Network Traffic Engineering of Heterogeneous Service Classes with Varied Survivability Requirements[†]

Shekhar Srivastava, S.R.Thirumalasetty[**], and Deep Medhi[*]
Computer Science & Electrical Engineering Division
School of Computing and Engineering
University of Missouri-Kansas City
Kansas City, MO-64110

February 2004

Abstract— **In this paper, we consider a backbone network which allocates bandwidth to service classes with varied survivability requirement. Such guaranteed services may be offered along side best-effort services in the same network. Due to different (potentially conflicting) objectives for guaranteed and best-effort services, attempts to optimize them together requires solving a multi-criteria decision problem. Here, we present an integrated optimization formulation where we unify these different objectives into a single objective function while considering varied survivability requirement for different service classes. We enforce explicit routes (single path allocation) for survivable service class, and at the same time, ensure that the best-effort services are not drastically effected. We provide two heuristic approaches towards solving this formulation. We present experimental results to show that the heuristics perform well; and more importantly, the optimization model effectively captures the different objectives. We also observed that the availability of capacity and tunnels play equally important role in ensuring the optimal utilization of network resources.**

Index Terms— **Survivable services, Optimization formulation, Algorithm.**

I. Introduction

In this paper, we consider a mixed network services environment where along with best-effort services such as email, ftp, web (as in the current Internet), there is also a class of book ahead guaranteed survivable (BAGS) service classes which have a requested degree of survivability (in essence, we use the term “survivable” to indicate that the allocation is done ahead of time, rather than provide the capability through network restoration *after* a failure). The goal is to compute off-line and allocate bandwidth for different service classes ahead of time so that certain network (traffic) engineering objectives can be met along with any other restrictions from the network.

For example, multi-protocol label switching (MPLS) technology [1], provides the ability to set up bandwidth for different service classes through label-switched paths, but however may require certain restrictions due to to tunneling limitations. An important feature of MPLS is its capability to set up multiple label switched paths (LSP) for different services. However, each LSP setup requires a label on each intermediate node which is used for switching the input traffic to the destined output port. Hence setting up each new LSP introduces additional labels to each intermediate node. To route each packet, a label switched router (LSR) would need to search through the Label Swapping table to find the matching label and the port to get the output label and the port. It then appends the output label to the packet and sends the packet to the output port. Hence each activated LSP leads to more labels at the LSR, thereby requiring more processing to forward of each packet.

For a highly connected heterogeneous network running on a sparse fiber-optic network, setting up enough LSPs to yield the benefits of traffic engineering can become an issue. Most traffic engineering formulations in the literature fail to account for the processing restrictions on the routers and hence choose tunnels based on link speeds only. Due to heterogeneity of LSRs, processing speed at certain LSRs can become a bottleneck. Such a traffic engineering formulation is presented in [3]. The approach makes sure that on engineering the network, certain routers will not get overloaded. In this paper, we extend the approach to networks where BAGS services are provided along with best-effort service.

Depending on the requirement of the user (here, the term “user” is used in a generic sense to mean access customers rather than each individual ‘human user’), several different levels of BAGS services can be envisioned: (i) guarantee the service only under normal network operating conditions, (ii) full guarantee of bandwidth under normal situations plus a reduced level of service in the event of a major link failure, and (iii) finally, fully guaranteed bandwidth both under normal as well as under a major link failure situation. For the sake of simplicity, we will refer to these three service levels as *zero*, *fractional*, and *full* BAGS services. Note that zero BAGS services do not provide any survivability. The specifics of

[†] Supported by DARPA and Air Force Research Lab under agreement no. F30602-97-1-0257.
[**] Currently at Ciena Corporation, CA, USA.
[*] Corresponding author: dmedhi@umkc.edu

using protocols such as RSVP or CR-LDP to invoke MPLS traffic engineering is outside the scope of the present paper; the interested reader is directed to [1]. We also point out that although we use LSPs and MPLS to explain and present our framework, it is indeed applicable to the class of networks that have bounded tunneling functionality.

The focus of this paper is to consider a traffic (network) engineering problem where we have BAGS and best-effort service classes sharing a network. The BAGS service classes are to be supported using LSPs such that the number of LSPs active on a link is bounded. Traffic engineering such a network faces multiple (and possibly conflicting) objectives. Hence, we also present several possible objective functions and demonstrate the interplay between them while providing BAGS services.

While, over the years, network survivability has been addressed for circuit-switched, ATM and fiber networks (see, for example, [4]–[13] for a sampling of work), MPLS allows the capability to address traffic engineering along with survivability for a new class of problems. For example, Wang and Wang [14] have addressed the explicit routing models for MPLS traffic engineering. The work that is closest to ours is by Kodialam and Lakshman [15] where they present optimization models and algorithms for guaranteed tunnels with restoration. However, there are several differences. In our problem, we focus more on survivability (rather than restoration), address book-ahead guaranteed survivable services (with different survivability requirements); further, these aspects are considered in the presence of conflicting objectives. To our knowledge, this traffic engineering problem has not been addressed so far. Further, we provide a novel modeling approach where we show how to capture different BAGS services within a single modeling framework. The reader may want to note that this approach is for off-line traffic engineering determination where the network provider/operator would like to perform such updates once a day [16].

The rest of the paper is organized as follows. In section II, we provide a generic IP optimization formulation of the BAGS traffic engineering problem with tunneling constraints. In section III, we present multiple objective functions which can be used along with the generic formulation. In section IV, we present an efficient heuristic based algorithm to solve the IP problem using a series of continuous problems followed by a reduced IP problem. In section V, we present numerical results for small and large networks (experimental and randomly generated).

II. Generic Formulation

We consider an aggregated-flow based network, where data arriving to a source for a specific destination needs to be sent over one of the active LSPs between the source and the destination. The data belongs to one of the BAGS service classes and hence can only be sent on the LSPs with required survivability. Each service class maintains its own set of LSPs between source and destinations. The total LSPs chosen to be activated across the network are such that the total number of LSPs flowing through each link are restricted. The formulation restricts that the active LSP constraint and Link Capacity constraint are honored on every link.

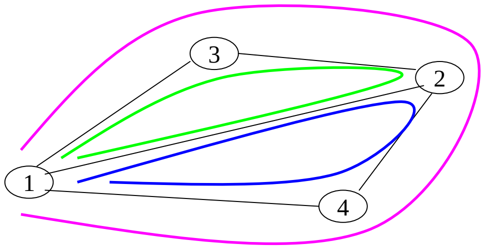

Fig. 1. Illustration of Cycles

Consider a BAGS service request for demand k and service class s that requires a bandwidth demand volume of d_k^s units between the ingress node i_k and the egress node e_k for either full, fractional or zero BAGS services. We assume that all nodes (or routers) in the network are capable of providing this functionality. For example, in an MPLS-capable routers, this demand may be set up on any possible LSPs in the network such that the active LSP constraint is not violated on any of the traversed links. In our scenario, if this demand request is at the zero-BAGS level, then only a path with bandwidth d_k^s needs to be set up ahead of time (or allocated ahead of time); on the other hand, if the request is for a full-BAGS service, a back-up path (to address for a failure situation/state, i.e., situation-disjoint) needs to be available and bandwidth d_k^s needs to be reserved also on the backup path. It is important to note that this also imposes the connectivity requirement on the network to be two-edge connected (in other words, we assume that the network design process addresses the topological need for survivability requirement). While in the case of fractional-BAGS services, the back-up path needs to be allocated the bandwidth sufficient to carry a fraction of d_k^s in order to address partial survivability.

We first discuss how to model the three different BAGS service classes. For the zero BAGS service class, a path needs to be selected. This path may *not* be the shortest (e.g. hop) path since depending on the traffic engineering goal. Thus, we can consider a set of candidate paths for the flow d_k^s of BAGS service class s of demand pair k. On the other hand, for fully-survivable BAG service class, flowing the demand d_k^s requires both a primary path and a secondary path which are situation-disjoint. While such paths can be independently modeled, we use a pairing idea, i.e., consider a pair of paths consisting of primary and secondary paths that are situation-disjoint. For simplicity, we refer to a pair of such paths as a *cycle* or, a *cycle path* (note that this cycle path idea has been used earlier for fiber network survivability in [17]). An illustration of three candidate cycles for demand with end nodes 1 and 2 (for a four-node network) is shown in Figure 1.

Similar to the case of zero-survivable services, the selection of the shortest *cycle* for a demand d_k^s may not be in the best interest of the traffic engineering objective. Thus, we can consider a candidate set of cycles for a flow demand d_k^s, for full survivability.

The fraction BAGS service class also requires a pair of disjoint paths as in the case of full BAGS service class. The difference is that, on the back-up path, only a fraction of the demand is required to be reserved. If we denote the fraction by α_k^s (where $0 \leq \alpha_k^s \leq 1$), then the primary path would reserve d_k^s while the back-up path would reserve $\alpha_k^s d_k^s$. Combining the above three cases, we can see that it is actually not necessary to model each BAGS service class independently if the idea of cycle along with the fractional parameter (α) is used. In other words, a set of candidate cycles can be considered for all BAGS classes. If it is for a full BAGS class, then α_k^s is set to 1, and for fraction BAGS, this parameter is set to a value between 0 and 1 (as requested by the user through a service-level agreement) while for zero-survivable, this parameter takes the value zero. Thus, an unified view can be considered which is part of goal in the problem formulation.

