Congestion-Constrained Virtual Link Embedding with Uncertain Demands

Fatemeh Hosseini, Alexander James, and Majid Ghaderi
Department of Computer Science
University of Calgary
Email: {fatemeh.hosseini, alexander.james, mghaderi}@ucalgary.ca

***Abstract*—Network virtualization enables multiple virtual networks to co-exist on the same physical network. Each virtual network requires specific amounts of physical network resources such as node processing and link bandwidth. The problem of mapping virtual resource requirements to physical resources is extensively studied in the literature under the assumption that resource demands of virtual networks are known deterministically. In real deployments though, resource demands include significant uncertainty and fluctuate over time. This paper considers the problem of mapping virtual links to physical network paths subject to a constraint on each virtual link congestion probability under the assumption that bandwidth demands of virtual links are uncertain. A general uncertainty model is considered, where bandwidth demands are described by random variables for which only the mean and variance (or a range) are known. We formulate the problem as a nonlinear optimization problem, which is shown to be non-convex. Consequently, we develop an approximate formulation that results in a second-order cone program (SOCP) that can be solved efficiently even for large networks. We then provide simulation as well as Mininet experimental results to show the utility and efficiency of our exact and approximate models in various network scenarios. We apply our models to commonly studied USA and EON networks as well as randomly generated large networks. Our results show that both models are able to satisfy the link congestion constraint, and that the approximate model is very close to the exact model.**

***Index Terms*—Virtual networks, Uncertain demands, Bandwidth allocation**

I. Introduction

A. Background and Motivation

Network virtualization has emerged as one of the key technologies of future networks. In its general form, network virtualization enables multiple virtual networks (VNs) to co-exist on the same physical or substrate network through specific abstraction and isolation mechanisms. To operate efficiently and reliably, each virtual network demands specific amounts of physical network resources, *e.g.*, link bandwidth. This requires resource allocation algorithms that can efficiently share physical network resources among virtual networks. Specifically, the virtual nodes and virtual links that interconnect them have to be mapped to physical resources in the substrate network.

The problem of mapping virtual resource requirements to physical resources is known as the virtual network embedding (VNE) problem. The VNE problem is NP-hard and has been the subject of extensive research (see [1] for an extensive survey on the topic). A VN is represented by a set of virtual nodes and virtual links. Virtual nodes and links require some amount of resources, *i.e.*, processing power and bandwidth, which depend on the services provided by the corresponding VNs. The VNE problem is then to find a virtual to physical node-to-node and link-to-path mapping that does not exceed the node and link capacities of the physical network. Often a measure is defined to assess the quality of a mapping, *e.g.*, the cost or revenue of the mapping. The goal of VNE is to find a feasible mapping that is optimal with respect to the defined measure.

Most of the existing works on VNE assume that the resource demands of VNs are known *deterministically* [1], *i.e.*, the resource demands are fixed and known a priori by the mapping algorithm. In a real deployment though, the node and link demands (*i.e.*, processing power and bandwidth) include significant *uncertainty* (*e.g.*, because of estimation errors or variability over time) [2]–[5]. As such, when employing deterministic embedding algorithms, one has to consider either the "worst-case" or "average" resource demands for each VN. Both approaches, however, lead to inefficient use of network resources. The worst-case approach results in significant underutilization of physical resources, while the average approach results in under or over utilization depending on whether the actual demands are higher or lower than their presumed averages.

There are recently a few works on VNE that consider uncertainty in resource demands (*e.g.*, see [3]–[7]). These works either assume that the full statistical distributions of resource demands are known (*e.g.*, demands follow a Normal distribution with known parameters) [5], [6], or resort to robust optimization models where only limited information about resource demands is assumed (*e.g.*, demands fall within a given range) [3], [4], [7]. Their objective is to find a virtual-to-physical mapping that is feasible even when resource demands deviate from their nominal values, without sacrificing the utilization of physical resources. In either case, the embedding algorithm computes a mapping that satisfies virtual resource demands *probabilistically*. In other words, the computed mapping may not be feasible in certain scenarios, *e.g.*, when all resource demands deviate significantly from their nominal values. Consequently, a "congestion" happens at a node or link where the allocated physical resources are insufficient to meet the virtual resource demands. Congestions are detrimental to the performance of VNs, and hence it is critical to take them into consideration when computing a virtual-to-physical resource mapping under demand uncertainty.

In this paper, we consider the VNE problem under demand uncertainty with guaranteed congestion probability. Specifically, we focus on mapping virtual links to physical paths, where a pre-specified target on congestion probability is guaranteed for each virtual link. In the context of link-to-path mapping, guaranteeing a congestion probability for a virtual link reduces to guaranteeing the end-to-end congestion probability on the corresponding path. We show that under this constraint, the VNE problem becomes considerably different and more challenging compared to existing formulations without end-to-end congestion guarantee. Our objective is to design a bandwidth allocation algorithm that considers uncertainty in resource demands, provides guaranteed end-to-end congestion probability, and is fast enough to be applied to large-scale network virtualization.

B. Related Work

In the following, we briefly review several related works focusing on those that consider demand uncertainty.

Deterministic VNE. This category includes works that assume the resource demands of VNs are fixed and known in advance. This version of the problem, *i.e.*, the deterministic VNE problem, is extensively studied in the literature (see [1] for a survey on this topic). Generally, there are two approaches in dealing with the VNE problem in this setting. First approach is based on modeling the problem as an Integer Linear Program (see, among others, [8], [9]). Solving the resulting integer program, which is NP-hard in general, for large networks is very time consuming. Thus, the second approach focuses on heuristic solutions (see, among others, [10], [11]) that can tackle large networks and yet provide solutions that are reasonably efficient.

Stochastic VNE. This category includes works that assume the distribution of bandwidth demands is fully known. Their roots can be traced back to the literature on the effective bandwidth concept [12]. For instance, [6] and [5] assume that demands follow a Normal distribution and devise algorithms to solve the VNE problem accordingly. The congestion probability of each physical link can be computed using the tail probability of the Normal distribution. Wang *et al.* [13] also assume Normally distributed demands, but consider an online setting for the VNE problem in which VN requests arrive one-by-one over time. The work presented in [14] adopts a different approach by employing a seasonal ARMA model to estimate the unknown future demands. Then, to compute the link congestion probability, the authors assume that the prediction errors follow a zero-mean Normal distribution for which the variance can be estimated from past demand observations.

