# Optimal Configuration for BGP Route Selection

Thomas C. Bressoud
Denison University
Granville, Ohio 43023, and
Intel Research Pittsburgh
Pittsburgh, PA 15213 USA
Email: bressoud@denison.edu

Rajeev Rastogi
and Mark A. Smith
Lucent Technologies Bell Labs
600 Mountain Ave., Murray Hill, NJ 07974 USA
Email: {rastogi,marks}@research.bell-labs.com

*Abstract*—**An Internet Service Provider must provide transit service for traffic between its customers and its providers and, at the same time, attempt to minimize network utilization and balance traffic according to the capacities of its border routers. Central to the selection of border routers for transit traffic flows is the Border Gateway Protocol (BGP) between Autonomous Systems peers, through which route advertisements for network prefixes determine the selection of border routers for each traffic flow.**

**This paper examines the problem of determining an optimal set of border routers for the advertisement of network prefixes so as to minimize the cost of traffic across a transit service provider's network while maintaining egress bandwidth constraints at the border routers. Egress bandwidth constraints are considered because there is anecdotal evidence to suggest that the peering links between ASes are often bottleneck links in the Internet, and so the optimal utilization of these links is also critical. After precisely formulating the optimization problem in accordance with the operation of BGP, we relate the problem to the** *Generalized Assignment Problem* **and develop heuristic solutions for solving it. Simulation results from an implementation show up to a 37% improvement in the utilization of the peering links when compared to hot potato routing.**

*Index Terms*—**BGP, Border Router Advertisements, Interdomain routing, Load Balancing.**

## I. Introduction

The primary responsibility of an Internet Service Provider (ISP) is to provide transit service from its set of customers to the remainder of the Internet and to bring traffic from its own upstream providers and peers destined for its customers. The interface from the ISP to the customers, upstream providers, and peers is through a set of border routers of the ISP.

This responsibility is balanced with an objective of the ISP to minimize the resources used on its network in carrying transit traffic. The ISP wishes to get traffic "on its way" toward its ultimate destination as quickly as possible.

A poorly designed selection of border routers for the flows of traffic through the ISP can result in numerous problems. On one hand, ingress and/or egress traffic from/to neighbors may exceed the capacity of the selected border routers and its links, causing the ISP to fail to meet its responsibility. On the other hand, under utilization of the potential capacity at border routers, or carrying traffic across the ISP network longer than necessary, results in inefficient use of costly resources of the ISP. However, there is anecdotal evidence to suggest that the peering links at the border routers are often bottlenecks in the Internet, so it is important that these links be utilized efficiently.

ISPs today have few tools or algorithms to help with this problem. Policies governing inter-domain routing and border router/edge link selection are arrived at manually through applying intuition, ad-hoc methods and constant tuning [1].

The objective of the work presented here is to determine an optimal selection of outgoing links and their border routers to be used for egress of transit traffic where the selection minimizes provider network utilization and balances the load of traffic flows exiting the service provider across the selected egress links by respecting capacity constraints. By respecting capacity constraints, our goal is to also optimize the utilization of these outgoing links.

### A. BGP and Inter-domain Routing

To understand the mechanisms available to control the selection of border routers used for inter-domain routing of transit traffic flows and to understand the input information available to solve the problem, we must first understand the Border Gateway Protocol (BGP) and its use in inter-domain routing [2], [3].

The area of network infrastructure under a single technical and administrative control defines the boundaries of an Autonomous System (AS). Typically, an ISP is associated with a single AS. ASes interconnect via dedicated links and public network access points, and exchange routing reachability information through *external BGP peering sessions*. BGP itself is a path vector protocol that allows import and export policies to modify the routing decision from the shortest-path default.

The unit of routability provided by BGP is the *network prefix* (or just prefix), which is an aggregation of IP addresses in a contiguous block (e.g. 10.20.30.0/24). A *route advertisement* is received from a neighbor AS over a BGP peering session and contains a prefix, an IP address of the *next-hop*, a *multi-exit discriminator* (MED) and a list of ASes along the path to the specified destination prefix. Receipt of an advertisement from a neighbor AS conveys the ability to egress data traffic toward the given prefix through that neighbor across the next-hop link. Upon receiving an advertisement, a BGP speaker must decide whether or not to use this path and, if the path is chosen, whether or not to propagate the advertisement to neighboring ASes (after adding its own AS number to the AS path). When propagating an advertisement to a neighbor AS, the MED can be used by the neighbor to differentiate the preference of the AS among a set of ingress routers in common with that neighbor.

BGP *import policy* allows an AS to favor one advertisement over another by assigning a *local preference*. The advertisement, with the assigned local preference, may then be disseminated among all the BGP speakers within the receiving AS[1]. For each router, an *acceptance* process then occurs wherein the BGP speaker selects the "best" route advertisement for each prefix. The decision criteria for the acceptance process proceeds as follows:

1) Accept the advertisement with the highest local-preference.
2) Break ties by accepting the advertisement with the shortest AS path.
3) Break ties by prefering the route with the lowest origin type, where a route originally learned from an internal protocol (IGP) is preferable to a route learned from the External Gateway Protocol (EGP), which is preferable to a route learned through injection from another routing protocol (INCOMPLETE).
4) Break ties by accepting the advertisement with the smallest MED for routes with the same next-hop AS.
5) Break ties by prefering an external BGP advertisement over an internal BGP advertisement (to facilitate egress from the AS at the earliest opportunity).
6) Break ties by accepting the advertisement with the smallest intra-domain cost (IGP metric) to the egress border router.
7) Break any remaining tie by accepting the advertisement with the smallest next-hop address.

Note that, since step 6 utilizes the IGP metric representing an intra-domain cost, two different BGP speakers (routers) within the same AS may select different best advertisements for a given prefix, where each favors the "closest" (in cost) egress border router.

The import policy and decision process described above control the selection of the border router and *edge link* used to egress traffic for any particular prefix. We refer to this combination of border router and egress edge link as the *egress point* for the prefix. On the ingress side, BGP *export policy* allows an AS to control incoming traffic as advertisements are propagated to neighbor ASes.

## B. Problem Detail

Recall that the unit of routability for BGP, and thus the association of flows of traffic from an ingress point to an egress point, is the network prefix. A *traffic flow* is defined by its ingress point (ingress router and edge link) and its destination network prefix. The problem addressed in this paper becomes: for each neighbor, ingress edge link, and prefix (i.e. traffic flow) the ISP must transit traffic for, select an egress border router/edge link (egress point) for that traffic with the objective of optimizing network utilization. This selection must be accomplished respecting egress capacity constraints of the egress points, thus balancing traffic flows between routers where such constraints would be violated.

[1]In the literature [4], External BGP is used to refer to the updates and advertisements between ASes and Internal BGP is used to refer to the dissemination within the AS. We use BGP to encompass both and, if necessary, will make explicit internal or external.

A common approach to improving network resource utilization is to egress incoming traffic as quickly as possible. This is essentially equivalent to minimizing the total distance traversed by transit traffic within the ISP network, which in some sense represents the "cost" incurred by the ISP to carry the transit traffic. In this paper, we focus on minimizing this cost. Note that while our description uses the term intra-domain cost (or IGP Metric), this is not meant to imply that our methods only apply to a hop count or other static measure. The cost metric employed could also incorporate factors of congestion or other dynamic criteria.