We first describe the notation:

$\mathcal{N}$	:	Set of Nodes in the Network
$\mathcal{L}$	:	Set of links in the network
$\mathcal{K}$	:	Set of demand pairs generating traffic in the network
$\mathcal{R}$	:	Set of failure situations
$\mathcal{L}_\sigma$	:	Set of link(s) that fail in failure situation $\sigma \in \mathcal{R}$
$\mathcal{S}_k$	:	Set of BAGS service classes for demand k of the network
ξ_k^s	:	Revenue from carrying service class $s \in \mathcal{S}_k$ of $k \in \mathcal{K}$
d_k^s	:	Volume of traffic generated by service class $s \in \mathcal{S}_k$ of $k \in \mathcal{K}$
α_k^s	:	Survivability requirement of service class $s \in \mathcal{S}_k$ of $k \in \mathcal{K}$
T_l	:	Maximum number of tunnels allowed on link $l \in \mathcal{L}$
C_l	:	Capacity of link $l \in \mathcal{L}$ (in BBU)

Let $|P_k^s|$ be the number of candidate cycles generated for service class $s \in \mathcal{S}_k$ of demand $k \in \mathcal{K}$. We now introduce the *decision variable* x_{km}^s associated with the cycle m for service class $s \in \mathcal{S}_k$ of request $k \in \mathcal{K}$ which takes the value 1 if this cycle is selected (by the design); otherwise, it takes the value 0. As discussed earlier, due to capacity limitation, it is quite possible that a demand may not be routed (while proper network design would try to avoid such situations by over-engineering; from a traffic engineering modeling standpoint, it is necessary to incorporate this variable to avoid infeasibility of the problem). To consider this aspect, we also introduce an artificial variable w_k^s with each service class $s \in \mathcal{S}_k$ of demand request $k \in \mathcal{K}$; this takes the value 1 if the demand is NOT accommodated and 0, otherwise. Thus, to consider the scenario of either choosing a cycle from a set of candidate cycles or not selecting at all for each of the demand requests, we have the following constraints

$$w_k^s + \sum_{m \in \mathcal{P}_k^s} x_{km}^s \; = \; 1.0 \quad s \in \mathcal{S}_k, k \in \mathcal{K} \tag{1a}$$

$$x_{km}^s, w_k^s \in \{0,1\} \qquad m \in \mathcal{P}_k^s, s \in \mathcal{S}_k, k \in \mathcal{K}. \tag{1b}$$

Now, for each cycle (because of the way each of them are generated) we have a 'main' path and the backup situation disjoint path. For notational clarity, the main path will be superscripted with p (for 'primary'), and the backup path by s (for 'secondary'). To address for flowing the demand on each link (for each path), we now introduce the indicator notation to map between the demand, the cycle and the link, as they relate to primary or the back-up (secondary) path, as follows:

$\delta_{km}^{s\ell}$:	1, if candidate cycle $m \in \mathcal{P}_k^s$ for service class $s \in \mathcal{S}_k$ of request $k \in \mathcal{K}$ uses link $\ell \in \mathcal{L}$ in its primary path 0, Otherwise
$\beta_{km}^{s\ell}$:	1, if candidate cycle $m \in \mathcal{P}_k^s$ for service class $s \in \mathcal{S}_k$ of request $k \in \mathcal{K}$ uses link $\ell \in \mathcal{L}$ in its back-up/secondary path 0, Otherwise

Note that, by definition, for a specific link and a cycle, both indicators can not be one (since the path pairs are link-disjoint). As discussed earlier, all the three types of classes (full, fractional and zero BAGS) can be modeled in our case by the parameter α_k^s. Thus, the bandwidth needed on any link ℓ (denoted by F_ℓ) to carry both for primary path routing and for back-up path routing for different BAG demand request can now be captured by the amount

$$F_\ell = \sum_{k \in \mathcal{K}} \sum_{s \in \mathcal{S}_k} \sum_{m \in \mathcal{P}_k^s} x_{km}^s [\delta_{km}^{s\ell} + \beta_{km}^{s\ell} \alpha_k^s] d_k^s.$$

Note the inclusion of parameter α_k^s with the second term which is dictated by the level of survivability. Since each link ℓ has capacity C_ℓ, we thus have the following constraints for each link $\ell \in \mathcal{L}$:

$$\sum_{k \in \mathcal{K}} \sum_{s \in \mathcal{S}_k} \sum_{m \in \mathcal{P}_k^s} x_{km}^s [\delta_{km}^{s\ell} + \beta_{km}^{s\ell} \alpha_k^s] d_k^s \leq C_\ell \quad \ell \in \mathcal{L}. \tag{2}$$

Next, we consider the number of active LSPs sharing a link. Since we want that the number of active tunnels on any link ℓ should be less that T_ℓ. Since, for zero BAGS services back-up path is not allocated, no additional tunnels are created. While for the fractional and full BAGS the backup path needs to be created and hence it contributes to the number of active LSPs. We capture the variation by using an indicator function $1_{\{\alpha_k^s > 0\}}$, which is 1 when $\alpha_k^s > 0$ and 0 otherwise. Hence, we have following constraint to account for tunnel constraint.

$$\sum_{k \in \mathcal{K}} \sum_{s \in \mathcal{S}_k} \sum_{m \in \mathcal{P}_k^s} [\delta_{km}^{s\ell} + 1_{\{\alpha_k^s > 0\}} \beta_{km}^{s\ell}] \; x_{km}^s \leq T_\ell \;\; \ell \in \mathcal{L}. \tag{3}$$

We now assimilate all the previously discussed constraints and present the generic formulation. The objective functions to be used will be discussed later in section III. The formulation minimizes the objective function in the space of the presented constraints. The problem (**P**) can be formulated as:

$$F^* = \min_{\{\boldsymbol{x}, \boldsymbol{w}\}} \; f(\boldsymbol{x}, \boldsymbol{w}) \tag{4}$$

Subject to:

$$w_k^s + \sum_{m \in \mathcal{P}_k^s} x_{km}^s \ = \ 1.0 \quad s \in \mathcal{S}_k, k \in \mathcal{K} \tag{5a}$$

$$\sum_{k \in \mathcal{K}} \sum_{s \in \mathcal{S}_k} d_k^s \sum_{m \in \mathcal{P}_k^s} [\delta_{km}^{s\ell} + \alpha_k^s \beta_{km}^{s\ell}] \, x_{km}^s \leq C_\ell \ \ \ell \in \mathcal{L} \tag{5b}$$

$$\sum_{k \in \mathcal{K}} \sum_{s \in \mathcal{S}_k} \sum_{m \in \mathcal{P}_k^s} [\delta_{km}^{s\ell} + 1_{\{\alpha_k^s > 0\}} \beta_{km}^{s\ell}] \, x_{km}^s \leq T_\ell \ \ \ell \in \mathcal{L} \tag{5c}$$

$$x_{km}^s, w_k^s \in \{0, 1\} \qquad m \in \mathcal{P}_k^s, s \in \mathcal{S}_k, k \in \mathcal{K} \tag{5d}$$

In the next section, we present various objective functions which need to be minimized to ensure good performance in one aspect or the other. These objective could be conflicting with each other and hence provides fertile environment to observe and establish their interplay.

III. Objective Functions

Our goal is to account for several possible objectives in a unified formulation. These objectives capture various conflicting requirements which are relevant to a network engineer. First consideration is that although best effort services are offered no guarantee of Quality of Service (QoS) in terms of bandwidth resources, it is desirable to improve the QoS of the best effort services without effecting the performance of BAGS service classes. Such an objective can be achieved by maximizing the residual capacities of all links in the network. Another possible objective is to minimize the total demand routing cost for BAGS services. Such a reservation attempts to derive shortest-path type allocations. However, this will not be the case in a capacitated network, especially in the presence of survivability requirements. Still, another objective is to minimize the penalty if a demand can not be accommodated due to, say, capacity limitation. Contrary to the minimization of penalty objective, another goal can be the maximization of revenue (especially if capacity limitation imposes the decision on demand selection to maximize the revenue). Thus, the problem has several objectives while some of them may be contradictory to others.

We first maximize the residual capacity in the network so as to achieve better QoS for the best-effort services. This can be written as:

$$F^* = \max_{\{\boldsymbol{x}, \boldsymbol{w}\}} \sum_{\ell \in \mathcal{L}} [C_\ell - r_\ell] \tag{6}$$

where r_ℓ is the capacity consumed on link ℓ by different BAGS service requests. It is easy to see that the maximization is equivalent to minimizing the sum of link flow r_ℓ's. Hence, this objective can be rewritten as

$$F^* = \min_{\{\boldsymbol{x}, \boldsymbol{w}\}} \sum_{\ell \in \mathcal{L}} r_\ell. \tag{7}$$

There are two ways of computing the r_ℓ in the context of the first objective (7). In the first case, r_ℓ is computed assuming that both primary and backup paths are allocated bandwidth where as in the second case only primary path is assumed to be allocated capacity. These two requirements are referred to as hard and soft requirements, respectively.

A. Hard Requirement

When we want to make sure that the backup paths are reserved *apriori* to the failures, we use hard requirements. Here, r_ℓ is the capacity consumed by both the primary and the backup path. It refers to the situation where the capacity required by both the primary and the backup paths is explicitly allocated at the time of service request reservation. For such a requirement, load on the network can be computed as

$$f_c = \sum_{\ell \in \mathcal{L}} \sum_{k \in \mathcal{K}} \sum_{s \in \mathcal{S}_k} \sum_{m \in \mathcal{P}_k^s} x_{km}^s [\delta_{km}^{s\ell} d_k^s + \beta_{km}^{s\ell} \alpha_k^s d_k^s] \tag{8}$$

Next, we need to capture the routing cost of the flows in the formulation. If c_{km}^{sp} is the routing cost on the primary path, and c_{km}^{sb} is the routing cost on the back-up path of the candidate cycle $m \in \mathcal{P}_k^s$ of service class $s \in \mathcal{S}_k$ and $k \in \mathcal{K}$. Then the objective function in order to minimize the total routing cost is

$$f_r = \sum_{k \in \mathcal{K}} \sum_{s \in \mathcal{S}_k} \sum_{m \in \mathcal{P}_k^s} x_{km}^s [c_{km}^{sp} + \gamma c_{km}^{sb}] \tag{9}$$

where γ is a cost proportional constant. As we are having hard survivability requirement, we already have backup paths signalled and provisioned across the network. Since the primary path requires more sophisticated handling (it is carrying packets) than the backup path, we incorporate such an asymmetry by weighing the backup path cost with γ. On one hand, if γ is set 1.0, it would mean that backup paths require same routing cost as the primary path. On the other hand, if $\gamma = 0.0$, then backup paths do not incur any routing cost, they are allocated and just kept alive. Having intermediate values of γ would mean that some fraction (possibly important sections) of the traffic is still sent over the backup paths even when the primary path is working. This would ensure that in the event of a failure and possible disruption of primary path, important sections of the traffic are still delivered to the destination.