Robust VNE. This category includes works that apply robust optimization techniques to solve the VNE problem by assuming that, while the distribution of demands is unknown, some uncertainty model can be used to describe their variability. Heckmann *et al.* [15] consider a model where the demand uncertainty is described by a number of scenarios. Each scenario corresponds with a specific set of resource demands. By considering different scenarios, the authors can account for various bandwidth allocation strategies ranging from average to worst-case allocation. Lee *et al.* [7] formulated the VNE problem as a robust optimization problem, where demand uncertainty is described by a range (the so-called Box Uncertainty model [16]). Their objective is to guarantee a congestion level for each physical link independently of other links. They convert the

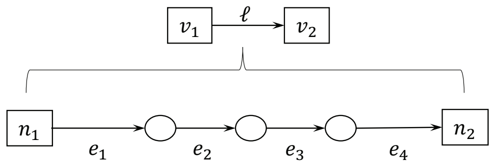


Fig. 1: Virtual link to physical path mapping.

problem to a chance constrained optimization problem [16], which can be solved efficiently. Coniglio *et al.* [3], [4] also model the VNE problem as a robust optimization problem following the framework of Γ-Robustness [17]. In this approach, the level of uncertainty in demands can be controlled by the parameter Γ, which determines how many demands can deviate from their nominal values simultaneously.

As mentioned earlier, none of the above works can guarantee the congestion probability experienced by a virtual link, as they consider congestion on each physical link independently from other links on the end-to-end path that maps to the virtual link.

C. Our Work

Recall that a virtual link is mapped to a physical path. Thus, to guarantee a given congestion level for a virtual link, the end-to-end congestion on the corresponding physical path must be bounded. The stochastic and robust works described above, however, only guarantee that the probability of congestion on each physical link is bounded. Therefore, the congestion probability achieved for the corresponding virtual link depends on the structure of the underlying path, and hence is not guaranteed.

Consider the simple example depicted in Fig. 1, in which the virtual link ℓ between virtual nodes v_1 and v_2 is mapped to path $P = \langle e_1, e_2, e_3, e_4 \rangle$ in the physical network (*i.e.*, v_1 and v_2 are placed on n_1 and n_2). Suppose that the pre-specified target congestion probability for the virtual link ℓ is given by ϵ. Let ε_k denote the congestion probability on physical link e_k. In order to satisfy the virtual link congestion requirement ϵ, the following constraint should be satisfied,

$$1 - \prod_{e_k \in P}(1 - \varepsilon_k) \leq \epsilon \,. \tag{1}$$

Notice that ε_k is a function of the total resource demand on link e_k, and should be (optimally) determined by the embedding algorithm. The complexity of the problem arises from the fact that 1) the optimal path connecting the physical nodes n_1 and n_2 is not known in advance, and 2) the bandwidth demand on each link, and consequently the link congestion probabilities are functions of the unknown paths. In other words, the problem is a joint optimization of path selection and link congestion allocation, which we show results in a non-convex nonlinear optimization problem that is computationally intractable. The above mentioned works assume that it is sufficient to guarantee a fixed congestion probability ε for every physical link e_k. As such, the actual end-to-end congestion probability achieved by these works depends on the length of the physical path, and hence cannot be guaranteed.

Our contributions in this work can be summarized as follows:

- We consider demand uncertainty in a flexible and general model, where only limited information about resource demands, namely mean and variance, is needed.

TABLE I: Principal Notation Used in the Paper.

Symbol	Definition
Input parameters	
$\mathcal{N}$	Set of physical nodes in the substrate network
$\mathcal{E}$	Set of physical links in the substrate network
$\mathcal{L}$	Set of virtual links from all virtual networks
ℓ_i	Virtual link i ($\ell_i \in \mathcal{L}$)
e_k	Physical link k ($e_k \in \mathcal{E}$)
$O(\ell_i)$	Physical origin node of ℓ_i
$D(\ell_i)$	Physical destination node of ℓ_i
C_k	Capacity of physical link e_k
B_i	Bandwidth demand of virtual link ℓ_i
$\mathcal{P}_i$	Set of candidate physical paths for $\ell_i \in \mathcal{L}$
$\mathcal{P}$	Set of all candidate paths in substrate network
$\lvert P_j \rvert$	Length of path $P_j \in \mathcal{P}$
ϵ	Required congestion probability on virtual links
Decision variables	
$0 \le x_{ij} \le 1$	Fraction of demand B_i on path $P_j \in \mathcal{P}_i$
$0 \le y_{ik} \le 1$	Total fraction of demand B_i on link $e_k \in \mathcal{E}$
$0 \le \varepsilon_k \le 1$	Congestion probability on link $e_k \in \mathcal{E}$
$\alpha \ge 0$	Utilization of the most congested physical link

- We formulate the link mapping problem with constrained end-to-end congestion probability as a non-linear optimization problem that can be solved using global optimization solvers for small network instances.
- We propose an approximate link mapping solution for the problem, which is formulated as a second-order cone program (SOCP) that can be solved efficiently even for large network instances.
- We present simulation as well as Mininet [18] experimental results to show the efficiency and utility of our solutions in both small and large network scenarios.

D. Paper Organization

The remainder of this paper is organized as follows. We discuss our model and assumptions in Section II. Exact and approximate problem formulations are presented in Section III. Performance evaluation results are presented in Section IV. Section V provides some concluding remarks.

II. System Model and Assumptions

To cope with the complexity of the problem, most existing works on VNE decompose the problem into two sub-problems in which node and link mappings are performed sequentially [1], [3]. In this work, we focus on link mapping in VNE to avoid cluttering our model and algorithms. Thus, we assume that the placement of virtual nodes in the substrate network is already determined and focus on mapping virtual links to physical paths, which we refer to as the *virtual link embedding (VLE)* problem. Our framework, however, can be included in any node placement algorithm (such as the ones proposed in [5], [19] for datacenters, or [11] for general networks) for a complete VNE solution. Table I lists the principal notation used throughout the paper.

A. Physical Network

There is a single physical network specified by an undirected graph $G = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ denotes the set of physical nodes and $\mathcal{E}$ denotes the set of physical links (or edges) between the nodes. Each physical link e_k has a fixed capacity denoted by $C_k > 0$.

B. Virtual Network Requests

Setting up virtual networks takes time. Therefore, we assume that virtual network (VN) requests are processed in batches. Each batch corresponds to requests that have arrived during the previous time interval. Our link mapping algorithm is run over each batch in an offline manner. Each virtual network request is represented as a weighted undirected graph with a given set of nodes and links. The weight of each link indicates the (uncertain) bandwidth demand of that link. Each virtual link requires its congestion probability to be bounded by some target ϵ, for $0 \le \epsilon \le 1$.

