

Determining Link Weight System under Various Objectives for OSPF Networks using a Lagrangian Relaxation-based Approach

Shekhar Srivastava, Gaurav Agrawal, Michał Pióro[†], Deep Medhi*

University of Missouri–Kansas City, USA

[†]Warsaw University of Technology, Poland and Lund University, Sweden

Abstract—An important traffic engineering problem for OSPF networks is the determination of optimal link weights. Certainly, this depends on the traffic engineering objective. Regardless, often a variety of performance measures may be of interest to a network provider due to their impact on the network. In this paper, we consider different objectives and discuss how they impact the determination of the link weights and different performance measures. In particular, we propose a composite objective function; furthermore, we present a Lagrangian relaxation-based dual approach to determine the link weight system. We then consider different performance measures and discuss the effectiveness of different objectives through computational studies of a variety of network topologies. We find that our proposed composite objective function with Lagrangian relaxation-based dual approach is very effective in meeting different performance measures and is computationally very fast.

Index Terms—OSPF Networks, Traffic Engineering, Link Weight system, Optimal routing.

I. INTRODUCTION

Link-state routing protocols such as OSPF (Open Shortest Path First, [19], [20]) and IS-IS (intermediate-system-to-intermediate-system, [5]) are commonly deployed as intra-domain routing protocols by Internet service providers. In OSPF (or IS-IS), each node finds all shortest paths for every destination based on the provided link weight system. With the equal-cost multi-path (ECMP) mechanism, all the traffic coming to a node (originating and transiting) having the same destination is aggregated and split almost equally amongst the outgoing links which are on the shortest paths to the destination. Note that OSPF splits traffic among the outgoing links (not paths) of the shortest paths for a source-destination pair.

Recently, there has been a growing interest in the problem of efficient engineering of an intra-domain IP network using OSPF [11], [22], [23]. Such initiatives aim to tailor the link weight system based on the expected traffic demand so as to ensure efficient use of the network. By adjusting the weight system appropriately, demand can be carried by operating the network at a proper utilization level with reduced end-to-end delay, hence obviating the short-term need to add additional capacity to the network. Moreover, it gives the operators a way of controlling the network by manipulating weights of links. Typically, such a weight adjustment mechanism is more

suited as an *offline* option. Based on projected demand volume, network bandwidth and network goal, an off-line mechanism computes an optimal weight system which is communicated to the routers. Such mode of operation, on one hand, gives the operators more control over the behavior of their network and, on the other hand, makes it possible to use fairly sophisticated algorithms for the computation of better link weights. In this backdrop, we present our work as further exploration of OSPF weight system as off-line calculation mechanism.

Historically, IP networks using OSPF have been engineered based on simple rules of thumb. An approach recommended by Cisco [6] was to set link weights as inverse of link capacity. The idea was that the links with slower rates should get less traffic and the ones with higher rates can handle much more traffic. An explanation of this weight can be made based on the M/M/1 queueing model; the average packet delay on a link based on the M/M/1 queueing model is given by $1/(c-f)$, where f is the flow on a link (“link load”) and c its corresponding (normalized) capacity. If a network is designed to have low utilization values, then $f \ll c$, and therefore, the delay $1/(c-f) \approx 1/c$. Thus, in such situations, weights based on inverse of the link capacities can be suitable. Although fairly simple and intuitive, such an approach does not account for the expected traffic or when the link load is non-negligible compared to the capacity of the link.

A. Related Work

There have been several recent studies of the OSPF link weight determination problem. An early work was by Fortz and Thorup [11] where the authors have used a piece-wise linear approximation as the link cost function to account for delay and discussed the performance and complexity issues of the problem. They have presented a local-search heuristic. For their study, they have considered two networks one with 90 nodes and 274 links, and the other with 100 nodes and 360 links. They showed that a weight system designed based on expected traffic pattern performs much better than the weight system based on inverse of link capacities. They also showed that, for arbitrary weights in the worst case (for the assumed link cost function), delays for an OSPF network could be 5000 times higher than that of a network using optimal routing; they concluded that, to avoid such a pitfall, weights should be determined based on expected traffic. The computing time for

* Corresponding Author: Tel: +1 816 235 2006, e-mail: dmedhi@umkc.edu

their heuristics to compute weights was about an hour for the 100-node network. Ericsson *et al* have presented a genetic algorithm based approach in [9]; they have used a similar objective function and same networks presented by Fortz and Thorup in [11]. The gap between the solution proposed by Ericsson *et al* and the lower bound was reported to be larger than the corresponding gap for solution provided by Fortz and Thorup; however the computing time for the genetic algorithm was between 10 to 40 minutes.

In a work independent to the above work, Pióro *et al* [22] have shown that finding a weight system is NP-Complete even for a single demand. They have presented heuristics that are based on weight adjustment, simulated allocation and simulated annealing approaches. For a network with 56 nodes and 208 links, the computing time for simulated allocation is reported to be approximately 54 minutes. Harmatos has presented an iterative heuristic in [14] where the author considers the same networks presented in [22]. The author has shown that the heuristic is successful in finding better solutions than the randomized optimization heuristics presented in [22] and comparatively in much less time than most of the heuristics, especially for the 56-node network.

Wang *et al* [27] have shown that any given set of optimal routes can be converted to shortest paths with respect to a set of positive link weights and link weights can be calculated by solving the dual of a linear programming formulation; the relation of dual solution to link weight has also been independently discussed in [1] and [22]. Ramakrishnan and Rodrigues [23] have proposed a combinatorial approximation algorithm. They have considered minimization of the average delay of packets in the network as the objective function. The authors have shown that their solution gave results within 0.5% of the optimal solution for the general routing problem for a 16-node sparse network.

Some recent works [3], [4] address the OSPF link weight problem by avoiding multiple shortest paths. [4] have presented a branch-and-cut algorithm by maximizing the minimum residual capacity as the performance criterion and reported results for networks with 9 nodes and 15 links. Bley [3] has presented an integrated formulation for designing a cost minimal IP network design problem (as opposed to the traffic engineering problem) that uses non-bifurcated shortest path routing. Results are reported for networks with up to 36 nodes, with computing time of 23 minutes.

It may be noted that most of these works consider only a particular objective function. Furthermore, while each work uses different computing platforms, the computing time is still noticeably significant as the number of nodes and links grow. In this paper, we consider three different objective functions and their impact on traffic engineering of an OSPF network. Furthermore, we present a Lagrangian-based dual approach for determining link weight while incorporating ECMP allocation and integer weights. As we report later in Section V-D, our approach is very fast in solving large network problems with good quality solutions; for instance, the computing time of our algorithm for a network with 100 nodes and 579 links is often less than a minute on a standard pentium-based computer while meeting a duality gap of 0.5%.

B. Scope of the paper

In this paper, we consider three objectives: i) minimization of total link flow, ii) minimization of maximum link utilization, and iii) a composition of the first two objectives. We also present a new solution approach based on the Lagrangian relaxation-based dual method to determine the optimal weight system; two key aspects about this approach are that 1) equal-cost multi-path (ECMP) is incorporated into the core of this approach, 2) the Lagrangian relaxation-based dual is used for link weight determination rather than the typical application of the such dual approach to obtain the primal solution through dual. Furthermore, we address how to obtain an integer-based weight system. We have tested our algorithm on large networks to show that it is a viable approach. Through computational studies, we demonstrate the effectiveness of the Lagrangian relaxation-based heuristic. Furthermore, we identify a set of performance measures that might be of interest to service providers, and report how these measures are impacted due to different objective functions.