It may, however, be noted that if we consider the optimization problem of *just* minimizing (9) with the constraint set being (5) when the routing cost components c_{km}^{sp} and c_{km}^{sb} are positive, the optimal solution to the optimization problem is the solution of selecting the artificial variable w_k^s (i.e. NOT routing any demand) for all the demand requests! (essentially, there is no cost if nothing is routed.) Thus, we need to introduce the penalty cost, η_k^s, of choosing the artificial variable, and this cost component to the objective function; this, then, also addresses the third objective. Thus, the objective that incorporates the penalty cost also (for the third objective) can be re-written as

$$f_r = \sum_{k \in \mathcal{K}} \sum_{s \in \mathcal{S}_k} \left\{ \sum_{m \in \mathcal{P}_k^s} x_{km}^s [c_{km}^{sp} + \gamma c_{km}^{sb}] + \eta_k^s w_k^s \right\} \tag{10}$$

The two objective functions f_c and f_r constructed so far minimize the used capacity and the routing cost in the network. Since, both the objectives are relevant in engineering a network, it is desirable to combine the two objective functions and construct a unified function which should be minimized. This can be accomplished by weighing one of the functions by a weight factor and take the sum with the other function and then minimizing the combined cost. If we use the normalized weight for objective function f_r, then the weight factor θ (specifically, θ_k^s for each $s \in \mathcal{S}_k$ and $k \in \mathcal{K}$) is needed only for objective function f_c. Thus, we have the following combined objective function.

$$f_{cr} = \sum_{k\in\mathcal{K}} \sum_{s\in\mathcal{S}_k} \left\{ \sum_{m\in\mathcal{P}_k^s} \left[x_{km}^s (c_{km}^{sp} + \gamma c_{km}^{sb}) \right] + \eta_k^s w_k^s \right\}$$

$$+ \sum_{k\in\mathcal{K}} \sum_{s\in\mathcal{S}_k} \theta_k^s \left\{ \sum_{\ell\in\mathcal{L}} \sum_{m\in\mathcal{P}_k^s} x_{km}^s [\delta_{km}^{s\ell} d_k^s + \beta_{km}^{s\ell} \alpha_k^s d_k^s] \right\} \quad (11)$$

Note that in the general case, the cost of routing a path (primary or backup) may depend on many considerations. One of the most important consideration would be whether the path is allocated or not but upon allocation many other restrictions may apply. For example, it may depend on the memory size and processing power of the nodes through which the path traverses in which case cost of maintaining the path could be the sum of costs of traversing those routers. A more sophisticated way of capturing the routing cost is to consider the amount of forwarding of packets required for maintaining the path. Such a criteria would be additive in the demand that needs to flow on each link of the path, it can be captured as

$$c_{km}^{sp} = \sum_{\ell\in\mathcal{L}} \delta_{km}^{s\ell} d_k^s \quad (12a)$$

$$c_{km}^{sb} = \sum_{\ell\in\mathcal{L}} \beta_{km}^{s\ell} \alpha_k^s d_k^s \quad (12b)$$

Substituting the values of c_{km}^{sp} and c_{km}^{sb} into the function f_{cr} and rearranging, we get:

$$\sum_{k\in\mathcal{K}} \sum_{s\in\mathcal{S}_k} \left\{ \eta_k^s w_k^s + \sum_{m\in\mathcal{P}_k^s} x_{km}^s [(1+\theta_k^s) c_{km}^{sp} + (\gamma + \theta_k^s) c_{km}^{sb}] \right\}$$

For simplicity, we'll use the following notation

$$\xi_{km}^s = (1+\theta_k^s) c_{km}^{sp} + (\gamma + \theta_k^s) c_{km}^{sb}$$

Therefore, after the change of variables the function f_{cr} takes the form

$$f_{cr} = \sum_{k\in\mathcal{K}} \sum_{s\in\mathcal{S}_k} [\eta_k^s w_k^s + \sum_{m\in\mathcal{P}_k^s} \xi_{km}^s x_{km}^s] \quad (13)$$

Observe that ξ_{km}^s is the cost of accepting the demand request from node pair k and service class s. The cost only takes into account the cost associated with the resources requested by the request. Such a scenario is impractical since their is a cost with accepting the flow but no gains corresponding to the flow. Hence, in the next step we incorporate revenue for accepting a demand; this will be reflected in the form of a utilization parameter. The utility varies from one request to the other. A request may have to be allocated a cycle even at higher costs (that is, it consumes more bandwidth), if the utility of that request is higher. A request which has not been allocated any cycle path will generate no utility. We need to select an optimal set of cycles for the given set of requests at minimal possible costs while generating maximum possible revenue. Let u_k^s be the (normalized) utility of service class $s \in \mathcal{S}_k$ of request $k \in \mathcal{K}$ with u_m varying between 0.0 and 1.0. Incorporating such a utility in the function f_{cr} leads us to

$$f_h = \sum_{k\in\mathcal{K}} \sum_{s\in\mathcal{S}_k} [\eta_k^s w_k^s + \sum_{m\in\mathcal{P}_k^s} (\xi_{km}^s - u_k^s R) x_{km}^s]. \quad (14)$$

where R is utility weighing factor. The value of the utility weighing factor R dictates the importance of utilities of requests over the costs. When we minimize the function f_h as the objective function constrained by (5), we refer to the problem as (**P**$_h$).

B. *Soft Requirement*

When we wish that only the backup paths are known before hand but not reserved, i.e. the capacity on the backup path is "allowed" for use by best-effort services *as long as* there is no failure; backup paths are immediately allocated to BAGS service class as soon as a failure occurs through a signalling message (thus, bumping out best-effort services). Although such a benefit comes at the cost of increase in time required to survive a failure. Since, the backup paths are not allocated capacity until the failure happens, it takes additional time in clearing the best-effort traffic and providing the required bandwidth to the BAGS service class. For such a scenario, the total load on the network due to the primary path can be captured as

$$f_c = \sum_{\ell\in\mathcal{L}} \sum_{k\in\mathcal{K}} \sum_{s\in\mathcal{S}_k} \sum_{m\in\mathcal{P}_k^s} x_{km}^s \delta_{km}^{s\ell} d_k^s \quad (15)$$

The routing cost for the primary path will be

$$f_r = \sum_{k\in\mathcal{K}} \sum_{s\in\mathcal{S}_k} \left\{ \sum_{m\in\mathcal{P}_k^s} x_{km}^s c_{km}^{sp} + \eta_k^s w_k^s \right\} \quad (16)$$

Similar to the hard requirement scenario, we construct the combined used capacity and routing cost function f_{cr} as

$$\sum_{k\in\mathcal{K}} \sum_{s\in\mathcal{S}_k} \left\{ \eta_k^s w_k^s + \sum_{m\in\mathcal{P}_k^s} \left[x_{km}^s c_{km}^{sp} + \theta_k^s \sum_{\ell\in\mathcal{L}} x_{km}^s \delta_{km}^{s\ell} d_k^s \right] \right\}$$

Incorporating the routing cost in the same fashion as in (12a), then rearranging the terms we get

$$f_{cr} = \sum_{k\in\mathcal{K}} \sum_{s\in\mathcal{S}_k} \left\{ \eta_k^s w_k^s + \sum_{m\in\mathcal{P}_k^s} x_{km}^s (1+\theta_k^s) c_{km}^{sp} \right\}$$

Using the same notation for simplicity as in the case of hard requirement, we have

$$\xi_{km}^{s} = (1+\theta_k^s)c_{km}^{sp}$$

Incorporating the utility for the carried demands, we get the final objective function as

$$f_s = \sum_{k\in\mathcal{K}}\sum_{s\in\mathcal{S}_k}[\eta_k^s w_k^s + \sum_{m\in\mathcal{P}_k^s}(\xi_{km}^s - u_k^s R)x_{km}^s]. \tag{17}$$

when we minimize the function f_s as the objective function constrained by (5), it is referred to as problem (**P**$_s$).

To recap, we have presented above an integrated optimization model for traffic engineering a network supporting BAGS services along side best-effort traffic were modeled along with differing objective criteria. We presented the problem in two variations, namely hard and soft requirement. A salient feature of this model is the use of the cycle path idea which integrates primary and disjoint back-up paths needed for fully and fractional BAGS services in the same framework.

IV. Solution Approach

It should first be noted that since book-ahead traffic engineering design is to meet survivable BAG demand ahead of time, a real-time/on-line algorithm is not required; at the same time, an efficient algorithm is required so that the solution can be loaded to the network. This is further complicated by the fact that the optimization model (**P**) is an integer linear programming (ILP) problem. Typically, generic ILPs are solved using the branch-and-bound algorithm and/or Gomory's cutting plan method. But such approaches are not quite practical for problems of large size. The formulation (**P**) presented grows with the size of the network, number of service classes and with the number of path couples considered. Therefore, using direct methods will restrict the applicability of the approach to smaller networks. In order to make the approach suitable for large networks, we present two heuristics $gIP()$ and $gSA()$ which provide a feasible solution (not necessarily optimal) within a reasonable amount of time.

A. Heuristic I (gIP)

We now present algorithmic details and implementation of Heuristic I. It is based on successive approximations by continuous relaxations of the problem. Such an approach is guided by following observations about the structure of the problem. Notice that the size of the problem grows as the number of demand and service class increases (and also the number of candidate cycles increases). Further, the linear programming relaxation is much easier to solve; this requires relaxing the zero/one requirement (1b) on the cycle path decision variables. Also, the LP relaxation provides a lower bound to the solution. Thus, we have developed a simple heuristic (Algorithm 1) based on solving the LP relaxations.

When solving (**P**) using heuristic $gIP()$, we define sets X_f and X_v containing the variables **x** which have been fixed (their value is already decided) and the ones that are still variables, respectively. Initially, X_f is empty and X_v contains all the **x** variables.

Algorithm 1 Successive Approximation Approach

```
    $gIP()$set $X_v = \{x_{km}^s, m \in \mathcal{P}_k^s, s \in \mathcal{S}_k, k \in \mathcal{K}\}$ set
    $X_v = X_v \cup \{w_k^s, s \in \mathcal{S}_k, k \in \mathcal{K}\}$ set $X_f = \{\phi\}$ done $\leftarrow$
    0 change $\leftarrow$ 1 while done = 0 AND change = 1 do
    x $\leftarrow$ solve $Relaxed\_gIP(X_v)$
    done $\leftarrow$ 1
    change $\leftarrow$ 0
    for all $X \in X_v$ do
      if $x \leq \eta$ then
        $x = 0$
        $X_f = X_f \cup \{x\}$
        $X_v = X_v \backslash \{x\}$
        change = 1
      else if $x \geq 1.0 - \eta$ then
        $x = 1$
        $X_f = X_f \cup \{x\}$
        $X_v = X_v \backslash \{x\}$
        change = 1
      else
        done = 0
      end if
    end for
  end while
  if done = 0 OR change = 1 then
    x $\leftarrow$ solve $Integer\_gIP(X_v)$
  end if
  return x
```

Using the value of X_v, we solve the problem $Relaxed\_gIP(X_v)$, which is the relaxed version of problem $gIP()$ with the members of the set X_v as continuous variables and elements of X_f as fixed to the already decided values. $Relaxed\_gIP(X_v)$ is a continuous linear programming problem which can be solved effectively by Simplex method even for fairly large number of variables and constraints. Upon obtaining the solution, we first inspect the solution for the values of variable $x_{km}^s \in X_v$. The values obtained could be (a) $x_{km}^s \geq 1.0 - \eta$, (b) $x_{km}^s \leq \eta$, (c) $\eta < x_{km}^s < 1.0 - \eta$, where η is error margin (we assumed 0.05).