C. Virtual Link Embedding

Let $\mathcal{L}$ denote the set of the virtual links of all VNs in a batch. Let $O(\ell_i) \in \mathcal{N}$ and $D(\ell_i) \in \mathcal{N}$, for virtual link ℓ_i, denote the physical nodes embedding the origin and destination of virtual link ℓ_i, respectively. The VLE problem is to map every virtual link $\ell_i \in \mathcal{L}$ to a set of physical paths connecting $O(\ell_i)$ to $D(\ell_i)$ in the substrate network (*i.e.*, we allow multi-path routing) so that the virtual link ℓ_i satisfies the target congestion probability ϵ.

D. Bandwidth Demand Uncertainty Model

Let B_i denote the (uncertain) bandwidth demand of virtual link $\ell_i \in \mathcal{L}$. We assume that only limited statistical information about B_i is available. Specifically, we assume that the mean and variance of B_i, denoted by $\mathbb{E}[B_i] = \mu_i$ and $\mathrm{Var}[B_i] = \sigma_i^2$, are known. It is relatively straightforward to estimate the mean and variance based on historical traffic data [20]. In fact, as long as σ_i^2 provides an upper bound on the actual variance of B_i, our formulation holds.

We emphasize that the considered uncertainty model is quite general. For example, in the literature on Robust Optimization (RO), a common uncertainty model is the so-called Box Uncertainty model [16]. In this model, each uncertain variable B_i is allowed to deviate from its nominal value by a maximum deviation Δ_i. That is, $B_i \in [\mu_i - \Delta_i, \mu_i + \Delta_i]$. Our uncertainty model can easily accommodate the Box Uncertainty model by computing an upper bound on the variance of an arbitrary random variable that is confined to interval $[\mu_i - \Delta_i, \mu_i + \Delta_i]$. In this case, it can be shown that $\sigma_i^2 = \Delta_i^2$, which is attained when all the probability mass is assigned to the extreme points of the interval. Alternatively, a less conservative approach is to assume B_i is uniformly distributed over the interval, which leads to $\sigma_i^2 = \Delta_i^2/3$.

E. Congestion Probability

Let W_{ik} denote the bandwidth demand of virtual link $\ell_i \in \mathcal{L}$ on physical link $e_k \in \mathcal{E}$, where $\mathbb{E}[W_{ik}] = \mu_{ik}$ and $\mathrm{Var}[W_{ik}] = \sigma_{ik}^2$. Note that W_{ik} has to be determined by our embedding algorithm based on bandwidth demands of all virtual links (*i.e.*, B_i's) and capacities of all physical links (*i.e.*, C_k's). The congestion probability on physical link e_k is then given by,

$$\mathbb{P}\{\text{congestion on } e_k\} = \mathbb{P}\left\{\sum_{\ell_i \in \mathcal{L}} W_{ik} \ge C_k\right\}, \forall e_k \in \mathcal{E}\,. \quad (2)$$

For simplicity of notation, in the following derivation, we abbreviate the summation index and use i in place of $\ell_i \in \mathcal{L}$. To avoid making any assumptions about the distribution of the uncertain bandwidth demands, we use a *concentration bound* to estimate (2). Specifically, we use the Chernoff bound [21], as follows:

$$\mathbb{P}\left\{\sum_i W_{ik} \geq C_k\right\} \leq \inf_{\theta\geq 0} \frac{\mathbb{E}\left[e^{\theta \sum_i W_{ik}}\right]}{e^{\theta C_k}}\,. \tag{3}$$

If $\operatorname{Var}[W_{ik}]$ is bounded, *i.e.*, $\operatorname{Var}[W_{ik}] \leq \sigma_{ik}^2$, then we have [22],

$$\mathbb{E}\left[e^{\theta W_{ik}}\right] \leq e^{\mu_{ik}\theta + \frac{1}{2}\sigma_{ik}^2\theta^2}\,. \tag{4}$$

By minimizing over θ, and noting that W_{ik}'s are independent from each other, it is obtained that,

$$\mathbb{P}\left\{\sum_i W_{ik} \geq C_k\right\} \leq \exp\left(-\frac{(\sum_i W_{ik} - \sum_i \mu_{ik})^2}{2\sum_i \sigma_{ik}^2}\right)\,. \tag{5}$$

III. Exact and Approximate VLE

We assume that each virtual link $\ell_i \in \mathcal{L}$ can be mapped to multiple paths from the set of candidate physical paths $\mathcal{P}_i$. In other words, $\mathcal{P}_i$ consists of multiple paths, *e.g.*, first K shortest paths, between origin and destination nodes $O(\ell_i)$ and $D(\ell_i)$. Denote the set of all candidate paths in the substrate network by $\mathcal{P}$, that is $\mathcal{P} = \cup_{\ell_i\in\mathcal{L}}\mathcal{P}_i$. Let x_{ij}, for $0 \leq x_{ij} \leq 1$, denote the fraction of bandwidth demand B_i that is allocated on path $P_j \in \mathcal{P}_i$ for virtual link ℓ_i. Our goal is to find the routing variables x_{ij} that minimize the utilization of the most congested link in the substrate network subject to a constraint on the congestion probability on ℓ_i. Let α denote the utilization of the most congested link. The virtual link embedding problem is feasible only if $\alpha \leq 1$.

A. Congestion on Physical Links

Let y_{ik}, for $0 \leq y_{ik} \leq 1$, denote the total fraction of bandwidth demand B_i for virtual link $\ell_i \in \mathcal{L}$ that is allocated on physical link $e_k \in \mathcal{E}$. We have,

$$y_{ik} = \textstyle\sum_{P_j\in\mathcal{P}_i} x_{ij} \cdot \mathbf{I}_{e_k\in P_j}\,, \tag{6}$$

where $\mathbf{I}_{e_k\in P_j}$ denotes the indicator function, which is 1 if $e_k \in P_j$, and 0 otherwise. Notice that $W_{ik} = y_{ik}B_i$, and thus, $\mathbb{E}[W_{ik}] = y_{ik}\mu_i$ and $\operatorname{Var}[W_{ik}] = y_{ik}^2\sigma_i^2$. Next, applying (5), for a desired link utilization α, it is obtained that,

$$\mathbb{P}\left\{\sum_{\ell_i\in\mathcal{L}} y_{ik}B_i \geq \alpha C_k\right\} \leq \exp\left(-\frac{(\alpha C_k - \sum_{\ell_i\in\mathcal{L}} y_{ik}\mu_i)^2}{2\sum_{\ell_i\in\mathcal{L}} y_{ik}^2\sigma_i^2}\right)\,. \tag{7}$$

