Client-Constrained Virtual Network Embedding under Uncertainty

Junkai He, Makhlouf Hadji, Djamal Zeghlache

Client-Constrained Virtual Network Embedding under Uncertainty

1st Junkai He
Department RS2M
Télécom SudParis
Évry, France
junkai.he02@gmail.com

2nd Makhlouf Hadji
Technological Research Institute SystemX
2 BVD Thomas Gobert
Palaiseau, France
makhlouf.hadji@gmail.com

3rd Djamal Zeghlache
Department RS2M
Télécom SudParis
Évry, France
djamal.zeghlache@telecom-sudparis.eu

***Abstract*—This paper addresses uncertainty in resource demands and heterogeneous requests with affinity and anti-affinity constraints on virtual nodes and links in traditional Virtual Network Embedding. This is realized using stochastic modeling and methods based on an initial Integer-Linear Programming (ILP) model formulation of the VNE problem. The ILP is extended to build a non-linear Chance-Constrained Programming (CCP) model to address uncertainty. The derived CCP model is then linearized for exploitation by standard solvers. Numerical experiments and comparisons with state-of-the-art methods illustrate the efficiency of our approaches. The results provide insight to cloud service providers on their resource investment to serve clients with affinity and anti-affinity requirements under uncertainty.**

***Keywords*—Virtual network embedding, Uncertainty, Customized request, Anti-affinity, Chance constraint**

I. Background and Motivations

The Virtual Network Embedding (VNE) problem, mapping virtual Networks (VN) onto Substrate Networks (SN), has received widespread attention from researchers (1). So far, the state-of-the-art addressing the NP-hard VNE problem (2) has paid limited attention to uncertainty in resource demands and specific client constraints. Clients requesting network services can have diverse and significant variations in their requirements within their own VNs and across different clients' requests as hinted in (3). Therefore, there is a need to address the client-constrained VNE problem under uncertainty. The uncertainty in resource demands appears because clients often have partial and incomplete knowledge of their needs, leading to imprecise consumption and user requirements in computing power (CPU) and bandwidth. This poses challenges for service providers to ensure the provisioning of resources with sufficient margins and confidence to maintain user quality of service. Compared with the deterministic case, the stochastic VNE problem has rarely been studied in the current state of the art. The works (4; 5; 6; 7) handle uncertain CPU and bandwidth requirements via stochastic simulation. The authors of (8; 9; 10; 11) use chance constraints to quantify uncertainty and apply robust techniques (the τ-robust method) to solve the stochastic VNE problem. In comparison with existing stochastic VNE works, we resort to probability distributions to derive a deterministic equivalent formulation for the non-linear chance-constrained mathematical model. The deterministic equivalent formulation replaces uncertain elements (non-linear) by linear representations that can be handled by solvers (12).

The specific client constraints correspond to separation and co-localization requirements on virtual nodes and links also known as affinity and anti-affinity constraints. To our knowledge, existing VNE works focus on these constraints on virtual nodes, often neglecting them on virtual links (3). Network slicing for example faces demands of isolated VN requests with specific link anti-affinity constraints stemming from different verticals. This motivates us to fill the identified gap in current VNE studies by integrating client-specific constraints and restrictions on virtual nodes and links in mathematical modeling. Further, the majority of state-of-the-art studies address to some extent (anti-) affinity requirements *within a specific VN request* (see (3; 13; 7; 14; 15)). However, clients' VN requests come along with security, resilience, isolation, and performance requirements and thus impose constraints on hosting and connectivity services *across different VN requests*. As far as we know, previous works in (16) and (17) consider the mapping of service function chains (a particular case of the VNE) under anti-affinity constraints across VN requests using simulations rather than formal optimization approaches.

We aim to fill these research gaps and consequently support cloud service providers in resource investment, in order to satisfy client requirements with affinity and anti-affinity constraints under uncertainty. Our contributions are summarized as follows:

1) We address the VNE problem by not only addressing uncertainty and variations in client demands but also by complying with node and link affinity and anti-affinity constraints within and across VN requests (see Section II). An Integer Linear

Programming (ILP) model for jointly mapping virtual nodes and links considering affinity and anti-affinity constraints;
2) We propose a **stochastic programming** with **chance constraints** for the VNE problem considering uncertain CPU and bandwidth requests;
3) We derive an equivalent formulation of the chance-constrained programming model to resort to standard solvers.

Section II describes the stochastic client-constrained VNE problem. Section III provides an ILP-based model for the deterministic issue and then addresses the stochastic version using chance-constrained programming and its deterministic equivalent formulation. Section IV assesses the performance of our proposed algorithms. Conclusions and future work are provided in Section V.

II. Problem description

A. Network Topology and Attributes

We model the SN as an undirected graph $G^{\mathcal{S}} = (N^{\mathcal{S}}, L^{\mathcal{S}}, CPU^{\mathcal{S}}, BW^{\mathcal{S}})$, where $N^{\mathcal{S}}$ and $L^{\mathcal{S}}$ represent respectively the sets of physical nodes and links. Parameters $CPU^{\mathcal{S}}$ and $BW^{\mathcal{S}}$ denote the available CPU of nodes and bandwidth of links, respectively. We make sure that the generated SN graphs guarantee the existence of a "path" between any pair of physical nodes. If there is more than one path available between physical nodes m and n, the **Shortest Available Path** is selected (a path containing links with bandwidths no less than the requested bandwidth of a virtual link).

