Bandwidth Sharing Network Design for Multi-class Traffic

MohammadTaghi Hajiaghayi[‡], Li (Erran) Li[†], Vahab S. Mirrokni[⋆], and Marina Thottan[†]

***Abstract*— With the increasing commercial interest in supporting voice and multimedia services over the IP network there is a need for bandwidth guaranteed services. For example, guaranteeing the peak demand of VoIP traffic entails high costs in terms of bandwidth reservation requirements. To effectively make use of the reserved peak bandwidth, it is imperative that this bandwidth is shared with best effort data traffic during non peak periods. In this paper, we formulate this bandwidth sharing network design problem. Our goal is to minimize the total cost of bandwidth reservation while satisfying (1) the peak demand for real time traffic, and (2) the average demand of both real time and best effort data traffic. We show that, the problem is polynomially solvable if we do not restrict the number of paths used. It is strongly NP-hard if we use only one path between any pair of nodes. We present a simple 2-approximation algorithm and through simulation studies show that this algorithm can be further improved using a local search heuristic. Our simulation results also show that sharing significantly reduces the total bandwidth reservation costs and our local search heuristic can find a solution which is very close to or equal to the optimal in most cases.**

Keyword —Mathematical programming, optimization, Graph theory, Combinatorics

I. Introduction

To provide QoS assurance for real time multimedia traffic, Multi-Protocol Label Switching (MPLS) label switched paths (LSPs) with QoS (Quality of Service) constraints are set up between pairs of network end-points. The most commonly used QoS constraints are in the form of average or peak bandwidth guarantees per LSP [6], [17]. Research on QoS-assured design has focused on two areas: (1) the protocol aspects[6] and (2) on algorithms that minimize bandwidth reservation given traffic demands between any pair of end points [7], [1]. However, *these algorithms only consider aggregate traffic demands and do not distinguish among the different classes of traffic that may be specified in the traffic demand matrix*.

Recently there is an increased commercial interest to support voice and other multimedia services on IP networks. Companies like Vonage and 8×8 are already setting up Voice Over IP (VoIP) networks over the public Internet. Major network service providers such as Verizon [13] have also deployed voice and other multimedia applications over their networks. The support for these varied types of network services will require different levels of QoS guarantees for the different application traffic types. For example, voice traffic has real time requirements and therefore it is important that voice bandwidth reservations in the network satisfies the peak demand. Meanwhile, support for regular data traffic may only need to meet the average traffic demand. Identifying optimal bandwidth reservation needs for supporting different traffic types is important for both application customers as well as network service providers. From a customer's perspective, it is necessary to minimize the bandwidth reservation cost while for a network service provider it is important to optimize the network utilization.

Studies in [16], [23] have shown that, it is possible to police peak rate, while trying to enforce the mean rate is difficult. As a result, charging based on peak rate is used in practice. For example, it is pointed out in [9] that, many Internet Service Providers (ISPs) use percentile-based charging. In this model, an ISP records the traffic volume a user generates during every t time interval (e.g. $t = 5$ min). At the end of a complete charging period, the ISP uses the qth-percentile traffic volumes among all the t time intervals for charging (e.g. $q = 95$). However, the peak period traffic can be very small due to the bursty nature of network traffic. Furthermore, the average bandwidth requirement may be very small compared to the peak rate. Thus using the peak rate charging model, the cost incurred can be much higher than what is really needed if the traffic is constant bit rate instead. Therefore, the bandwidth reservation both in terms of the cost of reservation as well as network utilization must be

MohammadTaghi Hajiaghayi is with Department of Computer Science at Carnegie Mellon University. Part of this work has been done while Mohammad Taghi Hajiaghayi was in Microsoft Research and Computer Science and Artificial Intelligence Laboratory at MIT. Email: hajiagha@theory.csail.mit.edu

Vahab S. Mirrokni is with Computer Science and Artificial Intelligence Laboratory at MIT. Supported in part by ONR grant N00014-05-1-0148. Email: mirrokni@theory.csail.mit.edu

Li (Erran) Li, and Marina Thottan are with the Center for Networking Research at Bell Labs. Email:{erranlli,marinat}@dnrc.bell-labs.com

made such that the unused bandwidth during non peak periods can be utilized by best effort data traffic. Hence to provide satisfactory service to both voice and data traffic, and to minimize bandwidth reservation (or cost), we want the total bandwidth reservation to satisfy the peak demand of real time applications such as voice and the average demand for voice and data.

In this work we assume that network routers implement QoS mechanisms such as those proposed in the DiffServ model [3]. We remark that VoIP equipments such as Juniper's VoIP products [11] has the DiffServ capability. This capability is deemed essential to provide carrier grade VoIP services and their VoIP platforms have been deployed in many networks [12]. The QoS mechanisms in DiffServ can be used to allocate the required bandwidth for different traffic classes. It also dynamically reduces the bandwidth allocation to best effort traffic in order to satisfy the demand of real time (Expedited Forwarding Class) traffic. Note that, best effort data traffic is elastic in nature since TCP will adjust the sending rate to the available network bandwidth. In this paper we refer to the peak demand specification of real time traffic as the *type 1* demand and the average demand of real time and best effort traffic as the *type 2* demand.

The bandwidth sharing problem described above is relevant to both network service providers as well as for application service providers. However, for the purpose of this paper we take the perspective of a network service provider such as Verizon. Service providers would like to design a network that can support the emerging multimedia services over their IP infrastructure. We assume that the cost of building such a network is equal to the total cost of bandwidth reservations. We must now solve the abstract problem of minimal-cost network design jointly for both real time and best effort traffic. A feasible solution to this problem is one that can satisfy the demands of both type 1 and type 2 traffic.

The traffic demand in a provider network can be either expressed as two matrices or two vectors. The two traffic matrices specify the amount of type 1 and type 2 demands among any given pair of edge nodes. The two vectors specify the amount of incoming and outgoing traffic of type 1 and type 2 at each of the edge nodes respectively. In this paper we primarily consider the point-to-point traffic reservation model. The path between any two edge routers of the network can be a single path or multiple paths. We refer to the former case as the unsplittable version; and the latter as the splittable version of the bandwidth sharing problem. If routing from all other end points to any given end point forms a tree, we refer this specific unsplittable flow version as the *confluent* flow version [4], [1]. Confluent flow simplifies MPLS routing for a VPN-based network design since the paths to a single destination form a tree. We consider both the capacitated and the uncapacitated version of the confluent flow problem.

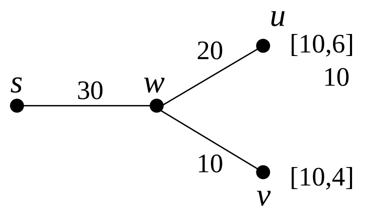
Our key contributions are as follows:

- To the best of our knowledge this work presents *the first formulation and study of the problem of bandwidth sharing network design*. Given the bursty nature of network traffic, the peak rate charging model and the elastic nature of best effort data traffic, bandwidth sharing between real time traffic and best effort data traffic is appealing to both customers (e.g. ASPs such as VoIP companies) and service providers for its cost effectiveness.
- We show that, the problem can be solved optimally if we do not restrict the number of paths used.
- We show that, the problem is strongly NP-hard in the confluent flow version. We give a simple approximation algorithm which costs at most two times the minimal cost. We further propose a local minimum heuristic to improve on this simple algorithm. If the network is directed, we show that, there does not exist any polynomial algorithm that in the worst case achieves a cost less than $3/2$ times the optimal.
- We evaluate the effectiveness of our algorithms to reduce the total bandwidth reservation through extensive simulation studies. Our results show substantial cost reduction when compared with the simple bandwidth reservation algorithms. We also evaluate the impact of the number of demands and the connectivity of the network on the performance of our algorithms.

