. Networks that focus on reliability and availability have used multihoming as a form of redundancy for several years. In the most common deployment scenario, one ISP is used as the primary provider while another is used as backup when the primary fails. Switching from the primary to the backup can be performed automatically by the border router of the multihomed network when it detects loss of connectivity with the primary ISP. The use of multihoming has seen a dramatic increase in the last few years. It is now estimated that more than 70% of the stub networks in the US are multihomed to at least two ISPs. The widespread proliferation of multihoming is due to several reasons. First, as more and more enterprises rely heavily on the Internet for their transactions, reliability and availability become of primary importance. Second, multihoming can be used to drive down the costs of Internet access. This is the case when the multihomed network can use a lower-cost ISP for their bulk traffic and a higher-cost but better ISP for more performance-sensitive traffic.

This work was supported by the NSF CAREER award 0347374, NSF award 0519756, and by an equipment donation from Intel Corporation.

Multihoming capabilities have expanded tremendously with the use of "Intelligent Route Control" (IRC). Multihoming-IRC systems allow a stub network to automatically switch parts of their ingress or egress traffic from one provider to another, driven by cost and/or performance considerations. A number of vendors are currently providing such systems [2], [3], [4], [5], [6], [7], [8], [9].

IRC products typically assume that the set of upstream ISPs has been already chosen and it is fixed. A stub network, however, has several choices of upstream ISPs. The problem of which ISPs to select has been largely ignored so far, or it has been viewed as a non-technical decision (a notable exception is the recent work by Wang et al. [10]). Intuitively, a good set of ISPs should provide low cost, good performance, and significant path diversity (for robustness to network failures), at least for the major traffic destinations. In the first part of this work, we propose a methodology to select a set of upstream ISPs taking into account monetary cost, inter-domain level performance, and path diversity considerations. We show, based on traffic and topology measurement data, that the proposed algorithm can improve robustness to single inter-AS link failures by selecting the best possible combination of ISPs. The algorithm also performs well in the presence of double and triple inter-AS link failures.

Once the set of ISPs has been selected, the egress traffic can be routed so that we minimize the cost incurred by the source network, subject to the important constraint that the chosen outgoing paths do not experience persistent congestion. This is different from the current IRC practice, which is to change dynamically the traffic allocation in a reactive manner, to avoid transient periods of congestion. In the second part of this work, we propose an algorithm for egress path selection that determines a congestion-free solution (if it exists) with minimum cost for the source network. This is a challenging problem, because the source network does not have a priori knowledge about the topology and capacity of the upstream paths that reach each of its major destinations. We propose a stochastic search algorithm based on simulated annealing to find a feasible egress path for each major destination. Using simulations, we show that this algorithm performs well in meeting the objectives, when a feasible path selection exists.

This rest of the paper is structured as follows. Section II summarizes the related work in this area. Section III states

our objectives and describes the network and traffic models. In Section IV, we describe the ISP selection problem and the proposed methodology, which is then evaluated in Section V. In Section VI, we describe the egress path selection problem and our stochastic search solution, which is then evaluated in Section VII. We conclude in Section VIII.

II. Related Work

We discuss two bodies of related work. The first focuses on the problem of ISP selection for a multihomed network. The second deals with the allocation of egress traffic to the selected set of ISPs to improve performance and/or cost.

[11] was one of the first papers to consider the optimization of multihomed networks. That early work took a topological approach, with nodes representing potential attachment points for subscribers, and constraints on the number of subscribers that a node allows. The objective was to find the optimal set of nodes that each subscriber should connect to with the aim of minimizing the distance between the subscriber and every other node in the network. After several years, the ISP subscription problem was studied again in [10]. That work proposed algorithms for selecting a set of upstream ISPs with a (monetary) cost minimization objective. A main difference between the methodology of [10] and our work is that we also consider performance and path diversity objectives. In particular, we are interested in choosing ISPs that provide significantly diverse paths to the major destinations of traffic. Also, the results of [10] are rather specific to a class of pricing functions that are based on the "percentile charging" model.

Existing IRC products choose egress paths dynamically, avoiding congestion in a reactive manner. Even though most commercial multihoming-IRC systems do not expose much technical information about their internal operation, one of them (the ISMD device of Rether Networks) is described with significant detail in a research publication [12]. Another good description and evaluation of an operational multihoming-IRC system is given in [13]. These papers, however, do not consider the monetary cost of allocating traffic to different upstream ISPs. The recent work by Goldenberg et al. [14] approaches the traffic allocation problem with the objective of optimizing performance given a maximum cost constraint. That work considers latency as the performance metric, and proposes algorithms to dynamically reroute egress traffic upon transient congestion periods. The results of [14] are also based on the percentile charging model, while we use a more general pricing model.

An experimental study, based on measurements from the Akamai content distribution network, showed that multihoming can lead to significant benefits in terms of both availability and performance for both ingress and egress traffic [15]. The authors also showed that having up to four upstream providers is enough to gain the full benefit of multihoming. Another experimental work that evaluated the benefits of multihoming is described in [16].

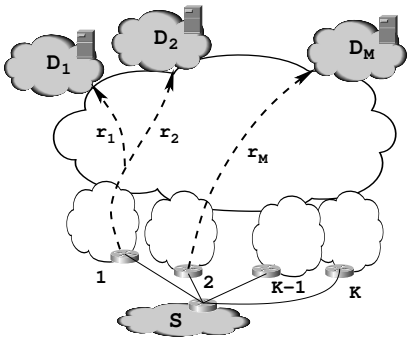

Fig. 1. A multihomed network with K upstream ISPs, and M major destinations

III. Problem Description and Objectives

The aim of this work is to provision the multihoming configuration of a source network S. We assume that S is a content provider, i.e., mostly a source rather than destination of traffic. Hence, we are concerned with the *egress* traffic from S. We also assume that before exploring multihoming options, the administrator of S has a good idea about the outgoing traffic profile at S. In particular, the operator knows the set of M "major destinations" that account for a large fraction of the outgoing traffic from S. These major destinations can be large networks. So, what we refer to as "flows" destined to these major destinations are large aggregates, rather than individual end-to-end flows. The operator of S should also have an estimate for the average rate of the traffic that is sent to each of these major destinations. It is possible to assemble this traffic profile through passive measurements at the outgoing links of S. Figure 1 shows our basic underlying model: a source network connected to the Internet through K ISPs, with each ISP providing a network path (potentially different) to each of the M destinations. Provisioning the multihoming configuration of S involves the following two tasks:

1) Choose the ISPs that S will subscribe to. Typically, several ISPs would have a point-of-presence at the location of S. These ISPs may differ based on their performance, the level of reliability that they guarantee, and of course their pricing policies. S will choose K of these ISPs as upstream providers. K depends on the level of reliability that S desires, and also on monetary or other practical constraints (e.g., number of available ports at the border router). We assume that K is given. We term this part of the problem as *ISP Selection*.
2) Once the set of K ISPs has been chosen, the operator of S can determine the ISP that should be used to reach each of the M major destinations. The objective is to minimize the total cost paid to the K ISPs, subject to the constraint that none of the chosen paths to the major destinations is congested. We term this part of the problem as *Egress Path Selection*.

Note that the ISP selection phase is "semi-static", meaning that it would be typically performed over very long timescales,

say months. The set of upstream ISPs would be modified only if there are major changes in the destinations of the outgoing traffic or in the underlying ISP market.

The egress path selection problem aims to find an allocation of destinations to ISPs that minimizes cost and avoids long-term congestion. However, there may still be periods of short-term congestion in which some paths are overloaded and see poor performance. These short-term congestion events can be dealt with in a reactive way using dynamic path switching, as done by most IRC products. However, such dynamic path switching can result in a sub-optimal configuration. Hence, the egress path selection algorithm should be run periodically so that S returns to a more optimized configuration. We envision that the path selection phase would be repeated every few hours, while dynamic path switching could take place in the time scales of seconds to deal with short-term congestion.