We represent heterogeneous VN requests as undirected graphs denoted by $G^{\mathcal{V}} = (N^{\mathcal{V}}, L^{\mathcal{V}}, \widetilde{CPU}^{\mathcal{V}}, \widetilde{BW}^{\mathcal{V}}, \alpha, \vec{\beta}(\beta n, \beta l))$, where $N^{\mathcal{V}}$ and $L^{\mathcal{V}}$ represent respectively the sets of virtual nodes and links. The **uncertain** CPU on nodes and bandwidth on links are represented by $\widetilde{CPU}^{\mathcal{V}}$ and $\widetilde{BW}^{\mathcal{V}}$. Notably, we define **affinity** and **anti-affinity** constraints (for nodes α and links $\vec{\beta}$) as follows:

- **Within a given VN**, and for each pair of the same VN nodes i and j, we use α to represent node affinity (co-localization when $\alpha_{ij} = 1$) and node anti-affinity (separation when $\alpha_{ij} = 0$). We do not consider link separation in VN hosting requests since tenants (owners of the VNs) can use flow classifiers to build their own forwarding graphs (just like in ETSI NFV's VNF-FG solutions (18)).
- **Across different VN requests** (VN_t and VN_{-t}, where "$-t$" indicates other VNs different from t), we use βn for node anti-affinity (when $\beta n = 0$) between nodes of VN_t and nodes of VN_{-t}. We use βl to denote link anti-affinity (when $\beta l = 0$) between links of VN_t and links of VN_{-t}.

B. Problem Characteristics and Measures

We consider that VN requests arrive dynamically as Poisson processes, and embedded requests depart following an exponential service time when substrate resources are released (3). For simplicity and without loss of generality, we handle VN requests sequentially although they could be processed in a batch mode since a batch is also just a more complex graph with more constraints. Monitoring the conditions and states of the SN at each allocation step is essential to capture the dynamic behavior of the SN. The update of SN topology and resources is determined by three factors: 1) Mapping a VN request leads to resource (CPU and bandwidth) occupation. The ratio of resource occupation is algorithm-dependent, determining SN resource availability for future embedding; 2) Decommissioning the mapped VN requests frees/releases resources after VN departures; 3) Anti-affinity constraints may cause disconnected SN into multiple specific sub-graphs depending on the used algorithm.

For each VN request (or one allocation step), we minimize the usage of bandwidth resources in the SN. We focus on the following metrics to assess the performance of our algorithms:

- Acceptance ratio that indicates the number of successfully embedded VNs out of the total number of received VN requests.
- Resources usage ratio that records the percentage of used SN nodes and links.
- Convergence time of a successful mapping that reflects the efficiency of the proposed methods.

Since the classical VNE problem is NP-Hard (3), the VNE problem incorporating (anti)-affinity constraints and uncertainty is harder and consequently NP-Hard.

III. Mathematical Modeling

We present mathematical models and algorithms for both deterministic and stochastic VNE problems.

A. Deterministic Integer Linear Programming

The parameters and decision variables used in the deterministic ILP are defined in Tab. I.

For each VN request, our objective is an optimal mapping minimizing the total used SN bandwidth. This is formulated by (1).

$$\min \sum_{i \in N^{\mathcal{V}}} \sum_{j \in N^{\mathcal{V}}} \sum_{m \in N^{\mathcal{S}}} \sum_{n \in N^{\mathcal{S}}} BW^{\mathcal{V}}_{(i,j)} \cdot Q^{\mathcal{S}}_{(m,n)} \cdot y^{(m,n)}_{(i,j)} \tag{1}$$

Our optimization is subject to the linear constraints:

$$\sum_{m \in N^{\mathcal{S}}} x^m_i = 1, \quad \forall i \in N^{\mathcal{V}} \tag{2}$$

$$\sum_{m \in N^{\mathcal{S}}} \sum_{n \in N^{\mathcal{S}}} y^{(m,n)}_{(i,j)} = 1, \quad \forall (i,j) \in L^{\mathcal{V}} \tag{3}$$

