A Novel Approach to Bottleneck Analysis in Networks

Nikhil Gopinath Shetty

Electrical Engineering and Computer Sciences
University of California at Berkeley

September 19, 2008

A Novel Approach to Bottleneck Analysis in Networks

Nikhil Shetty

Department of Electrical Engineering and Computer Sciences,
University of California Berkeley, California - 94720
Email: {nikhils@eecs.berkeley.edu}

Abstract— In this paper, we devise a novel method for bottleneck analysis of UDP networks based on the concept of network utility maximization. This method does not rely on time-consuming packet-level simulations, but is instead based on robust mathematical models. For fluid flows, one could solve a fixed point to determine the losses on links and extend this to random rates using a Monte Carlo simulation. However, lack of knowledge of convergence makes it difficult to predict the end of the simulation. Our method is more advantageous as it involves solving an optimization problem, the solution to which can be numerically determined to the desired accuracy. Also, compared to a black and white approach between worst-case analysis and average-case analysis, our method offers network managers the flexibility of choosing the shades of gray in between. To determine the losses on the links in a UDP network, we propose a geometric program formulation for which we find and prove conditions under which it accurately determines the true losses. We further extend this analysis to stochastic rates using stochastic optimization techniques, and show how the desired flexibility is obtained.

I. Introduction

Networks are built to satisfy certain goals and it is the aim of network design to ensure that those goals are satisfied suitably. Metrics should be defined so that they suitably capture the satisfaction of those goals and can also be applied easily to network design. For example, if a military network has to satisfy a goal, say, the quick download of satellite images, then in this case the metrics would be the latency of download and this puts certain restriction on the network path between the satellite and the base. To satisfy the latency requirements, given the size of the file and the amount of traffic sharing the same links, one would like to know what the bottleneck links in the network are and what is the investment required in them to provide the desired quality of service. Three factors control this decision process - the strictness of the requirement on the latency, statistical variation of cross-traffic and the congestion behavior of the protocols used. In this project, we make an attempt at defining a probabilistic metric for the satisfiability of such requirements and try to find methods by which networks could be designed based on this metric. To simplify this task, we assume that the protocols involved are not based on control loops but are instead open-loop. And that the statistical variation of traffic (i.e. number of users and size of their files) was well-known or determined in advance. We argue that a network with strong congestion analysis techniques can "satisfy" users even in the absence of feedback-based control by the end users. Also, in such a network, the collection of traffic statistics also is simplified and is assumed in our analysis.

Our goal in this paper is to find a method to identify bottleneck links in a network essentially composed with UDP traffic. The general approach to congestion analysis in present networks involves ceaseless monitoring of the network and heavy packet-level simulations. Network administrators keep track of the network utilization and availability by keeping a constant watch on factors such as round-trip times, jitter, latency and packet drop rates. These not only help to ensure that the networks satisfy the SLAs signed with their customers but also help them to gauge the performance and plan for future capacity augmentation. They make predictions of traffic generated by the customers by analyzing past data and use these to generate computer simulations using well-known network simulation tools to determine the bottleneck links in the network. In the short term, the traffic is rerouted among the links to improve the performance by eliminating the bottlenecks. In the long term, improvement in the network bottlenecks involves assignments of capacity or addition of new links, based on experience, followed by re-simulation of the scenarios to ascertain the "goodness" of the network to the changed conditions.

However, these network simulations are long and sensitive to the parameters of the distributions used to create the random nature of the traffic. Further, to ensure smooth running of the network, network administrators invariably go in for conservative scenarios and designs, resulting in an under-utilization of network resources. To prevent this, we need a metric for the probabilistic satisfaction of capacity requirements, which the managers may use according to their level of conservativeness. In this paper, we try to address this problem by restricting ourselves to UDP networks.

Though more than 90% of the traffic in the Internet consists of TCP [1], there are many motivational reasons for considering UDP networks. Firstly, real-time bandwidth-intensive applications such as video conferencing react very poorly to TCP's congestion control, making the use of UDP a lot more suitable for such applications. With an expected rise of the use of these applications, we expect the Internet to shift heavily to UDP-based transport (with heavy traffic policing and shaping). Secondly, though UDP does not provide reliable data transfer like TCP, multimedia applications such as IPTV, real-time

video conferencing, VoIP, audio and TV/video streaming can tolerate a small fraction of packet loss and reliable data transfer is not critically required. Further, if reliability is essential, it can be achieved over UDP by building it into the application itself. However, running multimedia applications over UDP is controversial because of the lack of decentralized congestion control. If everyone were to stream high bit-rate video without congestion control, the Internet would probably collapse [2]. Thus, for such a UDP network to be potentially realizable, we need strong congestion analysis methods.

In this paper, we attempt, for a network essentially composed with UDP traffic, to devise a nontraditional method for bottleneck analysis, utilizing only solutions of optimization problems. We achieve this by formulating the link loss allocation problem as a utility maximization problem. For a network without influence loops, we prove the existence of utility functions that accurately model the link losses. Further, we extend this analysis to stochastic rates and provide a metric to analyze bottlenecks in the network. This metric is derived from the probabilistic satisfaction of link capacity constraints and is an indicator of the link losses.

Average link losses can be determined using a Monte Carlo simulation of the fluid traffic model. However, this method suffers from convergence issues that make it difficult to predict the end of the simulation. Our method overcomes this limitation as it is the result of an optimization problem and the solution can be determined to the required accuracy. Further, we know exactly what the CPU requirements for such accuracies are and we can use the appropriate tradeoffs. Also, our method offers the flexibility of choosing any operational point between this average-case analysis and the worst-case analysis obtained using maximum values of the rates.

The paper is organized as follows. In section II, we explain the basic model for the network. Then, in section III, we show how the rate allocation problem can be solved using fixed-point method and prove that, if the network is *influence-loop-free*, the solution of such a fixed-point turns out to be unique. A brief review of the utility maximization techniques in literature is presented in section IV. We also provide the reasons for opting to use utility maximization to solve the allocation problem and argue for their validity. In V, we formulate the problem as a geometric program (GP) and we study, in section VI, the conditions under which the GP gives the same result as the fixed-point method. The extension of the result to random rates is presented in section VII. Finally, we conclude the paper with the major results explained in section IX.

II. Modeling

Consider the network to be a directed graph where the vertices represent the possible points of entry of data traffic. Let the directed edges in the graph represent the links in the network with the direction being the direction of flow of traffic on them. Most links in wired networks are full-duplex links. Hence, such a link can be modeled as two simplex links in opposite directions and can be incorporated in the

model. Half-duplex links are a rarity in the Internet and we will not consider them here. This process of *simplexification* may divide the network up into two unconnected parts. In that case, the following analysis will be true for each of the parts.

Henceforth, throughout this paper, we define a network as the triplet $(\mathcal{L}, \bar{C}, A)$ where $\mathcal{L} = \{l_1, \ldots, l_L\}$ defines a set of simplex links, where link l_i has capacity C_i and L is the total number of links. We summarize the link capacities $C_1, C_2, \ldots, C_L$ in the vector $\bar{C}$. We assume that the link capacities are constant and non-changing in the following analysis. This is generally true of wired links, which are most widely used in networks today[*]. We define a route to be the set of links traversed by a flow from the vertex it enters to the vertex it leaves. We will use the words flow and route interchangeably to mean the same thing. Let R be the number of such routes and $r_1, r_2, \ldots, r_R$ be the rates entering on the routes. In general, the rates entering on the different routes may be assumed to be independent. This is true, except in cases where there is, say, a live webcast of a popular event, in which case, the traffic gets correlated. The matrix $A = [a_{ij}]_{L \times R}$ is the link-route matrix with the element $a_{ij} = m$ if the link i is the m'th link on route j, counting from its ingress, and 0 otherwise.

