Network Utilization: the Flow View

Avinatan Hassidim*, Danny Raz*‡, Michal Segalov* Ariel Shaqed*
*Google, Inc. Israel R&D Center
‡Technion, Israel
{avinatan, razdan, msegalov, arielshaqed}@google.com

***Abstract*—**

Building and operating a large backbone network can take months or even years, and it requires a substantial investment. Therefore, there is an economical drive to increase the utilization of network resources (links, switches, etc.) in order to improve the cost efficiency of the network. At the same time, the utilization of network components has a direct impact on the performance of the network and its resilience to failure, and thus operational considerations are a critical aspect of the decision regarding the desired network load and utilization. However, the actual utilization of the network resources is not easy to predict or control. It depends on many parameters like the traffic demand and the routing scheme (or Traffic Engineering if deployed), and it varies over time and space. As a result it is very difficult to actually define real network utilization and to understand the reasons for this utilization.

In this paper we introduce a novel way to look at the network utilization. Unlike traditional approaches that consider the average link utilization, we take the flow perspective and consider the network utilization in terms of the growth potential of the flows in the network. After defining this new Flow Utilization, and discussing how it differs from common definitions of network utilization, we study ways to efficiently compute it over large networks. We then show, using real backbone data, that Flow Utilization is very useful in identifying network state and evaluating performance of TE algorithms.

I. Introduction

One of the worst kept secrets in the networking industry is that utilization of backbone links is very low. In a paper titled “Data Networks are Lightly Utilized, and Will Stay That Way” Andrew Odlyzko (then at AT&T Research–Labs) argued that this is a fundamental property of large scale networks and that this situation is unlikely to change (see [1]). In this paper we re-examine these observations and challenge that conclusion. Clearly, the economic pressure pushes operators to increase the utilization and thus increase the return on their investment. On the other hand higher utilization may have an operational impact on the service and too high utilization may cause higher levels of packet loss, or serious crisis in case of link failure.

The actual utilization of the network resources is not easy to measure, let alone predict or control. It changes quickly in time and in space, and it is hard to draw decisive conclusions by looking on summary statistics. However, hard as it may be, understanding the utilization pattern is a crucial first step when considering ways to optimize a network and is therefore important both from the operations point of view as well as for cost saving.

In this paper we explore the current state of the art in this area, try to identify the most important parameters and introduce a novel utilization metric that can be used to allow a better view of the network state.

Traditionally, the main parameter used to describe network utilization is link utilization. For each link, utilization is defined as the amount of traffic traversing it divided by the link capacity. Since modern networks consist of many links, the (weighted) average of the link utilization is used as a single number representing network utilization. Note that the time frame here is important since the average utilization of a link over short time periods is very noisy. For performance management usage it is common to average link utilization over five minute periods.

While link utilization is an important metric and indeed provides meaningful information, it is not always sufficient. First, examining some percentiles or max link utilization do not provide enough information. It is not clear how to evaluate a given maximal link utilization. What really matters is the link utilization distribution over all the network links and over time.

Second, link utilization does not necessarily reflect network performance, as it is possible and even common for the link utilization to be much lower than the actual traffic, due to built in redundancy (addressing possible failures) and in order to allow flows (demands) to grow. The additional traffic that the network can really accommodate depends on the specific TE used as well as on the utilization of links and on the demand pattern.

Finally, the link utilization does not tell the story from the client side, and does not describe how any specific user is experiencing the network.

In this paper we introduce a new view of network utilization – the *flow* view[1]. We define a new notion, *Generalized Flow Utilization* (GFU) which is a quantitative measure for the ability of the TE mechanism to support the current demand and possible growth (of the same set of demands). We compare the traditional link utilization data from one of the Google backbone networks to the flow utilization, given the TE scheme, and show that the new view indeed provides more insight regarding the full picture of network utilization than the traditional link utilization.

As indicated by its name, GFU is a general framework and can be used in different ways depending on the choice of a specific utilization function. We study theoretically the

[1] Since we mainly deal with backbone networks, we consider long lived aggregated flows.

properties of this view from the computational point of view and provide algorithms to compute it for a large set of interesting utilization functions and hardness results for other (also natural) utilization functions.

We then concentrate on two practical cases, one dealing with traffic risk assessment and one with longer term planning and the ability of the network to accommodate growth in flows (demands). We show how to compute flow utilization in these two cases, and demonstrate the applicability of using the scheme in realistic backbones using real production data.

The paper is structured as follows. In the next section we examine the traditional link utilization view, then in Section III we define the new flow view and study some of its theoretical properties. In Section IV we examine the growth motivated utilization and in Section V the risk motivated one. We provide related work in Section VII and conclude with a short discussion.

II. Link Utilization in Backbone Networks

Historically, it is well accepted that backbone networks are poorly utilized, due to the operational constraints and the need for stable fault resilient network solutions. However most of the information on this topic was kept confidential by the operators and not very many research papers addressed this important issue. One of the most notable early exceptions is the 1999 paper by Andrew Odlyzko (then at AT&T Research–Labs) titled: “Data Networks are Lightly Utilized, and Will Stay That Way” (see [1]).

Later on, in the early 2000s, a series of papers described backbone data from Sprint Networks (see [2], [3]). Again, the average utilization of the links is very low (around 10%) and links are reported to be utilized over 50% only when there are failures in the network. This common belief about operators maintaining low utilization is well described in the talk from NANOG 2002 (see: http://www.nanog.org/meetings/nanog26/presentations/telkamp.pdf), where the author explicitly talks about the backbone planning process and says “... upgrade (buy new capacity) at 40% or 50% utilization”, where the goal is to arrive at “maximum 75% utilization under (a single) failure”. Similar utilization numbers in backbone links during the years 2007–2008 are reported in http://arstechnica.com/uncategorized/2008/09/what-exaflood-net-backbone-shows-no-signs-of-osteoporosis/.

Very recently, it was reported that the emerging use of Software Designed Networks (SDN) (see [4]) on the Google backbone was motivated partially by the need to increase utilization from the aforementioned 40%–50% to close to 100% utilization (see http://www.networkworld.com/news/2012/060712-google-openflow-vahdat-259965.html).