TABLE I: Parameters and decision variables

Parameters	
$N^{\mathcal{S}}$	Set of physical nodes in the SN graph.
$N^{\mathcal{S}}_{occupied}$	Set of **occupied** physical nodes in the SN graph. It is initialized to an empty set and is updated after each VNE.
$L^{\mathcal{S}}$	Set of physical links in the SN graph.
$P^{\mathcal{S}}$	Set of shortest **available** paths (for all given pairs of physical nodes) in the SN graph.
$P^{\mathcal{S}}_{occupied}$	Set of paths containing **occupied** physical links (hosting at least one virtual link) in the SN graph.
$CPU^{\mathcal{S}}_m$	**Available** CPU resources of physical node $m \in N^{\mathcal{S}}$.
$BW^{\mathcal{S}}_{(m,n)}$	**Available** bandwidth resources of physical link $(m,n) \in L^{\mathcal{S}}$ and $m,n \in N^{\mathcal{S}}$.
$Q^{\mathcal{S}}_{(m,n)}$	Number of physical links in path (m,n), where $Q^{\mathcal{S}}_{(m,n)} \geqslant 2$ refers to a path with at least two links.
$N^{\mathcal{V}}$	Set of virtual nodes in a VN graph.
$N^{\mathcal{V}}_{iso}(\subseteq N^{\mathcal{V}})$	Subset of virtual nodes with isolation (anti-affinity) requirements in a VN graph.
$L^{\mathcal{V}}$	Set of virtual links in a VN graph.
$L^{\mathcal{V}}_{iso}(\subseteq L^{\mathcal{V}})$	Subset of virtual links with isolation (anti-affinity) requirements in a VN graph.
$CPU^{\mathcal{V}}_i$	**Required** CPU resources of virtual node $i \in N^{\mathcal{V}}$.
$BW^{\mathcal{V}}_{(i,j)}$	**Required** bandwidth resources of virtual link $(i,j) \in L^{\mathcal{V}}$ and $i,j \in N^{\mathcal{V}}$.
α_{ij}	= 0 if virtual nodes i and j must be separated; 1 if must be co-located, where $i,j \in N^{\mathcal{V}}$.
βn_i	= 0 if virtual node i does not share physical nodes with other VNs (node anti-affinity); 1 otherwise, $i \in N^{\mathcal{V}}$.
$\beta l_{(i,j)}$	= 0 if virtual link (i,j) does not share physical links with other VNs (link anti-affinity); 1 otherwise, $(i,j) \in L^{\mathcal{V}}$ and $i,j \in N^{\mathcal{V}}$.
Decision variables	
x^m_i	= 1 if virtual node $i \in N^{\mathcal{V}}$ is mapped to physical node $m \in N^{\mathcal{S}}$; 0 otherwise.
$y^{(m,n)}_{(i,j)}$	= 1 if virtual link $(i,j) \in L^{\mathcal{V}}$ is mapped to physical path $(m,n) \in P^{\mathcal{S}}$; 0 otherwise.

$$\sum_{n \in N^{\mathcal{S}}} y^{(m,n)}_{(i,j)} = x^m_i, \quad \forall (i,j) \in L^{\mathcal{V}}, m \in N^{\mathcal{S}} \tag{4}$$

$$\sum_{m \in N^{\mathcal{S}}} y^{(m,n)}_{(i,j)} = x^n_j, \quad \forall (i,j) \in L^{\mathcal{V}}, n \in N^{\mathcal{S}} \tag{5}$$

$$\sum_{i \in N^{\mathcal{V}}} CPU^{\mathcal{V}}_i \cdot x^m_i \leqslant CPU^{\mathcal{S}}_m, \quad \forall m \in N^{\mathcal{S}} \tag{6}$$

$$\sum_{i \in N^{\mathcal{V}}} \sum_{j \in N^{\mathcal{V}}} BW^{\mathcal{V}}_{(i,j)} \cdot y^{(m,n)}_{(i,j)} \leqslant BW^{\mathcal{S}}_{(m,n)}, \ \forall (m,n) \in P^{\mathcal{S}} \tag{7}$$

$$x^m_i + x^m_j \leqslant 1 + \alpha_{i,j}, \quad \forall i,j \in N^{\mathcal{V}}, m \in N^{\mathcal{S}} \tag{8}$$

$$x^m_i \geqslant -1 + x^m_j + \alpha_{i,j}, \quad \forall i,j \in N^{\mathcal{V}}, m \in N^{\mathcal{S}} \tag{9}$$

$$x^m_i \leqslant \beta n_i, \quad \forall i \in N^{\mathcal{V}}_{iso}, \ m \in N^{\mathcal{S}}_{occupied} \tag{10}$$

$$y^{(m,n)}_{(i,j)} \leqslant \beta l_{(i,j)}, \quad \forall (i,j) \in L^{\mathcal{V}}_{iso}, \ (m,n) \in P^{\mathcal{S}}_{occupied} \tag{11}$$

$$x^m_i, y^{(m,n)}_{(i,j)} \in \{0,1\}, \quad \forall i,j \in N^{\mathcal{V}}, m,n \in N^{\mathcal{S}} \tag{12}$$

Constraints (2) (resp. (3)) guarantee the hosting of each virtual node i (resp. virtual link (i,j)) in one and only one available physical node m (resp. in one and only one available physical path (m,n), no splitting). To realize the joint mapping of virtual nodes and links, we rely on constraints (4) and (5), ensuring that for a virtual node i mapped to physical node m, a virtual link (i,j) needs to be mapped to a physical path (m,n) with node m as the source or endpoint and node n as the other point. Constraints (6) and (7) state that the occupied resources (CPU and bandwidth) due to VN embedding must respect the available limit of the updated SN. Constraints (8) and (9) meet the affinity and anti-affinity requirements of virtual nodes of the same VN request, i.e., their co-localization and separation. Specifically, if $\alpha_{i,j}$ = 0, we obtain $x^m_i + x^m_j \leqslant 1$, indicating two nodes i and j in this VN are separated; If $\alpha_{i,j}$ = 1 and x^m_j = 1, then we get x^m_i = 1, achieving the co-localization of these two nodes. Constraints (10) ensure that if virtual node i has an anti-affinity requirement $\beta n_i = 0$ (i.e., i belongs to the set $N^{\mathcal{V}}_{iso}$), then i cannot be hosted by physical node m that is not free (i.e., serves at least one virtual node). Similarly, constraints (11) enforce that a physical path containing non-free links cannot be used to serve a virtual link with an anti-affinity requirement. Constraints (12) define the ranges of decision variables.