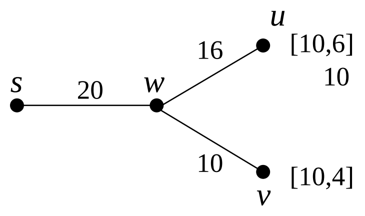
The paper is organized as follows: Section II provides the motivation and the problem formulation. This is followed by Section III on splittable and confluent bandwidth reservation studies. In Section IV we present the local minimum heuristic that improves on the simple algorithm. Section V describes the extension of our work for general cost functions and for sharing among multiple demands. Simulation results and a discussion on the heuristic is provided in Section VI and VII respectively. Related work is provided in Section VIII. We conclude our paper with a discussion on future work in Section IX.

II. Motivation and Problem Formulation

We illustrate the benefit of bandwidth sharing using the example in Figure 1. Real time traffic demand is represented by [peak rate, average rate] and best effort traffic demand is represented only by average rate. Node u has a peak demand of real time traffic of 10 units of bandwidth, average demand for real time traffic of



(a) No bandwidth Sharing Case: total bandwidth reserved 60 units



(b) Optimal Bandwidth Sharing Case: total bandwidth reserved 46 units

Fig. 1. Benefit of Bandwidth Sharing

6 units of bandwidth; this is represented as [10,6] in Figure 1. Node u also has an average demand of 10 for best effort data traffic; this is represented as 10 in Figure 1. Node v has a peak demand for real time traffic of 10 units and average demand of real time traffic of 4 units; this is represented as [10,4] in Figure 1. If there is no sharing as illustrated in Figure 1-a, link (u, w) must reserve 20 units of bandwidth to satisfy both peak demand of real time traffic and average demand of data traffic; link (v, w) must reserve 10 units of bandwidth to satisfy peak demand of real time traffic; link (s, w) must reserve 30 units of bandwidth to satisfy both peak demand of real time traffic and average demand of best effort traffic from node u and v. If we allow sharing, the situation is much different as illustrated in Figure 1-b. All we need is to reserve bandwidth such that, the reservation satisfies both peak demand of real time, and average demand of real time and best effort data. It is not necessary to reserve the sum of the peak demand for real time and the average demand of best effort data traffic. Thus, link (u, w) needs to reserve only 16 units of bandwidth; link (v, w) needs to reserve a bandwidth of 10 units; and link (s, w) needs to reserve only 20 units of bandwidth. Note that, the naive solution which is to take the sum of the reservation required for traffic on incoming links is not optimal. If this naive solution is used, link (s, w) will need to reserve 26 units of bandwidth.

We stress that the average demand assumed in this paper are not the average over long time periods. The average is on small time intervals such as $t = 5min$. Best-effort traffic has lower priority and will not be transmitted if the real-time traffic queue is not empty. Best-effort traffic is statistically multiplexed onto the residual bandwidth of real-time traffic.

We formalize the above problem as a variant of the minimum cost network design problem.

Definition 1: Let $G(V, E)$ be a network of n nodes and m links. For each link e of the network, we are given a per-unit cost, $c(e)$. This is the cost of reserving a unit of bandwidth on link e. Also we are given a set of k demands. Each demand corresponds to a source-destination request and is defined by a source s_i and a destination t_i. For each demand i, we are given bandwidth requests for two types of traffic, i.e, $d_{i,1}$ for type 1 and $d_{i,2}$ for type 2 traffic. Our goal is to reserve enough bandwidth such that we can satisfy the requirements for each traffic class. The *minimum bandwidth sharing network design problem* (Min-BSND) is to find the minimum cost subset of edges that can satisfy both classes of traffic demands.

In the following section we consider several variants of the Min-BSND problem. In the splittable Min-BSND problem, we want to find a set of paths for each demand. In the unsplittable Min-BSND problem, we need to assign one path to each demand. In the confluent Min-BSND problem, we need to find a flow such that for all the flows with the same destination traversing a given node, that given node has one outgoing edge for these flows. In other words, for each destination we want to find a tree to send the flow on that tree. All these variants can be implemented in a network with MPLS capabilities. Note that the confluent flow case has the minimal overhead in terms of setting up LSPs.

III. Splittable and Confluent Bandwidth Reservation

In this section, we study the minimum cost bandwidth sharing network design problem for splittable and confluent cases. First, we observe that if there exists only one class of demand, then in the splittable and the confluent variant of the problem, the shortest path tree is an optimum solution and can be found in polynomial time. The following example shows that with two classes of requests, the optimum solution is not necessarily a tree (and thus not necessarily a shortest path tree).

Example 1: As illustrated in Figure 2, graph G is an undirected graph with 6 vertices $\{v_1, \ldots, v_6\}$ and edge set $\{(v_2, v_1), (v_3, v_1), (v_4, v_2), (v_5, v_3), (v_5, v_4), (v_6, v_5)\}$. The demand pairs on vertices v_1 to v_6 are $(0, 0)$, $(1, 0)$, $(0, 1)$, $(1, 0)$, $(0, 1)$, $(1, 1)$ that they all want to send the demands to vertex 1 (i.e., the root). The first demand in the demand pair is the demand of type-1 traffic and the second is the demand of type-2 traffic. Recall that we define the peak rate of real-time traffic as type-1 traffic, and the sum of the average rate of

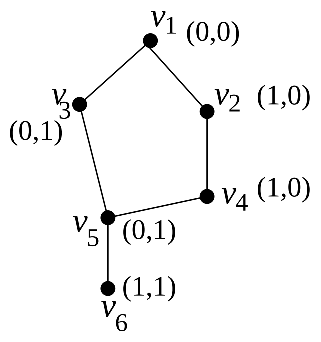


Fig. 2. Optimal solution is not necessarily a tree

both real-time and best-effort traffic as type-2 traffic. The per-unit cost of capacity on each link is 1. The optimum network design is to buy 2 units of capacity on edge (v_2, v_1), 1 unit on edge (v_4, v_2), 1 unit on edge (v_5, v_4), 2 units on edge (v_3, v_1), 1 unit on edge (v_5, v_3), and finally 1 unit on edge (v_6, v_5). In this case, nodes 2, 3 and 4 send their flows directly to 1 (via their shortest path). Node 6 sends the flow via node 5 and then node 3 to 1. Finally node 5 sends its flow via node 4 and then node 2 to 1. The cost of this solution is 8. It is not hard to check that the cost on any shortest path tree in this example is greater than 8 (the main reason is that both nodes 5 and 6 try to send their flows from edge (v_5, v_3) in the shortest path tree, and thus we need capacity 2 instead of 1 on this edge.