The values of x_{km}^s's of type (a) are set to 1, type (b). are set to 0 and sets X_v and X_f are updated as, $X_v \leftarrow X_v \backslash \{x_{km}^s\}$, $X_f \leftarrow X_f \cup \{x_{km}^s\}$. The variables of type (c) are left as variables. We, then delete capacities and tunnels on each link based on the values of x_{km}^s which are in X_f. We then solve the reduced problem $Relaxed\_gIP(X_v)$. We repeat the procedure till we find variables of types (a) and (b) i.e. problem size reduction is achieved.

Finally, when we are unable to get any more reduction in the size of the problem, we go for solving the IP problem using direct branch-and-bound methods in $Integer\_gIP(X_v)$. During experiments it was observed that the size of such a reduced problem is fairly small and hence does not prove to be a limitation. Hence, we have an acceptable quality solution with $\mathbf{x} \in \{0, 1\}$.

B. Heuristic II (gSA)

In this subsection, we discuss the construction and implementation of Heuristic II. We observe that the problem (**P**) is a multicommodity integral flow problem. We base the heuristic II on *simulated allocation* approach [18], [19]. The idea of SA has its source in discrete event simulation of the performance of alternative call routing in circuit switched telecommunication networks. The approach has been effectively used in reconfiguration and design of transmission networks such as SDH and ATM.

We describe the implementation of the algorithm based on simulated allocation for generic formulation of the problem (**P**). The algorithm is a minor variant in the sense that allocation never violates the constraints and is inspired by [20]. The algorithm $gSA()$ minimizes the function $f(\mathbf{x}, \mathbf{w})$ while honoring constraints (5). The $gSA()$ algorithm works with partial allocation sequences $\mathbf{x} = (x^s_{km}, m \in \mathcal{P}^s_k, s \in \mathcal{S}_k, k \in \mathcal{K})$. We choose the values of $\mathbf{w}$ in such a way that the constraint (5a) is always satisfied, i.e., if

$$\sum_{m \in \mathcal{P}^s_k} x^s_{km} = 0 \quad \Rightarrow w^s_k = 1 \tag{18}$$

else $w^s_k = 0$; for all $s \in \mathcal{S}_k, k \in \mathcal{K}$. Additionally we define,

$$c(\mathbf{x}, \ell) = \sum_{k \in \mathcal{K}} \sum_{s \in \mathcal{S}_k} d^s_k \sum_{m \in \mathcal{P}^s_k} \left[\delta^{s\ell}_{km} + \alpha^s_k \beta^{s\ell}_{km}\right] x^s_{km} \tag{19a}$$

and

$$t(\mathbf{x}, \ell) = \sum_{k \in \mathcal{K}} \sum_{s \in \mathcal{S}_k} \sum_{m \in \mathcal{P}^s_k} \left[\delta^{s\ell}_{km} + 1_{\{\alpha^s_k > 0\}} \beta^{s\ell}_{km}\right] x^s_{km}. \tag{19b}$$

Observe that $c(\mathbf{x}, \ell)$ and $t(\mathbf{x}, \ell)$ determine the present state of the constrained resources (allocated capacities and tunnels on link ℓ) for a given allocation sequence $\mathbf{x}$. A path m' of set $\mathcal{P}^s_k$ is said to be an *accessible path* from present allocation sequence $\mathbf{x}$ if

$$c(\mathbf{x}, \ell) + [\delta^{s\ell}_{km'} + \alpha^s_k \beta^{s\ell}_{km'}] d^s_k \leq C_\ell \quad \ell \in \mathcal{P}^s_k \tag{20a}$$

and

$$t(\mathbf{x}, \ell) + [\delta^{s\ell}_{km'} + 1_{\{\alpha^s_k > 0\}} \beta^{s\ell}_{km'}] \leq T_\ell \quad \ell \in \mathcal{P}^s_k. \tag{20b}$$

Hence setting the chosen $x^s_{km'} = 1$ does not violate constraints (5b) and (5c). We define a set $\mathcal{M}$ as the set of maximum allocation sequence, such that $\mathbf{x} \in \mathcal{M}$ means that for an allocation $\mathbf{x}$ their exists no unallocated demand ($w^{s'}_{k'} = 1$) with an accessible path m'.

The algorithm starts with $\mathbf{x} = \mathbf{0}$ and $\mathbf{w} = \mathbf{1}$. At each step, we either choose to $allocate(\mathbf{w})$ or to $deallocate(\mathbf{w})$ based on the current state of allocation ($\mathbf{x}$). For $x \notin \mathcal{M}$, we execute $allocate(\mathbf{x})$ subroutine, otherwise $deallocate(\mathbf{x})$. The routine $allocate(\mathbf{x})$ collects all the unallocated demands ($w^s_k = 1$) and amongst them randomly chooses a $s \in \mathcal{S}_k$ of $k \in \mathcal{K}$. All the paths in the set of candidate paths $\mathcal{P}^s_k$ are chosen in the order of increasing cost (ξ^s_{km}) and checked for accessibility. The first accessible path m' is allocated capacity and tunnels and corresponding variable $x^s_{km'}$ is set to 1.

The $deallocate(\mathbf{w})$, chooses to call subroutine $deallocate\_1(\mathbf{w})$ with probability $q(\mathbf{x})$ otherwise it calls $deallocate\_2(\mathbf{w})$. The $deallocate\_1(\mathbf{w})$ subroutine, randomly chooses a $s \in \mathcal{S}_k$ of $k \in \mathcal{K}$ with $w^s_k = 0$ and sets it to 1 and finds the path m' with $x^s_{km'} = 1$ and frees the resources (capacity and tunnels) used by the path and sets the $x^s_{km'}$ to 0. While $deallocate\_2(\mathbf{w})$ evaluates the current value of $c(\mathbf{x}, \ell)$ and $t(\mathbf{x}, \ell)$ and locates the critically loaded links. These links ($\ell' \in \mathcal{L}$) are either critically loaded in capacity (i.e. $c(\mathbf{x}, \ell') = C_{\ell'}$) or in tunnel requirement (i.e. $t(\mathbf{x}, \ell') = T_{\ell'}$). Next, it locates a $s \in \mathcal{S}_k$ and $k \in \mathcal{K}$ with a path m' such that $\delta^{s\ell'}_{km'} x^s_{km'} = 1$ or $(1_{\alpha^s_k > 0}) \beta^{s\ell'}_{km'} x^s_{km'} = 1$. For the chosen path m' of service class $s \in \mathcal{S}_k$ and demand $k \in \mathcal{K}$, it relinquishes the capacity and tunnels used by $x^s_{km'}$ and sets $w^s_k = 1$ and $x^s_{km'} = 0$.

Algorithm 2 Simulated Allocation Approach

```
    $gSA()$step $\leftarrow$ 0 count $\leftarrow$ 0 $F^* \leftarrow \infty$ $(\mathbf{x}, \mathbf{w}) \leftarrow (\mathbf{0}, \mathbf{1})$
    while (step $<$ step$_{max}$ AND $F^* > F^*_{min}$) do
    step = step + 1
    if $x \in \mathcal{M}$ then
      $allocate(\mathbf{w})$
    else
      if random $\leq q(\mathbf{x})$ then
        $deallocate\_1(\mathbf{w})$
      else
        $deallocate\_2(\mathbf{w})$
      end if
      if $f(\mathbf{x}, \mathbf{w}) < F^*$ then
        $F^* = f(\mathbf{x}, \mathbf{w})$
        $(\mathbf{x}, \mathbf{w})_{min} \leftarrow (\mathbf{x}, \mathbf{w})$
      end if
    end if
  end while
  return $(\mathbf{x}, \mathbf{w})_{min}$
```

The value of $q(\mathbf{x})$ plays an important role in the convergence and solution quality of the algorithm. We know that $q(\mathbf{x})$ should depend upon the value of the objective function as compared to the F^*_{min} and for our implementation we compute

$$X(\mathbf{x}) = \frac{f(\mathbf{x}, \mathbf{w}) - F^*_{min}}{F^*_{min}}.$$

We choose the value of $q(\mathbf{x})$ using a fairly simple technique. We derive a reasonable value for threshold X^* based on our anticipated proximity of F^* to F^*_{min}. If we set X^* to 0.1 then we expect to find solutions with in 10% of F^*_{min}. It depends upon what can be considered as acceptable solution. It also depends upon the way in which F^*_{min} was computed. Then, we set $q(\mathbf{x}) = q_1$ if $X(\mathbf{x}) < X^*$ and $q(\mathbf{x}) = q_2$ otherwise. For our case, F^*_{min} was computed as the solution obtained by relaxing the integrality constraint of the problem (**P**). Since, we can not be sure that an integral solution exists in the proximity of F^*_{min}, we use $X^* = 0.1$. When in the acceptable solution region, whether the algorithm allocates randomly chosen demand or de-allocates randomly chosen demand depends on the value of q, we use $q_1 = 0.95$ and $q_2 = 0.8$.

The values of F^*_{min} needs to be determined to ensure acceptable quality results within reasonable amount of time. As already discussed , we use F^*_{min} as the solution obtained by relaxing the integrality constraint of the problem (**P**). The set $\mathcal{M}$ contains the pairs $(\mathbf{x}, \mathbf{w})$ which are maximal allocations, hence given the present partial allocation sequence, no more flows can be accepted without violating the constraints (5b) and (5c). We set the step$_{max} = 10000$. During the experiments conducted, it was found that the algorithm performs fairly well and leads to improvements in the results obtained by the Algorithm 1 (gIP()).

A component that feeds into the model (of both the heuristics) is the generation of candidate cycle paths. It may be noted that Suurballe and Tarjan have developed an algorithm for generating shortest pair of disjoint paths [21], [22]; this, however, helps in generating only the shortest cycle, not a set of candidate cycles. In our case, we have implemented a simple procedure by extending the K-shortest path algorithm where the K paths generated by the K-shortest path algorithm are compared to each other to filter out common links to generate a set of candidate cycles containing only disjoint pair paths. Note that the candidate set is *only* a feeder to the optimization model, and certainly, the eventual solution can depend on how many are included in the initial cycle (this is discussed later). Note that this candidate list needs to be generated only once. For the set of examples tested (discussed below), we have found this procedure to be very efficient (only takes a couple of seconds of computing time).

