# On Configuring BGP Route Reflectors

**Yuri Breitbart** *Kent State University*
**Minos Garofalakis** *Intel Research Berkeley*
**Anupam Gupta** *Carnegie Mellon University*
**Amit Kumar** *IIT New Delhi*
**Rajeev Rastogi** *Bell Labs India*

*Abstract*— **The *Border Gateway Protocol (BGP)* is the standard protocol for exchanging routing information between border routers of Autonomous Systems (ASes) in today's Internet. Within an AS, border routers exchange externally-learned BGP route advertisements via Internal-BGP (I-BGP) peerings. Naive solutions for these I-BGP peering sessions (e.g., based on full-mesh topologies) simply cannot scale to the sizes of modern AS networks. Carefully designed *route-reflector configurations* can drastically reduce the total number and connection cost of the required I-BGP sessions. Nevertheless, no principled algorithmic approaches exist for designing such configurations, and current practice relies on manual reflector selection using simple, ad-hoc rules. In this paper, we address the novel and challenging optimization problems involved in designing effective BGP route-reflector configurations for AS networks. More specifically, we consider the problems of selecting route reflectors in an AS topology to minimize: (1) the total connection cost of all I-BGP peering sessions, and (2) the average distance traversed by route advertisements within the AS. We present $\mathcal{NP}$-hardness results that establish the intractability of these problems, and propose several polynomial-time approximation algorithms (based on LP-rounding and combinatorial techniques) with guaranteed (constant-factor or logarithmic) bounds on the quality of the approximate solution. Our simulation results validate our approach, demonstrating the effectiveness of our configuration algorithms over a wide range of network topologies.**

## I. Introduction

The *Border Gateway Protocol (BGP)* [1], [2] has become the de facto inter-domain routing protocol for exchanging reachability information between Autonomous Systems (ASes) in today's Internet. BGP is a path-vector protocol that constructs inter-AS paths by successively propagating advertisements of *address prefixes* (i.e., aggregated IP addresses) between AS border routers that are configured as *BGP peers* [1]. Upon receiving an advertised route, a BGP-configured router determines whether it defines the "best" path (based on local routing policies [3]) to the corresponding prefix. If this is indeed the case, the router installs the path and address prefix in its routing table; further, if the route was imported from an external AS, the router also needs to advertise the route to all border routers in its own AS through *Internal-BGP (I-BGP)* peering sessions.

Before the recent explosion in the size of Internet Service Provider (ISP) networks, advertisements of externally-learned routes were implemented using a full mesh of I-BGP peerings amongst all border routers in the AS. Thus, each BGP speaker had to propagate external advertisements to *all* other border routers in its AS via distinct I-BGP peering sessions (i.e., TCP connections). Under such a brute-force solution, it is easy to see that, as the number $n$ of border routers in the AS grows, the total number of required I-BGP peerings (i.e., $n(n-1)/2$) quickly gets out of hand [4]. Maintaining such large numbers of I-BGP sessions is simply infeasible given the hardware capabilities of contemporary router architectures. Furthermore, the total "length" of the required TCP connections (and, as a consequence, the intra-AS network distance traversed by BGP route advertisements) also explodes quadratically with the number of border routers in the AS. This can result, for example, in wasting significant amounts of network link bandwidth over I-BGP exchanges. Obviously, using a full mesh of I-BGP peerings severely limits the scalability of AS networks and makes for extremely inefficient use of precious AS resources.

*Route reflectors* [1], [4], [5] tackle this scaling problem by dividing the I-BGP topology into clusters of border routers and defining a *route reflector* server for each cluster. Briefly, a route reflector is a router in the AS network that takes on the responsibility of propagating BGP route advertisements to and from the *client* border routers in its cluster. Route reflectors are connected over a full mesh of I-BGP peerings and also have peering sessions with the clients in their cluster. Upon learning a new external BGP route, a client router forwards the advertisement to the route reflector for its cluster, which, in turn, can propagate the advertisement to all other clients in its cluster as well as all other route reflectors in the AS. When receiving an advertisement from a route-reflector peer, a reflector can forward the advertisement

Work done while all the authors were with Bell Labs, Lucent Technologies, 600 Mountain Avenue, Murray Hill, NJ 07974.

*only* to clients in its cluster (and not to other reflectors in the I-BGP mesh). Essentially, one can think of a route reflector as a proxy agent that disseminates routes between its reflector peers on one side and its clients on the other (in both directions) [4].

An important advantage of route reflector configurations is that client routers behave in the exact same way that is required in the original BGP specification and, thus, can be completely oblivious to the fact that some of the route updates that they receive are reflected. Thus, deploying route reflectors requires a software upgrade only on the AS routers that are designated as reflectors. This is in contrast to other I-BGP scaling techniques, like *AS confederations*, that typically require a fork-lift software upgrade of all the routers in the AS [1], [4].

Using route reflectors can result in dramatic reductions in the number of required I-BGP peering sessions and the total length of TCP connections for I-BGP peerings. For instance, assuming a total of $n$ client (border) routers and $r$ route reflector servers in the AS, the total number of I-BGP sessions becomes $n + r(r-1)/2$, where the first term accounts for all client-reflector sessions and the second term accounts for the full reflector-reflector mesh. Obviously, this is a much smaller number than the number of sessions in the full client-client mesh since, typically, $r << n$.

*Example 1:* Consider the simple AS-network topology depicted in Figure 1. Further, assume that each network edge depicted has an associated weight/length of 1. Obviously, using a complete client-client mesh, gives rise to $8 \cdot 9/2 = 36$ I-BGP peerings. For the total length (i.e., connection cost) of all these I-BGP sessions, note that each client is within a distance of 2 of two other clients and within a distance of 3 of the remaining six clients, giving a total connection cost of $\frac{1}{2} \cdot 9 \cdot (2 \cdot 2 + 3 \cdot 6) = 99$.

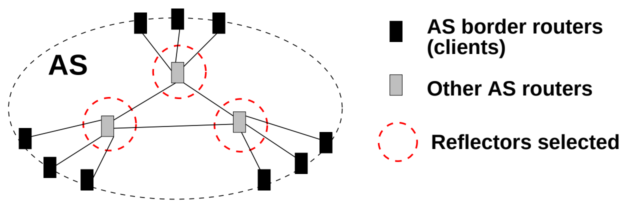


Fig. 1. Example AS-Network Topology

On the other hand, using the three highlighted AS routers as route reflectors reduces the number of peerings as well as the total connection cost to only $3 \cdot 3 + 3 = 12$. Note that the relative improvement obtained using reflectors can be made arbitrarily large by appropriate choices for the edge weights and the sizes of the client clusters. ∎

Clearly, deploying route-reflector configurations can result in drastic reductions in the total number of I-BGP peerings, the total length of I-BGP connections, as well as the total distance traversed by BGP route advertisements within an AS. However, the benefits attained by such configurations are highly dependent on how reflectors are chosen in the network. Current practice is to select reflectors manually, using ad-hoc rules typically relying on human judgement or geographical proximity; an example rule could be: "choose one reflector from each Point-Of-Presence (POP) in the network and assign all clients in the POP to that reflector". Such ad hoc solutions can result in poor reflector selections that fail to minimize I-BGP connection costs or distances traversed by advertisements.

In this paper, we formalize and address the challenging optimization problems involved in effectively configuring BGP route reflectors in large AS networks. More specifically, we consider the problems of selecting route reflectors in an AS network topology in order to minimize (1) the total connection cost (i.e., "length") of all I-BGP peering sessions, and (2) the average distance traversed by route advertisements within the AS. Despite the obvious practical importance of these problems, to the best of our knowledge, ours is the first systematic study and collection of algorithmic results on the optimization issues that arise in BGP route-reflector configurations. The main technical contributions of our work can be summarized as follows.

• **Formalization and $\mathcal{NP}$-hardness Results for BGP Route Reflector Configuration Problems.** We present formal statements of the *optimal BGP route-reflector configuration problem* with the objective of minimizing (1) total I-BGP connection cost and (2) average route-advertisement distance. Our reflector-selection problems bear some similarity to the well-known *Facility Location Problem (FLP)* that has been extensively studied in the combinatorial optimization literature [6], [7]. However, our optimality metrics have very distinctive, novel characteristics that distinguish our BGP configuration problems from FLP and its many known variants. Unfortunately, our reflector-configuration problems are also *computationally intractable*: we present $\mathcal{NP}$-hardness proofs for both of our BGP optimization problems.