Note that going in this direction (dramatically increasing the backbone utilization) requires a powerful network planning component, an improved monitoring ability and a deeper understanding of where the bottlenecks are.

During 2012, we collected data from a part of the Google backbone that was not controlled by SDN. The data was collected from backbone links over typical work days. This data includes a measurement every minute throughout the day for all relevant links, where for each link we collected the capacity and actual utilization.

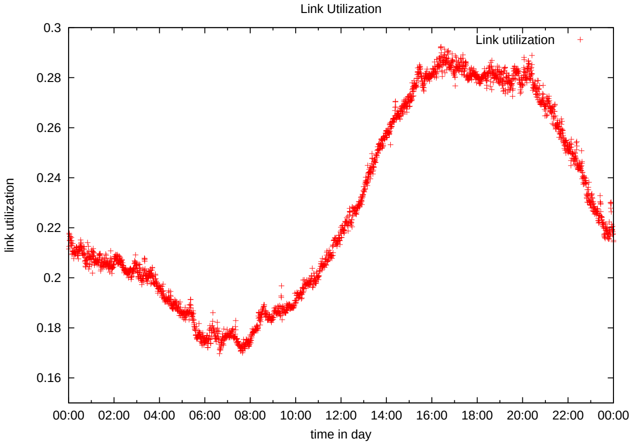

Fig. 1. Typical Backbone Link Utilization.

A large backbone network has many links, and can have even thousands of data points per day, depending on the time scale at which we sample each of these links. This amount of data can be overwhelming for humans to look at, hence the need to present a compact yet meaningful view of the data. A classical point of view is the *Link Utilization* view. Here, at each point in time we look at the (weighted) average utilization of all of the links. Per timestamp, we compute $\frac{\text{link flow}}{\text{link capacity}}$. We average this number over all the links per timestamp weighted by their capacity[2]. The results are presented in Figure 1. One can see that the average link utilization varies over the day between 17% and 29% with a typical daily pattern. The pattern is very smooth and similar when examining different days. This is expected, given the network size, as averaging the utilization of links smooths out the extreme cases. However, it is these extreme links, network operations care most about. On the other hand, not all links that are highly utilized or not utilized are always interesting. We would like to represent the ability of the network to scale, or ability to accommodate for more traffic somehow.

In terms of capacity planning, a link is useful if it was utilized sometime during the relevant period. To check this, we plotted in Figure 2 the cumulative peak daily utilization of the links in two separate dates. The distribution is similar in both days, about 6% of the links got to 100% utilization, but only about 30% of the links were ever utilized over 50% during that day.

This still reveals very little information about the actual utilization of specific links over time. In Figure 3 we present four typical graphs. We note that the type of traffic the link carries and the geographical location of the link have a great impact on this behavior. The real date was anonymized. Some links are bursty while others are smooth, some are highly

[2] This is actually the sum of link flows over the entire network divided by the sum of link capacities.

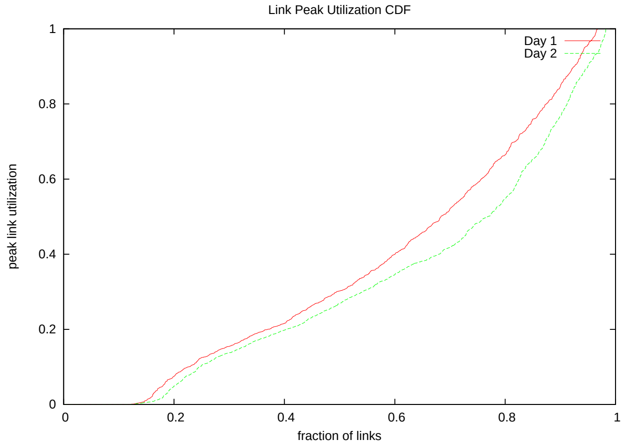

Fig. 2. Cumulative Peak Utilization.

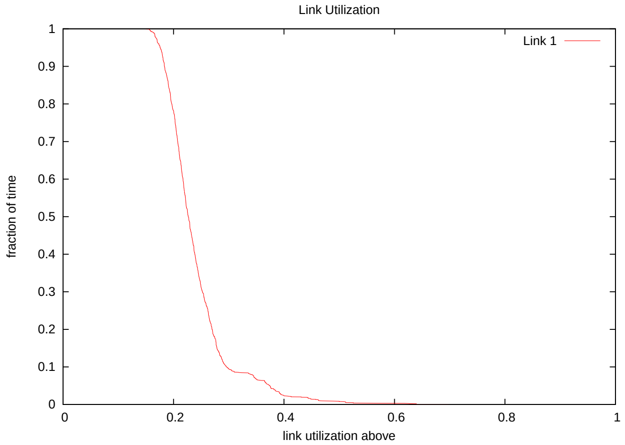

Fig. 4. Link Utilization Distribution of a specific link over one day.

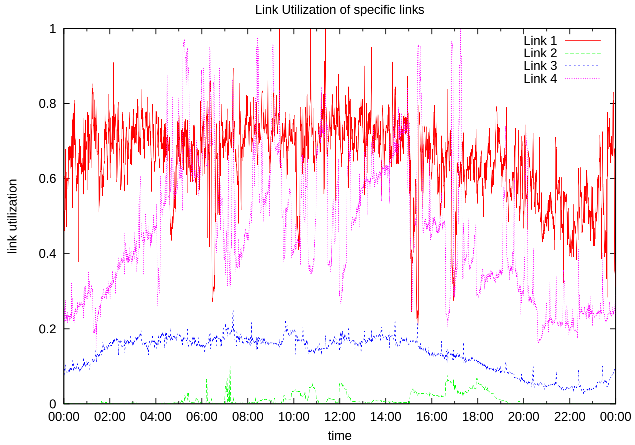

Fig. 3. Link Utilization of several links.

utilized while other are not. Also, the geographical location of links with similar behaviour can shift the time of their peak utilization. In many cases operators manually monitor important links. These could be links of critical impact on the service level, links that are prone to fail, or expensive links. For these links the exact behavior is studied and in many cases manually optimized. However, it is not scalable to do this for every link in the network and other methods are required in order to monitor the performance level of the entire network.