The paper is organized as follows: In Section II, we define the OSPF weight system problem. In Section III, we present and discuss various possible objective functions. Next, in Section IV we present our approach based on Lagrangian relaxation and dual optimization. In Section V, we present numerical results for experimental networks and randomly generated networks.

II. PROBLEM DEFINITION

We consider an OSPF network where the traffic volume from an ingress router (source) to an egress router (destination) is given. Note that such traffic volume data can be determined from operational networks; see [10], [28]. In our case, we use the term *flow* to refer to the total traffic volume between an ingress node and an egress node, either sent on the shortest path, or split over multiple shortest paths using ECMP. Following [22], we first give a general framework for OSPF routing flow model using a link-path based multi-commodity flow formulation. This formulation inherently distinguishes flows between source-destination pairs whereas OSPF distinguishes flows based only on destinations. It has been shown in [25] that if we are distributing flows over shortest paths, it does not make any difference since the segments of shortest paths are also shortest paths. In other words, if source nodes A and B have a shortest path to node Z which goes through transit node T , path segment T - Z is also the shortest path from T to Z . Hence flow from A and B (multiple sources) to Z which meet at T will stay together after T all the way till Z . That is, we can be assured that any two flows having same destination meet only once and then stay together. Such a property allows us to use path based formulation and still ensure that the destination based forwarding of OSPF is captured.

We denote the link weight system by $\mathbf{w} = (w_\ell, \ell \in \mathcal{L})$ where w_ℓ is the weight of link ℓ ; note that our goal is to determine this link weight system (see Table I for the list of notations). The space of the weight system is defined as \mathcal{W} , where $\mathcal{W} = \{\mathbf{w} | w_\ell \in \{0, 1, 2, \dots, \bar{w}\}\}$; here, \bar{w} is the maximum allowable value. It may be noted that while OSPF

| | |
|--------------------|---|
| \mathcal{N} | : Set of nodes in the network |
| \mathcal{D} | : Set of origin-destination demand pairs in the network |
| \mathcal{L} | : Set of links in the network |
| \mathcal{P}_d | : Set of candidate paths for demand pairs $d \in \mathcal{D}$ |
| c_ℓ | : Capacity of link $\ell \in \mathcal{L}$ |
| h_d | : Traffic demand volume for demand pair $d \in \mathcal{D}$ |
| δ_{dj}^ℓ | : Entries of link-path incidence matrix; 1 if path j of demand pair d use link ℓ , 0 otherwise |
| w_ℓ | : Link weight of link $\ell \in \mathcal{L}$ |
| b_ℓ | : unit cost of flow on link $\ell \in \mathcal{L}$ |
| y_ℓ | : link flow (load) on link $\ell \in \mathcal{L}$ |

TABLE I
NOTATIONS USED IN FORMULATION

allows the minimum value of the weight to be 1 and maximum to be $2^{16} - 1$, IS-IS allows the minimum to be 0 and the maximum to be 63 (which has now been extended to $2^{24} - 1$). Note that it is important to have enough paths in the set \mathcal{P}_d such that for any given weight system \mathbf{w} , there are no shortest paths at optimality that is outside the set \mathcal{P}_d . Hence, fairly large number of candidate paths need to be pre-computed. However, we can circumvent this computational overhead by using path-generation based techniques which can add relevant paths in an iterative fashion; for details, see [8].

We define flow allocated to path $j \in \mathcal{P}_d$ of demand pair $d \in \mathcal{D}$ due to the chosen weight system \mathbf{w} by $x_{dj}(\mathbf{w})$. Note that flows are allocated to paths \mathcal{P}_d while honoring the ECMP principle. That is, based on the given weight system \mathbf{w} , we first find the set of all the shortest paths (subset of \mathcal{P}_d); these shortest paths are allocated flows based on the ECMP rule. Flows on different paths dictated by weight system \mathbf{w} need to satisfy the following system:

$$\sum_{j \in \mathcal{P}_d} x_{dj}(\mathbf{w}) = h_d, \quad d \in \mathcal{D}, \quad (1a)$$

$$\sum_{d \in \mathcal{D}} \sum_{j \in \mathcal{P}_d} \delta_{dj}^\ell x_{dj}(\mathbf{w}) \leq c_\ell, \quad \ell \in \mathcal{L} \quad (1b)$$

$$\mathbf{w} \in \mathcal{W}. \quad (1c)$$

The first constraint refers to the fact that paths are to carry all the demand volume for a demand pair $d \in \mathcal{D}$. The second constraint is to ensure that the flow on a link does not violate the capacity constraints. This system implicitly assumes that the network has the capacity to carry all the demand for a link weight system; otherwise, the above system will be infeasible. The unknown in the above system is the weight system \mathbf{w} ; given this weight system, values for $\mathbf{x}(\mathbf{w})$ can be obtained. We refer to the above system as **(PA)**. It is important to note that the system shown in (1) is *not* a set of linear equations/inequalities due to the functional dependency of \mathbf{x} on weight system \mathbf{w} . For convenience, we will also denote the expression on the left side of (1b), i.e., the link flow on link ℓ , by y_ℓ .

III. OBJECTIVE FUNCTIONS

In this section, we present three objective functions which can address the goals of interest to backbone service providers.

A. Objective Function–A

Consider the objective function that minimizes the used capacity in the network. This objective was used in [22]. Incorporating this objective, we can write the following formulation **(PA₁)**:

$$\bar{F}_1 = \min_{\{\mathbf{w}\}} f_1(\mathbf{w}) = \sum_{\ell \in \mathcal{L}} b_\ell \sum_{d \in \mathcal{D}} \sum_{j \in \mathcal{P}_d} \delta_{dj}^\ell x_{dj}(\mathbf{w}) \quad (2)$$

subject to

$$\sum_{j \in \mathcal{P}_d} x_{dj}(\mathbf{w}) = h_d, \quad d \in \mathcal{D} \quad (3a)$$

$$\sum_{d \in \mathcal{D}} \sum_{j \in \mathcal{P}_d} \delta_{dj}^\ell x_{dj}(\mathbf{w}) \leq c_\ell, \quad \ell \in \mathcal{L} \quad (3b)$$

$$\mathbf{w} \in \mathcal{W}. \quad (3c)$$

This formulation attempts to allocate most demands to (closest to) minimum hop paths, based on unit cost b_ℓ , subject to capacity c_ℓ . The main difficulty with this objective is that at optimality, a link can still have 100% utilization which is not desirable in an OSPF network due to queueing delay.

B. Objective Function–B

In order to avoid the problem faced by the first objective function, utilization on any link in the network is desirable to be as low as possible. This can be achieved by minimizing the load of the maximum utilized link. This objective has been known in the literature for sometime (for example, see [2]). We define $t(\mathbf{w})$ as the maximum link utilization when flows are allocated based on ECMP-based shortest paths, dictated by weight system \mathbf{w} . The goal is to minimize $t(\mathbf{w})$. We refer to this formulation as **(PA₂)**.

$$\bar{F}_2 = \min_{\{\mathbf{w}\}} f_2(\mathbf{w}) = t(\mathbf{w}) \quad (4)$$

subject to

$$\sum_{j \in \mathcal{P}_d} x_{dj}(\mathbf{w}) = h_d, \quad d \in \mathcal{D} \quad (5a)$$

$$\sum_{d \in \mathcal{D}} \sum_{j \in \mathcal{P}_d} \delta_{dj}^\ell x_{dj}(\mathbf{w}) \leq t(\mathbf{w}) c_\ell, \quad \ell \in \mathcal{L} \quad (5b)$$

$$\mathbf{w} \in \mathcal{W}. \quad (5c)$$

Note that this formulation can avoid the short-comings of the first objective function but can introduce its own. For example, a demand could choose an extremely long path and increase the overall capacity usage of the allocation. Although, low value of link load/utilization leads to smaller delays on each hop, the end-to-end delay could still be noticeable if a path consists of many links.