• **Novel Approximation Algorithms for Effectively Configuring BGP Route Reflectors.** Given the intractability of optimal reflector configuration, we propose several novel, polynomial-time algorithms that are *provably near optimal* for our two reflector-selection problems.

For the problem of minimizing total I-BGP connection costs, we present two *constant-factor* approximation algorithms with similar worst-case performance guarantees. Our first algorithm is based on solving a continuous, *Linear-Programming (LP) relaxation* [6] of the problem and rounding off the LP solution to a near-optimal configuration. Our second algorithm is completely combinatorial (i.e., does not require solving an LP) and relies on the interesting observation that exploring several constant-factor FLP approximations over a space of input parameters is guaranteed to produce a constant-factor approximate solution to our problem.

Similarly, for the problem of minimizing the average distance traversed by route advertisements, we propose two approximation algorithms and prove their worst-case guarantees. Our first algorithm is based on LP rounding and provides a *constant factor* approximation. Our second algorithm is combinatorial and combines Bartal's general results for approximating finite metric spaces using trees [8] with a *dynamic-programming* algorithm to give an $O(\log n \log\log n)$ approximation guarantee.

• **Experimental Results Validating our Reflector Configuration Techniques.** To gauge the effectiveness of out BGP reflector configuration techniques, we have conducted a series of simulation experiments on a broad range of AS-network topologies. Our results clearly demonstrate that, compared to "naive", ad-hoc reflector-selection strategies, our proposed algorithms can offer substantial benefits and result in drastic relative improvements (as high as 62%) in the I-BGP connection and routing costs.

Due to space constraints, several of our theoretical results are presented here without proof. All the details can be found in the full version of this paper [9].

## II. System Model and Problem Formulation

**Notation and System Model.** We model the AS network as an *edge-weighted undirected graph* $G = (V, E)$, where $V = \{1, 2, \ldots, n\}$ denotes the set of network nodes (i.e., AS routers) and $E$ is a set of undirected edges representing the physical links that connect the routers. Each edge in $E$ has a weight associated with it, which essentially represents the cost of routing I-BGP message traffic using the edge. As an example, one possibility for the edge weights is to use the link weights assigned by the intra-AS routing protocol (e.g., OSPF). A subset of network nodes $C \subseteq V$ are identified as *reflector clients* (or, simply, *clients*); they basically correspond to the border routers in the AS that peer with BGP speakers in neighboring ASes to exchange prefix reachability information. We let $n = |V|$, $m = |E|$, $|C|$ denote the number of elements (i.e., nodes or edges) in sets $V$, $E$, and $C$, respectively. Table I summarizes some of the notation used throughout the paper with a brief description of its semantics. Additional notation is introduced when necessary.

| Symbol | Semantics |
|---|---|
| $G = (V, E)$ | AS Network graph |
| $n = \|V\|$, $m = \|E\|$ | Number of nodes/edges in $G$ |
| $C \subseteq V$ | Client (border) routers in the AS |
| $R \subseteq V$ | Route reflector servers in the AS |
| $d(i, j)$ | Length of shortest path between nodes $i$ and $j$ in $G$ |
| $r(i)$ | Reflector server for client node $i \in C$ |
| $\Delta_R(j) = \sum_{i \in R} d(j, i)$ | Total distance between reflector $j$ and all reflector nodes in $R$ |
| $\xi(R)$ | URS or WRS cost for reflector set $R$ |

TABLE I
Notation.

We use $d(i, j)$ to denote the length (i.e., total edge weight) of the *shortest path* between nodes $i$ and $j$ in $G$. (Note that $d()$ is obviously a *metric* over $V \times V$.) It is easy to see that deploying a full I-BGP mesh between clients in $C$ results in a total connection cost ("length") of $\sum_{i \in C} \sum_{j \in C} d(i, j)/2$. On the other hand, assume that a subset of AS nodes $R \subseteq V$ have been designated as route reflectors, and let $r(i)$ denote the route reflector server for the cluster of client node $i$. Then, the total cost of I-BGP peering sessions is reduced to $\sum_{i \in C} d(i, r(i)) + \sum_{i \in R} \sum_{j \in R} d(i, j)/2$. This cost is drastically smaller than that of the full client-client mesh, since the quadratic cost term is restricted to reflector nodes in $R$ and, typically, $|R| << |C|$.

Let us now consider the total network distance traversed by route advertisements in the AS. Under a full client-client I-BGP mesh, an advertisement received at client $i$ is forwarded directly to all other clients, giving a total distance of $\sum_{j \in C} d(i, j)$. Thus, assuming each client receives one external route advertisement packet, the distance traversed by these packets in the AS network once again explodes in a quadratic fashion to give a total distance of $\sum_{i \in C} \sum_{j \in C} d(i, j)$. On the other hand, by deploying a set of route reflectors $R$, the distance traversed by an advertisement packet received at client $i$ becomes $\sum_{i \in C} d(i, r(i)) + \sum_{j \in R} d(r(i), j)$; the first term is essentially the transmission cost from each reflector to all of its clients, while the second is the transmission cost from the reflector for client $i$ to all other reflectors. Assuming one external route advertisement per client, the total AS distance covered by these advertisement packets is $\sum_{i \in C} \left[ n \cdot d(i, r(i)) + \sum_{j \in R} d(r(i), j) \right]$. It is once again easy to see that, by appropriately choosing reflectors in $R$, this distance can be made drastically smaller than the corresponding distance for the full client-client mesh.

**Problem Statement.** Despite the obvious practical importance of optimizing the configuration of BGP route reflectors in an AS network, to the best of our knowledge, there have been no previous results on this very interesting problem. In our work, we consider two distinct problem variants. The first variant (referred to as *"Unweighted Reflector Selection (URS)"*) tries to minimize the *total connection cost* of all I-BGP peerings in the configuration. The second variant (referred to as *"Weighted Reflector Selection (WRS)"*) attempts to minimize the *average distance traversed* by BGP route advertisements within the AS.

More formally, let $\Delta_R(j)$ denote the sum of the shortest-path distances from a reflector $j$ to all reflectors in $R$ or, equivalently, $\Delta_R(j) = \sum_{i \in R} d(j,i)$. Our URS and WRS problems can be stated as follows.

**[Unweighted Reflector Selection (URS) Problem]:** Given an AS $G = (V, E)$ and a set of clients $C \subseteq V$, compute a set of reflectors $R \subseteq V$ such that $\xi(R) = \sum_{i \in C} \min_{j \in R}\{d(i,j)\} + \sum_{j \in R} \Delta_R(j)/2$ is minimum. ▮

**[Weighted Reflector Selection (WRS) Problem]:** Given an AS $G = (V, E)$, a set of clients $C \subseteq V$, and a weight parameter $0 < w < 1$, compute a set of reflectors $R \subseteq V$ such that $\xi(R) = \sum_{i \in C} \min_{j \in R}\{d(i,j) + w \cdot \Delta_R(j)\}$ is minimum. ▮

Note that both the URS and the WRS problems, essentially boil down to computing a set of reflector nodes $R$ and not the individual assignments of clients to reflectors. This is because, given a set of reflectors $R$, assigning each client $i$ to the reflector $j \in R$ that minimizes the $\min\{\}$ term in the objective function (i.e., $d(i,j)$ in URS and $d(i,j) + w \cdot \Delta_R(j)$ in WRS) always results in minimizing the overall problem objective. Thus, assignments and overall configuration costs are, in a sense, "fixed" once $R$ is selected.

The objective function in the WRS problem tries to capture the average distance traversed by externally-learned BGP route advertisements under the assumption that client routers receive advertisements at a uniform rate. For example, when $w = 1/n$ then $n$ times the WRS objective function gives exactly the total distance traversed by advertisements in the AS assuming each client receives one route advertisement from external BGP peers (derived earlier in this section). As we demonstrate in the full paper [9], different values of the weight parameter $w$ can allow WRS to also model more complicated scenarios, e.g., situations where reflectors are allowed to *selectively* propagate routes learned by their clients. Thus, we address the general case of WRS with arbitrary values for the weight parameter $w$.

Finally, we should note that our system model and problem formulation are very general and allow both clients (i.e., border routers) as well as non-clients in the AS to be chosen as reflectors. Of course, if certain network nodes are not equipped to reflect BGP route advertisements, then our model can prevent such nodes from being chosen as reflectors by simply excluding them from $V$.