B. Stochastic Chance-Constrained Programming

To handle uncertain CPU and bandwidth requirements, namely $\widetilde{CPU}^{\mathcal{V}}_i$ and $\widetilde{BW}^{\mathcal{V}}_{(i,j)}$, we introduce chance constraints to ensure that the probability of respecting available physical resource capacity expressed in constraints (6) and (7) remains above specified levels. This helps service providers in managing risks caused by uncertainty. The following constraints (13) and (14) provide a mapping guarantee that confidence levels will not exceed the amount of available SN resources (CPU and bandwidth, respectively).

$$\mathbb{P}\left(\sum_{i \in N^{\mathcal{V}}} \widetilde{CPU}^{\mathcal{V}}_i \cdot x^m_i \leqslant CPU^{\mathcal{S}}_m \right) \geqslant \pi_1, \ \forall m \in N^{\mathcal{S}} \tag{13}$$

$$\mathbb{P}\left(\sum_{i \in N^{\mathcal{V}}} \sum_{j \in N^{\mathcal{V}}} \widetilde{BW}^{\mathcal{V}}_{(i,j)} \cdot y^{(m,n)}_{(i,j)} \leqslant BW^{\mathcal{S}}_{(m,n)} \right) \geqslant \pi_2, \forall (m,n) \in P^{\mathcal{S}} \tag{14}$$

where π_1 (resp. π_2) represents the **confidence level** of respecting available CPU (resp. available bandwidth) resources of a physical node (resp. link) in the SN. With chance constraints (13) and (14), we obtain the

Chance-Constrained Programming (CCP) model:

Model CCP Objective: (1)
Subject to: $(2)-(5),(8)-(12),(13),(14)$

To solve the CCP model using classical solvers, we transform the chance constraints using approximation.

C. Chance Constraint Approximation

The Gaussian distribution is commonly used and found adequate to approximate chance constraints (see (19)), we use it in Proposition 1 to approximate chance constraints (13) and (14) in the CCP model. For the sake of simplicity and with no loss of generality, we represent variables x_i^m and $y_{(i,j)}^{(m,n)}$ by $\mathbf{x}$ and $\mathbf{y}$.

Proposition 1. *If uncertain CPU requirements (resp. bandwidth) of virtual node i (resp. link (i,j)) rely on $\widetilde{CPU}_i^{\mathcal{V}} \sim \mathcal{N}(\mu_i, \sigma_i^2)$ (resp. $\widetilde{BW}_{(i,j)}^{\mathcal{V}} \sim \mathcal{N}(\mu_{(i,j)}, \sigma_{(i,j)}^2)$), then chance constraints* (13) *and* (14) *can be respectively approximated by:*

$$\sum_{i\in N^{\mathcal{V}}} \mu_i \cdot \mathbf{x} + \Phi^{-1}(\pi_1) \cdot \sqrt{\sum_{i\in N^{\mathcal{V}}} \sigma_i^2 \cdot \mathbf{x}^2} \leqslant CPU_m^{\mathcal{S}}, \forall m \in N^{\mathcal{S}} \tag{15}$$

$$\sum_{i\in N^{\mathcal{V}}}\sum_{j\in N^{\mathcal{V}}} \mu_{(i,j)} \cdot \mathbf{y} + \Phi^{-1}(\pi_2) \cdot \sqrt{\sum_{i\in N^{\mathcal{V}}}\sum_{j\in N^{\mathcal{V}}} \sigma_{(i,j)}^2 \cdot \mathbf{y}^2} \leqslant BW_{(m,n)}^{\mathcal{S}}, \quad \forall (m,n) \in P^{\mathcal{S}} \tag{16}$$

where $\Phi(\cdot)$ denotes the cumulative distribution function of the Standard Normal (Gaussian) distribution.

Proof: As $\widetilde{CPU}_i^{\mathcal{V}} \sim \mathcal{N}(\mu_i, \sigma_i^2)$, chance constraints (13) are equivalent to:

$$\mathbb{P}\left[\frac{\sum_{i\in N^{\mathcal{V}}} \widetilde{CPU}_i^{\mathcal{V}} \cdot \mathbf{x} - \sum_{i\in N^{\mathcal{V}}} \mu_i \cdot \mathbf{x}}{\sqrt{\sum_{i\in N^{\mathcal{V}}} \sigma_i^2 \cdot \mathbf{x}^2}} \leqslant \frac{CPU_m^{\mathcal{S}} - \sum_{i\in N^{\mathcal{V}}} \mu_i \cdot \mathbf{x}}{\sqrt{\sum_{i\in N^{\mathcal{V}}} \sigma_i^2 \cdot \mathbf{x}^2}}\right] \geqslant \pi_1, \quad \forall m \in N^{\mathcal{S}} \tag{17}$$