C. Combined Objective Function

By considering the objectives described in the previous subsections, we construct a composite objective function which combines the benefit of minimization of total flow and the goal to reduce maximum link utilization. We introduce scaling factor α (> 0) to the first objective function (2) and β (> 0) to the second objective function (4) to construct the new objective. We refer to the formulation with the new objective as (\mathbf{PA}_3) .

$$\begin{aligned} \bar{F}_3(\alpha, \beta) &= \min_{\{\mathbf{w}\}} f_3(\mathbf{w}, \alpha, \beta) \\ &= \min_{\{\mathbf{w}\}} \alpha \sum_{\ell \in \mathcal{L}} b_\ell \sum_{d \in \mathcal{D}} \sum_{j \in \mathcal{P}_d} \delta_{dj}^\ell x_{dj}(\mathbf{w}) + \beta t(\mathbf{w}) \end{aligned}$$

subject to

$$\sum_{j \in \mathcal{P}_d} x_{dj}(\mathbf{w}) = h_d, \quad d \in \mathcal{D} \quad (6a)$$

$$\sum_{d \in \mathcal{D}} \sum_{j \in \mathcal{P}_d} \delta_{dj}^\ell x_{dj}(\mathbf{w}) \leq t(\mathbf{w}) c_\ell, \quad \ell \in \mathcal{L} \quad (6b)$$

$$\mathbf{w} \in \mathcal{W}. \quad (6c)$$

Note that by tuning values of α and β , we can control the importance given to one (hop-count) or the other (link utilization). However, there is a difference of scale in the values of α and β . On one hand, α is multiplied to the total flow in the network which is many times the total network-wide demand volume; on the other hand, β is multiplied to the load of maximum utilized link which is between zero and one. Hence, choosing α and β close to each other, or larger values of α will make the composite objective function behave like the first objective function. Choosing very small values of α and very high value of β will make the composite objective function behave like the second objective function.

To summarize, we have presented a composite objective function by combining two objectives that can be useful to service providers for engineering OSPF networks. There is, however, a difficulty with all the three formulations—they are *not* mathematical programming problems due to functional dependency of $x_{dj}(\mathbf{w})$ on link weight system \mathbf{w} that can not be stated explicitly. On the other hand, if we ignore the dependency \mathbf{w} , then the resulting problems with just flow variables, x_{dj} , are linear programming problems; then, optimal objective function values to these linear programming problems serve as lower bounds to the optimal objective for the original problems since the dependency on \mathbf{w} can be thought of as another set of constraints. It may be noted that problems (\mathbf{PA}_1) , (\mathbf{PA}_2) and (\mathbf{PA}_3) can be still considered as multi-commodity flow problems; in our case, they are represented using the link-path formulation since we assume that the candidate paths are pre-processed. The weight determination problem, even for a single demand, have been shown to be NP-complete (see [21], [22]). In the next section, we present details of a decomposition algorithm for solving the above formulations.

IV. DECOMPOSITION ALGORITHM (DA)

Our heuristic approach is based on Lagrangian relaxation and dual maximization using subgradient optimization ([12], [13]). It is important to note that the purpose here is to find link weight system \mathbf{w} ; this is different than the typical use of such Lagrangian relaxation-based dual approach where the goal is to determine solution for explicitly defined primal variables. Furthermore, we need to make an adjustment that requires the algorithm to allocate flows to the shortest paths mimicking ECMP, as the iteration progresses. We describe our approach for (\mathbf{PA}_3) while the approach is applicable to the first two formulations as well. Briefly, here we optimize for variable \mathbf{w} ; using the expression (14) for link weights as a guiding principle, we determine the dual multiplier $\boldsymbol{\pi}^k$ at each dual iteration k ; in turn, we compute \mathbf{w}^k to serve as the link weight in iteration k —this is then used to determine the flow x_{dj}^k at iteration k using the ECMP rule. This process is continued as the dual multiplier is updated using the Lagrangian relaxation framework.

Consider Lagrangian function L obtained when capacity constraint (6b) is relaxed:

$$\begin{aligned} L(\mathbf{w}; \boldsymbol{\pi}) &= \alpha \sum_{\ell \in \mathcal{L}} b_\ell \sum_{d \in \mathcal{D}} \sum_{j \in \mathcal{P}_d} \delta_{dj}^\ell x_{dj}(\mathbf{w}) + \beta t(\mathbf{w}) \\ &+ \sum_{\ell \in \mathcal{L}} \pi_\ell \left(\sum_{d \in \mathcal{D}} \sum_{j \in \mathcal{P}_d} \delta_{dj}^\ell x_{dj}(\mathbf{w}) - t(\mathbf{w}) c_\ell \right). \end{aligned}$$

Dual problem (D), corresponding to problem (\mathbf{PA}_3) , is

$$s_D = \max_{\boldsymbol{\pi} \geq 0} g(\boldsymbol{\pi}), \quad (7)$$

where

$$g(\boldsymbol{\pi}) = \min_{\mathbf{w} \in \bar{\mathcal{W}}} L(\mathbf{w}; \boldsymbol{\pi}). \quad (8)$$

Here, set $\bar{\mathcal{W}}$ is given by

$$\bar{\mathcal{W}} = \{\mathbf{w} \mid \sum_{j \in \mathcal{P}_d} x_{dj}(\mathbf{w}) = h_d, d \in \mathcal{D}; x_{dj}(\mathbf{w}) \geq 0; \mathbf{w} \in \mathcal{W}\}. \quad (9)$$

The basic algorithmic structure to solve (\mathbf{PA}_3) is given in Algorithm 1.

Algorithm 1 Decomposition Algorithm

procedure $DA(\mathbf{h}, \mathcal{G})$

1. Initialize dual variable $\boldsymbol{\pi}$
 2. Obtain $g(\boldsymbol{\pi})$ based on the present value of the dual variable $\boldsymbol{\pi}$
 3. Update dual variables \mathbf{w} with a goal to solve s_D using subgradient optimization
 4. Check for the convergence; if not converged, go to Step 2
-

A. Determining $g(\boldsymbol{\pi})$

Determining $g(\boldsymbol{\pi})$ involves minimizing Lagrangian function L when dual multiplier $\boldsymbol{\pi}$ is given. Rearranging the Lagrangian

function, we arrive at

$$g(\boldsymbol{\pi}) = \min_{\boldsymbol{w} \in \mathbb{V}} \left\{ \begin{array}{l} \sum_{d \in \mathcal{D}} \sum_{j \in \mathcal{P}_d} [\sum_{\ell \in \mathcal{L}} (\pi_\ell + \alpha b_\ell) \delta_{dj}^\ell] x_{dj}(\boldsymbol{w}) \\ + (\beta - \sum_{\ell \in \mathcal{L}} \pi_\ell c_\ell) t(\boldsymbol{w}) \end{array} \right\}. \quad (10)$$

Note that the evaluation of $g(\boldsymbol{\pi})$ depends on determining $x_{dj}(\boldsymbol{w})$ and $t(\boldsymbol{w})$ first. However, they can not be decoupled (as would be the case in a standard Lagrangian relaxed based dual approach) due to the direct relation between the values of $x_{dj}(\boldsymbol{w})$ and $t(\boldsymbol{w})$ through the link weight variable \boldsymbol{w} since

$$t(\boldsymbol{w}) = \max_{\ell \in \mathcal{L}} \left\{ \sum_{d \in \mathcal{D}} \sum_{j \in \mathcal{P}_d} \delta_{dj}^\ell x_{dj}(\boldsymbol{w}) \right\}. \quad (11)$$