A. Splittable Min-BSND

Although the above example shows that the Min-BSND problem cannot be solved using the combinatorial algorithms for shortest path tree, we observe that the splittable Min-BSND problem, can be formalized as a linear program and can be solved in polynomial time. The linear programming formulation is as follows:

Our objective is:

$$Minimize \sum_{e=(u,v)\in E(G)} c_e X_e \tag{1}$$

subject to

$$\sum_{e=(u,v)\in E(G)} f^i_{e,t} - \sum_{e=(v,u)\in E(G)} f^i_{e,t} = 0,$$
$$\text{for each } i = 1 \cdots k, u \neq s_i, t_i \in V(G), t = 1,2 \tag{2}$$

$$\sum_{e=(s_i,v)\in E(G)} f^i_{e,t} - \sum_{e=(v,s_i)\in E(G)} f^i_{e,t} = d_{i,t},$$
$$\text{for each } i = 1 \cdots k, t = 1,2 \tag{3}$$

$$\sum_{e=(t_i,v)\in E(G)} f^i_{e,t} - \sum_{e=(v,t_i)\in E(G)} f^i_{e,t} = -d_{i,t},$$
$$\text{for each } i = 1 \cdots k, t = 1,2 \tag{4}$$

$$X_e \geq \sum_{1\leq i\leq k} f^i_{e,t},$$
$$\text{for each } e \in E, t = 1,2 \tag{5}$$

where c_e is the per unit bandwidth cost; X_e is the total amount of bandwidth reserved on edge e; $f^i_{e,t}$ is the amount of type t flow between the ith source and destination pair which goes through e. The first condition is the one that requires preservation of flow for each node except sources and sinks. The second condition asks for sinks to receive the desired amount of flow and the third condition asks for sources to send the desired amount of flow. Finally, the last condition says the total amount of flow that we send via an edge e for all commodities should be less that X_e.

B. Confluent Min-BSND

Here, we consider the other variant of the problem which is the confluent Min-BSND problem. Despite the fact that the splittable case can be solved optimally in polynomial time, the unsplittable (confluent or not) Min-BSND is NP-Hard.

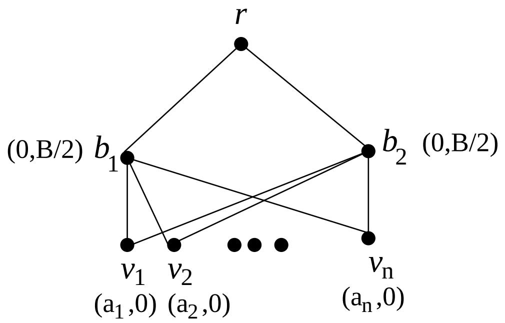


Fig. 3. Illustration of the Reduction From Partitioning Problem.

Theorem 1: The unsplittable Min-BSND problem is NP-Hard.

Proof: We give a reduction from the partitioning problem. An instance of the partitioning problem is as follows: Given n numbers $a_1, a_2, \ldots, a_n$ such that $\sum_{i=1}^n a_i = B$, find a subset of a_i's that sum to $\frac{B}{2}$. Given an instance of the partitioning problem, we construct the following instance of the Min-BSND problem. Put a vertex v_i corresponding to each element a_i. The demand on vertex v_i is $(a_i, 0)$. Put two other vertices, b_1 and b_2 in the graph. The demands on these two vertices are $(0, \frac{B}{2})$. The only other vertex is the root r. The edge set of the graph is $\{(b_1, r), (b_2, r)\} \cup \{1 \leq i \leq n | (v_i, b_1), (v_i, b_2)\}$. Figure 3 illustrates this construction. The per-unit cost of all edges is one. Now it is easy to see that there exists a solution with bandwidth reservation equal to $2B$ for this instance if and only if there exists a subset of elements that sum to $\frac{B}{2}$. ■

The following theorem shows that the problem is even hard to approximate up to a constant factor.

Theorem 2: If the graph is directed, i.e., the per-unit cost of edges is asymmetric, then the confluent Min-BSND problem is not approximable within a factor better than $3/2$, unless P=NP.

Proof: We use a transformation from the following famous NP-complete problem: Given a directed graph H and four distinct vertices u, v, u', v', we want to know whether there are two edge-disjoint paths in H, one from u to u' and the other from v to v'. We call this problem, 2DIRPATH. We construct a graph G whose skeleton is H, but we also add another vertex r and directed edges $v'r$ and $u'r$ to the graph. We assume the per-unit cost of edges $v'r$ and $u'r$ are one and the per-unit cost of all other edges of G is zero. Suppose that u and v' have $(0, 1)$ type demands, u' and v have $(1, 0)$ type demands, and all other vertices have $(0, 0)$ type demands. Finally, assume all the flow should be sent to r.

Now, we are ready to state the reduction. First, we observe that if H has two edge-disjoint paths from u to u' and from v to v', then u and u' can share the edge $u'r$, and v and v' can share the edge $v'r$ and pay only cost 1. Therefore, if there are two edge-disjoint paths in H, the total cost of all edges in the graph is 2. On the other hand, if there are not two aforementioned edge-disjoint paths, it is easy to see that either the flow from u should go to r via v' or the flow from v should go to r from u' in the confluent flow. Without loss of generality assume that the former case happens. In this case, we need to reserve at least two units of bandwidth on $v'r$ and one unit on $u'r$ (for the demand of u'). Thus the total cost is at least 3. Therefore, if there are not two edge-disjoint paths from u to u' and from v to v' in H, the cost is at least 3. It means there is no approximation algorithm for the directed Min-BSND, whose factor is better than $3/2 - \epsilon$, otherwise we can solve 2DIRPATH in polynomial time and thus P=NP. ■

Theorem 3 shows that actually the problem has a simple 2-approximation algorithm.

Theorem 3: The confluent Min-BSND problem has a 2-approximation algorithm.

Proof: The following is a simple 2-approximation algorithm for this problem: find a shortest path tree and buy enough bandwidth on the edges of this tree to satisfy the request of both types. To prove that this is in fact a 2-approximation, we note that the bandwidth required for each set of demands is a lower bound. Using these two lower bounds, we note that the cost of the solution with the above algorithm is at most the sum of these two lower bounds. This shows that the cost of this solution is at most twice the cost of the optimum solution. ■

Finally, it is worth mentioning that in the case of undirected graphs, we do not have any inapproximability result. For this case, we only know that the problem is strongly NP-hard. The proof of strongly NP-hardness is based on a reduction from 3-Partitioning. The reduction is very similar to the reduction in the proof of Theorem 1 and is omitted here. We can show that the integrality gap of the splittable linear program is greater than 2. Consider the following example. The vertex set, $V(G)$, of graph G is $\{r, v_1, v_2, \ldots, v_n, t\}$. The edge set is $\{1 \leq i \leq n | (v_i, r), (t, v_i)\}$. The demand on nodes $v_1, \ldots, v_n$ is $(1, 0)$ and the demand on node t is $(0, n)$. The per-unit cost of edges (v_i, r) is one and per-unit cost of edges (t, v_i) is zero. Then the bandwidth reservation in the optimal solution of the splittable problem is n, but the bandwidth reservation in the optimal confluent tree is $2n$. This shows that the integrality gap of the aforementioned linear programming formulation is at least 2. Thus we cannot hope to get a better than 2-approximation for this problem based on this linear programming relaxation. Note that, our reduction uses non-uniform edge cost. The result is easily extended to uniform edge cost case if we replace the edge from v_i to r with a sufficiently long path.

