New Models of Network Routing under Active Congestion Control

Stanko Dimitrov* Marina Epelman Dushyant Sharma
Department of Industrial and Operations Engineering
University of Michigan, Ann Arbor, MI 48109
{sdimitro, mepelman, dushyant}@umich.edu

September 22, 2009

Abstract

In this paper we propose a model for creating routing policies in networks with active congestion control. Though we prove the resulting model is $\mathcal{NP}$-hard, we show that routing policies generated using the model outperform those currently deployed in a real world computer network. We further propose a formulation of the problem that takes into account typical demand fluctuations in designing a robust routing policy that, again, outperforms current routing policies in a computer network.

1 Introduction

In this paper we consider network routing under congestion control. We focus on *active* congestion control. We say that a congestion control method is active if the amount of flow sent into a network component, such as an arc, is a function of the network status; for example, in computer networks, some routers are designed for congestion control and as information is passed through the routers some packets may be dropped (Contrast this with a congestion control method that preserves the amount of flow on every component of the network; for instance, a road network, in which cars travel slower on a congested highway, but remain in the network until they reach their destinations.). Though the proposed model and techniques easily extend to most networks with congestion control, we focus on computer networks with active congestion control. In the remainder of this section we first present background required to understand the current routing policies and congestion control techniques used in computer networks, and then describe the question we address in this paper. In Section 2 we present our mathematical model for determining routing policies in a network with a particular type of congestion control and show, in Section 3, that it is $\mathcal{NP}$-hard. In Section 4 we present a real world computer network, and discuss the performance of different routing policies generated using the model in Sections 5. Finally, in Section 6 we propose a robust formulation of the model and present its performance relative to the routing policy currently used.

*corresponding author

1.1 Model Idea

We design a routing policy in a computer network using a generalization of a multi-commodity network flow model described, for instance, by Ahuja *et al.* [3]. In our model, every origin and destination (OD) pair in the network constitute a commodity, and every commodity has a fixed amount of demand that is sent from the origin node to the destination node. The main difference, as we discuss later, is that the amount of flow of each commodity received at the head of an arc is a function of the total flow on that arc. After introducing some background, we will revisit the model and explain it, and its relation to previous work, in greater detail.

1.2 Computer Network Background

In this section we discuss some of the current network protocols used in computer networks and point out which ones will be captured by the proposed mathematical model. In a telecommunications network, one has to make two decisions: one is what path(s) the information is going use from the source to the destination, using a routing protocol, and the other is what type of flow management will be used to improve quality of service in the network, using a network end-to-end protocol. In particular, we will describe Open Shortest Path First (OSPF) and Multiprotocol Label Switching (MPLS) routing protocols, and the implications of each one for finding routing policies used in the network. We will then describe Random Early Detection (RED), an active congestion control method currently available in computer networks, and the implications of taking RED into account when generating an optimal routing policy.

As a notational issue, it is important to note that computer networks transmit information in packets, or discrete blocks of information. However, in a network flow model each commodity is thought to be continuous and as such we can think of each commodity's flow in the proposed model as the rate of packets for that commodity in the computer network.

1.2.1 Network Protocols

Computer networks tend to use a variant of one of two end-to-end transmission protocols. One is the transmission control protocol (TCP) and the other is the user datagram protocol (UDP).

A packet sent using TCP is acknowledged by the receiver and, if a sender does not receive the acknowledgment in a given timeframe, the packet is re-sent after waiting an exponentially increasing amount of time. As one would expect, TCP inherently slows down the throughput due to the constant acknowledgments that are sent back to the senders. A network using TCP, by definition of the protocol, has built-in active congestion control in that it modifies the transmission rate, *i.e.*, the effective demand of each commodity, in response to congestion in the network.

UDP is a protocol commonly used to transmit Voice Over Internet Protocol (VoIP), Internet Protocol Television (IPTV), and Instant Messaging (IM) services traffic. It does not have built-in acknowledgment for every received packet, and all packets are sent once, with no guarantee of being received. If UDP is used without any congestion control, severe congestion could occur in the network. Moreover, a network using UDP without congestion control is susceptible to a denial of service attack during which an attacker floods the network with UDP packets. With the growing prominence of VoIP, IPTV and IM traffic on the Internet, examining the impact of active congestion control on computer networks with UDP traffic is of interest.

1.2.2 Routing Protocols

Open Shortest Path First (OSPF)

The Open Shortest Path First routing protocol, OSPF, was first proposed in 1989 and has been modified four times since the original request for comments (RFC) was posted. The RFC for the current version of OSPF was posted in April of 1998 [2]. OSPF is a routing policy used in intranet networks, *i.e.*, Autonomous Systems, which are networks administered by a single organization. Abstractly OSPF can be described as routing demand along a single path for every origin and destination (OD) pair in the network, namely, the shortest path between those origin and destination nodes. As a rule of thumb, Cisco Systems recommends the arc weights used in the shortest path calculation to be set to the inverse of the arc capacities [27]. In practice, since arc capacities seldomly change, the paths between nodes are updated rather infrequently. As most networks, specifically the ones we examine in Section 4, are still using OSPF, we will use network performance under OSPF routing as a benchmark for assessing network performance under Multiprotocol Label Switching, which we will describe next.

Multiprotocol Label Switching (MPLS)

Multiprotocol Label Switching, MPLS, was proposed in January of 2001 [22]. MPLS differs from OSPF in that each OD pair in the network has multiple paths, which may or may not be disjoint, simultaneously able to carry positive flow from the origin to the destination.

When a packet first enters the network, the incoming router looks at the destination of the packet and chooses which of the possible paths it should follow to its destination. It then assigns the appropriate label to the packet and forwards it to the first node in the determined path.

Associated with each intermediate router in the network is an *MPLS routing table*. Each row of the table contains information that determines, given the neighbor the packet came from and its current label, the next node on the packet's path to its destination, and a new label to attach to the packet (the new label can be interpreted by the next router in the same manner). Once a packet is received by a router, the router removes the current label of the packet, identifies the appropriate row of its MPLS table, and attaches a new label and forwards the packet accordingly.

The proposed research is to find a routing policy which is feasible in a network using MPLS while accounting for both active congestion control and demand uncertainty. Though currently OSPF is used in intranet networks, most networks are beginning to port to MPLS. Therefore, now is the time to address the issue of finding good MPLS routing policies for networks facing congestion control and demand uncertainty.

1.2.3 Random Early Detection (RED)

In a computer network, whenever a packet, or a datagram of information, is forwarded from one router to another, that packet must be examined by the forwarding router. If two or more packets need to be examined, then all packets not being examined are placed in a queue of finite capacity, say u, and serviced in a first in first out (FIFO) manner. However, if the rate of incoming packets is greater than the router service rate, the queue will reach capacity u and no incoming packets will be enqueued. To address the issue of the resulting starvation, the Random Early Detection, RED, congestion avoidance mechanism was introduced [14]. The mechanism separates the router queue capacity into three regions, characterized by parameters $0 \leq \beta \leq \gamma \leq u$. In the first region, between

an empty queue and a queue length of β, all incoming packets are enqueued to wait for service. In the second region, with queue length between β and γ, the probability that a packet is enqueued is determined by a decreasing linear function with a slope of $-\alpha$, for $\alpha > 0$. Finally, in the third region, with queue length between γ and u, none of the incoming packets are enqueued into the router queue. Since γ determines the effective capacity of the queue, without loss of generality we will let $\gamma = u$ for the remainder of this paper.