Consider set $\boldsymbol{\Pi}$ defined as

$$\boldsymbol{\Pi} = \left\{ \boldsymbol{\pi} \mid \sum_{\ell \in \mathcal{L}} \pi_\ell c_\ell = \beta \right\}. \quad (12)$$

Observe that when $\boldsymbol{\pi} \in \boldsymbol{\Pi}$, the co-efficient term of $t(\boldsymbol{w})$ in (10) vanishes and thus finding $g(\boldsymbol{\pi})$ becomes independent of the value of $t(\boldsymbol{w})$; this then simplifies the minimization problem. We can incorporate this restriction since at the optimal solution to the original problem, $\boldsymbol{\pi}$ would naturally satisfy the above constraint; this can be independently verified through standard linear programming dual of the original problem along with the optimality conditions. Thus, if we consider (7) only for the values of $\boldsymbol{\pi}$ in set $\boldsymbol{\Pi}$, that is for the revised problem

$$s_D = \max_{\boldsymbol{\pi} \in \boldsymbol{\Pi}} g(\boldsymbol{\pi}), \quad (13)$$

then flows $\boldsymbol{x}(\boldsymbol{w})$ in minimization of L in (10) can be allocated to the paths which are shortest with respect to link weight

$$w_\ell = \pi_\ell + \alpha b_\ell, \quad \ell \in \mathcal{L}. \quad (14)$$

That is, any such allocation gives minimal value of $L(\boldsymbol{w}; \boldsymbol{\pi})$. In order for the procedure to work, we need to ensure that, at every dual iteration $\boldsymbol{\pi}$ satisfies constraint (12). Now, with the disappearance of the second term in (10), we incorporate the ECMP feature of OSPF (in every update of the dual iteration) through appropriate setting of flow allocation $x_{dj}(\boldsymbol{w})$ based on weights as given in (14).

In Algorithm 2, we present our implementation of the ECMP algorithm for the link-path multi-commodity flow formulation; this algorithm is invoked at every dual iteration. This approach is derived from [9] which uses a node-link formulation. At each dual iteration k , procedure *ECMP Allocation* is executed for every demand and is used for determining values of \boldsymbol{x} that follows ECMP allocation. We refer to \mathcal{P}^{nr} as the set of all shortest paths from source node n and destination r with w_ℓ as link weights; this set is a subset of the pre-processed candidate paths, and is thus readily available based on the link weights. If a demand has only a unique shortest path, then we do not need to execute this procedure, rather we can directly allocate the demand to the unique shortest path.

Algorithm 2 ECMP Flow Allocation Algorithm (given demand d with source node s and destination node r and the set of ECMP paths \mathcal{P}^{sr})

```

procedure EA ( $s, r, h_d, \mathcal{P}^{sr}$ )
   $S^{sr} = \{\ell \in \mathcal{L} \mid \ell \text{ is first link of path } \mathcal{P} \in \mathcal{P}^{sr}\}$ 
   $h' = \frac{h_d}{|S^{sr}|}$ 
  for all  $\ell \in S^{sr}$  do
     $n \leftarrow \text{otherend}(\ell, s)$ 
     $flow_\ell \leftarrow flow_\ell + h'$ 
    if  $n \neq r$  then
       $\mathcal{P}^{nr} = \{\mathcal{P} \setminus \{\ell\} \mid \ell \text{ is first link of path } \mathcal{P} \in \mathcal{P}^{sr}\}$ 
      EA ( $n, r, h', \mathcal{P}^{nr}$ )
    end if
  return
end for

```

B. Solving s_D

Problem, s_D as given in (13) is an optimization problem over the set $\{\boldsymbol{\pi} \mid \boldsymbol{\pi} \in \boldsymbol{\Pi}\}$. Function $g(\boldsymbol{\pi})$ to be maximized is a non-smooth function; thus, we use subgradient optimization approach [15] to solve dual problem s_D . Suppose at dual iteration k , we have the dual multiplier, $\boldsymbol{\pi}^k$. Then, we can determine \boldsymbol{w}^k using (14) which can be used for computing $g(\boldsymbol{\pi}^k)$. A dual subgradient, $\boldsymbol{\Delta} = (\Delta_\ell)$ for $g(\cdot)$ can be computed as

$$\Delta_\ell = \sum_{d \in \mathcal{D}} \sum_{j \in \mathcal{P}_d} \delta_{dj}^\ell x_{dj}(\boldsymbol{w}^k) - t(\boldsymbol{w}^k) c_\ell, \quad \ell \in \mathcal{L}. \quad (15)$$

Dual multiplier, $\boldsymbol{\pi}$, is updated as

$$\pi_\ell^{k+1} = \max \{0, \pi_\ell^k + \gamma_k \Delta_\ell\} \quad (16)$$

$$\pi_\ell^{k+1} \leftarrow \pi_\ell^{k+1} \frac{\beta}{\sum_{m \in \mathcal{L}} \pi_m^{k+1} c_m}. \quad (17)$$

The second step is done to ensure that $\boldsymbol{\pi} \in \boldsymbol{\Pi}$. In the update, γ_k is computed as

$$\gamma_k = \rho \frac{g^\# - g(\boldsymbol{\pi}^k)}{\|\boldsymbol{\Delta}\|^2}, \quad (18)$$

where $g^\#$ is the lowest value of the feasible primal objective function through iteration k . In our implementation, we set the initial dual iterate as

$$\pi_\ell^0 = \frac{\beta}{\sum_{m \in \mathcal{L}} c_m}, \quad \ell \in \mathcal{L} \quad (19)$$

and set $\rho = 2$. We reduce the value of ρ in half only if the solution value does not change for 40 consecutive iterations. We limit the maximum number of iterations to 1000. We estimate the proximity of the current point with the optimal solution based on the value of duality gap computed as $(\frac{g^\# - g(\boldsymbol{\pi})}{g^\#})$. When the value of duality gap is less than ϵ , we stop the algorithm; otherwise, the procedure continues until the maximum number of iterations is reached. In our implementation, we have used $\epsilon = 0.005$. An important point is that we have assumed that all π_ℓ 's are not zero at any given iteration—this ensures that the division in (17) is valid; this assumption is realistic since $\boldsymbol{\pi}$ needs to satisfy (12). Finally,

note that π_ℓ^{k+1} as computed in (17) is used for updating the link weight (14) in the next iteration which is, in turn, used in (10) to determine the ECMP flows.

Observe that the decomposition algorithm does not ensure that the weights obtained at the end (referred to as w_ℓ^*) are integer valued. Since the OSPF protocol allows only integer weights, we have used a simple heuristic rule to obtain integer weights based on the solution of the decomposition algorithm as follows:

$$w_\ell^* = \lceil w_\ell^* \times G \rceil, \quad \ell \in \mathcal{L}. \quad (20)$$

Here, G is a large positive integer, and $\lceil \cdot \rceil$ is the nearest integer value obtained after round-off. After experimenting with several values of G , we found that $G = 1000$ seems to work particularly well in the sense that flow allocation was not affected between the non-integer and integer weights derived when this value of G is used. Thus, all our results discussed in the next section will be based on using integer weights obtained from this scaling factor.