IV. A Heuristic based on Local Minimum

As mentioned in Section III, for Min-BSND, designing algorithms with approximation factor better than that of finding the shortest path tree seems quite hard (for the directed case almost impossible.) However, in practice, we always have the option of using some heuristics to obtain better experimental results (In our case, we will have the approximation factor of the shortest path tree in the worst case). To this end, we consider a heuristic for the confluent Min-BSND as follows.

First we construct the shortest path tree. We note that given any tree to route the confluent flow of our demands, we can easily compute the cost of each edge in a Min-BSND optimal solution by obtaining the cost of each edge for each demand type and take the maximum. Thus we can compute our objective function, i.e., the total bandwidth reservation, for the given tree (see Figure 4, for the formal description of the Algorithm). Let C be the total amount of bandwidth for the shortest path tree. Now, we run the following heuristic. We consider each node of the tree at each round. Then we take the shortest path from that node to each of the other nodes in the tree except its descendants at each step, and we replace the edge connecting the leaf to its parent by the aforementioned path. We again compute the total bandwidth of the resulting tree and we call it

```
Algorithm **Tree-Bandwidth**

Input: tree $T = (V, E)$, root $r$, demands $(d_1^1, d_1^2)$,
$(d_2^1, d_2^2)$, …,$(d_k^1, d_k^2)$ at nodes $s_1$,$s_2$,…, $s_k$
let $C = 0$
for each edge $e \in E(T)$
    let $c_e^1 = c_e^2 = 0$
    for each $1 \le i \le k$
        if the path from $s_i$ to $r$ in $T$ goes through $e$
            add $d_i^1$ to $c_e^1$
            add $d_i^2$ to $c_e^2$
    add $\max(c_e^1, c_e^2)$ to $C$
report $C$ as the output
end
```

Fig. 4. A formal description of Tree-Bandwidth algorithm

C'. Now, instead of the original tree, we take the tree which maximizes $C^* = C - C'$, if $C^* > 0$; we stop the algorithm by reporting the current tree as the output otherwise. We note that in the latter case, the current tree is the local minimum for the problem. The formal description of the algorithm is depicted in Figure 5. Though this is a very simple hill-climbing algorithm, it works quite well in practice, as shown in Section VI.

Finally, it is worth mentioning that the above heuristic can be easily generalized to the case in which we have multiple demands.

V. Extensions of the problem

In this section, we consider different variants and extensions of the problems, study the complexity of these problem, and report our results on them.

A. General Cost functions

In the Min-BSND problem, we consider the cost of reserving bandwidth as a linear function of the amount of bandwidth that we reserve on a link. Here we consider other cost functions.

Capacitated Min-BSND Problem

The first cost model is the *Capacitated Min-BSND* problem. In this model, there is a capacity constraint on each link and the reserved bandwidth on this link cannot exceed this capacity. In the splittable case, by adding the capacity constraint, we can extend the above linear programming formulation for this variant. Thus the problem can be solved in polynomial time. On the other hand, the confluent Capacitated Min-BSND problem is not approximable within any approximation factor. The following theorem proves this fact:

```
Algorithm **Heuristic**
Input: $G(V, E)$, root $r$, demands $(d_1^1, d_1^2)$, $(d_2^1, d_2^2)$,
…,$(d_k^1, d_k^2)$ at nodes $s_1$,$s_2$,…, $s_k$
// compute the shortest path tree
let tree $T = (\{r\}, \emptyset)$
for each $1 \le i \le k$
    find the shortest path $P$ from $s_i$ to tree $T$
    add all edges and vertices of $P$ to $T$
let $C_T$ be the result of Tree-Bandwidth
    $(T, (d_i^1, d_i^2)_{i=1}^k, s_1, s_2, \ldots, s_k)$
// Find the local minimum tree $T'$
let $T' = T$ and $C_{T'} = C_T$
repeat
    let $T = T'$ and $C_T = C_{T'}$
    for each $u \in T$
        for each $v \in T$ that is not a descendent
        of $u$ in T
            let $P$ be the shortest path from $u$ to $v$ in $G$
            if $P$ intersects $T$ in only $u$ and $v$
                obtain tree $T''$ by replacing the edge
                $(u, \text{parent}(u))$ by path $P$ in $T$
                let $C_{T''}$ be the result of Tree-Bandwidth
                $(T'', (d_i^1, d_i^2)_{i=1}^k, s_1, s_2, \ldots, s_k)$
                if $C_{T''} < C_{T'}$ let $T' = T''$ and $C_{T'} = C_{T''}$
until $C_T = C_{T'}$
report tree $T$ as the output
end
```

Fig. 5. A formal description of the local search heuristic algorithm

Theorem 4: Given capacity constraints on the links, the confluent Min-BSND problem cannot be approximated within any factor in polynomial time unless P=NP.

Proof: We know that in the presence of capacity constraints, checking if there exists a confluent flow respecting all capacity constraints is NP-Hard as the minimum congestion confluent flow problem is NP-Hard. In an instance of the minimum congestion confluent flow problem, we are given a graph G with capacities on its edges and we want to find a confluent flow tree with minimum congestion, i.e, a confluent flow with minimum maximum ratio of the bandwidth consumption over the capacity of the edges. Given an instance of the minimum congestion confluent flow problem, we can put per-unit cost zero on the edges of the original graph and put extra edges with infinite capacity and infinite per-unit cost from all the nodes to the root. After adding these edges, we have an instance of the Capacitated confluent Min-BSND problem. In this instance, there exists a solution with bandwidth reservation of cost zero if and only if the optimal congestion in the minimum congestion instance is less than 1. This shows that distinguishing between the bandwidth reservation of zero and nonzero is NP-Hard. Hence, approximating the bandwidth reservation within any factor is NP-Hard. ■

A more general variant of the Capacitated Min-BSND is when the cost function on edges is a general convex function. For this variant of the problem, the complexity of both splittable and confluent Min-BSND is the same as the Capacitated case. In the splittable Min-BSND problem, we can model the convex cost function by a convex programming in which we can minimize a convex function over a convex polytope (of linear constraints). This can be done in polynomial time using the interior point methods.

A more natural cost model for the bandwidth is the concave cost function. In this case the problem is NP-Hard even with only one demand type. The reduction comes from the *single-sink Rent-or-Buy* problem (see e.g. [24]). In this problem, each node v has a demand $d_v \geq 0$ and we have a root r. We want to route the demands from the vertices to the root. We can either *rent* capacity on an edge, the renting cost being proportional to the amount of capacity rented, or we can pay a one-time expense of M per unit length and *buy* unlimited capacity. In fact, it is easy to see that the solution to the problem should be always a tree with minimum cost. Now in *single-sink Rent-or-Buy*, of course, the cost model for the bandwidth is the concave cost function. This reduction shows that both splittable and confluent Min-BSND even with only one demand type are NP-hard.