Therefore, to restrict the congestion probability on physical link e_k by ε_k, the following inequality should be satisfied,

$$\exp\left(-\frac{(\alpha C_k - \sum_{\ell_i} y_{ik}\mu_i)^2}{2\sum_{\ell_i} y_{ik}^2\sigma_i^2}\right) \leq \varepsilon_k, \tag{8}$$

which leads to the following inequality,

$$\big(2\ln\frac{1}{\varepsilon_k}\big)\sum_{\ell_i\in\mathcal{L}} \sigma_i^2 y_{ik}^2 \leq \Big(\alpha C_k - \sum_{\ell_i\in\mathcal{L}} \mu_i y_{ik}\Big)^2. \tag{9}$$

B. Congestion on Physical Paths

A path is a sequence of links, thus, we have,

$$\mathbb{P}\{\text{congestion on path } P_j\} = 1 - \textstyle\prod_{e_k\in P_j}(1-\varepsilon_k)\,. \tag{10}$$

In practice, we have $\varepsilon_k \ll 1$, and thus using the union bound, we obtain the following approximation for the path congestion probability,

$$\mathbb{P}\{\text{congestion on path } P_j\} \approx \textstyle\sum_{e_k\in P_j} \varepsilon_k\,. \tag{11}$$

Therefore, to restrict the congestion probability on path P_j by ϵ, the following inequality should be satisfied,

$$\textstyle\sum_{e_k\in P_j} \varepsilon_k \leq \epsilon\,. \tag{12}$$

C. Exact VLE Formulation

The VLE problem can be formulated as a non-linear optimization problem, as presented in Problem 1, where,

- Constraint (13a) enforces bandwidth embedding over all possible candidate paths.
- Constraint (13c) enforces end-to-end congestion probability ϵ on every path.
- Constraint (13d) enforces link congestion probability ε_k on link e_k, so as to satisfy Constraint (13c).

Problem 1 Exact VLE.

minimize α

subject to:

$$\sum_{P_j\in\mathcal{P}_i} x_{ij} = 1, \qquad \forall \ell_i \in \mathcal{L} \tag{13a}$$

$$y_{ik} = \sum_{P_j\in\mathcal{P}_i} x_{ij}\mathbf{I}_{e_k\in P_j}, \qquad \forall \ell_i \in \mathcal{L}, \forall e_k \in \mathcal{E} \tag{13b}$$

$$\sum_{e_k\in P_j} \varepsilon_k \leq \epsilon, \qquad \forall P_j \in \mathcal{P} \tag{13c}$$

$$\big(2\ln\frac{1}{\varepsilon_k}\big)\sum_{\ell_i\in\mathcal{L}} \sigma_i^2 y_{ik}^2 \leq \big(\alpha C_k - \sum_{\ell_i\in\mathcal{L}} \mu_i y_{ik}\big)^2, \forall e_k \in \mathcal{E} \tag{13d}$$

$$x_{ij}, y_{ij}, z_{ij} \in [0,1], \tag{13e}$$

$$\alpha \geq 0\,. \tag{13f}$$

All the constraints in Problem 1 are linear except (13d), which is non-linear and non-convex. To show the non-convexity of this constraint, we transform it to an equivalent system of inequalities, as follows. Let $\theta_k = 2\ln\frac{1}{\varepsilon_k}$. Define the following auxiliary constraint,

$$\theta_k y_{ik}^2 \leq z_{ik}^2\,. \tag{14}$$

The Constraint (13d) is then equivalent to the following constraints,

$$\textstyle\sum_{\ell_i\in\mathcal{L}} \sigma_i^2 z_{ik}^2 \leq u_{ik}^2, \tag{15a}$$

$$u_{ik} = \alpha C_k - \textstyle\sum_{\ell_i\in\mathcal{L}} \mu_i y_{ik}, \tag{15b}$$

$$\theta_k y_{ik}^2 \leq z_{ik}^2, \tag{15c}$$

$$\theta_k \geq 0\,. \tag{15d}$$

It is clear that (15a) is a second-order cone (SOC) constraint, (15b) and (15d) are linear, but (15c) is non-convex.

To solve Problem 1, one approach is to use a global nonlinear solver such as Knitro [23]. We have implemented this approach and experimented with various network configurations. While it is possible to solve the problem for small network instances, it takes prohibitively long time to solve the problem for any realistic network size. Moreover, since the problem is not convex, the computed solutions may not even be the global optimal ones. Therefore, to solve the problem for large network instances, we design an approximate solution, as presented in the next subsection. We show that the approximation results in a second-order cone program (SOCP) [24] that can be solved efficiently (in polynomial time) using conventional solvers such as Gurobi [25].

D. Approximate VLE Formulation

The complication in Constraint (13d) is due to the fact that the program tries to *optimally* assign congestion probabilities to each link on a given path. If we could pre-compute ε_k for each link e_k, then Problem 1 could be converted to a SOCP, as demonstrated by (15a)-(15d). Specifically, Constraint (13d) is equivalent to the following constraints,

$$\textstyle\sum_{\ell_i} \sigma_i^2 z_{ik}^2 \leq u_{ik}^2 \tag{16a}$$

$$u_{ik} = \frac{1}{\sqrt{-2\ln \varepsilon_k}}\Big(\alpha C_k - \textstyle\sum_{\ell_i} \mu_i y_{ik}\Big) \tag{16b}$$

which are SOC for a given ε_k.

