# Traffic Routing in Stochastic Network Function Virtualization Networks

Song Yang[a,*], Fan Li[a], Stojan Trajanovski[b], Xiaoming Fu[c]

[a]*School of Computer Science and Technology, Beijing Institute of Technology, China*
[b]*Microsoft, London, United Kingdom*
[c]*Institute of Computer Science, University of Göttingen, Göttingen, Germany*

## Abstract

Network Function Virtualization (NFV) is emerging as an attractive solution, which can transform complex network functions from the dedicated hardware implementations into software instances running in a virtualized environment. In NFV, the requested service is implemented by a sequence of Virtual Network Functions (VNF) that can run on generic servers by leveraging the virtualization technology. These VNFs are placed in a predefined order, which is also known as the Service Function Chaining (SFC). While most of the existing work on traffic routing in NFV networks assume deterministic link delay and bandwidth, the real-life network usually behaves in a stochastic manner, due to e.g., inaccurate data, expired exchanged information, insufficient estimation to the network. Motivated by this, we consider the stochastic NFV networks, where the link bandwidth and delay are assumed to be random variables and their cumulative distribution functions are known. We first study how to calculate the delay and bandwidth value in an SFC such that their realizing probabilities are satisfied. Subsequently, we formally define the traffic routing problem in stochastic NFV networks and prove it is NP-hard. We present an exact solution and a tunable heuristic to solve this problem. The proposed heuristic is a sampling-based algorithm, and it leverages the Tunable Accuracy Multiple Constraints Routing Algorithm (TAMCRA) to find a multi-constraint path for each adjacent VNF pair. It dynamically adjusts the link weights as well as delay and bandwidth realizing probability constraint after finding the path for each VNF pair so that the cumulated probabilities will not violate the specified values. Finally, we evaluate the performance of the proposed algorithms via extensive simulations.

*Keywords:* Network function virtualization, routing, stochastic, delay, bandwidth

*Corresponding author. Tel.: +8618701190071
*Email addresses:* `S.Yang@bit.edu.cn` (Song Yang), `fli@bit.edu.cn` (Fan Li), `sttrajan@microsoft.com` (Stojan Trajanovski), `Fu@cs.uni-goettingen.de` (Xiaoming Fu)

## 1. Introduction

Nowadays, data communication networks have been witnessing exponential growth in users' data traffic. According to the forecast from Cisco [1], the global IP traffic will reach 3.3 ZB by 2021, with a Compound Annual Growth Rate (CAGR) of 24 percent since 2016. In the traditional network services provisioning paradigm, network functions (e.g., firewall or web proxy) which are also called middleboxes are usually implemented by dedicated hardware appliances. Needless to say, it is costly to deploy these hardware middleboxes due to their high design and production cost and also these middleboxes need to be configured and managed manually, which further increases the costs of service providers. Therefore, the traditional network service paradigm fails to keep pace with satisfying the ever-increasing users' QoS requirements from the perspective of CAPital EXpenditures (CAPEX) and OPerational EXpenditures (OPEX), which poses a big challenge to network service providers.

Network Function Virtualization (NFV) which is first proposed by European Telecommunications Standards Institute (ETSI) [2] in 2012 has emerged as an appealing solution, since it offers replacement of dedicated hardware implementations with software instances running in a virtualized environment. In NFV, the requested service is implemented by a sequence of Virtual Network Functions (VNF) that can run on generic servers by leveraging the virtualization technology. Service Function Chaining (SFC) is therefore defined as a chain-ordered set of placed VNFs that handles the traffic of the delivery and control of a specific application [3]. NFV is therefore able to allocate network resources in a more scalable and elastic manner, offer a more efficient and agile management and operation mechanism for network functions, leading to a large reduction of the overall costs.

Traffic routing (and VNF placement)[1] in NFV networks is a fundamental issue to construct an SFC without violating the QoS requirement, and it has been extensively studied in the recent works [4–6] where the link weight is assumed to be deterministic. However, in many real-life networks (e.g., data communication networks, wireless sensor networks [7]), the link weight such as bandwidth and delay usually varies and is uncertain [8]. These uncertainties [9, 10] usually arise from inaccurate data, expired exchanged information or insufficient estimation to the network. For example, especially in large networks, it is difficult to obtain an accurate view on the link characteristics like bandwidth utilization or latency, because their dynamics are usually of the same order as the time it would take to distribute information on the link state throughout the network [8]. Similarly in Software Defined Networks (SDN) [11], the controller collects network statistics periodically to achieve the link conditions in the network. The dynamics such as configuration changes [12] may lead to inaccurate/stochastic network

[1] As we will show later, the traffic routing problem alone in (stochastic) NFV networks is already NP-hard, hence jointly considering traffic routing and VNF placement will be more difficult to solve. We therefore only consider the traffic routing problem in stochastic NFV networks in this paper.

state information in terms of delay or bandwidth. Another example is that the delay and available bandwidth are affected by diurnal patterns, interference in wireless networks [13], or by failure and maintenance events.

Based on the above concerns, we assume the link delay and bandwidth are random variables (stochastic) and assume their cumulative distribution functions (CDF) are known, which can be easily achieved based on historical data. Suppose that the VNFs are placed in the network, we investigate how to construct an SFC by finding the appropriate path for each requested VNF pair. To the best of our knowledge, this work is the first to address the traffic routing problem in stochastic NFV networks. Our key contributions are as follows:

- We mathematically model the stochastic link weight in terms of delay and bandwidth. More specifically, given that the link delay and bandwidth follow a known distribution, we show how to calculate the delay and bandwidth in an SFC such that their realizing probabilities satisfy the required values.
- We define the Traffic Routing problem in Stochastic NFV Networks and prove it to be NP-hard. We further formulate this problem as an exact optimization solution.
- We devise a tunable heuristic that first discretizes the network by $k$-samplings, and then dynamically applies Tunable Accuracy Multiple Constraints Routing Algorithm (TAMCRA) to find the multi-constraint path for each adjacent VNF pair.

The remainder of this paper is organized as follows. Section 2 presents the related work. Section 3 introduces stochastic link weight model. Section 4 defines the Traffic Routing in Stochastic NFV Networks (TRSN) problem and presents an exact solution to formulate it. In Section 5, we propose a tunable sampling-based heuristic to solve the TRSN problem. Section 6 provides the simulation results and we conclude in Section 7.

## 2. Related Work

In this section, we will divide and present the related work about traffic routing and VNF placement in both deterministic and stochastic NFV networks. In deterministic NFV networks, we will further present the related work on three categories, namely, (1) traffic routing in deterministic NFV networks, where VNFs are assumed to be placed on networks, (2) VNF Placement in Deterministic NFV Networks, where the path between each node pair is known, (3) (jointly) VNF Placement and Routing in Deterministic NFV Networks.

### *2.1. Traffic Routing and VNF Placement in Deterministic NFV Networks*

A comprehensive survey about NFV can be found in [5, 14, 15]. [4, 16] provides a survey about resource allocation in NFV networks and [6] presents a survey about traffic steering in NFV networks. Laghrissi and Taleb [17] present a survey of Virtual Machine placement and VNF placement.

### *2.1.1. Traffic Routing in Deterministic NFV Networks*

Suppose that the VNFs are placed in networks, the traffic routing problem in NFV networks refers to find a path among each requested VNF pair. This problem can be proved to be NP-hard by a reduction to the NP-hard Hamiltonian path problem [18]. Therefore, only approximation algorithm or heuristic can exist to solve it in polynomial time. Dwaraki and Wolf [19] devise a layered-graph based heuristic to solve the delay-aware traffic routing problem in NFV networks. Wang *et al.* [20] propose a distributed Alternating Direction Method of Multipliers (ADMM) algorithm to solve the traffic routing problem in NFV networks for multiple requests. The proposed algorithm in [20] is proved to have a fixed coverage rate. Yu *et al.* [21] propose a Fully-Polynomial Time Approximation Scheme (FPTAS)[2] to find the multipath between each VNF pair such that during an arbitrary single service failure, at most a portion of bandwidth is lost for each request. Gao and Rouskas [22] present an online competitive traffic routing algorithm based on the Shortest Path Tour problem to minimize the congestion.