Note that our problem as defined allows the flexibility of the provider to select *egress* points to allow optimization of network utilization and balancing of traffic flows to respect egress capacity constraints, but does not manipulate the potential *ingress* points in a similar manner. We assume the ingress points of each traffic flow are an input to the problem, and are fixed by an outside entity. This is because, in the world of service providers, such ingress points are dictated by the customer, and where the customer wishes to ingress traffic. The most a provider can do is to "suggest," through the use of MEDs, the preference of one ingress point over another. In practice, even this mechanism is not uniformly respected, and the customer has the last word.

For our problem, this also implies that traffic from a given neighbor destined for a particular prefix may ingress from multiple points. In this work, these are treated as distinct traffic flows.

*1) Problem Variants:* We consider two variants of the problem that distinguish themselves in how egress traffic is constrained to an egress point.

In the first, simpler, variant, a single egress point is selected for each destination prefix. Thus, for all neighbors and their ingress points generating traffic destined for that prefix, traffic will egress from the same egress point. We call this problem variant Single Egress Selection (SES). In practice, this would occur as a matter of course in the cases where, for a set of advertisements for the same prefix, the acceptance process executed at all BGP speakers will select the same "best" route provided the tie-breaking procedure does not come down to step 6, the IGP Metric.

In practice, it is possible that egressing traffic flows with a common destination prefix through *multiple* egress points could improve network utilization. Such would be the case when the different ingress points and potential egress points share geographic proximity for a wide area provider. Here, the BGP acceptance process resolves identically for the first five steps and breaks ties between multiple candidate egress points for a prefix (in step 6) based on the IGP Metric to each router.

The Multiple Egress Selection (MES) variant of the problem we consider allows the egress point for a given prefix to differ for distinct traffic flows.

For both problem variants, mechanisms for controlling egress point selection through BGP are detailed in Section II.

In addition to available topological information, including the set of border routers, edge links and their capacities, the set of neighbor connections (ingress border routers and their edge links), and the cost metric between all ingress points and egress points, and the BGP provided information of the advertisements and prefixes per egress point, we assume we have available traffic information. Whether measured or estimated, this specifies the transit traffic from each ingress point to each prefix for each traffic flow. This implicitly gives us the fixed ingress points of each flow.

This work treats this egress point selection problem as an off line problem, which is already quite challenging. We observe that most of the information cited above is relatively static. For instance, in [5], the authors claim that only a very small portion of BGP route advertisement updates each day reflect network events such as router failures and leased line disconnectivity. In fact, a majority of BGP updates were found to consist entirely of pathological duplicate withdrawals. The exception is the expected traffic which prior work has shown to have a periodicity across times of day and days of the week. This could be handled in our work by solving the problem multiple times, once for each equivalence class of traffic pattern.

### *C. Results and Contributions*

The work presented in this paper is the first to address the problem of optimizing the cost of routing traffic through a provider's network while also considering load balancing based on the egress capacity of the border router links. In [6], the authors assume that the network operator of an ISP knows how he/she wants to move traffic between links, and they show how to do this in the context of the existing BGP framework. Further, by taking traffic measurements from the AT&T backbone, the authors also show that the scale of the traffic engineering problem can be considerably reduced by focusing on a small fraction of destination prefixes. In many respects, the work of [6] is complementary to ours, since unlike our work, [6] does not deal with the problem of where to egress transit traffic so as to optimize the utilization of network resources. Our work can also be viewed as a form of traffic engineering, but previous work in this area has centered mainly around intra-domain routing and the setting of weights for OSPF links in the provider network [7]. To the best of our knowledge, ours is the first attempt at performing traffic engineering using BGP policies to control inter-domain transit traffic flow. Other related work on BGP policies has focused primarily on providing guidelines to assure the stability of Internet routing [8]–[12].

Our problem formulations are specified as integer programs, and we show that the *Generalized Assignment Problem* (GAP) is a special case of our formulations. Solving GAP is well-known to be NP-hard, so we present heuristics for solving both the SES and the MES variants of our problem. We implement both the SES and MES heuristic, and perform extensive simulations on synthetic topologies of ISP networks. For the SES heuristic, our experiments show that we can get improvements of up to 17% over an intuitive heuristic based on routing the biggest traffic flows first. We also compare our MES heuristic to hot potato routing[2] (HPR). Note that if there are no capacity constraints, or if the constraints are above a certain threshold for a given volume of traffic, HPR yields a minimal cost. The benefit of the MES heuristic can most clearly be seen in the presence of non-trivial capacity constraints. It is precisely in this case that an ISP must continue to meet customer demands with limited capacity. Given a level of ingress customer traffic, we compare MES and HPR on the metric of requisite capacity of the egress links, in addition to the internal cost function. Our results show that our MES algorithm can route the same amount of traffic as HPR with up to 37% less capacity on the egress links. Furthermore, the internal cost of routing this traffic is only typically 1% to 5% more that the cost of HPR routing with higher capacity egress links. Note that our MES algorithm will always have the same internal cost as HPR if the same capacity egress links are used.

The remainder of the paper is organized as follows. In Section II we present our notation and precisely specify the problem, as well as give the linear program formulation. Section III then examines Single Egress Selection in more detail, arriving at a heuristic for its solution. The Multiple Egress Selection variant is then addressed in Section IV and a heuristic for its solution is presented. In Section V, we present our experimental setup and results for a simulated network topology across a set of capacity and traffic input sets. We conclude in Section VI.

## II. System Model

For the transit provider AS under consideration, we are given a set of neighbors $A_1, \ldots, A_q$ and a set of edge links $b_1, \ldots, b_n$ through which traffic is carried between the AS under consideration and its neighbors. Each edge link is associated with exactly one border router of the AS and with one or more neighbors. Multiple edge links may be associated with the same border router, and multiple neighbors may be connected to the AS through a given edge link. Each neighbor may be connected to the AS through multiple edge links. For each neighbor $A_h$, let $In(h)$ denote the set of edge links through which $A_h$ may ingress data traffic. For each edge link $b_j$, we have an egress capacity constraint $C_j$. The intra-domain topology provides the shortest path distance between any two edge links $b_i$ and $b_j$, which we denote $d(i, j)$.

The external BGP peering sessions at the border routers receive advertisements for network prefixes across the edge links. Let $P_1, \ldots P_m$ denote the set of prefix advertisements received across all edge links and, for each such prefix $P_k$, let

[2]Hot potato routing is based on sending incoming traffic to the closest allowable egress point.

TABLE I

NOTATION USED IN THE PAPER

| Notation | Description |
|---|---|
| $P_1, \ldots, P_m$ | Set of network prefixes for transit routing. |
| $A_1, \ldots, A_q$ | Set of AS neighbors. |
| $b_1, \ldots, b_n$ | Set of edge links. |
| $Out(k)$ | Set of egress edge links for $P_k$. |
| $In(h)$ | Set of ingress edge links from neighbor $A_h$. |
| $d(i, j)$ | Intra-domain distance between $b_i$ and $b_j$. |
| $t(h, i, k)$ | Traffic from neighbor $A_h$ through ingress edge link $b_i$ destined for prefix $P_k$. |
| $C_j$ | Egress bandwidth capacity for edge link $b_j$. |
| $\delta$ | Function that maps traffic to an egress point. |

$Out(k)$ denote the set of edge links at which an advertisement for $P_k$ has been received. We use this notation since these are the border routers and edge links (i.e. egress points) that may egress outgoing data traffic destined for $P_k$. Also, for simplicity of exposition, we assume that the set of prefixes are non-overlapping[3]. We assume that the routes learnt for all prefixes are advertised to all ingress routers and their edge links, so that the neighbor customer ASes have the option to choose which ingress points to use. Traffic from a neighbor $A_h$ ingressing through edge link $b_i$ with destination prefix $P_k$ may be measured or estimated. Let $t(h, i, k)$ denote such traffic.