## III. The URS Problem: Complexity and Approximation Algorithms

Our Unweighted Reflector Selection (URS) formulation bears some similarities to the well-known *Facility Location Problem (FLP)* [6], [7]. There is, however, one crucial difference. If we view the reflector nodes in $R$ as "facilities" then, in our URS problem, the cost of each individual facility is zero; nevertheless, the set of chosen facilities *as a whole* has an associated cost given by the total of their pairwise distances ($\sum_{j \in R} \Delta_R(j)/2$) (since reflectors need to be fully meshed). Associating a cost with *a set of facilities* rather than individual facilities makes URS a novel combinatorial optimization problem that is very different from FLP and its many known variants. Unfortunately, as the following theorem demonstrates, it is highly unlikely that we can find an optimal solution to URS in a computationally-efficient manner.

*Theorem 1:* The URS problem is $\mathcal{NP}$-hard. ▮

Given the intractability of URS, we seek to find efficient (i.e., polynomial-time) approximation algorithms with a guaranteed upper bound on the deviation from the optimal solution. We propose two novel, *constant-factor* approximation algorithms for URS; our first algorithm is based on LP rounding, whereas the second is completely combinatorial (i.e., does not require solving an LP).

### A. Integer Programming Formulation and LP Rounding-Based Approximation Algorithm

The URS problem can be formulated as the following Integer Program (IP). Variables $x_{ij}$, $y_{ii'}$, and $z_{iji'j'}$ are 0-1 variables with the following semantics. Variable $x_{ij}$ is 1 iff node $i$ is selected as a reflector and client $j$ is assigned to it. Variable $y_{ii'}$ is 1 iff both $i$ and $i'$ are selected as reflectors. Finally, variable $z_{iji'j'}$ is 1 iff nodes $i$ and $i'$ are chosen as the reflectors for clients $j$ and $j'$, respectively.

$$\text{Minimize} \sum_{j \in C, i \in V} d(i,j) \cdot x_{ij} + \sum_{i,i' \in V} \frac{d(i,i')}{2} \cdot y_{ii'} \quad (1)$$

subject to the following constraints:

$$\forall j \in C: \quad \sum_{i \in V} x_{ij} = 1 \qquad (2)$$

$$\forall j, j' \in C, i \in V: \quad \sum_{i' \in V} z_{iji'j'} = x_{ij} \qquad (3)$$

$$\forall j, j' \in C, i, i' \in V: \quad z_{iji'j'} = z_{i'j'ij} \qquad (4)$$

$$\forall j, j' \in C, i, i' \in V: \quad z_{iji'j'} \leq y_{ii'} \qquad (5)$$

$$\forall j, j' \in C, i, i' \in V: \quad x_{ij}, y_{ii'}, z_{iji'j'} \in \{0, 1\} \qquad (6)$$

In the above formulation, Equation (1) essentially captures the objective function (cost) of our URS problem formulation. Constraint (2) ensures that a reflector server is selected for every client, while Constraints (3–5) guarantee that values of the $y$ and $z$ variables are consistent with values of $x$. Let $(x, y, z)$ be an optimal solution to the above IP; then, the optimal set of reflectors $R$ is simply $R = \{i : x_{ij} = 1 \text{ for some } j\ \}$.

Directly solving an IP formulation (like the one presented above) is known to be computationally intractable [6]. We have instead used this formulation to derive a constant-factor approximation algorithm to URS based on *LP-rounding techniques*. Briefly, our algorithm is based on solving the continuous LP relaxation of the IP, and then rounding the fractional LP solution to a nearby integer solution (i.e., a URS configuration) in a way that increases the cost of the fractional solution by a constant factor. Our LP-rounding scheme for URS is, in fact, similar to the LP-rounding scheme we have devised for WRS which is developed in detail in Section IV. We state our main theorem below and refer the interested reader to the full paper [9] for all the details.

*Theorem 2:* There exists a polynomial-time approximation algorithm for URS based on LP-rounding that is guaranteed to be within a factor of 16 of the optimal URS solution. ∎

## B. Combinatorial Approximation Algorithm

We now propose a *combinatorial* constant-factor approximation algorithm for URS. The key observation exploited in our algorithm is that we can produce a near-optimal solution to URS by exploring near-optimal solutions to multiple instances of the *Bounded Facility Location (BFL)* problem [7]. Briefly, the BFL problem can be stated as follows: "Given a set of facilities $V$, clients $C$, costs $c(i)$ for locating each facility $i$, distances $d(i, j)$ between clients and facilities, and a constant $k$, find a subset of facilities $R \subseteq V$ of size no more than $k$ such that $\sum_{j \in C} \min_{i \in R}\{d(i, j)\} + \sum_{i \in R} c(i)$ is minimized." BFL has been extensively studied in the literature and, recently, Jain and Vazirani [7] have proposed the best-known combinatorial approximation algorithm. Their algorithm is based on the primal-dual method, provides a guarantee of 6 on the quality of the computed BFL solution, and has a worst-case time complexity of $O(|V|^2 \log |V|)$.

The instances of the BFL problem that we solve for our URS algorithm are parameterized by two variables: (1) a network node $i \in V$, and (2) the upper bound $k$ on the number of facilities. In the parameterized BFL instance BFL$(i, k)$, the set of clients $C$ and the distances $d()$ are identical to those defined in the URS problem instance; further, we use $V$ as the set of all facilities, $k$ as the bound on the number of facilities to be opened, and set the cost of each facility $l \in V$ as $c(l) = k \cdot d(i, l)$. Thus, for instance BFL$(i, k)$, node $i$ essentially serves as the basis/origin for defining the costs of individual facilities. The set of facilities $R$ that are returned as an (approximate) solution to instance BFL$(i, k)$ obviously correspond to to a subset of the network nodes $V$; further, the cost of the solution $R$ for the BFL$(i, k)$ instance is $\xi_B(R, i, k) = \sum_{j \in C} \min_{i \in R}\{d(i, j)\} + \sum_{l \in R} k \cdot d(i, l)$. The following lemma demonstrates that, given any subset of nodes $R \subseteq V$, the cost of $R$ as a solution to BFL$(i, k)$ always dominates the cost of the same subset $R$ when viewed as a solution to the URS problem. Lemma 2 then shows that, given any $R \subseteq V$, there always exists a node $i \in R$ such that the cost of $R$ as a solution to BFL$(i, |R|)$ is within a factor of two of $R$'s cost as a URS solution.

*Lemma 1:* Let $R$ be a solution to the BFL instance BFL$(i, k)$. Then, $\xi(R) \leq \xi_B(R, i, k)$. ∎

*Proof:* Both $\xi$ and $\xi_B$ are identical with respect to the cost of assigning clients to facilities/reflectors in $R$, since in each case clients are assigned to the closest node in $R$. So we only need to show that the cost of connecting reflectors in $\xi(R)$ (that is, $\sum_{l \in R} \Delta_R(l)/2$) does not exceed $\sum_{l \in R} k \cdot d(i, l)$. Consider an arbitrary pair of reflectors $l, l'$ in $R$. Then, due to the triangle inequality, $d(l, l') \leq d(i, l) + d(i, l')$. Thus, in $\xi(R)$, since each reflector $l$ is connected to at most $k - 1$ other reflectors in $R$ (note that $|R| \leq k$), it follows that $\sum_{l \in R} \Delta_R(l)/2 \leq \sum_{l \in R}(k - 1) \cdot d(i, l)$. Thus, $\xi(R) \leq \xi_B(R, i, k)$. ∎

*Lemma 2:* Let $R$ be a solution to the URS problem, and let $|R| = k$. Then there exists a node $i \in R$ such that $\xi_B(R, i, k) \leq 2 \cdot \xi(R)$. ∎

*Proof:* Let $i$ be the reflector in $R$ for which $\Delta_R(i) = \sum_{l \in R} d(i, l)$ is minimum. Thus, $\xi(R) \geq \sum_{j \in C} \min_{l \in R}\{d(l, j)\} + k \cdot \Delta_R(i)/2$. Now, consider the problem instance BFL$(i, k)$. The cost of the solution to BFL$(i, k)$ consisting of facilities $R$ is $\xi_B(R, i, k) = \sum_{j \in C} \min_{l \in R}\{d(l, j)\} + k \cdot \Delta_R(i)$, which is less than or

```
procedure URSviaBFL($G = (V, E)$ , $C$)
1.  minCost := $\infty$, $R := \emptyset$
2.  for $k := 1$ to $|V|$
3.      for each node $i$ in $V$ {
4.          Let $R'$ be the near-optimal solution to BFL($i, k$)
                due to the Jain-Vazirani primal-dual algorithm
5.          if $\xi_B(R', i, k) <$ minCost then
6.              minCost := $\xi_B(R', i, k)$, $R := R'$
7.      }
8.  return $R$
```

Fig. 2. Polynomial-Time Combinatorial Algorithm for URS

equal to $2 \cdot \xi(R)$. ▮

Algorithm URSviaBFL (depicted in Figure 2) exploits the results in Lemmas 1 and 2 to produce a near-optimal URS solution. The key idea in URSviaBFL is to explore approximate solutions to BFL($i, k$) for all possible choices of the number of facilities $k = 1, \ldots, |V|$ and basis node $i \in V$, and select the one that results in the minimum BFL cost. As Theorem 3 shows, this solution is guaranteed to be within a constant factor of the optimal URS solution.