One way to do this is of course the average utilization number. A more detailed information is presented in the cumulative utilization graph corresponding to one link. Figure 4 depicts such a graph for a specific link out of the many links considered in Figure 1. The x-axis shows link utilization levels and the matching y-axis is the fraction of time the link was utilized above that level. For instance, the link we examined was over 20% utilized 80% of the time, and reached a peak utilization of almost 70% in a very small fraction of the time. One can see here what fraction of the time the link was highly utilized during this day, and from this get a better view of the load distribution in the network.

III. Flow Utilization

The utilization of backbone networks, as reflected from the available link utilization data described in the previous section, seems to be low. However, this view is based on link data and it represents the state from the infrastructure point of view. Another relevant aspect is the user (or customer) view which may be better described by the flow in the network. Thus, we suggest an alternative definition for the term *backbone utilization* based on looking at the problem from a flow perspective. In the context of backbone networks, the flows are long lived aggregated entities going from ingress nodes to egress nodes over the backbone. This approach can provide greater insight into the actual performance of the network since it treats the flows as the “basic entities” rather than the links, and uses metrics to measure flow utilization rather than link utilization. One obvious gain from the flow point of view is the ability to distinguish between different types of flow, and give the required weight to the more “important” flows. This is much more complicated to do in the link utilization view, since the same link with its entire capacity serves many flows.

The link utilization view summarizes the network state by a vector $u = (u_1, u_2, \ldots, u_n)$, where u_i is the utilization of link i, and n is the number of links in the network. The network utilization is the average of all elements u_i weighted by their capacity. We wish to develop a similar vector $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_m)$, where α_i represents the state of flow i, and m is the number of flows in the network.

There are various options or objectives when defining this vector and the specific choice of the exact value of α_i will represent a different aspect of the current network or the TE performance state. We want to look at a subset of these vectors which we call *admissible*.

Consider a network with feasible flows $f_1, f_2, \ldots, f_m$; that is, the flows $f_1, f_2, \ldots, f_m$ are routed in the network without violating the capacity constraints. We say that a vector $\alpha_1, \alpha_2, \ldots, \alpha_m$ is *admissible* if it is possible to route the flows $\alpha_1 f_1, \alpha_2 f_2, \ldots, \alpha_m f_m$ together in the network. Clearly, if $\forall i,\ \alpha_i = 1$ then the vector is admissible. We only consider vectors with $\alpha_i \geq 1$ for every i. In most cases, the TE scheme implies restrictions on the flows (e.g., by dictating that each flow can only use a specific set of paths, or that the ratio between the different paths of the same flow must be fixed). In these cases, we require that the flow $\alpha_1 f_1, \alpha_2 f_2, \ldots, \alpha_m f_m$ will respect these restrictions.

As mentioned before when discussing link utilization, presenting the data is an important issue and presenting a set of parameters (the α_i) for each timestamp is indeed problematic. To address this point we define a single parameter, based on the values of α_i, that represents the network flow utilization state. Let F be the total flow in the network at a given time, then the *Generalized Flow Utilization* (GFU) is defined by:

$$\text{GFU} = 1/F \sum_{i=1}^{k} f_i U(1/\alpha_i),$$

where U is a nondecreasing utilization function. For $U(x) = x$, this value is the weighted average of $1/\alpha_i$, which is always (like the average link utilization) a number between 0 and 1. The exact meaning of this value depends on the exact set of admissible αs.

Given a set of flows, if there is an admissible vector $\alpha_1, \alpha_2, \ldots, \alpha_m$ such that all α_i's are large, then the network is under utilized. It is easy to check whether a given vector $\alpha_1, \alpha_2, \ldots, \alpha_m$ is admissible, but it is computationally infeasible to characterize the set of all admissible vectors. Therefore, we look for vectors which provide useful information regarding the state of the network and the flows. We require the following conditions from all the vectors we examine:

- Measuring utilization across the board. Commonly in every network there is a flow that can easily be increased (e.g. a flow between two adjacent nodes which happen to be a source-destination pair of a link). The interesting question is whether there are many flows which can be increased simultaneously.
- Getting to the sweet spot. Over utilization of links in the network can cause packet loss while under utilization is a waste. We want a metric which captures the sweet spot, with many values of α_i which are all slightly higher than 1.
- Computability. We need to be able to find such a vector in an efficient way.

We start with a negative result. Let $V(k, \alpha)$ be the set of vectors with with k entries equal to α for some value $\alpha > 1$, and the rest of entries are equal to 1. Is there an admissible vector of α's in $V(k, \alpha)$? This question is interesting for given k, α: it indicates that k flows could be increased by a factor of α without hurting the rest. Thus, for example if there is an admissible vector in $V(0.95n, 1.2)$ then packet loss in most flows should not be too high, and if there is an admissible vector in $V(0.8n, 4)$ then the network is underutilized. However, we show that $V(k, \alpha)$ cannot be computed, or even well approximated:

Lemma 1: For any $\alpha > 1$ and $\epsilon > 0$, it is NP hard to distinguish between a network of n flows in which $V(n^{1-\epsilon}, \alpha)$ is admissible and a network in which $V(n^{\epsilon}, \alpha)$ is admissible. In particular, for $k = \Omega(n)$ it is NP hard to approximate $V(k, \alpha)$ to within a factor of $n^{1-\epsilon}$.

Proof: The proof is by reduction from independent set. Let $H = \langle V_H, E_H \rangle$ be a graph, where we want to know if H has an independent set of size k. We build a new graph G, where all the capacities of all edges in G are exactly $1 + \alpha$. The graph G will have $|V_H| + 2|E_H| + 1$ vertices:

1) It will have one target vertex t.
2) It will have $|V_H|$ source vertices, denoted s_h for every $s \in V_H$. Each of these vertices will originate a flow to t.
3) For every edge $e \in E_H$, it will have two vertices e_{in} and e_{out}.

Each flow can only use one specific path, and the edges of G are the edges of all the these paths. For each source vertex s_h where $h \in V_H$ we define a flow: Let $e^1, e^2, \ldots, e^h$ be the edges adjacent to the vertex h in the original graph H. The path of the flow which starts from s_h is

$$s_h \rightarrow e^1_{in} \rightarrow e^1_{out} \rightarrow e^2_{in} \rightarrow e^2_{out} \ldots \rightarrow e^h_{in} \rightarrow e^h_{out} \rightarrow t$$

The flow f_i will consist of s_i passing one unit to the target. The following claim is easy:

Claim 1: The graph H has an independent set of size k if and only if $V(k, \alpha)$ is admissible for G.

