TeaVaR: Striking the Right Utilization-Availability Balance in WAN Traffic Engineering

Jeremy Bogle Nikhil Bhatia Manya Ghobadi

Ishai Menache* Nikolaj Bjørner* Asaf Valadarsky† Michael Schapira†

Massachusetts Institute of Technology * Microsoft Research † Hebrew University

ABSTRACT

To keep up with the continuous growth in demand, cloud providers spend millions of dollars augmenting the capacity of their wide-area backbones and devote significant effort to efficiently utilizing WAN capacity. A key challenge is striking a good balance between network *utilization* and *availability*, as these are inherently at odds; a highly utilized network might not be able to withstand unexpected traffic shifts resulting from link/node failures. We advocate a novel approach to this challenge that draws inspiration from financial risk theory: leverage empirical data to generate a probabilistic model of network failures and maximize bandwidth allocation to network users subject to an operator-specified *availability target*. Our approach enables network operators to strike the utilization-availability balance that best suits their goals and operational reality. We present TeaVaR (Traffic Engineering Applying Value at Risk), a system that realizes this risk management approach to traffic engineering (TE). We compare TeaVaR to state-of-the-art TE solutions through extensive simulations across many network topologies, failure scenarios, and traffic patterns, including benchmarks extrapolated from Microsoft's WAN. Our results show that with TeaVaR, operators can support up to twice as much throughput as state-of-the-art TE schemes, at the same level of availability.

CCS CONCEPTS

• **Networks** → **Network algorithms**; **Traffic engineering algorithms**; **Network economics**; **Network performance evaluation**;

KEYWORDS

Utilization, Availability, Traffic engineering, Network optimization

ACM Reference Format:

Jeremy Bogle, Nikhil Bhatia, Manya Ghobadi, Ishai Menache, Nikolaj Bjørner, Asaf Valadarsky, Michael Schapira. 2019. TeaVaR: Striking the Right Utilization-Availability Balance in WAN Traffic Engineering. In *SIGCOMM '19: 2019 Conference of the ACM Special Interest Group on Data Communication, August 19–23, 2019, Beijing, China.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3341302.3342069

SIGCOMM '19, August 19–23, 2019, Beijing, China

ACM ISBN 978-1-4503-5956-6/19/08…$15.00

https://doi.org/10.1145/3341302.3342069

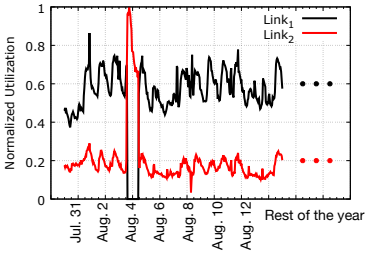


Figure 1: $Link_2$'s utilization is kept low to sustain the traffic shift when failures happen.

1 INTRODUCTION

Traffic engineering (TE), the dynamic adjustment of traffic splitting across network paths, is fundamental to networking and has received extensive attention in a broad variety of contexts [1, 2, 8, 21, 27, 29, 31, 34, 35, 38, 43, 57]. Given the high cost of wide-area backbone networks (WANs), large service providers (e.g., Amazon, Facebook, Google, Microsoft) are investing heavily in optimizing their WAN TE, leveraging Software-Defined Networking (SDN) to globally optimize routing and bandwidth allocation to users [27, 29, 37, 42, 43].

A crucial challenge faced by WAN operators is striking a good balance between network *utilization* and *availability* in the presence of node/link failures [5, 25, 28, 43, 48]. These two objectives are inherently at odds; providing high availability requires keeping network utilization sufficiently low to absorb shifts in traffic when failures occur. To attain high availability, today's backbone networks are typically operated at fairly low utilization so as to meet user traffic demands while providing high availability (e.g., 99%+ [28]) in the presence of failures.

Fig. 1 plots the link utilization of two IP links in a backbone network in North America with the same source location but different destinations. The utilization of each link is normalized by the maximum achieved link utilization, hence, the actual link utilization is lower than plotted. On August 4, $Link_1$ failed, and its utilization dropped to zero. This, in turn, increased the utilization of $Link_2$. Importantly, however, under normal conditions, the normalized utilization of $Link_2$ is only around 20%, making $Link_2$ underutilized almost all the time. While network utilization can be increased by sending low-priority background traffic over underutilized links, this does not improve network utilization for high priority traffic, which is the focus of this paper (§6).

We show that state-of-the-art TE schemes fail to maximize the traffic load that can be supported by the WAN for the desired level of availability (§5). Under these schemes, the ratio of the bandwidth allocated to users to the available capacity must be kept lower than necessary, resulting in needlessly low network utilization. We argue that to remedy this, operators should *explicitly* optimize network utilization subject to target availability thresholds. Today's TE schemes do not explicitly consider availability. Instead, the number of concurrent link/node failures the TE configuration can withstand (e.g., by sending traffic on link-disjoint network paths) is sometimes used as a proxy for availability. However, the failure probability of a single link can greatly differ across links, sometimes by three orders of magnitude [23]. Consequently, some failure scenarios involving two links might be more probable than others involving a single link. Alternatively, some failure scenarios might have negligible probability, and so lowering network utilization to accommodate them is wasteful and has no meaningful bearing on availability.

Operators actually have high visibility into failure patterns and dynamics. For example, link failures are more probable during working hours [25] and can be predicted based on sudden drops in optical signal quality, "with a 50% chance of an outage within an hour of a drop event and a 70% chance of an outage within one day" [23]. We posit that this wealth of timely empirical data on node/link failures in the WAN should be exploited to explicitly reason about the probability of different failure scenarios when optimizing TE. We present TeaVaR (Traffic Engineering Applying Value at Risk), a TE optimization framework that enables operators to harness this information to tune the tradeoff between network utilization and availability and, by so doing, strike a balance that best suits their goals. To the best of our knowledge, TeaVaR is the first formal TE framework that enables operators to jointly optimize network utilization and availability. We refer the reader to Section 7 for a discussion of related work on TE, capacity planning, and other risk-aware approaches to networking.

Under TeaVaR, a probabilistic model of failure scenarios is first generated from empirical data. Then, TE optimization that draws on the notion of *Conditional Value at Risk* (*CVaR*) [50] minimization is applied to assign bandwidth shares to network users. TeaVaR enables formulating guarantees such as "user i is guaranteed b_i network bandwidth at least β% of the time," and computing bandwidth assignments that achieve these guarantees for a operator-specified value of β.

To realize this approach to TE, we grapple with the *algorithmic* challenges of formulating CVaR-based TE, such as how to achieve fairness across network users, and also with various *operational* challenges, such as ensuring that the running time of our algorithm scales well with the size and complexity of the network. In particular, we cast the CVaR-based TE as a Linear Program (LP) with a manageable number of constraints for realistic network topologies, thus enabling the efficient computation of optimal TE solutions.

To evaluate TeaVaR, we conduct extensive simulations, comparing its performance with that of other TE systems across a variety of scenarios, traffic matrices, and topologies. We first analyze the failure data collected from the inter-datacenter backbone network of Microsoft. Our dataset consists of time-to-failure and failure duration of links over a year at 15-minute granularity. We compute the failure probability for individual links as well as for Shared Risk Groups (SRGs) [54] corresponding to correlated link failures. We then apply these probability distributions to various network topologies, including ATT, B4, IBM, and Microsoft.

Our results show that with TeaVaR the operator can support up to twice as much traffic as with state-of-the-art TE schemes, at the same level of availability. Importantly, TeaVaR, which optimizes how user traffic is split across network tunnels, can be coupled with *any* scheme for WAN tunnel selection, including oblivious routing [38], k-shortest paths, and link-disjoint routes. We also show that our optimization is fairly robust to inaccuracies in failure probability estimations. Indeed, a surprising takeaway from our evaluation results is that as long as the probabilistic failure model used is within 20% of actual failure probabilities, the optimization results in roughly only 6% error in loss calculation.

To enable the community to explore our ideas and to facilitate the reproducibility of our results, our code is available online.[1] This work does not raise any ethical issues.

2 MOTIVATING TEAVAR

The number of concurrent node/link failures a TE configuration can withstand is sometimes used as a proxy for availability. This can be manifested, e.g., in sending user traffic on multiple network paths (tunnels) that do not share any, or share only a few, links, or in splitting traffic across paths in a manner resilient to a certain number of concurrent link failures, as advocated in [43]. In this section we explain why reasoning about availability in terms of the number of concurrent failures that can be tolerated is often not enough. We demonstrate this using the recently proposed Forward Fault Correction (FFC) TE scheme [43].

FFC as an illustration. FFC maximizes bandwidth allocation to be robust for up to k concurrent link failures, for a configurable value k. To accomplish this, FFC optimization sets a cap on the maximum bandwidth b_i each network flow i (identified by source/destination pair) can utilize and generates routing (and rerouting) rules, such that the network can simultaneously support b_i bandwidth for each flow i in any failure scenario that involves at most k failures.

We illustrate FFC in Fig. 2, where source node s is connected to destination node d via three links, each of capacity 10Gbps. Suppose that the objective is to support the maximum total amount of traffic from s to d in a manner that is resilient to at most two concurrent link failures. Fig. 2(b) presents the optimal solution under FFC: rate-limiting the (s, d) flow to send at 10Gbps and *always* splitting traffic equally between all links that are intact; e.g., when no link failures occur, traffic is sent at $\frac{10}{3}$Gbps on each link, when a single link failure occurs, each of the two surviving links carries 5Gbps, and with two link failures, all traffic is sent on the single surviving link. Thus, this solution guarantees the flow-reserved bandwidth of 10Gbps without exceeding link capacities under any failure scenario that involves at most two failed links. Observe, however, that this comes at the cost of keeping each link underutilized (one-third utilization) when no failures occur.

Striking the right balance. We ask whether high availability can be achieved without such drastic over-provisioning. Approaches such as FFC are compelling in that they provide strong availability

[1] http://teavar.csail.mit.edu

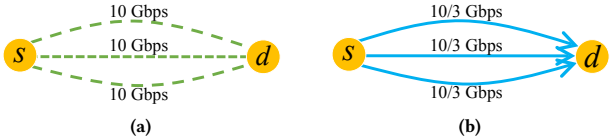


Figure 2: (a) A network of three links each with 10Gbps bandwidth; (b) Under conventional TE schemes, such as FFC [43], the total admissible traffic is *always 10Gbps*, split equally between paths (each carrying $\frac{10}{3}$Gbps).

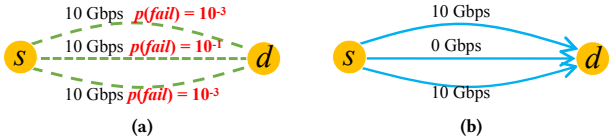


Figure 3: (a) The same network as in Fig. 2(a), with added information about link failure probabilities; (b) A possible flow allocations under TeaVaR with total admissible traffic of 20Gbps 99.8% of the time.

guarantees; in Fig. 2(b), the (s, d) flow is guaranteed a total bandwidth of 10Gbps even if two links become *permanently* unavailable. Suppose, however, that the availability, i.e., the fraction of time a link is up, is consistently 99.9% for each of the three links. In this scenario, the network can easily support 30Gbps throughput (3× improvement over FFC) around 99.9% of the time simply by utilizing the full bandwidth of each link and never rerouting traffic.