*Theorem 3:* Algorithm URSviaBFL computes a set of reflectors $R \subseteq V$ for which $\xi(R)$ is within a factor of 12 of the cost of the optimal URS solution. ▮

*Proof:* Suppose that $R_{opt}$ is the optimal solution to the URS problem and further, it contains $k_{opt}$ reflectors. Note that URSviaBFL iterates through all nodes $i \in V$ for all values of $k$, and chooses $R$ to be the near-optimal set of reflectors, say $R'$, computed by the Jain and Vazirani primal-dual algorithm for the BFL instance BFL($i', k'$) and for which the cost $\xi_B(R', i', k')$ is minimum. Let $R'_{opt}$ be the optimal solution over all BFL instances BFL($i, k$), $i \in V$ and $k = 1, \ldots, |V|$. Further, let $i'_{opt}$ and $k'_{opt}$ be the values such that for all $i \in V$ and $k = 1, \ldots, |V|$, and for all $R \subseteq V$, $\xi_B(R'_{opt}, i'_{opt}, k'_{opt}) \leq \xi_B(R, i, k)$. Then, since the solution to BFL($i'_{opt}, k'_{opt}$) computed by the Jain-Vazirani algorithm is within a factor of 6 of optimal, we have $\xi_B(R', i', k') \leq 6 \cdot \xi_B(R'_{opt}, i'_{opt}, k'_{opt})$. By Lemma 2, $\xi_B(R'_{opt}, i'_{opt}, k'_{opt}) \leq 2 \cdot \xi(R_{opt})$. Further, by Lemma 1, for the set of reflectors $R = R'$ returned by the procedure, $\xi(R') \leq \xi_B(R', i', k')$. Thus, $\xi(R) \leq 12 \cdot \xi(R_{opt})$. ▮

**Time Complexity.** It is easy to see that the worst-case time complexity of Algorithm URSviaBFL is $O(n^4 \log n)$: the Jain-Vazirani algorithm runs in time $O(n^2 \log n)$ and our algorithm invokes it $O(n^2)$ times.

## IV. The WRS Problem: Complexity and Approximation Algorithms

Theorem 4 establishes the intractability of our Weighted Reflector Selection (WRS) problem. The remainder of this section then proposes two novel, provably near-optimal algorithms for WRS: a *constant-factor* approximation algorithm based on LP rounding and a combinatorial $O(\log n \log \log n)$ approximation algorithm.

*Theorem 4:* The WRS problem is $\mathcal{NP}$-hard. ▮

### A. Integer Programming Formulation and LP Rounding-Based Approximation Algorithm

Similar to URS, WRS can be formulated as the following Integer Program (IP). The 0-1 variables $x_{ij}$ and $z_{iji'j'}$ used below have exactly the same semantics as in our URS IP formulation, while the 0-1 variables $y_{iji'}$ have a value of 1 iff both $i$ and $i'$ are selected as reflectors and client $j$ is assigned to reflector $i$. The semantics of the various IP equations/constraints are also similar to the URS case.

$$\text{Minimize} \sum_{j \in C, i \in V} d(i,j) \cdot x_{ij} + w \cdot \sum_{j \in C} \sum_{i, i' \in V} d(i, i') \cdot y_{iji'} \quad (7)$$

subject to the following constraints

$$\forall j \in C: \quad \sum_{i \in V} x_{ij} = 1 \quad (8)$$

$$\forall j, j' \in C, i \in V: \quad \sum_{i' \in V} z_{iji'j'} = x_{ij} \quad (9)$$

$$\forall j, j' \in C, i, i' \in V: \quad z_{iji'j'} = z_{i'j'ij} \quad (10)$$

$$\forall j, j' \in C, i, i' \in V: \quad z_{iji'j'} \leq y_{iji'} \quad (11)$$

$$\forall j, j' \in C, i, i' \in V: \quad x_{ij}, y_{iji'}, z_{iji'j'} \in \{0, 1\} \quad (12)$$

We now propose a constant-factor approximation algorithm for WRS that relies on rounding the solution to the continuous, LP relaxation of the above IP. Our LP-rounding algorithm consists of two phases. First, we apply the filtering technique of Lin and Vitter [10] to obtain a new fractional solution, with the property that whenever a client $j$ is fractionally assigned to a (partially chosen) node $i$, the distance $d(i, j)$ associated with that assignment is not too big. Second, we show how a fractional solution with this "closeness property" can be rounded to a near-optimal integer solution.

Consider the LP relaxation of the IP (7)-(12), where the 0-1 value constraints (12) are replaced with $\forall j, j' \in C, i, i' \in V$, $x_{ij} \geq 0$, $y_{iji'} \geq 0$ and $z_{iji'j'} \geq 0$. We can solve this LP in polynomial time [6]. Let $(x, y, z)$ denote the optimal fractional solution to the LP. In the following, we show how to round this fractional solution to an integer solution without increasing the value of the LP objective function by more than a constant factor.

The first step is *filtering* [10]. Let $\alpha$ be a constant between 0 and 1 (we fix its value later). For each client $j$, we define a quantity $c_j(\alpha)$ as follows. Sort the nodes in $V$ in ascending order of distance from $j$, and let $\pi$ be the resulting permutation on nodes such that $d(\pi(1), j) \leq d(\pi(2), j) \leq \cdots \leq d(\pi(n), j)$. Define $i^* = \min\{i' : \sum_{i=1}^{i'} x_{\pi(i)j} \geq \alpha\}$ and $c_j(\alpha) = d(\pi(i^*), j)$. We will use the following fact later to prove our approximation result.

$$(1-\alpha)c_j(\alpha) \;\leq\; \sum_{i \geq i^*} x_{\pi(i)j} d(\pi(i), j) \;\leq\; \sum_{i \in V} d(i,j) x_{ij} \quad (13)$$

Let $V_j = \{i \in V : d(i,j) \leq c_j(\alpha)\}$, i.e., the "neighborhood" of $j$ in the graph within radius $c_j(\alpha)$. Note that $\sum_{i \in V_j} x_{ij} \geq \alpha$. Also, due to Equation (13) above, it follows that assigning $j$ to any node in $V_j$ will increase the connection cost by at most a constant factor (i.e., $1/(1-\alpha)$). We can show the following property about the fractional LP solution.

*Lemma 3:* Consider two clients $j$ and $j'$. Then $\sum_{i \in V_j, i' \in V_{j'}} z_{iji'j'} \geq 2\alpha - 1$. ∎

Given a client $j$, define $\Delta(j) = \sum_{i,i' \in V} d(i,i') y_{iji'}$. This is exactly the contribution of $j$ to the second term in our WRS objective function (Equation (7)).

We now show how to get a near-optimal integral solution. Algorithm WRSRound shown in Figure 3 computes a feasible integer solution $(\hat{x}, \hat{y}, \hat{z})$ that is within a constant factor of the optimal fractional solution $(x, y, z)$. Briefly, our rounding procedure works by *clustering* all the clients in $C$ in its main **while**-loop (Steps 4–15). Each cluster has a client $j$ that is the *seed* of the cluster. The set of seed clients are stored in seedSet and $S_j$ is the cluster with client $j$ as the seed. By the operation of WRSRound, every client $j' \in S_j$ has the following properties: (1) $(\beta+1) \cdot c_j(\alpha) + \gamma \cdot w \cdot \Delta(j) \leq (\beta+1) \cdot c_{j'}(\alpha) + \gamma \cdot w \cdot \Delta(j')$ (Step 5), and (2) $d(j,j') \leq \beta \cdot \max\{c_j(\alpha), c_{j'}(\alpha)\}$ (Steps 9–12). Here, $\beta > 2$ and $\gamma$ are parameters whose value we shall specify later. After the construction of cluster $S_j$ is complete, we select an arbitrary node in $V_j$ as the route reflector for all the clients in $S_j$. For the sake of concreteness, let this reflector be denoted as $i^*(j)$. Thus, the set of reflectors returned by WRSRound is exactly $R = \{i^*(j) : j \in \text{seedSet}\}$.