Remark The above decomposition algorithm, presented for (\mathbf{PA}_3) , is also applicable to (\mathbf{PA}_1) and (\mathbf{PA}_2) . The main difference is how the weight is selected in each dual iteration. By using Lagrangian duality, as we have done above, it is easy to show that instead of using (14), for (\mathbf{PA}_1) , it becomes

$$w_\ell = b_\ell + \pi_\ell, \quad \ell \in \mathcal{L}, \quad (21)$$

and for (\mathbf{PA}_2) , it becomes

$$w_\ell = \pi_\ell, \quad \ell \in \mathcal{L}. \quad (22)$$

In addition, there is another important difference: in the case of (\mathbf{PA}_2) , the dual multiplier space is the same as (\mathbf{PA}_3) while for (\mathbf{PA}_1) , the dual multipliers need to be just the non-negative space.

V. RESULTS

We have implemented the decomposition algorithm using C^{++} . A network provider is usually interested in a set of measures (rather than just the lowest objective function value) to see whether the network is engineered properly. The goal of this section is to see how various performance measures are impacted due to different objectives chosen, and whether the algorithm performs well.

A. Performance Measures

In our study, we consider the following measures:

- i) Maximum Link Utilization (ML) captures the utilization of the link which is maximum loaded in the entire network.
- ii) Fraction of Used Capacity (FU) captures the total used capacity in the final solution as a fraction of total capacity in the network.
- iii) Number of Overloaded Links (NO) refers to the number of links which requires extra capacity to make the solution feasible. This metric is important only when the obtained solution is infeasible.
- iv) Fraction of required Extra Capacity (FE) captures the additional capacity required to make the solution feasible

as a fraction of total capacity of the network. This is only relevant when the solution is infeasible.

- v) Fortz-Thorup Function (FT) [11] captures the the total congestion cost incurred where the cost per link ℓ is given by

$$\phi_\ell = \begin{cases} y_\ell & \text{for } 0 \leq \frac{y_\ell}{c_\ell} < \frac{1}{3} \\ 3y_\ell - \frac{2}{3}c_\ell & \text{for } \frac{1}{3} \leq \frac{y_\ell}{c_\ell} < \frac{2}{3} \\ 10y_\ell - \frac{16}{3}c_\ell & \text{for } \frac{2}{3} \leq \frac{y_\ell}{c_\ell} < \frac{9}{10} \\ 70y_\ell - \frac{178}{3}c_\ell & \text{for } \frac{9}{10} \leq \frac{y_\ell}{c_\ell} < 1 \\ 500y_\ell - \frac{1468}{3}c_\ell & \text{for } 1 \leq \frac{y_\ell}{c_\ell} < \frac{11}{10} \\ 5000y_\ell - \frac{16318}{3}c_\ell & \text{for } \frac{11}{10} \leq \frac{y_\ell}{c_\ell} < \infty. \end{cases} \quad (23)$$

The scaled (normalized) cost ($\sum_{\ell \in \mathcal{L}} \phi_\ell / \varphi$) is the ratio of total cost of current allocation ($\sum_{\ell \in \mathcal{L}} \phi_\ell$) for the given capacitated network as compared to the cost in case the network was uncapacitated (φ). Observe that for an uncapacitated network with convex link cost function, cost is minimal when flows are allocated to hop count based shortest paths.

The above five measures are often of interest to service providers. We have also considered two additional measures: the gap between the objective function of the original problem and its linear programming relaxation, and the duality gap between the dual solution and the objective value determined based on the weights determined by our Lagrangian relaxation-based dual approach; these ‘‘gap’’ measures are useful indicators of the quality of the solution. We have found that in almost all test cases we considered, these two gap measures were less than a fraction of 1%. There are a few instances where the gaps were about 6% when the maximum iteration count for the dual iteration is reached. Thus, in general, the convergence property is found to be excellent in determining link weights that works with the ECMP principle. While there have been several papers that present various heuristics for determining link weights, heuristics do not contain enough details to be able to accurately implement them; thus, no numerical comparison is done with existing heuristics. In the rest of our discussion, we will primarily concentrate on the five measures that are of interest to service providers.

B. Network Study Cases

We have used three sets of test networks in our study. The first set is based on well-known network topologies (‘‘experimental networks’’) found in the literature; the second set is based on generating random network topologies using BRITE [17]; in the third case, large networks were used to see the effectiveness of our decomposition algorithm in solving a large-scale traffic engineering problem from a computational point of view.

Demand volumes between all pairs of nodes are assumed to be 100 Mbps each for the uniform case, while for the random demand case 100 Mbps is used as the *average* demand volume for all the demand pairs. We then use the Fortz-Thorup linear programming formulation [11] (i.e., without any link weight computation) in an iterative fashion to derive the baseline link capacities such that all the links have less than 67%

utilization. Networks with 50% additional capacity are derived by multiplying the baseline capacity for each link by 1.5. In the next subsection, we will primarily concentrate detailed discussion for the first set of experimental networks, with both uniform and random demand volumes. Then in the following subsection, we will comment on random networks and results for large-scale networks.

C. Study on Experimental Networks

The network topologies used in this study are taken from already published literature [16], [26]. We consider four networks (Figures 1 to 4): 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 2.6. Observe that EN-IV is the most well connected network while EN-I is the least. Recall that our formulation is based on using a set of pre-processed candidate path list: for EN-I, EN-II, EN-III and EN-IV, we have used 15, 6, 15, and 15 candidate paths for each demand pair, respectively. It is certainly possible that the true shortest path at optimality at least for some demands are not included in the initially processed list; however, this is unlikely when a large number of paths are considered to begin with. Based on the convergence results observed (from “gap” measures), the set of pre-processed paths is believed to be sufficient. Certainly, a path generation technique can be used to incorporate additional paths (see [21]) if the solution quality were not acceptable; however, this is outside the scope of the current paper since our goal is to see the effectiveness of the composite objective function and the decomposition algorithm.

From our initial study, we have found that using the Lagrangian relaxation-based dual approach on the second objective function (i.e., Problem (\mathbf{PA}_2)) did not give feasible results for almost all of the experimental networks when the baseline capacity was used. This behavior is counter-intuitive since this objective is to minimize the maximum link utilization. On closer scrutiny, we found that the weight selection based on the dual solution, i.e., (22), along with the requirement on π_ℓ due to (12) and the initial starting point for π_ℓ given in (19) leads almost uniformly to zero for all π_ℓ 's at the end; consequently, all paths are shortest paths since each link weight is zero; this often resulted in allocations that violated capacity constraints. Thus, in the rest of the discussion, we will present results for (\mathbf{PA}_1) and (\mathbf{PA}_3) .

We present values of various performance measures for experimental networks in Tables II- V when uniform demand volume is used. For each problem, (\mathbf{PA}_1) and (\mathbf{PA}_3) , we also include results for their linear programming (LP) relaxation, denoted by (\mathbf{PA}_1^L) and (\mathbf{PA}_3^L) , respectively. The LP relaxations were solved using CPLEX. Following the layout of the results presented in the tables, we discuss two types of comparison: horizontal comparison (i.e., compare between LP relaxation and the weight determination), and vertical comparison (i.e.,

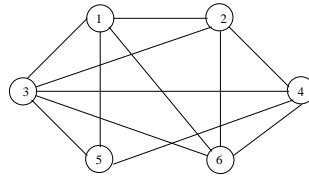


Fig. 1. Experimental Network I

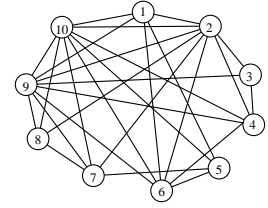


Fig. 2. Experimental Network II

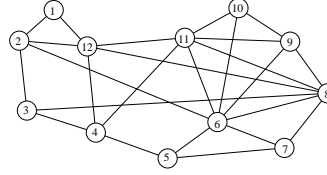


Fig. 3. Experimental Network III

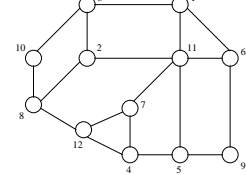


Fig. 4. Experimental Network IV

between two different objectives, and also increase in capacity).