1.3 Gain Functions

Though our work is motivated by an application to computer network routing, the nominal problem being addressed is an extension of the generalized network flow problem, as described by Ahuja *et al.* [3, Chapter 15]. In a single commodity setting, we denote by x_{ij} the amount of flow sent from node i to node j on arc (i, j) and by y_{ij} the amount of flow received at node j from node i on arc (i, j). In the classic network flow model we would have $y_{ij} = x_{ij}$. In a generalized flow setting, however, we have $y_{ij} = x_{ij}\mu_{ij}$, where $\mu_{ij} > 0$ represents the proportional loss or gain of flow on arc (i, j). In a network with congestion control imposed by the RED algorithm, we have $y_{ij} = x_{ij}f_{ij}(x_{ij})$. In this setting $f_{ij}(x_{ij})$ represents the loss, due to congestion at router at node i, that takes place on arc (i, j). Since $f_{ij}(x_{ij})$ is a function of x_{ij}, this new model is a further generalization of the generalized network flow model.

As noted in section 1.2.3, we can define a function $g(t)$, the probability of a packet being enqueued by the router as a function of the queue length t, as

$$g(t) = \begin{cases} 1 & 0 \le t \le \beta, \\ 1 - \alpha(t - \beta) & \beta \le t \le u, \\ 0 & u \le t. \end{cases}$$

We chose $\alpha = \frac{1}{u-\beta}$ to guarantee continuity of $g(\cdot)$. Note also that with this choice of α the effective capacity of the arc is u, as desired. As defined, $g(t)$ determines the loss on an arc as a function of y_{ij}, because every enqueued packed will be serviced. Therefore, the $f(t)$ function satisfies $f(t) = g(tf(t))$. When $g(t)$ is defined as above, the resulting $f(t)$ function is:

$$f(t) = \begin{cases} 1 & 0 \le t \le \beta, \\ \frac{1+\alpha\beta}{1+\alpha t} & \beta \le t. \end{cases} \tag{1}$$

We denote by $h(t) = tf(t)$ the amount of flow received as a function of t. With $f(t)$ given by (1),

$$\begin{aligned} h(t) &= tf(t) \\ &= \begin{cases} t & 0 \le t \le \beta \\ \frac{1+\alpha\beta}{\alpha} - \frac{1+\alpha\beta}{\alpha(1+\alpha t)} & \beta < t, \end{cases} \end{aligned}$$

and with α as above, $h(t) \to u$ as $t \to \infty$, ensuring that the capacity is not exceeded. Observe also that $h(t)$ is non-decreasing and concave.

In a multi-commodity setting that we will consider in this paper, the flow conservation relationship becomes $y_{ij}^k = x_{ij}^k f\left(\sum_l x_{ij}^l\right)$ for each arc (i, j) and each commodity k.

Shigeno [26] studied the problem of finding optimal routing polices for a single commodity using congestion control. He refers to the function $f(t)$ above as a *gain function*, and defines a *concave gain function* to be a gain function $f(t)$ such that $h(t) = tf(t)$ is concave and non-decreasing. Note that the RED congestion function $f(t)$ given by (1) is less than or equal to 1 for all t, and thus represents a *loss* of flow on an arc. However, to stay consistent with Shigeno we refer to $f(t)$ as a gain function.

1.4 Proposed problem

In this paper we propose a mathematical model for Internet routing under active congestion control. We propose a continuous flow model (rather than a packet burst model) that is motivated by computer networks using UDP with RED deployed in the network. Though the proposed model is $\mathcal{NP}$-hard, we show that routing policies obtained by applying a nonlinear programming solver to the model improve network performance over existing routing policies for a computer network.

1.5 Previous Research

Shigeno [26] introduced the concept of a *concave gain function*, and associated a concave gain function with every arc in a network. He showed that in a single commodity network flow problem with concave gains a routing policy maximizing the total flow received at the destination node can be found in polynomial time. As our problem is a multi-commodity flow problem with concave gains, it helps to think of the proposed problem as the multi-commodity flow generalization of Shigeno's work.

Several studies besides Shigeno have looked at generating routing policies in networks while taking into account congestion. For example, Sheffi [24] addresses the issue of roadway congestion in optimal traffic selection. In his models he proposes a convex function representing the travel time on a roadway (*i.e.*, an arc of a network). As the number of users of a roadway segment increases, so does their travel time on that segment, until the capacity of the roadway is reached. The users of the roadway are assumed to minimize their total travel time. This model is similar to the one of interest; however, all passengers on a roadway remain on a roadway, while in our model we can remove users. Studies addressing demand uncertainty using a robust routing scheme will be discussed in detail in section 6.

2 Multi-Commodity Network Flow problem with Nonlinear Gains

We begin by defining the Multi-Commodity Network Flow problem with Nonlinear Gains (MCFPNG). Let $G = (N, A)$ be a directed graph with node set N and arc set A, and $(o^k, d^k) \in N \times N$ for $k = 1, \ldots, K$ be origin-destination node pairs for K commodities. We consider the arcs to have infinite capacity, and let $f_a : \mathbb{R}_+ \to \mathbb{R}_+$ for $a \in A$ be *gain functions* associated with each of the arcs. Let s^k be the supply of commodity k originating at its origin o^k, and c^k be the value per unit of this commodity delivered to destination d^k. We will use the following notation:

- $x_i^k \equiv$ amount of flow of commodity k present at node $i \in N$;
- $\alpha_{ij}^k \equiv$ the fraction of commodity k present at node i sent to node j, on arc $(i, j) \in A$;

- $\delta^+(v)$ and $\delta^-(v)$ denote the sets of nodes in N that are end points of arcs coming out of node v and coming into node v, respectively.

There are several possible ways of defining the MCFPNG; our version is, essentially, a weighted maximum flow problem with multiple commodities and (nonlinear) gain functions on each arc.

Specifically, we define (MCFPNG) as:

$$\text{(MCFPNG)} \quad \max_{\alpha,\, x} \quad \sum_{k=1}^{K} c^k x^k_{d^k} \tag{2}$$

$$\text{s.t.} \quad x^k_{o^k} = s^k \qquad k = 1, \ldots, K \tag{3}$$

$$\sum_{(i,j)\in A} \alpha^k_{ij} = 1 \qquad k = 1, \ldots, K,\ i \in N : i \neq d^k \tag{4}$$

$$\sum_{(i,j)\in A} \alpha^k_{ij} = 0 \qquad k = 1, \ldots, K,\ i = d^k \tag{5}$$

$$\sum_{i\in\delta^-(j)} \alpha^k_{ij} x^k_i f_{ij} \left(\sum_{l=1}^{K} \alpha^l_{ij} x^l_i \right) - x^k_j = 0 \qquad k = 1, \ldots, K,\ j \in N : j \neq d^k \tag{6}$$

$$\alpha^k_{ij} \geq 0 \qquad (i,j) \in A,\ k = 1, \ldots, K. \tag{7}$$

Here, the objective function (2) maximizes the weighted sum of flows of each commodity delivered to the destination nodes, while constraints (3) indicate the supply of each commodity. Constraints (4) and (5), together with (7), ensure that the entire amount of commodity k available at node i is routed along the edges emanating from i, with the exception of the destination node for that commodity. Constraints (6) calculate the amount of commodity k available at node j by tracking the flow of that commodity routed, and lost, on each of the arcs coming into node j.

3 Complexity of MCNFCG

This section is dedicated to proving the following

Theorem 1. *MCFPNG is $\mathcal{NP}$-hard.*

The proof is done by reduction to the Set Cover problem:

Definition 2. Set Cover *in minimization form is defined as follows: given*

- *set $U = \{e_1, e_2, \ldots, e_m\}$ and*
- *collection of subsets $S_j \subseteq U, \quad j \in \{1, 2, \ldots, n\}$,*

find a minimum set cover, i.e., set $J \subseteq \{1, 2, \ldots, n\}$ such that $\cup_{j\in J} S_j = U$, of minimum cardinality.