Due to the manner in which seeds are chosen in WRSRound, clients in seedSet satisfy the following property.

*Lemma 4:* Let $j$ and $j'$ be two clients in seedSet. For nodes $i \in V_j$ and $i' \in V_{j'}$, $d(i,i') > (\beta-2) \cdot \max\{c_j(\alpha), c_{j'}(\alpha)\}$. ∎

```
procedure WRSRound($c(\alpha)$[ ], $V$[ ], $\Delta$[ ])
 1.  $R := \emptyset$
 2.  activeSet := $C$
 3.  seedSet := $\emptyset$
 4.  while activeSet $\neq \emptyset$ {
 5.     Let $j$ be a client in activeSet with the minimum
           value for $(\beta+1) \cdot c_j(\alpha) + \gamma \cdot w \cdot \Delta(j)$
 6.     seedSet := seedSet $\cup \{j\}$
 7.     $S_j := \{j\}$
 8.     for each client $j'$ in activeSet {
 9.        if $d(j,j') \leq \beta \cdot \max\{c_j(\alpha), c_{j'}(\alpha)\}$ {
10.           activeSet := activeSet $- \{j'\}$
11.           $S_j := S_j \cup \{j'\}$
12.        }
13.     }
14.     Add an arbitrary node in $V_j$ to $R$
15. }
16. return $R$
```

Fig. 3. Algorithm for Rounding the Fractional LP Solution to a Route-Reflector Configuration

We next show that for each client $j \in$ seedSet, the sum of the distances of its reflector $i^*(j)$ to other reflectors in $R$ is a good approximation of $\Delta_j$. This is the key lemma for our constant-factor approximation proof.

*Lemma 5:* Let $j$ be a client in seedSet. Then,

$$\sum_{j' \in \text{seedSet}} d(i^*(j), i^*(j')) \leq \frac{\beta+2}{(\beta-2)\cdot(2\alpha-1)} \Delta_j. \quad ∎$$

*Proof:* Due to Lemma 4, it follows that for $j, j' \in$ seedSet, $V_j \cap V_{j'} = \emptyset$. Also, due to Constraint (11), $y_{iji'} \geq z_{iji'j'}$. Thus,

$$\Delta_j = \sum_{i,i' \in V} d(i,i') \cdot y_{iji'} \geq \sum_{i \in V_j} \sum_{j' \in \text{seedSet}} \sum_{i' \in V_{j'}} d(i,i') \cdot z_{iji'j'}.$$

Now, we know from Lemma 4 that $d(i,i') \geq (\beta-2) \cdot \max\{c_j(\alpha), c_{j'}(\alpha)\}$, where $i \in V_j$ and $i' \in V_{j'}$. But since $i, i^*(j) \in V_j$ and $d$ satisfies the triangle inequality, $d(i, i^*(j)) \leq 2 \cdot c_j(\alpha)$. Similarly, we can show that $d(i', i^*(j')) \leq 2 \cdot c_{j'}(\alpha)$. So it follows that

$$\begin{aligned} d(i,i') &\geq d(i^*(j), i^*(j')) - 2c_j(\alpha) - 2c_{j'}(\alpha) \\ &\geq d(i^*(j), i^*(j')) - \frac{4}{\beta-2} d(i,i'). \end{aligned}$$

So we get $d(i,i') \geq \frac{\beta-2}{\beta+2} d(i^*(j), i^*(j'))$. Now, we can modify the above equation for $\Delta_j$ as follows by first substituting for $d(i,i')$ and then applying Lemma 3.

$$\begin{aligned} \Delta_j &\geq \frac{\beta-2}{\beta+2} \sum_{i \in V_j} \sum_{j' \in \text{seedSet}} \sum_{i' \in V_{j'}} d(i^*(j), i^*(j')) z_{iji'j'} \\ &\geq \frac{\beta-2}{\beta+2} \sum_{j' \in \text{seedSet}} d(i^*(j), i^*(j')) \sum_{i \in V_j} \sum_{i' \in V_{j'}} z_{iji'j'} \\ &\geq \frac{(\beta-2)\cdot(2\alpha-1)}{\beta+2} \sum_{j' \in \text{seedSet}} d(i^*(j), i^*(j')). \end{aligned}$$

This proves the desired result. ∎

The following lemma builds on Lemma 5 to show that assigning each client $j' \in S_j$ to route reflector $i^*(j)$ does not increase the contribution of $j$ to the overall objective function value by too much. Let $\gamma = \frac{\beta+2}{(\beta-2)\cdot(2\alpha-1)}$.

*Lemma 6:* Consider a client $j \in \mathsf{seedSet}$. Let $j' \in S_j$. Then, $d(i^*(j), j') + w \cdot \sum_{i \in R} d(i^*(j), i) \leq 2(\beta + 1)c_{j'}(\alpha) + \gamma \cdot w \cdot \Delta_{j'}$. ∎

We are now in a position to show the near-optimality of the final rounded integer solution $(\hat{x}, \hat{y}, \hat{z})$ obtained through WRSROUND. In this solution, all variables are set to 0 except the following, which are set to 1:

1) $\hat{x}_{i^*(j)j'}$, where $j \in \mathsf{seedSet}$ and $j' \in S_j$.
2) $\hat{y}_{iji'}$, where $\hat{x}_{ij}$ is set to 1, in Step (1) above, and $i' \in R$.
3) $\hat{z}_{iji'j'}$, where $\hat{x}_{ij}$ and $\hat{x}_{i'j'}$ are both set to 1 in Step (1) above.

This integer solution is clearly feasible and, as the following theorem shows, within a constant factor of the optimal LP solution.

*Theorem 5:* The cost of integer solution $(\hat{x}, \hat{y}, \hat{z})$ derived above is within a factor of 21 of the cost of the optimal LP solution $(x, y, z)$. ∎

*Proof:* The cost of the integer solution $(\hat{x}, \hat{y}, \hat{z})$ is given by $\sum_{j \in C} \cdot (\sum_{i \in V} d(i,j) \cdot \hat{x}_{ij} + \sum_{i,i' \in V} \hat{y}_{iji'} \cdot d(i,i'))$. Due to Lemma 6, $\sum_{i \in V} d(i,j) \cdot \hat{x}_{ij} + \sum_{i,i' \in V} \hat{y}_{iji'} \cdot d(i,i') \leq 2(\beta+1) \cdot c_j(\alpha) + \gamma \cdot w \cdot \Delta_j$. Further, applying Equation (13) and the fact that $1 - \alpha < 1$, we get $\sum_{j \in C} \cdot (\sum_{i \in V} d(i,j) \cdot \hat{x}_{ij} + \sum_{i,i' \in V} \hat{y}_{iji'} \cdot d(i,i')) \leq \sum_{j \in C} \left( \frac{2(\beta+1)}{1-\alpha} \sum_{i \in V} d(i,j) \cdot x_{ij} + \gamma \cdot w \cdot \Delta_j \right)$. Thus, the cost of $(\hat{x}, \hat{z})$ is within $\max\left\{ \frac{2(\beta+1)}{1-\alpha}, \gamma \right\}$ of the optimal fractional solution (to the LP). Choosing $\alpha = 13/21$ and $\beta = 3$, we get the (optimal) value of 21 for the constant approximation factor. ∎

## B. Combinatorial Approximation Algorithm

The rounding-based algorithm in the previous section can be computationally expensive due to the large number of variables in the LP that must be solved. We now present a *combinatorial* algorithm for computing a near-optimal set of reflectors $R$ whose WRS cost $\xi(R) = \sum_{j \in C} \min_{i \in R}\{d(i,j) + w \cdot \Delta_R(i)\}$ is within $O(\log n \log\log n)$ of the optimal solution. Further, the worst-case time complexity of the algorithm is $O(n^4|C|)$ (remember that $|C|$ is the number of clients in $G$).

Our overall approach is based on transforming the network graph $G$ into a *2-hierarchically well-separated tree (2-HST)* $T$ [8], and then using *dynamic programming* to compute a constant-factor approximation to the optimal set of reflectors for $T$. Our rationale is that it is much easier to develop good approximation algorithms for trees compared to general graphs. Using Bartal's general results [8], we are guaranteed that transforming the network graph to a 2-HST can increase the cost of the final set of reflectors by a factor of at most $O(\log n \log\log n)$.