The incoming traffic on the routes causes losses on the links where the sum of the rates of the flows entering that link is greater than the capacity of the link. We assume that losses are proportional, i.e., all the flows entering the link suffer the same fractional loss in packets. This is a good assumption if the flows are not differentiated and loss occurs with equal likelihood for all the flows. Let the fraction lost at link i be l_i and let the corresponding fraction going through be k_i. Thus, we assume that the network has no storage, which is generally true for the network at steady state. Also, we assume in this case that there is no rerouting of traffic in case of congestion. This assumption is good when routing is pinned down using a least-hop metric or using label switching.

The loss model presented above seems very simplistic even for a UDP network because we know that sources can reduce their rates based on the packet losses that they observe. For example, a video stream could adopt a lower coding rate and hence a lower quality video to account for packet losses. However, we do not consider this feedback from the network here and instead argue that, though real applications may implement this feedback, it is in the interest of the network to allocate the required rates to the users with high probability.

III. Fixed-Point to solve the allocation problem

The rate offered by a flow on a link is modulated by the losses suffered by it on the previous links on its route. If r_j^i is the rate offered by route j on link i, it is given by

$$r_j^i = r_j(1 - l_{(1)_j})(1 - l_{(2)_j}) \ldots (1 - l_{(m-1)_j}), \quad (1)$$

where $(1)_j, (2)_j, \ldots, (m-1)_j$ are the $m-1$ links on route j with $a_{ij} = m$, i.e., $(x)_j = y$ iff $a_{yj} = x$.

With the proportional loss model, the losses on the links will be given by

$$l_i = max(0, \frac{\sum_{i \in j} r_j^i - C_i}{\sum_{i \in j} r_j^i}). \tag{2}$$

The final rate (end-to-end) seen on a route j, r_j^f, is given by $r_j^f = r_j \prod_{j \in i}(1 - l_i)$. Thus, the fraction of loss seen $t_j = \frac{r_j - r_j^f}{r_j} = 1 - \prod_{j \in i}(1 - l_i) = 1 - \prod_{j \in i} k_i$. Given rates and capacities, we obtain a system of L equations for the links with L unknown variables l_i. Such a system can be solved using a Fixed-Point (FP) method. Furthermore, we have that:

Theorem 3.1: For an *influence-loop-free* network, there exists a unique fixed-point solution (thus a unique network allocation solution).

Definition Two routes i and j in a network can be related in the following ways

- *Direct Influence:* They share links in *order*. Shared links are said to be in order if after ordering all the links for each route from source to destination, the sequence of appearance of the shared links is the same for both routes. This influence is defined to be on the first shared link (l_{min}) and is represented as $i \overset{l_{min}}{\Longleftrightarrow} j$.
- *Influence loop:* They share links but these links are not in order.
- *Indirect Influence:* There exists a chain of routes $r_0 = i, r_1, r_2, \ldots, r_n = j$ for $n \geq 2$ such that $r_0 \overset{l_1}{\Longleftrightarrow} r_1 \overset{l_2}{\Longleftrightarrow} r_2 \overset{l_2}{\Longleftrightarrow} \ldots r_{n-1} \overset{l_n}{\Longleftrightarrow} r_n$ and $l_{i+1} \neq l_i$, $1 \leq i \leq n - 1$.
- *No Influence:* They neither share links nor have indirect influence on each other.

An *influence-loop* exists whenever a pair of routes forms an influence loop or when a route *indirectly influences* itself.

Fig. 1(b) and Fig. 1(a) show networks with and without influence loops respectively.

Note also that, because we simplexify duplex links, two routes traveling in opposite directions may not directly influence each other.

Proof: We prove the theorem by constructing the unique fixed-point solution. For that, we adopt the following *message passing* procedure in which the links are playing the double role of computing their allocation and sending information to links downstream via the route. The procedure works as follows

1) A given link waits for a message from its precedent link in each of its incoming routes. The messages received by link j are the offered rates r_j^i where $i = 1, \ldots, R_j$ indexes the incoming flows. If the link is the first link of the route, it receives the ingress rate r_j.
2) Upon receiving a message from all incoming routes, link j computes its allocation $k_j = \min(1, C_j / \sum_{i=1}^{R_j} r_j^i)$, and sends the message $k_j r_j^i$ to the next link in the outgoing route i.
3) The procedure stops when all links have computed their allocations.

The above procedure solves the fixed-point problem, it terminates and outputs a unique solution.

Let us first show that the procedure will terminate i.e. it will neither *deadlock* nor *run to infinity*. This is equivalent to showing that there is a finite number of steps and at each step, at least one message is sent by some link. To show this for the first step, we claim that

Claim 3.2: There exists at least one link ready to send a message i.e. that is not waiting for a message from any other link.

We prove this claim by contradiction. A link is waiting only if it is preceded by another link in at least one route. Now suppose that all the links are waiting. Then, pick a link say l_{o_1}; it is the j_{o_1}'th ($j_{o_1} > 1$) link of some flow r_{o_1}. Let link l_{o_2} be the first link of that flow. l_{o_2} is also the j_{o_2}'th link of some route r_{o_2} with first link l_{o_3}. If $l_{o_3} \in \{l_{o_1}, l_{o_2}\}$, then we have formed an influence-loop, hence, contradicting the assumption that the network is influence-loop-free. Otherwise, we continue by finding l_{o_4}, l_{o_5} and so on. Since the number of links is finite, we will eventually have a link l_{o_n} such that $l_{o_n} \in \{l_{o_1}, \ldots, l_{o_{n-1}}\}$, giving a contradiction.

Hence, in the first step, there is at least one link that computes its allocation and sends its messages. After sending the messages, such a link does not participate in the procedure anymore. In fact, it is not waiting for a message and no other link is expecting a message from it. So, it can be removed. Now, the claim is still valid for the rest of the network (removing links does not introduce traffic loops). Since the number of links is finite, by iteratively removing links in this manner, the procedure will eventually terminate. At termination, each link has computed exactly one allocation which corresponds to a unique solution for the fixed-point. ∎

Showing the existence and uniqueness of a fixed-point solution in network with loops is beyond the scope of this paper.

Theorem 3.1 guarantees existence and uniqueness of an allocation solution for influence-loop-free network. However, using fixed-point, this solution can only be determined numerically: there is no closed-form expression. Without a closed-form expression, however, we cannot extend our analysis to random rates.

IV. UTILITY MAXIMIZATION APPROACH

In the previous section, we have shown that for an influence-loop-free network, the solution of the network allocation problem can be determined using a fixed-point method. However, fixed-point determines a numerical solution and does not give a closed-form expression. Without a closed-form expression, extending this analysis to random rates is difficult. Hence, we attempt to model the problem as a utility maximization that can easily incorporate random rates.