Given a Set Cover instance, we construct the following directed graph G, as a 4 layer network:

Layer 1 consists of one node for every element, $e_i \in U,\ i \in \{1, \ldots, m\}$,

Layer 2 consists of one node for every subset, $S_j,\ j \in \{1, \ldots, n\}$,

Layer 3 consists of one node, I,

Layer 4 consists of one node, t.

The layers are connected in the following manner:

- For every i and j such that $e_i \in S_j$ there is a directed arc (e_i, S_j),
- For every j there is a directed arc (S_j, I),
- There is a directed arc (I, t).

For example, consider the following instance of set cover:

$$\begin{array}{rcl} U & = & \{e_1, e_2, e_3, e_4, e_5\} \\ S_1 & = & \{e_1, e_2, e_3\} \\ S_2 & = & \{e_1, e_2, e_3, e_4\} \\ S_3 & = & \{e_1, e_4\} \\ S_4 & = & \{e_1, e_5\}. \end{array}$$

The corresponding directed graph constructed as described above is depicted in Figure 1.

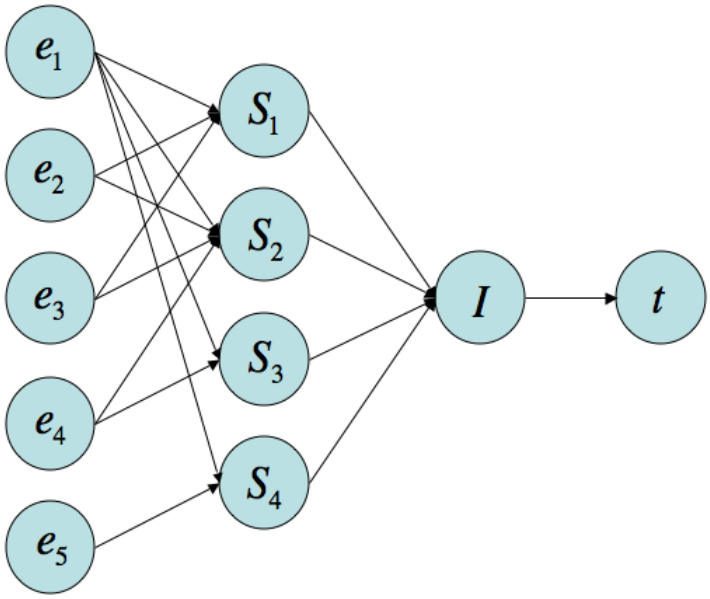


Figure 1: Example of graph construction

To define an instance of MCFPNG, we define the following $m + 1$ commodities:

Commodity 0 from origin node I to destination node t, *i.e.*, $o^0 = I$ and $d^0 = t$, with $s^0 = 1$ and $c^0 = 1$;

Commodity i, for $i = 1, \ldots, m$ from origin node e_i to destination node t, *i.e.*, $o^i = e_i$ and $d^i = t$, with $s^i = 1$ and $c^i = 0$.

Finally, the gain function of each arc is

$$f_a(t) = f(t) = \begin{cases} 1 & t < 1 \\ \frac{1}{t} & 1 \leq t \end{cases}$$

for all $a \in A$, fitting the definition of a concave gain function. Though we can consider the capacities of the arcs to be infinite, notice that the form of the gain function $f(t)$ above implies that the effective capacity of each arc is 1.

Note that the there were $O(nm)$ steps required to transform the instance of set cover into an instance of MCFPNG.

Given an instance of Set Cover, the corresponding MCFPNG instance resulting from this transformation is:

$$\begin{aligned}
(\text{MCFPNG}) \quad \max_{\alpha, x} \quad & x_t^0 && && (8)\\
\text{s.t.} \quad & x_I^0 = 1 && && (9)\\
& x_{e_k}^k = 1 && k = 1, \ldots, m && (10)\\
& \sum_{(i,j) \in A} \alpha_{ij}^k = 1 && k = 0, \ldots, m,\ i \in N : i \neq d^k && (11)\\
& \sum_{(i,j) \in A} \alpha_{ij}^k = 0 && k = 0, \ldots, m,\ i = d^k && (12)\\
& \sum_{i \in \delta^-(j)} \alpha_{ij}^k x_i^k f_{ij} \left(\sum_{l=0}^{m} \alpha_{ij}^l x_i^l \right) - x_j^k = 0 \qquad && k = 0, \ldots, m,\ j \in N : j \neq d^k && (13)\\
& \alpha_{ij}^k \geq 0 && (i,j) \in A,\ k = 0, \ldots, m. && (14)
\end{aligned}$$

Below we explore some of the properties of optimal solutions to this problem. For convenience, we will use the following notation in the rest of this section:

$$X_v = \sum_{i=1}^{m} x_v^i,\ v \in N, \tag{15}$$

i.e., X_v denotes the total amount of flow of commodities 1 through m present at node v.

Proposition 3. *A feasible solution (α, x) of (8) – (14) is optimal if and only if it minimizes X_I.*

Proof. Recall that

$$X_I = \sum_{i=1}^{m} x_I^i.$$

For any feasible solution (α, x), $\alpha_{It}^k = 1\ \forall k$ due to (11), as there is only one arc out of I. In addition, $x_I^0 = s^0 = 1$, and so $X_I + x_I^0 \geq 1$, implying that $f(X_I + x_I^0) = \frac{1}{X_I + x_I^0}$. Moreover, $x_t^0 = x_I^0 \cdot f(X_I + x_I^0) = \frac{1}{1 + X_I}$. Therefore, as the objective of (8) is to maximize x_t^0, a feasible solution is optimal if and only if it minimizes X_I. □

Proposition 4. *There exists an optimal solution (α, x) which has integral values (0 or 1) of variables α and x associated with all arcs from layer 1 to layer 2.*

Proof. If in a feasible solution the values of α are integral (0 or 1) for all arcs from layer 1 to layer 2, then, according to (11), this solution has positive flow on exactly one arc (e_i, S_j) for each $i = 1, \ldots, m$. Since the supply at each node e_i is $s^i = 1$, the amount routed on all arcs from layer 1 to layer 2 is 0 or 1. Thus, we only need to show existence of a solution with integral values of α associated with all such arcs.

Let (α, x) be an optimal solution, and suppose that there is a node e_i in layer 1 such that the α values at this node to layer 2 split commodity i between two or more arcs. Without loss of generality, let the corresponding nodes in layer 2 be S_1 and S_2, *i.e.*, $\alpha_{e_i,S_1} > 0$ and $\alpha_{e_i,S_2} > 0$.

Recall that the total flow of commodities $1, \ldots, m$ present at node I is equal to $X_I = \sum_{j=1}^{n} X_{S_j} f(X_{S_j})$, by construction of G. Without loss of generality, assume $X_{S_1} \geq X_{S_2}$.

Consider solution $(\tilde{\alpha}, \tilde{x})$ that is obtained from solution (α, x) by moving the flow of commodity i from arc (e_i, S_2) to arc (e_i, S_1), *i.e.*, $\tilde{\alpha}^i_{e_i,S_2} = 0$ and $\tilde{\alpha}^i_{e_i,S_1} = \alpha^i_{e_i,S_1} + \alpha^i_{e_i,S_2}$, while all other α values at layer 1 nodes remains the same. It is easy to verify that solution $(\tilde{\alpha}, \tilde{x})$ is feasible. We will show that

$$\sum_{j=1}^{n} \tilde{X}_{S_j} f(\tilde{X}_{S_j}) \leq \sum_{j=1}^{n} X_{S_j} f(X_{S_j}),$$

and thus, in view of Proposition 3, $(\tilde{\alpha}, \tilde{x})$ is optimal. In fact, it only needs to be shown that

$$\tilde{X}_{S_1} f(\tilde{X}_{S_1}) + \tilde{X}_{S_2}, f(\tilde{X}_{S_2}) \leq X_{S_1} f(X_{S_1}) + X_{S_2} f(X_{S_2}),$$

since the other values remain unchanged.