### *2.1.2. VNF Placement in Deterministic NFV Networks*

When the path between each node pair is known/fixed, Ma *et al.* [23] study the VNF placement problem to minimize the maximum link load in three cases: (1) when there is no dependency between VNFs, (2) there is a total dependency order on the VNF set, and (3) there exists dependency among a subset of VNFs. They propose a polynomial-time algorithm for case (1), and prove that cases (2) and (3) are NP-hard. To solve them, a dynamic programming and an efficient heuristic are proposed to solve (2) and (3), respectively. Pham *et al.* [24] propose a sampling-based Markov approximation algorithm to jointly minimize the operational and network traffic costs for the VNF placement problem. Kuo *et al.* [25] relax/approximate the VNF placement problem based on the intuition that placing VNFs on a shorter path consumes less link bandwidth, but might also reduce VM reuse opportunities; reusing more VMs might lead to a longer path, and so it consumes more link bandwidth. Tomassilli *et al.* [26] study the VNF placement problem with the aim of minimizing total costs for servicing a set of requests. By transforming this problem into a hitting-cut problem, Tomassilli *et al.* [26] propose two logarithmic factor approximation algorithms. The first algorithm is based on LP rounding and the second one is a greedy algorithm. Marotta *et al.* [27] devise a three-phase heuristic for the robust power-aware VNF placement problem by considering the uncertainty of the demand. Song *et al.* [28] study the VNF placement problem in 5G edge networks by considering the user's mobility. They [28] first propose a user grouping model based on users' context geographical information and then define (and compute the optimum number of the) clusters to minimize the end-to-end delay of net-

[2]An FPTAS has a time complexity that is polynomial in both the problem size and $\frac{1}{\epsilon}$ and produces a solution that is within a factor $(1+\epsilon)$ of the optimal solution (or $(1-\epsilon)$ for maximization problems).

work services. Subsequently, a graph partitioning algorithm assigning VNFs to clusters in the edge network is presented to minimize user movement between clusters and optimize the data rate that users lose due to VNF migration.

#### *2.1.3. VNF Placement and Routing in Deterministic NFV Networks*

As we see above, only consider traffic routing or VNF placement in NFV networks has already been NP-hard to solve, therefore jointly considering the VNF placement and routing in NFV networks will make this problem ever harder. Ma *et al.* [23] prove that jointly considering the VNF placement and routing problem is NP-hard even under the non-ordered VNF dependence case by a reduction to the NP-hard Hamiltonian Cycle problem [18]. Guo *et al.* [29] propose a randomized approximation algorithm when the traffic matrix is known in advance and a competitive online algorithm when the future arriving traffic is not known. However, they assume that one configuration in data centers consists of one VNF placement and one routing path solution, and a (limited) set of configurations is given. Qu *et al.* [30] consider the VNF transmission and processing delays, and formulate the VNF Placement and Routing problem as a Mixed Integer Linear Program (MILP). However, they assume that the virtual link between two physical nodes can at most process one traffic flow at a time. Li *et al.* [31] address the latency-aware middlebox routing and placement problem by leveraging a packet queuing model. However, a fixed link transmission delay is assumed in their model. Hamann and Fischer [32] formulate the VNF placement and routing problem as an ILP where a set of $k$ paths between each node pair in the network are precalculated and known. Yang *et al.* [33] propose an approximation algorithm to solve the VNF placement and routing problem in edge clouds. The proposed approximation in [33] leverages randomized rounding technology and assumes that the paths between each node pair are known/given.

Apart from that, there is also work about availability-aware VNF provisioning [34, 35], VNF placement in operator data centers [36], the resource allocation of NFV in 5G mobile networks [37], IoT [38] and Radio Access Networks (RAN) [39], etc. Due to the complex nature of the VNF placement and/or traffic routing problem in NFV networks, we observe that it is difficult to design a general approximation algorithm to solve it. Instead, the current literature mainly simplify some problem inputs or constraints (e.g., assuming the paths between node pair are known [33] or multipath routing [21]) and devise proper approximation/heuristic algorithms. Nevertheless, all the above work assume deterministic link weight, therefore they cannot solve the proposed problem in this paper.

### *2.2. Traffic Routing and VNF Placement in Stochastic (NFV) Networks*

So far, the above literature assume that the link delay and bandwidth are deterministic, and this assumption is not always very true in practice. The reason is that in practice, the network usually behaves in a stochastic manner, which is mainly caused by inaccurate data, expired exchanged information or insufficient estimation [9, 10]. In what follows, we will provide the related work about routing and VNF placement on stochastic (NFV) networks:

**Traffic Routing in Stochastic Networks:** A survey about traffic planning models and routing algorithms in stochastic networks can be found in [8]. Lorenz and Orda [40] assume that each link $l$ has a function $p_l(d)$ that represents the probability that link $(l)$ introduces a delay of no more than $d$ time units. This so-called Delay-Based Routing (DBR) problem is to find a path that has the biggest probability of not exceeding $D$. Lorenz and Orda prove that the DBR problem is NP-hard, and by decomposing the end-to-end delay constraint $D$ into local delay constraints, they manage to develop an FPTAS. However, the proposed algorithms in [40] cannot solve the proposed problem in this paper. The reason is that it is required to find a path for each VNF pair and also the bandwidth requirement should be taken into account. Assuming the link's bandwidth and delay follows a log-concave distribution, we [41] propose a polynomial-time convex optimization formulation to find the maximum flow in the so-called stochastic networks. When the delay constraint is imposed on each path, the maximum flow problem is NP-hard. To solve it, we [41] propose an approximation algorithm and a tunable heuristic algorithm. However, the proposed approximation algorithm in [41] is to find multipath and hence cannot solve the considered problem in this paper accordingly.

**VNF Placement in Stochastic NFV Networks:** The most relevant work with us is [42], but it assumes that the service rate demands and available amounts of wireless resources of nodes are random and tackles the VNF placement problem in the so-called dynamic networks. Cheng *et al.* [42] further develop a distributed computing framework with two-level decomposition to solve this problem. Zeng *et al.* [43] present an online VNF placement and data packets scheduling framework in edge clouds by applying the Lyapunov optimization theory. However, the routing issues are not considered in [43]. While our work is on traffic routing in stochastic networks, and therefore this paper is orthogonal with [42, 43].

Moreover, Miao *et al.* [44] propose an analytical model based on stochastic network calculus to calculate the delay bound of the stochastic NFV networks. They further leverage the property of convolution associativity and leftover service technologies to calculate the available resources for VNF networks. However, [44] tackles the resource allocation problem in stochastic NFV networks from the system side, and routing issues are not considered.

In all, we conclude that the traffic routing problem in stochastic NFV networks has never been tackled by the existing literature, which remains our contribution in this paper. To that end, we first mathematically model the stochastic link delay and bandwidth in stochastic NFV networks and present an exact algorithm as well as a tunable heuristic MCTR for this problem. We also conduct simulations to evaluate the performance of the proposed algorithms.

## 3. Stochastic Link Weight

For each link $l \in \mathcal{L}$, we assume that its delay and bandwidth are stochastic and follow a given distribution. More specifically, it is assumed that the allocated bandwidth has a known cumulative distribution function $c_l(b)$, which indicates the probability of being able to allocate $b$ units of bandwidth; the allocated delay follows a known cumulative distribution function $p_l(d)$, which gives the probability of transporting data with no more than $d_l$ units of delay. We assume that the bandwidth allocated on each link $l$ ranges from 0 to $b_l^{\max}$, and the delay on each link $l$ ranges from $d_l^{\min} > 0$ to $d_l^{\max}$. For ease of notation we will sometimes write $c_l$ and $p_l$ to denote $c_l(b_l)$ and $p_l(d_l)$. The traffic request set is denoted by $R$, and for each $r(F, B, P_B, D, P_D) \in R$, $F$ symbolizes a set of required ordered VNFs, $P_B$ refers to the probability to realize $B$ available bandwidth, and $P_D$ represents the probability of realizing a total traversing delay $D$. Consequently, by finding an appropriate path between each VNF pair which is located on different nodes (if the VNF pair are placed on the same node, then it is not necessary to find path), we get an SFC for request $r$. The notations used in this paper are summarized in Table 1.