Our provisioning objectives are determined by the following factors:

1) **Cost:** S aims to minimize the total cost incurred for routing its egress traffic through the chosen set of upstream ISPs. Each ISP has a pricing function that is used to determine the cost it charges to S. A common practice is that the cost charged by an ISP depends on the total amount of traffic that it receives from a customer (volume-based pricing). In this work, we avoid any specific pricing function, mostly because different ISPs use significantly different pricing models. In our simulator, each pricing function is an increasing and concave function of the total traffic volume routed through that ISP.
2) **Performance:** S aims to avoid routing its traffic through congested paths. This is the objective of the path selection phase. We focus on long-term congestion, resulting from gross misallocation of traffic. For example, routing the traffic to a major destination, say 10Mbps, through an ISP that reaches that destination with a 5Mbps link.
3) **Robustness:** S should select ISPs that provide significant *path diversity* to its major destinations. Selecting ISPs with path diversity provides resiliency to upstream network failures. With a selection of ISPs that provides significant path diversity, a working alternate path is likely to exist in case the primary path to a destination fails or becomes congested. On the other hand, if the K paths to a certain destination are largely overlapping, then it is likely that none of the upstream paths can avoid the point of failure or congestion.

IV. Phase I - ISP Selection

A. Problem statement

In this section, we focus on the problem of selecting the egress ISPs for a source network S. Let $\mathcal{I}$ be the set of possible ISPs to which S can subscribe, out of which K will be selected. $\mathcal{D}$ represents the set of M major destinations of S. $\mathcal{R}$=$\{r_i\}$ is the average traffic rate destined to each destination in $\mathcal{D}$. We assume that the K links have the same capacity A. The operator of S can determine an appropriate value of A based on the average rate of the egress traffic, the number K, and the desired utilization level under ideal load balancing. For example, if the egress rate is 400Mbps and $K = 4$, the average load of each link would be 100Mbps. Since link capacities are typically available in a few discrete steps (e.g., Fast Ethernet, OC-3, OC-12), S would probably subscribe to four OC-3 links in this example. The assumption of equal capacities simplifies the ISP and egress path selection problems, and it may be necessary for practical reasons (e.g., homogeneous border router interfaces). Our approach can be easily modified for the case that S connects to each ISP with a different (but known) capacity.

Both monetary cost and performance should be taken into account when selecting ISPs. An important issue is whether S can evaluate the performance of an ISP *before* it actually subscribes to that ISP. Typically, an ISP would not allow a network S to perform extensive probing and performance measurements before S becomes its customer. However, most ISPs maintain public Looking Glass Servers (LGS). LGSs are routers inside an ISP that report AS-level paths to given destination networks. Many ISPs today deploy LGSs at different points-of-presence, mostly for the diagnosis of inter-domain routing problems. Here, we assume that each ISP in $\mathcal{I}$ has a LGS from which S can determine the AS-level paths to destinations in $\mathcal{D}$.

The ISP selection problem takes into account the following three factors:

Monetary Cost: The cost of routing the traffic of S through the selected set of ISPs should be minimized. The actual cost cannot be determined until S subscribes to a set of ISPs and allocates its traffic among them. This is because the cost charged by an ISP depends on the total amount of traffic that is routed through it. However, as we show next, there is a way for S to estimate the cost that would be incurred with each selection of ISPs.

AS-level path length: The AS-level paths to the networks in $\mathcal{D}$ through the chosen ISPs should be as short as possible. Long paths tend to translate into larger delays, and are more vulnerable to interdomain routing failures and pathologies.

Path diversity: The K chosen ISPs should be such that S has the maximum possible path diversity to the destinations in $\mathcal{D}$. Large path diversity improves the robustness to upstream network failures and congestion events. We define an AS-level path diversity metric later in this section.

Since we have to pick K ISPs out of $|\mathcal{I}|$ choices, there are $\binom{|\mathcal{I}|}{K}$ possible selections. With each selection we associate a cost metric for each of the three aforementioned factors. So, the ISP selection process can be viewed as an optimization problem with the objective to minimize the total (generalized) cost. $|\mathcal{I}|$ would typically not be higher than 10-20 ISPs. Hence, it is tractable to enumerate all possible combinations of K out of $|\mathcal{I}|$, and then choose the selection that minimizes the total cost. For example, if $|\mathcal{I}| = 15$ and $K = 4$, there are 1365 possible combinations. Recall that the ISP selection process is performed over very long timescales, and so exhaustive search

of all 1365 combinations should be feasible.

For each combination $\mathcal{C}$ of K ISPs, let $c_m(\mathcal{C})$ be the monetary cost, $c_p(\mathcal{C})$ be the cost associated with the AS-level path length, and $c_d(\mathcal{C})$ be the cost associated with path diversity. These three cost terms will be defined in the following paragraphs. To get the total cost $c_t(\mathcal{C})$ for the selection $\mathcal{C}$, we form a weighted sum of the previous three costs as follows

$$c_t(\mathcal{C}) = \alpha_m c_m(\mathcal{C}) + \alpha_p c_p(\mathcal{C}) + \alpha_d c_d(\mathcal{C}) \tag{1}$$

where α_m, α_p and α_d are the corresponding normalization factors. The administrator of S can choose the values of these factors depending on the relative importance of each factor.

Let $\mathcal{C}^{\mathcal{I}}$ be the set of all possible combinations of K ISPs from the set $\mathcal{I}$, with $|\mathcal{C}^{\mathcal{I}}| = \binom{|\mathcal{I}|}{K}$. For each selection $\mathcal{C} \in \mathcal{C}^{\mathcal{I}}$, we calculate $c_t(\mathcal{C})$, and the optimal choice of ISPs is the selection $\mathcal{C}^*$ with the minimum total cost, i.e.,

$$\mathcal{C}^* = \text{argmin}_{\mathcal{C}\in\mathcal{C}^{\mathcal{I}}} c_t(\mathcal{C}) \tag{2}$$

Note that it is not necessary to use this particular additive cost function given in Equation (1). Since we use a "brute force" approach to find the best set of ISPs, any cost function that expresses the total cost in terms of the three different cost components can be used instead.

B. Monetary cost

Each ISP j has a pricing function $f_j(T_j)$, where T_j is the total traffic routed through it. The total monetary cost of a selection $\mathcal{C}$ is $c_m(\mathcal{C})$=$\sum_{j\in\mathcal{C}} f_j(T_j)$. Note that T_j, and hence the total cost, depends on how the traffic of S is allocated to the ISPs in $\mathcal{C}$. This allocation however is not known before S subscribes to $\mathcal{C}$. To deal with this problem, we estimate $c_m(\mathcal{C})$ as the *minimum* cost that would be incurred to route the traffic of S through the set $\mathcal{C}$ of ISPs. To compute this cost we need to solve the following lower-level optimization.