We start with horizontal comparison. We observe that for LP relaxation with objective-A, (\mathbf{PA}_1^L) , the maximum link utilization (ML) is always 100% when the baseline capacity is used (also true in some cases when 150% of the baseline capacity was used); this is not surprising given the goal of this objective and since an LP is solved. On the other hand, for its counterpart (\mathbf{PA}_1) when the link weight determination is done through the Lagrangian relaxation-based dual approach, the measure ML does not always reach 100%—this can be attributed to the fact that our weight selection is indeed influencing the flow allocation on links in a positive manner. When we consider 150% of the baseline capacity, solving (\mathbf{PA}_1^L) still results in ML being 100% in two cases, while none are violated when (\mathbf{PA}_1) is solved. This suggests that even with objective-A, capacity violation can be avoided with our link weight determination algorithm if the network capacity is reasonably abundant.

Now consider the comparison between (\mathbf{PA}_3^L) and (\mathbf{PA}_3) , i.e., the case between LP relaxation and our algorithm when the composite function is used. We note that ML increases, but FT-value is not significantly higher. Thus, this suggests that the Lagrangian relaxation-based dual approach can arrive at a good solution.

When we compare vertically, i.e., (\mathbf{PA}_1) and (\mathbf{PA}_3) , we see that ML decreases with the composite objective, and the FT-value decreases as well. This is more pronounced when 150% of the capacity is used. We can infer that the composite objective with the Lagrangian relaxation-based dual approach is quite powerful in generating link weights without compromising different measures.

To avoid any possible artifacts due to uniform demands, we have also used the same networks with random demand volumes. The results are reported in Tables VI - IX. We observe a pattern similar to the case when uniform demand volume is used ruling out the possibility of any artifact.

It may be noted that in both sets of studies above, we reported results only for a certain combination of (α, β) for (\mathbf{PA}_3^L) and (\mathbf{PA}_3) . In actually, we have tested a variety of

TABLE II
RESULTS FOR (\mathbf{PA}_1) FOR EXPERIMENTAL NETWORKS (WITH UNIFORM DEMANDS)

| ENs | (\mathbf{PA}_1^L) | | | (\mathbf{PA}_1) | | | |
|--------|---------------------|------|------|-------------------|------|------|-------------|
| | ML | FU | FT | ML | FU | FT | F/I(NO, FE) |
| EN I | 1.00 | 0.66 | 3.84 | 0.87 | 0.66 | 2.41 | F(-) |
| EN II | 1.00 | 0.67 | 3.62 | 0.89 | 0.67 | 2.52 | F(-) |
| EN III | 1.00 | 0.65 | 4.51 | 1.04 | - | - | I(1,0.001) |
| EN IV | 1.00 | 0.66 | 4.51 | 0.99 | 0.66 | 2.87 | F(-) |

TABLE III
RESULTS FOR (\mathbf{PA}_1) FOR EXPERIMENTAL NETWORKS FOR 50% ADDITIONAL CAPACITY (WITH UNIFORM DEMANDS)

| ENs | (\mathbf{PA}_1^L) | | | (\mathbf{PA}_1) | | | |
|--------|---------------------|------|------|-------------------|------|------|-------------|
| | ML | FU | FT | ML | FU | FT | F/I(NO, FE) |
| EN I | 0.80 | 0.44 | 1.62 | 0.57 | 0.44 | 1.51 | F(-) |
| EN II | 0.87 | 0.45 | 1.79 | 0.59 | 0.45 | 1.53 | F(-) |
| EN III | 1.00 | 0.43 | 2.36 | 0.75 | 0.43 | 1.53 | F(-) |
| EN IV | 1.00 | 0.44 | 3.12 | 0.68 | 0.44 | 1.52 | F(-) |

TABLE IV
RESULTS FOR (\mathbf{PA}_3) FOR EXPERIMENTAL NETWORKS (WITH UNIFORM DEMANDS)

| ENs | (α, β) | (\mathbf{PA}_3^L) | | | (\mathbf{PA}_3) | | | |
|--------|-------------------|---------------------|------|------|-------------------|------|------|-------------|
| | | ML | FU | FT | ML | FU | FT | F/I(NO, FE) |
| EN I | (0.5, 4) | 0.79 | 0.66 | 2.49 | 0.82 | 0.66 | 2.41 | F(-) |
| EN II | (0.9, 32) | 0.67 | 0.67 | 2.00 | 0.67 | 0.67 | 2.00 | F(-) |
| EN III | (0.5, 1024) | 0.67 | 0.67 | 2.09 | 0.80 | 0.66 | 2.26 | F(-) |
| EN IV | (0.9, 64) | 0.74 | 0.66 | 2.37 | 0.84 | 0.66 | 2.44 | F(-) |

TABLE V
RESULTS FOR (\mathbf{PA}_3) FOR EXPERIMENTAL NETWORKS WITH 50% ADDITIONAL CAPACITY (WITH UNIFORM DEMANDS)

| ENs | (α, β) | (\mathbf{PA}_3^L) | | | (\mathbf{PA}_3) | | | |
|--------|-------------------|---------------------|------|------|-------------------|------|------|-------------|
| | | ML | FU | FT | ML | FU | FT | F/I(NO, FE) |
| EN I | (0.5, 32) | 0.53 | 0.44 | 1.49 | 0.53 | 0.44 | 1.49 | F(-) |
| EN II | (0.5, 32) | 0.45 | 0.45 | 1.49 | 0.45 | 0.45 | 1.49 | F(-) |
| EN III | (0.5, 1448) | 0.47 | 0.44 | 1.52 | 0.53 | 0.45 | 1.56 | F(-) |
| EN IV | (0.5, 64) | 0.50 | 0.44 | 1.49 | 0.64 | 0.44 | 1.50 | F(-) |

combination of α and β , and reported a specific combination to show that we observe a better behavior. We did not find any specific pattern regarding deciding on α and β ; that is, it is not possible to easily determine a rule of thumb on picking α and β (except for the obvious extreme cases that were discussed earlier in Section III-C). Since computational time required for running our algorithm is minimal (see next subsection), a user can run a series of values and choose a combination that works the best depending on the importance of the goals as indicated by the measures.

D. Study on Random Networks and Large-Scale Networks

We have also conducted studies on two additional cases. In the first case, we have generated random ten-node network topology using BRITE with different nodal degrees (i.e., different link connectivity) and compared (\mathbf{PA}_1^L) , (\mathbf{PA}_1) , (\mathbf{PA}_3^L) , and (\mathbf{PA}_3) . We found the behavior to be similar to experimental networks.

In the second case, we generated large network examples to gain insight on computational time. Our computation was performed on a pentium-4 computer running the linux operating system. We have considered several 100-node networks with the number of links ranging from 197 links to 579 links with nodal degree varying from 2 to 6. While we indicated earlier that a stopping criterion is when the duality gap threshold of 0.5% is reached, we still let the program run for a minimum of 250 dual iterations; the maximum number of iterations is kept at 1000. We report results using our algorithm described in

Section IV for both (\mathbf{PA}_1) and (\mathbf{PA}_3) in Tables X and XI which correspond to objective function-A and composite objective function, respectively.