Given this notation (see Table II for a summary), we are now ready to formulate the problem statement for the two problem variants discussed in the Introduction.

## A. Problem Statement

The *BGP egress selection* problem involves selecting an egress edge link for traffic from each neighboring AS to every advertised prefix such that the total cost of carrying transit traffic is minimized. We assume that each neighbor chooses independently which peering point(s) with the transit AS it will use to ingress traffic destined for a particular prefix.

**Multiple Egress Selection (MES) Problem:** Compute an assignment function $\delta : (\{1, \ldots, q\}, \{1, \ldots, n\}, \{1, \ldots, m\}) \rightarrow (\{1, \ldots, n\})$ from (neighbor, ingress edge link, prefix) triples to egress edge links such that $\sum_{h,i,k} t(h, i, k) \cdot d(i, \delta(h, i, k))$ is minimized, and $\delta$ satisfies the following constraints:

- If $\delta(h, i, k) = j$, then $j \in Out(k)$.
- Egress capacity constraints of edge links are satisfied; that is, for all $j$, $\sum_{h,i,k:\delta(h,i,k)=j} t(h, i, k) \leq C_j$.
- For a prefix $P_k$, if for some traffic flow $t(h, i, k)$, $\delta(h, i, k) = j$, then there does not exist a $h'$, $j' \in Out(k)$ and $i'$ such that $\delta(h', i', k) = j'$ and $d(i, j') < d(i, j)$.

The objective function of the MES problem requires that the computed $\delta$ minimizes the total distance traversed by transit traffic within the network, which reflects the cost of transporting transit traffic. While we use the intra-domain shortest path distance $d$ in the objective function, other distance measures like minimum number of hops could easily be substituted instead of $d$.

The final constraint, which we call the *proximity constraint*, is necessary to ensure that the choice of egress routers made by $\delta$ are enforceable in the context of the BGP selection process. In order to enable a set of edge links $S \subseteq Out(k)$ to egress traffic for $P_k$, the import policy at each edge link in $S$ assigns an equal (but high) local-preference to the advertisement for $P_k$ and manipulates the AS path so that they are equal. Thus, Step 6 of the BGP acceptance process is used to break ties, and each ingress point selects the closest edge link from $S$ to egress traffic for $P_k$. The proximity constraint ensures that $\delta$ is indeed consistent with this choice.

The *single egress selection* (SES) problem is identical to MES, except that the proximity constraint is replaced with the following constraint which forces all traffic for a prefix $P_k$ to egress through a single egress edge link.

- For all $h, h', i, i'$, $\delta(h, i, k) = \delta(h', i', k)$.

The single egress constraint for each prefix $P_k$ can be realized in BGP by setting a higher value for local-preference at the selected egress edge link for $P_k$ in the import policy.

*Example 1:* Consider the AS depiction in Figure 1(a) consisting of four border routers $b_1, \ldots, b_4$. Routers $b_1$ and $b_2$ serve as ingress routers. Routers $b_3$ and $b_4$ are egress routers with bandwidth constraints $C_3 = 75$ and $C_4 = 50$, respectively. The intra-domain distances between ingress and egress routers are as shown in Figure 1(a). Thus, $d(1, 3) = 10$ and $d(1, 4) = 50$. Two prefixes $P_1$ and $P_2$ are advertised at both egress routers, and so $Out(1) = Out(2) = \{3, 4\}$. Two AS neighbors $A_1$ and $A_2$ ingress data traffic through the ingress routers, so $In(1) = In(2) = \{1, 2\}$. Finally, the amount of traffic from the AS neighbors to the destination prefixes is given by $t(1, 1, 1) = 30$, $t(1, 1, 2) = 15$, $t(2, 2, 1) = 25$, and $t(2, 2, 2) = 30$.

Figure 1(b) depicts the optimal assignment $\delta_s$ for the single egress case. In the assignment, $\delta_s(1, 1, 1) = 3, \delta_s(1, 1, 2) = 4, \delta_s(2, 2, 1) = 3$ and $\delta_s(2, 2, 2) = 4$. Essentially, router $b_3$ egresses traffic for $P_1$ and $b_4$ egresses traffic for $P_2$. Assignment $\delta_s$ satisfies the egress capacity constraints of the routers, since the traffic egressing from $b_3$ and $b_4$ is 55 and 45, respectively. The total cost of transporting traffic is $t(1, 1, 1) \cdot d(1, 3) + t(1, 1, 2) \cdot d(1, 4) + t(2, 2, 1) \cdot d(2, 3) + t(2, 2, 2) \cdot d(2, 4) = 1850$. Observe that $b_4$ cannot be chosen to egress all 55 units of traffic for $P_1$ since this would exceed its capacity constraint of 50.

Figure 1(c) illustrates the optimal assignment $\delta_m$ for the multiple egress case. Here, as for the single egress case above, $b_3$ egresses traffic for $P_1$. However, egress traffic for $P_2$ is split between $b_3$ and $b_4$, with $b_3$ egressing traffic from $A_1$ to $P_2$ and $b_4$ egressing traffic from $A_2$ to $P_2$. The new assignment $\delta_m$ has a cost of 1250 which is lower than the cost of 1850 for $\delta_s$ presented above. In this case, the traffic from $A_1$ to $P_2$ traverses a shorter distance $d(1, 3) = 10$ in $\delta_m$ compared to $d(1, 4) = 50$ in $\delta_s$. Note that even though router $b_3$ egresses more traffic (70 units) in $\delta_m$, its capacity constraint of 75 is still not violated. Also, $\delta_m$ satisfies the proximity constraint

[3]If two prefixes $P_1$ and $P_2$ overlap, then one of $P_1$ or $P_2$ must contain the other. Thus, a set $S$ containing overlapping prefixes can be transformed into one containing no overlapping prefixes by deleting from each prefix $P_k$ in $S$, all other prefixes in $S$ that are contained in $P_k$.

since traffic for $P_2$ from $A_1$ and $A_2$ egress at routers $b_3$ and $b_4$, respectively, which are closest to the respective ingress routers for the traffic. Note that for the multiple egress case, if hot potato routing is used, $A_2$ will try to send traffic for both prefixes to router $b_4$. However, $b_4$ only has capacity 50, and so will not be able to accommodate all the traffic from $A_2$.

## *B. Integer Program Formulation*

The BGP egress selection problem can be formulated as an integer program. For each prefix $P_k$ and the traffic from a neighbor $A_h$ ingressing at edge link $b_i$ intended for that prefix, we must select an egress edge link $b_j$ such that $j \in Out(k)$. Define variable $x^j_{hik}$ to denote this selection, so $x^j_{hik} = 1$ if $b_j$ is selected as the egress edge link for traffic from $b_i$ to prefix $P_k$, and $x^j_{hik} = 0$ otherwise.