This example captures the limitations of *failure probability* agnostic approaches to TE, such as FFC; specifically, they ignore the underlying link availability (and the derived probability of failure). As discussed in [23, 25], link availability greatly varies across different links. Consequently, probability-oblivious TE solutions might lead to low network efficiency under prevailing conditions to accommodate potentially highly unlikely failure scenarios (i.e., with little bearing on availability). However, not only might a probability-oblivious approach overemphasize *unlikely* failure scenarios, it might even disregard *likely* failure scenarios. Consider a scenario where three links in a large network have low availability (say, 99% each), and all other links have extremely high availability (say, 99.999%). When the operator's objective is to withstand two concurrent link failures, the scenario where the three less available links might be simultaneously unavailable will not be considered, whereas much less likely scenarios in which two of the highly available links fail simultaneously will be considered.

To motivate our risk-management approach, we revisit the example in Fig. 2. Now, suppose the probability of a link being up is as described in the figure, and the link failure probabilities are uncorrelated (we will discuss correlated failures in §4). In this case, the probability of different failure scenarios can be expressed in terms of individual links' failure probabilities (e.g., the probability of all three links failing simultaneously is 10^{-7}). Under these failure probabilities, the network can support 30Gbps traffic almost 90% of the time simply by utilizing the full bandwidth of each link and not rerouting traffic in the event of failures. FFC's solution, shown in Fig. 2(b), can be regarded as corresponding to the objective of maximizing the throughput for a level of availability in the order of 7 nines (99.99999%), as the scenario of all links failing concurrently occurs with probability 10^{-7}. Observe that the bandwidth assignment in Fig. 3(b) guarantees a total throughput of 20Gbps at a level of availability of nearly 3 nines (99.8%).[2] Thus, the network administrator can trade network utilization for availability to reflect the operational objectives and strike a balance between the two.

[2]This is because the probability of the upper and lower links both being up, no matter what happens with the middle link, is $(1 - 10^{-3})^2 = 0.998$.

Our approach: risk-aware TE. Under TeaVaR, instead of reasoning about availability indirectly in terms of the maximum number of tolerable failures as in [43], network operators can generate a probabilistic failure model from empirical data (e.g., encompassing uncorrelated/correlated link failures, node failures, signal decay, etc.) and optimize TE with respect to an availability bound. We describe our approach in the following sections.

Note that our approach to risk-aware TE is orthogonal and complementary to the challenge of capacity planning. While capacity planning is focused on determining in what manner capacity should be augmented to the WAN to provide high availability, our goal is to optimize the utilization of available network capacity with respect to *real-time* information about traffic demands and expected failures. We elaborate on this relation in Section 7.

3 PROBABILISTIC TRAFFIC ENGINEERING

In this section, we relate the central concept of Value at Risk (VaR) in finance to resource allocation in networks and, more specifically, to TE. We then highlight the main challenges and ideas underlying TeaVaR—a probabilistic TE solution. A full description of TeaVaR appears in Section 4.

3.1 Probabilistic Risk-Management in Finance

In many financial contexts, the goal of an investor is to manage a collection of assets (e.g., stocks), also called a portfolio, so as to maximize the expected return on the investment *while* considering the probability of possible market changes that could result in losses (or smaller-than-expected gains).

Consider a setting in which an investor must decide how much of each of n stocks to acquire by quantifying the return from different investment possibilities. Let $x = (x_1, \ldots, x_n)$ be a vector representing an investment, where x_i represents the amount of stock i acquired, and let $y = (y_1, \ldots, y_n)$ be a vector that is randomly generated from a probability distribution reflecting market statistics, where y_i represents the return on investing in stock i. In financial risk literature, vector x is termed *the control* and vector y is termed *the uncertainty vector*. The loss function $L(x, y)$ captures the return on investment x under y and is simply $L(x, y) = -\Sigma_{i=1}^{n} x_i y_i$, i.e., the negative of the gain.

Investors wish to provide customers with bounds on the loss they might incur, such as "the loss will be less than \$100 with probability 0.95," or "the loss will be less than \$500 with probability 0.99." Value at Risk (VaR) [33] captures precisely these bounds. Given a probability threshold β (say $\beta = 0.99$), VaR_β provides a

probabilistic upper bound on the loss: the loss is less than VaR_β with probability β.

Fig. 4 gives a graphical illustration of the concepts of VaR_β (and $CVaR_\beta$ which we describe below). For a given control vector x and probability distribution on the uncertainty vector y, the figure plots the probability mass function of individual scenarios (x, y), sorted according to the loss associated with each scenario. Assuming all possible scenarios are considered, the total area under the curve amounts to 1. At the point on the x-axis marked by $\xi = VaR_\beta(x)$, the area under the curve is greater than or equal to β. Given a probability threshold β (say $\beta = 0.99$) and a fixed control x, $VaR_\beta(x)$ provides a probabilistic upper bound on the loss: the loss is less than $VaR_\beta(x)$ with probability β. Equivalently, $VaR_\beta(x)$ is the β-percentile of the loss given x. Value at Risk (VaR_β) is obtained by minimizing $VaR_\beta(x)$ (or ξ) over all possible control vectors x, for a given a probability threshold β. The VaR notion has been applied in various contexts, such as hedge fund investments [51], energy markets [14], credit risk [3], and even cancer treatment [45].

We point out that VaR_β does not necessarily minimize the loss at the *tail* (colored in red in Fig. 4), i.e., the worst-case scenarios in terms of probability, which have total probability mass of at most $1 - \beta$. A closely related risk measure that does minimize the loss at the tail is termed *β-Conditional Value at Risk* ($CVaR_\beta$)[50]; $CVaR_\beta$ is defined as the expected loss at the tail, or, equivalently, the expected loss of all scenarios with loss greater or equal to VaR_β. VaR minimization is typically intractable. In contrast, minimizing CVaR can be cast as a convex optimization problem under mild assumptions [50]. Further, minimizing CVaR can be a good proxy for minimizing VaR.

3.2 Probabilistic Risk Management in Networks

Optimizing traffic flow in a network entails contending with loss, which, in this context, is due to the possibility of failing to satisfy user demands when traffic shifts as link/node failures congest the network. We present a high-level overview of how the VaR and CVaR can be applied to this context and defer the formal presentation to Section 4.

We model the WAN as a network graph, in which nodes represent switches, edges represent links, and each link is associated with a capacity. Links (or, more broadly, shared risk groups) also have failure probabilities. As in prior studies [27, 29, 43], in each time epoch, a set of source-destination switch-pairs ("commodities" or "flows") wish to communicate where each such pair i is associated with a demand d_i, and a fixed set of possible routes (or tunnels) R_i on which its traffic can be routed.

Intuitively, under our formulation of TE optimization as a risk-management challenge, the control vector x captures how much bandwidth is allocated to each flow on each of its tunnels, and the uncertainty vector y specifies, for each tunnel, whether the tunnel is available or not (i.e., whether all of its links are up). Note that y is stochastic, and its probability distribution is derived from the probabilities of the underlying failure events (e.g., link/node failures). Our aim is to maximize the bandwidth assigned to users subject to a desired, operator-specified, availability threshold β.

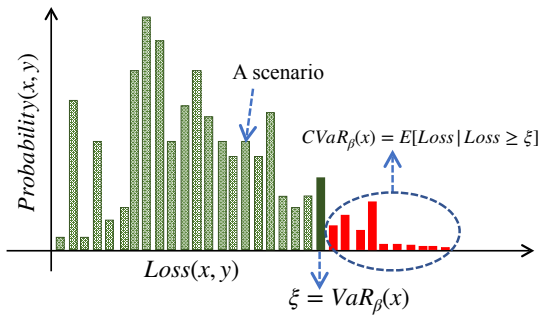


Figure 4: An illustration of Value at Risk, $VaR_\beta(x)$, and Conditional Value at Risk, $CVaR_\beta(x)$. Given a probability threshold β (say $\beta = 0.99$) and a decision vector x, $VaR_\beta(x)$ provides a probabilistic upper bound on the loss: the loss is less than $VaR_\beta(x)$ with probability β. $CVaR_\beta(x)$ captures the expected loss of all the scenarios where loss is greater than $VaR_\beta(x)$ [52].

However, applying $CVaR_\beta$ to network resource allocation faces three nontrivial challenges:

Challenge: Achieving fairness across network users. Avoiding starvation and achieving fairness are arguably less pivotal in stock markets, but they are essential in network resource allocation. In particular, TE involves multiple network users, and a crucial requirement is that high bandwidth and availability guarantees for some users not come at the expense of unacceptable bandwidth or availability for others. This, in our formulation, translates into carefully choosing the loss function $L(x, y)$ so that minimizing the chosen notion of loss implies such undesirable phenomena do not occur. We show how this is accomplished in §4.

Challenge: Capturing fast rerouting of traffic in the data plane. Unlike the above formulation of stock management, in TE the consequences of the realization of the uncertainty vector cannot be captured solely by a simple loss function such as $L(x, y) = -\Sigma_{i=1}^{n} x_i y_i$. This is because our CVaR-based optimization formalism must take into account that the unavailability of a certain tunnel might imply more traffic having to traverse *other* tunnels.

Providing high availability in WAN TE cannot rely on online re-computation of tunnels as this can be too time consuming and adversely impact availability [43, 54]. As in [43, 54], to quickly recover from failures, TeaVaR re-adjust traffic splitting ratios on surviving tunnels via re-hashing mechanisms implemented in the data plane. Thus, the realization of the uncertainty vector, which corresponds to a specification of which tunnels are up, impacts the control, capturing how much is sent on each tunnel.