Proof: Suppose that H has an independent set U of size k. For every $h \in U$ increase the flow from s_h by a factor of α. Given $h, u \in U$ the paths they have to t do not intersect.

For the other direction, if $V(k, \alpha)$ is admissible, let U be the set of flows which are increased. For any $s_h, s_u \in U$ their paths to the source do not intersect, and thus u, h are not neighbors in H. Therefore, we can let $U_H = \{h : \ s_h \in U\}$ be an independent set in H. ■

This concludes the proof of the lemma. ■

On the positive side we can show, that for many natural utility functions (i.e., choice of U) we can find the set of values $\alpha_1, \alpha_2, \ldots, \alpha_n$ that minimize the value of the GFU.

Lemma 2: Let $U_p(x) = x^p$ for some $p \geq 1$. Given a network with flows $f_1, f_2, \ldots, f_m$ one can efficiently find an admissible vector $\alpha_1, \alpha_2, \ldots, \alpha_m$ minimizing:

$$1/F \sum_{i=1}^{k} f_i U_p(1/\alpha_i) = 1/F \sum_{i=1}^{k} f_i / \alpha_i^p.$$

Proof: We find the vector $\alpha_1, \ldots, \alpha_m$ by using convex optimization methods. We write a convex program with m variables, $x_1, x_2, \ldots, x_m$. The constraints are the flow constraints, where x_i corresponds to flow i. We also add the constraint that $x_i \geq f_i$. The target function to minimize is:

$$\sum_{i=1}^{k} f_i (\frac{f_i}{x_i})^p$$

To show that one can solve this optimization problem efficiently, we need to show that the target function is concave (since this is a minimization problem). Formally, we need to show that if $x_1, x_2, \ldots, x_m$ and $y_1, y_2, \ldots, y_m$ obey the flow constraints, then:

$$2\sum_{i=1}^{k} f_i(\frac{2f_i}{x_i+y_i})^p \leq \sum_{i=1}^{k} f_i(\frac{f_i}{x_i})^p + \sum_{i=1}^{k} f_i(\frac{f_i}{y_i})^p.$$

We show that this holds for any term in the sum independently. It is enough to show that for every x_i, y_i,

$$2(\frac{2}{x_i+y_i})^p \leq (\frac{1}{x_i})^p + (\frac{1}{y_i})^p.$$

Multiplying by $(x_i+y_i)^p$ and denoting $t = x_i/(x_i+y_i)$, this holds if and only if

$$\forall t, 2^{p+1} \leq \frac{1}{t^p} + \frac{1}{(1-t)^p}.$$

The right-hand-side has a minimum at $t = 1/2$, where equality holds. This finishes the proof of the lemma. ■

It is possible to prove a generalization of Lemma 2 to the case in which the TE restricts the possible flows, as long as the restrictions done by the TE can be inserted into the optimization. This is indeed the case for most TE algorithms, and in particular to ones that only allow a subset of the routes to be used for each flow.

To capture the idea of a sweet spot between over and under utilization, one may consider a utility function which has different behavior for different values of x (e.g. $U(x) = 1$ if $x \geq 0.8$ and $U(x) = x$ otherwise). However, computing the optimal α_i's for these functions is NP-complete, with a similar reduction to the one in Lemma 1.

We turn to study two natural flow utilization definitions, each capturing a different objective, and we discuss how to measure them and how to use them in practical network setting.

IV. Accommodating Flow Growth

In this section we consider a vector of α_i values which predict the capability of the network to cope with larger demands across the board. As we do not a-priori know which demand will grow the most, we take a conservative view, and require that the smallest values of α_i will be as large as possible. We call this vector α^{Growth}.

Formally let β_1 be the largest factor such that all demands can be increased by a factor of β_1 and still be satisfied by the network. Now increase all demands by β_1, clearly at least one link is saturated otherwise β_1 is not maximal, and thus at least one flow cannot be increased. We denote by b_1 the set of flows that were blocked at this point, and by l_1 the size of b_1. We set $\alpha_1^{\text{Growth}} = \ldots = \alpha_{l_1}^{\text{Growth}} = \beta_1$, and continue to find β_2 the largest constant such that all demands that were not blocked in the previous steps can be increased by a factor β_2 and still be satisfied by the network. Again there is a set of blocked flows b_2 and we set the value of $\alpha_{l_1+1}^{\text{Growth}} = \ldots = \alpha_{l_1+l_2}^{\text{Growth}} = \beta_2$, and continue this process to generate α_i^{Growth} in general. This is very similar to the "max-min-fair" vector in the sense of [5].

Given demands, the iterative linear programming approach of [6] can be used to compute optimal $(\alpha_i^{\text{Growth}})$ for all flows. This view is useful for determining the relative performance of existing network TE with the optimum possible unified increase in demand.

In production backbones the paths of the flows are determined either by the IP routing protocol or more likely by the TE scheme, and the utopian view that flows can be sent over an unrestricted set of paths is far from reality. Thus, in order to understand the current behavior of the network, it is useful to compute the vector $(\alpha_i^{\text{Growth}})$ corresponding to the current routing paths of the network flows. That is, all flows can be increased by a factor of $(\alpha_1^{\text{Growth}})$ without changing their current paths, all non-blocked flows can be increased by a factor of $(\alpha_2^{\text{Growth}})$ without changing the current paths, and so on. We present two ways to compute this α^{Growth} in an efficient manner: either by first examining flows, or by first examining links.