The integer program, IP(1), for the BGP egress selection problem is then formulated as follows:

$$\min \sum_k \sum_h \sum_{i \in In(h)} \sum_{j \in Out(k)} x^j_{hik} \cdot d(i,j) \cdot t(h,i,k) \qquad (1)$$

subject to the following constraints

$$\forall j: \quad \sum_{k: j \in Out(k)} \sum_h \sum_{i \in In(h)} x^j_{hik} \cdot t(h,i,k) \leq C_j \qquad (2)$$

$$\forall h,i,k: \quad \sum_{j \in Out(k)} x^j_{hik} = 1 \qquad (3)$$

The objective function of (1) is the integer programming version of the minimization objective noted for both problem variants in the problem statement. Constraint (2) enforces the egress capacity constraints in the integer program. Equation (3) is used to specify that any traffic from a particular neighbor, ingressing at a particular edge link, and destined for a particular prefix, must go through one selected egress edge link toward the destination prefix.

The formulation as presented thus far allows for multiple egress edge links for a given prefix across *different* neighbors, but does not enforce the proximity constraint. In order to further constrain the formulation, we define an additional set of integer variables $z^j_k$ such that $z^j_k = 1$ if $b_j$ is chosen as an egress point for traffic to prefix $P_k$ for one or more ingress neighbors, and $z^j_k = 0$ otherwise. Then, the following equations enforce the proximity constraint.

$$\forall h,k,i,j: x^j_{hik} \leq z^j_k \qquad (4)$$

$$\forall h,k,i,j: x^j_{hik}, z^j_k \in \{0,1\} \qquad (5)$$

$$\forall h,k,i \in In(h),$$
$$\forall j \in Out(k), j' \in Q(i,j,k): \quad z^{j'}_k + x^j_{hik} \leq 1 \qquad (6)$$

In Equation (6) above, $Q$ is a utility function that is used to specify, for a given ingress and egress edge link pair $b_i$ and $b_j$ and a given prefix $P_k$, the set of alternative egress edge links for $P_k$ that are *closer* than $b_j$. Thus $Q(i,j,k)$ is defined as the set of edge links $\{l \mid l \in Out(k) \wedge d(i,l) < d(i,j)\}$[4]. This final constraint (6), which is the proximity constraint, ensures that an $(A_h, b_i, P_k)$ triple cannot be assigned an egress edge link $b_j$ if there exists a $h'$ or $i'$ such that $(A_{h'}, b_{i'}, P_k)$ is assigned to an egress edge link $b_{j'}$, $j' \in Q(i,j,k)$. This completes the integer programming formulation for the multiple egress selection.

[4]In case $d(i,l) = d(i,j)$, then $l \in Q(i,j,k)$ if and only if the IP address of $l$ is smaller than $j$.

To change the formulation for the single egress selection case, we simply further constrain $z^j_k$. To assure that for a prefix $P_k$, only a single egress is selected across all edge links, we replace Equation (6) by the following:

$$\forall k: \quad \sum_{j \in Out(k)} z^j_k = 1 \qquad (7)$$

Unfortunately, solving the above integer program IP(1) is known to be computationally intractable. However, in Section IV, we show that solving the linear relaxation of a variant of the above integer program enables us to derive efficient solutions for the MES problem.

## III. Algorithms for Single Egress Variant

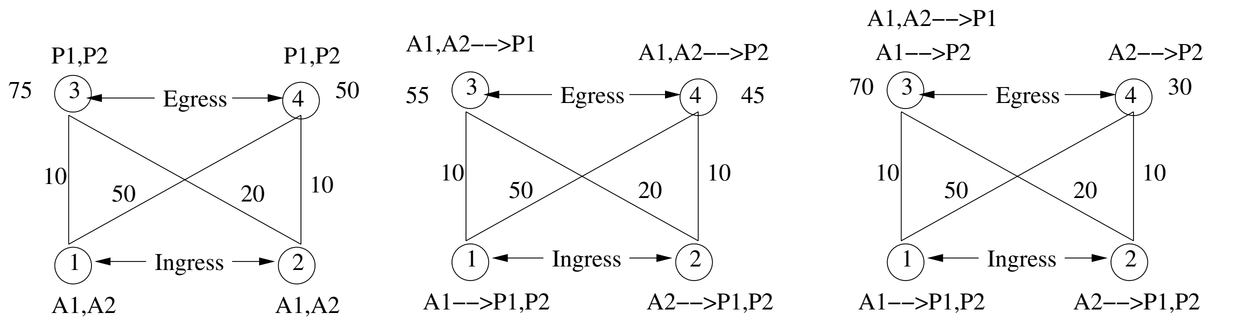
The SES problem can be shown to be a version of the *generalized assignment problem* (GAP) which is known to be NP-hard and has been well-studied in the operations research and theoretical computer science communities. See for example, [13] and [14]. The definition of GAP is as follows.

**Generalized Assignment Problem:** Given $\xi$ jobs and $\phi$ machines, a processing time $p_{rs}$ and cost $c_{rs}$ for processing job $r$ on machine $s$, and a total processing time $T_s$ available for each machine $s$, compute an assignment $f: \{1,\ldots,\xi\} \rightarrow \{1,\ldots,\phi\}$ of jobs to machines such that

- the total cost of processing jobs is minimized, that is, $\sum_r c_{rf(r)}$ is minimum, and
- the processing time for jobs on each machine $s$ does not exceed $T_s$, that is, $\sum_{r:f(r)=s} p_{rs} \leq T_s$.

We map the SES problem to GAP by considering each egress edge link $b_j$ to be a machine and each prefix $P_k$ to be a job with processing time and cost $\sum_h \sum_{i \in In(h)} t(h,i,k)$ and $\sum_h \sum_{i \in In(h)} d(i,j) \cdot t(h,i,k)$, respectively, on machine $j$. The constraint on the processing time available on each machine $b_j$ is the capacity constraint, $C_j$, of the egress edge link. Note that this is not the most general version of GAP as the processing times of the jobs do not vary with the machines (egress edge links).

GAP is not only intractable, but also very difficult to solve approximately. Even ignoring costs (e.g., setting all job costs to 0), it is intractable to compute an assignment of jobs to machines such that the total processing time constraints of machines is not violated [15]. The version described above where the processing time of a job does not vary with the machines is also intractable [16]. Thus, the only option available is to rely on heuristics to solve GAP.



(a) AS Ingress/Egress Routers (b) Optimal Assignment (Single Egress) (c) Optimal Assignment (Multiple Egress)

Fig. 1. Example of optimal assignments for single and multiple egress cases

### *A. Generalized Assignment Problem Heuristic*

As mentioned above, there is a lot of previous work on GAP and good polynomial-time approximation algorithms for GAP that relax machine processing time constraints exist in the literature. Let $C$ be the cost of the optimal solution to GAP that does not violate any capacity constraints. Then an $(\alpha, \beta)$ approximation algorithm for GAP is one that gives a solution with cost at most $\alpha C$ and with capacity constraints violated by at most a factor of $\beta$. The best known result for GAP is by Shmoys and Tardos in [14] where a $(1, 2)$ approximation algorithm is given. Lenstra, Shmoys, and Tardos [15] have also shown that it is NP-hard to obtain a $(1, \beta)$ approximation algorithm for GAP for $\beta < 3/2$.

The algorithm of Shmoys and Tardos in [14] is based on first solving the LP relaxation of the integer programming formulation for GAP, and then rounding the fractional solution to a nearby integer solution. The total cost of the assignment computed by the Shmoys and Tardos algorithm is optimal, but, as noted above, the processing times of jobs assigned to a machine may exceed the machine's total processing capacity by at most a factor of 2. In Figure 2, we present a greedy heuristic that uses the assignment $f$ computed by the Shmoys Tardos algorithm as the basis to compute a new assignment $f'$ that satisfies processing time constraints and still has a low cost.