Challenge: Achieving computational tractability. A naive formulation of CVaR-minimizing TE machinery yields a non-convex optimization problem. Hence, the first challenge is to transform the basic formulation into an equivalent convex program. We are, in fact, able to formulate our TE optimization as a Linear Program through careful reformulation with auxiliary variables (see Appendix A for details). In addition, because the number of all possible failure scenarios increases exponentially with the network size,

TE Input	$G(V, E)$	Network graph with switches V and links E.
	$c_e \in C$	The bandwidth capacity of link $e \in E$.
	$d_i \in D$	The bandwidth demand of flow i.
	$R_i \in R$	Set of tunnels for flow i.
Additional TeaVaR Input	β	The target availability level (e.g.,99.9%).
	$q \in Q$	The network state corresponding to a scenario of failed shared risk groups.
	p_q	Probability of network state q.
Auxiliary variables	s_q	The total loss in scenario q.
	$t_{i,q}$	The loss on flow i in scenario q
	$y_r(q)$	1 if tunnel x_r is available in scenario q, 0 otherwise
TE Output	b_i	The total bandwidth for flow i.
	x_r	The allocation of b_i on tunnel $r \in R_i$.
Additional TeaVaR Output	α	The "loss" (a.k.a the Value at Risk (VaR)).

$$
\begin{aligned}
\text{minimize} \quad & \alpha + \tfrac{1}{1-\beta}\Sigma_{q\in Q} p_q s_q \\
\text{subject to} \quad & \Sigma_{e\in r} x_r \le c_e && \forall e \\
& s_q \ge t_{i,q} - \alpha && \forall i, q \\
& s_q \ge 0 && \forall q \\
\text{where} \quad & t_{i,q} = 1 - \tfrac{\Sigma_{r\in R_i} x_r y_r(q)}{d_i} && \forall i, q
\end{aligned}
$$

Table 1: Key notations in the TeaVaR formulation. The *original* optimization problem is minimizing (4) subject to (2) – (3). Here, we show the derived LP formulation; see Section 4.2 and Appendix A for details.

solving this LP becomes intractable for realistic network sizes. To address this additional challenge, we introduce a pruning process that allows us to consider fewer scenarios. This substantially improves the runtime with little effect on accuracy, as shown in §5.

4 THE TEAVAR OPTIMIZATION FRAMEWORK

We now describe the TeaVaR optimization framework in detail. We first formalize the model and delineate the goals of WAN TE [27, 29, 38, 43] (§4.1). We then introduce TeaVaR's novel approach to TE, showing that it enables providing probabilistic guarantees on network throughput (§4.2).

4.1 WAN Traffic Engineering

Input. Like other WAN TE studies, we model the WAN as a directed graph $G = (V, E)$, where the vertex set V represents switches and edge set E represents links between switches. Link capacities are given by $C = (c_1, \ldots, c_{|E|})$ (e.g., in bps) and as in any TE formulation, the total flow on each link should not exceed its capacity. TE decisions are made at fixed time intervals (say, every 5 minutes [27]), based on the estimated user traffic demands for that interval. In each time epoch, there is a set of source-destination switch-pairs ("commodities" or "flows"), where each such pair i is associated with a demand d_i and a fixed set of paths (or "tunnels") $R_i \in R$ on which its traffic should be routed. TeaVaR assumes the tunnels are part of the input. In Section 5, we evaluate the impact of the tunnel selection scheme (e.g., k-shortest paths, edge-disjoint paths, oblivious-routing) on performance. Our evaluation results show that TeaVaR optimization improves the achievable utilization-availability balance for all considered tunnel-selection schemes.

Output. The output of TeaVaR consists of two parts (see Table 1): (1) the *total* bandwidth b_i that flow (source-destination pair) i is permitted to utilize (across all of its tunnels in R_i); (2) a specification for each flow i of how its allocated bandwidth b_i is split across its tunnels R_i. The bandwidth allocated on tunnel r is denoted by x_r.
Optimization goal. Previous studies of TE consider optimization goals such as maximizing total concurrent flow [7, 27, 43, 53], max-min fairness [16, 29, 49], minimizing link over-utilization [38], minimizing hop count [41], and accounting for hierarchical bandwidth allocations [37]. As formalized below, an appropriate choice for our context is selecting x_r (per-tunnel bandwidth allocations) in a manner that maximizes the well-studied maximum-concurrent-flow objective [53]. This choice of objective will enable us to maximize network throughput while achieving some notion of fairness in terms of availability across network users. In §4.2 we discuss ways to extend our framework to include other optimization objectives.

Under maximum-concurrent-flow, the goal is to maximize the value $\delta \in [0, 1]$ such that at least an δ-fraction of each flow i's demand is satisfied across all flows. For example, $\delta = 1$ implies that all demands are fully satisfied by the resulting bandwidth allocation, while $\delta = \frac{1}{3}$ implies that at least a third of each flow's demand is satisfied.

4.2 TeaVaR: TE with Probabilistic Guarantees

TeaVaR's additional inputs and outputs are listed in Table 1. Given a target availability level β, our goal is to cast TE optimization as a CVaR-minimization problem whose output is a bandwidth allocation to flows that can be materialized with probability of at least β. Doing so requires careful specification of (i) the "control" and "uncertainty" vectors, as described in §3, and (ii) a "loss function" that provides fairness and avoids starvation across network flows.
The probabilistic failure model. We consider a general failure model, consisting of a set of *failure events* Z. A failure event $z \in Z$ represents a single SRG becoming unavailable (the set of SRGs can be constructed as described in [36, 54],). Importantly, while failure events in our formulation are *uncorrelated*, this does not preclude modeling multiple links becoming concurrently unavailable in a correlated manner. Consider a failure event z representing a technical failure in a certain link l and another failure event z' representing a technical failure in a switch, or, alternatively, a power outage, which cause multiple links, including l, to become unavailable concurrently. Even though link l is inactive whether z or z' is realized, z and z' capture failures of different components and represent independent events. Thus, while there might be an overlap between the sets of links associated with two different failure events, the probabilities of the two events should still be independent if these correspond to different SRGs (e.g., the probability of a technical malfunction in a specific link and the probability of a failure in a switch incident to it in the example above).

Each failure event z occurs with probability p_z. As described earlier, the failure probabilities are obtained from historical data (see §5 for more details on failure estimation techniques, as well as sensitivity analysis of inaccuracies in these estimations). By $q = (q_1, \ldots, q_{|Z|})$, we denote a network *state*, where each element q_z is a binary random variable, indicating whether failure event z occurred ($q_z = 1$) or not. For example, for a network with 15 SRGs,

the possible set of events (Z) is the set of all Boolean vectors with 15 elements, where each element indicates whether the corresponding SRG has failed or not. For example, $\hat{q} = (0, \ldots, 0, 1)$ captures the network state in which only SRG_15 has failed. More formally, let Q be the set of all possible states, and let $p_{\hat{q}}$ denote the probability of state $\hat{q} = (\hat{q}_1, \ldots \hat{q}_{|Z|}) \in Q$. The probability of network state $\hat{q}$ can be obtained using the following equation:

$$p_{\hat{q}} = P(q_1 = \hat{q}_1, \ldots, q_{|Z|} = \hat{q}_{|Z|}) = \Pi_z \left(\hat{q}_z p_z + (1-\hat{q}_z)(1-p_z)\right). \quad (1)$$

where $\hat{q}_z \in \{0, 1\}$ for every z.

The uncertainty vector specifies which tunnels are up. We define y as a vector of size $|R|$, where R represents all possible tunnels across all flows, and each vector element y_r is a binary random variable that captures whether tunnel r is available ($y_r = 1$) or not ($y_r = 0$). This random variable depends on realizations of relevant failure events. For example, y_r will equal 0 if one of the links or switches on the tunnel is down. Since each random variable y_r is a function of the random network state q, we often use $y_r(q)$, and $y(q)$ to denote the resulting vector of random variables, though sometimes q is omitted to simplify exposition.

The control vector specifies how bandwidth is assigned to tunnels. Recall that the output x in our WAN TE formulation captures how much bandwidth is allocated to each flow on each of its tunnels. This is the control vector for our CVaR-minimization. As in TE schemes, such per-tunnel bandwidth assignment has to ensure the edge capacities are respected, i.e., satisfy the following constraint:

$$\Sigma_{e \in r} x_r \leq c_e, \quad \forall e \in E. \quad (2)$$

To account for potential failures, we allow the total allocated bandwidth per user i, $\sum_i x_{r \in R_i}$, to exceed its demand d_i.

The choice of loss function guarantees fairness across flows. We define the loss function in two steps. First, we define a loss function for each network flow. Then, we define a network-level loss as a function of the per-flow loss.

Flow-level loss function. Recall that in our TE formulation, the optimization objective is to assign the control variables x_r (per-tunnel bandwidth allocations) in a manner that maximizes the concurrent flow, i.e., maximizes the value δ for which each flow can send at least a δ-fraction of its demand. To achieve this, loss in our framework is measured in terms of the fraction of demand *not* satisfied (i.e., $1 - \delta$). Our goal thus translates into generating the per-tunnel bandwidth assignments that minimize the fraction of demand not satisfied for a specified level of availability β.

In our formulation, the maximal satisfied demand for flow i is given by $\Sigma_{r \in R_i} x_r y_r(q)$. Thus, the loss for each flow i with respect to its demand d_i is captured by $\left[1 - \frac{\Sigma_{r \in R_i} x_r y_r(q)}{d_i}\right]^+$, where $[z]^+ = \max\{z, 0\}$; note that the $[+]$ operator ensures the loss is not negative (hence, the optimization will not gain by sending more traffic than the actual demand). This notion of per-flow loss captures the loss of assigned bandwidth for a given network state q.

Network-level loss function. To achieve fairness, in terms of availability, we define the global loss function as the maximum loss across all flows; i.e.,

$$L(x, y) = \max_i \left[1 - \frac{\Sigma_{r \in R_i} x_r y_r}{d_i}\right]^+. \quad (3)$$

Although this loss function is nonlinear, we are able to transform the optimization problem into a Linear Program (LP). Details can be found in Appendix A.

Optimization formulation. To formulate the optimization objective, we introduce the mathematical definitions of VaR_β and $CVaR_\beta$. For a given loss function L, the $VaR_\beta(x)$ is defined as $V_\beta(x) = \min\{\xi \mid \psi(x, \xi) \geq \beta\}$, where $\psi(x, \xi) = P(q \mid L(x, y(q)) \leq \xi)$, and $P(q \mid L(x, y(q)) \leq \xi)$ denotes the cumulative probability mass of all network states satisfying the condition $L(x, y(q)) \leq \xi$. $CVaR_\beta$ is simply the mean of the β-tail distribution of $L(x, y)$, or put formally:

$$C_\beta(x) = \frac{1}{1-\beta} \Sigma_{L(x, y(q)) \geq V_\beta(x)} p_q L(x, y(q)).$$

Note that the definition of $CVaR_\beta$ utilizes the definition of VaR_β. To minimize $CVaR_\beta$, we define the following potential function

$$\begin{aligned} F_\beta(x, \alpha) &= \alpha + \frac{1}{1-\beta} E[[L(x, y) - \alpha]^+] \\ &= \alpha + \frac{1}{1-\beta} \Sigma_q p_q [L(x, y(q)) - \alpha]^+. \end{aligned} \quad (4)$$

The optimization goal is to minimize $F_\beta(x, \alpha)$ over $X, \mathcal{R}$, subject to (2) – (3). We leverage the following theorem, which states that by minimizing the potential function, the optimal $CVaR_\beta$ and (approximately) also the corresponding VaR_β are obtained.