Table 1: Notations.

| Notation | Description |
|---|---|
| $\mathcal{G}(\mathcal{N}, \mathcal{L})$ | A network $\mathcal{G}$ with node set $\mathcal{N}$ and link set $\mathcal{L}$ |
| $c_l(b)$ | CDF of being able to allocate $b$ units of bandwidth |
| $p_l(d)$ | CDF of transporting data with $d$ units of delay |
| $R$ | Traffic request set and for each $r(F, B, P_B, D, P_D) \in R$, $F$ symbolizes a set of required ordered VNFs, $P_B$ refers to the probability to realize $B$ available bandwidth, and $P_D$ represents the probability of realizing a total traversing delay $D$ |
| $\pi(f)$ | The set of nodes on which VNF $f$ is placed |
| $\Psi(n)$ | The set of VNFs placed on $n$ |
| $k$ | The number of samplings used in the heuristic |
| $q$ | The number of stored path per node in MCTR, where $q = ck$ |
| $W_{u,v}^{r,f_i,f_j}$ | A boolean variable and it is 1 (true) if $r$'s requested VNF pair $(f_i, f_j)$ traverses link $(u, v)$, and 0 (false) otherwise |
| $Z_{u,v}^{r,f_i,f_j}$ | A float variable indicating the allocated delay on link $(u, v)$ for $r$'s requested VNF pair $(f_i, f_j)$ |

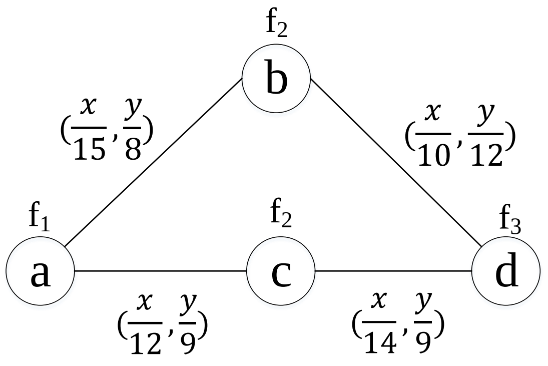
Let us consider the example from Fig. 1, where $f_1$ is placed on node $a$, $f_2$ is placed on node $b$ and $c$, and $f_3$ is placed on node $d$. The allocated link bandwidth $x$ and delay $y$ are assumed to follow a uniform distribution, and their CDFs are labelled above each link. Now, suppose there is a request $r(\{f_1, f_2, f_3\}, 5, 0.35, 16, 0.7)$. According to Fig. 1, if we follow the path $a - b - d$, then the probability of realizing available bandwidth of **at least** 5 is $(1 - \frac{5}{15}) \cdot (1 - \frac{5}{10}) \approx 0.33 < P_B$, and the probability of realizing a delay **at most**

16 is to solve the following equation:

$$\begin{aligned} \max \quad & \frac{y_1 \cdot y_2}{96} \\ \text{subject to:} \quad & y_1 + y_2 <= 16 \end{aligned} \tag{1}$$

The objective is achieved when $y_1 = y_2 = 8$ (theoretically can be derived from the inequality of geometric arithmetic [45]) which approximately leads to realizing probability of $\frac{8*8}{96} \approx 0.67 < P_D$.

However, both the requested $P_B$ and $P_D$ are violated if using this path, which indicates that this is not a feasible path to satisfy $r$. Similarly, we can calculate that if we choose the path $a{-}c{-}d$, the probability of realizing available bandwidth of **at least** 5 is $(1 - \frac{5}{12}) \cdot (1 - \frac{5}{14}) \approx 0.375 > P_B$ and the probability of realizing a delay of **at most** 16 is approximately 0.81, which is greater than $P_D$. As a result, we see that $a - c - d$ constitutes a feasible SFC and can satisfy $r$.



$f_2$
b
$(\frac{x}{15}, \frac{y}{8})$
$(\frac{x}{10}, \frac{y}{12})$
$f_1$
$f_2$
$f_3$
a
c
d
$(\frac{x}{12}, \frac{y}{9})$
$(\frac{x}{14}, \frac{y}{9})$

Figure 1: An Example of illustrating stochastic link bandwidth and delay.

## 4. Problem Definition and Formulation

### *4.1. Problem Definition and Complexity Analysis*

We consider a network $G(\mathcal{N}, \mathcal{L})$, where $\mathcal{N}$ represents a set of $N$ nodes and $\mathcal{L}$ denotes a set of $L$ links. For each link $l \in \mathcal{L}$ the allocated bandwidth has a known CDF $c_l(b_l)$, which gives the probability of being able to allocate no more than $b_l$ units of bandwidth. Moreover, if the possible bandwidth allocated on each link $l$ ranges from 0 to $b_l^{\max}$ $(0 < b_l^{\max})$, then the probability of allocating a bandwidth out of this range is 0, i.e. $c_l(b_l^{\max}) = 1$.

A certain number of VNFs are placed on network nodes, and for each $n \in \mathcal{N}$, we denote $\Psi(n)$ as the set of VNFs placed on it. Formally, the Traffic Routing in Stochastic NFV Networks (TRSN) problem can be defined as follows:

**Definition 1.** *For each request $r \in R$, the TRSN problem is to find route in each required VNF pair such that:*

- *The path between each VNF pair has* ***at least*** *$B$ bandwidth and the total probability to realize the bandwidth is* ***no less*** *than $P_B$.*
- *The total traversing delay in an SFC is* ***at most*** *$D$ and the probability to realize the delay is* ***no less*** *than $P_D$.*

**Theorem 2.** *The TRSN problem is NP-hard.*

**Proof 1.** *Let us first introduce the NP-hard Hamiltonian path problem [18]. A Hamiltonian path is a path in a graph that visits each node exactly once. For simplicity, we assume that the allocated bandwidth $x$ on each link $l$ follows*

$$x = \begin{cases} B & \textit{with probability=1} \\ 2B & \textit{with probability} < 1 \end{cases} \tag{2}$$

*and the allocated delay $y$ on each link $l$ follows*

$$y = \begin{cases} D_l & \textit{with probability=1} \\ D_l/\alpha & \textit{with probability} < 1 \end{cases} \tag{3}$$

*where $\alpha_l > 1$ symbolizes a coefficient for each link. We assume $N = |F|$, and on each node there is one distinct VNF placed on it. Now, suppose for a request $r(\{f_1, f_2, \ldots, f_N\}, B, 1, D, 1)$ where $D$ is set to $\sum_{l \in \mathcal{L}} D_l$, then the TRSN problem is equivalent to the Hamiltonian path problem. The proof is therefore complete.*

### *4.2. Problem Formulation*

In this section, we present an exact solution to formulate the TRSN problem. We start with some necessary notations and variables.

$R$: Traffic request set $R$.

$\pi(f)$: The set of nodes on which the VNF $f$ is placed.

$f_i, f_j$: The required VNF pair.

$W_{u,v}^{r,f_i,f_j}$: A boolean variable and it is 1 (true) if $r$'s requested VNF pair $(f_i, f_j)$ traverses link $(u, v)$, and 0 (false) otherwise.

$Z_{u,v}^{r,f_i,f_j}$: A float variable indicating the allocated delay on link $(u, v)$ for $r$'s requested VNF pair $(f_i, f_j)$.

**Constraints:**

**Routing constraints:**

$$\sum_{u \in \pi(f_i):(u,v) \in \mathcal{L}} W_{u,v}^{r,f_i,f_j} = 1 \quad \forall r \in R, (f_i, f_j) \in r, r \in R \tag{4}$$

$$\sum_{v \in \pi(f_j):(u,v) \in \mathcal{L}} W_{u,v}^{r,f_i,f_j} = 1 \quad \forall r \in R, (f_i, f_j) \in r, r \in R \tag{5}$$

$$\sum_{(u,v)\in\mathcal{L}:v\notin\pi(f_i)\cup\pi(f_j)} W_{u,v}^{r,f_i,f_j} = \sum_{(v,w)\in\mathcal{L}:v\notin\pi(f_i)\cup\pi(f_j)} W_{v,w}^{r,f_i,f_j} \quad \forall r\in R,(f_i,f_j)\in r \tag{6}$$

**Delay constraints:**

$$W_{u,v}^{r,f_i,f_j}\cdot M \geq Z_{u,v}^{r,f_i,f_j} \ \forall r\in R, f_i,f_j\in r,(u,v)\in\mathcal{L} \tag{7}$$

$$\sum_{f_i,f_j\in r}\sum_{(u,v)\in\mathcal{L}} Z_{u,v}^{r,f_i,f_j} \leq r.D \ \forall r\in R \tag{8}$$

$$\prod_{(u,v)\in\mathcal{L}} \max_{f_i,f_j\in r} p_{u,v}(Z_{u,v}^{r,f_i,f_j}) \geq P_D \quad \forall r\in R \tag{9}$$

**Bandwidth constraint:**

$$\prod_{f_i,f_j\in r}\prod_{(u,v)\in\mathcal{L}}(1-c_{u,v}(r.B)) \geq P_B \quad \forall r\in R \tag{10}$$

**Link capacity constraint:**

$$\sum_{r\in R,f_i,f_j\in r} W_{u,v}^{r,f_i,f_j}\cdot r.B \leq b_{(u,v)}^{max} \quad \forall (u,v)\in\mathcal{L} \tag{11}$$