V. Results and Discussion

We have implemented our Successive Approximation Algorithm ($gIP()$) in C^{++} using CPLEX callable libraries [23] to solve the lp relaxations. We have also implemented Simulated Allocation Algorithm ($gSA()$) using C^{++}. The goal of this section is to understand the effectiveness of the problem formulation in solving the survivable BAG traffic engineering design problem. We want to evaluate both the variations ($\mathbf{P}_h$) and ($\mathbf{P}_s$) of the basic problem. Specifically, we are interested in understanding the effect of the utility function, the benefit of soft vs. hard survivable strategies, and whether the unified functions f_h and f_s are effective in capturing the multiple objective criteria.

Consider experimental networks shown in Figures 2− 5. These networks are taken from already published literature [24], [2]. For these experimental networks we provide detailed results and help user derive insights into the behavior. EN I has 12 nodes, 18 edges and average nodal degree (ratio of number of edges to number of nodes) of 1.5. EN II has 6 nodes, 12 links and an average nodal degree of 2.0. EN III has 12 nodes, 25 links and an average nodal degree of 2.08. EN IV has 10 nodes, 26 links and an average nodal degree of

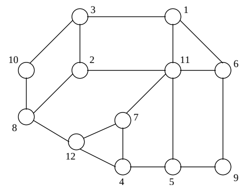
Fig. 2. Experimental Network I

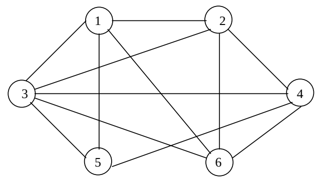
Fig. 3. Experimental Network II

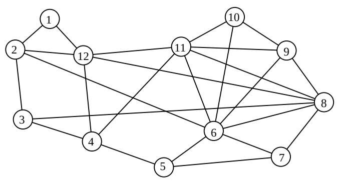
Fig. 4. Experimental Network III

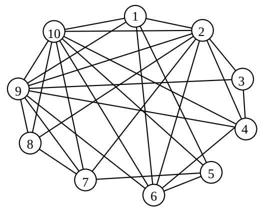
Fig. 5. Experimental Network IV

2.6. Observe that EN IV is the most well connected network where as EN I is the least.

For the given experimental networks, we consider capacity of 622 Mbps for each link (referred to as baseline capacity). We assume that three service classes are present in the network. We also assume that there are demands between all pairs of nodes of value 100 Mbps for each of the three considered service classes. These service classes have varied survivability requirements and corresponding utility to the service provider. We assume that the demands with higher survivability requirement have higher utility. We assume that service class 1 $(s = 1)$ is survivable critical in nature and hence for all $k \in \mathcal{K}$, α^1_k are chosen to be 1.0 and u^1_k is chosen as 5.0. We assume that service class 2 $(s = 2)$ also has a survivability requirement, although not as stringent as service class 1. Hence, we choose α^k_2 for all $k \in \mathcal{K}$ as 0.5 and u^2_k to be 3.0. Service class 3 does not have any survivability requirement and we choose for all $k \in \mathcal{K}$, $\alpha^3_k = 0$ and u^1_k to be 1.0. The penalty cost for each demand and service class η^s_k is computed as $\eta^*(1+\alpha)d^s_k$, where η^* is a weighing constant.

A. Comparison of gIP and gSA

We use the experimental networks for evaluating the solution quality of Heuristic I and II. The formulation ($\mathbf{P}_h$) is used for such a comparison. We assume $\gamma = 0.5$ and $\theta^s_k = 0.5$ for all $s \in \mathcal{S}_k$ and $k \in \mathcal{K}$. We choose the utility weighing factor $R = 100$. The penalty cost weighing factor η^* was chosen to be 5. The number of allowed tunnels on each link are assumed to be 50, 15, 25, 20 for EN I, II, III, IV, respectively. The number of candidate cycles were chosen to be 5 for EN I and 15 for others. We start with these values of parameter and experimental networks with baseline capacity and run both gIP() (HI) and gSA() (HII). We also use a hybrid heuristic (HIII) where we first derive results using HI. We then use the final solution of H1 as initial solution for HII, which is further

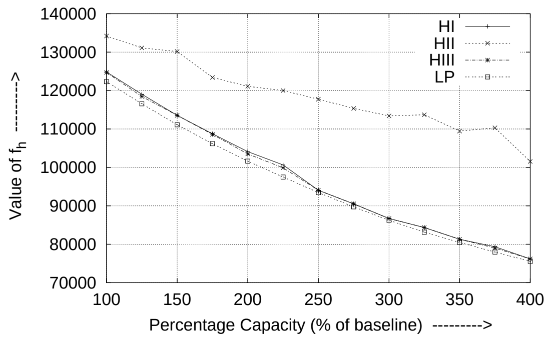

Fig. 6. Value of f_h for EN I with increasing capacity and $T_\ell = 50$

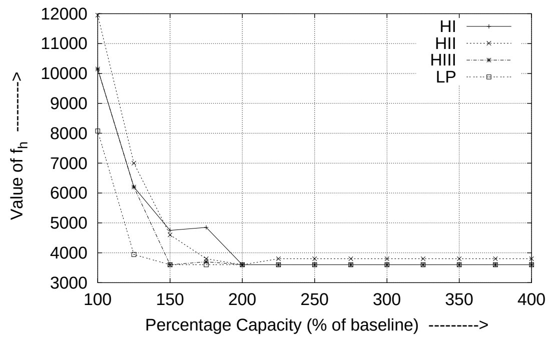

Fig. 7. Value of f_h for EN II with increasing capacity and $T_\ell = 15$

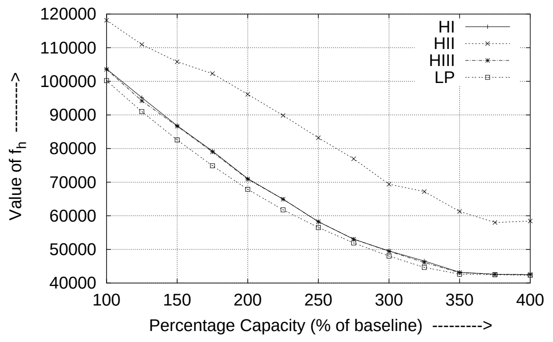

Fig. 8. Value of f_h for EN III with increasing capacity and $T_\ell = 25$

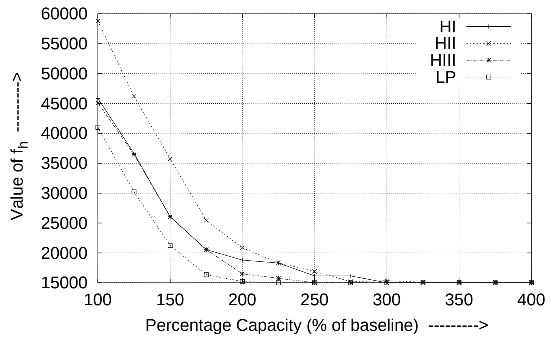

Fig. 9. Value of f_h for EN IV with increasing capacity and $T_\ell = 20$

improved by the heuristic II. We also solve the continuous relaxation of the IP variables in the ($\mathbf{P}_h$) problem which serves as a lower bound on the integer solution obtained.

We then increment the number of Tunnels by 10 and rerun the problem. We run the experiments till allowed tunnels are 50. We then increase the capacity to 125% of the baseline capacity and repeat the entire process. We continue till the network reaches 400% of the baseline capacity. Such set of experiments help us explore the solution quality under both tunnel and capacity constrained (both are small), to one of them being constrained and other relaxed, to both of them being relaxed (both are high). We present results for the experimental networks EN-I and EN-IV in plots presented in figures 6 to 9. We also present results for the continuous relaxation of the integer problem.

In figure 6, we present the value of f_h when $T_\ell = 20$. Results for other values of T_ℓ showed similar behavior. Observe that HI and HIII closely follow the LP solution. However, the performance of HII is not quite as good. Similar behavior is observed for EN III in figure 8. Based on these results, it seems that Heuristic II is not very useful and that HII and HIII hardly give better performance than HI.

However, for EN II with $T_\ell = 10$, the results presented in figure 7 draw our attention to the point that this may not be true in general. Similar behavior was observed for EN IV with $T_\ell = 20$ which is presented in figure 9. Similar results were obtained for other values of T_ℓ. For these two cases, we can observe that HII gives better results than HI. Interestingly, HIII closely follows the minimal of the two heuristics (I and II) and the continuous relaxation. Such an observation gives strength to the performance of heuristic III for more general scenarios. Hence, we use HIII to compute the best solution in the remaining of the results in this paper.

The results also demonstrate that capacity and tunnels are equally important while provisioning a network. Presence of fewer number of tunnels nullifies the presence of abundant capacity leading to under utilized links. More so, having too many tunnels is only as much useful as the amount of capacity available in the network. Hence, our first inference is that accounting for both capacity and tunnels leads to effective traffic engineering solutions. Both of them could be viewed as resource which impact the amount of traffic carried by a network. While engineering a network if we consider capacity in isolation, the results can have a limited utility in practical networks. More so, when used in real life networks, the performance observed might be much inferior to the expectations.

B. Effectiveness of the Formulation ($\mathbf{PA}_h$*)*

The optimization problem $\mathbf{P}_h$ was formulated so as to incorporate various conflicting objectives into an integrated problem. Not only was it important to allocate the BAG requests but also the residual capacity on the links which was to be used by best effort traffic. Hence, the effectiveness of the formulation can be studied in terms of the qualities

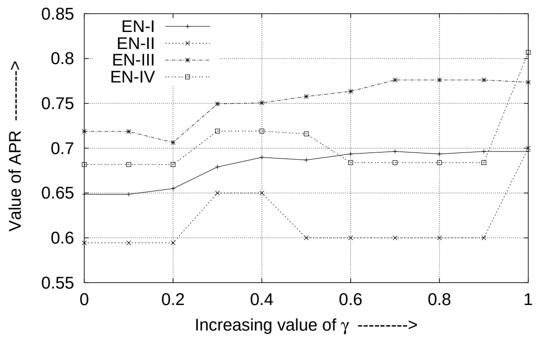

Fig. 10. Value of APR for experimental networks with increasing γ

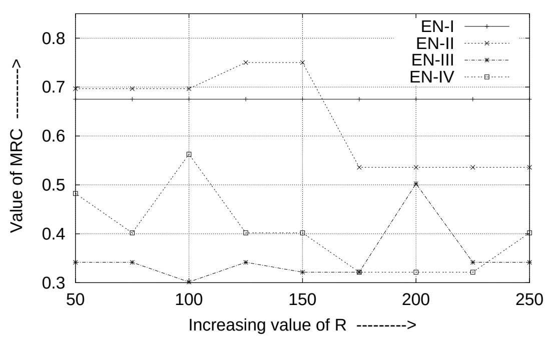

Fig. 11. Value of MRC for experimental networks with increasing R

of the obtained solution. Though there are many possible qualities that can be considered for this study, we choose four parameters which are γ, R, θ_k^s and η^*. Observe that these four parameters can be tuned by a service provider while traffic engineering a network and it would certainly have effect on the solution found. Based on the traffic engineering considerations, values of these parameters can be chosen. We evaluate in these set of experiments if the parameters influence the final solution in the intended way i.e. provide anticipated effect on a specific metric. We used the number of allowed tunnels as $T_\ell = 15, 20, 50, 50$, the # of candidate cycles as 5, 15, 15, 15 and the capacity of the links as 150%, 200%, 400%, 500% of the baseline capacity for experimental networks I, II, III, IV, respectively. The default values were chosen as: $\gamma = 0.5$, R=100, θ_k^s = 0.5 and $\eta^* = 5$.