Theorem 4.1. *[51] If (x^*, α^*) minimizes F_β, then not only does x^* minimize the $CVaR_\beta$ C_β over X, but also*

$$C_\beta(x^*, \alpha^*) = F_\beta(x^*, \alpha^*), \quad (5)$$

$$V_\beta(x^*) \approx \alpha^*. \quad (6)$$

The beauty of this theorem is that although the definition of $CVaR_\beta$ uses the definition of VaR_β, we do not need to work directly with the VaR_β function $V_\beta(x)$ to minimize $CVaR_\beta$. This is significant since, as mentioned above, $V_\beta(x)$ is a non-smooth function which is hard to deal with mathematically. The statement of the theorem uses the notation $\approx$ to denote that with high probability, α^* is equal to $V_\beta(x^*)$. When this is not so, α^* constitutes an upper bound on the VaR_β. The actual VaR_β can be easily obtained from α^*, as discussed in Appendix B.

From loss minimization to bandwidth allocations. The total bandwidth that flow i is permitted to utilize (across all tunnels in R_i) is given by b_i. Clearly, b_i should not exceed flow i's demand d_i to avoid needlessly wasting capacity. However, we do not add an explicit constraint for this requirement. Instead, we embed it implicitly in the loss function (3). Once the solution is computed, each flow i is given two values:

- the total allowed bandwidth, $(1 - V_\beta(x^*))$-fraction of its demand; i.e.,

$$b_i = (1 - V_\beta(x^*)) \times d_i. \quad (7)$$

- a weight assignment $\{w_r\}_{r \in R_i}$, where $w_r = \frac{x_r^*}{\Sigma_{r \in R_i} x_r^*}$.

Proportional assignment does not require global coordination upon failures and can be easily implemented in the data plane via (re-)hashing. The proportional assignment rule is not directly encoded in the CVaR minimization framework, and so traffic reassignment might result in congestion, i.e., violating constraint (2).

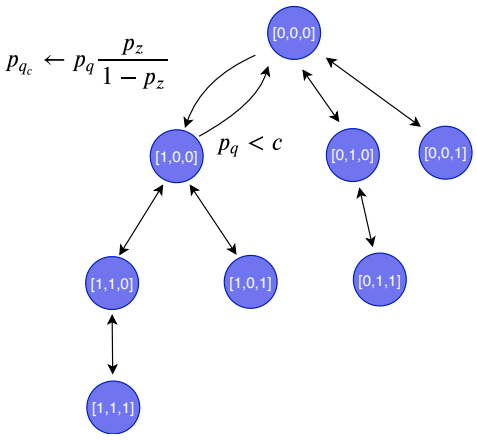


Figure 5: A diagram of the scenario-space pruning algorithm. The algorithm visits the tree nodes in a depth-first search order while updating p_q at each level. The scenario cutoff threshold, c, is set by the network operator. In our experiments, we use 10^{-5} as the cutoff threshold.

Nevertheless, it turns out that this rather simple rule guarantees such violation occurs with very low probability (upper-bounded by $1-\beta$). Formally,

Theorem 4.2. *Under* TeaVaR, *each flow i is allocated bandwidth $(1-V_\beta(x^*))d_i$, and link capacities are not exceeded with probability at least β.*

See Appendix C for the proof.

Scenario pruning. As discussed earlier, applying TeaVaR to a large network is challenging because the number of network states, which represent combinations of SRG failures, can increase exponentially with the network size. To deal with this, we devise a scenario pruning algorithm to efficiently filter out scenarios that occur with negligible probability. The main idea behind the algorithm is to use a tree representation of the different scenarios, and traverse this tree to efficiently identify every scenario q with probability p_q, that is lower than a specified *cutoff threshold*, c.

As mentioned above, the scenario pruning algorithm (see Fig. 5 for illustration) uses a tree to represent all the different failure scenarios. The root node is the scenario where no failure event occurs $[0,0,\ldots,0]$, and every child node differs from its parent by flipping a single bit from 0 to 1. The tree is constructed such that each flipped bit must be to the right of the previously flipped bit to prevent revisiting previously visited states. We assume (1) SRG failures are independent (and so failure events are independent in our model), and (2) that the probability of each failure event z is no higher than 0.5, i.e., that every SRG is more likely to not fail than to fail. Observe that, given these assumptions, the probability of a scenario decreases as the distance from the root increases. We traverse the tree in depth-first search (DFS) order until the condition $p_q < c$ is met, at which point no further scenarios down that path need to be visited. It is important to efficiently calculate the scenario probabilities while traversing the tree. Consider a child scenario q_c and a parent scenario q which differ in the bit representing event scenario z. We update the probability of q_c as follows: $p_{q_c} \leftarrow p_q \cdot \frac{p_z}{1-p_z}$. This update rule follows immediately from Eq. (1).

The pruned scenarios are used in the optimization as follows. To provide an upper bound on the CVaR (equivalently, a lower bound on the throughput), we collapse all the pruned scenarios into a single scenario with probability equal to the sum of probabilities of the pruned scenarios. We then associate a maximal loss of 1 with that scenario. In Section 5.4, we evaluate the impact of our scenario pruning algorithm on both run-time and accuracy.

Alternative loss functions. While the focus in this paper is on max-concurrent flow, our CVaR-optimization framework can incorporate other objective functions of interest. For example, we can have a loss function that corresponds to the objective of maximizing the total rate (or a weighted sum of user rates): $L_T(x,y) = \sum_i v_i[d_i - \sum_{r\in R_i} x_r y_r]^+$, where $v_i > 0$ is the priority (or weight) assigned to user i. Minimizing L_T can either replace our original loss function or be added as an additional term (e.g., the loss can be defined as $L(x,y) + \epsilon L_T(x,y)$, where ϵ is a positive constant) and still result in a linear program; we omit the details for brevity. More generally, defining loss functions of the form $L_c(x,y) = \sum_i F_i\left(\sum_{r\in R_i} x_r y_r\right)$, where F_i are convex functions, will lead to convex programs that can be solved numerically. We note that an additional post-processing step is required for these generalizations to interpret the per-user guarantees from the obtained solution; see Appendix B for details.

5 EVALUATION

In this section, we present our evaluation results for TeaVaR. We begin by describing our experimental framework and our evaluation methodology (§5.1). Our experimental results focus on the following elements:

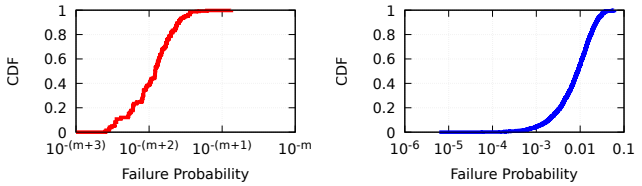
(1) Benchmarking TeaVaR's performance against the state-of-the-art TE schemes. (§5.2).
(2) Examining TeaVaR's robustness to noisy estimates of failure probabilities (§5.3).
(3) Quantifying the effect of scenario pruning on running time and the quality of the solution (§5.4).

5.1 Experimental Setting

Topologies. We evaluate TeaVaR on four network topologies: B4, IBM, ATT, and MWAN. The first three topologies (and their traffic matrices) were obtained from the authors of SMORE [38]. MWAN is short for Microsoft WAN and is derived from a subset of Azure's network topology. See Table 2 for a specification of network sizes.

Topology Name	#Nodes	#Edges
B4	12	38
IBM	18	48
ATT	25	112
MWAN	≈ 30	≈ 75

Table 2: Network topologies used in our evaluations. For confidentiality reasons we do not report exact numbers for the MWAN topology.



(a) Empirical data from MWAN (b) Weibull distribution

Figure 6: CDF of failure probabilities used in our experiments. The exact value of m in (a) is not shown for confidentiality reasons. The shape and scale parameters in (b) are 0.8 and 10^{-4}, respectively.

Our empirical data from the MWAN network consist of the following: the capacity of all links (in Gbps) and the traffic matrices (source, destination, amount of data in Mbps) over four months at a resolution of one sample per hour. For data on failure events, we collected the up/down state of each link at 15-minute granularity over the course of a year, as well as a list of possible shared risk groups. For the ATT, B4, and IBM topologies we obtained a set of at least 24 demand matrices and link capacities, but per-link failure probabilities are missing in these datasets. Hence, we use a Weibull distribution derived from MWAN measurements for these topologies. In all experiments, including the MWAN network, we use a range of scaling factors for this distribution to model networks under different failure rates.

Tunnel selection. TE schemes [27, 30, 35, 43] often use link-disjoint tunnels for each source-destination pair. However, recent work shows performance improves with the use of oblivious tunnels (interchangeably also referred to as oblivious paths) [38]. Because TeaVaR's optimization framework is orthogonal to tunnel selection, we run simulations with a variety of tunnel-selection schemes, including oblivious paths, link-disjoint paths, and k-shortest paths. As we show later in the section, TeaVaR achieves higher throughput regardless of the tunnel selection algorithm. We also study the impact of tunnel selection on TeaVaR and find that combining TeaVaR with the tunnel selection of oblivious-routing [38] leads to better performance (§5.2).

Deriving failure probability distributions. For each link e, we examine historical data and track whether e was up or down in a measured time epoch. Each epoch is a 15-minute period. We obtain a sequence of the form $(\psi_1, \psi_2, \ldots)$ such that' each ψ_t specifies whether the link was up ($\psi_t = 1$) or down ($\psi_t = 0$) during the t^{th} measured time epoch. From this sequence, another sequence $(\delta_1, \delta_2, \ldots, \delta_M)$ is derived such that δ_j is the number of consecutive time epochs the link was up prior to the j^{th} time it failed. For example, from the sequence $(\psi_1, \psi_2, \ldots, \psi_{12}) = (1, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0)$ we derive the sequence $(\delta_1, \delta_2, \delta_3) = (2, 3, 1)$ (the link was up for 2 time epochs before the first failure, 3 before the second failure, and 1 before the last failure). An unbiased estimator of the mean *uptime* is given by $U = \frac{\Sigma_{j=1}^{M} \delta_j}{M}$. We make a simplified assumption that the link up-time is drawn from a geometric distribution (i.e., the failure probability is fixed and consistent across time epochs). Then, the failure probability p_e of link e is simply the inverse of the mean uptime; that is, $p_e = \frac{1}{U}$. We note that we can use this exact analysis for other shared-risk groups, such as switches.

Figure 6(a) plots the cumulative distribution function (CDF) for the failure probability across the network links, derived by applying the above methodology to the empirical availability traces of the MWAN network. The x-axis on the plot represents the failure probability, parametrized by m. The exact value of m is not disclosed for confidentiality reasons. Nonetheless, the important takeaway from this figure is that failure probabilities of different links might differ by orders of magnitude. To accommodate the reproducibility of results, we obtain a Weibull probability distribution which fits the shape of our empirical data. The Weibull distribution, which has been used in prior study of failures in large backbones [46], is used here to model failures over time for topologies for which we do not have empirical failure data. We denote the Weibull distribution with shape parameter λ and scale parameter k by $W(\lambda, k)$. In Fig. 6(b), we plot the Weibull distribution used in our evaluation, as well as the parameters needed to generate it. Throughout our experiments, we change the shape and scale parameters of our Weibull distribution and study the impact of probability distribution on performance.