The inequality in the brackets $[\cdot]$ of constraints (17) thus follows a $\mathcal{N}(0,1)$ and (17) can be standardized using the properties of the standard normal distribution:

$$\Phi\left[\frac{CPU_m^{\mathcal{S}} - \sum_{i\in N^{\mathcal{V}}} \mu_i \cdot \mathbf{x}}{\sqrt{\sum_{i\in N^{\mathcal{V}}} \sigma_i^2 \cdot \mathbf{x}^2}}\right] \geqslant \pi_1, \quad \forall m \in N^{\mathcal{S}} \tag{18}$$

equivalent to:

$$\frac{CPU_m^{\mathcal{S}} - \sum_{i\in N^{\mathcal{V}}} \mu_i \cdot \mathbf{x}}{\sqrt{\sum_{i\in N^{\mathcal{V}}} \sigma_i^2 \cdot \mathbf{x}^2}} \geqslant \Phi^{-1}(\pi_1), \quad \forall m \in N^{\mathcal{S}} \tag{19}$$

Constraints (19) can be easily transformed to the expected constraints (15). Chance constraints (14) can be approximated into (16) using the same proof steps. ■

Since the obtained constraints (15) and (16) still make the resolution of (1) hard, we propose to transform them by introducing a continuous variable $\mathbf{s}_1^m \geqslant 0$ for constraints (15):

$$\sum_{i\in N^{\mathcal{V}}} \sigma_i^2 \cdot \mathbf{x}^2 \leqslant (\mathbf{s}_1^m)^2, \quad \forall m \in N^{\mathcal{S}} \tag{20}$$

$$\sum_{i\in N^{\mathcal{V}}} \mu_i \cdot \mathbf{x} + \Phi^{-1}(\pi_1) \cdot \mathbf{s}_1^m \leqslant CPU_m^{\mathcal{S}}, \forall m \in N^{\mathcal{S}} \tag{21}$$

Similarly, constraints (16) can be transformed by introducing a continuous variable $\mathbf{s}_2^{(m,n)} \geqslant 0$:

$$\sum_{i\in N^{\mathcal{V}}} \sigma_{(i,j)}^2 \cdot \mathbf{y}^2 \leqslant \left[\mathbf{s}_2^{(m,n)}\right]^2, \quad \forall (m,n) \in P^{\mathcal{S}} \tag{22}$$

$$\sum_{i\in N^{\mathcal{V}}}\sum_{j\in N^{\mathcal{V}}} \mu_{(i,j)} \cdot \mathbf{y} + \Phi^{-1}(\pi_2) \cdot \mathbf{s}_2^{(m,n)} \leqslant BW_{(m,n)}^{\mathcal{S}}, \quad \forall (m,n) \in P^{\mathcal{S}} \tag{23}$$

Given Proposition 1, we transform the CCP model to its deterministic equivalent formulation, namely the Second-Order Cone Programming (SOCP) model:

Model SOCP Objective: (1)
Subject to: $(2)-(5),(8)-(12),(20)-(23)$

IV. Numerical Simulations

We evaluate performance through numerical simulations using a commercial off-the-shelf PC with an Intel Core CPU of 1.60 GHz and 8 GB RAM. The proposed models are solved using the CPLEX 22.1. We generate VN requests with a maximum of 10 nodes and 15 links and SN graphs with up to 50 nodes and 200 links. For each virtual node, we randomly generate the associated required CPU resources in the [5, 20] range. For each physical node, we randomly generate an available amount of CPU in the range [50, 100]. For each virtual link, we randomly generate a bandwidth request in the range [5, 20] Mbps, and each available physical link has radom bandwidth resources in the range [50, 100] Mbps. The simulation duration is up to 10,000 time units with VN requests following a Poisson process arrival with an average rate λ of 1 arrival per 100 time units and an exponential service rate μ of 1 departure every 750 time units. For Gaussian distributions, the mean values of CPU and bandwidth

requirements correspond to the above generation. The standard deviations are set to 2 (using this value for the normal distribution, with $2\sigma = 4$, corresponds to 95% of the values for the generated samples to behave with an uncertainty amounting to about 25% in input requests and 40% uncertainty for 99.7% of the samples at 3σ). Confidence levels in chance constraints are reported to provide insight into these variations in average values obtained with 10 independent runs.

A. Impact of Anti-affinity Constraints

We examine our methods by first investigating the impact of the anti-affinity constraints on the acceptance ratio using the deterministic ILP. Fig. 1 depicts the evolution of the acceptance ratio with increasing received VN requests with anti-affinity constraints. We consider two scenarios: one with significant (80% of the received VNs) generations of anti-affinity constraints (simultaneously for nodes and links) and the second with weak anti-affinity constraints (20% of the received VNs). We use 10 runs for each performance point in both scenarios and the corresponding average for the acceptance ratio evaluation. The observed acceptance ratio is 80% for the highly constrained scenario 1 and as expected a much better 91% acceptance ratio for the less constrained scenario 2. This demonstrates the efficiency of the proposed algorithm that finds embedding solutions despite the stringent affinity and anti-affinity constraints, within and across VNs, that induce higher resource (nodes and links) utilization.