In Procedure GREEDY, jobs on machines whose processing times constraints are violated are re-assigned to other machines with sufficient capacity to process the jobs. This process of rescheduling jobs from violated machines to non-violated machines is continued until either no violated machines remain or no more jobs on violated machines can be re-assigned. The critical issue is which jobs on violated machines should be chosen for migration and which machines should they be migrated to. We adopt a greedy approach, choosing during each iteration, the job $r$ and machine $t$ for which transferring $r$ to $t$ results in the smallest increase in cost per unit decrease in the violation amount. Note that in Step 11, $c_{rt} - c_{rs}$ is the increase in cost associated with re-assigning $r$ from $s$ to $t$, while $\min\{p_{rs}, U_s - T_s\}$ is the decrease in the violation.

```
procedure GREEDY($c$[], $p$[], $T$[])
1.  Use Shmoys Tardos Algorithm to compute assignment
    $f$ of jobs to machines for costs and processing times
    $c$[], $p$[] and $T$[]
2.  foreach machine $s$, let $U_s := \sum_{s:f(r)=s} p_{rs}$
3.  Let VSet := $\{s : U_s > T_s\}$
4.  job := 1
5.  while job $> 0$ {
6.     job := 0
7.     min_cost := $\infty$
8.     foreach job $r$ such that $f(r) = s$ and $s \in$ VSet
9.        foreach machine $t$ not in VSet
10.          if $T_t - U_t > p_{rt}${
11.             cost := $\frac{c_{rt} - c_{rs}}{\min\{p_{rs}, U_s - T_s\}}$
12.             if cost $<$ min_cost{
13.                min_cost := cost
14.                job := $r$
15.                old := $s$
16.                new := $t$
17.             }
18.          }
19.    if job $> 0$ {
20.       $f$(job) := new
21.       $U_{\text{old}} := U_{\text{old}} - p_{\text{job,old}}$
22.       $U_{\text{new}} := U_{\text{new}} + p_{\text{job,new}}$
23.       if $U_{\text{old}} \leq T_{\text{old}}$ then VSet := VSet - {old}
24.    }
25. }
26. return $f$
```

Fig. 2. Greedy Heuristic for GAP

The time complexity of Procedure GREEDY is dominated by the running time of the Shmoys Tardos Algorithm in Step 1. The Shmoys Tardos algorithm rounds the fractional solution of the LP for GAP by finding a minimum-cost integer matching of a bipartite graph

containing $\xi$ job nodes on one side, at most $\xi + \phi$ machine nodes on the other, and no more than $2\xi\phi$ edges between the two sides. We solve the minimum cost integer matching problem using *successive shortest path algorithm* [17] for the minimum cost flow problem. This solution requires running the shortest path algorithm $\xi$ times, with each run taking $O((\xi\phi) \cdot \log(\xi + \phi))$ time. Thus, the overall time complexity is $O((\xi^2\phi) \cdot \log(\xi + \phi))$.

Note that $\xi$ corresponds to the number of prefixes in the mapping from SES to GAP. Even though an ISP may potentially need to deal with tens of thousands of routes, [6] points out that only a small fraction of prefixes are responsible for a large fraction of the traffic. Thus, by restricting ourselves to only these significant prefixes, we can ensure that $\xi$ will not be too big.

## IV. Algorithms for Multiple Egress Variant

For the multiple egress version of the problem, traffic destined for a given prefix can exit through multiple egress edge links, so we cannot simply use the approach of solving an instance of GAP with each prefix as a job for this problem variant. Instead, in this section, we propose a new heuristic that computes multiple egress edge links for each prefix such that for traffic flow $t(h,i,k)$, traffic leaves through the egress edge link for $P_k$ that is closest to $b_i$ without violating egress edge link bandwidth constraints.

### A. Integer Program Formulation

In the multiple egress case, traffic flows, $t(h,i,k)$ and $t(h',i',k)$ may be assigned to different egress edge links even though they are destined for the same prefix. This is a significant departure from the single egress case where egress edge links for all traffic flows were identical for a given prefix. The fact that only a single egress edge link was to be chosen per prefix allowed us to define a single job per prefix. However, for the multiple egress case, we need to define a separate job $(h,i,k)$ for each traffic flow $t(h,i,k)$. A simple approach would be to simply solve GAP to compute egress edge links, where the cost and processing time of job $(h,i,k)$ on machine $j$ (corresponding to egress edge link $b_j$) is as follows:

- $c_{(h,i,k),j}$: $t(h,i,k)\cdot d(i,j)$ if $j \in Out(k)$; $\infty$ otherwise.
- $p_{(h,i,k),j}$: $t(h,i,k)$ if $j \in Out(k)$; $\infty$ otherwise.
- $T_j : C_j$.

The problem with using GAP as described above for egress edge link computation is that it does not incorporate the additional *proximity* constraint which states that $t(h,i,k)$ cannot be assigned an egress edge link $b_j$ if there exists a $h'$ and $i'$ such that $t(h',i',k)$ is assigned to an egress edge link $b_{j'}$, $j' \in Q(i,j,k)$. Recall from Section II that $Q(i,j,k)$ denotes the set of egress routers in $Out(k)$ that are closer to $b_i$ than $b_j$. Fortunately, the proximity constraint can be captured in an integer program, whose linear relaxation can subsequently be solved and rounded (using the Shmoys Tardos technique for GAP from [14]) to yield a better solution than simply solving GAP to compute the egress edge links. The integer program formulation in Section II-B is one way to formulate the problem. Below, we present a slightly different formulation, which does not require the use of the variable $z_k^j$. This formulation is mostly the same as the previous integer program except that Constraints (4) through (6) are replaced by Constraints (11) and (12). We call this formulation IP(2).

$$\min \sum_k \sum_h \sum_{i \in In(h)} \sum_{j \in Out(k)} x_{hik}^j \cdot d(i,j) \cdot t(h,i,k) \qquad (8)$$

subject to the constraints:

$$\forall j : \quad \sum_{k: j \in Out(k)} \sum_h \sum_{i \in In(h)} x_{hik}^j \cdot t(h,i,k) \le C_j \qquad (9)$$

$$\forall h,i,k : \quad \sum_{j \in Out(k)} x_{hik}^j = 1 \qquad (10)$$

$$\forall h,i,j,k : \quad x_{hik}^j \in \{0,1\} \qquad (11)$$

$$\forall h,h' \quad \forall i \in In(h), i' \in In(h') \wedge \forall k \quad \forall l \in Out(k) :$$

$$\sum_{j \in Out(k) \setminus Q(i,l,k)} x_{hik}^j + \sum_{j' \in Q(i,l,k)} x_{h'i'k}^{j'} \le 1 \qquad (12)$$

Without Constraint (12), the remaining constraints essentially reduce to GAP, with costs and processing times for jobs as described for the GAP-based approach earlier in this subsection. Constraint (12) captures the proximity constraint by ensuring that if for some $h,i,k$ and $l \in Out(k)$, the egress edge link for $t(h,i,k)$ is not selected from $Q(i,l,k)$ (that is, the egress edge link for $t(h,i,k)$ is chosen from $Out(k) \setminus Q(i,l,k)$), then for all $h', i'$, the egress edge link for $t(h',i',k)$ cannot be chosen from $Q(i,l,k)$. Note that the optimal fractional solution to the linear relaxation of the above integer program is a feasible solution to the LP without Constraint (12), which is essentially GAP. Thus, the LP rounding technique of Shmoys and Tardos can be used to compute an assignment of traffic flows to egress routers from the optimal fractional solution to the LP relaxation of IP(2), since this fractional solution is a feasible solution to GAP. Note that since this optimal solution to the relaxed LP satisfies the proximity constraints, we expect the computed assignment to violate fewer proximity constraints than an assignment computed from simply solving GAP without the proximity constraint.