Optimization. Our optimization framework uses the Gurobi LP solver [26] and is implemented using the Julia optimization language [10].

5.2 Throughput *vs.* Availability

We examine the performance of different TE schemes with respect to both throughput and availability.

Setup. We benchmark TeaVaR against several approaches: SMORE [38], FFC [43], MaxMin (in particular, the algorithm used in B4 [16, 29]), and ECMP [20]. SMORE minimizes the maximum link utilization without explicit guarantees on availability, FFC maximizes the throughput while explicitly considering failures, MaxMin maximizes minimum bandwidth per user [16], and TeaVaR minimizes the CVaR for an input probability. When link failures occur, traffic is redistributed across tunnels according to the proportional assignment mechanism (see §4.2) without re-optimizing weights. In our evaluations, we care about *both* the granted bandwidth *and* the probabilistic availability guarantee it comes with. In TeaVaR, the probability is explicit (controlled by the β parameter in the formulation as shown in Eq. 4). In FFC, the per-user bandwidth is granted with 100% availability for scenarios with up to k-link failures. In our experiments, FFC_1 and FFC_2 refer to FFC's formulation with $k = 1$ and $k = 2$, respectively. To fairly compare the ability of the FFC algorithm to accommodate scaled-up demands, we let it send the entire demand at the expense of potential degradation in availability, unless otherwise stated.

Availability *vs.* demand scaling. We first analyze the availability achieved by various TE schemes as demand is scaled up. Given that current networks are designed with traditional worst-case assumptions about failures, all topologies are over-provisioned. Hence, we begin with the input demands, compute the availability achieved when satisfying them for different schemes, and then scale up the demands by introducing a (uniform) demand scale-up factor $s \geq 1$ by which each entry in the demand matrix is multiplied, a technique also used in prior work [38, 43].

Availability is calculated by running a post-processing simulation in which we induce failure scenarios according to their probability of occurrence and attempt to send the entirety of the demand

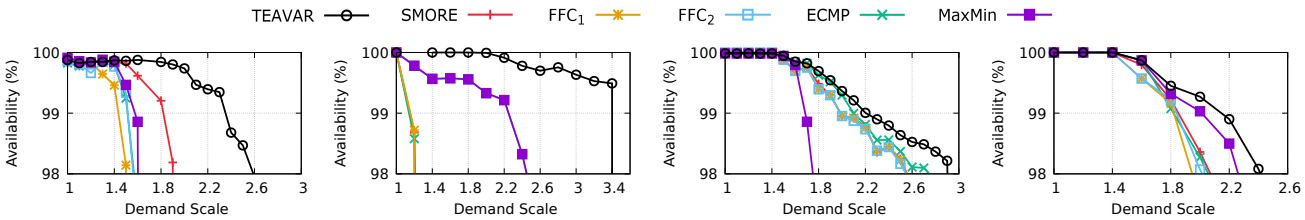


Figure 7: Comparison of TeaVaR to various TE schemes under different tunnel selection algorithms. All schemes in (a) use oblivious paths, in (b) use k-shortest paths ($k = 8$), and all schemes in (c) and (d) use edge disjoint paths. The term *availability* refers to the percentage of scenarios that meet the 100% demand-satisfaction requirement.

through the network. For each scenario we record the amount of unsatisfied demand (loss) for each flow, as well as the probability associated with that scenario. The sum of the probabilities for scenarios where demand is *fully satisfied* reflects the availability in that experiment. For example, if a TE scheme's bandwidth allocation is unable to fully satisfy demand in 0.1% of scenarios, it has an availability of 99.9%. We then scale the demand matrix and repeat the above analysis for at least 24 demand matrices per topology. In Fig. 7, we summarize the results by depicting the demand scale vs. the corresponding availability.

The results show a consistent trend: TeaVaR can support higher demand for a given availability level. In particular, for any target availability level, TeaVaR can support up to twice the demand supported by other approaches. Notably, in the MWAN topology with oblivious paths, TeaVaR is able to scale up the demand by a factor of 3.4, whereas MaxMin achieves a factor of 2.6. The remaining approaches cannot scale beyond 1.4× the original demand. In certain networks, like B4, we see less of an improvement over existing approaches. We believe that this is due to the specific structure of the network topology and its bottleneck links.

Impact of failure probabilities on availability and demand scaling. Fig. 7 illustrates TeaVaR's ability to scale up the demand matrix while maintaining high availability. To examine the gains in high-availability regions (99% and higher), we experiment with lower failure probabilities. Figure 8 shows that TeaVaR is able to scale the demand up by a factor of 3.7 even when availability is as high as 99.99%.

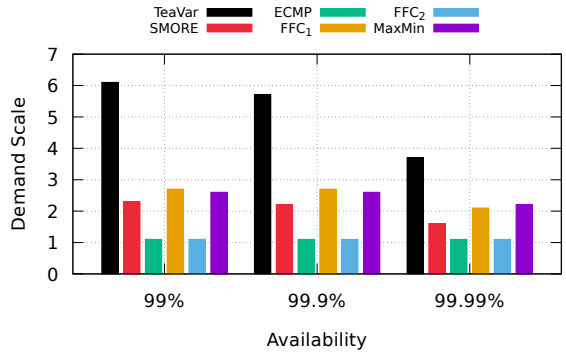


Figure 8: Comparison of TeaVaR to other TE schemes for MWAN network in the high-availability region.

Achieved throughput and tunable β. We next demonstrate the tradeoff between a target availability threshold and the achieved throughput without scaling the demand. In the previous set of experiments, availability is measured as the probability mass of scenarios in which demand is fully satisfied ("all-or-nothing" requirement). In contrast, in this set of experiments we measure the *fraction* of the total demand that can be guaranteed for a given availability target. This fraction is optimized explicitly in TeaVaR for a given value of β. For other TE schemes, we obtain the fraction of demand through a similar post-processing method as before: for each failure scenario, we simulate the outcome of sending the entire demand through the network, sort the scenarios according to loss values, and report the demand fraction at the β-percentile (i.e., the throughput is greater than or equal to that value for β percent of the scenarios). The range of availability values is chosen according to typical availability targets [28].

Fig. 9 plots the average throughput for ATT, B4, and IBM topologies. For each TE scheme, we report the results under the tunnel selection algorithm which has performed the best (SMORE, TeaVaR, and MaxMin with oblivious paths and FFC with link-disjoint paths). The results demonstrate that TeaVaR is able to achieve higher

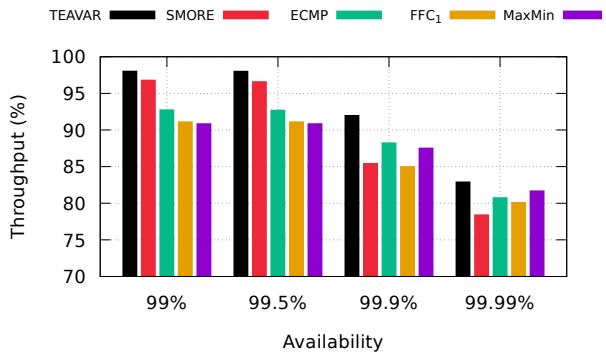


Figure 9: Averaged throughput guarantees for different β values on ATT, B4, and IBM topologies. TeaVaR is the only scheme that can explicitly optimize for a given β. For all the other schemes, the value on the x-axis is computed based on their achieved throughput.

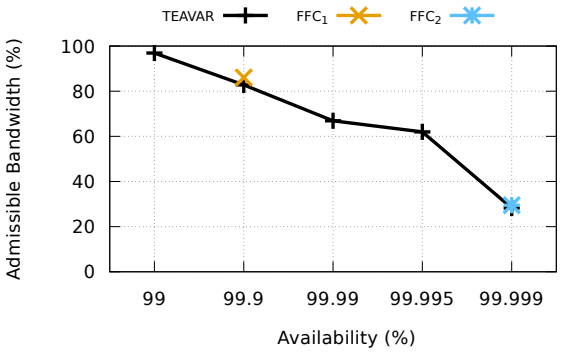


Figure 10: The admissible bandwidth of TeaVaR and FFC averaged across two topologies (IBM and B4), 10 demand matrices, and 10 different probabilities samples. TeaVaR is able to tune availability and bandwidth, while FFC's ability to balance the two is much more coarse-grained.

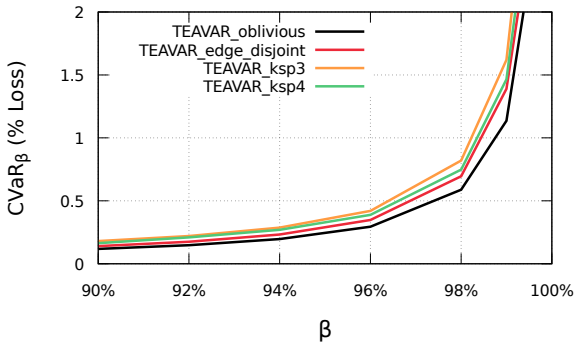


Figure 11: The effect of tunnel selection scheme on TeaVaR's performance, quantified by the resulting $CVaR_\beta$ (or loss) for different values of β. TeaVaR using oblivious paths has better performance (lower loss).

throughput for each of the target availability values because it can optimize throughput for an explicit availability target within a probabilistic model of failures.

The advantage of optimization with respect to an explicit availability threshold. We next illustrate the advantage over FFC of TeaVaR's explicit optimization of the admissible bandwidth for flows with respect to a target availability threshold (as in Eq. 7). Recall that FFC influences availability indirectly by requiring that the admissible bandwidth be supported even with up to k simultaneous link failures, for some predetermined value k. We show below that this approach is too coarse grained to strike desired utilization-availability tradeoffs. Figure 10 compares the average admissible bandwidth between FFC and TeaVaR across two topologies (B4 and IBM), 10 demand matrices, and 10 different probability samples. Note that in FFC_1 (where $k = 1$) the admissible bandwidth is nearly 80% with 99.9% availability (FFC_1 only provides guarantee with respect to a single failure, and the total probability of all failure events in which at most a single link fails is 99.9%). What if, however, the network operator is interested in achieving availability of 99.99%? Because of the limited expressiveness of FFC, achieving higher availability than FFC_1 translates to setting $k = 2$. However, while FFC_2 does indeed improve availability to 99.999% (the total probability of all failure events in which at most two links fail), as seen in the figure, this huge increase in availability comes at a dire price: the total admissible bandwidth drops to 27%. As seen in the figure, by optimizing for an explicit availability level, TeaVaR can find a sweet spot on the availability-bandwidth arc according to the operator's availability target.