1) Dependence on γ: We evaluate the role of γ in the first set of experiments. Observe that γ is the cost proportional constant for the routing cost of primary and backup paths. In other words, it determines the routing cost of the backup path as compared to the primary path. When γ is set to 0, their is no routing cost for backup path where as when it is set to 1, backup path has equal cost as the primary path. For experiment with increasing value of γ, we compute the average of the ratio of the length of primary path to backup path (APR) for all service classes and demands with survivability requirement ($\alpha > 0$). We present results in figure 10 for the experimental networks.

Observe that with the increasing value of γ, APR increases. This can be attributed to the following reasons: firstly, the routing cost on the primary (equation 12a) and backup paths (equation 12b) are computed based on the hops and the demand volume of a request. While computing the routing cost of a couple, we multiply the cost of backup path with γ and sum it to the routing cost of the primary path. Increasing value of γ causes increase in the contribution of the backup path towards the cost of a couple. Thus couples with longer backup paths, will end up having higher cost (assuming that other costs remain as before). Such a change forces the optimization problem to choose path pairs with shorter backup paths in terms of hops. Hence, the metric APR increases with increasing value of γ.

2) Dependence on R: We study the impact of the utility weighing parameter R on the allocation of demands. Although the individual relative utility (u_k^s) is mostly determined based on many considerations like importance of the client requesting the services, characteristics of the incoming traffic, etc. but the parameter R determines the overall utility as compared to routing costs and bandwidth/tunnel requirement costs. Since, we are also concerned about the residual bandwidth on the links which carries the best-effort traffic, the parameter R has deeper implication. Hence, we compare the load on the link with Minimum Residual Capacity (MRC) for increasing values of R. We present results in figure 11 for the experimental networks.

The value of R changes the chosen solution in two ways. Firstly, for some demands which were not allocated because for them $\xi_{km}^s - u_k^s R > \eta_k^s$ for all accessible paths and hence the formulation chooses to refuse them (although their is enough capacity and tunnels in the network). By increasing the value of R, one/many of the paths for some of these demands become acceptable and given that the network has enough bandwidth and tunnels, they are accepted. Hence, leading to an increase in MRC. Secondly, increasing R also changes the ordering of demands with respect to their overall costs ($\xi_{km}^s - u_k^s R$). If we consider the overall cost of the least cost acceptable path (m') for a request (d_k^s) as a function of R, we find that $\xi_{km'}^s$ is the intercept and u_k^s is the slope of the overall cost. Hence, increasing value of R changes the relative profitability of demands with respect to each other. In our case, this is more class based as the value of u_k^s is determined based on the class of the request.

The first behavior can be observed at higher value of R, since thats when it is large enough to make such impacts. When it leads to acceptance of a demand, the value of MRC obviously increases. However, for smaller value of R, the second observation plays the crucial role. Due to changes in ordering of demands, MRC follows no specific pattern, rather it sometimes increases and sometimes decreases depending upon the links used by the minimal cost acceptable path (m') of the now more profitable demand.

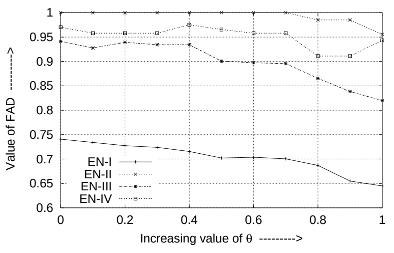

Fig. 12. Value of FAD with increasing θ

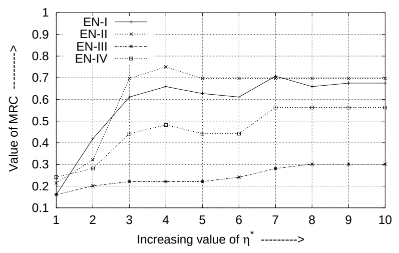

Fig. 13. Value of MRC with increasing η^*

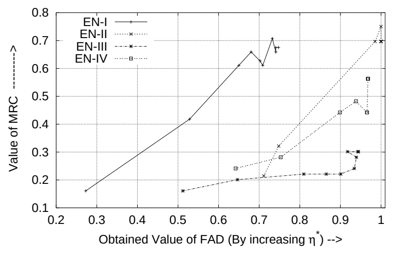

Fig. 14. Value of MRC with increasing FAD (by changing η^*)

3) Dependence on θ_k^s: We experiment on the characteristic of the final solution based on the chosen values of θ_k^s's. Observe that θ_k^s controls the importance given to the allocation cost of a request as compared to its normalized routing cost. When set to 0, only routing cost is accounted for in the optimization formulation. When set to 1, both have the same weight and hence play equally important role. Due to its role in determining the cost of a request, we evaluate it in terms of fraction of accepted demands (FAD). We present results for experimental networks in figure 12.

The parameter impacts the solution in a very direct way. As we increase the value of θ, the overall cost (ξ_{km}^s) goes on increasing. Due to this increase, for some demands the penalty cost is less than the minimum cost accessible cycle. Such a scenario forces the formulation to reject the demand. Such a behavior is evident in the figure, where FAD for each network decrease with increasing value of θ. More so, the cycles with higher number of hops pay heavier penalty.

Since the impact of increasing θ is only realized when the overall cost of the minimal cost accessible path increases above the penalty cost for rejecting the demand (η_k^s). Hence, we find regions in values of θ such that FAD is unaltered.

4) Dependence on η^*: In the next study, we estimate the role of the parameter η^* which attributes a penalty with the refusal to carry a specific demand request. The value of this parameter depends upon the importance given towards carrying a demand vis-a-vis the cost of carrying the demand. For a sufficiently high value of η^*, the network would accept all the demands that it can carry, leaving minimal or no bandwidth for best-effort traffic. However, if we choose very low value for η^*, most or all of the demands will be rejected and network would be largely under utilized. Hence, we compare the value of Minimum of the Residual Capacity on the links to the chosen value of η^*. We present results for experimental networks in figure 13. Alongside, we also present corresponding values of MRC for increasing value of fraction of accepted demands (FAD) obtained by changing the values of η^* in figure 14.

Observe that the penalty cost for refusing a demand (η_k^s) is based on the demand volume, demand survivability requirement and the parameter η^*. When deciding whether to allocate a demand or not, the formulation compares the minimum value of the overall cost ($\xi_{km}^s - u_k^s R$) for all the accessible candidate cycles with the value of η_k^s. The demand is only allocated if the latter is more. When increasing the value of η^*, we certainly make some demands more likely to be allocated under the assumption that enough capacity and tunnels are available. Hence, it is natural to expect that FAD increases with η^* as observed in 13. Observe that it does not ensure that the demand is allocated, rather it ensures that the formulation attempts to allocate it. However, the allocation (FAD) only goes as far as permitted by the amount of capacity and tunnels available in the network.

Interestingly, the value of MRC does not increase linearly or consistently with the increase in FAD as shown in 14. More so, the value of MRC shows varied behavior. For example, in EN II, to increase FAD from 0.75 to 0.95 requires almost doubling of MRC that is to say that at least half of the best-effort traffic on the most congested link needs to be thrown away. However, for EN III, to increase FAD from 0.5 to 0.95, MRC only requires to be changed from 0.18 to 0.22 which is hardly a performance loss for best-effort traffic. Moreover, for EN I, for some values of FAD, MRC decreases for increase in FAD, which is counter-intuitive. We would like to assert that the actual trade-off between FAD and MRC is dependent on many factors including network topology, traffic pattern, service classes, capacity availability and the tunnels.

The variations and trade-off's depicted in this subsection go on to show that the formulation effectively captures the role of various parameters and responds positively to the corresponding changes in the parameters. A network operator can choose appropriate values of these parameters based on his/her experience and requirements from the network. The derived solution depends upon the values chosen and adequately accommodates the intention of the service provider.

C. Choice of # of Candidate Cycles

A candidate cycle is a pair of node/link disjoint path between ingress and egress nodes of the BAG request. Of the two paths in the cycle, the path with lower number of hops is chosen as the 'primary' path. The number of possible candidate cycles varies from one request to the other depending on the ingress-egress nodes of the request and the topology of the network. Recall that the size of the optimization problem ($\mathbf{P}_h$) depends not only on the number of BAG service requests but

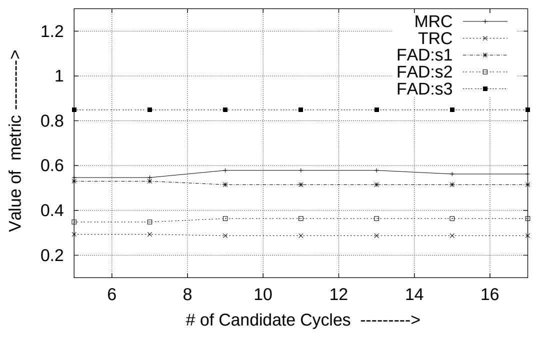

Fig. 15. Results for EN I with $T_\ell = 25$ with increasing # of candidate cycles

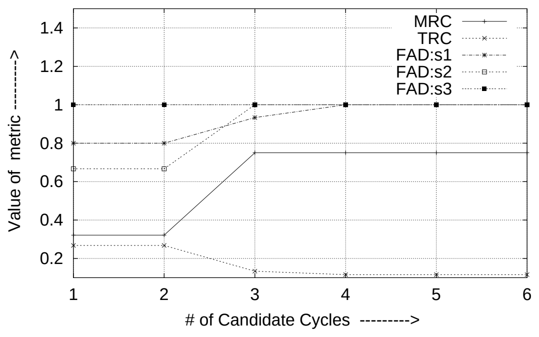

Fig. 16. Results for EN II with $T_\ell = 10$ with increasing # of candidate cycles