There is no objective (needed) in the proposed exact formulation, but one could include the objective of e.g., minimizing the number of links used. Eqs. (4)-(6) are the multicommunity constraints that account for the routing from a source to a destination and ensures to find a path from the node where $f_i$ is located to the node where $f_j$ is placed. More specially, for each VNF pair $f_i$ and $f_j$ of request $r$, Eq. (4) ensures that if $f_i$ is placed on node $u$ and $r$ selects $u$ as the node to place $f_i$ and transfer data (since more than 1 nodes can host $f_i$ in the network), then the sum of outgoing traffic from $u$ is equal to 1. Eq. (5) ensures that if $f_j$ is placed on node $v$ and $r$ selects $v$ as the node to place $f_j$ and transfer data (since more than 1 nodes can host $f_j$ in the network), then the sum of incoming traffic to $v$ is equal to 1. Eq. (6) ensures that for any intermediate node $v$ which does not host $f_i$ and $f_j$ (by setting $v\notin\pi(f_i)\cup\pi(f_j)$), the sum of incoming traffic to $v$ is equal to its outgoing traffic. Eq. (7) applies big $M$-method to set the relation between $W$ and $Z$. More specifically, by setting $M$ as a large enough number, when $Z$ is greater than 0, then $W$ has to be 1, otherwise when $Z$ is equal to 0, then $W$ is forced to be 0. Eq. (8) ensures that the total delay during the entire SFC is no greater than the specified. Eq. (9)

ensures that the probability of realizing the total allocated delay is at least $P_D$. Eq. (10) ensures that the probability of realizing the link bandwidth is at least $P_B$. Eq. (11) ensures that the total allocated bandwidth on each link cannot exceed its maximum possible link capacity.

It is worthwhile to mention that the complexity of the proposed exact formulation depends on the convexity of link delay and bandwidth distribution. For example, when link CDF and CCDF are log-concave functions (e.g., exponential distribution), Eqs. 4-11 become a convex optimization formulation.

## 5. Multi-Constrained Traffic Routing Heuristic

Considering that the running time of the proposed exact solution in Section 4.2 is exponential, especially when the problem size is large, the exact solution cannot return a solution in an extremely long time. Therefore, we propose a fast heuristic, called Multi-Constrained Traffic Routing (MCTR) for the problem in this section. The idea of MCTR is that it first discretizes the network in terms of link delay and link bandwidth, and then runs a heuristic multi-constrained routing algorithm. MCTR has three main features: (1) MCTR is a sampling-based heuristic, that is, the more samplings are used to discretize the network, the more accurate that the solution returned by the heuristic will have, but more running time it will consume since the problem size is enlarged. (2) MCTR leverages the Tunable Accuracy Multiple Constraints Routing Algorithm (TAMCRA) to find the multi-constraint path for each adjacent VNF pair. TAMCRA [46] is scalable in the number of constraints and in the size of the graphs by applying Dijkstra-like algorithm, and that multiple constraints routing problems can be solved to a very high accuracy within a short time that increases linearly with. (3) MCTR dynamically adjusts the link weights and delay and bandwidth realizing probability constraint after finding the path for each VNF pair so that the cumulated probabilities will not exceed the specified. For completeness, we first give a formal definition of the multi-constrained path selection problem as follows [47]:

**Definition 3.** *Consider a network $\mathcal{G}(\mathcal{N}, \mathcal{L})$, where each link $l \in \mathcal{L}$ is specified by a link weight vector with $m$ additive QoS link weights $w_i(l) \geq 0$ for all $1 \leq i \leq m$. Given $m$ constraints $T_i$, where $1 \leq i \leq m$, the problem is to find a path $P$ from a source to a destination such that*

$$\sum_{l \in P} w_i(l) \leq T_i \tag{12}$$

A path that satisfies all $m$ constraints is referred to be a feasible path. The multi-constrained path selection problem is proved to be NP-hard [48], we therefore have chosen a heuristic for the multi-constrained routing problem.

Next, we will illustrate how to discretize the network. For a link, $k$ samples are taken for each delay distribution and bandwidth distribution, in the format of link weight vector (delay, -log(delay probability), -log(bandwidth probability)). It is worthwhile to mention that the bandwidth value does not exist in the weight vector, since we only need to allocate just enough requested bandwidth. As a result, each link will be transformed into $k$ parallel links as illustrated in Fig. 2.

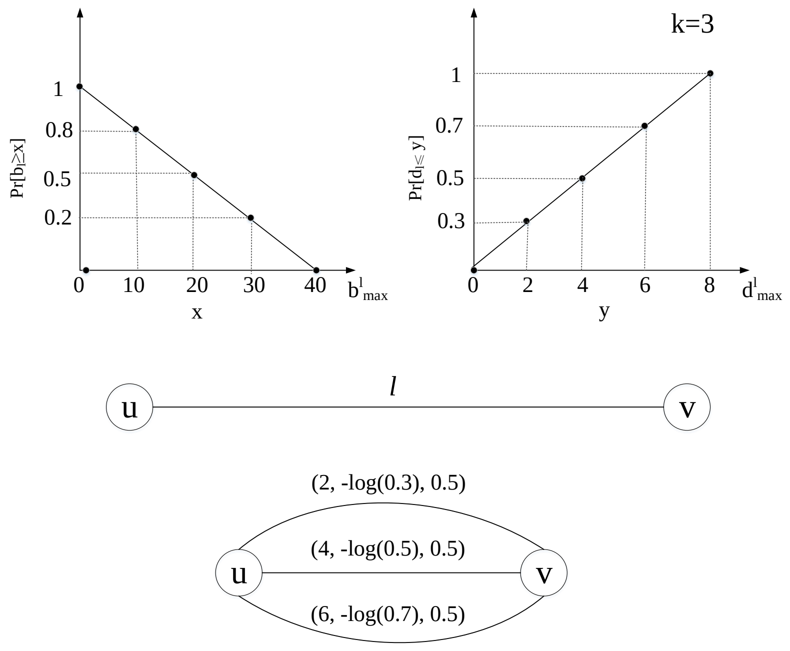
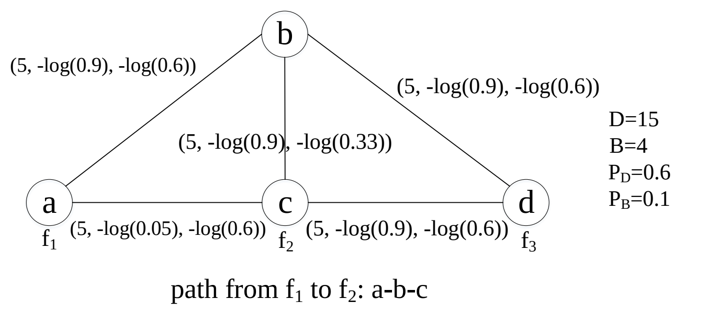


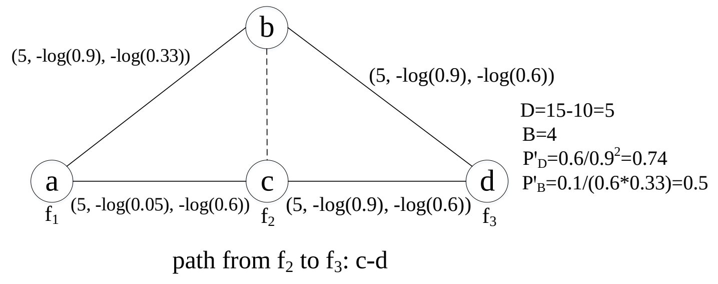
Figure 2: Link transformation for the heuristic algorithm.

Subsequently, for each VNF pair $f_i$ and $f_j$, we run the Tunable Accuracy Multiple Constraints Routing Algorithm (TAMCRA) [46], a heuristic for the multi-constrained path problem. The objective in TAMCRA is to minimize $\sum_{l \in p} d_l$, where $d_l$ indicates the delay of traversed link $l$ in path $p$. The delay probability constraint in TAMCRA is $-\frac{\log(P_D)}{\gamma}$ and the bandwidth probability constraint in TAMCRA is $-\frac{\log(P_B)}{\gamma}$. $\gamma$ is used to “scale” the original requirement to each requested VNF pair and is first set to be the number of requested VNF pairs for simplicity. In case under $-\frac{\log(P_D)}{\gamma}$ the path is not found by applying TAMCRA, we can enlarge $\gamma$ and run TAMCRA again. After that, the delay probability constraint is set $-\frac{\log(P_D/P_x)}{\gamma'}$, where $P_x$ stands for the product of already consumed probability for each VNF pair, and $\gamma'$ represents the number of VNF pairs that have not been iterated. TAMCRA never returns a path that violates QoS constraint, but it may fail to find a solution since it is a heuristic. TAMCRA is tunable in how many paths it stores per node, a parameter that we set equal to $q = ck$, a constant $c$ times the number of samples $k$.