Due to the supply constraint at node e_i, $x^i \alpha^i_{(e_i,S_1)} + x^i \alpha^i_{(e_i,S_2)} = x^i \tilde{\alpha}^i_{(e_i,S_1)} + x^i \tilde{\alpha}^i_{(e_i,S_2)} \leq 1$. Therefore, by the definition of $f(\cdot)$, $x^i_{S_1} = \alpha^i_{(e_i,S_1)} x^i_{e_i}$, $x^i_{S_2} = \alpha^i_{(e_i,S_2)} x^i_{e_i}$, $\tilde{x}^i_{S_1} = \tilde{x}^i_{e_i} = x^i_{e_i}(\alpha^i_{(e_i,S_1)} + \alpha^i_{(e_i,S_2)})$ and $\tilde{x}^i_{S_2} = x^i_{e_i} \tilde{\alpha}^i_{(e_i,S_2)} = 0$.

Consider the following two cases:

Case 1: $X_{S_2} < 1$**.** In this case $X_{S_2} f(X_{S_2}) = X_{(S_2,I)}$ and $\tilde{X}_{S_2} f(\tilde{X}_{S_2}) = \tilde{X}_{S_2} = X_{S_2} - x^i_{S_2}$. Furthermore, $\tilde{X}_{S_1} f(\tilde{X}_{S_1}) = (X_{S_1} + x^i_{S_2}) \cdot f(X_{S_1} + x^i_{S_2})$. Notice that the function $f(t)$ satisfies

$$(t + \Delta)\, f(t + \Delta) \leq t\, f(t) + \Delta \text{ for all } t \geq 0 \text{ and } \Delta \geq 0,$$

and so

$$\begin{aligned} \tilde{X}_{S_1} f(\tilde{X}_{S_1}) + \tilde{X}_{S_2}, f(\tilde{X}_{S_2}) &= X_{S_2} - x^i_{S_2} + (X_{S_1} + x^i_{S_2}) \cdot f(X_{S_1} + x^i_{S_2}) \\ &\leq X_{S_2} - x^i_{S_2} + X_{S_1} f(X_{S_1}) + x^i_{S_2} \\ &= X_{S_1} f(X_{S_1}) + X_{S_2} f(X_{S_2}), \end{aligned}$$

as desired.

Case 2: $X_{S_1} \geq X_{S_2} \geq 1$**.** Notice that in this case $X_{S_1} f(X_{S_1}) + X_{S_2} f(X_{S_2}) = 2$. In the new solution, $\tilde{X}_{(S_1,I)} \geq X_{(S_1,I)} \geq 1$, and hence $\tilde{X}_{S_1} f(\tilde{X}_{S_1}) = 1$. On the other hand, due to the form of the gain function, $\tilde{X}_{S_2} f(\tilde{X}_{S_2}) \leq 1$, and hence the total amount of commodities 1 though m arriving at node I will not increase.

Modifications above removed the flow on one of the arcs between layers 1 and 2 without loss of optimality. Applying this procedure repeatedly will generate an optimal solution with positive flow on exactly one arc (e_i, S_j) for each $i = 1, \ldots, m$.

Finally, notice that each step of the above modification procedure takes a constant amount of time to execute, and needs to be applied at most mn times. □

The procedure outlined in the above proof can be applied to any feasible solution (α, x), producing a feasible solution $(\tilde{\alpha}, \tilde{x})$ with integral values on arcs from layer 1 to layer 2 and with the same or better objective function value in at most mn steps.

Proposition 5. *A feasible solution of* (8) – (14) *with integral flow (0 or 1) on each arc from layer 1 to layer 2 is optimal if and only if it minimizes the total number of arcs from layer 2 to layer 3 with nonzero flow.*

Proof. In any feasible solution (α, x) with integral flow (0 or 1) on each arc from layer 1 to layer 2, $X_{S_j} = \sum_{i=1}^{m} x^i \alpha^i_{(e_i,S_j)} \in \mathbb{Z}_+$ for all $j = 1, \ldots, n$. Therefore, for any j,

$$X_{S_j} f(X_{S_j}) = \begin{cases} 0 & X_{S_j} = 0, \\ 1 & X_{S_j} > 0, \end{cases}$$

and hence

$$X_I = \sum_{j=1}^{n} X_{S_j} f(X_{S_j}) = \sum_{j=1}^{n} \mathbb{I}(X_{S_j} > 0),$$

which, in view of Proposition 3, implies the conclusions of the proposition. □

Proposition 6. *Suppose* (α, x) *is an optimal solution of* (8) – (14) *with integral flow (0 or 1) on each arc from layer 1 to layer 2. Then the set* $J = \{j : X_{(S_j,I)} > 0\}$ *is an optimal solution to the corresponding instance of the minimum set cover problem.*

Proof. First notice that any feasible solution of MCFPNG (8) – (14) with integral flow on each arc from layer 1 to layer 2 corresponds to a set cover J constructed as follows: $j \in J$ if and only if $\alpha^i_{(e_i,S_j)} = 1$ for some e_i, or equivalently, $X_{S_j} > 0$ (and integral). Supply constraints at e_i, $i = 1, \ldots, m$, imply that J is, indeed, a cover, and we have

$$X_I = \sum_{j=1}^{n} X_{S_j} f(X_{S_j}) = \sum_{j=1}^{n} \mathbb{I}(X_{S_j} > 0) = |J|.$$

Conversely, every minimal set cover J (i.e., one that does not contain a set cover of smaller cardinality) can be represented by a feasible solution of the corresponding MCFPNG with integral flow on each arc from layer 1 to layer 2 as follows: For each $i = 1, \ldots, m$, pick any $j \in J$ such that $e_i \in S_j$ and set $\alpha^i_{(e_i,S_j)} = 1$, *i.e.*, direct 1 unit of flow along the arc (e_i, S_j) (since J is a set cover, such j can always be found). Assign flows on arcs out of layer 2 and layer 3 nodes accordingly to satisfy flow gain constraints for the arcs and flow balance constraints for the nodes. Again, we have

$$|J| = \sum_{j=1}^{n} \mathbb{I}(X_{S_j} > 0) = \sum_{j=1}^{n} X_{S_j} f(X_{S_j}) = X_I.$$

Thus, by Proposition 5, the minimum set cover can be obtained by finding an optimal solution to MCNFNG (8) – (14) with integer flows on arcs from layer 1 to layer 2. □

To complete the reduction, in view of Proposition 6, one only needs to recall that an arbitrary optimal solution to (8) – (14) can be modified in at most mn steps into one with integral flows from layer 1 nodes to layer 2 nodes.

We have thus established that the MCFPNG as modeled by (2) – (7) is $\mathcal{NP}$-hard. Nonetheless, a nonlinear programming solver can be used to successfully find locally optimal solutions to instances of the MCFPNG. The remainder of the paper is dedicated to numerical experiments with this model.

4 Abilene Network

As a testbed for numerical experiments in the following sections we used the Abilene Network depicted in Figure 2. The Abilene Network is the backbone network of the Internet2 community. The Internet2 community is a not for profit consortium of universities, companies, and government agencies that develops and deploys advanced network applications critical to the progress of the Internet. We have obtained network usage data in the Abilene Network in the form of 24 weekly data sets during a 6-month period of time in 2004 (March 1, 2004 to September 4, 2004) [1]. As the usage data does not span the entire 6-month period, we do not present the data instances in absolute terms (e.g. 10 AM August 12), but instead use relative terms (e.g. instance 2783). Due to the time the data was collected, for the remainder of this paper we discuss the Abilene Network as it existed in 2004, during which time OSPF was used to deliver traffic between every origin and destination pair, and no congestion control protocols were implemented.