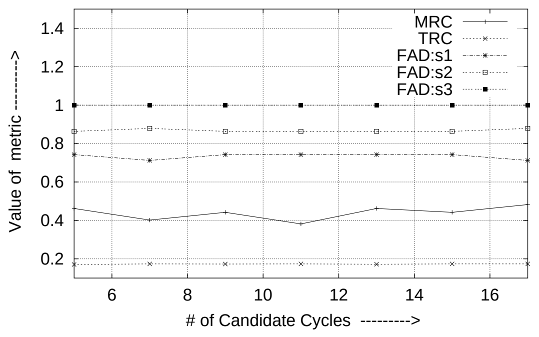

Fig. 17. Results for EN III with $T_\ell = 25$ with increasing # of candidate cycles

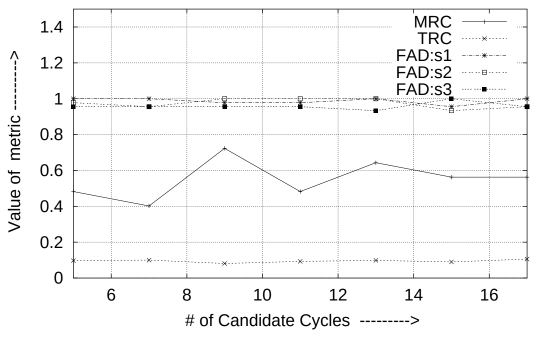

Fig. 18. Results for EN IV with $T_\ell = 20$ with increasing # of candidate cycles

also on the number of candidate cycles considered for each request. On one hand, to make the size of the optimization problem manageable, we have to limit the number of candidate cycles for each request. On the other hand, setting a low limit on the number of candidate cycles can result in higher cost of the final solution in terms of the value of the objective function. In figures 15- 18, we present the value of Minimum Residual Capacity (MRC) of links and Total Residual Capacity (TRC) in the network and FAD for each service class for increasing # of candidate cycles.

Observe that FAD does not necessarily increase with increasing # of candidate cycles. This can be attributed to the fact that the candidate cycles are increasing in length in terms of number of hops. That is to say that $(n+1)^{th}$ cycle has either the same number of hops or more than the n^{th} cycle. Due to overall cost considerations, when the formulation chooses to allocate a demand on its longer path (which was previously not present), it takes away the resources which could have been allocated to many other demands. Such a choice leads to a decrease in the value of FAD. At other times, presence of more candidate paths makes it possible for a demand to be routed over under utilized links (in terms of capacity and tunnels) and consequently leads to an increase in FAD.

Note that the metric MRC is affected by # of candidate cycles in an indirect way. Due to increase in the value of FAD, MRC is consequently increased. Such a behavior has also been observed in figure 14 and is quite natural to expect. However, when considering the increase in options (of choosing a path) while maintaining the same value of FAD, the impact on MRC is interesting. Observe that the formulation does not have any cost or penalty towards increased link utilization and hence when considering the paths for possible allocation, the presence of required resources is the only consideration. Consider the figure 17, observe that for K = 9, 11 and 13, the value of each FAD is same. However at K=11, the value of MRC is less which can be attributed to the allocation of a demand to a path which passes over under utilized links. But when K=13, the value of MRC again increases. Hence, the increase in MRC is not accounted for by the formulation but it does effect the final solution in many cases.

The metric total residual capacity is captured by the objective function f_h and hence is minimized by the formulation. Observe that the costs c^{sp}_{km} and c^{sb}_{km} are in terms of the used capacity by the primary and backup paths. Hence, barring the effect of parameters like θ and γ, the overall cost of a cycle ξ^s_{km} is proportional to the capacity used by the cycle. The TRC stays at minimal unless the cycles provided are too few.

D. *Impact of # of Tunnels*

In this subsection, we study the impact of number of tunnels on each link on the solution of the optimization problem. For this study we use the formulation ($\mathbf{P}_h$). We observe the value of Minimum of residual capacities of links (MRC) and the fraction of accepted demands (FAD) for increasing network capacity. We observe the results for increasing number of

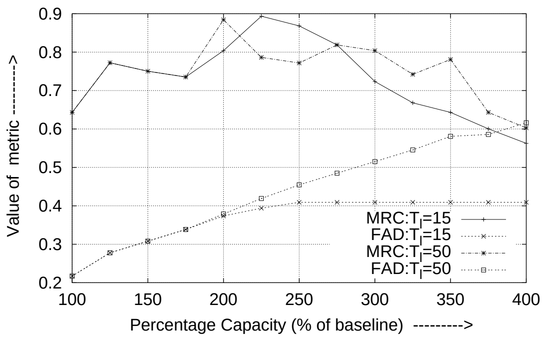

Fig. 19. MRC and FAD for EN I with increasing capacity

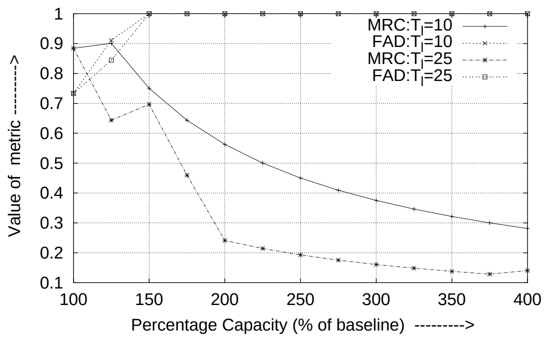

Fig. 20. MRC and FAD for EN II with increasing capacity

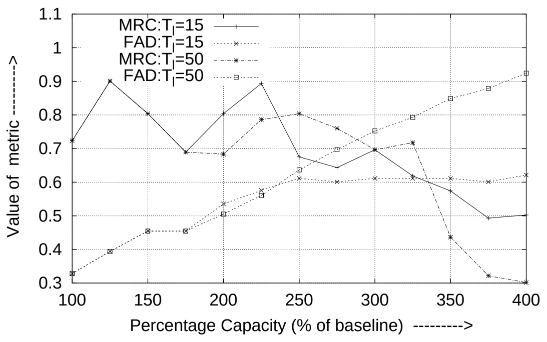

Fig. 21. MRC and FAD for EN III with increasing capacity

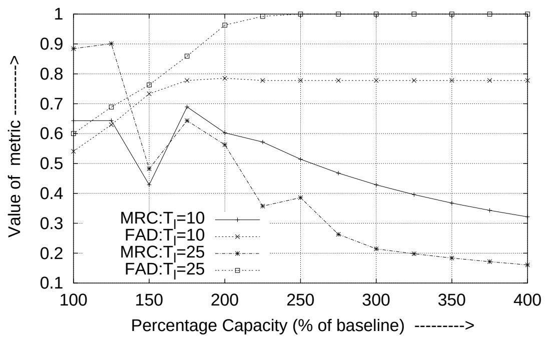

Fig. 22. MRC and FAD for EN IV with increasing capacity

tunnels and present the representative scenarios for experimental networks in figures 19- 22. We have used the value of parameters as $R = 100$, $\eta^* = 10$, $\theta = 0.5$ and $\gamma = 0.5$. We obtained results for $T_\ell = 10, 15, 20, 25, 50$, only.

Observe that the value FAD increases with increasing capacity until it stabilizes based on the allowed number of tunnels on the links. There of excess amount of capacity is of no consequence since the links are congested in terms of number of tunnels. Increasing the number of tunnels on each link affects the solution in two ways. One way is direct increase in the value of FAD and a consequent increase in the value of MRC. The acceptance of more demands makes links more utilized and leave lesser bandwidth for best-effort traffic. Second way is that it allows demands to use lesser hops paths which had excess capacity but no extra tunnels. Due to lack of tunnels on lesser hop paths, the demand had to previously take a longer hop path and in the process lead to a solution with higher value of MRC. With tunnels increasing on links, many more shorter paths (mostly less in overall cost too) become accessible.

So much so is the impact of tunnels on the solution that in some cases we find that although the FAD is higher with more tunnels, the value of MRC is still lower. This is attributed to the above mentioned behavior. As demands begin to choose cycles with fewer hops, more and more capacity gets freed out in the network. This capacity is used by other demands in a similar efficient way. Similar behavior on the part of all the demands leads to solutions with smaller value of MRC.

The increase in capacity of links also shows similar behavior. Starting with less capacity, as we increase it initially, the value of MRC decrease very fast (assuming same FAD) since some of the demands using the longer paths can now be moved on to shorter paths which leads to significant decrease in the value of MRC. However, once all the demands are moved to minimum hop cycles, the subsequent decrease in the value of MRC is only obtained by the increase in the capacity of each link.

The behavior of EN II depicted in figure 20 is interesting. Observe that the value of FAD reaches 1.0 at 150% of baseline capacity at which value MRC is almost equal for both the cases ($T_\ell = 10, 25$). Now, when we increase the capacity to 200%, the value of MRC for $T_\ell = 25$ falls dramatically as compared to $T_\ell = 10$. This again is due to the freed up capacity in the network due to demands shifting from longer to smallest hop cycle. At 200% of the baseline capacity, all the demands have got to smallest hop cycles and hence there of the decrease is at the same rate as $T_\ell = 10$.