FFC maximizes the admissible bandwidth, while TeaVaR's optimization also strives to achieve fairness across flows. Thus, FFC favors more bandwidth over fairness. Indeed, in the FFC_1 and FFC_2 outcomes plotted in Figure 10, some of the flows do not send traffic at all while others send their entire demands, making these extremely unfair. Yet, as the results in Figure 10 show, TeaVaR's fairness does not stand in the way of attaining admissible bandwidth comparable to FFC_1 and FFC_2 for the corresponding availability levels (99.9% and 99.999%, respectively).

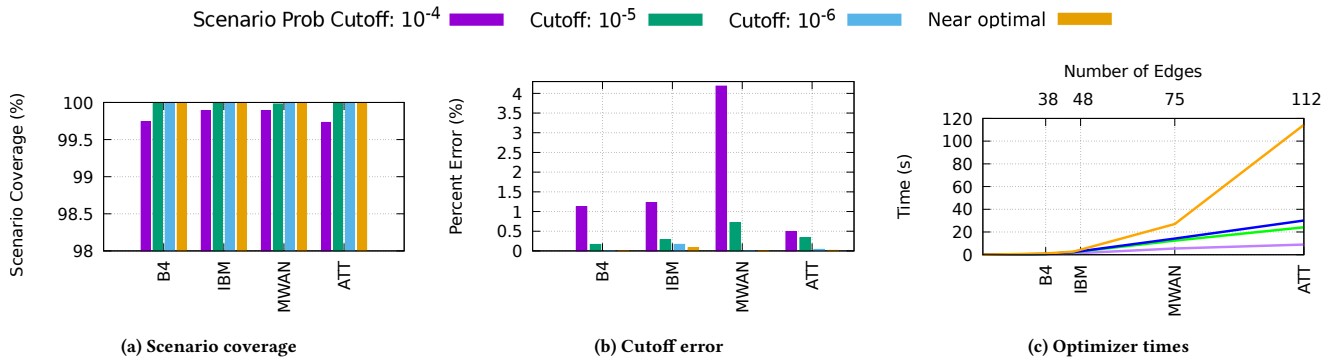
Impact of tunnel selection. So far, TeaVaR has been simulated with either link-disjoint or oblivious tunnels [38] schemes. We now analyze other tunnel selections. We demonstrate that while path selection is an important aspect of any TE scheme, no specific choice is needed for TeaVaR's success. In Fig. 11, we plot the obtained $CVaR_\beta$ as a function of β for k-shortest paths with 3 and 4 paths, FFC's link-disjoint paths, and SMORE's oblivious routing. TeaVaR performs comparably well regardless of the tunnel selection scheme. However, the figure shows that oblivious paths are still superior. For example, the obtained $CVaR_\beta$ value is at least 20% better for $\beta = 0.99$ than any other tunnel selection scheme. These results indicate that an oblivious routing tunnel selection is a good choice to complement TeaVaR. Oblivious routing is intended to avoid link over-utilization through diverse and low-stretch path selection, whereas k-shortest paths routing often yields many overlapping paths. Intuitively, these path properties are useful for TeaVaR, providing our optimization framework with a set of tunnels that, if utilized appropriately, can provide high availability.

5.3 Robustness of Probability Estimates

TeaVaR uses a probabilistic model of network failures. Probabilities of failure events in our experiments are estimated by analyzing historical time-series data of up/down status and are inherently prone to estimation errors (more sophisticated techniques, e.g., based on machine-learning, can be applied).

To examine the effects of such inaccuracies, we use the following methodology. We assume there is a set of probabilities that is the ground-truth, so the actual performance of any TE solution should be evaluated against these probabilities. In particular, we evaluate two versions of TeaVaR: (i) TeaVaR using the ground-truth probabilities; (ii) TeaVaR using a *perturbed* version of the ground-truth probabilities (reflecting estimation errors). To generate (ii), we assume a magnitude of noise n. The probabilities that are given to TeaVaR as input are generated via $\tilde{p}_z = p_z + p_z nr$, where r is random noise, distributed uniformly on $[-1, 1]$.

For each noise level in Table 3, we compare the percent error of average throughput across all scenarios to that achieved using the ground-truth probabilities. We observe that the noise has a



(a) Scenario coverage (b) Cutoff error (c) Optimizer times

Figure 12: Impact of scenario pruning on accuracy and runtime. In our evaluations we use 10^{-4} as the cutoff threshold for ATT and 10^{-5} for all other topologies. (a) Our cutoff thresholds cover more than 99.5% of all possible scenarios. (b) The error incurred by pruning scenarios is less than 5% in all cases. (c) The cutoffs we apply lead to manageable running times.

relatively small effect on the solution quality. For example, when the perturbed probabilities are within 10% from the ground truth, TeaVaR's throughput is within 3% of the solution obtained with the actual probabilities.

5.4 Sensitivity to Scenario Pruning

In Section 4.2 we described an efficient algorithm for pruning scenarios. We now elaborate on this algorithm and evaluate its impact on performance and accuracy. Our first step is to see what percentage of the scenario space is being pruned by various cutoff thresholds. The cutoff threshold is defined in Section 4. Fig. 12(a) shows the probability mass of all scenarios remaining in the optimization after a given cutoff. With modest cutoffs we are left with a large portion, over 95% of the total space. Fig. 12(b) shows the effect of pruning on accuracy. Specifically, we compare the resulting CVaR value with pruning to the case where we consider 100% of the scenario space ("optimal") using a standard error formula, $\frac{|CVaR_{\beta,cutoff} - CVaR_{\beta,optimal}|}{CVaR_{\beta,optimal}}$. With a cutoff similar to that used in the bulk of our experiments (10^{-4} for ATT and 10^{-5} for all other topologies), we achieve high scenario coverage with less than 5% error. It is important to note that the speed benefits of the cutoff are substantial, as shown in Fig. 12(c). Using cutoff values reduces the computation time from minutes (in the near-optimal case) to a few tens of seconds, a plausible compute overhead for typical TE periods of 5-15 minutes. All time complexity benchmarks were performed on a fairly standard processor (4-core, 2.60 GHz processor with 32 GB RAM).

Noise in probability estimations	% error in throughput
1%	1.43%
5%	2.95%
10%	3.07%
15%	3.95%
20%	6.73%

Table 3: The effect of inaccurate probability estimations on TeaVaR's performance. The decrease in throughout reflects running TeaVaR with the ground-truth probabilities $\{p_q\}$.

6 DISCUSSION

We now discuss certain natural extensions of our framework, elaborate on its practical use-cases, and present limitations.

Contending with demand uncertainty. We have considered the case of TE in the presence of failure events, but only for a fixed set of (empirically-derived) demands. An important part of today's TE schemes is a component that predicts traffic demands based on previously observed demands (see, e.g., [55]). State-of-the-art TE solvers apply sophisticated techniques to predict traffic demands, e.g., utilizing a combination of moving averages, decision forests, and random walks. An alternative approach would be to formulate the loss function in the joint probability space of both demands and link failures (e.g., the demand d_i in (3) is a random variable). We leave this to future work.

Estimating failure probabilities. TeaVaR takes as input the probabilities of failure events. While we use basic machinery for estimating failure probabilities from availability time series data, this is not the main focus of our paper. Refined estimation techniques may include a combination of learning failure patterns (e.g., diurnal) and additional information from the IP layer [56] or physical layer [23].

Control plane failures. While our focus is on data plane failures (including link failures caused by fiber cuts or hardware failures), our risk-aware approach can be extended to control plane failures. Indeed, resource allocation mechanisms for control plane failures reflect similar worst-case provisioning approaches [43].

Scalability. Our current solution utilizes a simple and practical technique for coping with an exponential number of network states (see §5.4). Looking forward, as networks become larger, an important future research direction will be to supplement our techniques with approaches to "state space reduction". These may include incremental solving [18], sampling [12], and clustering.

Multiple service priorities. A possible approach for increasing network utilization is sending background (scavenger) traffic over links/routes with available bandwidth. This is orthogonal to our

work: even if the provider is able to utilize the extra capacity by sending lower priority traffic, it is still required to provide availability guarantees for high-priority traffic. Intuitively, our framework allows for scaling down the provisioned capacity while maintaining adequate availability levels; low-priority traffic can still be sent at times when the network is not highly utilized. An interesting future research direction is to incorporate multiple priority classes with possibly different availability guarantees into our framework.

7 RELATED WORK

WAN traffic management. Optimizing WAN backbone traffic is a well-researched challenge. Prior work includes optimizing OSPF or IS-IS weights [22] and MPLS tunnels [19, 34], optimizing for bulk data transfers using relay nodes [39], optimizing under inaccurate knowledge of traffic demands [4, 38], and leveraging re-configurable optical devices [32, 44]. These studies focus on optimizing bandwidth allocation and disregard the possibility of failures. Recent interest in centralized TE for WANs is driven by software-defined approaches to running and optimizing such networks at scale (such as SWAN [27], B4 [29], FFC [43], and BwE [37]). These schemes exploit a global view of the network and perform global updates to configure flow allocations.

Risk management in TE *vs.* risk-management in capacity planning. Leveraging empirical data on failure probabilities to attain higher availability has been proposed in the context of capacity planning [5], i.e., the periodic augmentation of the WAN's capacity. However, capacity planning occurs on a much longer timescale than TE (months, as opposed to minutes), and this does not allow us to take into account timely information about failures (or the prevailing demands). For example, [23] establishes that outages can be predicted based on sudden drops in optical signal quality, with a 50% chance of an outage within an hour of a drop event and a 70% chance of an outage within one day. Even if capacity planning is informed by aggregated empirical statistics about failures, TeaVaR can still harness empirical data on the *current* traffic pattern and failure probabilities to optimize utilization and availability.

TE optimization *vs.* TE validation. [13] presents an optimization framework for quantifying the *worst-case* performance of an adaptive routing strategy *provided as input*. Thus, the framework in [13] can be used to *validate* that a *specific* TE configuration meets a certain performance goal under (input) variable demands and failure scenarios. Our aim, in contrast is to *optimize* the choice of TE configuration.

More risk-aware networking. [47] proposes a TE framework that takes into account demand uncertainty, as opposed to network failures in our case, and risk is expressed in terms of standard deviation. Consequently, the TE framework in [47] cannot enforce a particular level of availability (say, 99.9%). In addition, the focus in [47] is on offline TE and revenue. [11] analyzes the effects of demand fluctuations and proposes a TE scheme that adjusts the network topology in the long term and re-routes traffic in the short term. This combine ideas from oblivious routing and dynamic routing but do not consider failures. The impact of failures on availability is studied in [25], but this relationship is not formalized. Instead, [25] introduces design principles. Our findings agree with these principles. Ghobadi et al. [23] study failures in Microsoft's WAN and propose that WAN TE should take into account data on optical-layer performance, but do not provide a TE formulation addressing this. [54] achieves failure recovery by dynamically rebalancing traffic across paths after failures occur. This approach does not consider failure probabilities. [40] and [17] present routing algorithms for recovering from multiple failures in a probabilistic model of link failures. The objective there is computing diverse routes with minimum joint failure probability. These results are orthogonal and complementary to ours. Indeed, TeaVaR could be applied to tunnels computed in this manner (as with link-disjoint paths, oblivious paths, etc.). [56] presents analytical models for the dynamic estimation of failure risks and proposes accounting for risk when computing routes in OSPF networks so that service level agreement violations are minimized. Our formulation can be used in tandem with the failure model in [56], as well as other failure models from the literature, such as [46, 54].