In recent years, the framework of network utility maximization, that was proposed in the seminal paper by [3], has been used to calculate the rates allocated by TCP and TCP-like elastic flows. In the paper, the authors show that for a TCP network, depending on the fairness policy and the

(a) Influence-Loop-Free Network   (b) Example of network with influence-loop
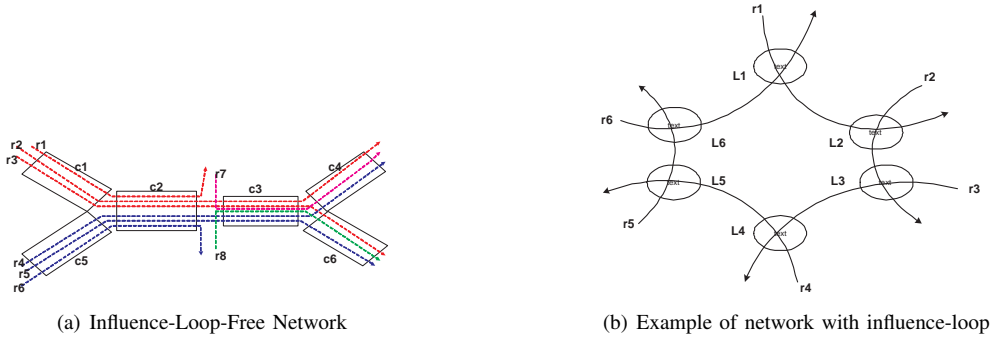
Fig. 1.   Example Networks

type of application, the rate allocation problem is equivalent to a utility maximization problem where the reward function equals $\sum U_j(r_j)$ where the sum is over the utilities of all the flows. Recently, there have been attempts to design protocols to extend this to inelastic flows [4] [5].

In this paper, we will distinguish between flow (or user) utility and network utility. The flow utility is the actual utility received by a flow for the rate allocated to it. Applications like email, file download, etc. are associated with concave utility functions. On the other hand, for applications like voice, video, etc., the utility derived from the rate may be S-shaped as proposed in [5]. Network utility, on the other hand, is the utility that the network assigns to the flow for the purpose of rate allocation. Thus, when the network tries to maximize the total utility generated by these assigned utilities, given the topological constraints, we end up obtaining the rates allocated for each flow in the real network. This has an analogy to the utility maximization used for TCP. We can consider, for example, a video application that runs on TCP. This application may be allocated a rate by the network according to a concave TCP utility function. But, the end user may not have the same utility function and hence does not derive the same utility from the allocated rate.

For proportional fairness, which TCP-like algorithms closely emulate, the log function is a good approximation for the utility assigned by the network [6]. In case of UDP flows, we will show, in the following sections, that the network utility depends directly upon the losses suffered on its links and hence indirectly upon the losses suffered by (and hence the effective rates allocated to) the individual routes.

V. Geometric Programming Modeling

In the following sections, we will show how, by using the Cobb-Douglas function [7] of the link gains as the utility function, we can model the network allocation problem as a geometric programming problem.

A. Simple utility maximization example

We will first show for a single link that, if the allocation among flows entering that link is proportionally fair, it is the same as the maximization of an aggregate log utility function. Suppose there are n flows offering traffic $r_1, r_2, \ldots, r_n$ to a link with capacity C. We know that the rate allocated to flow i in case of proportional losses is given by $min(r_i, Cr_i/(\sum_i r_i))$. Suppose the utility for flow i is given by $r_i \log(x_i)$ where x_i is the amount of rate allocated on the link, then consider the utility maximization problem,

$$Maximize \quad \sum_{i=1}^n r_i \log(x_i) \quad\quad (3)$$
$$s.t. \quad \sum_i^n x_i \le C, \; x_i \le r_i \; \forall i$$

Writing the Lagrangian and solving the unconstrained problem using complementary slackness conditions, we find that the allocation obtained is the same as that from proportional fairness. This shows that the proportional fair allocation can be reinterpreted as a utility maximization problem.

Now, using the fact that the $x_i = k \times r_i$, where k is the fraction of traffic that is let through on the link, we can rewrite the utility function as $\sum_{i=1}^n r_i \log(kr_i)$ in terms of k. Expanding this gives us $\sum_{i=1}^n r_i \log(kr_i) = (\sum_{i=1}^n r_i) \times \log(k) + \sum_{i=1}^n r_i \log(r_i)$. The second term is independent of k, while the first term is a constant multiple of $\log(k)$ and hence we get a maximization problem over k, given by

$$Maximize \quad \alpha \log(k) \quad\quad (4)$$
$$s.t. \quad k(\sum_{i=1}^n r_i) \le C \quad 0 \le k \le 1$$

where $\alpha = \sum_{i=1}^n r_i$ Thus, we see that the utility that is maximized for a single link can be expressed as a logarithm of the ratio of traffic allowed through.

B. Extension to a bigger network

As shown in the previous section, the utility function for a single link is the logarithmic utility ($\log(k_l)$). We may simply extend this to a bigger network by taking a linear combination of these utilities. Each of the utilities $\log(k_l)$ will be weighted by a factor α_l. This leads to a network utility function given by $\sum_{l=1}^L \alpha_l \log(k_l)$. The maximum amount of traffic flowing through each link is constrained to the capacity of the link. Put together, this gives us the following utility maximization problem.

$$Maximize \quad \sum_{l=1}^L \alpha_l \log(k_l)$$
$$s.t. \quad \sum_{j \in l} k_l \, r_j^l \le C_l \quad \forall \, l \in 1, 2, \ldots, L$$
$$k_l \le 1, \; k_l > 0$$

Fig. 2. Simple network 2-link topology

Fig. 3. Simplified version of the typical relationship between two links in series

where $r_j^l = r_j k_{(1)_j} k_{(2)_j} \ldots k_{(m-1)_j}$ is as defined in (1) with i replaced by l.

However, in (4), we see that α depends on the rates that enter the link. These rates themselves are modulated by the losses on the previous links, which we intend to find using the optimization. Hence, we have a *chicken-and-egg* problem here. Surprisingly however, we can find a set of α, not depending on the incoming rates, with which the utility maximization gives the solution to the network allocation problem. We will prove the existence of such a set of α in the following sections.

By rewriting the network utility function using the properties of logarithms, we get the objective function to be $\prod_{l=1}^{L} k_l^{\alpha_l}$, which is defined as a *monomial* for a geometric programming problem [8]. Also, interestingly, this utility function is the well-known Cobb-Douglas utility function in economics. Further, we note that the constraints resemble the definition of *posynomials* as mentioned in section 2.1 of [8]. Thus, this new network utility maximization fits the general framework for a geometric programming problem and can now be written as

$$Maximize \qquad \prod_{l=1}^{L} k_l^{\alpha_l} \qquad (5)$$
$$s.t. \qquad \sum_{j \in l} k_l \, r_j^l \le C_l \quad \forall \, l \in 1, 2, \ldots, L \quad (6)$$
$$k_l \le 1, \; k_l > 0$$

This optimization problem can be solved by well-known methods available in most optimization software packages.

VI. ANALYSIS

In this section, we would like to answer the question: " *Does there exist a set of α such that the GP formulation in (5) gives the same solution as the fixed point equations in (1) and (2)?*". In what follows, we prove the following theorem

Theorem 6.1: For all influence-loop-free networks, $\exists \, \bar{\alpha} = \{\alpha_1, \alpha_2, \ldots, \alpha_L\} \in \mathcal{R}^L$ such that the optimization problem in (5) yields the same solution as the solution of the fixed point equations defined in (1) and (2), for all values of the rates. We also give a method by which the set of α can be determined. Let's consider a motivating example for the above theorem.