### B. Heuristic for MES

The assignment of traffic flows, $t(h,i,k)$, to egress edge links obtained as a result of rounding the optimal fractional solution for the LP relaxation of IP(2) has two basic problems: (1) The capacity constraints of egress edge links may be violated (by at most a factor of 2), and (2) the proximity constraints may be violated. Procedure MultipleEgress in Figure 3 attempts to remedy this by using heuristics to re-assign traffic flows to egress edge links such that both capacity as well as proximity constraints are met. It computes in the function $\delta$ these new assignments.

As discussed earlier, the assignment $f$ computed in Step 2 of the procedure may not meet capacity and proximity constraints. Suppose that for each prefix $P_k$, $f_p(k)$ is the set of all egress edge links for prefix $P_k$ (Step 3). Then, there is a unique assignment $\delta$ that maps each $t(h,i,k)$ to an egress edge link, and that satisfies the following two properties: (1) $\delta$ satisfies proximity constraints, and (2) for all $h,i,k$, $\delta(h,i,k) \in f_p(k)$. To see this, suppose for a traffic flow $t(h,i,k)$, $j$ is the egress edge link in $f_p(k)$ closest to $b_i$ (the ingress edge link for the traffic). Then $\delta(h,i,k) = j$ satisfies the above two properties. The function compute_egress in Steps 8, 15 and 31 of the procedure returns such a $\delta$. Thus, the assignment $\delta$ computed in Step 8 satisfies proximity constraints, but may violate egress edge link capacity constraints.

Procedure MULTIPLEEGRESS iteratively applies one of two basic transformations to $\delta$ in order to reduce the total violation amount. The first is to delete a violated egress edge link $j$ from $f_p(k)$ for a prefix $P_k$. This has the effect of diverting all the egress traffic for $P_k$ passing through $j$ to other edge links, thus decreasing the degree to which $j$ is violated. Note, however, that the violation amount of other routers carrying the re-directed traffic from $j$ could increase. The second primitive transformation is to add an egress edge link $j$ that satisfies capacity constraints to $f_p(k)$ for a prefix $P_k$. This has the potential to reduce the violation amount by assigning to $j$, egress traffic for $P_k$ passing through other violated egress edge links. It is straightforward to observe that addition of router $j$ to $f_p(k)$ can cause the violation amount for only $j$ (and no other edge link) to increase. Procedure MULTIPLEEGRESS repeatedly applies one of the two transformations to $\delta$ until no capacity constraints are violated or there is no remaining transformation for reducing the violation amount.

In Procedure MULTIPLEEGRESS, $V^\delta(j)$ is the amount by which capacity of egress edge link $j$ is violated; that is, $V^\delta(j) = \max\{0, \sum_{h,i,k:\delta(h,i,k)=j} t(h,i,k) - T_j\}$; $C^\delta = \sum_h \sum_k \sum_{i:i\in In(h)} \sum_{j:\delta(h,i,k)=j} t(h,i,k)\cdot d(i,j)$ (the cost of transporting all the traffic to prefixes based on the egress edge links determined by $\delta$) and $\mathsf{VSet}^\delta = \{j : V^\delta(j) > 0\}$ (the set of egress edge links whose capacity constraints are violated). In each iteration of the while loop (Step 5), the procedure chooses the prefix $P_k$ and egress edge link $j$ for which the increase in cost per unit decrease in violation amount is minimum; that is, if $\delta'$ is the new assignment (that satisfies proximity constraints) after deleting or inserting $j$ from $f_p(k)$, then $\frac{C^{\delta'} - C^\delta}{\sum_l V_l^\delta - \sum_l V_l^{\delta'}}$ is minimum. An egress edge link $j$ is a candidate for addition/deletion from $f_p(k)$ only if (1) the operation results in a decrease in the overall amount of violation of the capacity constraints of egress edge links, and (2) the operation does not cause an egress edge link that previously satisfied capacity constraints to now violate them. Finally, for a prefix $P_k$, only for egress edge links $j$ that violate capacity constraints are candidates for deletion, while for insertion only edge links that satisfy capacity constraints are candidates.

The time complexity of Procedure MULTIPLEEGRESS is dominated by the running time of the Shmoys Tardos Algorithm in Step 2. Since there are at most $q \cdot n \cdot m$ flows and $n$ egress edge links, the worst-case time complexity of the procedure is $O((q\cdot n\cdot m)^2 \log(q\cdot n\cdot m))$.

```
procedure MULTIPLEEGRESS($c[], p[], T[]$)
1.  Solve linear relaxation of IP(2) for optimal solution $x$
2.  Compute assignment $f$ of flows $t(h,i,k)$ to egress
    edge $b_j$ by applying the Shmoys Tardos algorithm to $x$
3.  foreach $k$, $f_p(k) := \{f(h,i,k)\}$
4.  pref := 1
5.  while pref $> 0$ {
6.    pref := 0
7.    min_cost := $\infty$
8.    $\delta$ := compute_egress($p[], f_p$)
9.    foreach prefix $k${
10.     foreach egress edge link $j$ in $f_p(k)$ {
11.       if $V^\delta(j) > 0$ {
12.         $f'_p := f_p$
13.         $f'_p(k) := f'_p(k) - \{j\}$
14.         $\delta'$ := compute_egress($p[], f'_p$)
15.         if VSet$^{\delta'} \subseteq$ VSet$^\delta$ and
                 $\sum_l V^\delta(l) - \sum_l V^{\delta'}(l) > 0$ {
16.           cost := $\frac{C^{\delta'} - C^\delta}{\sum_l V^\delta(l) - \sum_l V^{\delta'}(l)}$
17.           if cost < min_cost {
18.             min_cost := cost
19.             pref := $k$
20.             egress := $j$
21.             action := delete
22.           }
23.         }
24.       }
25.     }
26.     foreach egress edge link $j \in Out(k) - f_p(k)$ {
27.       if $V^\delta(j) \le 0$ {
28.         $f'_p := f_p$
29.         $f'_p(k) := f'_p(k) \cup \{j\}$
30.         $\delta'$ := compute_egress($p[], f'_p$)
31.         if $V\delta'(j) \le 0$ and
                 $\sum_l V^\delta(l) - \sum_l V^{\delta'}(l) > 0$ {
32.           cost := $\frac{C^{\delta'} - C^\delta}{\sum_l V^\delta(l) - \sum_l V^{\delta'}(l)}$
33.           if cost < min_cost{
34.             min_cost := cost
35.             pref := $k$
36.             egress := $j$
37.             action := insert
38.           }
39.         }
40.       }
41.     }
42.   }
43.   if pref $> 0$ and action = delete
44.     $f_p$(pref) := $f_p$(pref) $- \{$egress$\}$
45.   else if pref $> 0$ and action = insert
46.     $f_p$(pref) := $f_p$(pref) $\cup \{$egress$\}$
47. }
48. return $\delta$
```

Fig. 3. Heuristic for computing egress edge links for MES.