B. Sharing among multiple types of demands

In the Min-BSND problem, we assumed that there are only two types of demands. A natural generalization of this problem is that, instead of two types of demands, we have p different classes of demands. The goal is to find a flow that can satisfy all classes of demands.

Again, for the splittable case, the LP can be extended to capture p classes of demands. The following theorem gives a $\min(O(\log(n)), p)$-approximation for this problem.

Theorem 5: There exists a polynomial time $\min($ $O(\log(n))$, $p)$-approximation for the Min-BSND problem.

Proof: The p-approximation is simple, since again finding a shortest path tree and buying enough bandwidth on the edges of this tree to satisfy the request of all types gives the desired algorithm. Now, we show how we can obtain $O(\log(n))$ approximation. We obtain such a logarithmic approximation factor based on Bartal's machinery [2] (or its slight improvement by Fakcharoenphol et al. [8]) for probabilistically embedding general metrics into tree metrics.

In [2], [8], it is shown that given a graph G, we can construct a family of trees $\mathcal{T} = \{T_1, T_2, \cdots, T_N\}$, where $N = O(n \log n)$, such that each vertex $v \in V(G)$ has a corresponding vertex in each T_i, $1 \leq i \leq N$,(in fact $V(T_i) = V(G)$) such that for each i, $d_{T_i}(u,v) \geq d_G(u,v)$ ($d_H(u,v)$ is the distance of u and v in H where the length of an edge is the cost of that edge.) In addition, there is a probability distribution on $\mathcal{T}$, i.e., there are probabilities $p_1, p_2, \cdots, p_N$ summing to one, such that $\sum_{i=1}^{k} p_i d_{T_i}(u,v) \leq O(\log n) d_G(u,v)$ for every $u, v \in V(G)$.

Now, our algorithm for Min-BSND is as follows. For each tree $T_i \in \mathcal{T}$, we consider the same (multiple) demands that we had in G. Since the paths in T_i are unique, we can solve the problem in polynomial time for T_i. Then we pick a tree T whose total cost is minimum among all trees in T. Now for each edge $e = (u,v)$ in T, we buy all edges in the shortest path of u and v in G for the amount of flow of e in T (in fact, we can apply some short-cuts in G, if it is necessary). Let C_T be the cost of solution in T and C_G be the cost of the corresponding solution in G (note that $C_T \leq C_G$). Also, let OPT_G be the cost of a minimum solution in G and OPT'_{T_i} be the cost of the corresponding mapping of the optimum solution of G in T_i (again we map each edge of the optimum solution in G to the corresponding path in T_i).

We are now ready to show that $C_G \leq O(\log n) OPT_G$ as follows:

$$\begin{aligned}
OPT(G) &\geq O(\log n) \sum_{i=1}^{N} p_i OPT'_{T_i} \\
&\geq O(\log n) \sum_{i=1}^{N} p_i C_{T_i} \\
&\geq O(\log n) \sum_{i=1}^{N} p_i C_T \\
&\geq O(\log n) C_T \geq O(\log n) C_G.
\end{aligned}$$

■

For directed graphs, we can improve the result of Theorem 2 when we have more classes of demands as follows:

Theorem 6: If the graph is directed, i.e., the per-unit cost of edges is asymmetric, then the confluent Min-BSND problem is not approximable within a factor better than $2 - 1/p$ when we have p different classes of demands, unless P=NP.

Proof: Again we use a transformation from 2DIRPATH introduced in the proof of Theorem 2. We construct a graph G whose skeleton is shown in Figure 6. We replace each vertex d_{ij}, $i > j$, with a copy of graph

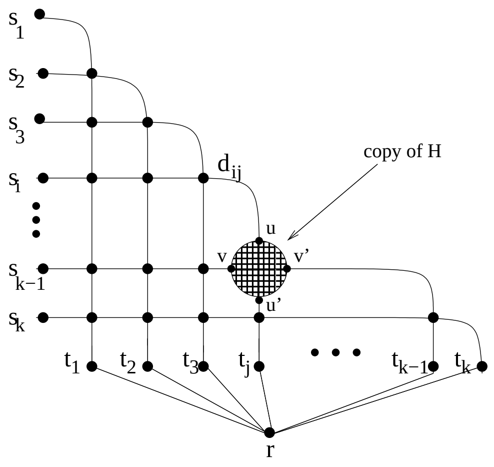


Fig. 6. The graph describing a reduction from 2DIRPATH to directed Min-BSND

H as shown in the Figure 6. Thus the vertex set of G contains r, s_i and t_i, $1 \leq i \leq k = p$, and all vertices of copies of the directed graph H which are placed instead of d_{ij}, $1 \leq j < i \leq k = p$. All edges of the graph not in a copy of H are directed from top to bottom and from left to right. In addition, we assume the per-unit cost of edges $t_i r$, $1 \leq i \leq p$, is one and the per-unit cost of all other edges including those in a copy of H are zero (we assume that only the edges shown in Figure 6 exist, and thus the per-unit cost of edges which do not exist are ∞.) We assume that each s_i, $1 \leq i \leq p$, has a vector of size p whose all elements are 1 except the i-th element which is 0 at its type demand, each t_i, $1 \leq i \leq p$, has a type demand which is the complement of s_i, and all other vertices are all-zero type demands. Again, assume all the flow should be sent to r confluently.

Now, we are ready to state the reduction. First, we observe that if H has two edge-disjoint paths as described, we can find edge-disjoint paths from s_i to t_i, $1 \leq i \leq p$, which carry a unit flow (the paths are $s_i - t_i$ paths in the skeleton except that instead of using $d_{i'j'}$ vertices, we use the edge-disjoint paths from u to u' and from v to v'). In this case, s_i and t_i, $1 \leq i \leq p$, can share the edge $t_i r$ and in total pay only cost 1. Therefore, if there are two edge-disjoint paths in H, the total cost of all edges in the graph is at most p. On the other hand, if there are not two edge-disjoint paths in H, when two flows from s_i and $s_{i'}$, $i \neq i'$, go to a copy of H, either they will mix (since the flow is confluent) or they keep their relative “planar” order. More precisely, since every two paths $s_i - t_i$ and $s_{i'} - t_{i'}$, $i \neq i'$, have an intersection in a copy of H, it is not possible to route the demand from s_i to t_i and from $s_{i'}$ to $t_{i'}$ simultaneously. It means in this case, we can only route one s_i to r through its corresponding t_i. In this case, since each t_i needs one unit for its demand, we need p unit of bandwidth reservation. For each demand from s_i which does not go to r through its t_i, we should reserve at least one extra unit of bandwidth. Thus the total cost is at least $p - 1 + p$. Therefore, if there are no two edge-disjoint paths from u to u' and from v to v' in H, the cost is at least $2p - 1$. It means there is no approximation algorithm for the directed Min-BSND, whose factor is better than $2 - 1/p$ unless P=NP. ■

C. Unsplittable Non-confluent Bandwidth reservation

Another interesting generalization of our problems in this paper is the problem in which we want to have unsplittable non-confluent bandwidth reservation. In this case each source has its own sink and the demand of each source should be routed via a single path to the corresponding sink, and the flows are not necessarily confluent after meeting each other. Note that here again sources can have more than one demand type. This generalized version simultaneously is the generalization of both the splittable version and the Unsplittable confluent version of our problem. This problem is very interesting especially from the theoretical point of view, and some of our results in this paper such as the algorithm for sharing among multiple types of demands work for this case. However, still more insight into this problem is needed to obtain deeper approximability/inapproximability results (Note that in this case, e.g., $2k$-approximation algorithm for k source-sink pairs and two demand types is trivial).