An example is given in Fig. 3. It is assumed that the bandwidth CDF for link $(b, c)$ is $\frac{x}{6}$, and the bandwidth CDF for all the other links is $\frac{x}{10}$. Moreover,


b
(5, -log(0.9), -log(0.6))
(5, -log(0.9), -log(0.6))
(5, -log(0.9), -log(0.33))
D=15
B=4
$P_D$=0.6
$P_B$=0.1
a
c
d
$f_1$
(5, -log(0.05), -log(0.6))
$f_2$
(5, -log(0.9), -log(0.6))
$f_3$
path from $f_1$ to $f_2$: a-b-c

(a) Path finding from $f_1$ to $f_2$


b
(5, -log(0.9), -log(0.33))
(5, -log(0.9), -log(0.6))
D=15-10=5
B=4
$P'_D=0.6/0.9^2=0.74$
$P'_B$=0.1/(0.6*0.33)=0.5
a
c
d
$f_1$
(5, -log(0.05), -log(0.6))
$f_2$
(5, -log(0.9), -log(0.6))
$f_3$
path from $f_2$ to $f_3$: c-d

(b) Path finding from $f_2$ to $f_3$

Figure 3: An example of the heuristic link changes for $k = 1$.

except for link $(a, c)$, the delay distribution are the same for all the other links. Assuming there is a request $r(\{f_1, f_2, f_3\}, 4, 0.5, 15, 0.1)$. The link weight assignment are shown in Fig. 3(a). Since $r$ asks for allocating at least a bandwidth of $B = 4$ and link $(b, c)$ has a bandwidth CDF $\frac{x}{6}$, then allocating at least a bandwidth of 4 leads to a probability of $1 - \frac{4}{6} \approx 0.33$. Similarly, the probability of allocating at least a bandwidth of 4 for the other links is equal to $1 - \frac{4}{10} = 0.6$. Since $f_1$ and $f_2$ are located on $a$ and $c$, respectively, we take the path $a - b - c$ to route traffic from $a$ $(f_1)$ to $c$ $(f_2)$. The path $a - c$ cannot be selected, otherwise $P_D$ will be violated. Afterwards, the link weights of $(a, b)$ and $(b, c)$ will be adjusted. More specifically, since 4 bits of bandwidth have already been allocated, the probability of allocating another 4 bits bandwidth on link $(a, b)$ is $\frac{c_{(a,b)}(8)}{c_{(a,b)}(4)} = \frac{1-8/10}{1-4/10} \approx 0.33$. Moreover, since the maximum possible bandwidth of link $(b, c)$ is 6 and it has already allocated 4 bits of bandwidth, link $(b, c)$ cannot allocate another 4 bits bandwidth. We therefore used dashed line to represent it which means it cannot be used. Afterwards in Fig. 3(b), we continue to find a path from $f_2$ to $f_3$. It can be seen that path $c - d$ is the only feasible path. In all, the final path is $a - b - c - d$ has a total delay of 15 by summing up the link delay of $(a, b)$, $(b, c)$ and $(c, d)$ and its realizing probability is equal to

**Algorithm 1:** MCTR $(\mathcal{G}(\mathcal{N},\mathcal{L}), R, k)$

```
1  foreach request $r \in R$ do
2      foreach possible corresponding SFC $\Omega$ do
3          foreach $f_i, f_j \in \Omega$ which are located on $n_i$ and $n_j$ do
4              if $n_i$ and $n_j$ have sufficient capability then
5                  Discrete the network link weights by $k$ samplings.
6                  Apply TAMCRA to find a path from $n_i$ to $n_j$.
7                  if step 6 succeeds then
8                      Adjust link weights and requirement.
9                      if $f_j$ is the last required VNF then
10                         Return this solution.
11                 else
12                     break;
```

$0.9 \cdot 0.9 \cdot 0.9 = 0.729$ by taking the product of probabilities to realize these delay. Similarly, each link in path is $a-b-c-d$ can allocate available bandwidth of 4 and the total realizing probability is equal to $0.6 \cdot 0.33 \cdot 0.6 = 0.1188$. The pseudo code of MCTR can be seen in Algorithm 1.

The complexity of MCTR is analyzed like this. There are in total $|R|$ requests, and for each request, there are constant number of possible corresponding SFCs and at most $|F|-1$ required VNF pairs. For each VNF pair, running TAMCRA calls for a time complexity of $O(qN\log(qN) + q^2kL)$, where $q$ is the maximum number of stored paths at each node. Consequently, the total time complexity of MCTR is $O(|R||F|qN\log(qN) + |R||F|q^2kL)$.

## 6. Simulations

### *6.1. Simulation Setup*

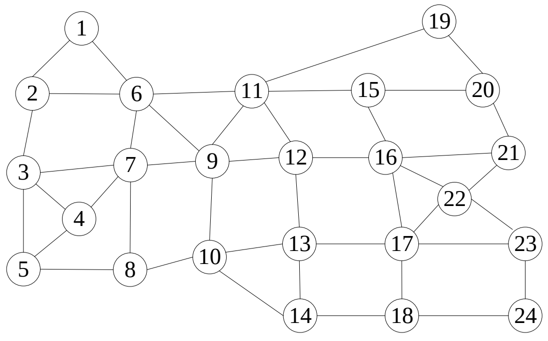


Figure 4: USA carrier backbone network.

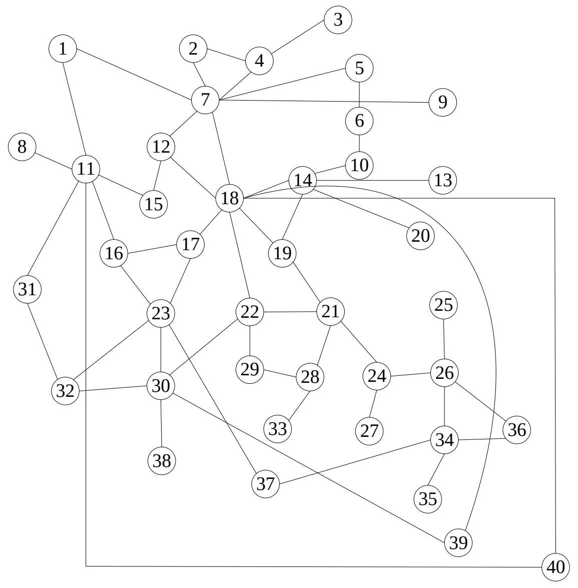


Figure 5: GÉANT pan-European research network.

In this section, we first conduct simulations on two backbone networks: USANet, displayed in Fig. 4, which is a realistic carrier backbone network, and GÉANT, shown in Fig. 5, which is a pan-European communications infrastructure. We also conduct simulations on a 100-node network: we generate 100 nodes and the link existence in each node pair occurs with a probability of $\frac{1}{2}$, which follows an Erdős-Rényi model [49]. We assume there are in total 15 VNFs, and each VNF is randomly placed on $m$ nodes, where $m \in [2, 4]$. The link delay and bandwidth in these 3 networks are assumed to follow three distributions, namely the exponential, uniform and weibull distributions. In the exponential distribution $1 - e^{-\lambda x}$, we choose $\lambda \in [0.001, 0.01]$ for the bandwidth distributions of different links and $\lambda \in [0.5, 1.5]$ for the delay distributions of different links. In the uniform distribution $\frac{x-\alpha}{\beta-\alpha}$, we choose $\alpha = 0$ for both bandwidth and delay, and $\beta \in [200, 300]$ for bandwidth and $\beta \in [3, 7]$ for delay. In the weibull distribution $1 - e^{-(x/\lambda)^k}$, we choose $k = 1$ for both bandwidth and delay, and $\lambda \in [1, 5]$ for delay and $\lambda \in [0.0001, 0.0005]$ for bandwidth.

Considering that there is no existing work dealing with the traffic routing problem in stochastic networks, we compare the exact solution and heuristic MCTR with two benchmark heuristics, namely Least Expected Path (LEP) and Random algorithm. For each request and its possible corresponding SFC, for each VNF pair that are located on different nodes $n_i$ and $n_j$, these two heuristics perform as follows:

- LEP: It first assigns each link with the expected delay value, and then runs the shortest path from $n_i$ to $n_j$.