*1) Mapping the General WRS Problem to 2-HSTs:* A *$k$-hierarchically well-separated tree ($k$-HST) $T$* for a graph $G$ is defined as a rooted, edge-weighted tree with the following properties [8], [11]: (1) Every node in $G$ is a leaf of $T$; (2) the distance of a node in $T$ to each of its children is the same; and, (3) the distances between successive nodes along any path from the root to a leaf are decreasing by a factor of at least $k$.

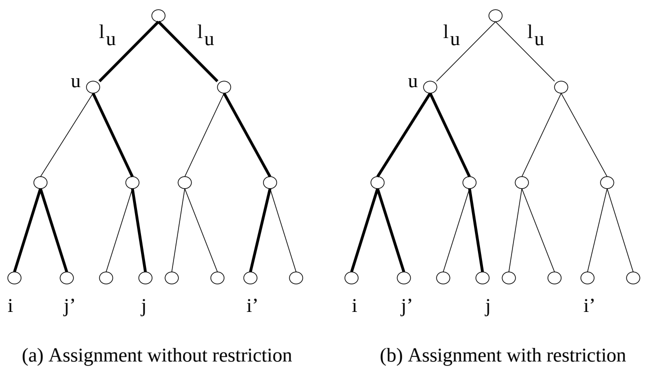
Bartal [8] has shown that $k$-HSTs can be used to probabilistically approximate an arbitrary finite metric space (i.e., edge-weighted graph $G$), so that the *expected distance* between any two nodes in the $k$-HST is within a factor of at most $O(\log n \log\log n)$ of their distance in $G$. Bartal's result has been used to develop effective approximation algorithms for several $\mathcal{NP}$-hard graph optimization problems, like $k$-median and group Steiner tree. Recent work has also demonstrated ways to de-randomize Bartal's result to obtain deterministic performance guarantees through HSTs [11].

Our approach is based on solving the WRS problem for the 2-HST approximation $T$ of the AS-network graph $G$. Let $u$ be an arbitrary node in the 2-HST $T$, and let $d_T()$ denote the node-distance metric defined by $T$ (i.e., $d_T(u,v)$ is the distance between $u$ and $v$ in $T$). We denote the subtree of $T$ rooted at node $u$ by $T_u$. We also use $p_u$ to refer to the parent of $u$ in $T$ and define $l_u = d_T(u, p_u)$. Note that, by the definition of 2-HSTs, the distance of an arbitrary node $u$ to a leaf in its subtree $T_u$ is at most $l_u$. Finally, observe that all the nodes in $V$, including all clients $C \subseteq V$, are leaves in $T$.

Given a set of reflectors $R$, the WRS cost of $R$ based on the distances in $T$ is given by $\xi_T(R) = \sum_{j \in C} \min_{i \in R}\{d_T(i,j) + w \cdot \Delta_{R,T}(i)\}$, where $\Delta_{R,T}(i) = \sum_{l \in R} d_T(i,l)$. Using the results in [8], [11] to construct $T$, we can guarantee that $\xi(R) \leq \xi_T(R) \leq O(\log n \log\log n) \cdot \xi(R)$. Thus, a constant-factor approximation algorithm for WRS in $T$ is always within $O(\log n \log\log n)$ of the optimal WRS solution for $G$.

Our goal is to design a *dynamic-programming* algorithm that exploits the hierarchical structure of the 2-HST approximation $T$ of $G$ to solve the WRS in $T$. The main difficulty in computing the truly optimal reflector set for $T$ using such an approach is that, given a set of reflectors $R$, a client $j$ in a subtree $T_u$ of the HST can be assigned to a reflector outside of $T_u$, even though $T_u$ already contains a reflector in $R$.

(Remember that WRS always assigns $j$ to the reflector $i$ that minimizes $d_T(i,j) + \Delta_{R'_{opt},T}(i)$.) However, if we impose a restriction on the assignment of clients to reflectors to avoid such scenarios, then it is possible to compute the ("restricted") optimal set of reflectors using dynamic programming.



l_u
l_u
u
i
j'
j
i'
(a) Assignment without restriction
l_u
l_u
u
i
j'
j
i'
(b) Assignment with restriction

Fig. 4. Example of Restricted Assignment

More formally, et $T(R,j)$ denote the smallest subtree of $T$ that contains client $j$ and a reflector from $R$. Then, in our restricted version of WRS in $T$, each client $j$ is assigned to the best reflector in $T(R,j)$ (rather than $T$). In other words, the cost function for a set of reflectors $R$ in our restricted WRS problem in $T$ (denoted by $\xi_T^{res}(R)$) can be expressed as $\sum_{j\in C} \min_{i\in T(R,j)}\{d_T(i,j) + \Delta_{R,T}(i)\}$. Figure 4 gives an illustrative example of the restriction that we impose for solving WRS in $T$, for clients $j, j'$ and reflectors $R = \{i, i'\}$. Note that $T(R,j) = T_u$ and, thus, the assignment shown in Figure 4(a) (where $j$ is assigned to $i'$) does not satisfy our restriction. In contrast, the assignment of $j$ to $i$ in Figure 4(b) is in line with our restriction on WRS. The key observation here is that, as the following lemma shows, restricting the assignment of clients in this way does not degrade the quality of the solution obtained by more than a (constant) factor of two.

*Lemma 7:* Let $R$ and $R'_{opt}$ be the reflector sets that minimize the cost functions $\xi_T^{res}(R)$ and $\xi_T(R'_{opt})$, respectively. Then, $\xi_T(R) \leq 2 \cdot \xi_T(R'_{opt})$. ∎

Next, we present our dynamic-programming algorithm that computes the optimal set of reflectors $R$ for our restricted WRS problem over $T$. From our discussion above and Lemma 7, we can then guarantee that the WRS cost for this set $R$ in $G$ (i.e., $\xi(R)$) is within a factor of $O(\log n \log\log n)$ of the optimal solution.

*2) Solving the Restricted WRS Problem over $T$:* Our goal is to compute a set of reflectors $R$ that minimizes the cost $\xi_T^{res}(R)$ using *dynamic programming* to exploit the hierarchical structure of $T$. The key observation is that we can view this cost $\xi_T^{res}(R)$ as the *sum of edge costs* in $T$, where the cost of edge $(u, p_u)$ is defined as follows. Assume that $R$ contains a total of $K$ reflectors and $T_u$ contains $k$ reflectors from $R$. Also, let $C(T_u)$ denote the number of clients in subtree $T_u$ and let $N$ be the number of clients outside of $T_u$ that are assigned to reflectors in $T_u$ in the computation of $\xi_T^{res}(R)$. We distinguish two cases for the cost of edge $(u, p_u)$:

1) If $k = 0$, then, since there are no reflectors from $R$ in subtree $T_u$, all clients in $T_u$ are assigned to reflectors outside $T_u$. Thus, edge $(u, p_u)$ contributes $l_u$ for each client in $C(T_u)$ to the overall cost, and the edge cost is $C(T_u) \cdot l_u$.
2) If $k > 0$, then (by our restriction) all clients in $T_u$ are assigned to reflectors in $T_u$. In this case, the contribution of edge $(u, p_u)$ to the overall cost is $N \cdot l_u + w \cdot (N + C(T_u)) \cdot (K - k) \cdot l_u + w \cdot (|C| - N - C(T_u)) \cdot k \cdot l_u$, where: (1) the first term is due to the $N$ clients outside $T_u$ that are assigned to reflectors in $T_u$ via edge $(u, p_u)$, (2) the second term is the contribution of edge $(u, p_u)$ to the connection cost between reflectors for the $N + C(T_u)$ clients assigned to reflectors in $T_u$, and (3) the final term is the contribution of edge $(u, p_u)$ to the connection cost between reflectors for the $|C| - N - C(T_u)$ clients assigned to reflectors outside $T_u$.

It is easy to verify that the cost $\xi_T^{res}(R)$ is exactly the sum of the above-defined costs over all edges of $T$. Note that the cost of edge $(u, p_u)$ depends entirely on the total number $K$ of reflectors in $R$, the number of reflectors $k$ from $R$ in $T_u$, and the number of clients $N$ outside $T_u$ that are assigned to reflectors in $T_u$. Further, assignments of clients to reflectors in $\xi_T^{res}(R)$ can be shown to satisfy the following property, which we use in developing our dynamic-programming algorithm.