A. A simple example

The optimization problem in (7) gives the GP formulation of the allocation problem for the network in Fig. 2, and Fig. 3(a) shows the graphical representation of the maximization. In this figure, we have plotted, for several values of U, $g(k_1) = \frac{U}{k_1^{\alpha_1/\alpha_2}}$, representing the contour curves of the utility. $f(k_1) = \min\left(1, \frac{C_2}{k_1 r}\right)$ which corresponds to the (second) constraint's curve (as a function of k_1) is also shown in the figure. The
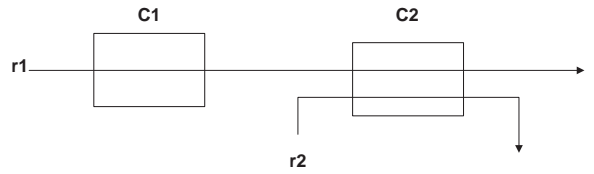
first constraint $k_1 r_1 \le C_1$ corresponds to the vertical line at $k_1^* = \min(1, \frac{C_1}{r_1})$.

$$Maximize \quad U(k_1, k_2) = k_1^{\alpha_1} k_2^{\alpha_2} \qquad (7)$$
$$s.t \qquad k_1 r_1 \le C_1, \qquad 0 \le k_1 \le 1$$
$$k_2 k_1 r_1 \le C_2, \qquad 0 \le k_2 \le 1$$

The network allocation solution is the upper right corner (circled) of Fig. 3(a). In fact, for a given value of r_1, the first link L_1 fixes the fraction of traffic going through to k_1^*. For all values of k_1, the fraction of traffic supported by L_2 is $f(k_1)$, and for the particular value of k_1^*, we obtain the point $(k_1^*, f(k_1^*))$ circled in the figure. Table I shows the GP

TABLE I
GP SOLUTIONS FOR DIFFERENT VALUES OF α_1 AND α_2 FOR
$C_1 = 1, C_2 = 0.5, r_1 = 2$ GIVING $(k_1^*, k_2^*) = (0.5, 0.5)$

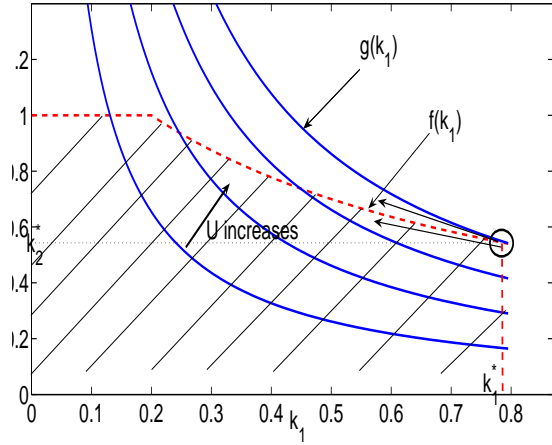| (α_1, α_2)	(0.6,1)	(0.8,1)	(1,1)	(1.2,1)	(1.4,1)
GP Solution	(0.25,1)	(0.25,1)	(0.35,0.71)	(0.5,0.5)	(0.5,0.5)
(α_1, α_2)	(1.6,2)	(1.8,2)	(2,2)	(2.2,2)	(2.4,2)
GP Solution	(0.25,1)	(0.25,1)	(0.35,0.72)	(0.5,0.5)	(0.5,0.5)
(α_1, α_2)	(8,10)	(9,10)	(10,10)	(11,10)	(12,10)
GP Solution	(0.25,1)	(0.25,1)	(0.35,0.73)	(0.5,0.5)	(0.5,0.5)

solution for different values of α_1 and α_2. It can be seen that the GP solution differs from the FP solution for certain values of $\alpha_i, i = 1, 2$, implying that one should properly choose the weights α_i. By observing the geometrical representation of the GP, we see that α_1 and α_2 have to be chosen such that:
Matching Condition: *The contour curve for the utility passing through (k_1^*, k_2^*), where $k_2^* = f(k_1^*)$, should remain above the constraints 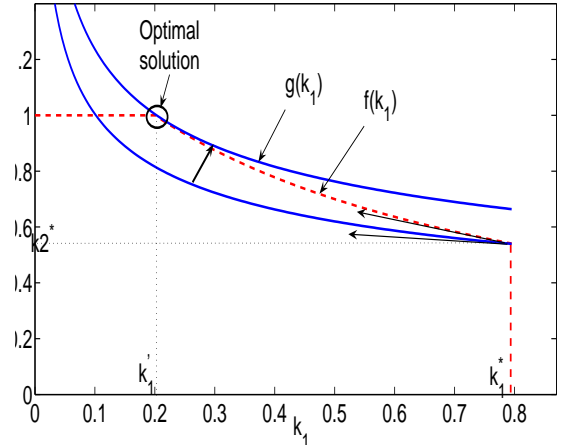in the entire constraint (feasible) region.* This condition is needed to guarantee that the maximum utility for the optimization is obtained at (k_1^*, k_2^*). In fact, if the condition is violated, as in Fig. 3(b) (dashed curve), a higher value of the utility is achieved at the point circled in the figure (the utility curve is translated to sit at the boundary of the constraint region). For this network, from Table I, we see that the weights apparently need to satisfy $\alpha_1 > \alpha_2$. (Note, however, that this is for a specific choice of link capacities and rates.) We will now try to derive such conditions for the general network.
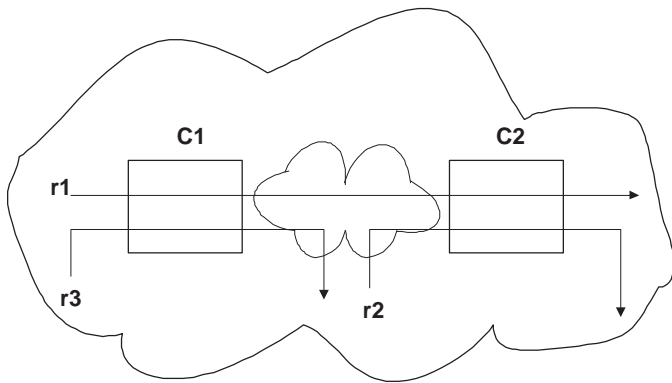
B. Introducing the typical topology

As mentioned in Theorem 6.1, we need to find the appropriate values of α for all the links. In this section, we will show that the problem of finding those α can be broken down into
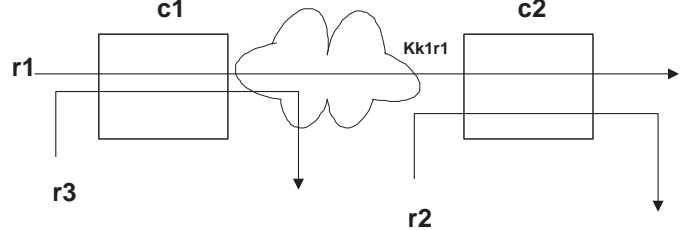
(a) Condition under which GP solution = FP solution

(b) Violating the Matching Condition

(c) Network topology: general relationship between two links in the network

(d) Network topology 1: typical relationship between two links in series

the problems of finding the α for all pairs of links. Any pair of links in an influence-loop-free network will have the topology shown in Fig. 3(d) that we will call the *typical topology*.