Operations research perspective. The optimization of networks that exhibit stochastic behavior has been studied in operations research literature in different contexts (e.g., transportation, wireless networks). The set of tools used to address uncertainty includes stochastic and robust optimization [6, 9, 15, 24]. In addition, there is a rich body of literature on CVaR following the seminal work in [50]. Boginski et al. [12] apply CVaR to the minimum-cost flow problem (MCF) under uncertain link availability. Importantly, [12] introduces a CVaR-related constraint on the total loss, but since it does not lead to per-commodity availability guarantees (as in TeaVaR), it cannot be used to generate SLOs for individual network users.

8 CONCLUDING REMARKS

Inspired by financial risk theory, we introduce a novel TE paradigm that explicitly accounts for the likelihood of different failure events with the goal of minimizing a formal notion of risk to a level deemed acceptable by network operators. We design and evaluate TeaVaR, a TE optimization framework that allows operators to optimize bandwidth assignment subject to meeting a desired availability bar (e.g., providing 99.9% availability). In our design, we address algorithmic challenges related to the tractability of risk minimization in our context, as well as operational challenges. We apply TeaVaR to real-world data from the inter-datacenter backbone of a large service provider in North America. Our results reveal that TeaVaR can support up to twice as much traffic as today's state-of-the-art TE schemes at the same level of availability. TeaVaR illustrates the usefulness of adopting the notion of Conditional Value at Risk to network resource allocation challenges, and we believe that this approach might find other important applications in the networking domain.

9 ACKNOWLEDGEMENTS

We thank Hari Balakrishnan, Jeff Cox, Kimia Ghobadi, Arpit Gupta, Srikanth Kandula, Praveen Kumar, Hongqiang Liu, the anonymous SIGCOMM reviewers, and our shepherd Michael Mitzenmacher. This work was partially supported by NSF grant CNS-1563826 and the Israel Science Foundation.

REFERENCES

[1] Ian F. Akyildiz, Ahyoung Lee, Pu Wang, Min Luo, and Wu Chou. 2014. A Roadmap for Traffic Engineering in SDN-OpenFlow Networks. *Computer Networks* 71 (Oct. 2014), 1–30.

[2] Mohammad Alizadeh, Tom Edsall, Sarang Dharmapurikar, Ramanan Vaidyanathan, Kevin Chu, Andy Fingerhut, Vinh The Lam, Francis Matus, Rong Pan, Navindra Yadav, and George Varghese. 2014. CONGA: Distributed Congestion-aware Load Balancing for Datacenters. In *ACM SIGCOMM (2014)*. 503–514.

[3] Fredrik Andersson, Helmut Mausser, Dan Rosen, and Stanislav Uryasev. 2001. Credit risk optimization with conditional value-at-risk criterion. *Mathematical Programming* 89, 2 (2001), 273–291.

[4] David Applegate and Edith Cohen. 2013. Making intra-domain routing robust to changing and uncertain traffic demands: Understanding fundamental tradeoffs. In *ACM SIGCOMM (2013)*.

[5] Ajay Kumar Bangla, Alireza Ghaffarkhah, Ben Preskill, Bikash Koley, Christopher Albrecht, Emilie Danna, Joe Jiang, and Xiaoxue Zhao. 2015. Capacity planning for the Google backbone network. In *ISMP (2015)*.

[6] Ron Banner and Ariel Orda. 2007. The power of tuning: A novel approach for the efficient design of survivable networks. *IEEE/ACM TON* (2007).

[7] Cynthia Barnhart, Niranjan Krishnan, and Pamela H. Vance. 2009. Multicommodity Flow Problems. In *Encyclopedia of Optimization*. Springer, 2354–2362.

[8] Theophilus Benson, Ashok Anand, Aditya Akella, and Ming Zhang. 2011. MicroTE: Fine grained traffic engineering for data centers. In *ACM CoNEXT (2011)*.

[9] Dimitris Bertsimas and Melvyn Sim. 2003. Robust discrete optimization and network flows. *Mathematical programming* 98, 1-3 (2003), 49–71.

[10] Jeff Bezanson, Stefan Karpinski, Viral B. Shah, and Alan Edelman. 2012. Julia: A Fast Dynamic Language for Technical Computing. *CoRR* abs/1209.5145 (2012).

[11] Yingjie Bi and Ao Tang. 2019. Uncertainty-Aware optimization for Network Provisioning and Routing. In *CISS (2019)*.

[12] Vladimir L. Boginski, Clayton W. Commander, and Timofey Turko. 2009. Polynomial-time identification of robust network flows under uncertain arc failures. *Optimization Letters* 3, 3 (2009), 461–473.

[13] Yiyang Chang, Sanjay Rao, and Mohit Tawarmalani. 2017. Robust Validation of Network Designs under Uncertain Demands and Failures. *USENIX NSDI (2017)*.

[14] Antonio J. Conejo, Miguel Carrión, Juan M. Morales, et al. 2010. *Decision making under uncertainty in electricity markets*. Vol. 1. Springer.

[15] G. A. Corea and V. G. Kulkarni. 1990. Minimum Cost Routing on Stochastic Networks. *Operations Research* 38, 3 (1990), 527–536.

[16] Emilie Danna, Subhasree Mandal, and Arjun Singh. 2012. A practical algorithm for balancing the max-min fairness and throughput objectives in traffic engineering. In *IEEE INFOCOM (2012)*.

[17] Oscar Diaz, Feng Xu, Nasro Min-Allah, Mahmoud Khodeir, Min Peng, Samee Khan, and Nasir Ghani. 2012. Network Survivability for Multiple Probabilistic Failures. *IEEE Communications Letters* 16, 8 (August 2012), 1320–1323.

[18] Maxime Dufour, Stefano Paris, Jeremie Leguay, and Moez Draief. 2017. Online Bandwidth Calendaring: On-the-fly admission, scheduling, and path computation. In *IEEE ICC (2017)*.

[19] Anwar Elwalid, Cheng Jin, Steven H. Low, and Indra Widjaja. 2001. MATE: MPLS adaptive traffic engineering. In *IEEE INFOCOM (2001)*.

[20] Bernard Fortz, Jennifer Rexford, and Mikkel Thorup. 2002. Traffic engineering with traditional IP routing protocols. *IEEE Communications Magazine* 40, 10 (Oct. 2002), 118–124.

[21] Bernard Fortz and Mikkel Thorup. 2000. Internet traffic engineering by optimizing OSPF weights. In *IEEE INFOCOM (2000)*.

[22] Bernard Fortz and Mikkel Thorup. 2002. Optimizing OSPF/IS-IS weights in a changing world. *IEEE journal on selected areas in communications* 20, 4 (2002), 756–767.

[23] Monia Ghobadi and Ratul Mahajan. 2016. Optical Layer Failures in a Large Backbone. In *ACM IMC (2016)*.

[24] Gregory D. Glockner, George L. Nemhauser, and Craig A. Tovey. 2001. Dynamic Network Flow with Uncertain Arc Capacities: Decomposition Algorithm and Computational Results. *Computational Optimization and Applications* 18, 3 (1 Mar 2001), 233–250.

[25] Ramesh Govindan, Ina Minei, Mahesh Kallahalla, Bikash Koley, and Amin Vahdat. 2016. Evolve or Die: High-Availability Design Principles Drawn from Googles Network Infrastructure. In *ACM SIGCOMM (2016)*.

[26] Zonghao Gu, Edward Rothberg, and Robert Bixby. 2012. Gurobi Optimizer Reference Manual, Version 5.0. *Gurobi Optimization Inc., Houston, USA* (2012).

[27] Chi-Yao Hong, Srikanth Kandula, Ratul Mahajan, Ming Zhang, Vijay Gill, Mohan Nanduri, and Roger Wattenhofer. 2013. Achieving high utilization with software-driven WAN. In *ACM SIGCOMM (2013)*.

[28] Chi-Yao Hong, Subhasree Mandal, Mohammad Al-Fares, Min Zhu, Richard Alimi, Kondapa Naidu Bollineni, Chandan Bhagat, Sourabh Jain, Jay Kaimal, Shiyu Liang, Kirill Mendelev, Steve Padgett, Faro Rabe, Saikat Ray, Malveeka Tewari, Matt Tierney, Monika Zahn, Jonathan Zolla, Joon Ong, and Amin Vahdat. 2018. B4 and After: Managing Hierarchy, Partitioning, and Asymmetry for Availability and Scale in Google's Software-defined WAN. In *ACM SIGCOMM (2018)*.

[29] Sushant. Jain, Alok Kumar, Subhasree Mandal, Joon Ong, Leon Poutievski, Arjun Singh, Subbaiah Venkata, Jim Wanderer, Junlan Zhou, Min Zhu, Jonathan Zolla, Urs Hölzle, Stephen Stuart, and Amin Vahdat. 2013. B4: Experience with a Globally-deployed Software Defined WAN. In *ACM SIGCOMM (2013)*.

[30] Virajith Jalaparti, Ivan Bliznets, Srikanth Kandula, Brendan Lucier, and Ishai Menache. 2016. Dynamic pricing and traffic engineering for timely inter-datacenter transfers. In *ACM SIGCOMM (2016)*.

[31] Wenjie Jiang, Rui Zhang-Shen, Jennifer Rexford, and Mung Chiang. 2009. Cooperative content distribution and traffic engineering in an ISP network. In *ACM SIGMETRICS (2009)*.

[32] Xin Jin, Yiran Li, Da Wei, Siming Li, Jie Gao, Lei Xu, Guangzhi Li, Wei Xu, and Jennifer Rexford. 2016. Optimizing Bulk Transfers with Software-Defined Optical WAN. In *ACM SIGCOMM (2016)*.

[33] Philippe Jorion. 2001. *Value at Risk: The New Benchmark for Managing Financial Risk*. McGraw-Hill. `https://books.google.com/books?id=S2SsFblvUdMC`

[34] Srikanth Kandula, Dina Katabi, Bruce Davie, and Anna Charny. 2005. Walking the tightrope: Responsive yet stable traffic engineering. In *ACM SIGCOMM (2005)*.

[35] Srikanth Kandula, Ishai Menache, Roy Schwartz, and Spandana R. Babbula. 2014. Calendaring for wide area networks, Fabián E. Bustamante, Y. Charlie Hu, Arvind Krishnamurthy, and Sylvia Ratnasamy (Eds.). *ACM SIGCOMM (2014)*.

[36] Fernando A. Kuipers. 2012. An Overview of Algorithms for Network Survivability. *ISRN Communications and Networking* 2012 (Jan. 2012), 24.