To compute by examining flows we make use of a sequence (b_i) of the flows blocked at step i. Define the *residual capacity* of link l at step i as

$$c_i(l) = c(l) - \sum_{\substack{l \,\in\, \text{path}(f): \\ \exists j \,<\, i \,:\, f \,=\, b_j}} \alpha_f^{\text{Growth}} \cdot f,$$

where $c(l)$ is the capacity of link l and f is the flow value Similarly, define the *residual utilization* of link l at step i as

$$u_i(l) = \sum_{\substack{l \,\in\, \text{path}(f): \\ \forall j<i: f \neq b_j}} f.$$

At step i, define a growth factor for each flow f as

$$g_{f,i} = \min_{l \,\in\, \text{path}(f)} \frac{c_i(l)}{u_i(l)}.$$

Select a flow f with minimal $g_{f,i}$, and set

$$\alpha_i^{\text{Growth}} = \alpha_f^{\text{Growth}} = g_{f,i}$$
$$b_i = f$$

Note that in the above process, there is always a link l such that all unblocked flows through l achieve the minimum $g_{f,i}$. This suggests examining the links directly. To compute by examining links, it is useful to define the sequence of α values assigned to links before assigning these values to the flows. Set $B_0 = \emptyset$ and, iteratively for $i = 1, 2, \ldots$,

$$\alpha_i^{\text{Growth}} = \min_{l\text{: link}} \frac{c(l) - \sum_{\substack{l \,\in\, \text{path}(f): \\ f \in \cup_{j<i} B_j}} \alpha_f^{\text{Growth}} f}{\sum_{\substack{l \,\in\, \text{path}(f): \\ f \notin \cup_{j<i} B_j}} f} \tag{1}$$

$$B_i = \{\text{set of all } f \text{ through } l \text{ appearing above}\} \setminus \bigcup_{j<i} B_j \tag{2}$$

Now set $\alpha_f^{\text{Growth}} = \alpha_i^{\text{Growth}}$ for every flow f where $f \in B_i$.

Computing by links performs fewer operations than computing by flows. In particular, fewer subtractions are needed, and those subtractions can be performed once, before blocking a link and do not affect subsequent comparisons. This reduces floating-point cancellation and stability issues when many flows traverse the same link.

When computing the above process, every flow is blocked exactly once. The per-link values appearing in Equation 1 can be stored in a mutable priority queue. When blocking a flow, all of the links that it traverses in the priority queue must have their priorities updated. Removal of a flow with $\leq k$ links triggers $\leq k$ updates of the priority queue, for complexity $O(k \cdot \log n)$ of each flow removal. Accordingly, the complexity for all updates is $O(m \cdot k \cdot \log n)$, where there are n flows traversing at most k links each, and m links. The total complexity including initialization is $O((n + m \cdot k) \cdot \log n)$.

The order in which links are blocked is also of interest. This order precisely prioritizes the links for procuring additional capacity when max-min-fair TE is employed, or, alternatively, the degree to which such TE is sensitive to congestion along that link when comparable flow patterns exist.

What can we infer on the network by looking at an α^{Growth} vector? Many small values (near 1) indicate that the network is over-utilized, and cannot accommodate growth. If the values are consistently large over time, the network is under-utilized, and one can probably serve the clients well with less resources (contrast this to the link utilization view where low average link utilization does not necessarily indicate a waste). The worst situation is when there are both small values and large ones. This means that some of the network cannot grow and may even experience packet loss or delays, while other parts of the network are under-utilized. Moreover, the TE algorithm is unable to use the extra bandwidth it has available in the under-utilized links to better serve the demands which are at risk. This situation calls for an assessment of the TE or the network structure.

The measure α^{Growth} is mostly concerned with accommodating future demand growth. The next subsection presents a measure tailored for risk assessment and for estimating the current quality of service experienced by clients.

V. Risk Assessment

While the previous definition is very good for capturing an overall network performance score, it is geared towards long term effects and planning. In this section, we focus our attention on the packet loss (and delays) different flows experience in the current network state. Understanding which flows are prone to failures and why is a crucial step in minimizing traffic at risk, which is an important objective for network operators.

Flows are bounded by the most congested link on their path (or paths), so in a way, not all links are equally important. The links that function as actual bottlenecks for certain flows are the ones that have the most critical impact on performance (in terms of delay and loss) and on the network ability to serve more traffic. For simplicity, we begin by assuming that each flow has a single path, and later expand the discussion to the multi-path scenario.

This sets the ground for another metric capturing the risk of each flow, or in other words, estimating the risk levels for each service or user. We call this definition α^{Risk} and define it formally as follows:

$$\alpha^{\text{Risk}}(f_j) = \frac{1}{\max\{\text{util}(e_i) | e_i \in \text{path}(f_j)\}},$$

where $c(e)$ is the capacity of e and

$$\text{util}(e) = \frac{\sum_{i=1}^{m} f_i(e)}{c(e)}.$$

Directly computing α^{Risk} is easy, for each flow along a single path we find the bottleneck link and use it to compute α_i^{Risk}. This can be done independently for each flow path and thus this computation can be easily distributed.

Since α^{Risk} is defined in terms of bottleneck utilization independently for each link we need first to show that it is indeed admissible.

Lemma 3: α_i^{Risk} is admissible.

Proof: Let e_i be an edge and let $\beta = \frac{1}{\text{util}(e_i)}$. Let $F(e) = \sum_{i=1}^{m} f_i | e \in f_i$ be the total flow over e and let $F'(e)$ be the total flow over e where every flow f_j has been multiplied by α_j^{Risk}. Let $f'_i = f_i \dot{\alpha}^{\text{Risk}}{}_i$ be the utilizations of the individual flows for $1 \leq i \leq m$. We will show that $F'(e_i) \leq c(e_i)$.

Indeed for all i,

$$F'(e_i) = \sum_{f_j \in e_i} f'_j = \sum_{f_j \in e_i} \frac{f_j}{\max\{\text{util}(e_k) \mid e_k \in \text{path}(f_j)\}} \tag{3}$$

$$\leq c_i \cdot \sum_{f_j \in e_i} \frac{f_j}{\text{util}(e_i)} = c_i. \tag{4}$$

$$\tag{5}$$

because e_i is one of the links in $\text{path}(f_j)$. ■

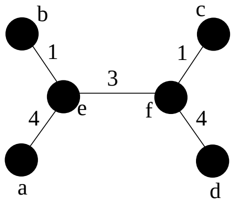

Fig. 5. α^{Risk} is not maximal.