For a given ISP selection $\mathcal{C}$, let $j = G(i)$ represent the ISP that carries the traffic to destination i; we refer to the function $G(\cdot)$ as a "mapping". There are K^M possible ways to map the M flows in $\mathcal{D}$ to $\mathcal{C}$. Let $\mathcal{G}$ be the set of all such mappings. Some of these mappings may be infeasible, because the amount of traffic routed through one or more ISPs exceeds the corresponding access capacity. So, the minimum monetary cost for the selection $\mathcal{C}$ is

$$c_m(\mathcal{C}) = \min_{G\in\mathcal{G}} \sum_{j\in\mathcal{C}} f_j(T_j) \tag{3}$$

where the minimization is performed over all possible mappings in $\mathcal{G}$, subject to the constraints

$$T_j < A, \quad j = 1 \dots K \tag{4}$$

This is a variation of the bin-packing problem with M items of size r_i ($i = 1 \dots M$) and K bins, each of capacity A. The bin packing problem is NP-hard and so we need to use a heuristic solution, especially if the number of destinations is large. The heuristic that we use is similar to the *First Fit Decreasing* (FFD) algorithm. The basic idea is to start with the largest destination, in terms of rate, and route it through the lowest-cost ISP in which it satisfies the capacity constraint. Algorithm 1 shows the pseudo-code for our heuristic. The total running time of the algorithm is $O(MlogM)+O(MKlogK)$. Simulations showing the performance of the FFD algorithm, in terms of being able to find a solution when one exists, and in terms of finding the optimal solution, are presented in Section VII.

Algorithm 1 FFD-like heuristic

Require: Rates $\mathcal{R} = \{r_i\}$, $i = 1 \dots M$
Require: Access capacity A of each ISP
Require: Pricing functions $\mathcal{F} = \{f_j\}$, $j = 1 \dots K$
Initialize $G = null$ {Constructed mapping}
Initialize $T_j = 0$, $j = 1 \dots K$ {Total rate through each ISP}
Initialize $\bar{A}_j = A$, $j = 1 \dots K$ {Residual access capacity of each ISP}
Sort destinations in decreasing order of r_i
for each destination i in sorted sequence **do**
    $c_j = f_j(T_j + r_i)$, $j = 1 \dots K$ {Cost if destination i was mapped to ISP j}
    Sort c_j in increasing order
    for each ISP j in sorted sequence **do**
        if $\bar{A}_j > r_i$ **then**
            $T_j = T_j + r_i$
            $G(i) = j$ {Map destination i to ISP j}
            $\bar{A}_j = \bar{A}_j - r_i$
            break {Route next destination}
        end if
    end for
end for
if there is a destination that could not be routed **then**
    return null
else
    return G {final mapping}
end if

Finally, the monetary cost of selection $\mathcal{C}$ is given by

$$c_m(\mathcal{C}) = \sum_{j\in\mathcal{C}} f_j(T_j) \tag{5}$$

where the traffic through ISP j is the sum of the rates of the destinations that are routed through ISP j, i.e.,

$$T_j = \sum_{i:G^*(i)=j} r_i \tag{6}$$

and G^* is the mapping reported by Algorithm-1

$$G^* = \text{FFD}(\mathcal{R}, A, \mathcal{F}) \tag{7}$$

It is possible that Algorithm-1 will fail to find a feasible mapping. The simulations in Section VII show that that happens, almost always, when there is no feasible mapping. Also, the same simulation results show that the cost of the mapping reported by Algorithm-1 is within 5% of the minimum cost.

C. AS-level path length cost

The calculation of the AS-level path length cost for a selection $\mathcal{C}$ proceeds along similar lines as the monetary cost. Let $p_j(i)$ denote the AS-level path length to reach a destination i through ISP j. Thus, we can think of $p_j(i)$ as a cost for a given destination-ISP pair. As we did for monetary cost, the total path length cost of an ISP selection $\mathcal{C}$ can be estimated as the *minimum* that can be obtained with $\mathcal{C}$. The minimization is performed over all possible mappings in $\mathcal{G}$, given the ISPs in $\mathcal{C}$, i.e.,

$$c_p(\mathcal{C}) = \min_{G \in \mathcal{G}} \sum_{i=1 \ldots M, j=G(i)} p_j(i) \tag{8}$$

subject to the constraints

$$T_j < A, \quad j = 1 \ldots K \tag{9}$$

This is identical to the monetary cost problem, except that the cost function in this case is given by path lengths. We use the same FFD algorithm to compute the optimal mapping G^*

$$G^* = \text{FFD}(\mathcal{R}, A, \mathcal{P}) \tag{10}$$

where $\mathcal{P}$ is the set of path length costs from the ISPs in $\mathcal{C}$ to the destinations in $\mathcal{D}$. If such a mapping exists, the minimum path length cost is given by

$$c_p(\mathcal{C}) = \sum_{i=1 \ldots M} p_j(i) \quad \text{where } j = G^*(i) \tag{11}$$

D. Path diversity cost

A selection $\mathcal{C}$ of ISPs provides K paths to each destination i. We focus on AS-level paths, because these paths can be directly observed through LGSs. If an inter-AS link is shared by *all* K paths to i, then a failure of that link will make i unreachable. For single-link failures, a destination i will become unreachable *only* by the failure of a link shared by all K paths. We call such links as *K-shared*. Obviously, a selection of ISPs that has fewer K-shared links will provide better resiliency to inter-AS link failures.

For a destination i and a selection of ISPs $\mathcal{C}$, we define the path diversity metric $\kappa(i, \mathcal{C})$ as the number of K-shared links to destination i though the ISPs in $\mathcal{C}$. We use $\kappa(i, \mathcal{C})$ as the cost term for path diversity,

$$c_d(i, \mathcal{C}) = \kappa(i, \mathcal{C}) \tag{12}$$

The path diversity cost for a selection of ISPs $\mathcal{C}$ is then given by the sum of $c_d(i, \mathcal{C})$ over all destinations, weighted by the rate of each destination,

$$c_d(\mathcal{C}) = \sum_{i=1 \ldots M} r_i * \kappa(i, \mathcal{C}) \tag{13}$$

We choose this weighted average, so that there is a higher cost associated with the potential failure of large destinations.

V. Phase I - Path Diversity

The evaluation of Algorithm-1, used in the minimization of the monetary and AS-level path length costs, appears in Section VII. In this section, we focus on the path diversity cost instead.

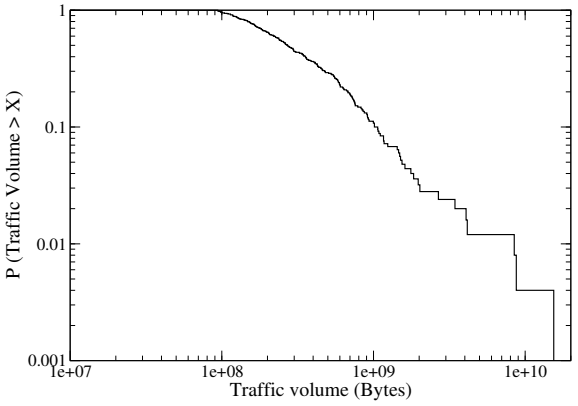

Fig. 2. Complementary CDF of egress traffic to the 250 largest destination networks.

A. Destination networks and rate distribution

To evaluate the achievable path diversity from a real network, we first need to characterize its main destinations of traffic and the corresponding rate distribution. We analyzed a large packet trace from the Internet egress link of our university's network. This campus network is a significant source of traffic because it hosts some popular Web and FTP servers. For each destination IP address, we find the corresponding destination network by searching for the longest matching prefix in the BGP routing table of the border router. We then measure the total traffic to each destination network and rank those destinations based on their aggregate rate. The top 500-1000 destination networks account for about 80-90% of the total egress traffic, while the top 250 destinations account for about 65% of the traffic. In the following, we work with those 250 largest destinations (M=250). Figure 2 shows the complementary CDF of the traffic volume to those destinations. The approximately linear distribution on the log-log plot indicates a Pareto distribution. We estimated the shape parameter as $\alpha = 1.08$ using the *aest* tool [17].