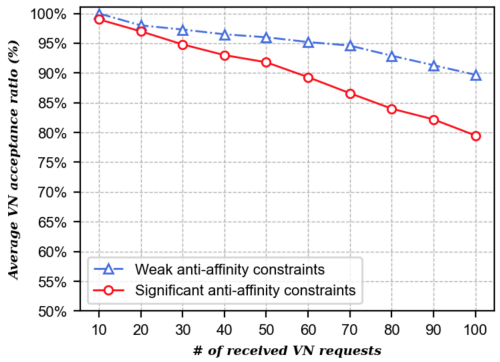

Fig. 1: VN acceptance ratios under significant and weak anti-affinity constraints by the deterministic ILP.

B. Impact of Confidence Levels in Chance Constraints

We now test the impact of the confidence levels π_1 and π_2 on acceptance ratios based on the SOCP model. We consider an SN of 30 nodes and 134 links confronted with 50 VN requests with a significant number of generated anti-affinity constraints (80% of the received VNs). The curve for π_1 in Figure 2 is obtained by first varying π_1 from 80% to 99% in confidence levels while fixing π_2 at a given value. We repeat the π_1 assessment for each ensuing fixed value of π_2 until all values of π_2 in the interval [80%, 99%] are spanned. The curve for π_2 is obtained similarly, by now fixing the value of π_1. For each fixed value of π_1, π_2 will vary from 80% to 99% in confidence levels in order to obtain a single point for the π_2 curve. This time the abscissa values correspond to the successive values of π_1 from 80% to 99%. The plots report the averaged acceptance ratio for each point in the curves. The acceptance ratios in the worst case decrease to 86% from 88% as the requirement for confidence levels becomes more stringent, specifically at the highest value of 99% confidence level. The small performance loss in acceptance ratio of 2% (from 88% to 86%) illustrates the robustness of the SOCP algorithm for increasing confidence level requirements concerning uncertainty management. This robustness of the SOCP model is due to expressions in the mathematical model constraints (21) (resp. 23) that make available fewer resources via $\phi^{-1}(\pi)$ that acts as an increasing function for increasing values of π causing less embedding success and thus decreasing acceptance ratio. Improving the robustness of the obtained solutions requires an increase in confidence levels but induces lower acceptance ratios. This result also informs service providers that they need to trade off robustness and acceptance ratios. The more important decreasing trend for π_1 is due to fewer nodes than links (paths) in the SN.

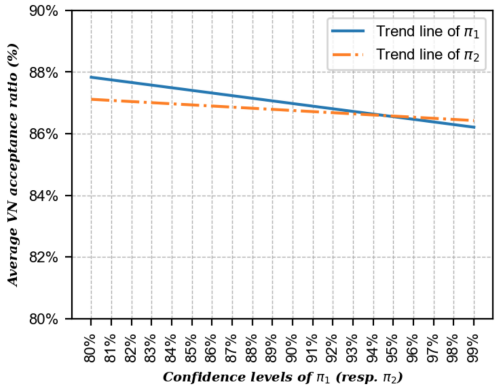

Fig. 2: VN acceptance ratios under different confidence levels π_1 and π_2 in the CCP model.

C. Comparison to the State-of-the-Art Method

To benchmark our approaches, we compare them with the only robust solution, i.e., the τ-RP model (9) in **Appendix**. We use physical networks with 30 nodes and 94 links subject to a significant number of anti-affinity constraints for 5,000 time units. Figure 3 depicts resource usage rates (in terms of servers and links) by the deterministic ILP, SOCP (using 95% for both π_1 and π_2), and τ-RP model (with τ calculated using (28). One can observe that the deterministic ILP-based solution, assuming perfect knowledge of the VN requests, therefore unaware of the inherent demand uncertainty in the VN resource requirements, would underestimate the needs and erroneously expect to use only 57% of available servers and 23% of available links in the SN. Taking into account uncertainty, the SOCP model uses

70% of the servers and 29% of the links in the SN, while the τ-RP method needs 85% servers and 34% links. This also clearly highlights that our proposed approximation (SOCP) outperforms the state-of-the-art τ-RP method that overestimates the needs in servers and links while SOCP uses fewer resources to meet target acceptance ratios. The stochastic methods protect service providers from SLA violations as seen around 5,000 time units in the reported results corresponding to high and nominal load in VN requests.

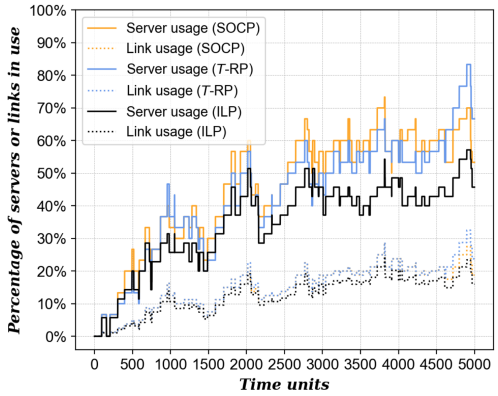

Fig. 3: Node/link usage rates by ILP, SOCP, τ-RP.