[37] Alok Kumar, Sushant Jain, Uday Naik, Nikhil Kasinadhuni, Enrique C. Zermeno, C. Stephen Gunn, Jing Ai, Björn Carlin, Mihai Amarandei-Stavila, Mathieu Robin, Aspi Siganporia, Stephen Stuart, and Amin Vahdat. 2015. BwE: Flexible, Hierarchical Bandwidth Allocation for WAN Distributed Computing. In *ACM SIGCOMM (2015)*.

[38] Praveen Kumar, Yang Yuan, Chris Yu, Nate Foster, Robert Kleinberg, Petr Lapukhov, Ciun L. Lim, and Robert Soulé. 2018. Semi-Oblivious Traffic Engineering: The Road Not Taken. In *USENIX NSDI (2018)*.

[39] Nikolaos Laoutaris, Michael Sirivianos, Xiaoyuan Yang, and Pablo Rodriguez. 2011. Inter-datacenter Bulk Transfers with Netstitcher. In *ACM SIGCOMM (2011)*.

[40] Hyang-Won Lee, Eytan Modiano, and Kayi Lee. 2010. Diverse routing in networks with probabilistic failures. *IEEE/ACM TON* 18, 6 (2010), 1895–1907.

[41] Youngseok Lee, Yongho Seok, Yanghee Choi, and Changhoon Kim. 2002. A constrained multipath traffic engineering scheme for MPLS networks. In *ICC (2002)*. IEEE, 2431–2436.

[42] George Leopold. 2017. Building Express Backbone: Facebook's new long-haul network. `http://code.facebook.com/posts/1782709872057497/`. (2017).

[43] Hongqiang Harry Liu, Srikanth Kandula, Ratul Mahajan, Ming Zhang, and David Gelernter. 2014. Traffic engineering with forward fault correction. In *ACM SIGCOMM (2014)*.

[44] Ajay Mahimkar, Angela Chiu, Robert Doverspike, Mark D. Feuer, Peter Magill, Emmanuil Mavrogiorgis, Jorge Pastor, Sheryl L. Woodward, and Jennifer Yates. 2011. Bandwidth on Demand for Inter-data Center Communication. In *ACM HotNets (2011)*.

[45] Houra Mahmoudzadeh. 2015. *Robust Optimization Methods for Breast Cancer Radiation Therapy*. Ph.D. Dissertation. University of Toronto.

[46] Athina Markopoulou, Gianluca Iannaccone, Supratik Bhattacharyya, Chen N. Chuah, Yashar Ganjali, and Christophe Diot. 2008. Characterization of Failures in an Operational IP Backbone Network. *IEEE/ACM TON* 16, 4 (Aug 2008), 749–762.

[47] Debasis Mitra and Qiong Wang. 2005. Stochastic traffic engineering for demand uncertainty and risk-aware network revenue management. *IEEE/ACM TON* 13, 2 (2005), 221–233.

[48] Jeffrey C. Mogul, Rebecca Isaacs, and Brent Welch. 2017. Thinking about Availability in Large Service Infrastructures. In *ACM HotOS (2017)*.

[49] Dritan Nace and Michal Pióro. 2008. Max-min fairness and its applications to routing and load-balancing in communication networks: A tutorial. *IEEE Communications Surveys and Tutorials* 10, 1-4 (2008), 5–17.

[50] R. Tyrrell Rockafellar and Stanislav Uryasev. 2000. Optimization of conditional value-at-risk. *Journal of risk* 2 (2000), 21–42.

[51] R. Tyrrell Rockafellar and Stanislav Uryasev. 2002. Conditional value-at-risk for general loss distributions. *Journal of banking & finance* 26, 7 (2002), 1443–1471.

[52] Sergey Sarykalin, Gaia Serraino, and Stan Uryasev. 2008. Value-at-Risk vs. Conditional Value-at-Risk in Risk Management and Optimization.

[53] Farhad Shahrokhi and David W. Matula. 1990. The Maximum Concurrent Flow Problem. *ACM* 37 (1990), 318–334.

[54] Martin Suchara, Dahai Xu, Robert Doverspike, David Johnson, and Jennifer Rexford. 2011. Network Architecture for Joint Failure Recovery and Traffic Engineering. In *ACM SIGMETRICS (2011)*.

[55] Paul Tune and Matthew Roughan. 2017. Controlled Synthesis of Traffic Matrices. *IEEE/ACM TON* (2017).

[56] Bruno Vidalenc, Laurent Ciavaglia, Ludovic Noirie, and Eric Renault. 2013. Dynamic risk-aware routing for OSPF networks. In *IFIP/IEEE IM (2013)*. 226–234.

[57] Hong Zhang, Kai Chen, Wei Bai, Dongsu Han, Chen Tian, Hao Wang, Haibing Guan, and Ming Zhang. 2017. Guaranteeing deadlines for inter-data center transfers. *IEEE/ACM TON* (2017).

APPENDIX

Appendices are supporting material that has not been peer reviewed.

A LP FORMULATION

We formulate the CVaR minimization problem as a Linear Problem and prove its correctness.

Recall that our loss function is given by

$$L(x,y) = \max_i \left[1 - \frac{\Sigma_{r\in R_i} x_r y_r}{d_i}\right]^+. \tag{8}$$

We are interested in minimizing

$$F_\beta(x,\alpha) = \alpha + \frac{1}{1-\beta} E[\max\{0, L(x,y) - \alpha\}] \tag{9}$$

$$= \alpha + \frac{1}{1-\beta}\Sigma_q p_y [L(x, y(q)) - \alpha]^+, \tag{10}$$

subject to the link capacity constraints

$$\Sigma_{e\in r} x_r \le c_e \ \forall e. \tag{11}$$

Note that the latter constraint is linear (hence, we ignore it in the sequel for brevity). In what follows, we "linearize" the objective function by adding additional (linear) constraints.

We introduce a new set of variables $s = \{s_q\}$ and rewrite the objective function as

$$\tilde{F}_\beta(s,\alpha) = \alpha + \frac{1}{1-\beta}\Sigma_q p_q s_q, \tag{12}$$

and add the following constraints

$$s_q \ge L(x, y(q)) - \alpha \quad \forall q \tag{13}$$

$$s_q \ge 0. \quad \forall q \tag{14}$$

Observe that minimizing F w.r.t. x and α is equivalent to minimizing $\tilde{F}$ w.r.t. s, x, α. To establish this, assume by contradiction that there exists a q for which the two inequalities (13)–(14) are held in the strict sense (>) at optimality; hence, the max operation is not enforced as in the original objective. However, because p_q is positive and s_q is positive, we can decrease s_q by some small $\epsilon > 0$, and decrease the objective value. This contradicts the optimality of the solution.

To complete the construction, we replace (13) (because $L(\cdot,\cdot)$ involves two max operators). This is achieved by rewriting (13) as $s_q + \alpha \ge L(x, y(q))$. Now we can materialize the max operators through the following inequalities

$$s_q + \alpha \ge 0, \quad \forall q \tag{15}$$

$$s_q + \alpha \ge t_{i,q} \quad \forall i, q, \tag{16}$$

where

$$t_{i,q} = 1 - \frac{\Sigma_{r\in R_i} x_r y_r(q)}{d_i}. \tag{17}$$

We end up with an LP with decision variables $x, \alpha, s, \{t_{i,q}\}$ where s and $\{t_{i,q}\}$ can be viewed as auxiliary variables. The objective of the LP is to minimize (12) subject to (11), (14)–(17).

B CALCULATING VaR_β

We describe a post-processing procedure for calculating the VaR_β. We emphasize that this procedure is essentially not needed in our current formulation (details below); nonetheless, we present it here for completeness, as some of the extensions we highlight in Section 6 may require it.

We note that the procedure is generic, and applicable to any setting with a discrete number of states (recall that in our case, each network state corresponds to a different combination of links, switches, etc. that are up or down). Fixing x, we sort the states in increasing order of their loss. With some abuse of notations, we enumerate the states according to the sorted order. Let the corresponding state losses be $\ell_1 \le \ell_2 \le \cdots \le \ell_K$. Let K_β be the unique index such that $\Sigma_{k=1}^{K_\beta} P_k \ge \beta > \Sigma_{k=1}^{K_\beta - 1} P_k$, where P_k is the probability of state k. Then the VaR_β is given by $V_\beta(x) = \ell_{K_\beta}$. See Proposition 8 in [51] for a proof.

We next describe when and how this procedure is used.

Extract the VaR_β in the rare case where $V_\beta(x^*) < \alpha^*$ (see Theorem 4.1). We get this inequality in the case where $\Sigma_{k=1}^{K_\beta} P_k = \beta$. This is unlikely to occur because the probabilities correspond to empirical measures of up and down time (hence, they are often numbers with several more decimal positions than β). But even if this case occurs, we have $V_\beta(x) = \ell_{K_\beta}$ as described above.

Determining the VaR_β for individual user SLAs. As described in Section 6, we need an extra post-processing step in some extensions (e.g., maximizing total bandwidth). Unlike in our current formulation, the per-user SLA (allowed demand and the corresponding probabilistic guarantee) is not explicitly obtained as an output of the optimization framework. Therefore, we apply the above procedure on each individual user as follows. The key observation is that the tunnel allocation per user is given from the optimization. We also know the demand d_i. For each user i, we first reduce the dimension of the state-space to include events that correspond only to links, switches, etc. belonging to one or more of its routes. We sort the states and obtain the VaR_β as described above; by using this procedure, the network provider can actually give different probabilistic guarantees for different users, e.g., by fixing a different β for each user.

C PROOF OF THEOREM 4.2

Since the theorem's guarantee is with probability greater or equal to β, by definition of VaR_β, we may restrict our attention to failure states whose loss is less than or equal to $V_\beta(x^*)$. Consider any such state; let $\bar{R}_i \subseteq R_i$ be the subset of i's tunnels that are available in that state. Since $V_\beta(x^*)$ is an upper bound for each loss scenario up to the β percentile, it follows from the definition of the loss function (3) that

$$\Sigma_{r\in\bar{R}_i} x_r^* \ge \big(1 - V_\beta(x^*)\big) d_i.$$

Using this inequality together with the proportional assignment rule, we obtain that each active tunnel $r \in \bar{R}_i$ carries flow f_r,

satisfying

$$f_r = \frac{w_r}{\Sigma_{r' \in \bar{R}_i} w'_r} b_i \tag{18}$$

$$= \frac{x^*_r}{\Sigma_{r' \in \bar{R}_i} x^*_{r'}} \big(1 - V_\beta(x^*)\big) d_i \tag{19}$$

$$\leq \frac{x^*_r}{\big(1 - V_\beta(x^*) d_i\big)} \big(1 - V_\beta(x^*)\big) d_i = x^*_r. \tag{20}$$

The theorem then follows immediately by recalling that each feasible solution satisfies the capacity constraint (2).