*Lemma 8:* Consider a subtree $T_u$ containing a reflector from $R$. Let $C'$ be a subset of clients not in $T_u$ that are assigned to a reflector in $T_u$. Then, all clients in $C'$ are assigned to the *same* reflector in $T_u$. ∎

We now use our edge-cost formulation to compute the set of reflectors $R$ in $T$ that minimizes $\xi_T^{res}(R)$. We begin by describing how to compute the optimal reflector set $R$ containing *exactly* $K$ reflectors; the overall optimal for $\xi_T^{res}()$ can then be computed by iterating over all possible values of $K$ in $\{1, \ldots, |V|\}$ and selecting the solution with minimum cost.

Consider a *set of subtrees* $\mathcal{T}$ of $T$, and let $\xi_T^{res}(\mathcal{T}, k, K, N)$ denote the minimum value of the total cost for all edges incident on nodes belonging to $\mathcal{T}$ when: (1) $K$ total reflectors are chosen in $T$, (2) $k$ reflectors are chosen in the subtrees in $\mathcal{T}$, and (3) $N$

clients outside $\mathcal{T}$ are assigned to the $k$ reflectors in $\mathcal{T}$. (If there are fewer than $k$ leaves in $\mathcal{T}$, then define $\xi_T^{res}(\mathcal{T}, k, K, N) = \infty$.) Thus, $\xi_T^{res}(\{T\}, K, K, 0)$ is essentially the optimal value of the cost for our restricted WRS over $T$ when exactly $K$ reflectors are chosen. Our algorithm computes the values $\xi_T^{res}(\mathcal{T}, k, K, N)$ recursively, using the dynamic-programming relationships outlined below. Note that our algorithm considers each case in the order listed below.

1) $\underline{k > \text{no. of leaves in } \mathcal{T}}$: Set $\xi_T^{res}(\mathcal{T}, k, K, N) = \infty$
2) $\underline{k = 0 \text{ and } \mathcal{T} = \{T_u\}}$: Set $\xi_T^{res}(\mathcal{T}, k, K, N) = C(T_u) \cdot l_u + \sum_{v:p_v=u} \xi_T^{res}(\{T_v\}, 0, K, 0)$
3) $\underline{\mathcal{T} = \{T_u\} \text{ and } T_u \text{ is a leaf } (k = 1)}$: Set $\xi_T^{res}(\mathcal{T}, k, K, N) = = N \cdot l_u + w \cdot (N + C(T_u)) \cdot (K - 1) \cdot l_u + w \cdot (|C| - N - C(T_u)) \cdot l_u$
4) $\underline{\mathcal{T} = \{T_u\} \text{ and } T_u \text{ is an internal node } (k > 0)}$: Set
$$\begin{aligned}\xi_T^{res}(\mathcal{T}, k, K, N) = N \cdot l_u + w \cdot (N + C(T_u)) \cdot (K-1) \cdot l_u \\ + w \cdot (|C| - N - C(T_u)) \cdot k \cdot l_u \\ + \xi_T^{res}(\{T_v : p_v = u\}, k, K, N)\end{aligned}$$
5) $\underline{\mathcal{T} = \{T_1, \ldots, T_l\} \text{ and } l > 1\ (k > 0)}$: Set $\xi_T^{res}(\mathcal{T}, k, K, N)$ to be the *minimum* of:
   a) $\xi_T^{res}(\{T_1\}, 0, K, 0) + \xi_T^{res}(\{T_2, \ldots, T_l\}, k, K, N + C(T_1))$
   b) $\min_{0<i<k, j\in\{0,N\}} \{\xi_T^{res}(\{T_1\}, i, K, j) + \\ + \xi_T^{res}(\{T_2, \ldots, T_l\}, k - i, K, N - j)\}$
   c) $\xi_T^{res}(\{T_1\}, k, K, N + \sum_{u=2}^{l} C(T_u)) + \\ + \sum_{u=2}^{l} \xi_T^{res}(\{T_u\}, 0, K, 0)$

Using induction on the number of HST nodes in $\mathcal{T}$, we can prove that the above set of dynamic-programming rules correctly computes the optimal value of $\xi_T^{res}(\mathcal{T}, k, K, N)$. The induction step is based on the following two cases.

1) $\mathcal{T} = \{T_u\}$ *and* $T_u$ *is an internal node*: In this case, if $k = 0$, then the cost of the edge $(u, p_u)$ is simply $C(T_u) \cdot l_u$ (since every client in $T_u$ is assigned to a reflector outside $T_u$) and, by the induction hypothesis, $\sum_{v:p_v=u} \xi_T^{res}(T_v, 0, K, 0)$ is the cost of edges associated with the subtrees rooted at $T_u$'s children. If, on the other hand, $k > 0$, then (since at least one reflector is chosen in $T_u$) no clients in $T_u$ are assigned to reflectors outside $T_u$. Thus, the cost of the edge $(u, p_u)$ is $N \cdot l_u + w \cdot (N + C(T_u)) \cdot (K - k) \cdot l_u + w \cdot (|C| - N - C(T_u)) \cdot k \cdot l_u$, which is due to the outside clients assigned to reflectors in $T_u$ and the connection costs between reflectors within $T_u$ and outside of $T_u$. Finally, since the $k$ reflectors and $N$ outside clients are distributed among the subtrees of $T_u$, the cost of edges within $T_u$ is $\xi_T^{res}(\{T_v : p_v = u\}, k, K, N)$. Cases (2) and (4) above handle these two scenarios for $k = 0$ and $k > 0$, respectively.
2) $\mathcal{T} = \{T_1, \ldots, T_l\}$ *and* $l > 1$: In this case, we can partition $\mathcal{T}$ into two subsets $\{T_1\}$ and $\{T_2, \ldots, T_l\}$. By Lemma 8, in the optimal solution, the $N$ external clients are assigned to a *single* reflector either in $\{T_1\}$ or $\{T_2, \ldots, T_l\}$; thus, we only need to consider only two possibilities (0 and $N$) for the number external clients assigned to each subset. Further, the $k$ reflectors can be arbitrarily distributed among $\{T_1\}$ and $\{T_2, \ldots, T_l\}$. Thus, considering all possibilities for distributing $k$ and $N$ between the two subsets of $\mathcal{T}$, and selecting the one that results in minimizing the total cost guarantees that the resulting value for $\xi_T^{res}(\mathcal{T}, k, K, N)$ is optimal (Case (5b)).
   Note that the two extremes, when zero or all $k$ reflectors are chosen from $T_1$, need to be handled separately (Cases 5(a) & 5(c)). In the former case, all $N$ external clients must be assigned to $\{T_2, \ldots, T_l\}$, including clients from $T_1$, since no reflector is chosen in $T_1$ and clients in $T_1$ must be assigned to some reflector in $\mathcal{T}$ (remember that $k > 0$). Similarly, when all $k$ reflectors are assigned in $T_1$, all $N$ external clients as well as all the clients in $\{T_2, \ldots, T_l\}$ must be assigned to $T_1$. (When a non-zero number of reflectors is assigned to both subsets, then all internal clients are assigned to internally-chosen reflectors.)

While computing the optimal value for the cost function $\xi_T^{res}()$ as described above, we can also discover the corresponding optimal set of reflectors $R$ using standard dynamic-programming techniques. The details can be found in the full paper [9].

**Time Complexity.** Our dynamic-programming algorithm is based on the recursive relationships defined above and has a worst-case time complexity of $O(n^4|C|)$. To see this, note that there are $O(n^3|C|)$ different combinations of parameter values for $\xi_T^{res}()$. The 2-HST $T$ contains at most $2n$ nodes and, assuming a fixed order on the child subtrees of each internal node, there are at most $O(n)$ possibilities for $\mathcal{T}$. Furthermore, $k, K \leq n$ and $N \leq |C|$. Finally, since computing $\xi_T^{res}()$ for a specific choice of parameter values may involve $O(n)$ steps in the worst case (Case (5b)), the worst-case time complexity of our dynamic-programming algorithm is $O(n^4|C|)$.

Note that mapping the network graph $G = (V, E)$ to the 2-HST $T$ using Bartal's results requires only $O(n(n + m))$ time, which is clearly dominated by the complexity of our dynamic-programming step. Thus, the overall time complexity of our combinatorial WRS approximation algorithm (denoted by 2HST+DP) is $O(n^4|C|)$.

## V. Experimental Study

In this section, we present results from a simulation-based experimental study designed to test the effectiveness of our proposed reflector-configuration algorithms for both the unweighted (URS) and the weighted (WRS) cost objectives. Our study focuses on our *combinatorial* approximation algorithms for URS and WRS since they are most likely to be of interest in practice, where solutions to large-scale LPs can prove to be inordinately expensive. Our goal is to demonstrate that our algorithms are not only theoretically sound but also give significant benefits over naive solutions in practice for a variety of AS topologies.