VI. Simulation Results

The performance of the linear program and the local tree search heuristic was studied on several different simulated topologies. We first consider known topologies that have been used in previous work. Most of these topologies are relatively small and provide an intuitive understanding of our framework. We also generated large topologies using BRITE [20]. The BRITE topology generator uses a Waxman model to generate flat topologies [25]. The model parameters used were $\alpha = 0.15$ and $\beta = 0.20$, where α captures the relationship between short and long edges and β measures the degree of connectivity in the network. We generated two sets of topologies, the first set had 100 nodes with 100, 200 and 400 bidirectional edges and the second set had 60 nodes with 120 bidirectional edges. The linear program was solved using the commercially available CPLEX solver [5]. We used CPLEX since it is based on simplex and automatically produces basic solutions for an LP. In our simulation scenarios, the traffic demands are unidirectional and directed to a single destination node. Each traffic demand comprises of two types of traffic

k	No Sharing	LP	Before LS	After LS	BW gain %	ρ %
2	2.1k	1.7k	2.1k	1.7k	19	0
5	3.7k	2.8k	3.7k	3.1k	16	11
10	6.7k	4.9k	5.1k	4.9k	27	0
15	9.5k	6.5k	7.4k	6.5k	32	0
20	13k	8.7k	9.3k	8.8k	32	1
25	16k	10.4k	10.6k	10.4k	35	0

TABLE I

BANDWIDTH RESERVATIONS USING LP AND THE LOCAL SEARCH HEURISTIC FOR KL TOPOLOGY; NUMBER OF NODES = 15; NUMBER OF EDGES = 28. K: NUMBER OF DEMANDS, BW GAIN IS W.R.TO NO SHARING, ρ: PERCENT BW GAIN W.R.TO LP

k	No Sharing	LP	Before LS	After LS	BW gain %	ρ %
2	29k	24k	29k	24k	17	0
5	70k	39k	41k	39k	44	0
10	140k	78k	85k	81k	42	4
15	186k	110k	110k	110k	41	0
20	244k	139k	139k	139k	43	0
25	311k	180k	180k	180k	42	0

TABLE II

BANDWIDTH RESERVATIONS USING LP AND THE LOCAL SEARCH HEURISTIC FOR US TOPOLOGY; NUMBER OF NODES = 8; NUMBER OF EDGES = 10. K: NUMBER OF DEMANDS, BW GAIN IS W.R.TO NO SHARING, ρ: PERCENT BW GAIN W.R.TO LP

classes: one with maximum bandwidth and the other with average bandwidth requirements. This simulation framework can also be extended for demands to multiple destinations. The demands are obtained from a uniform distribution over the range of [50, 400] bandwidth units. In each simulation scenario, we report the total bandwidth reservation obtained using the LP (demand between a given source and destination pair can go through multiple paths) and the bandwidth obtained in the confluent case (demand between a given source and destination pair must be routed in a single path) both before and after the local search (LS) heuristic is implemented. We also report the No sharing bandwidth reservation as a baseline comparison. The percentage bandwidth reduction from using LS heuristic when compared with the No Sharing case is referred to as BW gain. *Note that, this gain is at most 50% which means that the No Sharing bandwidth reservation is at most 2 times the optimal for two traffic types*. We define ρ to be the percentage of bandwidth reservation which exceeds the corresponding LP bandwidth reservation. $\rho = 0$ *means our algorithm achieves the optimal LP solution.* For ease in presentation the actual bandwidth reservations are given as scaled values.

A. Results for Known Topologies

Our simulation framework was run on 3 different known topologies: (1) a *regular* topology that has a lattice like structure [26], (2) the KL topology [15], and (3) US map topology [21]. Tables I,II,III show the results for the known topologies.

First we would like to point out that, as expected, *the bandwidth reservation obtained using the LP is the lowest possible reservation in all the three topologies for all demands*. In the case of the KL topology we find that the *local search heuristic reduces the bandwidth reservation of the No Sharing case by as much as 35%; it improves the bandwidth reservation of the basic Tree-Bandwidth algorithm (shown as "before LS" in the table) under all demands and by as much as 19% (the 2 demand case); in 4 out of 6 cases it obtains the optimal reservation as provided by the LP*. It is very close to the LP bandwidth reservation, at most reserves 11% more bandwidth. In the US topology we find that the local search heuristic reduces the bandwidth reservation of the No Sharing case by as much as 44%; it improves the bandwidth reservation of the basic Tree-Bandwidth algorithm in 3 out of 6 cases, with an maximum percentage bandwidth gain of 17% (the 2 demand case); also in 2 out of the 3 cases where a gain is obtained the bandwidth reservation is the same as that obtained using the LP; it reserves at most 4% more bandwidth than the LP. In the more regular lattice like topology we find that the local search heuristic provides a gain in bandwidth reservation over the No Sharing case by as much as 40%; in 5 out of the 6 cases the bandwidth reservation is the same as that obtained using the LP; after applying local search heuristic, the bandwidth reservation is reduced by as much as 13% (the 10 demand case) when compared with that obtained from the Tree-Bandwidth algorithm. The local search heuristic reserves at most 20% more bandwidth than the LP.

k	No Sharing	LP	Before LS	After LS	BW gain %	ρ %
2	900	700	700	700	22	0
5	3.5k	2.5k	3k	3k	14	20
10	7.5k	5k	5.8k	5k	33	0
15	10.4k	6.9k	7.4k	6.9k	34	0
20	15.4k	9.5k	10.6k	9.5k	38	0
25	19.6k	11.8k	12.1k	11.8k	40	0

TABLE III

BANDWIDTH RESERVATIONS USING LP AND THE LOCAL SEARCH HEURISTIC FOR REGULAR TOPOLOGY; NUMBER OF NODES = 14; NUMBER OF EDGES = 29. K: NUMBER OF DEMANDS, BW GAIN IS W.R.TO NO SHARING, ρ: PERCENT BW GAIN W.R.TO LP

k	No Sharing	LP	Before LS	After LS	BW gain %	ρ %
2	11.9k	11.9k	11.9k	11.9k	0	0
5	25.9k	23.9k	25.9k	23.9k	8	0
10	45.8k	31.8k	35.8k	33.8k	26	6
15	67.7k	49.8k	57.7k	49.8k	26	0
20	87.6k	65.7k	73.6k	65.7k	25	0
25	105.5k	75.6k	87.6k	75.6k	28	0
50	217k	147k	151k	149k	31	1