- Random algorithm: It finds $w$-shortest hop path from $n_i$ to $n_j$, and randomly selects one path from these $k$ paths.

After determining the path for each VNF pair, we let $L_x$ represent the total number of links (hops) that the required SFC traverses. Subsequently, for both LEP and Random algorithm, each traversed link $l$ will allocate $\frac{r.D}{L_x}$ delay and $r.B{\cdot}t_l$ bandwidth, where $t_l$ denotes the times that the link is traversed[3]. For each traversed link $l$, the probability of allocating delay is calculated as $CDF_l(\frac{r.D}{L_x})$, and the probability of realizing bandwidth calculated as $CCDF_l(r.B \cdot t_l)$. By taking the product of all allocated probabilities, we can calculate whether the request can be satisfied or not. Random algorithm will try at most $p$ times from these $w$ paths if the cumulated constraint (e.g., delay, bandwidth, realizing probability) is not obeyed. In particular, we set $w = 10$ and $p = 3$ in Random algorithm. We first set in MCTR that $k = 4, 8, 16$ and $c = 1$, which means that the maximum number of stored paths is set $q = 4, 8, 16$, respectively, when compared to LEP and Random algorithm. Later for a fixed $k = 8$, we will vary $q$ in order to further evaluate the heuristic algorithm.

In order that the feasible solution exists and increase the difficulty for the algorithms to find solutions, the request $r(F, B, P_B, D, P_D)$ is set like this (in line with [41]): For exponential distribution, we have set $D \in [5, 15]$, $P_D \in [0.1, 0.4]$, $B \in [1, 10]$ and $P_B \in [0.1, 0.3]$; for uniform distribution, we have set $D \in [15, 30]$, $P_D \in [0.1, 0.2]$, $B \in [1, 10]$ and $P_B \in [0.05, 0.1]$; for weibull distribution, we have set $D \in [15, 30]$, $P_D \in [0.01, 0.02]$, $B \in [1, 10]$ and $P_B \in [0.005, 0.01]$. Moreover, $F$ is between 2 and 5 for all 3 distributions. Table 2 provides the simulation parameters.

The simulations are run on a high-performance desktop PC with 3.4 GHz and 16 GB memory. We use C# to implement all the heuristics, and we use CVX in Matlab to implement the exact solution, a package for specifying and solving convex optimization problems [50].

Table 2: Simulation Parameters.

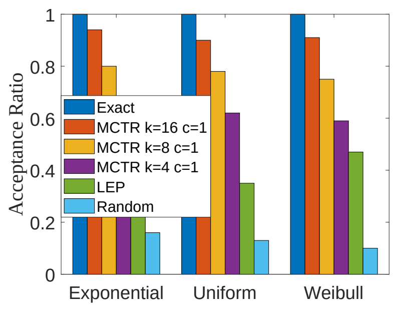
| Distribution | $r(F, B, P_B, D, P_D)$ | Link Bandwidth | Link Delay |
|---|---|---|---|
| $1 - e^{-\lambda x}$ | $B \in [1, 10], P_B \in [0.1, 0.3]$<br>$D \in [5, 15], P_D \in [0.1, 0.4]$ | $\lambda \in [0.001, 0.01]$ | $\lambda \in [0.5, 1.5]$ |
| $\frac{x-\alpha}{\beta-\alpha}$ | $B \in [1, 10], P_B \in [0.05, 0.1]$<br>$D \in [15, 30], P_D \in [0.1, 0.2]$ | $\alpha = 0$<br>$\beta \in [200, 300]$ | $\alpha = 0$<br>$\beta \in [3, 7]$ |
| $1 - e^{-(x/\lambda)^k}$ | $B \in [1, 10], P_B \in [0.005, 0.01]$<br>$D \in [15, 30], P_D \in [0.01, 0.02]$ | $k = 1$<br>$\lambda \in [0.0001, 0.0005]$ | $k = 1$<br>$\lambda \in [1, 5]$ |

[3]It is possible that one link is traversed more than once in an SFC.

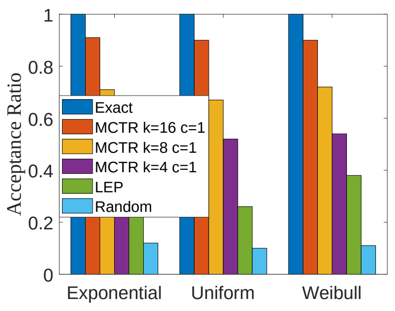
## *6.2. Simulation results*

### *6.2.1. Backbone networks*

We first generate in total 100 requests for each distribution, and it is assumed that the request arrives in network one by one in an online fashion. Accordingly, the algorithm runs 100 times sequentially for each request. We first compare the algorithms in terms of Acceptance Ratio (AR), which is defined as the number of accepted requests divided by the total number of requests. As expected, Fig. 6 shows that the exact solution can always find paths for each request (if it exists) with AR=1, which also verifies its exactness and correctness. With $k$ (and $q$) increasing, we found that the achieved AR value of MCTR approaches to the close-to-optimal (above 90% when $k = 16$). The reason is that more samples are created in networks, and therefore there are more parallel links with various delay and bandwidth allocation weights between each node pair, which makes MCTR to have more choices to find feasible solutions. Since MCTR leverages TAMCRA, which is a heuristic to find the multi-constrained path, MCTR may not always find a feasible solution. However, with the maximum number of stored paths increasing, it increases the chance for MCTR to return the feasible solution. On the other hand, both the LEP and Random algorithm behave poorly by having a much lower AR value. For LEP, it indicates that the least expected path may not always satisfy the request with specified delay and bandwidth probability requirement. For the Random algorithm, although it adopts $w = 10$ paths for each VNF pair, the quality of paths is not defined very well according to the request, and its randomness further leads to the worst performance especially under a stricter problem condition.



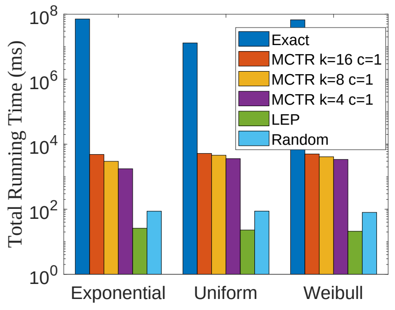
(a) USANet



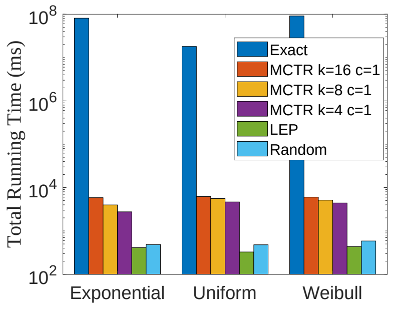
(b) GÉANT

Figure 6: Acceptance Ratio on two backbone networks: (a) USANet and (b) GÉANT.

Fig. 7 plots the total running time of all the algorithms. We see that the exact solution consumes a significantly higher running time than 3 heuristics. Although it can achieve the best performance in terms of AR, it still cannot be adopted to compute the solution for when the request arrives in an online fashion. The proposed MCTR has much lower running time than the exact



(a) USANet



(b) GÉANT

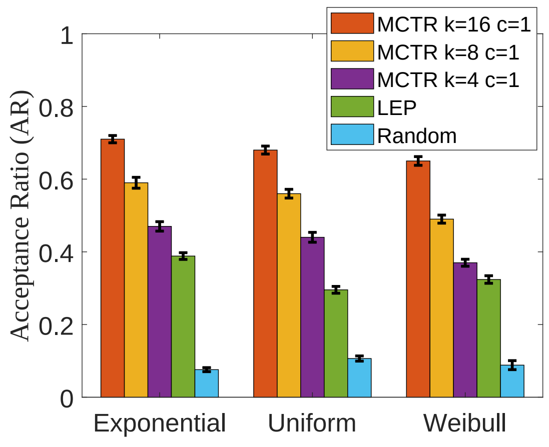
Figure 7: Running Time on two backbone networks: (a) USANet and (b) GÉANT.

solution, but its running time is higher than the two benchmark heuristics. However, it is still acceptable since it can achieve close-to-optimal AR value. We observe that with $k$ increasing, its running time also increases. This is caused by the larger network scale and larger maximum number of stored paths. Nevertheless, we can tune the parameters of MCTR in practice to strike a tradeoff between accuracy and running time.