Our approximate formulation is based on the simplification that all links in a path achieve the same congestion probability. Clearly, this results in a sub-optimal solution because the optimal solution may assign different congestion probabilities to different links on the same path. Let ε_j denote the link congestion probability for each link in path P_j, that is $\varepsilon_k = \varepsilon_j$, for all $e_k \in P_j$. Let $|P_j|$ denote the length of path P_j, we have,

$$\mathbb{P}\left\{\text{congestion on path } P_j\right\} = 1 - (1-\varepsilon_j)^{|P_j|}\,. \tag{17}$$

Therefore, to satisfy the end-to-end path congestion probability, we obtain that,

$$1 - (1-\varepsilon_j)^{|P_j|} \leq \epsilon \Leftrightarrow \varepsilon_j \leq 1 - \sqrt[|P_j|]{1-\epsilon}\,. \tag{18}$$

One problem arising from the above congestion probability allocation policy is that a link may be common among multiple paths of different lengths. In such cases, the longest path determines the required congestion probability on the common links, as it requires lower congestion on each link. As a result, the congestion probability assignment for the uncommon links must be updated (*i.e.*, increased) to satisfy the end-to-end path congestion constraint, as described next. Consider some path P_j. Let $\hat{P}_j$ denote the set of links in this path whose congestion probabilities are not assigned yet. For the remaining links in the path, *i.e.*, $e_k \in P_j \setminus \hat{P}_j$, their congestion probabilities have already been assigned as they are common with some longer paths. Let $\hat{\varepsilon}_j$ denote the congestion probability that should be assigned to every link in $\hat{P}_j$. The following relation should be satisfied:

$$1 - (1-\hat{\varepsilon}_j)^{|\hat{P}_j|} \textstyle\prod_{e_k \in P_j \setminus \hat{P}_j} (1-\varepsilon_k) \leq \epsilon, \tag{19}$$

which yields the following relation,

$$\hat{\varepsilon}_j \leq 1 - \sqrt[|\hat{P}_j|]{\frac{1-\epsilon}{\prod_{e_k \in P_j \setminus \hat{P}_j}(1-\varepsilon_k)}}\,. \tag{20}$$

The congestion assignment process is described in Algorithm 1.

Algorithm 1 Link Congestion Assignment.

```
1:  $\vec{\mathcal{P}} \leftarrow$ sort $\mathcal{P}$ from longest to shortest path
2:  $\varepsilon_k \leftarrow 0, \quad \forall e_k \in \mathcal{E}$
3:  for $j = 1$ to $|\vec{\mathcal{P}}|$ do
4:      $\pi \leftarrow 1$
5:      $P_j \leftarrow \vec{\mathcal{P}}[j]$
6:      $\hat{P}_j \leftarrow \{e_k \in P_j | \varepsilon_k = 0\}$
7:      for all $e_k \in P_j \setminus \hat{P}_j$ do
8:          $\pi \leftarrow \pi \times (1-\varepsilon_k)$
9:      end for
10:     for all $e_k \in \hat{P}_j$ do
11:         $\varepsilon_k = 1 - \sqrt[|\hat{P}_j|]{\frac{1-\epsilon}{\pi}}$
12:     end for
13: end for
```

The optimization problem is then reduced to the SOCP problem presented in Problem 2. Once the routing variables x_{ij} are computed, we may find that some paths are not used by any virtual network. Thus, we can adjust link congestion probabilities accordingly. To adjust link congestion probabilities, we simply remove the unused paths, re-assign link congestion probabilities and solve the optimization problem again.

Problem 2 Approximate VLE.

minimize α

subject to:

$$\sum_{P_j \in \mathcal{P}_i} x_{ij} = 1, \qquad \forall \ell_i \in \mathcal{L} \tag{21a}$$

$$y_{ik} = \sum_{P_j \in \mathcal{P}_i} x_{ij} \mathbf{I}_{e_k \in P_j}, \qquad \forall \ell_i \in \mathcal{L}, \forall e_k \in \mathcal{E} \tag{21b}$$

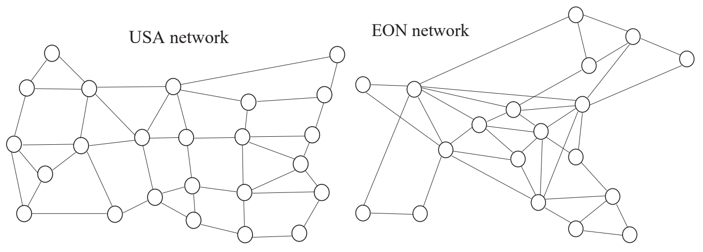
$$\big(2\ln\frac{1}{\varepsilon_k}\big) \sum_{\ell_i \in \mathcal{L}} \sigma_i^2 y_{ik}^2 \leq \big(\alpha C_k - \sum_{\ell_i \in \mathcal{L}} \mu_i y_{ik}\big)^2, \forall e_k \in \mathcal{E} \tag{21c}$$

$$x_{ij}, y_{ik} \in [0,1], \tag{21d}$$

$$\alpha \geq 0\,. \tag{21e}$$

IV. Performance Evaluation

We have conducted extensive simulations and Mininet experiments in order to assess the performance of our exact and approximate models. Knitro [23], an advanced solver for nonlinear optimization, is used to solve the exact VLE problem. Moreover, AMPL [26] modeling language is employed to enhance the accuracy of the local optimal solution. The approximate VLE model is a SOCP, which is solved using Gurobi [25]. Computations are carried out on a single machine with an Intel(R) Core(TM) i7-4770 CPU@3.40 GHz, 4 Cores and 8 Logical Processors, with 16 GB RAM.



(a) USA network topology. (b) EON network topology.

Fig. 2: Small-scale network topologies.

Two sets of evaluations are conducted: 1) small-scale evaluations, 2) large-scale evaluations. Small-scale evaluations consider small network topologies and are used to compare the exact and approximate VLE solutions. Recall that solving the exact problem for large networks is computationally intractable. Large-scale evaluations consider randomly generated large network topologies and are used to study the effect of various system parameters on the utility and scalability of the approximate VLE solution.

A. Simulation Parameters

For small-scale evaluations, the USA (24 nodes and 43 links) and European Optical Network (EON) (19 nodes and 37 links) topologies are used (see Fig. 2). These topologies are widely used in the literature for similar evaluation purposes (*e.g.*, see [7], [27]). For large-scale evaluations, random topologies are generated, as discussed in sub-section IV-D.

We assume that all virtual links have the same mean bandwidth demand of μ. Physical link capacities are also assumed to be equal and given by C, where C is scaled with respect to the mean bandwidth demand μ (*i.e.*, $C = \rho$ means that the link capacity is $\rho\mu$ in bps). In our evaluations, C varies between 10 and 100, but is set to 20 by default. We note that using unscaled values for link capacities (*e.g.*, 10 Gbps) actually slows down the optimization solver considerably. We also set the variance of bandwidth demand to σ^2 for all virtual links. In this section, we use the Coefficient of Variation, denoted by CoV and defined as CoV $= \sigma/\mu$, to describe the variability of bandwidth demands. A high CoV indicates high uncertainty in bandwidth demands, and vice versa.

TABLE II: Default simulation parameters.