B. AS-level paths

To determine the cost with respect to path length (c_p) and path diversity (c_d), we need to know the AS-level paths to the major destinations in $\mathcal{D}$ through each of the ISPs in the set $\mathcal{I}$. For this purpose, we make use of the LGSs that many ISPs provide. In this paper, we considered nine ISPs with points-of-presence in Atlanta (Qwest, Level 3, SAVVIS, Broadwing, Williams Communications, Teleglobe, Cogent, Global Crossing, and 1A Networks). Each of them provides an LGS. We queried these nine LGSs for each of the 250 destination prefixes. The collected AS-level paths provide us the required information for calculating c_p and c_d.

C. Evaluation of path diversity

We wrote a simple simulator that takes as input the AS-level topology from each of the nine potential ISPs to each of the 250 destination networks. This topology is basically nine different trees with the same 250 leaves, rooted in each

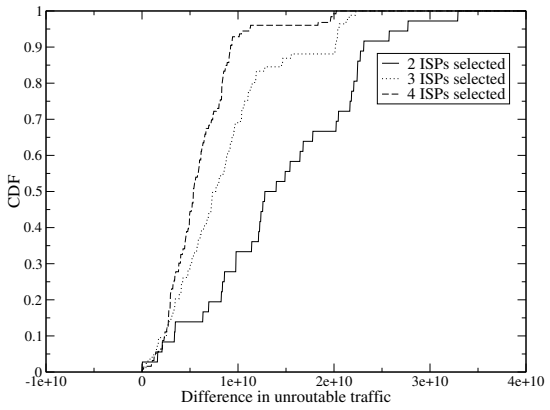

Fig. 3. CDF of Δu for single-link failures.

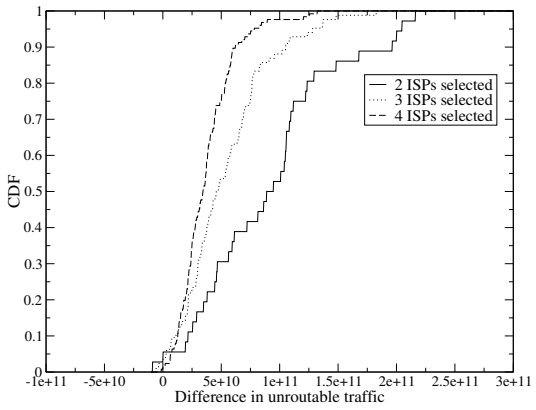

Fig. 4. CDF of Δu for double-link failures.

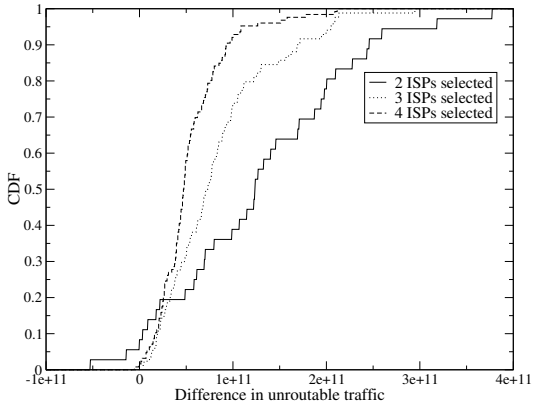

Fig. 5. CDF of Δu for triple-link failures.

of the ISPs. The internal nodes represent traversed ASes and the edges represent inter-AS links. The nine trees share some internal nodes and edges. We next calculate the path diversity metric $\kappa(i, \mathcal{C})$ for each destination i and selection of ISPs $\mathcal{C}$.

Given a specific number K, we compute the selection $\mathcal{C}^*$ of K ISPs with the minimum path diversity cost, as described in Section IV. To evaluate the robustness of that selection, the simulator considers each link in the topology and counts the amount of traffic that would not be routable if that link will fail. Recall that with single-link failures, a destination is unreachable only if a K-shared link fails.

Let $u(\mathcal{C})$ be the total amount of traffic that would not be routable with a selection of ISPs $\mathcal{C}$, considering all possible single-link failures in the topology. The difference between the unroutable traffic with an arbitrary selection $\mathcal{C}$ and our selection $\mathcal{C}^*$ is

$$\Delta u(\mathcal{C}) = u(\mathcal{C}) - u(\mathcal{C}^*) \tag{14}$$

Figure 3 shows the CDF of $\Delta u(\mathcal{C})$ for K=2, 3 and 4 ISPs. Note that all differences $\Delta u(\mathcal{C})$ are positive or zero, confirming that our selection $\mathcal{C}^*$ is optimal in terms of providing robustness to single-link failures. This is not surprising, as $\mathcal{C}^*$ minimizes the number of K-shared links, which represent the Achilles' heel for single-link failures. Furthermore, $\Delta u(\mathcal{C})$ can often be very large, meaning that a selection of ISPs that ignores path diversity can lead to poor availability.

We repeated the previous experiments for double and triple link failures. As the number of possible failures in that case is very large, the simulator randomly picks 1000 cases of double or triple link failures, and measures the traffic that would not be routable through the K chosen ISPs in an arbitrary $\mathcal{C}$. Figures 4 and 5 show the corresponding CDFs of $\Delta u(\mathcal{C})$. In this case, the selection $\mathcal{C}^*$ is not always optimal. Especially when we only have two ISPs (K=2), there are some selections of ISPs that are better than $\mathcal{C}^*$ in terms of unroutable traffic. Nevertheless, $\mathcal{C}^*$ is still among the best 5% of ISP selections in terms of robustness to double and triple link failures, and it is almost optimal when we have three or four ISPs.

VI. Phase II - Egress Path Selection

A. Problem statement

Once Phase-I is completed, S is connected to the Internet through the set $\mathcal{C}^*$ of the K best ISPs. In Phase-II, we aim to determine an optimal path allocation for each of the M destinations in $\mathcal{D}$. The notation in Phase-II remains the same as in Phase-I. In particular, $G(\cdot)$ is the "mapping function" such that destination i is mapped to ISP $j = G(i)$ ($j \in \mathcal{C}^*$), and $\mathcal{G}$ is the set of all possible mappings. The main objective in Phase-II is to determine a mapping in $\mathcal{G}$ that minimizes the total cost, given by $\sum_{j \in \mathcal{C}^*} c_j(T_j)$, subject to the constraint that none of the paths $P_{i,j}$ to the destinations in $\mathcal{D}$ is congested. A path $P_{i,j}$ to destination i through ISP j is congested if it has a positive loss rate $l(P_{i,j})$ >0. Note that congestion can occur either at the access links of S or in the upstream networks. Hence, Phase-II can be stated as the following problem: Determine the mapping $G \in \mathcal{G}$ that minimizes the cost

$$\min \sum_{j \in \mathcal{C}^*} c_j(T_j)$$

subject to the constraint:

$$l(P_{i,j}) = 0 \quad \text{for all } i \in \mathcal{D}.$$

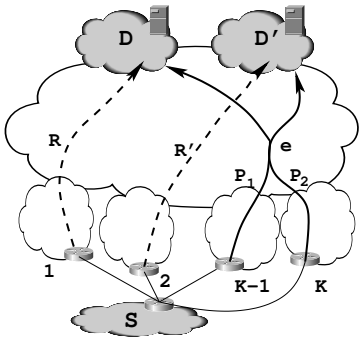

Fig. 6. Link e is not the bottleneck of paths P_1 and P_2, but it can become the joint bottleneck of the two paths when they are used simultaneously.

The previous problem may appear at first as a classical network flow problem. This is not the case however. Network flow and optimal routing problems assume that the topology of the network is given, and that the capacity of each link is known. In our context, this is not the case. Even though S knows the capacity of its own access links to the K ISPs, it does not know the topology or the capacity of the upstream network paths $P_{i,j}$. This is a key issue in Phase-II and it is the main reason we consider stochastic search techniques in the following.