α^{Risk} is conservative in the sense that it assumes that all demands grow at once and therefore distributes the residual capacity of each link along all the demands that traverse it. In a way, this is a pessimistic point of view, which goes hand in hand with risk assessments. The flow obtained by multiplying each flow i by α_i^{Risk} is not necessarily maximal, as can be seen in Figure 5. Two demands are depicted, one from a to d and another from b to c. Both demands send one unit of flow. Link capacities are shown in the figure. We get that $\alpha_{(a,d)}^{\text{Risk}} = 3/2$, since the bottleneck edge is (e, f) and has two units of flow on it and a capacity of 3. $\alpha_{(b,c)}^{\text{Risk}} = 1$, since the bottleneck edge is (b, e) (or (f, c)) and is saturated. If we take each of the flows and multiply by its α^{Risk} we get that $f(a-d) = 1.5$ and $f(b-c) = 1$. Note that this is not a maximal flow in the graph – the flow $(a-d)$ can still be increased. This demonstrates the conservativeness of this definition and is due to the locality of the computation. When computing $\alpha^{\text{Risk}}_{(a-d)}$ we ignore the other flow and assume it is not bottlenecked before $(a-d)$.

VI. Practical Use of Flow Utilization

In this section we show how flow utilization can be used in practice to provide a better understanding of the network state. Figure 6 depicts both the link and the flow utilization in the Google backbone. The Flow utilization is the Generalized Flow Utilization (GFU) from Section III were we used $U(x) = x$. We averaged the flow and link utilization values over all the links per timestamp and presented all the timestamps for each hour on the x-point matching that hour. In this case the flow utilization varies over the day between 40% and 50%, so on average flows go through (fairly high) congested links during peak utilization time during the day. This suggests that the network is in fact much more utilized than indicated by the link utilization curve alone.

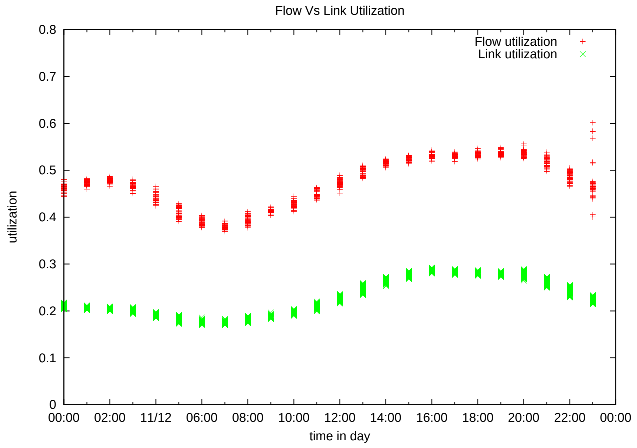

Fig. 6. Flow vs. Link Utilization.

We turn to show how to use different utilization functions. Figure 7 depicts traffic at risk on the Google backbone. Each line corresponds to a different risk level. The x-axis is the time and the y-axis shows the percentage of traffic (out of the total Google backbone traffic) that is over the matching risk level. In other words, we can see what percentage of Google traffic had α^{Risk} greater than 1.42 (for 70%), 1.25 (for 80%) and 1.11 (for 90%). Since the timescales in which these were computed is rather small, the data varies a lot. Smoothing can easily be applied on this data. During the peak of this day, about 28% of the traffic was at α^{Risk} levels of 1.42, or in other words had bottleneck links that were at least 70% utilized.

So far, we ignored the issue of how multiple paths should be handled. Note that this is not an issue for α^{Growth} since α^{Growth} computes by how much a flow can grow on all of its paths. When there are multiple paths for each flow, we compute for every flow i a vector of numbers $\alpha^{\text{Risk}}_{i,j}$ where j is the j-th path for flow i. However, to analyze the quality of service experienced by the client of flow i, it is desired to aggregate this vector into a single number. This can be done in two different ways, depending on what we are trying to model:

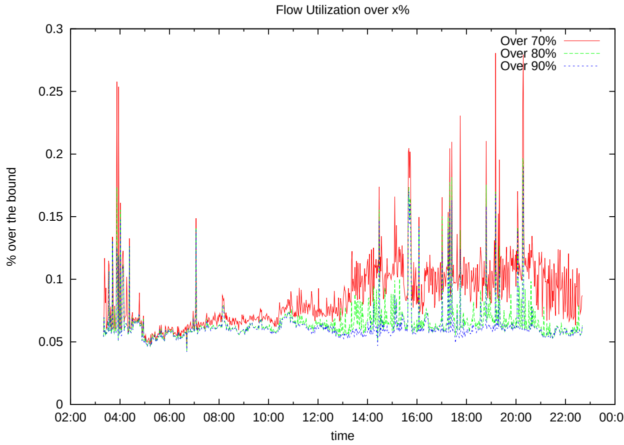

Fig. 7. Traffic at risk over a day.

1) To estimate the packet loss that flow i experiences due to network contention, it makes sense to take a weighted average of $\alpha^{\text{Risk}}_{i,j}$, where the weights are the amount of flow in each path.
2) To estimate the delay which flow i experiences due to network contention, it makes sense to take the maximum, or $\max_j \alpha^{\text{Risk}}_{i,j}$.

An interesting question is how big is the gap between the two vectors α^{Growth} and α^{Risk}. Theoretically it can be very large as shown in the following example. Specifically, we show a scenario where $\alpha^{\text{Growth}} = (1, n, n, n)$, while $\alpha^{\text{Risk}} \sim (1, 2, 2, 2)$, where n is the number of nodes in the graph. In this example, we have four flows: S-T, A-B, C-D and E-F. Edge capacities are shown on the graph. S-T is sending n units of flow along the path S-A-B-C-D-E-F-T and is saturated, while the other 3 are sending one unit of flow each on a path (composed of a single edge) of capacity $2n$. Each of the edges $A \rightarrow B$, $C \rightarrow D$, $E \rightarrow F$ still has available capacity of $n-1$.

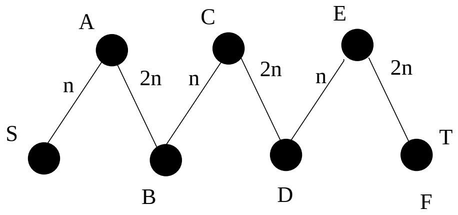

Fig. 8. Example for the gap between α^{Growth} and α^{Risk}.

The flow S-T cannot grow, so it has an α^{Growth} of 1. All the other flows can grow by factor n and therefore have an α^{Growth} of n.