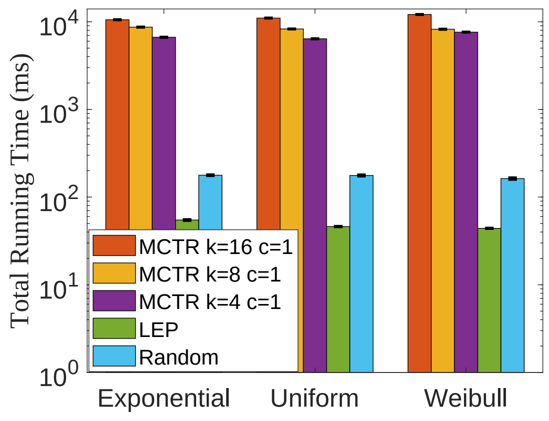
### *6.2.2. 100-node network*

In this scenario, we run all the algorithms on a larger 100-node network, but we found that the exact algorithm cannot return a solution because of its exponential time complexity. This also indicates that the exact solution cannot scale well with the problem size and cannot be adopted in practice. Due to the lack of the exact solution, we generate 100 sets of 100 traffic requests for distributions, and evaluate all the heuristic algorithms for those 100 sets of 100 traffic requests (100 runs). By doing this, we want to establish confidence on the performance of heuristics. Figs. 8 (a) and 8(b) respectively depict the AR and running time (in log scale) of all these algorithms, where the confidence interval is set to 95%. The 95% confidence interval is calculated for all the figures, but in those where it is not visible, the interval is negligibly small[4]. More specifically, in Fig. 8 (a), the confidence intervals of AR values for all the algorithms stay around 0.01. In Fig. 8 (b), the confidence intervals of running time for MCTR and LEP ranges from 0.3 to 0.6, and the confidence intervals of running time for Random ranges from 1.9 to 3.3, which is mainly caused by its randomness. We see that the AR values of all the algorithms in 100-node network are lower than the AR values in backbone networks, due to the reason that SFC may traverse more links which consumes more delay and hence violate the required delay

[4]We note here that some plots are log-scale that additionally contributes to the confidence interval visibility.



(a) Acceptance Ratio



(b) Total Running Time

Figure 8: Performance on a 100-node network: (a) Acceptance Ratio and (b) Running Time.

value. Also, the enlarged network scale makes all the algorithms to consume more time than backbone networks. Nevertheless, the proposed MCTR achieves much higher AR value than LEP and Random, but this comes at the expense of a bit higher running time.

*6.2.3. Varying the Maximum Number of Stored Paths for MCTR*

Finally, in MCTR we keep $k = 8$ the same and vary the value of $c = 1, 2, 3, 4$. Our aim is to evaluate the performance of MCTR when the maximum number of stored paths $q = ck$ changes. In particular, we only show the average value

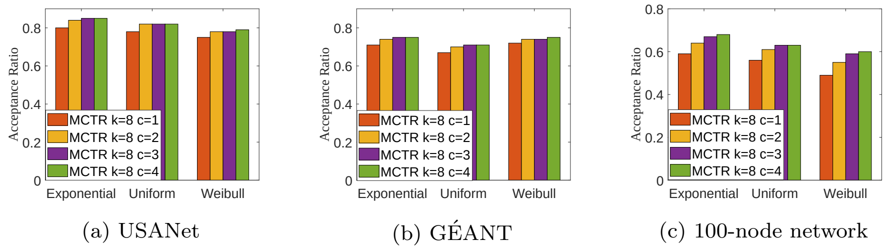


Figure 9: Acceptance Ratio of MCTR on three networks for $k = 8$ and $q = ck$, where $c = 1, 2, 3, 4$: (a) USANet (b) GÉANT (c) 100-node network.

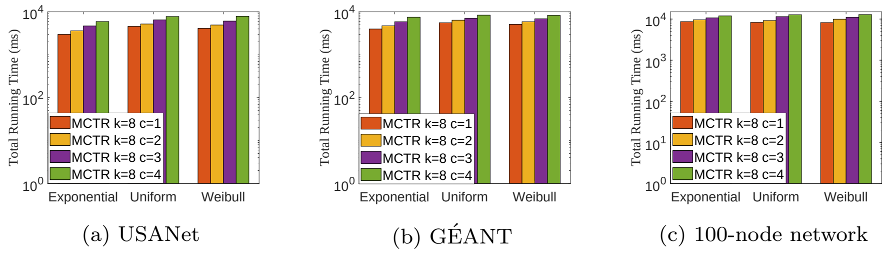


Figure 10: Running Time of MCTR on three networks for $k = 8$ and $q = ck$, where $c = 1, 2, 3, 4$: (a) USANet (b) GÉANT (c)100-node network.

of all the algorithms in the 100-node network since the confidence intervals in this case are very small. As expected, Fig. 9 shows that with $q$ increasing, its achieved AR value increases, which indicates that a bigger $q$ may lead to better performance. However, the running time of MCTR increases when $q$ increasing, as we see in Fig. 10.

In all, we conclude that the exact solution can always find the optimal solution (if there exists), but this comes at the expense of significantly higher running time. When the problem size becomes larger (e.g, $|N| = 100$), the exact solution cannot return a feasible solution within quite a long time. Our proposed MCTR, can return a feasible solution at a much quicker time regardless of the problem size, which can be used when the traffic requests arrive in an online fashion.

## 7. Conclusion

In this paper, we have studied the Traffic Routing problem in Stochastic NFV Networks. The randomness/uncertainties in networks usually arise from inaccurate data, expired exchanged information, insufficient estimation to the network, etc. Under the assumption that the link delay and link bandwidth are random variables and their CDFs are known, we have shown that the problem is NP-hard. To solve it, we present an exact optimization formulation as well as a tunable sampling-based heuristic MCTR. MCTR leverages the Tunable Accuracy Multiple Constraints Routing Algorithm (TAMCRA) to find the multi-constraint path for each adjacent VNF pair. Moreover, it dynamically adjusts the link weights and delay and bandwidth realizing probability constraint after finding the path for each VNF pair in order that the cumulated probabilities will not exceed the specified. The simulation results demonstrate that the proposed heuristic can achieve close-to-optimal performance in terms of acceptance ratio, but its running time is much lower than the exact solution. Moreover, MCTR can scale well when the network size is enlarged to 100-node, but the exact solution cannot return a feasible in this scenario within a reasonably long time, which indicates that the exact solution cannot be adopted for when the provisioning time needs to be short. We found that with more samplings are allowed in MCTR, its achieved AR value gets higher which means it can accept more requests. Meanwhile, when the number of stored paths is increased in MCTR, its performance also gets better. This has verified that MCTR is a tunable and efficient heuristic that can achieve a tradeoff between accuracy and running time. In our future work, we would like to additionally consider the VNF placement problem in stochastic NFV networks and implement the proposed algorithms in an NFV-based simulator.

**Acknowledgments**

The work of Song Yang is partially supported by the National Natural Science Foundation of China (NSFC, No. 61802018) and Beijing Institute of Technology Research Fund Program for Young Scholars. The work of Fan Li is partially supported by the NSFC (No. 61772077, 61432015), and the Beijing Natural Science Foundation (No. 4192051). The work of Xiaoming Fu is partially supported by EU FP7 CleanSky ITN (No. 607584) and H2020 RISE COSAFE (No. 824019) projects.

## References

[1] Cisco visual networking index: Forecast and methodology, 2016–2021 (2018).
URL `https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html`

[2] ETSI Publishes First Specifications for Network Functions Virtualisation, `http://www.etsi.org/news-events/news/700-2013-10-etsi-publishes-first-nfv-specifications`.

[3] A. M. Medhat, T. Taleb, A. Elmangoush, G. A. Carella, S. Covaci, T. Magedanz, Service function chaining in next generation networks: State of the art and research challenges, IEEE Communications Magazine 55 (2) (2017) 216–223.

[4] J. G. Herrera, J. F. Botero, Resource allocation in NFV: A comprehensive survey, IEEE Transactions on Network and Service Management 13 (3) (2016) 518–532.

[5] B. Yi, X. Wang, K. Li, M. Huang, et al., A comprehensive survey of network function virtualization, Computer Networks 133 (2018) 212–262.

[6] H. Hantouti, N. Benamar, T. Taleb, A. Laghrissi, Traffic steering for service function chaining, IEEE Communications Surveys & Tutorials 21 (1) (2018) 487–507.

[7] M. Krishnan, S. Yun, Y. M. Jung, Improved clustering with firefly-optimization-based mobile data collector for wireless sensor networks, AEU - International Journal of Electronics and Communications 97 (2018) 242 – 251.