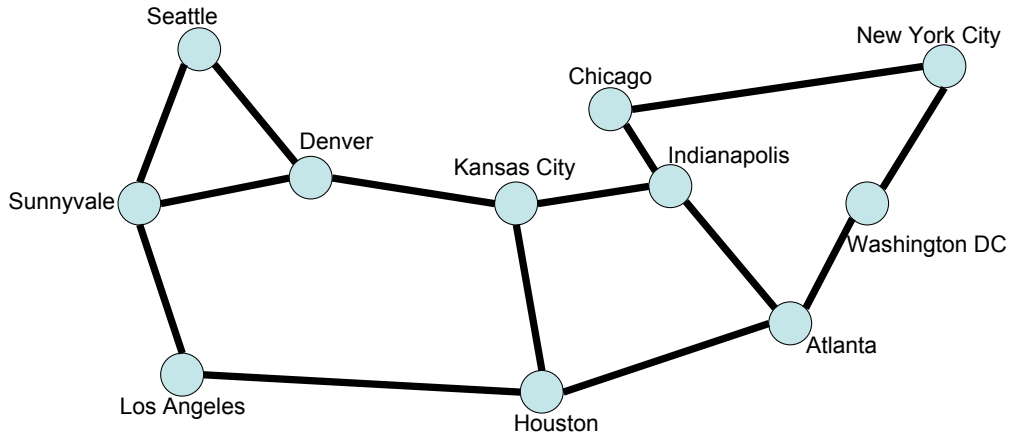


Figure 2:
The Abilene Network

5 Numerical Experiments

In the remainder of the paper we will present results of our numerical studies. In this section, we first describe the specifics of how we applied our model to the Abilene network, including a discussion of the objective function used, and present the first set of numerical results.

The Abilene network usage data, provided at [1], consists of the demands between every origin and

destination pair in the 11-node network over five minute intervals. It should be pointed out that the data collected consist of arc flows during the specified period of time (*e.g.*, 5 minutes). This arc flow data is then converted into estimates of demand between every OD pair with a method such as the one discussed by Roughan *et al.* [23]. The method used is not the topic of this paper, but it is important to note that the demand data provided at [1] is only an estimate of the true demand.

For our numerical experiments we encoded the model (2) – (7) in AMPL [15], and used the data captured from the Abilene Network to define a family of data files, one for each time interval (we aggregated the data into hour-long intervals). We used SNOPT [18], a nonlinear programming solver, to solve the resulting problem instances. In preliminary experiments with MCFPNG models, we found SNOPT to perform better than several other popular nonlinear solvers, possibly due to the high level of nonlinearity of the constraints. SNOPT uses a sequential quadratic programming algorithm to find locally optimal solutions, and has been successfully used in solving nonlinear mathematical programs [13, 20]. We approximated the (non-differentiable) RED function outlined in Section 1.3 by

$$f(t) = \frac{1}{t+1}. \tag{16}$$

Note that this gain function implies that every arc in the network has effective capacity of 1. Correspondingly, we scaled the demand data provided at [1] by a constant factor, in part so that no commodity had a demand greater than 1. (The scaling factor is intended to calibrate the demands in the network to be consistent with arcs having capacity of 1 unit, while maintaining the *relative* demand levels between the OD pairs, and to be sufficiently high to justify deployment of congestion control. Thus, the value of the scaling factor was arrived at, to some extent, by trial and error. It is worthwhile pointing out that, after scaling, the overall level of demand in the network, as compared to the capacity, is quite high. In particular, under routing policies considered below, only a small fraction of some of the commodities reaches the destination.)

5.1 Role of Objective Function

In (2) – (7) we presented the model with an objective function that maximizes the weighted total of commodities received at all the destinations:

$$\max_{\alpha,\, x} \sum_{k=1}^{K} c^k x^k_{d^k}. \tag{17}$$

Alternatively, we may choose to maximize the weighted total *fractions* of commodities received at all of the destinations:

$$\max_{\alpha,\, x} \sum_{k=1}^{K} c^k \frac{x^k_{d^k}}{s^k}. \tag{18}$$

This objective function is appropriate in models motivated by the use of congestion control in UDP networks, common in VoIP and IPTV applications, in which the fraction of commodity delivered reflects the quality of service for that commodity. Moreover, using objective (18) provides an incentive for commodities with smaller demands to be routed to their destinations.

Unfortunately, optimizing with respect to either one of these objective functions may still lead to starvation, *i.e.*, a situation in which some commodities are ignored and not routed to their destination because of their lower relative values. This can be circumvented, for instance, by adding constraints assuring a minimum performance guarantee for all of the commodities. (For

example, constraints can stipulate that at least $z\%$ of every commodity must the received at the destination node.)

However, in a network with congestion control, imposing a minimum performance guarantee could lead to an infeasible problem instance. An alternative approach to avoiding starvation is to utilize a "max-min" objective function. For example, objective function (17) can be modified as:

$$\max_{\alpha, x} \min_{k \in \{1,\ldots,K\}} c^k x^k_{d^k}. \tag{19}$$

Similarly, we can define the max-min modification of objective function (18). The issue with either of these formulations is that the resulting routing policies may be hindered by a commodity that simply has very little demand to begin with. An optimal routing policy will optimize for the commodity with lowest demand and may actually perform much worse, by any other metric, than one obtained with the cumulative objective function approach.

In the numerical experiments discussed in this paper, we used objective function (18), as our work is motivated by the use of congestion control in UDP networks. However, the above considerations should be taken into account, and alternative objective functions should be considered, when applying similar optimization models to determine routing policies in networks with specific cost and quality of service considerations.

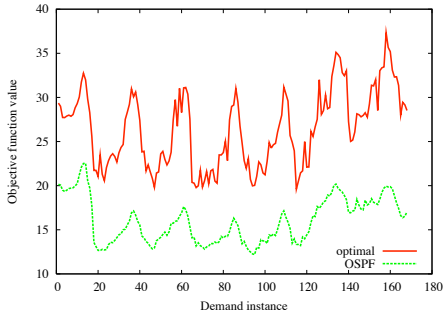
5.2 Numerical Results

In the first set of numerical experiments, we considered the first week of usage data (aggregated into hour-long increments). For each of the resulting 168 data instances, we compared the performance of the current routing policy, OSPF, and the MPLS routing policy optimized for that data instance, when each is subject to congestion control. Recall that performance comparisons were done with respect to the objective function in the form (18); we varied the weight coefficients c^k to arrive at three different sets of problem instances.

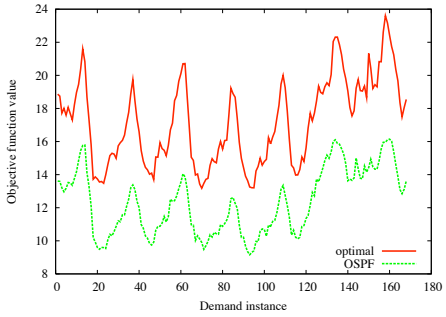
It should be pointed out that, since the feasible region of MCFPNG is not convex, there might be multiple local, but not global, solutions to (2) – (7). Thus, the routing policy found by the solver may not be truly optimal for the problem. Moreover, which (local) optimum is found by the solver is dependent on the initial routing policy used as a starting point of the optimization algorithm. We chose to initialize the solver with the robust routing policy, discussed in section 6, when finding the proposed MPLS routing policy. The reasons for this choice will become clear in the following section. In reporting our computational results, we nonetheless refer to the solutions found by the solver as optimal, for simplicity of presentation.