The flow S-T cannot grow, so it also has an α^{Risk} of value 1. Once this flow cannot grow further, all the other flows have a bottleneck with two flows traversing it, completing a demand for $n+1$ on the bottleneck. The α^{Risk} we get for these flows is $\frac{2n}{n+1} \sim 2$. Overall, we get $\alpha^{\text{Growth}} = (1, n, n, n)$ while $\alpha^{\text{Risk}} \sim (1, 2, 2, 2)$.

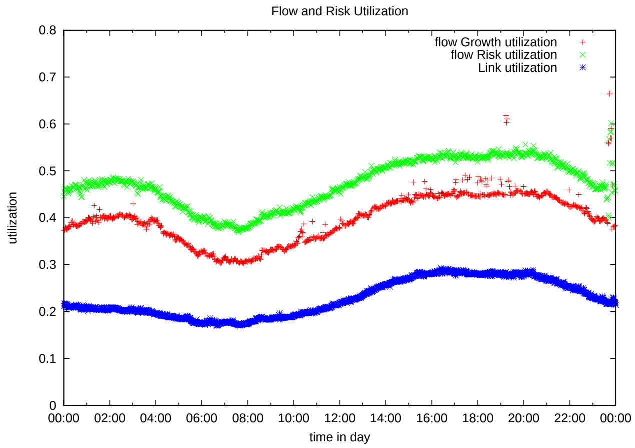

Fig. 9. Empirical gap between α^{Growth} and α^{Risk} on the Google backbone.

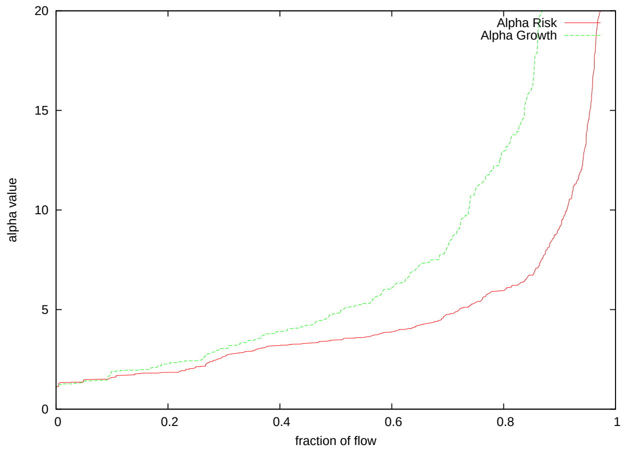

Fig. 10. α^{Growth} and α^{Risk} on the Google backbone at a specific timestamp.

In reality the situation is somewhat different. While the two definitions do address different aspect of flow view and are definitely not identical, the gap between them is not so big.

Figure 9 depicts the flow utilization (GFU) using both α^{Risk} and α^{Growth} and the link utilization values on the Google backbone (over all links and all day). First, as expected the GFU that uses α^{Growth} is smaller than the GFU that uses α^{Risk} since $\alpha^{\text{Growth}} \geq \alpha^{\text{Risk}}$ as α^{Risk} is more conservative. Also, it clearly shown in the figure that both GFUs are higher than the link utilization. The daily traffic pattern is shown on all three lines, as expected. The actual network utilization is around 30%-40% according to α^{Risk} and the α^{Growth} is at 25%-33%. On the other hand, the link utilization peaks at 20%. In fact, that network can not grow by a factor of 5 as the link utilization suggests. Under the Google demands and TE scheme the network can only really grow by a factor closer to 2.5.

Another interesting view is depicted in Figure 10 where we show the α^{Risk} α^{Growth} in one specific timestamp. We collected both values for each link. The x-axis presents the percentage of traffic that has an alpha of at most the matching y-value. We left out very large alpha values which occur on links with little or not utilization at this specific timestamp. About 50% of the traffic has α^{Growth} of at most 5, meaning it cannot grow by a factor greater than 5, compared to 80% of the traffic with α^{Risk} of at most 5. This demonstates the fact that α^{Risk} is in fact riskier. According to it 80% of the traffic is *at risk 5* compared to 50% according to α^{Growth}.

One can see that the general notion of flow utilization is useful when presenting complex network data. Different aspects like the long term ability to accommodate growth in the demand or the acute risk can be examine and appropriate action can then be taken.

VII. Related Work

As mentioned in Section II only a handful of papers were devoted to the utilization of production backbone network. In addition to the papers mentioned there (in Section II), several papers study traffic over backbones, and specifically mention changes in backbone link utilization (see [7], [8], [9], and [10]). The overall picture is that traffic varies significantly over time and that random anomalies occur. The flow utilization view described in this paper can be helpful for long term network design, and potentially also evaluating network performance and for real time anomaly detection.

Traffic engineering has been heavily studied both in the theoretical and empirical aspects. Many works compare TE approaches on real-life networks (see [5], [11], [12], [13]) using various metrics such as fairness, throughput and utilization. The authors of [11] discuss the problem of finding a routing scheme that optimizes the network under several (possible contradicting) objectives and how to match these to the TE goals. While their experiments compare important metrics such as the maximal and mean link utilization, as well as latency metrics and the amount of residual bandwidth in the network, it is not clear that these parameters reflect the actual traffic growth factor that could be routed by the network.

The authors of [12] compare of metrics widely used to evaluate the effects of TE on application level performance. Based on their empirical results, link utilization is not a good measurement for this. In this paper we use a flow centric view to address the same issues. Furthermore, the flow utilization view can be used with utility functions to put extra emphasis on certain applications that are business critical.

Several recent works, such as [14], [15], and [16], discuss possible ways to change peers, services and links to achieve optimal TE under specified metrics. This demonstrates the key role of TE performance assessment and the importance of good metrics that provide meaningful information regarding the state of the network with respect to TE performance.

This is even more important today, when networks and services are both being changed continuously, and software defined networks (SDNs) (see for example [4]) are gaining popularity in the intra-cluster domain, and more recently also addressing the backbone scenario [17].