From results for the 100-node network examples, we can see that our algorithm is very fast, often taking less than a minute of run time. We found that in general (\mathbf{PA}_1) took more computing time than (\mathbf{PA}_3) . On closer scrutiny, we found that for (\mathbf{PA}_1) , the ECMP flow allocation procedure (Algorithm 2) was invoked more often compared to the corresponding case for (\mathbf{PA}_3) . In addition to taking less computing time, various performance measures (such as ML, FT) are also found to be more effective with (\mathbf{PA}_3) than (\mathbf{PA}_1) .

VI. SUMMARY

In this work, we explore the problem of traffic engineering an OSPF network by adjusting the link weights for different objectives. In addition to flow minimization and minimize maximum utilization as objectives, we also consider a composite function that combines both these objectives. We present a new approach to find such a weight system. Our approach is based on Lagrangian relaxation and dual optimization. The technique is new in the sense of applying relaxation to the dual of a problem. We found the technique to be very efficient for weight system computation and very fast computationally. In addition, the composite function as the objective was able to generate solutions that perform well in terms of various performance measures that are of interest to network providers.

TABLE VI
RESULTS FOR (\mathbf{PA}_1) FOR EXPERIMENTAL NETWORKS (WITH RANDOM DEMANDS)

| ENs | (\mathbf{PA}_1^L) | | | (\mathbf{PA}_1) | | | |
|--------|---------------------|------|------|-------------------|------|------|-------------|
| | ML | FU | FT | ML | FU | FT | F/I(NO, FE) |
| EN I | 1.00 | 0.68 | 3.42 | 0.91 | 0.68 | 2.59 | F(-) |
| EN II | 1.00 | 0.61 | 4.43 | 0.94 | 0.61 | 2.60 | F(-) |
| EN III | 1.00 | 0.63 | 4.38 | 0.96 | 0.63 | 2.83 | F(-) |
| EN IV | 1.00 | 0.68 | 3.42 | 0.98 | 0.65 | 2.93 | F(-) |

TABLE VII
RESULTS FOR (\mathbf{PA}_1) FOR EXPERIMENTAL NETWORKS FOR 50% ADDITIONAL CAPACITY (WITH RANDOM DEMANDS)

| ENs | (\mathbf{PA}_1^L) | | | (\mathbf{PA}_1) | | | |
|--------|---------------------|------|------|-------------------|------|------|-------------|
| | ML | FU | FT | ML | FU | FT | F/I(NO, FE) |
| EN I | 0.91 | 0.45 | 1.71 | 0.61 | 0.45 | 1.55 | F(-) |
| EN II | 1.00 | 0.41 | 3.93 | 0.92 | 0.41 | 2.25 | F(-) |
| EN III | 1.00 | 0.42 | 2.88 | 0.77 | 0.42 | 1.50 | F(-) |
| EN IV | 1.00 | 0.42 | 3.27 | 0.74 | 0.42 | 1.56 | F(-) |

TABLE VIII
RESULTS FOR (\mathbf{PA}_3) FOR EXPERIMENTAL NETWORKS (WITH RANDOM DEMANDS)

| ENs | (α, β) | (\mathbf{PA}_3^L) | | | (\mathbf{PA}_3) | | | |
|--------|-------------------|---------------------|------|------|-------------------|------|------|-------------|
| | | ML | FU | FT | ML | FU | FT | F/I(NO, FE) |
| EN I | (0.9, 2048) | 0.73 | 0.68 | 2.45 | 0.84 | 0.68 | 2.51 | F(-) |
| EN II | (0.9, 4) | 0.87 | 0.61 | 2.50 | 0.87 | 0.61 | 2.33 | F(-) |
| EN III | (0.5, 181) | 0.95 | 0.63 | 3.56 | 0.96 | 0.63 | 2.67 | F(-) |
| EN IV | (0.1, 90) | 0.68 | 0.65 | 2.13 | 0.98 | 0.65 | 3.17 | F(-) |

TABLE IX
RESULTS FOR (\mathbf{PA}_3) FOR EXPERIMENTAL NETWORKS WITH 50% ADDITIONAL CAPACITY (WITH RANDOM DEMANDS)

| ENs | (α, β) | (\mathbf{PA}_3^L) | | | (\mathbf{PA}_3) | | | |
|--------|-------------------|---------------------|------|------|-------------------|------|------|-------------|
| | | ML | FU | FT | ML | FU | FT | F/I(NO, FE) |
| EN I | (0.9, 8) | 0.61 | 0.45 | 1.52 | 0.61 | 0.45 | 1.54 | F(-) |
| EN II | (0.9, 4) | 0.58 | 0.41 | 1.48 | 0.58 | 0.41 | 1.41 | F(-) |
| EN III | (0.5, 32) | 0.64 | 0.42 | 1.46 | 0.64 | 0.42 | 1.42 | F(-) |
| EN IV | (0.9, 90) | 0.69 | 0.42 | 1.59 | 0.69 | 0.42 | 1.50 | F(-) |

REFERENCES

- [1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, "Network Flows: Theory, Algorithms, and Applications," Prentice Hall, 1993.
- [2] D. Bertsekas and R. Gallager, "Data Networks—2nd Edition," Prentice Hall, 1992.
- [3] A. Bley, "A Lagrangian Approach for Integrated Network Design and Routing in IP Networks," *Proceedings of International Network Optimization Conference (INOC)*, 2003.
- [4] N. Bouruia, W. B. Ameer, E. Gourdin and P. Tolla, "Optimal Shortest path routing for Internet networks," *Proceedings of International Network Optimization Conference (INOC)* 2003.
- [5] R. Callon, "Use of OSI IS-IS for routing in TCP/IP and dual environments," Internet Engineering Task Force, Requests for Comments (Standard), RFC 1195, December, 1990.
- [6] Cisco, *Configuring OSPF*, 1997.
- [7] *Manual of Cplex Callable Libraries*, ILOG Corporation, USA.
- [8] G. Dahl, M. Stoer, "A Cutting Plane Algorithm for Multi-commodity Survivable Network Design Problems," *INFORMS Journal of Computing*, vol. 10, pp. 1–11, 1998.
- [9] M. Ericsson, M. G. C. Resende, P. M. Pardalos, "A Genetic Algorithm for the weight setting problem in OSPF routing," *Journal of Combinatorial Optimization*, Vol. 6, No. 3, pp. 229–333, 2002.
- [10] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, J. Rexford, and F. True, "Deriving traffic demands for operational IP networks: Methodology and experience," *IEEE/ACM Transactions on Networking*, pp. 265–279, 2001.
- [11] B. Fortz, M. Thorup, "Internet traffic engineering by optimizing OSPF weights," *Proceedings of INFOCOM*, pp. 519–528, 2000.
- [12] A. M. Geoffrion, "Lagrangian Relaxation for Integer Programming," *Mathematical Programming Study*, vol. 2, pp. 82–114, 1974.
- [13] A. M. Geoffrion, and R. McBride, "Lagrangian Relaxation Applied to Capacitated Facility Location Problems," *AIIE Transactions*, vol. 10, no. 1, pp. 40–47, 1978.
- [14] J. Harmatos, "A Heuristic Algorithm for Solving the Static Weight Optimisation Problem in OSPF Networks," *Proceeding of GLOBECOM*, 2001.
- [15] M. Held, P. Wolfe and H. Crowder, "Validation of subgradient optimization," *Mathematical Programming*, vol. 6, 62–88, 1974.
- [16] D. Medhi, D. Tipper, "Some approaches to solving a multihour broadband network capacity design problem with single-path routing," *Telecommunications System*, vol. 13, pp. 269–291, 2000.
- [17] A. Medina, A. Lakhina, I. Matta and J. Byers, "BRITE: Universal Topology Generation from a User's Perspective," Boston University Technical Report BU-CS-TR-2001-003. April 05, 2001. See also: <http://www.cs.bu.edu/brite/>
- [18] M. Minoux, *Mathematical Programming: Theory and Algorithms*, John Wiley & Sons, 1986.
- [19] J. Moy, "OSPF version 2," Internet Engineering Task Force, Request for Comments (Standard), RFC 2328, April 1998.
- [20] J. Moy, *OSPF: Anatomy of an Internet Routing Protocol*, Addison Wesley, 1999.
- [21] M. Pióro and D. Medhi, *Routing, Flow, and Capacity Design in Communication and Computer Networks*, Morgan Kaufmann Publishers, 2004.
- [22] M. Pióro, A. Szentsi, J. Harmatos, A. Juttner, P. Gajowniczek and S. Kozdrowski "On open shortest path first related network optimisation problems," *Performance Evaluation*, vol. 48, pp. 201–223, 2002.
- [23] K. G. Ramakrishnan, M. A. Rodrigues, "Optimal Routing in Shortest-Path Data Networks," *Bell Labs Technical Journal*, vol. 6, no. 1, pp. 117–138, 2001.
- [24] R. Rastogi, Y. Breitbart, M. Garofalakis, A. Kumar, "Optimal