Parameter	Value
C	20
μ	1
K	3
CoV	1.0
ϵ	0.1
Network Topology	USA

The optimization programs take as input the set of virtual links that need to be embedded in the network. Each virtual link is assigned to a randomly chosen origin-destination pair in the physical network. Then, a K-shortest-path algorithm [28] is run to compute K candidate paths between the chosen origin-destination nodes for each virtual link. Next, the optimization models are solved based on the computed candidate paths and network parameters. The default values for the parameters are presented in Table II. The value of a parameter changes only when its impact is investigated. Each point in the plots (for simulations) is the average of four simulation runs. The error bars (showing the min and max values) are not presented in cases where the deviation from the average was very small.

B. Performance Measures

We use the following measures to compare the performance of the models: 1) the number of admitted virtual links, 2) the achieved congestion probability, and 3) the utilization of the most congested link (denoted by α). To compute the number of admitted virtual links, we solve the problems starting with a small set of virtual link requests. Then, we iteratively increase the number of virtual link requests until the problems become infeasible. While this linear search can be improved, *e.g.*, by a binary search, it takes only seconds to find the maximum number of admitted links for the approximate model. For the exact model, however, it is a time-consuming process and the starting point matters significantly due to local optimality of solutions returned by the nonlinear solver. Therefore, we have used starting points based on the approximate model solution to speedup the search process.

C. Exact and Approximate Models

In this subsection, we compare the performance of the exact and approximate models in terms of the average number of admitted virtual links and achieved congestion probability. Our results show that the approximate model produces results that are very close to those of the exact model.

Number of Admitted Links. Fig. 3 shows the the number of admitted virtual links by each model for different coefficients of variation (CoV = $0, 0.5, 1, 1.5$). By increasing CoV, demand uncertainty increases and more bandwidth is reserved per virtual link, which causes a sharp decline in the number of admitted virtual link requests. We observe that the results achieved by the approximate and exact model are very close to each other. The reason for slightly higher number of admitted links under the approximate model can be explained by looking at Fig 4. We can see that the approximate model generally achieves higher congestion probabilities, which translates to more admitted virtual link requests.

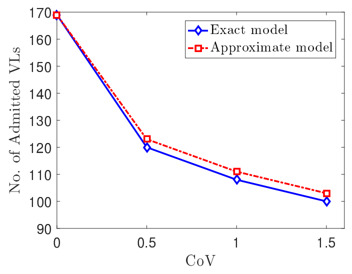
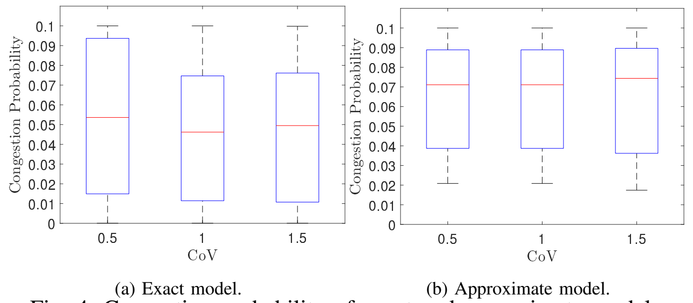


Fig. 3: Average number of admitted virtual links with exact and approximate models for different uncertainty levels.

Congestion Probability. Fig. 4 depicts the achieved end-to-end congestion probabilities for exact and approximate algorithm. For both models, the objective is to satisfy a maximum congestion probability of $\epsilon = 0.1$. We observe that: 1) both models generally satisfy the target congestion probability, and 2) the exact model generally achieves lower congestion probabilities compared to the approximate one. This means that, in some cases, the approximate model may admit more virtual links. The reason is that the approximate model is forced to achieve a specific congestion level at each physical link (as discussed in subsection III-D), which may not be optimal, but the exact model is free to decide about the optimal level of congestion for each link.



(a) Exact model. (b) Approximate model.

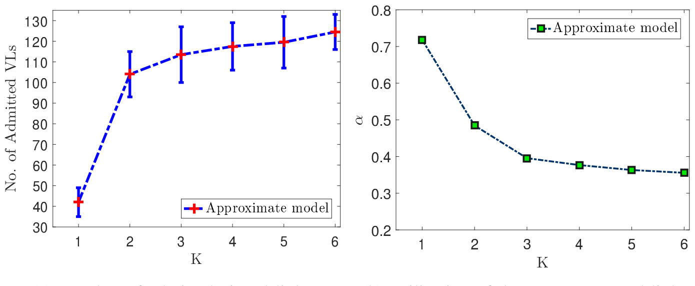
Fig. 4: Congestion probability of exact and approximate models.

D. *Effect of Network Parameters*

In this section, we focus on the approximate model, as it is computationally fast and is reasonably accurate compared to the exact model.

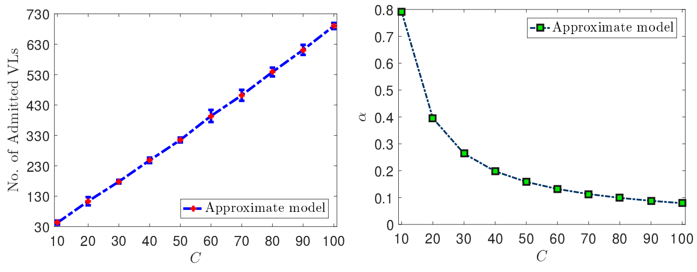
Number of Candidate Paths. Fig. 5 shows the effect of increasing the number of candidate paths K on the performance of the approximate model. As can be seen in Fig. 5(a), the maximum number of admitted links increases drastically by increasing the number of candidate paths from 1 to 3. However, the gain diminishes as the number of paths increases beyond 3. Therefore, we set the default number of candidates paths to 3 in the rest of the simulations in this section. Fig. 5(b) illustrates the impact of K on the most congested link, *i.e.*, the link with the highest utilization α. In this figure, the number of virtual links is fixed at $|\mathcal{L}| = 30$. As expected, the highest link utilization drops as we increase the number of candidate paths. Again, there is no considerable reduction after $K = 3$.

Physical Link Capacity. Figs. 6(a) and 6(b) show the effect of the physical link capacity on the number of admitted virtual links and most congested link utilization. In this scenario, CoV and μ are set to their default values, and the scaled link capacity C varies between $C = 10$ to $C = 100$. As expected, the maximum number of accepted links increases linearly with respect to the link capacity. In Fig. 6(b), the number of virtual links is fixed at $|\mathcal{L}| = 30$. We observe that the utilization α drops by almost 87% when C increases from 10 to 100. Also, the reduction in link utilization diminishes as the link capacity increases beyond $C = 30$.



(a) Number of admitted virtual links. (b) Utilization of the most congested link.