Note that even though "traceroute" measurements can be used to infer the topology of upstream paths, such measurements do not provide us information about the capacity of those paths. Without such information it is not possible to determine whether a given mapping will result in congestion-free paths.

Also, even though there are techniques to estimate the available bandwidth in a path through end-to-end measurements, those techniques cannot determine the available bandwidth in a bottleneck link that is simultaneously used by two or more paths from S. To illustrate this point, suppose that two paths from S share a link e with available bandwidth $A(e)$, as shown in Figure 6. The available bandwidth in the two paths is $A(P_1)$ and $A(P_2)$, and let us assume that $A(P_1) + A(P_2) > A(e)$ while $A(P_i) < A(e)$, $i = 1, 2$. This means than e is *not* the bottleneck of the two paths when they are considered in isolation, but it is their shared bottleneck when they are jointly used. Existing available bandwidth estimation tools can measure $A(P_i)$, but they cannot measure $A(e)$. Obviously, we need an estimate of $A(e)$ to infer the maximum traffic load that the two paths can jointly carry.

In conclusion, if we cannot know a priori whether a given mapping will be congestion-free or not, we need to consider *iterative routing approaches*. By iterative routing we mean that S routes its egress traffic based on a certain mapping for some time while measuring the loss rate in the corresponding paths. If any of these paths is congested the traffic is rerouted based on a different mapping. The loss rate $l(P_{i,j})$ in path $P_{i,j}$ can be estimated with active probing or passive measurements at the border router of S.

An iterative routing approach has the drawback that it causes rerouting. This can be a problem for TCP-based or streaming applications. Consequently, when the minimum-cost mapping is not congestion-free and routing iterations are necessary, we allow a certain cost increase while trying to keep the amount of rerouted and dropped traffic as low as possible. Note that if the minimum-cost mapping is congestion-free, then the path allocation problem is solved without routing iterations.

B. The algorithm

We propose a two-step algorithm. In the first step, we assume that the bottlenecks of all paths $P_{i,j}$ are the K access links of S. So, if the minimum-cost mapping is such that the traffic T_j routed through ISP j is less than the access link capacity A, that allocation will also be congestion-free. Under this assumption, the optimal path allocation problem is reduced to a variation of the bin-packing problem, for which we have already presented an efficient heuristic (Algorithm-1).

In the second step of the algorithm, we route the traffic based on the minimum-cost mapping and examine whether any of the egress paths is congested. If that is the case, our earlier assumption about the location of the bottlenecks is false, and the congestion occurs somewhere in the upstream networks. A simulated annealing algorithm is then invoked to search for a congestion-free mapping in the vicinity of the minimum-cost solution, while trying to reduce the amount of rerouted and dropped traffic.

The key idea behind this two-step approach is that in many cases the bottlenecks are the access links. The reason is that most core networks and private peering points between major ISPs are currently overprovisioned. Consequently, we expect that this assumption will usually result in a good, if not optimal, mapping. If some paths are congested elsewhere in the network, then the stochastic search component of the algorithm performs a local modification of the initial mapping, rerouting only the congested egress flows.

C. Initial mapping

The initial mapping is computed with Algorithm-1. In Phase I, we used that algorithm to examine whether a set of K ISPs provides a feasible mapping, and to determine the minimum cost of that mapping. Here, we use the set C^* that resulted from Phase I to route the egress traffic towards the destinations in $\mathcal{D}$. Note that we only consider the M largest destinations of outgoing traffic. The rest of the destinations should be also accounted for. We assume that that part of S's traffic is evenly distributed among the K ISPs, and so the access capacity A in Algorithm-1 refers to the residual capacity that is available for the largest M destinations.

D. Stochastic search and simulated annealing

Simulated annealing was first proposed by Kirkpatrick [18] as a general methodology within the area of stochastic search and optimization. The underlying idea is based on the physical process of annealing (cooling) in the chemical industry. Simulated annealing has been applied with slight variations to many combinatorial optimization problems (for instance, see

[19], [20], [21]). In its most general form, the algorithm starts with an initial solution and an initial temperature. At each iteration, it first evaluates the cost of the current solution. If the cost is found to be unacceptable, the algorithm generates a new candidate solution, typically modifying certain aspects of the current solution. If the cost of the new solution is lower ("downhill move"), the solution is always accepted. Otherwise, the solution is accepted with a probability $e^{\frac{-\Delta c}{T}}$ ("uphill move"), where Δc is the cost increase due to the new solution and T is the current temperature. This is called the Metropolis criterion [22]. Accepting a move with increasing cost helps the algorithm to avoid local minima. The temperature T decreases across successive iterations, diminishing the possibility of uphill moves and forcing the algorithm to eventually terminate. The algorithm exits when a solution with an acceptable cost is found, or when the temperature has reached a certain "freezing point". The pseudocode of the basic simulated annealing algorithm is shown in Algorithm-2. Some parts of the algorithm that are more specific to our problem are described in the following paragraphs.

Algorithm 2 Simulated annealing pseudocode for Phase II

```
Calculate initial temperature T
Get initial mapping G from Algorithm I
Route traffic as in mapping G
c_curr = cost(G)
repeat
    if c_curr = 0 then
        return G {congestion-free solution}
    else
        Generate new solution G_new {as described in text}
        Route traffic as in mapping G_new
        c_new = cost(G_new)
        if c_new − c_curr ≤ 0 then
            G = G_new
            c_curr = c_new {new mapping is better}
        else
            With probability e^{−(c_new−c_curr)/T},
            G = G_new and c_curr = c_new {Metropolis criterion}
        end if
        T = ρT {cooling rate}
    end if
until T ≈ 0
```

Cost function: Recall that our objective is to find a congestion-free mapping in the vicinity of the minimum-cost mapping provided by the bin-packing step of the algorithm. Consequently, we consider a cost function that measures the overall congestion experienced by S's egress traffic. Specifically, suppose that $l(P_{i,j})$ is the loss probability measured at path $P_{i,j}$ after the last routing iteration. The cost (overall congestion) $c_c(G)$ of a mapping G is the total rate of dropped traffic, across all destinations in $\mathcal{D}$,

$$c_c(G) = \sum_{i=1\ldots M} r_i l(P_{i,j}) \tag{15}$$

where r_i is the average rate to destination i.

Initial temperature: In [23], Kirkpatrick suggests that the initial temperature should be chosen so that the probability of accepting an uphill move from the initial solution G_0 is about 0.8. We also assume that, initially, the worst move that should be accepted is one that at most doubles the initial cost. Hence, the initial temperature T_0 is set to $T_0 = \frac{-c_c(G_0)}{ln(0.8)}$.

Generating a new solution: A critical part of the algorithm is to determine a new mapping G_{new}, with lower cost than the current mapping G_{curr}. Since our cost function is congestion-related, we consider ways to reroute one or more congested destinations. We first simulated various schemes that reroute multiple congested flows at the same time. Those schemes performed consistently worse than schemes that move a single flow at a time. Hence, we examine mappings in which G_{new} and G_{curr} differ in the path of a single destination. Second, each time we reroute a destination, we allocate it to the ISP that will result in the *minimum cost increase*, among the set of ISPs with sufficient residual access capacity. The third issue is to determine the particular destination that should be rerouted. We evaluated the following three heuristics:

1) **Max-cost:** Reroute the congested destination with the highest cost in the current mapping.
2) **Max-loss:** Reroute the congested destination i with the highest loss rate $r_i l(P_{i,j})$ in the current mapping.
3) **Min-rate:** Reroute the congested destination i with the lowest rate r_i.