In Figures 3, we compare the performance of the OSPF and MPLS routing policies with three different objective functions. The objective functions differ by having different values of commodity weights, c^k. As we do not have precise information regarding the values of the commodities in the network we consider three different choices of the weights, to see if there is a performance change amongst the resulting problem instances. The key takeaway from Figures 3 is that, regardless of the commodity weights used the MPLS routing policy, optimized for every demand instance, performs better than the OSPF routing policy. Each objective function is the result of different valuations of commodities in the network; to provide a framework for comparison between results reported in Figures 3(a), 3(b) and 3(c), we note that node 1 (Sunnyvale) provides roughly 4% of the overall demand in the network, while over 23% of the overall demand originates in node 10 (Washington,

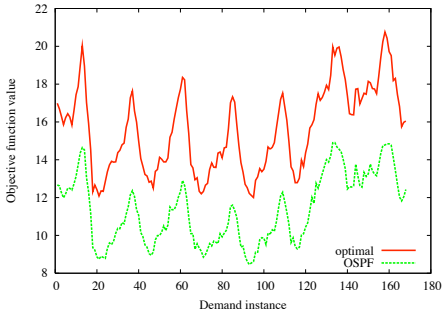
DC). Note that the MPLS routing policy, when optimized for each demand instance, performs much better than the OSPF routing policy in every instance and for every objective function used. As shown in Figure 3(d), the improvement over the OSPF routing policy is at least 27% when the objective function values all commodities equally; similar improvements are obtained for other objective functions as well. To summarize, a routing policy that (i) takes into account the actual demand, and (ii) allows routing of each commodity on multiple paths performs much better than a policy, such as OSPF, that does not.



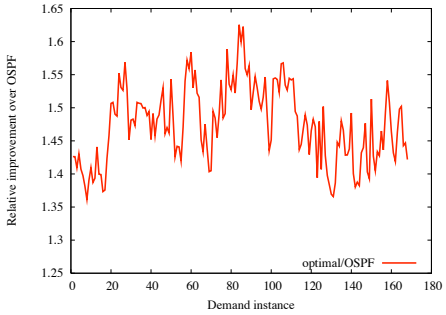
(a) Commodities from node 1 are 5 times more valuable than all other commodities.



(b) Commodities from node 10 are 5 times more valuable than all other commodities.



(c) All commodities are of equal value.



(d) All commodities are of equal value; fraction improvement.

Figure 3: Optimal MPLS routing policy vs. OSPF routing policy

6 Robust MPLS Routing Policies

In this section we present a robust reformulation of (2) – (7) and study empirical performance of the resulting routing policy.

In section 5.2 we compared the performance of the OSFP routing policy to the MPLS routing policy that is optimized for every demand instance. As one would expect, the latter outperformed the former in every instance. In practice, however, demand in a network fluctuates continuously and is not known in advance; thus, re-optimizing the routing policy for every short time period would not be feasible. Therefore, in this section we will use ideas of robust optimization to propose a robust

counterpart of MCFPNG and find one MPLS routing policy which performs well for a variety of demand instances observed in the network.

6.1 Robust Reformulation

Modern robust optimization was simultaneously introduced by Ben-Tal and Nemirovski [29] and El Ghaoui *et al.* [17]. Robust optimization is a mathematical programming modeling technique used when the problem data is not known exactly, but instead it is known (or assumed) that any data instance from an *uncertainty set* can be the problem data. The objective is to find a solution that is optimal among the solutions feasible under *all* possible data realizations. Using this approach Ben-Tal and Nermirovski [28] and El-Ghaoui and Lebret [16] pose and solve problems in robust truss topology design and robust least-square optimization, respectively. Recently robust optimization has attracted a lot of attention and has been considered for portfolio selection problems [19], integer programming and network flow problems [7], supply chain management [8], inventory theory [9], radiation treatment planning [11], *etc.*

Robust optimization and related approaches have also been applied to computer routing and network flow problems [4–6, 10, 25]. For example, Applegate and Cohn [4] look at minimizing the maximum link utilization over a set of feasible demand realization in a network using MPLS. Chekuri [10] provides a survey paper of the work to date on using robust optimization to create routing policies. Chekuri characterizes the work of Applegate and Cohn as working on *oblivious routing*, meaning that they design a routing policy that is used for a set of possible demand realizations without knowing the exact realization. The objective of an oblivious routing policy, as described by Chekuri, is to minimize congestion. This is a valid objective function to consider for a Virtual Private Network (VPN) setting, where there is a strict limit on the amount of flow that can pass through any one point in the network. The main distinguishing factor among works on oblivious routing is the structure of the uncertainty set, *i.e.*, the set of possible demand realizations, considered. For example, [4] considers a discrete set of demand realizations, while [6] is the first to consider a polyhedral set of demand realizations.

As far as we know, none of the current work in oblivious routing takes into account active congestion control. Though minimizing congestion seems like a good way to accomplish a level of congestion control, it may not capture the tradeoffs that need to take place between different commodities in the network. Moreover, since active congestion control is accomplished, in part, through packet loss, it is not clear whether the routing policies obtained from models that explicitly take active congestion control into account will be the same as those obtained by simply minimizing congestion without modeling congestion control.

A robust counterpart of the problem (2) – (7) takes into account multiple demand instances ($m \in \{1, \ldots, M\}$), and finds the routing policy maximizing the minimum performance over all of these

demand instances, mathematically written as:

$$
\begin{array}{lll}
\text{(RMCFPNG)} \quad \max\limits_{\alpha,\,x} \min\limits_{m} & \displaystyle\sum_{k=1}^{K} c^k x^k_{d^k,m} & \\
\text{s.t.} & x^k_{o^k,m} = s^k_m & k = 1,\ldots,K,\ m = 1,\ldots,M \\
& \displaystyle\sum_{(i,j)\in A} \alpha^k_{ij} = 1 & k = 1,\ldots,K,\ i \in N : i \neq d^k \\
& \displaystyle\sum_{(i,j)\in A} \alpha^k_{ij} = 0 & k = 1,\ldots,K,\ i = d^k \\
& \displaystyle\sum_{i\in\delta^-(j)} \alpha^k_{ij} x^k_{i,m} f_{ij} \left(\sum_{l=1}^{K} \alpha^l_{ij} x^l_{i,m} \right) - x^k_{j,m} = 0, & \\
& \qquad\qquad k = 1,\ldots,K,\ j \in N : j \neq d^k,\ m = 1,\ldots,M & \\
& \alpha^k_{ij} \geq 0 & (i,j) \in A,\ k = 1,\ldots,K.
\end{array}
\tag{20}
$$

Here, s^k_m is the demand for commodity k in data instance m, and $x^k_{i,m}$ is amount of flow of commodity k present at node $i \in N$ in instance m (note that the values of α's remain the same for all data instances, and thus specify a routing *policy*).

6.2 Numerical Results

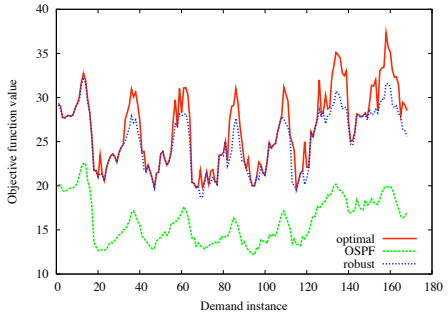
6.2.1 Robust MPLS Routing policy: First Week

To formulate the robust counterpart of MCFPNG as given by (20), we need to specify the uncertainty set, *i.e.*, the collection of demand instances to be considered. In considering the data for the Abilene network, we observed that the demand fluctuated following, to a large extent, a daily pattern. Roughly speaking, demand during the day was significantly higher than during early morning and night hours, which is to be expected. (This pattern was less pronounced during the weekends, but was still present.) Based on this observation, we constructed an uncertainty set with three demand instances as follows. We considered the demand data for the first week of usage, separated each of the 7 days into three eight-hour intervals (morning, day, and night), and averaged the demand over these eight-hour intervals across the week. Thus, the three demand instances included in the uncertainty set reflect average hourly demand during mornings, days and nights during the first week of usage.