Fig. 5: Effect of the number of paths (K).



(a) Number of admitted virtual links. (b) Utilization of the most congested link.

Fig. 6: Effect of the physical link capacity (C).

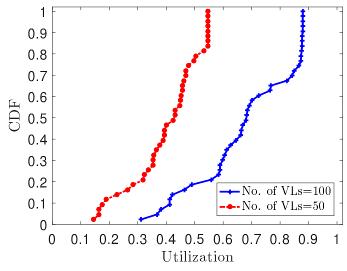


Fig. 7: Effect of the number of virtual links on utilization.

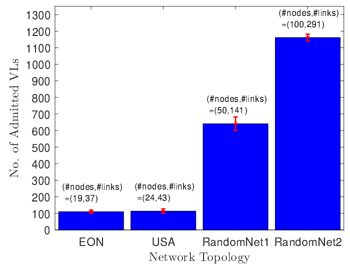


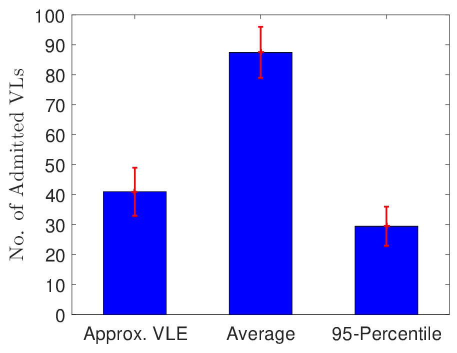
Fig. 8: Effect of the network topology on admitted links.

Number of Virtual Link Requests. Fig. 7 shows the CDF of the maximum link utilization when the number of virtual link requests is set to 50 and 100. All other parameters are set to their default values. For 50 virtual link requests, the network achieves at most 50% utilization, which increases to 90% by doubling the number of virtual link requests to 100.

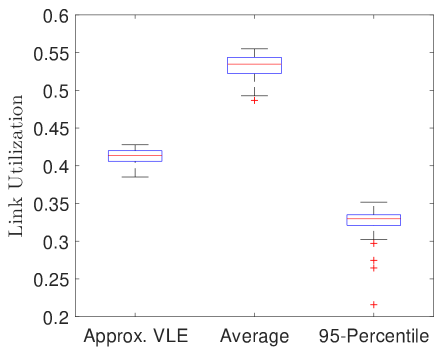
Network Topology. Fig. 8 shows the number of admitted links for different network topologies. Besides the USA and EON topologies, we have generated two large random networks with 50 (141 links) and 100 (291 links) nodes. Barabasi-Albert model [29] is used to generate random scale-free networks. The objective of this experiment is to show that the approximate model can be used to handle large networks with ease. Four different random sets of origin-destination pairs are generated for each topology. The average number of admitted virtual links along with bars showing the min and max values are plotted in the figure. As expected, the number of admitted links increases by increasing the size of the network.

E. *Mininet Experiments*

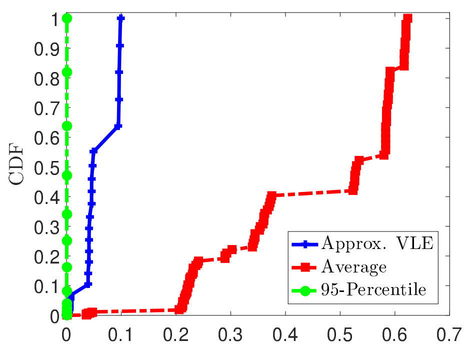
In this subsection, our goal is to determine whether the results obtained via simulations can qualitatively match the results measured on Mininet. In these experiments, we use the USA topology, origin-destinations corresponding to virtual links are randomly picked from physical nodes, and candidate paths between origin-destination pairs are based on K-shortest paths, as in the simulations. To implement multi-path routing in Mininet, we use the routing variables x_{ij} computed in



(a) Average number of admitted virtual links.



(b) Average utilization of physical links.



(c) CDF of packet drop probability.

Fig. 9: Mininet experimental results on the USA topology.

the optimization problem to split bandwidth demands statically among the candidate paths.

Implemented Algorithms. We implemented three VLE algorithms, as described below:

- **Approx VLE:** This is the approximate VLE model developed in this paper (see Problem 2).
- **Average:** This algorithm ignores demand variability and bases its bandwidth allocation decisions on the mean bandwidth demand only. Specifically, it allocates μ bps bandwidth to each virtual link.
- 95**-Percentile:** This algorithm actually considers demand variability when mapping virtual links. Specifically, it assumes that the bandwidth demands follow a Normal distribution with mean μ and variance σ^2 (which are assumed to be known). It then computes an *effective demand* for each virtual link, which is equal to the 95-percentile of the bandwidth demand given by $\mu + 1.65\sigma$. We chose the 95-percentile so that if a virtual link is mapped to a path of length 2, and no other link uses that path, then the end-to-end congestion probability of the link remains below $\epsilon = 0.1$.

Notice that **Average** and 95**-Percentile** are deterministic mapping algorithms, which were solved exactly by modifying our exact VLE formulation in Problem 1.

Experiment Setup. In Mininet, we generate traffic for each virtual link at a rate that is distributed Normally with parameters μ and σ. We set μ to 1 Mbps, CoV to 1, and C to 10 (*i.e.*, link capacity is 10 Mbps). We note that CoV = 1 represents a scenario with high level of uncertainty. However, it is chosen here to show that even in such a extreme scenario, the proposed model achieves reasonable performance. Even though 1 Gbps network links are very common, these small values are used to decrease the Mininet experimentation time, as every operation (*e.g.*, switching with Open vSwitch) is performed in software. To emulate demand variability, the traffic rates are changed over time. Specifically, for each virtual link, its rate of traffic is sampled from the Normal distribution every second during the course of the experiment. Each experiment is run for 15 minutes to achieve stable results.

Measurements. In each experiment, we measure the utilization of every physical link and compute the packet drop probability for each virtual link. To compute the packet drop probability, we count the number of packets transmitted and received at the origin and destination nodes of a link. The ratio of dropped packets to the transmitted packets gives us the packet drop probability. To compute link utilization, we keep track of every packet transmitted over each link throughout the experiment by creating a trace file. Given the size of packets (1066 bytes including link layer headers), we then compute the utilization of each link over 10 second intervals and use them to compute the overall average utilization of each link. Congestion events are the primary cause of packet drops in our experiments. However, during each congestion event, multiple packets could be dropped. Thus, while there is a strong correlation between packet drop probability and congestion probability, their values do not necessarily match.