[8] S. Yang, F. A. Kuipers, Traffic uncertainty models in network planning, IEEE Communications Magazine 52 (2) (2014) 172–177.

[9] R. A. Guerin, A. Orda, Qos routing in networks with inaccurate information: theory and algorithms, IEEE/ACM Transactions on Networking 7 (3) (1999) 350–364.

[10] T. Korkmaz, M. Krunz, Bandwidth-delay constrained path selection under inaccurate state information, IEEE/ACM Transactions on Networking 11 (3) (2003) 384–398.

[11] B. A. A. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, T. Turletti, A survey of software-defined networking: Past, present, and future of programmable networks, IEEE Communications Surveys & Tutorials 16 (3) (2014) 1617–1634.

[12] M. Reitblatt, N. Foster, J. Rexford, C. Schlesinger, D. Walker, Abstractions for network update, SIGCOMM Computer Communication Review 42 (4) (2012) 323–334.

[13] M. Krishnan, S. Yun, Y. M. Jung, Enhanced clustering and aco-based multiple mobile sinks for efficiency improvement of wireless sensor networks, Computer Networks 160 (2019) 33 – 40.

[14] D. Bhamare, R. Jain, M. Samaka, A. Erbad, A survey on service function chaining, Journal of Network and Computer Applications 75 (2016) 138–155.

[15] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, R. Boutaba, Network function virtualization: State-of-the-art and research challenges, IEEE Communications Surveys & Tutorials 18 (1) (2016) 236–262.

[16] G. MIRJALILY, Optimal network function virtualization and service function chaining: A survey, Chinese Journal of Electronics 27 (2018) 704–717.

[17] A. Laghrissi, T. Taleb, A survey on the placement of virtual resources and virtual network functions, IEEE Communications Surveys Tutorials 21 (2) (2019) 1409–1434.

[18] M. R. Garey, D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman & Co., New York, 1979.

[19] A. Dwaraki, T. Wolf, Adaptive service-chain routing for virtual network functions in software-defined networks, in: Proceedings of the 2016 Workshop on Hot Topics in Middleboxes and Network Function Virtualization, ACM HotMIddlebox, 2016, pp. 32–37.

[20] T. Wang, H. Xu, F. Liu, Multi-resource load balancing for virtual network functions, in: IEEE 37th International Conference on Distributed Computing Systems (ICDCS), 2017, pp. 1322–1332.

[21] R. Yu, G. Xue, X. Zhang, QoS-aware and reliable traffic steering for service function chaining in mobile networks, IEEE Journal on Selected Areas in Communications 35 (11) (2017) 2522–2531.

[22] L. Gao, G. N. Rouskas, On congestion minimization for service chain routing problems, in: IEEE International Conference on Communications (ICC), 2019, pp. 1–6.

[23] W. Ma, O. Sandoval, J. Beltran, D. Pan, N. Pissinou, Traffic aware placement of interdependent NFV middleboxes, in: IEEE INFOCOM, 2017.

[24] C. Pham, N. H. Tran, S. Ren, W. Saad, C. S. Hong, Traffic-aware and energy-efficient vNF placement for service chaining: Joint sampling and matching approach, IEEE Transactions on Services Computing (early access) (2017) 1–14.

[25] T.-W. Kuo, B.-H. Liou, K. C.-J. Lin, M.-J. Tsai, Deploying chains of virtual network functions: On the relation between link and server usage, in: IEEE INFOCOM, 2016, pp. 1–9.

[26] A. Tomassillik, F. Giroire, N. Huin, S. Pérennes, Provably efficient algorithms for placement of service function chains with ordering constraints, in: IEEE INFOCOM, 2018.

[27] A. Marotta, E. Zola, F. D'Andreagiovanni, A. Kassler, A fast robust optimization-based heuristic for the deployment of green virtual network functions, Journal of Network and Computer Applications 95 (2017) 42–53.

[28] S. Song, C. Lee, H. Cho, G. Lim, J. Chung, Clustered virtualized network functions resource allocation based on context-aware grouping in 5g edge networks, IEEE Transactions on Mobile Computing.

[29] L. Guo, J. Pang, A. Walid, Joint placement and routing of network function chains in data centers, in: IEEE INFOCOM, 2018.

[30] L. Qu, C. Assi, K. Shaban, Delay-aware scheduling and resource optimization with network function virtualization, IEEE Transactions on Communications 64 (9) (2016) 3746–3758.

[31] Q. Li, Y. Jiang, P. Duan, M. Xu, X. Xiao, Quokka: Latency-aware middlebox scheduling with dynamic resource allocation, Journal of Network and Computer Applications 78 (2017) 253–266.

[32] M. Hamann, M. Fischer, Path-based optimization of nfv-resource allocation in sdn networks, in: IEEE International Conference on Communications (ICC), 2019, pp. 1–6.

[33] S. Yang, F. Li, S. Trajanovski, X. Chen, Y. Wang, X. Fu, Delay-aware virtual network function placement and routing in edge clouds, IEEE Transactions on Mobile Computing (early access) (2019) 1–14.

[34] J. Fan, M. Jiang, O. Rottenstreich, Y. Zhao, T. Guan, R. Ramesh, S. Das, C. Qiao, A framework for provisioning availability of nfv in data center networks, IEEE Journal on Selected Areas in Communications 36 (10) (2018) 2246–2259.

[35] S. Yang, F. Li, R. Yahyapour, X. Fu, Delay-sensitive and availability-aware virtual network function scheduling for nfv, IEEE Transactions on Services Computing (early access) (2019) 1–14.

[36] H. Tang, D. Zhou, D. Chen, Dynamic network function instance scaling based on traffic forecasting and vnf placement in operator data centers, IEEE Transactions on Parallel and Distributed Systems 30 (3) (2019) 530–543.

[37] B. Blanco, J. O. Fajardo, I. Giannoulakis, E. Kafetzakis, S. Peng, J. Pérez-Romero, I. Trajkovska, P. S. Khodashenas, L. Goratti, M. Paolino, et al., Technology pillars in the architecture of future 5G mobile networks: NFV, MEC and SDN, Computer Standards & Interfaces 54 (2017) 216–228.

[38] X. Fu, F. R. Yu, J. Wang, Q. Qi, J. Liao, Dynamic service function chain embedding for NFV-enabled IoT: A deep reinforcement learning approach, IEEE Transactions on Wireless Communications (early access) (2019) 1–14.

[39] A. Garcia-Saavedra, X. Costa-Perez, D. J. Leith, G. Iosifidis, FluidRAN: Optimized vRAN/MEC orchestration, in: IEEE INFOCOM, 2018.

[40] D. H. Lorenz, A. Orda, Qos routing in networks with uncertain parameters, IEEE/ACM Transactions on Networking 6 (6) (1998) 768–778.

[41] F. A. Kuipers, S. Yang, S. Trajanovski, A. Orda, Constrained maxmum flow in stochastic networks, in: Proc. of IEEE ICNP 2014, North Carolina, October 2014.

[42] X. Cheng, Y. Wu, G. Min, A. Y. Zomaya, Network function virtualization in dynamic networks: A stochastic perspective, IEEE Journal on Selected Areas in Communications 36 (10) (2018) 2218–2232.

[43] D. Zeng, J. Zhang, L. Gu, S. Guo, Stochastic scheduling towards cost efficient network function virtualization in edge cloud, in: IEEE International Conference on Sensing, Communication, and Networking (SECON), 2018, pp. 1–9.

[44] W. Miao, G. Min, Y. Wu, H. Huang, Z. Zhao, H. Wang, C. Luo, Stochastic performance analysis of network function virtualization in future internet, IEEE Journal on Selected Areas in Communications 37 (3) (2019) 613–626.

[45] G. Hardy, J. Littlewood, G. Polya, Inequalities, Cambridge University Press, 1934.

[46] H. De Neve, P. Van Mieghem, Tamcra: a tunable accuracy multiple constraints routing algorithm, Computer Communications 23 (7) (2000) 667–679.

[47] F. A. Kuipers, P. F. Van Mieghem, Conditions that impact the complexity of QoS routing, IEEE/ACM transactions on networking 13 (4) (2005) 717–730.

[48] Z. Wang, J. Crowcroft, Quality-of-service routing for supporting multimedia applications, IEEE Journal on selected areas in communications 14 (7) (1996) 1228–1234.

[49] P. Erdős, A. Rényi, On random graphs i. publicationes mathematicae (debrecen) (1959) 290–297.

[50] M. Grant, S. Boyd, CVX: Matlab software for disciplined convex programming, version 2.1, `http://cvxr.com/cvx` (Mar. 2014).