Due to space constraints, we do not present a detailed evaluation and comparison of the previous three algorithms. To summarize the results of this simulation study, we found that the Max-loss performs best in terms of minimizing the amount of dropped traffic (as we would expect), but it also performs best in terms of the number of routing iterations. The Min-rate algorithm is better in terms of minimizing the amount of rerouted traffic, but the Max-loss algorithm does not do significantly worse. Consequently, in the following, we use the Max-loss algorithm.

Annealing Schedule: The annealing schedule determines the rate at which the temperature is decreased. The related literature proposes mostly geometric cooling for large combinatorial optimization problems [21]. Our simulations showed that an annealing schedule with very slow cooling rate, such as $\rho = 0.99$, is more suitable for our problem.

Termination conditions: Depending on the capacity and topology of the underlying network, a congestion-free mapping may not exist for a given traffic load. Allowing the simulated annealing algorithm to keep searching for a feasible solution until the temperature drops to zero may cause significant rerouting. Consequently, we set the following additional termination conditions.

1) The monetary cost of any considered mapping should not be too large. Specifically, if the monetary cost of a

mapping becomes larger than a factor $cost\_thresh$=2 of the initial cost, the search terminates.

2) If the congestion cost has not decreased significantly over a number of iterations, it is likely that there is no feasible solution. Specifically, if the congestion cost has not decreased by at least a factor $cong\_thresh$=1.1 in any of the last $iter\_inc\_cong$=10 iterations, the search terminates.

VII. Phase II - Evaluation

The objectives of this section are twofold. First, to evaluate Algorithm-1, needed in both Phase-I and Phase-II. Second, to evaluate Algorithm-2 of Phase-II, as well as some simpler algorithms for solving the same problem. The evaluation of the two algorithms is performed with flow-level simulations that use measured datasets for the outgoing traffic distribution and the underlying IP-layer topology.

A. Measured traffic and topology datasets

To simulate the model of Figure 1 more realistically, we rely on the following three measured datasets. First, as described in Section V-A, we collected traces of the outgoing traffic from our university network to determine the 250 largest destinations (accounting for about 65% of the total traffic) and the distribution of traffic among them. We found that that distribution can be modeled as Pareto with shape parameter 1.08.

The second dataset is related to the network topology of the "upstream cloud" from S to the major destination networks. To simulate the upstream ISPs of S, we used three Planetlab nodes that are geographically located in the state of New York: Columbia University, New York University, and Cornell University. These three hosts provide us with different IP-layer paths from the same (roughly) geographical area to different destinations of traffic.

Third, to simulate the IP-layer paths from the upstream ISPs to the major destination networks we used two approaches. First, we collected traceroute data from the previous three Planetlab nodes to the 250 largest destination networks in our university packet trace. Unfortunately, traceroute cannot report the entire route to several destination networks. However, if the traceroute outcomes from all three Planetlab nodes merge at a common intermediate node after a certain point, we consider that node (router) as the traffic destination. In the second approach, we run traceroute from the three Planetlab nodes to 100 randomly chosen destination IP addresses from Caida's Skitter datasets [24]. Of course, the drawback of this approach is that randomly picked destinations may not be large traffic sinks in reality.

B. Simulator parameters

The simulator aims to route M flows, modeling the traffic to each of the destination networks, through a given network topology. Each flow can originate from one out of K=3 access links. The flows are modeled as constant fluids, i.e., they are completely specified by a rate; we do not consider the short-term effects of traffic variability. The key simulation parameters are the following:

Flow rates and destination ranking: As previously noted, the distribution of flow rates follows the Pareto distribution. The average rate, across all M flows, controls the load in the network. Different simulation random seeds result in different flow rates. We always assign the largest rate to the same destination, the second largest rate to another destination, and so on. In other words, we assume that even though the traffic to each destination varies across simulations, the flows retain their ranking. We believe that this property resembles the characteristics of real egress traffic better than assuming that, for example, the 10th largest destination today can be the 100th largest destination tomorrow.

Location of bottlenecks: Each path $P_{i,j}$ from S to destination i through ISP j has a bottleneck link, which is the link with the minimum capacity. A parameter $bneck\_loc$, between 0 and 1, controls the location of that bottleneck. This parameter determines the link of $P_{i,j}$ that has the largest probability of being the bottleneck. Neighboring links also have a probability of being the bottleneck, which decreases geometrically with their distance from the most likely bottleneck. $bneck\_loc$=0 means that the access links of S are the most likely bottlenecks. $bneck\_loc$=1 means that the access links of the destination networks are the most likely bottlenecks. A value of $bneck\_loc$ around 0.5 will bring the bottlenecks close to the core of the upstream network.

Shared bottlenecks: As illustrated in Figure 6, a network link that is shared by two or more paths can become their joint bottleneck when those paths are used simultaneously. The presence of shared bottlenecks can cause strong coupling between paths to different destinations. For example, switching the traffic of destination i from ISP j to ISP j' can cause congestion in other paths and destinations, not routed through j'. A parameter $bneck\_shar$, between 0 and 1, controls the probability that a link which is shared by two or more paths is their joint bottleneck. By joint bottleneck we mean that that link becomes congested only when *all* those paths are active, i.e., used to reach the corresponding destinations. If only some of those paths are active, the shared link will not be congested. $bneck\_shar$=0 means that a shared link is never a joint bottleneck, while $bneck\_shar$=1 means that a shared link is always a joint bottleneck.

C. Evaluation of Algorithm-1

In this section, we focus on the evaluation of Algorithm-1. Recall that that algorithm attempts to identify the minimum-cost allocation of M destinations to K access links, subject to the same capacity constraint for each link. We are interested in two major questions. First, can Algorithm-1 find a solution to the previous problem, when a solution exists? And second, what is cost of the solution reported by Algorithm-1 relative to the cost of the optimal solution? Both questions require exhaustive search in order to know whether a solution exists and, when that is the case, the cost of the optimal solution. For this reason, in this part of the evaluation study we limit

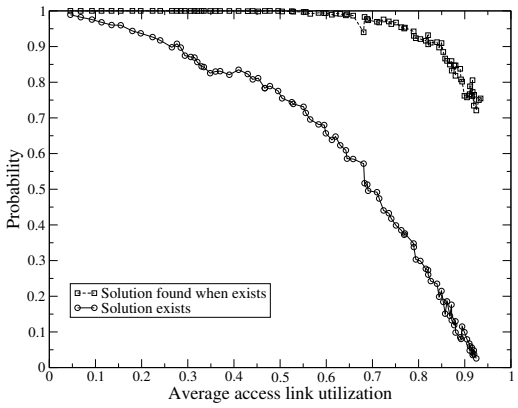

Fig. 7. Probability that solution exists, and probability that solution is found by Algorithm-1 when it exists.

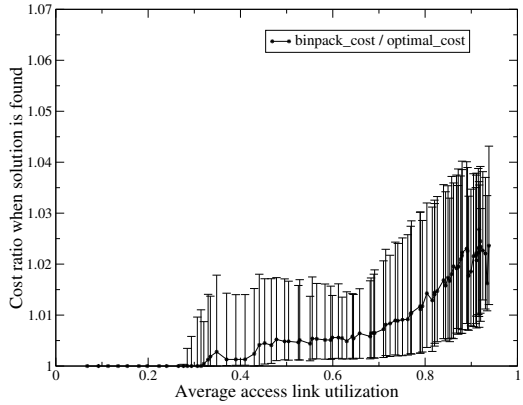

Fig. 8. Cost ratio between Algorithm-1 solution and optimal solution.

the topology to 10 randomly picked destinations from the 250 destinations included in our topology.