Results and Discussion. Mininet results are summarized in Fig. 9. Specifically, Fig. 9(a) depicts the average number of admitted virtual links with each algorithm. As expected, **Approx VLE** admits significantly fewer links compared to the very optimistic **Average**. It, however manages to admit more links compared to 95**-Percentile** as it takes advantage of statistical multiplexing of multiple demands on each link. We see a similar relation between average physical link utilizations in Fig. 9(b), where **Average** achieves much higher utilization compared to the other algorithms. However, the price to be paid for such a high utilization is the unacceptably high packet drop probability achieved by **Average**, as depicted in Fig. 9(c). As can be seen, while **Approx VLE** and 95**-Percentile** achieve less than 10% drop probability, the drop probability of **Average** can go as high as 60%, which would render the network unusable for any network services relying on TCP. Another interesting observation is that, 95**-Percentile** is very conservative and achieves almost 0% drop probability at the cost of admitting fewer virtual link requests.

V. Conclusion

In this paper, we considered the problem of congestion-constraint virtual link embedding with uncertain demands. We showed that the problem can be well approximated by a second order cone program, which can be solved efficiently even for large networks. Our simulations and Mininet experiments show that considering uncertainty in demands results in more efficient usage of network resources. An interesting extension of this work is to consider online virtual network embedding with uncertain demands subject to congestion constraints.

References

[1] A. Fischer *et al.*, "Virtual network embedding: a survey," *IEEE Communications Surveys and Tutorials*, vol. 15, no. 4, pp. 1888–1906, Oct. 2013.

[2] M. Wang, X. Meng, and L. Zhang, "Consolidating virtual machines with dynamic bandwidth demand in data centers," in *Proc. IEEE INFOCOM*, Shanghai, China, Apr. 2011.

[3] S. Coniglio, A. Koster, and M. Tieves, "Data uncertainty in virtual network embedding: robust optimization and protection levels," *Netw. Syst. Manage.*, vol. 24, no. 3, pp. 681–710, May 2016.

[4] ——, "Virtual network embedding under uncertainty: exact and heuristic approaches," in *Proc. IEEE Design of Reliable Commun. Netw.*, Kansas City, USA, Mar. 2015.

[5] L. Yu and H. Shen, "Bandwidth guarantee under demand uncertainty in multi-tenant clouds," in *Proc. IEEE ICDCS*, Madrid, Spain, Jun. 2014.

[6] G. Sun *et al.*, "Exploring online virtual networks mapping with stochastic bandwidth demand in multi-datacenter," *Photon. Netw. Commun.*, vol. 23, no. 2, Apr. 2012.

[7] S. S. W. Lee *et al.*, "Design of bandwidth guaranteed openflow virtual networks using robust optimization," in *Proc. IEEE GLOBECOM*, Austin, USA, Dec. 2014.

[8] J. Botero *et al.*, "Energy efficient virtual network embedding," *Comput. Netw.*, vol. 16, no. 5, pp. 756–759, May 2012.

[9] I. Houidi *et al.*, "Virtual network provisioning across multiple substrate networks," *Comput. Netw.*, vol. 55, no. 4, pp. 1011–1023, Mar. 2011.

[10] N. M. Chowdhury, M. R. Rahman, and R. Boutaba, "Virtual network embedding with coordinated node and link mapping," in *Proc. IEEE INFOCOM*, Rio de Janeiro, Brazil, Apr. 2009.

[11] ——, "Vineyard: virtual network embedding algorithms with coordinated node and link mapping," *IEEE/ACM Trans. Netw.*, vol. 20, no. 1, pp. 206–219, Feb. 2012.

[12] R. Guerin, H. Ahmadi, and M. Naghshineh, "Equivalent capacity and its application to bandwidth allocation in high-speed networks," *IEEE J. Sel. Areas Commun.*, vol. 9, no. 7, pp. 968–981, Sep. 1991.

[13] C. Wang and H. P. Luh, "Analysis of bandwidth allocation on end-to-end qos networks under budget control," *Computers and Mathematics with Applications*, vol. 62, no. 1, pp. 419–439, Jul. 2011.

[14] C. Qiu, H. Shen, and L. Chen, "Probabilistic demand allocation for cloud service brokerage," in *Proc. IEEE INFOCOM*, San Francisco, USA, Apr. 2016.

[15] O. Heckmann, J. Schmitt, and R. Steinmetz, "Robust bandwidth allocation strategies," in *IEEE IWQoS*, Miami Beach, USA, May 2002.

[16] A. Ben-Tal, L. E. Ghaoui, and A. Nemirovski, *Robust optimization*. Princeton University Press, 2009.

[17] M. S. Dimitris Bertsimas, "The price of robustness," *Operations Research*, vol. 52, no. 1, pp. 35–53, 2004.

[18] (2018) Mininet. [Online]. Available: http://mininet.org/

[19] L. Yu *et al.*, "Towards bandwidth guarantee for virtual clusters under demand uncertainty in multi-tenant clouds," *IEEE Trans. Parallel and Distrib. Syst.*, vol. 29, no. 2, pp. 450–465, Feb. 2018.

[20] M. Roughanx *et al.*, "Class-of-service mapping for qos: a statistical signature-based approach to ip traffic classification," in *Proc. ACM IMC*, Taormina, Italy, Oct. 2004.

[21] H. Chernoff, "A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations," *Math. Statist.*, vol. 23, no. 4, pp. 493–507, Dec. 1952.

[22] M. Wainwright, "Basic tail and concentration bounds," https://www.stat.berkeley.edu/~mjwain/stat210b/Chap2_TailBounds_Jan22_2015.pdf, 2015, online; accessed 28 May 2018.

[23] (2018) Artelys knitro. [Online]. Available: https://www.artelys.com/en/optimization-tools/knitro

[24] F. Alizadeh and D. Goldfarb, "Second-order cone programming," *Math. Program.*, vol. 95, no. 1, pp. 3–51, Dec. 2001.

[25] (2018) Gurobi optimization. [Online]. Available: http://www.gurobi.com/

[26] (2018) Ampl. [Online]. Available: https://ampl.com/

[27] A. Altin *et al.*, "Provisioning virtual private networks under traffic uncertainty," *Netw.*, vol. 49, no. 1, pp. 100–115, Jan. 2007.

[28] J. Y. Yen, "Finding the k shortest loopless paths in a network," *Manag. Science*, vol. 17, no. 11, pp. 712–716, Jul. 1971.

[29] A. Barabas and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, Oct. 1999.