VIII. Conclusion

Network utilization is a key metric in operating a network, as well as in planning future upgrades. However, looking at the links does not tell the whole story – the average utilization can look good, since one area of the network is over utilized, and the other is under-utilized. Even looking at the distribution of the utilization across links does not tell us much – for example a single link operating at capacity can mean a local problem, or a bottleneck which clogs the entire network. Moreover, this link can be used for high priority or low priority traffic, and the traffic traversing it can have different levels of sensitivity to delays and loss.

To circumvent these problems, and to offer a better understanding of the state of the network, we propose taking the flow perspective, and keeping track of the GFU of the network, or more concretely at α^{Growth} and α^{Risk} (other GFU vectors may also come in handy in some cases). By looking at these vectors, one can:

1) Take the user perspective, and understand how she experiences the network. This can also allow the network planner (and the network operator) to discriminate between high priority users and low priority ones, e.g. by upgrading the network when the high priority users experience low quality.
2) Focus the attention on the traffic at risk. By looking at α^{Risk} one can identify if there is traffic at risk, how much of it is in risk, and where is the risk coming from.
3) Tune the network to be in the sweet spot, where it is not under utilized and not over utilized. Relatively uniform values of α^{Growth}, which are bounded between 1.2 and 1.8 indicate that the network is properly utilized.

There are several important directions to pursue. It is not clear how to assess the network utilization under failure, without simulating a failure, solving the TE, and computing the utilization. Moreover, as failures can occur at many places, one needs to find some aggregate metric for this.

A related question is how to use the network utilization for capacity planing and network upgrades. Using α^{Growth}, one can know where are the bottlenecks, and what should be upgraded. However, predicting the full impact of the upgrade on the utilization is non trivial, as the TE may use the new links in unexpected ways. One can simulate the new network and see its behavior, but there is a limit to the accuracy of any such simulation.

Finally, now that we have the new metrics for utilization, it is important to measure it in many different networks, and see how they evolve over time, and especially how they behave under special network conditions (due either to faults or to rapid changes in the demand such as breaking news or the opening ceremony of the Olympiad).

References

[1] B. A. Odlyzko and A. Odlyzko, “Data networks are lightly utilized, and will stay that way,” *Review of Network Economics*, vol. 2, pp. 210–237, 1999.

[2] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and S. Diot, “Packet-level traffic measurements from the sprint ip backbone,” *Network, IEEE*, vol. 17, no. 6, pp. 6 – 16, nov.-dec. 2003.

[3] A. Nucci, N. Taft, C. Barakat, and P. Thiran, “Controlled use of excess backbone bandwidth for providing new services in ip-over-wdm networks,” *IEEE Journal on Selected Areas in Communications*, pp. 1692–1707, 2004.

[4] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, “Openflow: enabling innovation in campus networks,” *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Mar. 2008. [Online]. Available: http://doi.acm.org/10.1145/1355734.1355746

[5] E. Danna, A. Hassidim, H. Kaplan, A. Kumar, Y. Mansour, D. Raz, and M. Segalov, “Upward max min fairness.” in *INFOCOM*, A. G. Greenberg and K. Sohraby, Eds. IEEE, 2012, pp. 837–845.

[6] D. Bertsekas and R. Gallager, *Data Networks*. Englewood Cliffs: Prentice-Hall, 2001.

[7] A. Feldmann, A. C. Gilbert, W. Willinger, and T. Kurtz, “The changing nature of network traffic: Scaling phenomena,” *Computer Communication Review*, vol. 28, pp. 5–29, 1998.

[8] C. Lee, D. K. Lee, Y. Yi, and S. Moon, “Operating a network link at 100*Proceedings of the 12th international conference on Passive and active measurement*, ser. PAM’11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 1–10. [Online]. Available: http://dl.acm.org/citation.cfm?id=1987510.1987511

[9] M. Fomenkov, K. Keys, D. Moore, and k. claffy, “Longitudinal study of Internet traffic from 1998-2003,” in *Winter International Symposium on Information and Communication Technologies (WISICT) 2004*. Cancun, Mexico: Winter International Symposium on Information and Communication Technologies (WISICT), Jan 2003, pp. 1–6.

[10] P. Borgnat, G. Dewaele, K. Fukuda, P. Abry, and K. Cho, “Seven years and one day: Sketching the evolution of internet traffic,” in *INFOCOM*, 2009.

[11] S. Balon, F. Skivée, and G. Leduc, “How well do traffic engineering objective functions meet te requirements?” in *Proceedings of the 5th international IFIP-TC6 conference on Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; Mobile and Wireless Communications Systems*, ser. NETWORKING’06. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 75–86. [Online]. Available: http://dx.doi.org/10.1007/11753810_7

[12] A. Sharma, A. Mishra, V. Kumar, and A. Venkataramani, “Beyond mlu: An application-centric comparison of traffic engineering schemes,” in *INFOCOM’11*, 2011, pp. 721–729.

[13] E. Danna, S. Mandal, and A. Singh, “A practical algorithm for balancing the max-min fairness and throughput objectives in traffic engineering.” in *INFOCOM*, A. G. Greenberg and K. Sohraby, Eds. IEEE, 2012, pp. 846–854. [Online]. Available: http://dblp.uni-trier.de/db/conf/infocom/infocom2012.html#DannaMS12

[14] A. Sridharan, R. Guérin, and C. Diot, “Achieving near-optimal traffic engineering solutions for current ospf/is-is networks,” *IEEE/ACM Trans. Netw.*, vol. 13, no. 2, pp. 234–247, Apr. 2005. [Online]. Available: http://dx.doi.org/10.1109/TNET.2005.845549

[15] E. Keller, M. Schapira, and J. Rexford, “Rehoming edge links for better traffic engineering.” *Computer Communication Review*, vol. 42, no. 2, pp. 65–71, 2012. [Online]. Available: http://dblp.uni-trier.de/db/journals/ccr/ccr42.html#KellerSR12

[16] M. Agrawal, S. R. Bailey, A. Greenberg, J. Pastor, P. Sebos, S. Seshan, K. V. D. Merwe, and J. Yates, “Routerfarm: Towards a dynamic, manageable network edge,” in *In Proc. ACM SIGCOMM Workshop on Internet Network Management (INM*, 2006.

[17] A. R. Sharafat, S. Das, G. Parulkar, and N. Mckeown, “Mpls-te and mpls vpns with openflow,” in *SIGCOMM’11*.