We obtain the typical topology by the following transformation. For all the links l_i in a network, we fix the values of k_i, except for two links (say l_1 and l_2). The rates entering these two links are well-defined due to the fixed k_i for the other links. For this two-link network, if we draw the constraint region, given the values of k_i for the other links, it will look similar to the region in Fig. 3(a) and the solution point for the fixed point will be at the intersection of the constraints. Geometrically, we can interpret this constraint graph as a slice of the constraint/feasible region that is parallel to the plane generated by the axes k_1 and k_2. Further, this solution point changes as you assign different values for the k_i for the other links, yielding a curve in L-dimensional space, which we call a *solution curve*. Now, if we consider all the slices going through the FP solution, we know that this solution will be a solution point in all such slices. Thus, the following claim is true.

Claim 6.2: The network fixed point solution will lie on ALL the solution curves formed by the solution points in every 2-dimensional slice (parallel to the plane of the axes) of the

feasible region and this intersection point is unique.

Proof: Suppose that the above is not true. Then, in some 2-dimensional slice (parallel to plane formed by k_i and k_j say) passing through the network fixed point $\bar{k}^*$, the solution point is not the same as $\bar{k}^*$. Hence, if we fix k_l^* for other links l, (k_i^*, k_j^*) will not be the solution point in that slice. But, this violates the fixed point equations in (2). Hence, the above point cannot be a fixed point which contradicts the assumption made. Further, if this point were not unique, then it would mean that there exist two points in the feasible set that satisfy all the fixed point equations, which contradicts Theorem 3.1. ■

The typical topology in Fig. 3(d) corresponds to two links in series with the network cloud representing the links in between. In general, these two links will share some common routes (the total rate is shown as r_1 in the figure) and each link might carry some independent traffic (r_2 and r_3). The topology summarizes several particular scenarios. The case where the two links have exactly the same set of routes (Fig. 2) is captured by setting r_2 and r_3 to zero; when the links do not share any common traffic, $r_1 = 0$.

C. α-Conditions

Claim 6.3: For a typical topology (see Fig. 3(d)) with given capacities, $\exists$ a pair (α_1, α_2) such that the optimization problem in (5) for the topology yields the solution point for *all values of rates* (r_1, r_2, r_3). In fact, $\frac{\alpha_1}{\alpha_2} > 1$ if $C_1 > C_2$ and $\frac{\alpha_1}{\alpha_2} > \frac{C_1}{C_2}$ if $C_1 < C_2$.

The derivation of this theorem is presented in section VIII. The theorem gives the α-conditions for a typical topology. The following claim says that such a set of α exists for the whole network.

Claim 6.4: For all influence-loop-free networks, $\exists \bar{\alpha}$ such that the pairwise conditions specified in Claim 6.3 are satisfied by all pairs in the set.

Proof: Consider the set $\{1, 2, \ldots, L\}$. We will choose the α for each link from this set in the following way. Using Claim 3.2, we know $\exists$ a link in the network that wishes to send the message first (pick one, say l_{o_1}). In other words, all other links are downstream to this link for each of the routes passing through it. Hence, in all typical topologies formed with this link, we can consider this link to be link 1 (Fig. 3(d)). Hence, if we wish to use the condition $\alpha_1 > \alpha_2$ derived from Claim 6.3, we assign the value $\alpha_{o_1} = L$. Now, we remove this link and we can see that the rest of the network will again have at least one link that wishes to send the message first. Hence, we assign the values of α_i in that order from L to 1. Can a link that was eliminated earlier (say l_x) be the downstream link to a link eliminated in a later step (say l_y)? This cannot be true, since then, at the step involving elimination of l_x, a route passing through l_y had l_x as its n'th link for $n > 1$. But this contradicts the fact that l_x wants to send its message first, i.e. all routes going through it have l_x as their first link. Hence, the ordering is correct and we have determined a set of α that satisfies the conditions of Claim 6.3. ∎

D. Proof of Theorem 6.1

In influence-loop-free networks, we have proven that the FP problem has a unique solution. The GP formulation also admits a unique solution. In fact, any geometric program can be reduced to a convex optimization problem using appropriate logarithmic transformations and change of variables [9] and, by definition, a convex optimization problem admits a unique solution.

Claim 6.5: For $\bar{\alpha}$ satisfying the conditions in Claim 6.4, the GP solution is equivalent to the FP solution in influence-loop-free networks for all values of the rates.

Proof: Assume that the α-conditions are satisfied for all (2-dimensional) slices of the GP problem but that the GP solution does not correspond to the FP solution. From Claim 6.2, the FP solution is unique and there is only one point through which all solution curves pass. Since the GP and FP solutions are different, in at least one 2D-slice (parallel to the axes) passing through this FP solution, the GP solution does not correspond to a point on the solution curve. But, since we satisfy the α-conditions, this contradicts the fact in Claim 6.3. Hence, the initial supposition is wrong. q.e.d. ∎

Thus, we have proved Theorem 6.1 by showing the existence of a set of α so that the GP solution is the same as the FP solution. In Claim 6.4, we have also devised a procedure to determine that set. Fig. 1(a) is an example of a influence-loop-free network. For this network, using the $\bar{\alpha}$ generated from the α-conditions, we compare the GP and FP solutions for the single lossy link case in Fig. 4(a) and for two lossy links in Fig. 4(b).

VII. EXTENSION TO RANDOM RATES

In previous sections, we have shown how, by using a geometric program formulation, one can derive the link losses for a given set of input rates. In this section, we extend the problem to the case where the input rates are random. Using the geometric program formulation, we propose a new metric for the loss on links, that models how the links perform with random rates.

For random rates, the losses in the links are also random and have complicated distributions, even for simple input rate distributions. It is not our intent to study these distributions, but to develop a metric that characterizes the magnitude of the losses on each of the links. The metric that we develop should also depend on the conservativeness of the network manager. Thus, if the metric for a given link is below a certain threshold, we may flag that link as a bottleneck link since the metric captures both the loss on the links and the conservativeness.
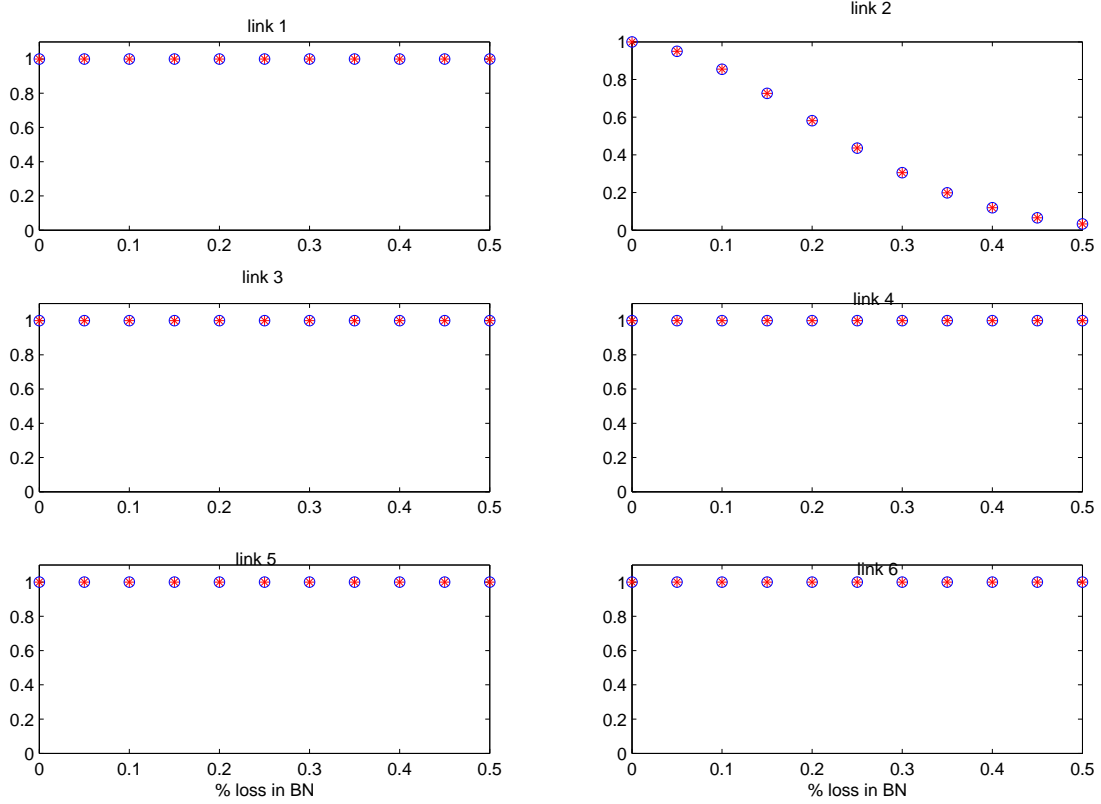
For e.g., if the network users are extremely sensitive to losses, then we may consider a worst-case scenario while choosing the capacity of links on the network. Thus, by replacing the rates in (6) by their highest possible values, we obtain the value of losses in the worst case. Since the rates are random, we may interpret this worst case as satisfying the constraints with a probability of unity. If we do not wish to be so conservative in managing the network we will instead allow the inequalities to be satisfied with a probability less than 1. Thus, letting the inequalities be satisfied with a probability greater than $1 - \epsilon$, we could rewrite (6) as the chance constraints