## V. EXPERIMENTS

Through experiments, we validate our heuristics for the BGP router selection problem. The experimental approach employed is one of generating synthetic topologies of the ISP and executing our heuristics against those topologies. We generate input problem instances by first generating a simulated border router topology and prefix advertisement distribution. For a specified number of border routers, the set of neighbors is generated by randomly selecting a multi-homing degree from a uniform distribution and assigning that many border routers to the current neighbor and repeating. For simplicity, we assume that all border routers of a given neighbor $A$ may ingress traffic from $A$, so $In(A)$ is immediately available as well. The intra-domain distances are drawn from a specified uniform distribution and we assume that $d(i, j) = d(j, i)$.

Given the border router and neighbor topology, we then generate a set of (egress) advertisements and their association to neighbors. We assume that, if a prefix $k$ is advertised at some border router of neighbor $A$, it is advertised at all border routers of $A$. A random subset of neighbors is chosen to egress each prefix.

We generate a synthetic set of traffic from neighbors to prefixes by first profiling the traffic load for each prefix. Given the traffic flow for a prefix, we then generate the neighbor specific traffic by drawing from a uniform random sampling around that profile. This method of traffic generation is based on the assumption that traffic for a particular destination prefix will be similar (but not identical) from different ingress neighbors. For each traffic instance we randomly choose an ingress point from $In(A)$ for the neighbor $A$ that originates the traffic.

In our experiments, we study the performance of our proposed heuristics for the SES and MES problems as the number of prefixes is varied between 100 and 1000. While a typical default-free routing table may contain routes for more than 90,000 prefixes, only a small fraction of prefixes are responsible for a large fraction of the traffic [6]. Feamster et al [6] also suggest other ways, such as grouping prefixes, by which the scale of the problem can be reduced. Thus, we expect our heuristics to be employed on a few thousand prefixes, and not tens of thousands. We believe that our implementations of the heuristics, with further fine tuning, can be deployed in practice to handle thousands of prefixes in a real world ISP.

## A. SES Experiments

For the SES variation of the problem we want to show the value of our heuristic in addressing the following questions:

1) Given a network topology, with border routers, capacities, distances, and neighbor associations, and with a problem instance of traffic and a set of network prefix advertisements, does a feasible solution to the selection problem exist? In the context of our solution, this question means both that a solution to the linear programs at each step exists, and that it is possible to move jobs (via GREEDY) so that any constraint violations are resolved.
2) If a solution exists, how good is the solution provided by our heuristic?

In answering the second question, we must have a metric against which we can compare our own solution. We propose two such metrics. The first metric is the objective function cost for the case of infinite capacity at all border routers. The second metric is the objective function cost for an alternative heuristic that one imagines might be used by an ISP. The alternative heuristic works as follows. The set of $(h, i, k)$ triples are sorted in decreasing order of traffic. An attempt is then made to assign egress border routers for each $(h, i, k)$. If $k$ has already been assigned to an egress point, we simply send the traffic to the already selected egress point, if capacity

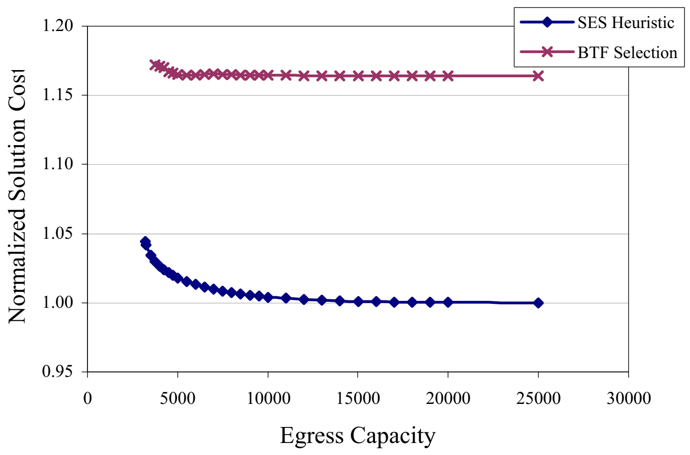


Fig. 4. Results for the SES experiments.

permits. If $k$ has not been assigned, we find the closest egress point that has the capacity to accept that traffic. We call this heuristic *Biggest Traffic First (BTF) Selection*.

The generated topology, advertisement distribution, and traffic define a *scenario* and in this section, in the interest of space, we present the results for one scenario for the Single Egress Selection heuristic. For a given scenario, we can generate either a random set of capacities, or fixed egress capacities, and then execute our heuristics. We take the simplifying approach of fixing all ingress and egress capacities to the same value and then varying that value for a scenario's traffic load to compare the cost (from the objective function) for each run at a given capacity.

We set the number of border routers to 100 and the number of prefixes to 1000. Here, neighbors peer with the ISP through 1 to 3 border routers. The AS has 48 neighbors. Distances are between 10 and 100 and are unitless. The traffic is between 0 and 20 (and is again unitless) from any neighbor to a particular prefix.

Figure 4 presents our results for this scenario. The graph shows the solution cost, normalized by the infinite capacity solution, as a function of increasing capacity. Here, the LP solver could not find a feasible solution for capacities below 3076, and constraint violations could not be resolved until capacity 3155. The solution cost at 3200 was 4% above the infinite capacity solution and converged by capacity 24150.

In this scenario, BTF Selection was unable to find solutions until capacity 3750 (compared with 3200 for SES) and, at this capacity, was 17% above the infinite capacity solution and was about 14% worse than the SES heuristic solution. By capacity 13,000, BTF had converged, and could do no better than 17% worse than the infinite capacity solution.

## B. MES Experiments

For the MES variant of the problem, we compare our MULTIPLEEGRESS heuristic to hot potato routing (HPR), which sends all incoming traffic to the closest allowable egress point. HPR is what ISPs typically use to route traffic. As mentioned before, if there are no constraints on egress bandwidths or if the capacities at egress routers are above a certain threshold for a given volume of traffic, HPR will always give the optimal cost. For the minimum capacity constraint required to route all the traffic with HPR, MULTIPLEEGRESS will also route at optimal cost. However, MULTIPLEEGRESS may be able to route a given volume of traffic (without violating egress capacity constraints) at a lower egress router threshold capacity than HPR, although the routing cost may be sub-optimal.

Since the egress link capacities are also an important resource, we look at the following two issues in these experiments:

1) Given a certain volume of traffic, what is the threshold capacity on the egress edge links required to route all of it using MULTIPLEEGRESS and HPR?
2) At this threshold capacity what is the internal cost of routing this traffic?

For these experiments we present three different scenarios. For each scenario we varied the volume of traffic by changing the traffic distribution range. We started with a range of 0 to 20 for each scenario, and then for each run increased the upper range to get a higher volume of traffic. As in the SES case, we used fixed capacities for the egress routers. For a given topology and traffic profile, we adjusted the fixed capacity of the egress routers until we found the minimum capacity that allowed the routing of all the traffic. In order to find this minimum we had to do several runs for a given volume of traffic. Furthermore, we had to wait until a run was finished in order to see whether to adjust the capacity up or down. For the SES experiments, this type of hand-tuning was not necessary, so here we use a smaller data set. Note that in practice the operator of an ISP would just run the heuristic for a given traffic matrix and the topology of his/her AS. The many runs we do to validate our algorithms versus HPR would not be necessary.