The $bneck\_loc$ factor is set to 0, meaning that the destinations are bottlenecked at the K access links. For each value of the average flow rate, Algorithm-1 and the exhaustive search routine are run 5,000 times. We then estimate the following metrics for each value of the average flow rate. First, the probability that a solution exists. Second, the probability that Algorithm-1 will find a solution, when a solution exists. Third, the cost ratio of the solution reported by Algorithm-1 and the optimal solution, when a solution exists.

Figure 7 shows the first two metrics, as a function of the average utilization of the access links (each value of the average flow rate corresponds to a different utilization). Note that as the load increases to more than 20%-30%, the probability to find a feasible allocation drops significantly, to less than 80%-90%. This "early saturation" effect is a result of the heavy-tailed nature of the Pareto distribution: a few flows have very large rate relative to the individual link capacities. The good news is that Algorithm-1 can identify a solution with very high probability (practically 100%) when a solution exists, as long as the average load is below 60%-70%.

Another positive result is that Algorithm-1 results in almost the minimum-cost solution, when a solution exists. Figure 8 shows the median and the inter-quartile range for the cost ratio between the Algorithm-1 solution and the optimal solution. We see that the median cost ratio is very close to 1, and the 75^{th} percentile value is less than 1.05.

D. Evaluation of Algorithm-2

In this section, we focus on the evaluation of Algorithm-2. Recall that the objective of that algorithm is the minimum-cost assignment of each egress flow to an upstream ISP, subject to the constraint that none of the egress paths is congested. Algorithm-2 is based on stochastic search and it may need several routing iterations before it finds a solution. We refer to the time period during which the algorithm searches for a solution as the "transient phase". We are interested in the following questions. First, what is the probability that Algorithm-2 will find a solution, as we increase the offered load from S? Second, how long is the transient phase in terms of the required number of iterations? Third, what is the total amount of dropped traffic due to congestion during the transient phase? And fourth, what is the total amount of rerouted traffic due to routing iterations during the transient phase? In the following evaluation study, we use the entire topology (250 destinations). An exhaustive search in a topology of this scale would be computationally prohibitive. Hence, we do not present results for the probability that a solution exists or for the cost of the optimal solution.

Instead of presenting results only for Algorithm-2, we also evaluate the following simpler algorithms. The objective of these comparisons is to get some insight in the relative performance of Algorithm-2 and to understand the significance of simulated annealing compared to methods that follow the "greedy search" paradigm.

1) **Access link bottlenecked (access-link):** In the simplest case, Phase-II can assume that all egress flows are bottlenecked at the K access links. In that case, Algorithm-1 can be used to approximate the minimum-cost congestion-free allocation. This algorithm does not require routing iterations.
2) **Greedy, moving a single flow at each iteration (greedy-single):** This is similar to Algorithm-2, but without the simulated annealing component. In each iteration, the flow with the highest loss rate (Max-loss) is moved to the ISP that will cause the minimum cost increase, among the ISPs with sufficient access link capacity. The algorithm never accepts uphill moves.
3) **Greedy, moving multiple flows at each iteration (greedy-mult):** In each iteration, the congested flows are first ordered in decreasing order of their loss rate. Then, each flow in that sequence is moved to the minimum-cost ISP that has sufficient access link capacity. Note that this is the only algorithm that moves more than one flow in the same iteration.
4) **Simulated annealing variations (SA-fast and SA-slow):** We examine two variations of Algorithm-2, one with very fast cooling (ρ=0.5), called SA-fast, and

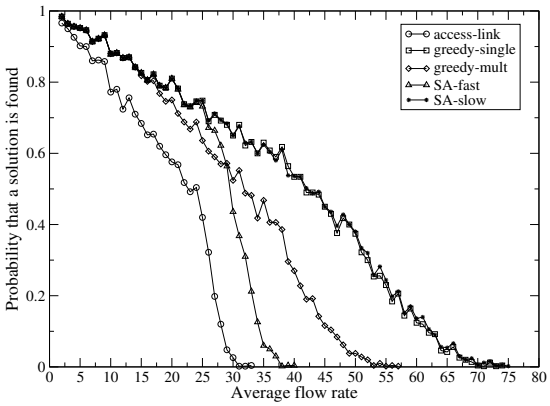

Fig. 9. Probability that a solution is found.

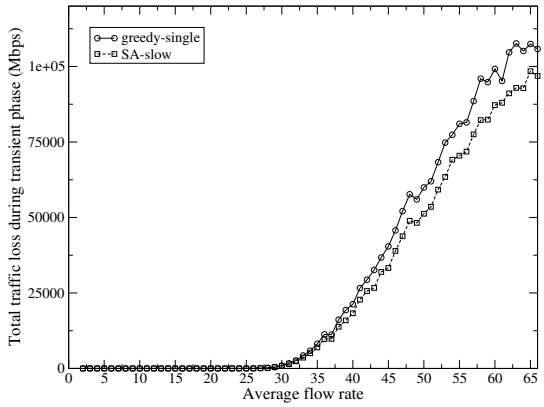

Fig. 11. Total traffic loss during transient phase.

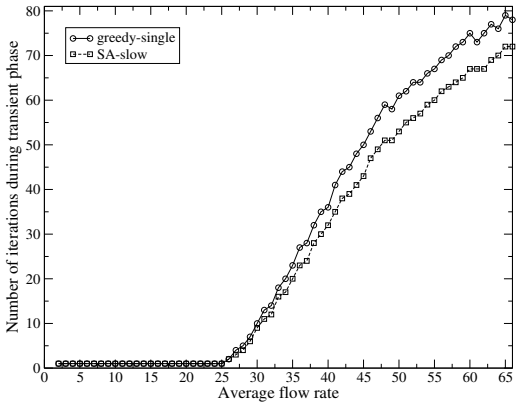

Fig. 10. Number of iterations during transient phase.

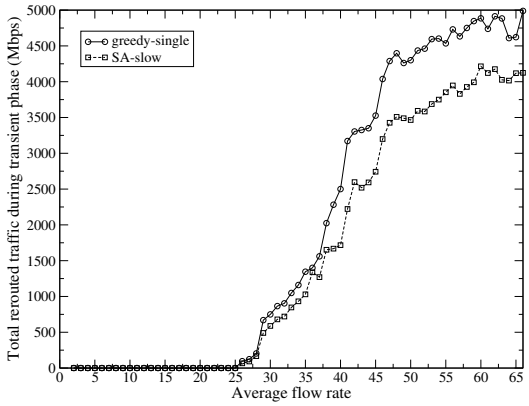

Fig. 12. Total rerouted traffic during transient phase.

another with very slow cooling (ρ=0.99), called SA-slow.

In the first set of simulations, we set the path bottlenecks at the access links of the three upstream ISPs. In this case, if a solution exists, we expect that Algorithm-1 will find it and Algorithm-2 will terminate without any routing iterations. If a solution does not exist, on the other hand, then the iterative routing approach of Algorithm-2, or of any other iterative algorithm, would obviously not help. The simulation results confirmed this intuition.

In the second set of simulations, we set the path bottlenecks inside the network ($bneck\_loc$=0.5). The shared links are not joint bottlenecks ($bneck\_shar$=0). Figures 9 to 12 show the simulation results for this configuration. In terms of the probability to find a solution, Figure 9 shows that *SA-slow* and *greedy-single* perform better than the other algorithms, and they actually give very similar results. On the other hand, *SA-fast* has significantly lower success probability than *SA-slow* after the load has become significant. The reason is that, when cooling happens very fast, a simulated annealing algorithm terminates too early, before it has the chance to find a solution. Also note that *greedy-mult* does not perform as well as the *greedy-single* algorithm. It is also interesting that even though *access-link*, *SA-fast*, and *greedy-mult* show a rapid decrease of the success probability after a certain load, the algorithms *SA-slow* and *greedy-single* observe a much more gradual degradation. Based on these results, in the following we focus on *SA-slow* and *greedy-single*.