TABLE IV

BANDWIDTH RESERVATIONS USING LP AND THE LOCAL SEARCH HEURISTIC FOR A 60 NODE TOPOLOGY WITH 120 EDGES; K: NUMBER OF DEMANDS, BW GAIN IS W.R.TO NO SHARING, ρ: PERCENT BW GAIN W.R.TO LP

m	k	No Shar.	LP	Before LS	After LS	BW gain %	ρ %
100	2	19.8k	17.8k	17.8k	17.8k	10	0
	10	91k	77k	77k	77k	15	0
	20	166k	132.7k	132.7k	132.7k	20	0
200	2	12k	10k	12k	10k	17	0
	10	55.7k	43.8k	47.8k	43.8k	21	0
	20	99.5k	73.6k	75.6k	73.6k	26	0
400	2	8k	8k	8k	8k	0	0
	10	47.8k	39.8k	39.8k	39.8k	17	0
	20	87.6k	65.7k	71.6k	67.7k	23	3
	50	199k	137k	151k	139k	30	2

TABLE V

BANDWIDTH RESERVATIONS USING LP AND THE LOCAL SEARCH HEURISTIC FOR A 100 NODE TOPOLOGY; M: NUMBER OF EDGES; K: NUMBER OF DEMANDS, BW GAIN IS W.R.TO NO SHARING, ρ: PERCENT BW GAIN W.R.TO LP

B. Results on Large Topologies

For large network topologies we generated networks with node sizes of 60 and 100 and with number of edges varied as 120, 100, 200 and 400. In each of our examples we used two types of traffic demands.

The results for the 60 node topology is tabulated in Table IV. When compared with the No Sharing case, the percentage bandwidth gain obtained for the local search heuristic is above 25% in 5 out of the 7 cases; the local search heuristic improves over the Tree-Bandwidth algorithm by as much as 14% (the 15 demand case) and in 6 out of the 7 cases. For the one case where no improvement is obtained by the search heuristic, algorithm Tree-Bandwidth obtained a bandwidth reservation equal to that in the case of the LP. Furthermore, in 4 out of the 6 cases where an improvement is observed the local search heuristic was able to yield a bandwidth reservation that was equal to that obtained using the LP.

The results for the 100 node topology is presented in Table V. When the number of edges is 100, we find that the solution obtained from the Tree-Bandwidth algorithm and the LP are exactly the same. However, we see that, in the 200 and 400 edges case, Tree-Bandwidth can reserve 20% (the 2 demand case) and 10% (the 50 demand case) more than that of the LP case. The local search heuristic is able to reduce this over-reservation to 0% and 2%. As we can see from the table, the bandwidth gain over the No Sharing case is very substantial in 9 out of the 10 cases; in the one case where there is no improvement, the No Sharing case finds the optimal solution. This is because the disjoint shortest paths for the 2 demand is optimal, thus sharing is not necessary.

VII. DISCUSSION

In this work we have described 3 different approaches to obtain the minimum bandwidth reservation for two different types of traffic demands. The key issue here is that both these types of traffic demands are allowed to share the bandwidth on any given edge in the network. One algorithm is the straightforward solution to the linear program for the minimum bandwidth reservation in the splittable flow case. This solution can be obtained in polynomial time and can be solved optimally. The second algorithm is for the confluent flow case and is a simple algorithm that is based on the computation of the shortest path tree. This solution is not optimal and can be further improved by using the local search heuristic on the computed shortest path tree. Our discussion focuses primarily on the intuition gained in how the heuristic works for the different example topologies. In all our example topologies, large and small, we find that as expected the LP always gives the minimum possible bandwidth reservation.

A. Local Search Heuristic

Based on experimental evidence we notice that the local search heuristic improves the solution of Tree-Bandwidth algorithm and in most cases was able to reduce the total bandwidth reservation to that obtained using the LP. However note that as expected the heuristic can never improve over the bandwidth reservation obtained using the LP (since the LP provides the lower bound). In most cases when the heuristic gives a 0 bandwidth gain over Tree-Bandwidth algorithm, Tree-Bandwidth algorithm has already yielded a solution that is close to or equal to that obtained by the LP. Figure 7 shows these characteristics for a 100 node 400 edge network topology.

There are several factors that contribute to the gains observed by using the local search heuristic. Two of the most important criteria are the number of demands and the degree of connectivity in the network. Of the two, the latter seems to play a crucial role.

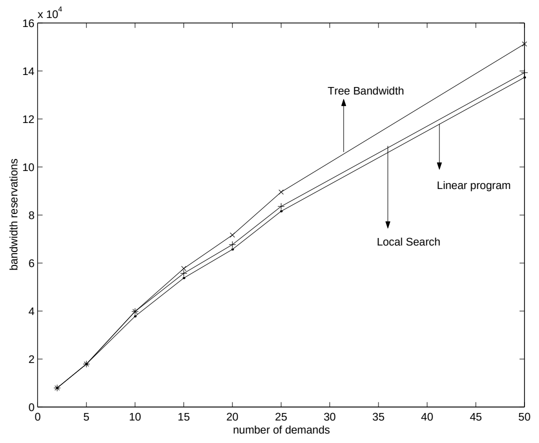


Fig. 7. Comparison of total bandwidth reservation obtained using the 3 algorithms for a 100 node 400 edge network topology

Impact of Network Connectivity:
In our example topologies such as the US topology, we found that when the network was sparsely connected or had proportionately fewer number of edges the local search heuristic does not provide much gains over Tree-Bandwidth algorithm. This is because in our local search, at every step we depend on the availability of many paths in order to choose a path that reduces bandwidth between any two nodes on the tree. Furthermore, as observed in the 100 node 100 edge topology we find that the low connectivity in the network yields identical solutions for both the tree as well as the LP and hence no improvement is possible from the local search. However when the number of edges increases to 400, the tree heuristic does show promise especially for larger number of demands as shown in Figure 7.

Impact of the Number of Demands:
Our experimental analysis (KL topology, 100 nodes

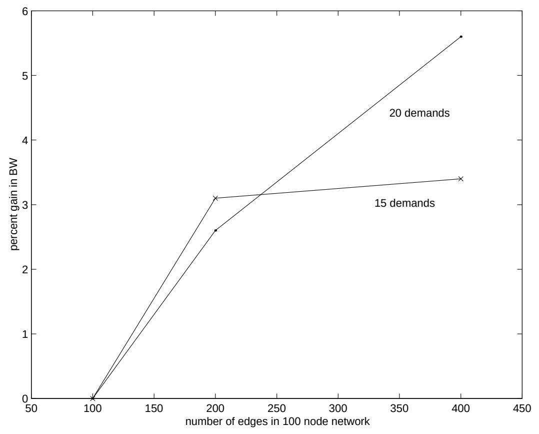


Fig. 8. Bandwidth gains obtained using the local search heuristic for a 100 node network topology

200 edges) shows that for lower network connectivity the bandwidth gains using the search heuristic yields better gains over Tree-Bandwidth algorithm when there are fewer number of demands. This is because of the greater flexibility in bandwidth availability along the different network edges. We also observed that as the connectivity in the network increases (regular topology, 100 nodes 400 edges) the larger number of demands generally yields greater improvements with the help of the search heuristic. This is shown in Figure 8 for two different demands. In the same example we find that for more network connectivity and fewer number of demands the simple Tree-Bandwidth algorithm yields the same solution as the LP.