TABLE X
RESULTS FOR (\mathbf{PA}_1) FOR 100-NODE NETWORKS

| Nodal Degree (# of Links) | (\mathbf{PA}_1) | | | | | | |
|---------------------------|---------------------|------|------|-------------|-------------|----------------|----------------|
| | ML | FU | FT | Duality Gap | F/I(NO, FE) | Dual Iteration | Computing Time |
| 2 (197) | 0.79 | 0.25 | 1.22 | 0.6% | F(-) | 1000 | 44s |
| 3 (294) | 0.68 | 0.24 | 1.15 | 0.5% | F(-) | 250 | 13s |
| 4 (390) | 0.48 | 0.18 | 1.04 | 0.6% | F(-) | 1000 | 60s |
| 5 (485) | 0.50 | 0.19 | 1.04 | 0.2% | F(-) | 250 | 20s |
| 6 (579) | 0.49 | 0.17 | 1.02 | 0.4% | F(-) | 250 | 23s |

TABLE XI
RESULTS FOR (\mathbf{PA}_3) FOR 100-NODE NETWORKS

| Nodal Degree (# of Links) | (α, β) | (\mathbf{PA}_3) | | | | | | |
|---------------------------|-------------------|---------------------|------|------|-------------|-------------|----------------|----------------|
| | | ML | FU | FT | Duality Gap | F/I(NO, FE) | Dual Iteration | Computing Time |
| 2 (197) | (0.9, 32) | 0.67 | 0.25 | 1.15 | 0.1% | F(-) | 250 | 5s |
| 3 (294) | (0.9, 11585) | 0.66 | 0.24 | 1.15 | 4.4% | F(-) | 1000 | 21s |
| 4 (390) | (0.9, 2896) | 0.37 | 0.18 | 1.00 | 0.7% | F(-) | 1000 | 19s |
| 5 (485) | (0.9, 2896) | 0.36 | 0.19 | 1.00 | 0.5% | F(-) | 774 | 16s |
| 6 (579) | (0.9, 16) | 0.39 | 0.17 | 1.00 | 0.1% | F(-) | 250 | 5s |

configuration of OSPF aggregates,” *IEEE/ACM Transactions on Networking*, vol. 11, pp. 181-194, 2003.

- [25] A. Sridharan, R. Guerin, C. Diot, “Achieving Near-Optimal Traffic Engineering Solutions for current OSPF/IS-IS Networks,” *Proceedings of IEEE INFOCOM*, 2003.
- [26] S. Srivastava, B. Krithikaivasan, D. Medhi and M. Pióro, “Traffic Engineering in the Presence of Tunneling and Diversity Constraints: Formulation and Lagrangian Decomposition Approach,” *Proceedings of 18th International Teletraffic Congress*, pp. 461-470, Berlin, Germany, 2003.
- [27] Y. Wang, Z. Wang and L. Zhang, “Internet Traffic Engineering without Full Mesh Overlaying,” *Proceedings of IEEE INFOCOM*, pp. 565-571, 2001.
- [28] Y. Zhang, M. Roughan, C. Lund, and D. Donoho, “An Information-Theoretic Approach to Traffic Matrix Estimation,” *Proceedings of ACM SIGCOMM*, 2003.

BIOGRAPHY

Shekhar Srivastava completed Ph.D. in Computer Networking from the Computer Science & Electrical Engineering Department at the University of Missouri-Kansas City in December 2004. He is currently a post-doctoral researcher in INRS-EMT, Montreal, Canada. He received his B.Tech. in Electronic and Communications Engineering from Govind Bhallabh Pant University of Agriculture & Technology, India in 1997, and his M.E. in Telecommunications from the Indian Institute of Science, Bangalore, India 1999. Between March 1999 and July 2000, he was a research engineer at Sasken Communication in Bangalore, India working on GPRS/GSM access control and simulation. His research interests are Traffic Engineering and QoS Routing, Performance Modeling and Queueing Theory, and Optimization Algorithms.

Gaurav Agrawal is a doctoral student in Computer Networking in the Computer Science & Electrical Engineering Department at the University of Missouri-Kansas City. He received his B.E. in Civil Engineering from the Indian Institute of Technology-Roorkee (formerly, University of Roorkee) in 1997, and his M.S. in Computer Science with emphasis in Computer Networking in 2001. His research interests are Internet Traffic Engineering, Network Design, Optimization and Performance, Multicasting in the Internet.

Michał Pióro is Professor and Head of Department of Computer Networks and Switching, Institute of Telecommunications, Warsaw University of Technology, Poland and

also Professor, Department of Communication Systems, Lund University, Sweden. He received his M.Sc. in Applied Mathematics (with distinction) from Warsaw University of Technology (WUT) in 1973, his PhD in telecommunications (with distinction) from WUT in 1979, and his DSc (Doctor of Science/Doctor Habilitated) in telecommunications, also from WUT, in 1990. He has published more than 100 technical papers in various telecommunication journals and conference proceedings. He has authored four books including co-authoring *Routing, Flow, and Capacity Design in Communication and Computer Networks*, published by Morgan Kaufmann Publishers (Elsevier). He has lead many research projects for telecom industry in the field of network modeling, design, and performance analysis.

Deep Medhi is Professor of Computer Networking in the Computer Science & Electrical Engineering (CSEE) Department at the University of Missouri-Kansas City (UMKC), USA where he was honored as a UMKC Trustees’ Faculty Fellow for the academic year 2004-2005. He received the B.Sc.(Hons.) degree in Mathematics from Cotton College, Gauhati University, India, the M.S. degree in Mathematics from the University of Delhi, India, and the M.S. and Ph.D. degrees in Computer Sciences from the University of Wisconsin-Madison, USA in 1981, 1983, 1985 and 1987, respectively. Prior to joining UMKC in 1989, he was a member of the technical staff in the traffic network routing and design department at the AT&T Bell Laboratories, Holmdel, New Jersey from 1987 to 1989. He has published over sixty peer-reviewed papers and is co-author of the recent book, *Routing, Flow, and Capacity Design in Communication and Computer Networks*. His research interests are in survivable network design and architecture, dynamic routing, next generation network architecture, IP network availability, network management, and medical informatics. Over the past decade, his research has been funded by Defense Advanced Research Project Agency (DARPA), National Science Foundation (NSF), Sprint Corporation, and the Doris Duke Charitable Foundation.