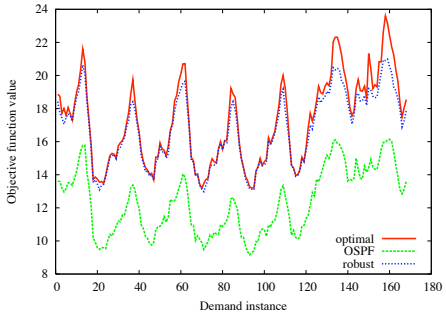
The resulting robust problem (20) has $M = 3$, and can be solved using MINOS. We then compared the performance of the resulting routing policy on the 168 demand instances considered in section 5.2 (recall that each of these instances corresponds to the demand during an hour-long interval during the first week). The results are summarized in Figure 4. In each plot of this figure, the robust MPLS routing policy is the (possibly local) solution of the single problem (20), while OSPF and optimal MPLS policies are the same as were found in section 5.2. In particular, each optimal MPLS routing policy has been optimized for the demand instance at hand, and thus changes every hour.

As we mentioned in section 5.2, each optimal MPLS policy was obtained by initializing the solver at the robust MPLS policy found by solving (20), thus assessing the improvement to the robust policy that can be made by modifying it to suit a particular demand instance. As demonstrated in Figure 4, it appears that fine-tuning the policy to a specific demand instance makes for very

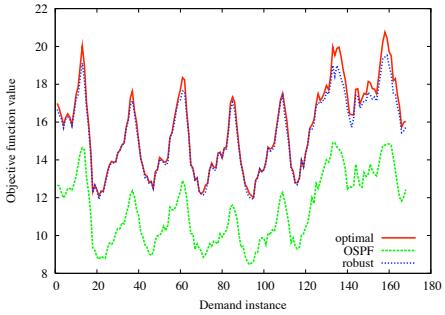
little improvement over the robust routing policy. In particular, Figure 4(d) compares, for each data instance, the improvement over the OSPF routing policy realized by the robust MPLS policy with the improvement over OSPF realized by the MPLS policies optimal for each instance. As expected, the improvement achieved by the robust policy is never better, but the plots are fairly similar: the robust policy improves performance by at least 27% in each instance, compared to the improvement of at least 36% realized by the optimal policies.



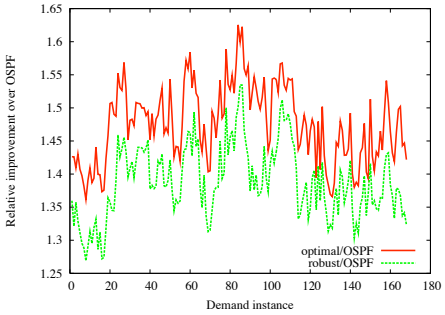
(a) Commodities from node 1 are 5 times more valuable than all other commodities.



(b) Commodities from node 10 are 5 times more valuable than all other commodities.



(c) All commodities equal value.



(d) All commodities equal value, fraction improvement.

Figure 4: Robust MPLS routing policy vs. optimal MPLS and OSPF routing policies

To summarize, simply by taking into account the natural demand fluctuations in a network, we propose a single, robust, MPLS routing policy which much better performance than OSPF, and comparable performance with each of the optimal MPLS policies.

6.2.2 Robust MPLS Routing policy: 24 Weeks

In section 6.2.1 we proposed a robust MPLS routing policy, obtained by considering an uncertainty set considering of three data instances capturing morning, day and night demand patters. Recall that we constructed these three instances by averaging demands in the corresponding 8-hour periods during the first of the 24 weeks of data available to us. Since at first glance the demand follows a weekly, as well as daily, pattern, a natural next step is to assess how the robust policy of section 6.2.1 would perform in the following weeks. This is the subject of this section. Here, we limit our

discussion to the case where all commodities are of equal value, but the results discussed carry over to the other cases as well. At this point it is important to note that 4 days of the 24 weekly datasets obtained overlap due to the way the data was partitioned at the time it was collected. This means that for the plots in Figure 5, 96 data instances (2.4% of all the instances) appear twice every time a robust routing policy is evaluated.

The plot in Figure 5(a) presents the performance of the robust routing policy obtained in section 6.2.1 (based on week 1 data) relative to the performance of the OSPF routing policy for each hour-long period in the 24-week period. This is done, as before, by plotting the ratio of the objective function values of the robust policy and the OSPF policy for each demand instance. If the ratio is greater than one, the robust policy performs better than OSFP for that demand instance, as is the case for 86% of the instances.

A closer study of Figure 5(a) reveals that the robust policy based on week 1 data performs better than the OSPF routing policy on all instances except for weeks 3, 4 and 5, which correspond to the period from March 15 until April 5 of 2004. Examining the data, we noted that these three weeks not only have the greatest demand over all twenty-four weeks, but also have somewhat different demand patterns than week 1. For example, 35% of the total demand in week 4 originated from a node that provided only 5% of the total demand in week 1. As the robust policy is catered to the week 1 demand patterns, it is no surprise that it did not perform well when the demand pattern was significantly different. (Unfortunately, we were unable to identify a cause or an explanation of such uncharacteristic demand patterns during this time period.)

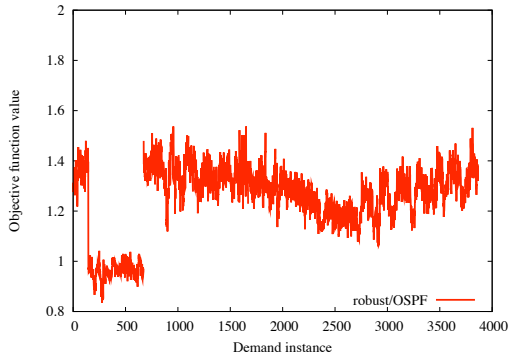
Based on the above observations, we computed a different robust routing policy, basing it on week 4 data, and plotted its performance (relative to that of the OSPF policy) in Figure 5(b). Notice that this robust policy performs well (i.e., better than OSPF) *only* during weeks 3, 4 and 5, further confirming that this behavior is due to the fact the demand pattern differs between these weeks and the rest of the time period considered.

We still hypothesized that, aside from anomalous behavior in weeks 3, 4 and 5, overall demand remains stable from week to week due to the self-similar nature of demand patterns [12, 21]. Indeed, the first robust policy continued to perform well in week 6 and beyond. We also looked at the performance of the robust routing policy based on week 12 data in Figure 5(c). This policy performs better than the other two routing policies, and seems to perform at least as well as the OSFP routing policy on most of the data instances. There is still, however, a drop in performance in weeks 3, 4 and 5.

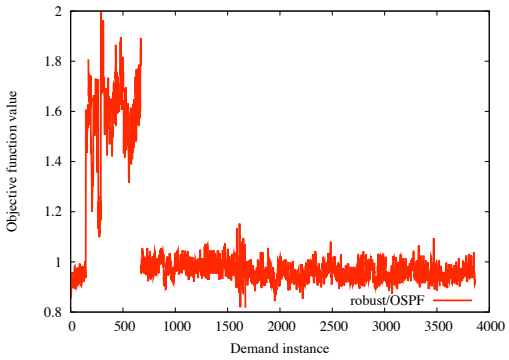
Finally, Figure 5(d) shows that the objective function values of the robust routing policy based on week 12 data and the OSPF policy over time follow the same trend, suggesting that, if the ratio of the objectives is greater than one, it is due to the improvement achieved by the robust policy over OSPF, not deterioration of OSPF performance (the same is observed with the other robust routing policies). Figure 5(d) also suggests that a sharp reduction in the performance of a currently implemented robust policy could serve as an indication that the demand pattern is changing and the robust routing policy needs to be reevaluated using more current demand data.