VIII. Related Work

The protocol and architecture aspects of VPN network design and MPLS networks have been extensively studied [6], [22]. For the hose model, efficient bandwidth reservation algorithms has been studied in [7], [1], [10]. Recently Gupta et al. proposed a simple 5.5-approximation algorithm for the problem. Fast failure recovery for VPN networks has been studied in [14]. However these papers only address bandwidth reservation schemes for a single type of demand and therefore do not consider the problem of minimizing bandwidth reservation when sharing bandwidth between multiple traffic classes.

For the pipe model, efficient bandwidth reservation with fast restoration has been extensively studied[18], [19]. These algorithms tradeoff bandwidth reservation with the speed of restoration. In the single link failure model, if a link e_i has reserved bandwidth b_i to backup a link e_j, if the backup path P_2 of a given primary path P_1 traverses link e_i, but not e_j, then b_i can be used by the backup path for free since the links in P_2 do not fail at the same time with e_j. The bandwidth sharing for path restoration is much more complicated than the sharing in our context. Their techniques do not have any provable bounds.

The work presented here is in the same spirit as the recent work of Goldenberg et al. [9]. In their work, they design smart routing algorithms to minimize the cost of multi-homing while achieving a minimal latency objective. Their work is restricted to distributing load among different network providers (the topology can be assumed as multiple parallel links) and does not consider different type of traffic. Our work proposes novel algorithms to reduce the cost of bandwidth reservation while satisfying QoS requirements for both real time traffic and best effort data traffic. With our solution the peak demand of real time traffic as well as the average demand of best effort data traffic are satisfied.

IX. Conclusion and Future Work

The theoretical analysis along with the simulation studies presented in this paper show that bandwidth sharing between multiple traffic classes is an important aspect of cost effective network design. This is especially true in cases where network bandwidth is priced based on peak rates and the duration of occurrence of these peak rates is relatively small when compared to the overall life time of the network service. In this work we show that in the multi-path (splittable flow) bandwidth sharing scenario, we can obtain an optimal solution through the solution of a linear program. However in the unsplittable (single path) confluent (all routes to the same destination form a tree as it is done by any routing protocol) flow case, the problem is NP-hard and is not approximable within a factor better than 3/2 in the directed case. We design a local search heuristic which has an approximation factor of 2. With respect to the no sharing case our Tree-Bandwidth algorithm along with the local search heuristic in most cases provides almost optimal solution (found in the LP lower bound).

For our future work, we plan to investigate whether there is any approximation algorithm with an approximation factor better than 2 for the (undirected) confluent flow case. We would also like to design backup networks to cope with link failure in our shared network design framework.

References

[1] B. Yener A. Kumar, R. Rastogi and A. Silberschatz. Algorithms for provisioning VPNs in the hose model. In *Proceedings of ACM SIGCOMM*, pages 135–146, 2001.

[2] Y. Bartal. On approximating arbitrary metrices by tree metrics. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 161–168, 1998.

[3] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An architecture for differentiated services. *Internet RFC 2475*, 1998.

[4] J. Chen, R. Rajaraman, and R. Sundaram. Meet and merge: approximation algorithms for confluent flows. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 373–382, 2003.

[5] ILOG CPLEX. http://www.ilog.com/products/cplex/.

[6] B. Davie and Y. Rekhter. *MPLS Technology and Applications*. Morgan Kaufmann, San Francisco, CA, 2000.

[7] N. G. Duffield, P. Goyal, A. G. Greenberg, P. P. Mishra, K. K. Ramakrishnan, and J. E. van der Merive. A flexible model for resource management in virtual private networks. In *SIGCOMM*, pages 95–108, 1999.

[8] J. Fakcharoenphol, S. Rao, and K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. In *Proceedings of the thirty-fifth ACM symposium on Theory of computing*, pages 448–455, 2003.

[9] D. Goldenberg, L. Qiu, H. Xie, Y.R. Yang, and Y. Zhang. Optimizing cost and performance for multihoming. In *Proceedings of ACM SIGCOMM*, pages 79–92, 2004.

[10] A. Gupta, A. Kumar, and T. Roughgarden. Simpler and better approximation algorithms for network design. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 365–372, 2003.

[11] Juniper Networks Inc. Feature Article : J-Voice program equips incumbent telcos to defend local access business against new packet voice competitors. `http://www.juniper.net/company/newsletter/feature_article_040601.html`.

[12] Juniper Networks Inc. Juniper Networks Delivers Carrier Grade VoIP: New High-Availability Features Transform Traditional Best-Effort Voice Services Into Carrier-Grade Solutions. `http://www.juniper.net/company/presscenter/pr/2004/pr-040607.html`, Jun 2004.

[13] Verizon Inc. Verizon VoiceWing: Broadband Phone Service Let your Voice Take Flight over Broadband Internet. `http://www22.verizon.com/ForYourhome/voip/voiphome.aspx`.

[14] G. F. Italiano, R. Rastogi, and B. Yener. Restoration algorithms for virtual private networks in the hose model. In *Proceeding of IEEE INFOCOM*, pages 131–139, 2002.

[15] K. Kar, M. Kodialam, and T. V. Lakshman. Minimum interference routing with applications to MPLS traffic engineering. *Proc. IEEE INFOCOM*, pages 884 – 893, 2000.

[16] F. P. Kelly. On tariffs, policing and admission control for multiservice networks. *Operations Research Letters*, 15:1–9, 1994.

[17] M. Kodialam and T. V. Lakshman. Minimum interference routing with applications to MPLS traffic engineering. In *Proceedings of IEEE INFOCOM*, pages 376–385, 2000.

[18] M. Kodialam and T.V. Lakshman. Dynamic routing of locally restorable bandwidth guaranteed tunnels using aggregated link usage information. In *Proceedings of IEEE INFOCOM*, pages 884–893, 2001.

[19] L. Li, M. Buddhikot, C. Chekuri, and K. Guo. Dynamic routing of bandwidth guaranteed paths with local restoration. *Journal of Selected Area of Communication (JSAC)*, 23(2):437–449, 2005.

[20] A. Medina, A. Lakhina, I. Matta, and J. Byers. Brite: An approach to universal topology generation. In *Proc. of International Workshop on Modeling, Analysis and Simulation of Computer and Telecommunications Systems (MASCOTS)*, pages 346–356, 2001.

[21] S. Nelakuditi and Z. Zhang. On selection of paths for multi-path routing. *IWQoS*, pages 170–186, 2001.

[22] E. Rosen and Y. Rekhter. BGP/MPLS VPNs. RFC 2547, IETF, March 1999.

[23] D. Songhurst and F. Kelly. Charging schemes for multi service networks. In *ITC 15, Elsevier, Amsterdam, Teletraffic Contributions for the Information Age*, pages 879–888, 1997.

[24] C. Swamy and A. Kumar. Primal-dual algorithms for connected facility location problems. *Algorithmica*, 40(4):245–269, 2004.

[25] B.M. Waxman. Routing of multipoint connections. *IEEE Journal of Selected Areas in Communication*, 6(9):1617–1622, 1988.

[26] S. Yilmaz and I. Matta. On the scalability tradeoffs in MPLS and IP routing. *Technical Report Boston University, Computer Science Department*, 2002.