Figure 10 shows the median number of iterations, across 500 simulation runs, until the algorithm terminates. Note that *SA-slow* performs better than *greedy-single*, especially in heavy load conditions, i.e., when the probability of success with these two algorithms is less than about 50%. We also calculated the confidence intervals for the number of routing iterations (not shown here). The two algorithms have wide and significantly overlapping confidence intervals. This means, first, that there is significant variability across different simulations (flow rates). The reason for this is that different rates across simulation runs could lead to different initial mappings produced by Algorithm-1. This, in turn could lead to a different set of congested flows and loss rates, significantly affecting the dynamics of the iterative algorithms. Consequently, even though *SA-slow* needs fewer iterations on the average, there is a significant fraction of simulations in which *greedy-single* performs better.

The amount of rerouting that an algorithm introduces is also important. We calculate the cumulative rate of rerouted traffic during the transient phase by counting after each iteration the

total rate of the flows that were rerouted. Figure 12 shows a similar trend with the number of routing iterations: *SA-slow* performs better than *greedy-single*, especially in heavy load conditions. The confidence intervals, however, again show significant variability and overlap. We see similar trends for the total rate of dropped traffic during the transient phase, shown in Figure 11.

In summary, the simulated annealing algorithm (with slow cooling) performs better, at least on the average, than a greedy algorithm which never accepts uphill moves. The difference between the two algorithms is more significant in terms of the number of required routing iterations and the amount of rerouted and dropped traffic in heavy load conditions, when the probability of finding a solution is less than 50%.

We also performed simulations in which the shared links are also joint bottlenecks ($bneck\_shar$=1). The results follow similar trends as the previous graphs, implying that the performance of the *SA-slow* and *greedy-single* algorithms does not depend significantly on the presence of joint bottlenecks.

VIII. Conclusions

Multihoming and Intelligent Route Control allow stub networks to optimize their cost, performance and availability without relying on complex end-to-end optimizations and interdomain traffic engineering. Before the wide deployment of IRC technologies, however, we need to first understand in depth their capabilities and limitations and to design systematic methodologies for their most efficient and safe usage.

In this paper we focused on methodologies for the provisioning of egress routing at a multihomed content provider. We proposed a two-phase methodology. In Phase-I, the objective is to choose the optimal set of upstream ISPs given a coarse traffic profile that captures the average rate and the AS path to each major destination network. The optimality of the resulting set of ISPs is determined by a weighted average of their monetary cost and AS path length, and of the path diversity that any set of ISPs provides. In Phase-II, the objective is to assign the traffic towards each destination to an upstream ISP so that the total monetary cost is minimized, without experiencing long-term congestion. The main difficulty in Phase-II is that the available bandwidth of the upstream network paths is generally unknown. Hence, we need to use stochastic search techniques and iterative routing. We envision that Phase-I can be repeated in long time scales, from weeks to months, while Phase-II can be repeated whenever there is a major change in the egress traffic distribution.

There are several open questions in the area of multihoming and IRC. First, it is important to examine whether the joint cost and performance optimization formulation that we proposed is consistent with practical constraints and current practices of ISPs and content providers. Second, we need to study the interaction between a path selection methodology, such as what we propose in Phase-II, with dynamic path changes performed in response to transient congestion events. We expect that the latter can gradually push the multihomed network to a high-cost operating configuration. Third, the problem of ingress traffic provisioning should be also investigated. Unfortunately, BGP does not allow the simultaneous use of several routes towards the same destination. Breaking up the destination prefix into multiple distinct prefixes is a possible solution, but it does not provide fine granularity in the traffic allocation and it increases the size of interdomain routing tables.

References

[1] T. Bates and Y. Rekhter, "Internet RFC 2260: Scalable Support for Multi-homed Multi-provider Connectivity," January 1998.

[2] InterNAP, "Premise-Based Route Optimization."

[3] Route Science, "Adaptive Networking Software."

[4] f5 networks, "BIG-IP Link Controller."

[5] Radware, "Peer Director."

[6] Rainfinity, "RainConnect."

[7] Stonesoft, "StoneGate Multi-Link Technology."

[8] FatPipe, "WARP."

[9] Cisco Systems, "Optimized Edge Routing (OER)."

[10] H. Wang, H. Xie, L. Qiu, A. Silberschatz, and Y. R. Yang, "Optimal ISP Subscription for Internet Multihoming: Algorithm Design and Implication Analysis," in *Proceedings of IEEE Infocom*, 2005.

[11] A. Orda and R. Rom, "Multihoming in Computer Networks: a Topology-design Approach," *Comput. Netw. ISDN Syst.*, vol. 18, no. 2, pp. 133–141, 9/90.

[12] F. Guo, J. Chen, W. Li, and T. Cker, "Experiences in Building a Multihoming Load Balancing System," in *Proceedings of IEEE Infocom*, 2004.

[13] A. Akella, S. Seshan, and A. Shaikh, "Multihoming Performance Benefits: An Experimental Evaluation of Practical Enterprise Strategies," in *Proceedings of USENIX Annual Technical Symposium*, 2004.

[14] D. K. Goldenberg, L. Qiuy, H. Xie, Y. R. Yang, and Y. Zhang, "Optimizing cost and performance for multihoming," in *SIGCOMM '04: Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*. New York, NY, USA: ACM Press, 2004, pp. 79–92.

[15] A. Akella, B. Maggs, S. Seshan, A. Shaikh, and R. Sitaraman, "A Measurement-Based Analysis of Multihoming," in *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*. New York, NY, USA: ACM Press, 2003, pp. 353–364.

[16] S. Tao, K. Xu, Y. Xu, T. Fei, L. Gao, R. Guerin, J. K. and D. Towsley, and Z. Zhang, "Exploring the Performance Benefits of End-to-End Path Switching," in *Proceedings of IEEE ICNP*, 2004.

[17] M. E. Crovella and M. S. Taqqu, "aest: A Tool For Estimating the Heavy Tail Index from Scaling Properties," http://www.cs.bu.edu/faculty/crovella/aest.html.

[18] S. Kirkpatrick, C. Gelatt, and M. Vecchi, "Optimization by Simulated Annealing," *Science, Number 4598, 13 May 1983*, vol. 220, 4598, pp. 671–680, 1983. [Online]. Available: citeseer.ist.psu.edu/kirkpatrick83optimization.html

[19] L. Ingber, "Simulated annealing: Practice versus Theory," Lester Ingber, Tech. Rep. 93sa, 1993, available at http://ideas.repec.org/p/lei/ingber/93sa.html.

[20] S. Nahar, S. Sahni, and E. Shragowitz, "Simulated Annealing and Combinatorial Optimization," in *DAC '86: Proceedings of the 23rd ACM/IEEE conference on Design automation*. Piscataway, NJ, USA: IEEE Press, 1986, pp. 293–299.

[21] A. Anagnostopoulos, L. Michel, P. V. Hentenryck, and Y. Vergados, "A Simulated Annealing Approach to the Traveling Tournament Problem," in *Proceedings of CPAIOR'03, 2003.*, 2003.

[22] R. Metropolis, A. Rosenbluth, A. Teller, and E. Teller, "Simulated Annealing," in *Journal of Chemical Physics, 21:1087–1092*, 1953.

[23] S. Kirkpatrick, "Optimization by Simulated Annealing: Quantitative Studies," *Journal of Statistical Physics 34:975–986*, 1984.

[24] CAIDA, "The Skitter Project," http://www.caida.org/tools/measurement/skitter/.