**Experimental Testbed.** We present simulation results for AS topologies generated using the following two models. For each model, the number of nodes $n$ is varied between 50 and 200, and the set of clients coincides with the set of all nodes (i.e., $V = C$).

- *Waxman model:* The Waxman model [12] is a popular topology model for networking research (e.g., [13]) that allows the generation of different network structures by varying two key parameters: (1) $\alpha$, that controls the density of short edges in the network; and (2) $\beta$, that controls the average node degree. Edge weights in the final Waxman graph are assigned using a uniform distribution.
- *TreeClusters model:* Our TreeClusters model is intended to model topologies with router clusters that, we believe, occur naturally in large ISP networks. Briefly, this model distributes routers into a number of tree-shaped clusters whose roots are connected in a full mesh with edges of higher weight (i.e., "long-haul" lines). The key parameters for our TreeClusters model include: (1) the number of clusters, (2) the branching factor for the tree shapes, (3) the ratio of the edge weights between inter-cluster and intra-cluster edges, and (4) the distribution of nodes to clusters.

We compare the performance of our combinatorial approximation algorithms for URS (i.e., URSviaBFL) and WRS (i.e., 2HST+DP) with that of a "naive" randomized strategy (termed NAIVE). Briefly, NAIVE works by selecting $k$ routers in the AS-network at random to serve as reflectors for each value of $k = 1, \ldots, |V|$, and choosing the reflector configuration with minimum (URS or WRS) cost. (Our experiments have shown the corresponding costs for the full client-client I-BGP mesh to be *one or two orders of magnitude* worse than those of our algorithms [9], so we do not consider the full-mesh approach further.) In the remainder of this section, we present our results for networks with $n = 100, 150$ nodes, the $\alpha$ parameter and edge-weight distribution for Waxman graphs fixed at $0.1$ and `uniform(1,60)`, respectively, and specific choices for other model parameters; similar results were obtained for other parameter settings. To minimize the effects of randomization, we ran our simulations 10 times with different random seeds, and compared the *best* costs obtained for both NAIVE and our algorithms.

**Experimental Results.** Table II presents the results of our simulations for URS over Waxman graphs with 150 nodes. The first column in the table represents the average node degree in the generated Waxman network (which increases with larger values of $\beta$), and the final column gives the relative cost improvement of our URSviaBFL algorithm over NAIVE. Our results indicate that our algorithm clearly outperforms naive random reflector selections, offering relative cost benefits as high as 38%. As expected, the benefits of URSviaBFL decrease as the connectivity (i.e., average node degree) in the network increases: for very "tightly-connected" ASes choosing reflector servers intelligently does not make much of a difference, since all routers are within a short distance of each other.

| Avg. Node Degree | NAIVE Cost | URSviaBFL Cost | Relative Benefit |
|---|---|---|---|
| 2.00 | 29875 | 18528 | .379 |
| 2.52 | 14547 | 10304 | .291 |
| 4.79 | 6424 | 5256 | .181 |

TABLE II
URSviaBFL vs. NAIVE for Waxman Graphs, $n = 150$, $\alpha = 0.1, \beta \in \{0.1, \ldots, 0.4\}$

.

A small subset of our results for the URS problem with 100-node graphs generated using the TreeClusters model is depicted in Table III. (The "Cluster Distance" column gives the ratio of inter-cluster to intra-cluster edge weights.) The results shown are for the case of *asymmetric* tree-cluster sizes, where a randomly chosen cluster is populated with 50 nodes and all other clusters are of (approximately) equal size; the results for symmetric clusters are qualitatively similar. Clearly, our URSviaBFL algorithm offers substantial performance benefits (as high as 62%) relative to NAIVE, in terms of overall URS cost.

Finally, Table IV depicts a subset of the results of our simulation-based comparison of the 2HST+DP and

| No. of Clusters | Branching Factor | Cluster Distance | Relative Benefit |
|---|---|---|---|
| 6 | 3 | 20 | 0.584 |
| 6 | 4 | 20 | 0.571 |
| 6 | 3 | 100 | 0.473 |
| 6 | 4 | 100 | 0.619 |

TABLE III

URSviaBFL vs. naive for TreeClusters Graphs, $n = 100$, Asymmetric Clusters

.

naive algorithms for WRS over 100-node TreeClusters graphs with asymmetric clusters. (Due to space constraints, the corresponding table for Waxman graphs can be found in [9].) Our 2HST+DP algorithm emerges as the clear winner, offering relative WRS-cost benefits as high as 48% over naive. Note that making the clusters more "compact", by either increasing the cluster distance (i.e., the ratio of inter-cluster to intra-cluster edge weights) or increasing the branching factor of the tree clusters, typically increases the relative benefits of our 2HST+DP algorithm. The same observation holds for the number of clusters, as increasing the number of clusters, in general, implies higher client-reflector and reflector-reflector WRS costs for uninformed reflector selections.

| No. of Clusters | Branching Factor | Cluster Distance | Relative Benefit |
|---|---|---|---|
| 8 | 3 | 20 | 0.208 |
| 12 | 3 | 20 | 0.261 |
| 8 | 5 | 20 | 0.227 |
| 12 | 5 | 20 | 0.279 |
| 8 | 3 | 100 | 0.222 |
| 12 | 3 | 100 | 0.380 |
| 8 | 5 | 100 | 0.470 |
| 12 | 5 | 100 | 0.481 |

TABLE IV

2HST+DP vs. naive for TreeClusters Graphs, $n = 100$, Asymmetric Clusters

.

## VI. Conclusions

In this paper, we have proposed the first principled, algorithmic approach for designing effective route-reflector configurations for AS networks. We have presented $\mathcal{NP}$-hardness results that establish the intractability of these configuration problems, and proposed several polynomial-time approximation algorithms (based on LP-rounding and combinatorial techniques) with guaranteed (constant-factor or logarithmic) bounds on the quality of the approximate solution. The effectiveness of our algorithms has been verified through a preliminary simulation evaluation.

## References

[1] J. Stewart, *"BGP4 Inter-Domain Routing in the Internet"*, Addison-Wesley Networking Basics Series, 1999.

[2] Y. Rekhter and T. Li, "A Border Gateway Protocol (BGP-4)," Internet RFC-1771, Mar. 1995.

[3] T.G. Griffin and G. Wilfong, "An Analysis of BGP Convergence Properties," in *Proc. of ACM SIGCOMM*, Sept. 1999.

[4] Rohit Dube, "A Comparison of Scaling Techniques for BGP," *SIGCOMM Computer Communication Review*, vol. 29, no. 3, July 1999, (See also addendum in October 1999 issue.).

[5] T. Bates and R. Chandra, "BGP Route Reflection: An Alternative to Full Mesh IBGP," Internet RFC-1966, June 1996.

[6] D. S. Hochbaum, *"Approximation Algorithms for NP-Hard Problems"*, PWS Publishing Company, 1997.

[7] K. Jain and V. V. Vazirani, "Approximation Algorithms for Metric Facility Location and $k$-Median Problems Using the Primal-Dual Schema and Lagrangian Relaxation," in *Proc. of the Annual Symposium on Foundations of Computer Science (FOCS)*, 1999.

[8] Y. Bartal, "On approximating arbitrary metrics by tree metrics," in *Proc. of the Annual ACM Symposium on the Theory of Computing (STOC)*, 1998.

[9] Y. Breitbart, M. Garofalakis, A. Gupta, A. Kumar, and R. Rastogi, "On Configuring BGP Route Reflectors," Tech. Rep., Bell Labs, July 2001.

[10] J. H. Lin and J. H. Vitter, "$\epsilon$-approximations with minimum packing constraint violation," in *Proc. of the Annual ACM Symposium on the Theory of Computing (STOC)*, 1992.

[11] M. Charikar, C. Chekuri, A. Goel, and S. Guha, "Rounding via trees: deterministic approximation algorithms for group steiner trees and $k$-median," in *Proc. of the Annual ACM Symposium on the Theory of Computing (STOC)*, 1998.

[12] B. M. Waxman, "Routing of multipoint connections," *IEEE Jrnl. on Selected Areas in Communications*, vol. 6, no. 9, Dec. 1988.

[13] Y. Breitbart, C.Y. Chan, M. Garofalakis, R. Rastogi, and A. Silberschatz, "Efficiently Monitoring Bandwidth and Latency in IP Networks," in *Proceedings of IEEE INFOCOM'2001*, Apr. 2001.