E. Effectiveness of formulation ($\mathbf{PA}_s$*)*

In this subsection, we study the soft requirement formulation on its capability to incorporate the various objectives in an integrated fashion. Observe that soft requirement is based on the circumstances that the backup paths are only found *apriori* but are not reserved, rather they are used by best-effort traffic. In the event of failures, the effected demands are routed over

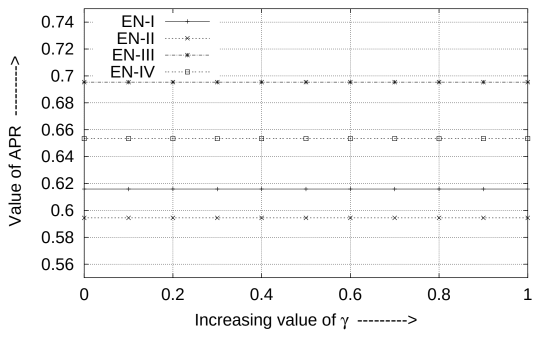

Fig. 23. Value of APR for experimental networks with increasing γ

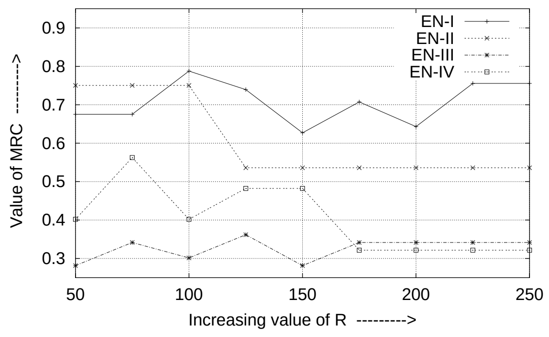

Fig. 24. Value of MRC for experimental networks with increasing R

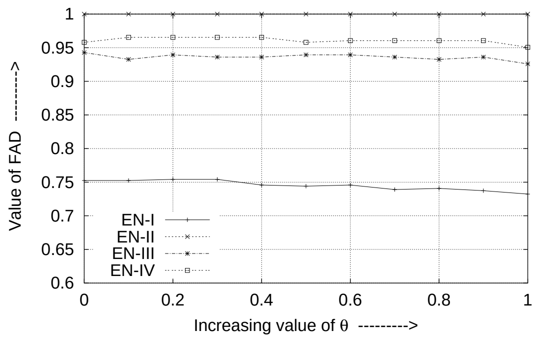

Fig. 25. Value of FAD for experimental networks with increasing θ

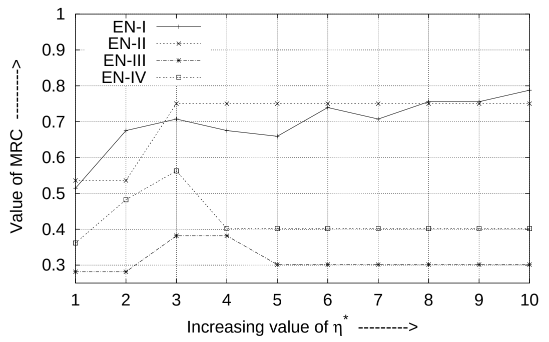

Fig. 26. Value of MRC for experimental networks with increasing η^*

the already chosen backup paths after cleaning them of the best-effort traffic. Hence, their is no cost of reserving the backup paths since the capacity is still used by best-effort traffic. Such a difference fundamentally changes the way the formulation ($\mathbf{P}_s$) is affected by changes in the parameters. In figures 23-26, we present results on impact of changes in parameters γ, R, θ and η^* upon the nature of the final solution. We used the number of allowed tunnels as $T_\ell = 15, 20, 50, 50$, the # of candidate cycles as 5, 15, 15, 15 and the capacity of the links as 150%, 200%, 400%, 500% of the baseline capacity for experimental networks I, II, III, IV, respectively. The default values were chosen as: $\gamma = 0.5$, R=100, θ^s_k = 0.5 and $\eta^* = 5$.

Since, the formulation assumes no cost for the backup path, the parameter γ has no role to play. Hence, we find that in figure 23, in the chosen couple for allocation, the average ratio of the primary path length to the backup path length (APR) stays same for all the values of γ. But when comparing the plot with that of hard requirement in figure 10, we find that the values are smaller even as compared to $\gamma = 0.0$. Such a behavior can be attributed to the lack of cost from the backup paths. Since, their is no additional cost for the backup path, the formulation chooses the couple with least cost primary path (cost is directly proportional to hops) and does not care about the cost of the backup path. This leads to smaller values of APR in the final solution.

The impact of increase in R on the minimum residual capacity (MRC) is shown in figure 24. The behavior observed is similar as the hard requirement formulation 11, although their are minor differences. Here, we have smaller values of ξ^s_{km} since they do not include the cost of backup path and hence smaller values of R provide the same over all cost to a demand. This leads to similar behavior of soft requirement formulation at smaller values of R as that of hard requirement formulation at higher values of R.

Similarly for increasing value of θ, we observe that the behavior shown in 25 is similar to that of hard requirement formulation 12. On closer look, we observe that the impact of θ has been diluted. The decrease in the value of FAD is much less than the one observed for hard requirement. This can be attributed to the absence of routing cost for the backup paths. Observe that θ determines the routing cost vis-a-vis allocation cost of a cycle. Since for soft requirement, the cost of cycle is reduced to that of the primary path, the impact of θ towards the overall cost is there of decreased.

The impact of increase in η^* on the FAD is shown in figure 26. The performance is similar to that of hard requirement scenario 13. Here also we observed that the minor differences between them are due to the relative change of the value of ξ^s_{km}. Now, they only have the cost of primary path (no cost for backup path) and hence smaller values of η^* provide same relative over all cost to a demand. This leads to similar behavior of soft requirement formulation at smaller values of η^* as that of hard requirement formulation at higher values.

The soft requirement formulation ($\mathbf{P}_s$)gives adequate importance to the parameters R, θ and η^*. We mostly observed the dilution of impact of parameters on the solution as compared to the hard parameter requirement ($\mathbf{P}_h$). Since the soft requirement captures a subset of parameters as compared to hard requirement, we expect that similar behavior will be observed for various scenarios. Evaluating the impact of # of candidate cycles, # of tunnels, etc. was deemed as unnecessary since we expected the trend of dilution of impact to continue.

VI. Summary

In this paper, we consider the problem of traffic engineering a heterogenous network supporting services with varied survivability requirements. The heterogeneity aspect of the network is accounted by considering a tunneling constraint on the links. The varied survivability requirement is incorporated in the model using a cycle path concept so that service classes with different survivability needs can still be modeled in the same framework. We also introduce different objectives, with seemingly different goals, into a unified, single objective function. The Objectives considered not only considered the allocation of the service requests but also the bandwidth available for the best-effort service class. Moreover, we have also addressed the hard vs. soft survivability requirements.

We have presented two heuristics to sole the integrated formulation which was an IP(integer program) in nature. First heuristic is iterative in nature and uses continuous relaxations of the problem to derive integer solutions. Second heuristic is based on Simulated Allocation technique. We compare the two heuristics and show that a hybrid approach combining both the methods works quite well. We then proceed to show the effect on various factors that are integral to the integrated traffic engineering problem. We have presented extensive experimental results showing that solving the problem based on the aggregated objective function can satisfy other objectives and criteria in a satisfactory manner. We also discuss the interplay between various parameters and resources and show their relative impact on each other. Tradeoff between accepting new requests of survivable classes and the residual bandwidth for the best-effort services was also evaluated. The results also showed that capacity and tunnels have an equal role in ensuring effective traffic engineering of a network.

Acknowledgements

We would like to thank Michał Pióro for his suggestions towards the application of Simulated Allocation method for solving the formulation.

References

[1] B. Davie and Y. Rekhter, *MPLS: Technology and Applications*, Morgan Kaufmann, 2000.

[2] S. Srivastava, B. Krithikaivasan, C. Beard, D. Medhi, A. Van de Liefvoort, W. Alanqar, A. Nagarajan *Benefits of Traffic Engineering using QoS Routing Schemes and Network Controls*, to appear in Computer Communications, June 2002.

[3] S. Srivastava, B. Krithikaivasan, D. Medhi, M. Pioro, *Traffic Engineering in the presence of Tunneling and Diversity Constraints: Formulation and Lagrangean Decomposition Approach*, proceedings of ITC 18, January 2003.

[4] A. Fumagalli, I. Cerutti, M. Tacca, F. Masetti, R. Jagannathan, S. Alagar, *Survivable Networks Based on Optimal Routing andnd WDM Self-Healing Rings*, Proc. of IEEE INFOCOM '99, Mar 1999, Ney York, USA.

[5] B. Gavish, P. Trudeau, M. Dror, M. Gendreau, Lorne Mason, *F*iberoptic Circuit Network Design under Reliability Constraints, IEEE Journal on Selected Areas of Communication, Vol. 7, No. 8, pp. 1181-1187, 1989.

[6] B. Gavish, I. Neuman, *Routing in a Network with Unreliable Components*, IEEE Transactions on Communications, Jul 1992, Vol. 40, No. 7, P. No. 1248 - 1258.

[7] Y. Kajiyama, N. Tokura, K. Kikuchi, *An ATM VP-based Self Healing Ring*, IEEE Journal on Selected Areas of Communication, Vol. 12, No. 1, pp. 171-187, 1994.

[8] R. Kawamura, K. Sato, I. Tokizawa, *Self Healing ATM Networks Based on Virtual path Concept*, IEEE Journal on Selected Areas of Communication, Vol. 12, No. 1, pp. 120-127, 1994.

[9] D. Medhi, *A Unified Approach to Network Survivability for Teletraffic Networks: Models, Algorithms and Analysis*, IEEE Transactions on Communications, Vol. 42, pp. 534-548, 1994.

[10] D. Medhi and R. Khurana, *Optimization and Performance of Network Restoration Schemes for Wide-Area Teletraffic Networks*, Jouranl of Network and Systems Management, Vol. 3, pp. 265-294, 1995.

[11] S. Ramamurthy, B. Mukherjee, *Survivable WDM Mesh Networks, Part I - Protection*, Proc. of IEEE INFOCOM, March 1999, Ney York, USA.

[12] Y. Xiong, L. G. Mason, *Restoration Strategies and Spare Capacity Requirements in Self Healing ATM Networks*, IEEE/ACM Transactions on Networking, Vol. 7, No. 1, pp. 98-110, 1999.

[13] T. H. Wu, *Fiber Network Service Survivability*, Artech House.

[14] Y. Wang and Z. Wang, *Explicit Routing Algorithms for Internet Traffic Engineering*, Proc. of IEEE ICCCN, March 1999, New York, USA.

[15] M Kodialam and T. V. Lakshman, *Dynamic routing of bandwidth guaranteed tunnels with restoration*, Proc. of IEEE INFOCOM, March 2000, Tel Aviv, Israel.

[16] A. Nagarajan, Sprint Corporation, private communication.

[17] D. Medhi, *Diverse Routing for Survivability in a fiber-based Sparse Network*, Proc. of IEEE ICC, June 1991, Denver, CO, USA.

[18] M. Pioro, P. Gajowniczek, *Simulated allocation: a suboptimal solution to the multicommodity flow problem*, 11^{th} IEE Teletraffic Symposium on Performance Engineering in Telecommunications Networks, March 1994, UK.

[19] M. Pioro, P. Gajowniczek, *Solving multicommodity integral flow problems by simulated allocation*, Telecommunication Systems, 7(1-3), pp. 17-28, 1997.

[20] M. Pioro, Warsaw University, private communication.

[21] J. W. Suurballe, *Disjoint Paths in a Network*, Networks, Vol. 4, pp. 125-145, 1974.

[22] J. W. Suurballe, R. E. Tarjan, *A Quick Method for Finding Shortest Pairs of Disjoint Paths*, Networks, Vol. 14, pp. 325-336, 1986.

[23] *Manual of Cplex Callable Libraries*, ILOG Corporation, USA.

[24] D. Medhi, D. Tipper, *Some approaches to solving a multihour broadband network capacity design problem with single-path routing*, Telecommunications System, 13, pp 269-291, 2000.