The results for the MES experiments are shown in Tables II – IV. In the tables "Traffic" represents the units of traffic that is routed; "MES cap." is the minimum border router capacity required to route the traffic (without violating capacity constraints) using the MULTIPLEEGRESS heuristic, "HPR cap." is the minimum border router capacity required if hot potato routing is used; "MES/HPR cap." gives the normalized minimum capacity of the egress links in the MULTIPLEEGRESS heuristic with respect to HPR; and "MES cost" gives the normalized internal cost of the MULTIPLEEGRESS heuristic with the minimum capacities with respect to the internal cost of HPR.

The scenario for Table II consists of 25 border routers and 35 prefixes. Here 12 neighbors peer with the ISP through one to four border routers. Distances between border routers range from 10 to 100. For this scenario MULTIPLEEGRESS requires between 31% and 35% less capacity to route the same volume of traffic as HPR with an additional cost ranging from 2% to 3%.

Table III presents a scenario consisting 25 border routers and 75 prefixes. Again the ISP has 12 neighbors and each peers with the AS through one to four border routers. Distances between border routers range between 10 and 100. Here the MULTIPLEEGRESS heuristic can route the same volume of traffic with between 30% and 28% less capacity on the border routers, but with an increase in cost of less than 2%.

The scenario for Table IV consists of 30 border routers and 110 prefixes. The ISP has 17 neighbors and each peers with the AS through one to three border routers. Distances between border routers is the same as for the other two scenarios. Of the three scenarios presented, MULTIPLEEGRESS shows the most improvement in the bandwidth capacity utilization of the border router links for this one. The improvement ranges from over 37% to 36%. However, for this scenario the increase in internal cost is higher, ranging between 3% and 5%.

TABLE II
RESULTS: MES EXPERIMENTS FOR TOPOLOGY 1

| Traffic | MES cap. | HPR cap. | MES/HPR cap. | MES cost |
|---|---|---|---|---|
| 2871 | 197 | 299 | .689 | 1.021 |
| 3651 | 251 | 381 | .659 | 1.020 |
| 4404 | 299 | 458 | .653 | 1.034 |
| 6631 | 453 | 697 | .650 | 1.033 |
| 8139 | 561 | 854 | .657 | 1.020 |
| 10389 | 717 | 1090 | .658 | 1.022 |
| 14912 | 1011 | 1563 | .647 | 1.027 |
| 22406 | 1517 | 2347 | .647 | 1.027 |

TABLE III
RESULTS: MES EXPERIMENTS FOR TOPOLOGY 2

| Traffic | MES cap. | HPR cap. | MES/HPR cap. | MES cost |
|---|---|---|---|---|
| 6280 | 455 | 632 | .720 | 1.014 |
| 9580 | 683 | 963 | .709 | 1.016 |
| 12843 | 912 | 1288 | .708 | 1.016 |
| 32528 | 2293 | 3268 | .702 | 1.018 |
| 65292 | 4589 | 6552 | .700 | 1.018 |
| 163627 | 11475 | 16403 | .700 | 1.018 |
| 327567 | 22942 | 32844 | .699 | 1.018 |
| 655382 | 45811 | 65700 | .697 | 1.017 |

TABLE IV
RESULTS: MES EXPERIMENTS FOR TOPOLOGY 3

| Traffic | MES cap. | HPR cap. | MES/HPR cap. | MES cost |
|---|---|---|---|---|
| 12938 | 524 | 842 | .622 | 1.045 |
| 19715 | 809 | 1293 | .626 | 1.038 |
| 66877 | 2777 | 4398 | .631 | 1.032 |
| 134319 | 5596 | 8832 | .634 | 1.036 |
| 201857 | 8847 | 13253 | .637 | 1.035 |
| 269429 | 11222 | 17694 | .634 | 1.035 |
| 336837 | 14025 | 22122 | .634 | 1.035 |
| 674261 | 28089 | 44288 | .634 | 1.036 |

## VI. CONCLUSION

In this paper, we precisely define and formulate the problem of BGP router selection, including the two problem variants of Single Egress Selection and Multiple Egress Selection. We show the correspondence of the problem to the operation of BGP and the objectives of an Internet Service Provider. After arguing the difficulty of the problem, we develop and propose heuristic solutions for both problem variants and show the results of an implementation embodying those heuristics on simulated network topologies and traffic for a varying set of border router capacities. We conclude that the problem formulation is sound and that our heuristic solutions show marked improvement over alternatives. Both the formulation and the solutions should be of great value to providers as they migrate away from ad-hoc methods of configuration and look to optimize their network utilization and balance traffic across the capacities of their border routers.

## References

[1] G. Huston, *ISP Survival Guide Strategies for Running a Competitive ISP*. John Wiley and Sons, 1999.

[2] Y. Rekhter and T. Li, “A border gateway protocol 4,” Internet-Draft (RFC1771), February 1998.

[3] J. W. S. III, *BGP4: Inter-Domain Routing in the Internet*. Addison-Wesley, 1999.

[4] B. Halabi, *Internet Routing Architectures*. Cisco Press, 1997.

[5] C. Labovitz, G. R. Malan, and F. Jahanian, “Internet routing instability,” in *Proceedings of ACM SIGCOMM*, 1997.

[6] N. Feamster, J. Borkenhagen, and J. Rexford, “Controlling the impact of BGP policy changes on ip traffic,” AT&T Labs - Research, Tech. Rep. HA173000-011106-02TM, 2001.

[7] B. Fortz and M. Thorup, “Internet traffic engineering by optimizing OSPF weights,” in *Proceedings of IEEE INFOCOM*, 2000, pp. 519–528.

[8] L. Gao and J. Rexford, “Stable internet routing without global coordination,” in *Proceedings of ACM SIGMETRICS*, June 2000.

[9] R. Govindan and A. Reddy, “An analysis of internet inter-domain topology and route stability,” in *Proceedings of INFOCOM’97*, April 1997.

[10] T. G. Griffin, F. B. Shepherd, and G. Wilfong, “Policy disputes in path vector protocols,” in *Proceedings of the 7th International Conference on Network Protocols (INCP’99)*, Toronto, Canada, November - December 1999.

[11] T. G. Griffin and G. Wilfong, “An analysis of bgp convergence properties,” in *Proceedings of SIGCOMM’99*, Cambridge, Massachusetts, August 1999.

[12] ——, “A safe path vector protocol,” in *Proceedings of INFOCOM’00*, Tel Aviv, Israel, March 2000.

[13] J.-H. Lin and J. S. Vitter, “$\epsilon$-approximations with minimum packing constraint violation,” in *Proceedings of the 24th Annual ACM Symposium on the Theory of Computation*, Victoria, Canada, May 1992, pp. 771–782.

[14] D. B. Shmoys and E. Tardos, “An approximation algorithm for the generalized assignment problem,” *Mathematical Programming A*, vol. 62, pp. 461–474, 1993.

[15] J. K. Lenstra, D. B. Shmoys, and E. Tardos, “Approximation algorithms for scheduling unrelated parallel machines,” *Mathematical Programming A*, vol. 46, pp. 259–271, 1990.

[16] C. Chekuri and S. Khanna, “A PTAS for the multiple knapsack problem,” in *In Proceedings of the Twelfth Annual ACM-SIAM Syposium on Discrete Algorithms (SODA)*, January 2000.

[17] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows*. Prentice Hall, 1993.