This claim on the SOCP method is verified by analyzing the performance results jointly with the achieved VN requests acceptance ratio and the algorithms convergence speed. SOCP outperforms τ-RP across all metrics. Both extend the ILP with uncertainty protection capabilities and the needed margins in required hosting resources to minimize SLA violations. The stochastic approaches also provide insights depicted in Figure 3 that an additional proportion of resources is required when VN demands are uncertain (29% instead of 20% in link resources compared with the values indicated by the ILP and 70% instead of 55% in node resources). We rely on acceptance ratio and convergence time to further compare the performance of SOCP with τ-RP with additional simulations. Tab. II reveals SOCP advantage in acceptance ratio compared with τ-RP (with a gap close to 1.5% for different confidence levels). SOCP converges in less than 4 seconds, while τ-RP finds feasible solutions in at best 18 seconds. Hence, SOCP offers service providers robust capabilities to handle uncertainties without compromising computational efficiency, unlike τ-RP.

V. Conclusions and Future Work

We addressed virtual network embedding faced with uncertain, integrating node and link affinity and anti-affinity constraints within a VN request and across different VN requests. We provided first an ILP-based approach for the case of exactly known clients' demands with affinity (e.g., co-localization) and anti-affinity (e.g., separation or isolation) constraints in nodes and links. For VN requests that are typically uncertain in required hosting nodes (e.g., CPU) and links (e.g., bandwidth), we propose a stochastic optimization approach based on chance constraints to provide robust solutions for the VNE problem under uncertainty. We confirmed the relevance of the stochastic approach through performance evaluation and a comparison with a state-of-the-art robust method. Our method provides insight into service providers' additional resources and investment needed to absorb or handle uncertainty in the clients' VNE demands. Future work will address stochastic VNE when the CPU and bandwidth probability distribution functions are not known in advance, using Distributionally Robust Optimization for assumption-free approximation solutions.

TABLE II: SOCP and τ-RP comparison

Confidence level		Acceptance ratio (%)			Convergence time (s)	
π_1	π_2	SOCP	τ-RP	*Gap*[a]	SOCP	τ-RP
98%	98%	86.4	85.2	**1.3%**	3.8	18.6
	95%	88.6	87.8	**0.9%**	3.8	18.5
	90%	90.4	89.2	**1.3%**	3.8	18.7
95%	98%	88.4	87.2	**1.3%**	3.8	18.6
	95%	88.8	87.7	**1.1%**	3.8	18.8
	90%	89.8	88.4	**1.3%**	3.8	18.7
90%	98%	89.8	88.0	**1.9%**	3.8	18.6
	95%	90.2	88.6	**1.7%**	3.8	18.7
	90%	90.8	89.4	**1.5%**	3.8	18.5

[a]Acceptance Ratio Gap = (SOCP - τ-RP) / SOCP * 100%

Appendix: τ-Robust Method

To realize the comparison of our SOCP approach and the τ-Robust method in (9), we introduce continuous variables $\mathbf{u}^m \geqslant 0$ and $\mathbf{v}_i^m \geqslant 0$ (denoted as $\mathbf{u}_1$ and $\mathbf{v}_1$ for simplicity) for virtual nodes (CPU), and continuous variables $\mathbf{u}^{(m,n)} \geqslant 0$ and $\mathbf{v}_{(i,j)}^{(m,n)} \geqslant 0$ (denoted as $\mathbf{u}_2$ and $\mathbf{v}_2$) for virtual links (bandwidth). Then, we obtain the following inequalities consistent with (9):

$$\sum_{i \in N^{\mathcal{V}}} \left(\overline{CPU}_i^{\mathcal{V}} \cdot \mathbf{x} + \mathbf{v}_1 \right) + \tau_1 \cdot \mathbf{u}_1 \leqslant CPU_m^{\mathcal{S}}, \quad \forall m \in N^{\mathcal{S}} \tag{24}$$

$$\mathbf{u}_1 + \mathbf{v}_1 \geqslant \widehat{CPU}_i^{\mathcal{V}} \cdot \mathbf{x}, \quad i \in N^{\mathcal{V}}, m \in N^{\mathcal{S}} \tag{25}$$

$$\sum_{i \in N^{\mathcal{V}}} \sum_{j \in N^{\mathcal{V}}} \left(\overline{BW}_{(i,j)}^{\mathcal{V}} \cdot \mathbf{y} + \mathbf{v}_2 \right) + \tau_2 \cdot \mathbf{u}_2 \leqslant BW_{(m,n)}^{\mathcal{S}}, \quad \forall (m,n) \in P^{\mathcal{S}} \tag{26}$$

$$\mathbf{u}_2 + \mathbf{v}_2 \geqslant \widehat{BW}_{(i,j)}^{\mathcal{V}} \cdot \mathbf{y}, (i,j) \in L^{\mathcal{V}}, (m,n) \in P^{\mathcal{S}} \tag{27}$$

where $\overline{CPU}_i^{\mathcal{V}}$ and $\overline{BW}_{(i,j)}^{\mathcal{V}}$ represent the mean values of uncertain CPU and bandwidth, while $\widehat{CPU}_i^{\mathcal{V}}$ and $\widehat{BW}_{(i,j)}^{\mathcal{V}}$ denote their standard deviation. This transformation leads to the τ-Robust Programming model:

Model τ-RP Objective: (1)
Subject to: $(2)-(5), (8)-(12), (24)-(27)$

To determine the value of τ_1 (resp. τ_2) that can transform the chance constraints with a probability of π_1 (resp. π_2), (20) proposed the following relationship:

$$\pi_1 = 1 - \Phi\left(\frac{\tau_1 - 1}{\sqrt{|N^{\mathcal{V}}|}} \right) \tag{28}$$

REFERENCES

[1] A. Fischer, J. F. Botero, M. T. Beck, H. De Meer, and X. Hesselbach, "Virtual network embedding: A survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 1888–1906, 2013.

[2] H. Cao, H. Zhu, and L. Yang, "Collaborative attributes and resources for single-stage virtual network mapping in network virtualization," *Journal of Communications and Networks*, vol. 22, no. 1, pp. 61–71, 2019.

[3] M. Mechtri, M. Hadji, and D. Zeghlache, "Exact and heuristic resource mapping algorithms for distributed and hybrid clouds," *IEEE Transactions on Cloud Computing*, vol. 5, no. 4, pp. 681–696, 2017.

[4] J. Liu, W. Yao, C. Wang, and Q. Yang, "Provisioning network slice for mobile content delivery in uncertain mec environment," *Computer Networks*, vol. 224, p. 109629, 2023.

[5] A. Fendt, C. Mannweiler, L. C. Schmelz, and B. Bauer, "An efficient model for mobile network slice embedding under resource uncertainty," in *2019 16th International Symposium on Wireless Communication Systems (ISWCS)*. IEEE, 2019, pp. 602–606.

[6] B. Farkiani, B. Bakhshi, and S. A. MirHassani, "Stochastic virtual network embedding via accelerated benders decomposition," *Future Generation Computer Systems*, vol. 94, pp. 199–213, 2019.

[7] X. Cheng, Y. Wu, G. Min, and A. Y. Zomaya, "Network function virtualization in dynamic networks: A stochastic perspective," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 10, pp. 2218–2232, 2018.

[8] G. Chochlidakis and V. Friderikos, "Robust virtual network embedding for mobile networks," in *2015 IEEE 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. IEEE, 2015, pp. 1867–1871.

[9] S. Coniglio, A. Koster, and M. Tieves, "Data uncertainty in virtual network embedding: robust optimization and protection levels," *Journal of Network and Systems Management*, vol. 24, pp. 681–710, 2016.

[10] M. Tieves, "Discrete and robust optimization approaches to network design with compression and virtual network embedding," Ph.D. dissertation, Dissertation, RWTH Aachen University, 2016, 2016.

[11] A. Baumgartner, T. Bauschert, F. D'Andreagiovanni, and V. S. Reddy, "Towards robust network slice design under correlated demand uncertainties," in *2018 IEEE International Conference on Communications (ICC)*. IEEE, 2018, pp. 1–7.

[12] J. R. Birge and F. Louveaux, *Introduction to stochastic programming*. Springer Science & Business Media, 2011.

[13] Z. Allybokus, N. Perrot, J. Leguay, L. Maggi, and E. Gourdin, "Virtual function placement for service chaining with partial orders and anti-affinity rules," *Networks*, vol. 71, no. 2, pp. 97–106, 2018.

[14] M. Gao, B. Addis, M. Bouet, and S. Secci, "Optimal orchestration of virtual network functions," *Computer Networks*, vol. 142, pp. 108–127, 2018.

[15] Z. Li, Z. Lu, S. Deng, and X. Gao, "A self-adaptive virtual network embedding algorithm based on software-defined networks," *IEEE Transactions on Network and Service Management*, vol. 16, no. 1, pp. 362–373, 2018.

[16] N. Bouten, M. Claeys, R. Mijumbi, J. Famaey, S. Latré, and J. Serrat, "Semantic validation of affinity constrained service function chain requests," in *2016 IEEE NetSoft Conference and Workshops (NetSoft)*. IEEE, 2016, pp. 202–210.

[17] N. Bouten, R. Mijumbi, J. Serrat, J. Famaey, S. Latré, and F. De Turck, "Semantically enhanced mapping algorithm for affinity-constrained service function chain requests," *IEEE Transactions on Network and Service Management*, vol. 14, no. 2, pp. 317–331, 2017.

[18] ETSI, "Etsi gr nfv 003: Network functions virtualisation (nfv); terminology for main concepts in nfv," ETSI, Tech. Rep., 2023.

[19] M. L. Bentaha, O. Battaïa, A. Dolgui, and S. J. Hu, "Second order conic approximation for disassembly line design with joint probabilistic constraints," *European Journal of Operational Research*, vol. 247, no. 3, pp. 957–967, 2015.

[20] D. Bertsimas and M. Sim, "The price of robustness," *Operations Research*, vol. 52, no. 1, pp. 35–53, 2004.