7 Conclusion and Future Work

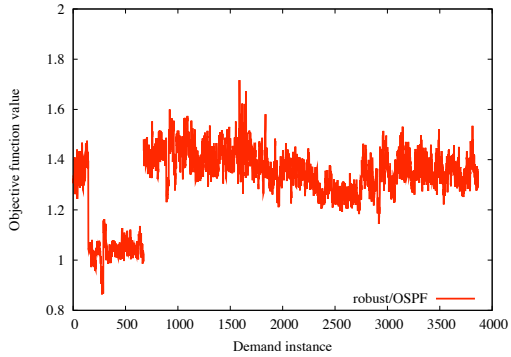
We introduced a way to model congestion control in networks facing congestion via the multicommodity network flow problem with nonlinear gains (MCFPNG), and proved that the resulting



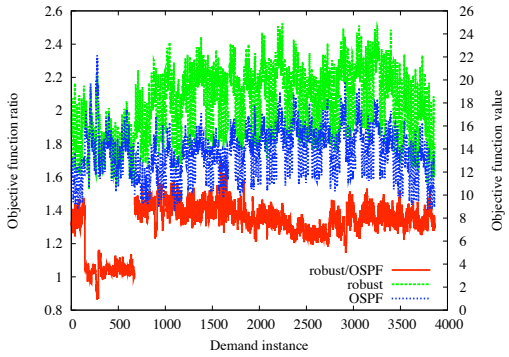
(a) Relative performance of the robust routing policy based on week 1 data and the OSFP policy



(b) Relative performance of the robust routing policy based on week 4 data and the OSFP policy



(c) Relative performance of the robust routing policy based on week 12 data and the OSFP policy



(d) Left axis: Relative performance of the robust routing policy based on week 12 data and the OSFP policy; Right axis: Objective function values of the two policies

Figure 5: Robust routing policy based on data from weeks 1, 4 and 12 vs. OSPF routing policy

model is $\mathcal{NP}$-hard. We applied the model to the Abilene network and showed that a better routing policy can be developed by taking into account demand fluctuations — either by designing optimal routing policies for the demand at hand, or, when the above is not possible or desirable, designing robust routing policies.

As presented, our results only give relative performance guarantees by showing empirically that the robust routing polices relatively outperform the OSPF routing policy currently used in the Abilene network. In the future we would like to give absolute performance guarantees on any routing policy we generate by using an approximation algorithm to solve our model.

As a further avenue of research, we would like to consider applications of our model in other areas of network routing. For instance, a modified version our model could be used to find the minimum power levels for wireless routers so as to meet a predetermined performance guarantee. For example, we would like the examine a problem such as: what is the minimum power need in a wireless network such that no more than 5% of packets in the network are lost? We believe that we can extend the model we presented to find a near optimal solution to these types of problems.

Acknowledgements

We would like to thank Z. Morley Mao, Xu Chen, and Yin Zhang for their helpful discussions and feedback. Stanko Dimitrov was supported by NSF IGERT grant 0114368 and 2009 Spring/Summer Research Grants Program at the University of Michigan.

References

[1] Abilene network demand data `http://www.cs.utexas.edu/~yzhang/research/abilenetm/`.

[2] OSPF Version 2 `http://tools.ietf.org/html/rfc2328`.

[3] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, February 1993.

[4] David Applegate and Edith Cohen. Making intra-domain routing robust to changing and uncertain traffic demands: understanding fundamental tradeoffs. In *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 313–324, New York, NY, USA, 2003. ACM.

[5] Yossi Azar, Edith Cohen, Amos Fiat, Haim Kaplan, and Harald Racke. Optimal oblivious routing in polynomial time. In *STOC '03: Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 383–388, New York, NY, USA, 2003. ACM Press.

[6] Walid Ben-Ameur and Hervé Kerivin. Routing of uncertain traffic demands. *Optimization and Engineering*, 6(3):283–313, 2005.

[7] D. Bertsimas and M. Sim. Robust discrete optimization and network flows. *Math. Program.*, 98(1-3, Ser. B):49–71, 2003.

[8] D. Bertsimas and A. Thiele. A robust optimization approach to supply chain management. In *Integer programming and combinatorial optimization*, volume 3064 of *Lecture Notes in Comput. Sci.*, pages 86–100. Springer, Berlin, 2004.

[9] D. Bertsimas and A. Thiele. A robust optimization approach to inventory theory. *Oper. Res.*, 54(1):150–168, 2006.

[10] Chandra Chekuri. Routing and network design with robustness to changing or uncertain traffic demands. *SIGACT News*, 38(3):106–129, 2007.

[11] M. Chu, Y. Zinchenko, S. Henderson, and M. Sharpe. Robust optimization for intensity modulated radiation therapy treatment planning under uncertainty. *Physics in Medicine and Biology*, 50(23):5463–5477, 2005.

[12] Mark E. Crovella and Azer Bestavros. Self-similarity in world wide web traffic: evidence and possible causes. *IEEE/ACM Trans. Netw.*, 5(6):835–846, December 1997.

[13] A. Eriksson. Optimization in target movement simulations. *Computer Methods in Applied Mechanics and Engineering*, 197(49-50):4207–4215, September 2008.

[14] Sally Floyd and Van Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Trans. Netw.*, 1(4):397–413, 1993.

[15] Robert Fourer, David M. Gay, and Brian W. Kernighan. *AMPL: A Modeling Language for Mathematical Programming*. Duxbury Press, November 2002.

[16] El L. Ghaoui and H. Lebret. Robust solutions to least-squares problems with uncertain data. *SIAM J. Matrix Anal. Appl.*, 18(4):1035–1064, 1997.

[17] El L. Ghaoui, F. Oustry, and H. Lebret. Robust solutions to uncertain semidefinite programs. *SIAM J. Optim.*, 9(1), 1999.

[18] Philip E. Gill, Walter Murray, and Michael A. Saunders. *User's Guide for SNOPT Version 7: Software for Large-Scale Nonlinear Programming*, June 2008.

[19] D. Goldfarb and G. Iyengar. Robust portfolio selection problems. *Math. Oper. Res.*, 28(1):1–38, 2003.

[20] Takao Hagishita and Makoto Ohsaki. Topology mining for optimization of framed structures. *Journal of Advanced Mechanical Design, Systems, and Manufacturing*, 2(3):417–428, 2008.

[21] Will E. Leland, Murad S. Taqqu, Walter Willinger, and Daniel V. Wilson. On the self-similar nature of ethernet traffic (extended version). *IEEE/ACM Trans. Netw.*, 2(1):1–15, February 1994.

[22] Multiprotocol Label Switching Architecture `http://www.ietf.org/rfc/rfc3031.txt`.

[23] Matthew Roughan, Mikkel Thorup, and Yin Zhang. Traffic engineering with estimated traffic matrices. In *IMC '03: Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*, pages 248–258, New York, NY, USA, 2003. ACM.

[24] Yosef Sheffi. *Urban Transportation Networks: Equilibrium Analysis with Mathematical Programming Methods*. Prentice-Hall, Englewood Cliffs, NJ, 1985.

[25] Rui Z. Shen and Nick Mckeown. Designing a predictable internet backbone network. In *HotNets-III Proceedings*, 2004.

[26] Maiko Shigeno. Maximum network flows with concave gains. *Math. Program.*, 107(3):439–459, July 2006.

[27] Shekhar Srivastava, Gaurav Agrawal, Michal Pioro, and Deep Medhi. Determining link weight system under various objectives for OSPF networks using a Lagrangian relaxation-based approach. *IEEE Transactions on Network and Service Management*, 2(1):9–18, 2005.

[28] Ben A. Tal and A. Nemirovski. Robust truss topology design via semidefinite programming. *SIAM J. Optim.*, 7(4):991–1016, 1997.

[29] Ben A. Tal and A. Nemirovski. Robust convex optimization. *Math. Oper. Res.*, 23(4):769–805, 1998.