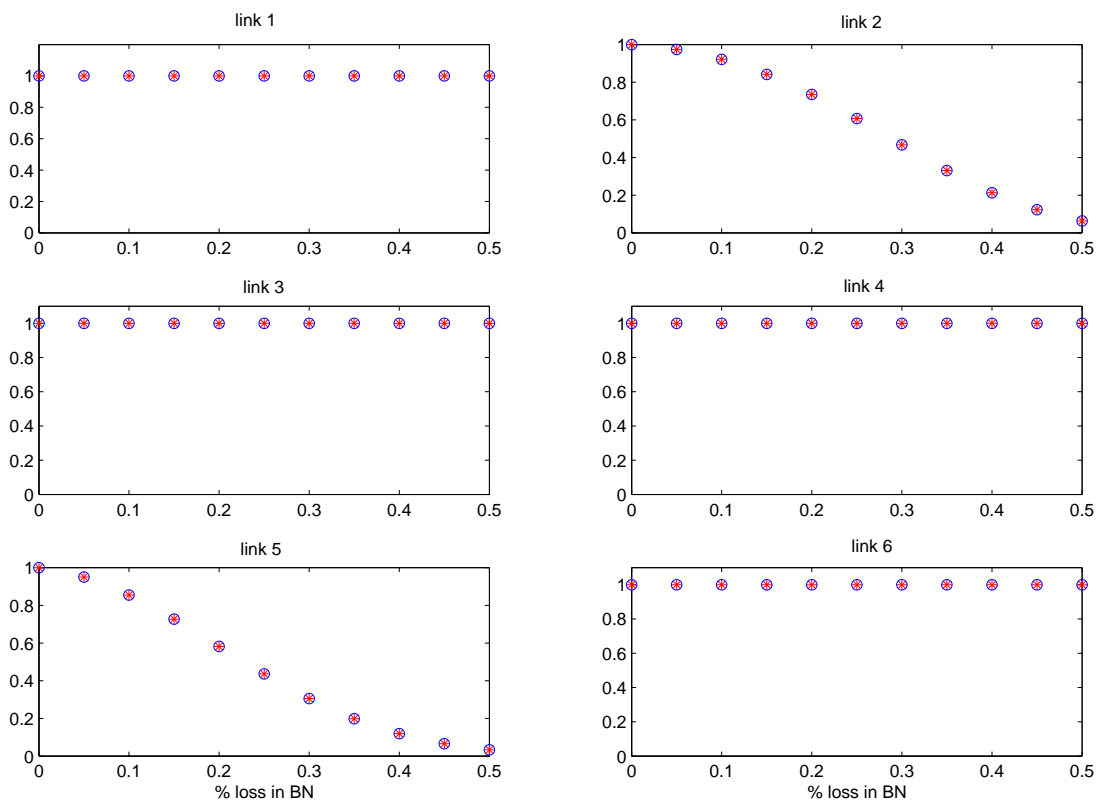$$Prob\{\sum_{j \in l} k_l\, r_j{}^l \leq C_l \;\forall\; l \in \{1, 2, \ldots, L\}\} \geq 1 - \epsilon \quad (8)$$

By solving the maximization problem with these new probabilistic constraints, we obtain $\bar{k}_i^\epsilon$, *which corresponds to the fraction of loss in link i for the given level of conservativeness* ϵ. We define this $\bar{k}_i^\epsilon$ to be our new bottleneck link metric.

Note that the ϵ is a choice that is left to the network manager. This gives the network manager a smooth handle over how conservative he wants his estimates to be, compared to the situation where he only has a binary choice of worst-case analysis and average rate analysis (Monte Carlo techniques). The case $\epsilon = 0$ corresponds to the worst-case analysis, as explained, while $\epsilon = .5$ loosely corresponds to the average case analysis.

The chance constraint using the joint distributions of the rates (8) is difficult to work with. However, one can approximate it using multiple chance constraints for the different links.

(a) Allocation when link 2 is lossy as a function of percentage capacity for no loss on the link

(b) Allocation when links 2 and 5 are lossy as a function of percentage capacity for no loss on the links

Fig. 4.   Network Allocation Results (Blue Circles - From optimization. Red pluses - From fixed point equations.)

Accordingly, we get the following optimization problem.

$$Maximize \prod_{l=1}^{L} k_l^{\alpha_l} \tag{9}$$

$$s.t. \quad Prob\{\sum_{j \in l} k_l^j \, r_j^l \leq C_l\} \geq 1 - \epsilon \tag{10}$$

$$k_l \leq 1, k_l > 0 \qquad \forall \, l \in \{1, 2, \ldots, L\}$$

If the distributions of the incoming rates are known, we can find the set of $\bar{k}$ that satisfies the probability constraints. In general, this region may not be convex. However, in the special case of Gaussian distributions, this region is convex resulting in a convex program. Gaussian distributions for the rates is a reasonable assumption, especially when the rates are aggregated from/to a large number of users. If r_j is given by $N(\mu_j, \sigma_j^2)$, where μ_j is the mean and σ_j^2 the variance of the Gaussian distribution, the constraints may be manipulated for each of the links l as follows.

$$Prob\{\sum_{j \in l} r_j k_l^j \leq C_i\} \geq 1 - \epsilon$$

$$Prob\{N(\sum_{j \in l} \mu_j k_l^j, \sum_{j \in l} \sigma_j^2 k_l^{j^2}) \leq C_i\} \geq 1 - \epsilon$$

$$Prob\{N(0, 1) \leq \frac{C_l - \sum_{j \in l} \mu_j k_l^j}{\sqrt{\sum_{j \in l} \sigma_j^2 k_l^{j^2}}}\} \geq 1 - \epsilon$$

$$\frac{C_i - \sum_{j \in l} \mu_j k_l^j}{\sqrt{\sum_{j \in l} \sigma_j^2 k_l^{j^2}}} \geq Q^{-1}(1 - \epsilon)$$

$$\sum_{j \in l} \mu_j k_l^j + \sqrt{\sum_{j \in i} \sigma_j^2 k_l^{j^2}} Q^{-1}(1 - \epsilon) \leq C_i \tag{11}$$

where $k_l^j = k_l k_{(1)_j} k_{(2)_j} \ldots k_{(m-1)_j}$, similar to the expression in 1. We may rewrite the constraint in (11) using $t_l \geq \sqrt{\sum_{j \in l} \sigma_j^2 k_l^{j^2}} Q^{-1}(1 - \epsilon)$ giving us two constraints per link l namely

$$\sum_{j \in l} k_l^j \mu_j + t_l \leq C_l \tag{12}$$

$$\frac{\sum_{j \in l} \sigma_j^2 k_l^{j^2} Q^{-1}(1 - \epsilon)}{t_l^2} \leq 1 \quad \forall \, l \in \{1, 2, \ldots, L\}$$

We observe that these new constraints themselves are posynomials. Thus, remarkably, we find that the stochastic optimization problem generated by replacing the constraints (10) in the optimization problem (9) by the constraints in (12) is also a geometric program (hence a convex program) and hence can be solved efficiently. In Fig. 5, we see how the results for the k generated by the stochastic optimization vary with choice of ϵ. As expected, the ϵ analysis results approach the average case results from a Monte Carlo simulation (the flat line) as ϵ tends to 0.5. For $\epsilon < 0.05$, the graphs drop rapidly to 0 (not shown), again as expected from the Gaussian distribution. However, as we can see, the simplifications done in (9) seem to have displaced the results a little bit. We believe that values of ϵ between 0 and 0.5 model the spectrum of possible conservativeness.
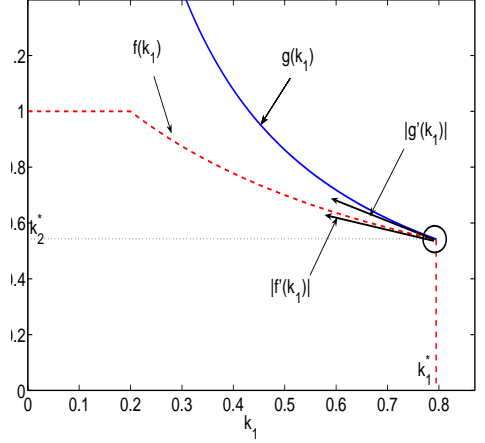

Fig. 6. Illustrating claim 8.1 for the simplified typical topology 3.

VIII. DERIVATION OF THE α-CONDITIONS

In this section, we present a derivation of the α-conditions given in 6.1. For the typical topology in Fig. 3(d), the **Matching condition** is equivalent to:

$$Maximize \quad U(k_1, k_2) = k_1^{\alpha_1} k_2^{\alpha_2} \tag{13}$$

$$s.t \quad k_1(Kr_1 + r_3) \leq C_1, \quad 0 \leq k_1 \leq 1$$

$$k_2(Kk_1 r_1 + r_2) \leq C_2, \quad 0 \leq k_2 \leq 1$$

where $0 \leq K \leq 1$ is the effect of the potential intermediary links on the traffic r_1. A geometrical interpretation of this maximization problem is shown in Fig. 3(a). From this figure, we see that the **Matching condition** is equivalent to:

$$Cond1: \quad g(k_1^*) = f(k_1^*) \tag{14}$$

$$Cond2: \quad g(k_1) > f(k_1) \quad \forall \quad 0 \leq k_1 < k_1^* \tag{15}$$

where $g(k_1, U) = \frac{U}{k_1^x}$ is the utility curve ($x = \frac{\alpha_1}{\alpha_2}$), and $f(k) = \min(1, \frac{C_2}{Kk_1 r_1 + r_2})$ is the constraint curve (k_2 as a function of k_1); $k_1^* = \min(1, C_1/(r_1 + r_3))$ is the value of k_1 for which the inequality in first constraint is tight. Note that the only effect of r_3 is to push k_1^* to the left. This does not change the nature of the conditions $Cond1$ and $Cond2$. Thus, we can further simplify the analysis by considering $r_3 = 0$. The equivalence in (14) is trivial. Relation 15 says that the utility curve must remain above the constraint curve. This is needed otherwise there will be more than one point of intersection between the curve $g(\cdot)$ and the (feasibility) utility region. Because of the structure of the utility and the constraint functions, we claim that:

Claim 8.1: The following conditions:

$$Cond1: \quad g(k_1^*) = f(k_1^*)$$

$$Cond2': \quad |g'(k_1^*)| > |f'(k_1^*)| \tag{16}$$

imply the Matching Condition (14 and 15).

Proof: These conditions are illustrated in Fig. 6. The equivalence in 14 is trivial. Relation 15 says that the utility curve must remain above the constraint curve. This is needed
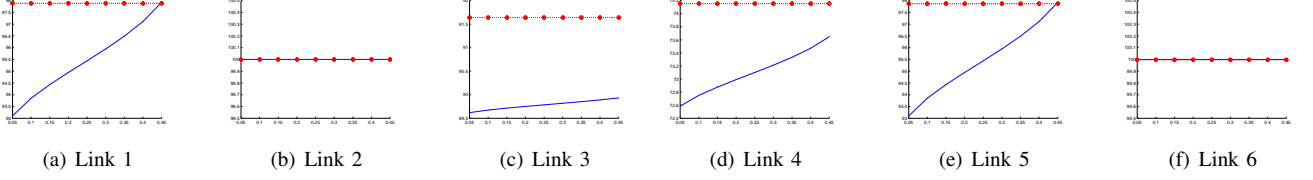
| (a) Link 1 | (b) Link 2 | (c) Link 3 | (d) Link 4 | (e) Link 5 | (f) Link 6 |

Fig. 5. Variation of k (percentage) with ϵ for Links of Network in Fig. 1(a) compared to the Monte Carlo solution. All rates have a mean value of 1 and standard deviation of 0.1, $C_1 = 3, C_2 = 6, C_3 = 5.4, C_4 = 2, C_5 = 3, C_6 = 3$

otherwise there will be more than one point of intersection between the curve $g(\cdot)$ and the (feasibility) utility region. From $Cond1$ and $Cond2'$ we obtain the relation

$$U = \frac{C_2(k_1^*)^x}{Kk_1^*r_1 + r_2} \quad \text{and} \quad x > \frac{r_1 k_1^*}{Kr_1 k_1^* + r_2}.$$

Thus, proving that condition 16 implies 15 is the same as showing that for x satisfying the inequality above it must be the case that:

$$g(k_1) > f(k_1), \forall k_1 < k_1^*, \text{ or } g(k_1) - f(k_1) > 0, \forall k_1 < k_1^*.$$

To show this, we distinguish two cases.

- If $f(k_1)$ is identically equal to 1, then there is nothing to prove. In fact $g(k_1^*) = 1$ and $g(\cdot)$ is a decreasing function for all positive x. So, $g(k_1) - 1 > 0, \forall k_1 \leq k_1^*$, independent of the value of x.
- If $f(k_1) = \frac{C_2}{Kk_1 r_1 + r_2}$ is not identically equal to 1, we should show that for $x > \frac{r_1 k_1^*}{Kr_1 k_1^* + r_2}$

$$\frac{C_2}{Kk_1^* r_1 + r_2}\left(\frac{k_1^*}{k_1}\right)^x > \min\left(1, \frac{C_2}{Kk_1 r_1 + r_2}\right),$$
$$\forall k_1 \in (0, k_1^*)$$

This condition turns out to be equivalent (for the given functions) to

$$\frac{C_2}{Kk_1^* r_1 + r_2}\left(\frac{k_1^*}{k_1}\right)^x > \frac{C_2}{Kk_1 r_1 + r_2}$$
$$\forall k_1 \in (\max(0, k_1'), k_1^*) \quad (17)$$

for $x > \frac{r_1 k_1^*}{Kr_1 k_1^* + r_2}$, where $k_1' = \frac{C_2 - r_2}{Kr_1}$ is obtained by setting $\frac{C_2}{Kk_1 r_1 + r_2}$ equal to 1 and solving for k_1. Equation 17 can be rewritten as (after rearranging and taking the log)

$$x > \frac{\log(\frac{Kk_1^* r_1 + r_2}{Kk_1 r_1 + r_2})}{\log(\frac{k_1^*}{k_1})}, \quad \forall k_1 \in (\max(0, k_1'), k_1^*).$$

Thus, since $x > \frac{r_1 k_1^*}{Kr_1 k_1^* + r_2}$, to show the condition above, it is sufficient to show that

$$\frac{r_1 k_1^*}{Kr_1 k_1^* + r_2} > \frac{\log(\frac{Kk_1^* r_1 + r_2}{Kk_1 r_1 + r_2})}{\log(\frac{k_1^*}{k_1})}, \quad \forall k_1 \in (\max(0, k_1'), k_1^*).$$

We need to find the maximum value of the RHS as a function of k_1. The function in the RHS is a monotonically increasing function of k_1 for $k_1 \in (\max(0, k_1'), k_1^*)$.

So let's compute its limit as $k_1 \to k_1^*$. The limit is of the kind $\frac{0}{0}$ but by applying l'Hospital's rule we get

$$\lim_{k_1 \to k_1^*} \frac{\log(\frac{Kk_1^* r_1 + r_2}{Kk_1 r_1 + r_2})}{\log(\frac{k_1^*}{k_1})} = \lim_{k_1 \to k_1^*} \frac{-\frac{Kr_1}{Kr_1 k_1 + r_2}}{-\frac{1}{k_1}}$$
$$= \frac{Kr_1 k_1^*}{Kr_1 k_1^* + r_2}$$

This implies that $\frac{Kr_1 k_1^*}{Kr_1 k_1^* + r_2}$ is a strict upper bound of the function $\beta(k_1) = \frac{\log(\frac{Kk_1^* r_1 + r_2}{Kk_1 r_1 + r_2})}{\log(\frac{k_1^*}{k_1})}$ for all $k_1 \in (\max(0, k_1'), k_1^*)$. Since $x > \frac{r_1 k_1^*}{Kr_1 k_1^* + r_2}$ and $K \leq 1$, it is also bigger than $\beta(k_1)$, which ends the proof of Claim 8.1.

$\blacksquare$

Note: The worst case above was achieved when $K = 1$. Thus, considering it equal to 1, we could have got rid of K in Fig. 3(d), leading to the simplified network in Fig. 3. We will use this network to find the rate-independent conditions on x in the following paragraph.

Now, we know that $x > \frac{r_1 k_1^*}{r_1 k_1^* + r_2}$ is a sufficient condition to get the fixed point solution in every 2-D slice. Hence, to find the value of x that works for all values of the rates, we have to find the upper bound of the expression in the RHS. Interpreting the RHS to be a function of $r_1 k_1^*$, we know that the RHS is maximized when $r_1 k_1^*$ is maximum. The maximum value of $r_1 k_1^*$ is restricted to the capacity C_1, since it is the fraction of traffic of r_1 let through by link 1. This gives us the condition $x > \frac{C_1}{C_1 + r_2}$. Now, we have to minimize the denominator. For the denominator, note that if $C_1 + r_2 < C_2$, $f(k_1)$ is identically equal to 1, and from case 1 of Claim 8.1, the required condition on the utility function is always true, independent of the value of x. If $C_1 + r_2 > C_2$, then there are two cases. If $C_1 > C_2$, the minimum in the denominator is obtained by putting $r_2 = 0$, giving us $x > \frac{C_1}{C_1} = 1$. If $C_1 < C_2$, the minimum in the denominator is obtained by putting $r_2 = C_2 - C_1$, giving us $x > \frac{C_1}{C_2}$. Thus, we obtain the required α-conditions in Claim 6.3.

IX. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed a novel way to analyze bottlenecks in networks using network utility maximization. We developed a geometric programming problem using the Cobb-Douglas utility function and showed that it actually models the

losses on the links accurately when the network is influence-loop-free. Also, we provided the much-needed flexibility in network analysis by extending the problem to stochastic rates and approximating the stochastic optimization with a convex program. This flexibility is not provided by a Monte Carlo simulation. Our method also removes the uncertainty about the duration of the simulation because the time complexity of the algorithm to achieve the desired accuracy is well-known and one can apply the appropriate tradeoffs. Thus, we have placed the congestion analysis of UDP networks on a firm mathematical footing, thus enabling us to envision a network with mostly UDP flows.

Note that we have proven the uniqueness of the solution to the fixed point equations only for influence-loop-free networks. We know of no results where the uniqueness of fixed point equations in general networks has been proven. In fact, even for a simple two-link network, we were unable to show a contraction mapping for the fixed point equations. We believe that most networks will settle to a unique operation point, though the dynamics followed to reach that point may be extremely slow, due to which the traffic might seem to oscillate as rates vary. Convex optimization problems give us a unique solution and hence, if multiple solutions were to exist in a network with influence loops, we would not know how to relate this unique solution to those operating points.

The aim of this paper is to bring the reader's attention to the fact that it might be possible to approximate the losses suffered in a UDP network using network utility maximization problems.

Just like the rate allocations for TCP can be approximated by assuming that certain utility functions are maximized, in the same way, the losses in the network can be assumed to be the result of maximization of utility functions. We acknowledge the importance of validating theoretical results against well designed real network scenarios. We hope, in the near future, to follow up with such validations.

REFERENCES

[1] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and C. Diot, "Packet-level traffic measurements from the sprint ip backbone," *IEEE Network*, vol. 17, pp. 6–16, 2003.
[2] K. W. Ross and J. F. Kurose. (2000) Connectionless transport: Udp. [Online]. Available: http://www-net.cs.umass.edu/kurose/transport/UDP.html
[3] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, "Rate control for communication networks: shadow prices, proportional fairness and stability," *Journal of the Operational Research Society*, vol. 49, pp. 237–252, 1998.
[4] M. Fazel and M. Chiang, "Nonconcave network utility maximization through sum of squares method," in *Proc. IEEE Control and Decision Conference*, Seville, Spain, Dec. 2005.
[5] P. Hande, S. Zhang, and M. Chiang, "Distributed rate allocation for inelastic flows," in *Proc. IEEE INFOCOM 2005*, vol. 4.
[6] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Transactions on Networking*, vol. 8, no. 5, p. 556567, Oct. 2000.
[7] Wikipedia. Cobb-Douglas. [Online]. Available: http://en.wikipedia.org/wiki/Cobb-Douglas
[8] S. Boyd, S.-J. Kim, L. Vandenberghe, and A. Hassibi, "A tutorial on geometric programming," *Optimization and Engineering*, vol. 8, no. 1, pp. 67–127, Mar. 2007.
[9] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004, ch. 4.