Quantifying the Effect of Data Sovereignty in Service Function Chain Embedding

Trond Vatten*, Jacek Rak†, Yuming Jiang*, Poul E. Heegaard*
*Department of Information Security and Communication Technology, NTNU
†Department of Computer Communications, Gdańsk University of Technology
E-mail: {trond.vatten,yuming.jiang,poul.heegaard}@ntnu.no, jrak@pg.edu.pl

***Abstract*—Network function virtualization (NFV) enables operators to flexibly place virtual network functions (VNFs) wherever processing capacity is available. Emerging *data sovereignty* requirements, however, restrict where traffic may be processed and which network domains it may traverse, turning placement and routing into demand-specific feasibility problems. In this paper, we quantify the operational "cost of sovereignty" in service function chain (SFC) embedding, measured primarily through the fraction of admitted demands (admission ratio) and associated resource-efficiency trade-offs. We develop and compare two sovereignty-aware mixed-integer linear programming (MILP) formulations: (i) a link-flow MILP baseline that jointly optimizes VNF placement and routing and can be solved to global optimality for moderate instances; and (ii) a scalable configuration-selection approach that precomputes a finite set of end-to-end candidate embeddings per demand and then solves a smaller MILP over these candidates. Using topologies of real backbone topologies under mixed sovereign and non-sovereign traffic, we observe a pronounced "sovereignty penalty": enforcing sovereignty constraints can reduce the admission ratio by up to 30 % in certain scenarios.**

***Index Terms*—SFC embedding, data sovereignty, NFV orchestration, MILP, path-based formulation, admission control**

I. Introduction

Modern networks are increasingly virtualized, with virtual network functions (VNFs) deployed as software on commodity servers in data centers. Packet-processing tasks traditionally implemented by dedicated middleboxes are no longer bound to specialized hardware, but can instead be realized as VNFs through network function virtualization (NFV) [1]. For example, in 5G networks, the 5G Core adopts a service-based architecture in which the control-plane and user-plane functions are deployed as VNFs that can be flexibly instantiated in centralized or edge clouds near the radio access network. End-to-end services are then provided as ordered chains of VNFs across the infrastructure, called Service Function Chains (SFCs) [2]. The virtualization paradigm enables operators to allocate resources dynamically and route traffic by scaling VNF instances and adapting SFC routing in response to changing traffic load. Efficiently placing and scaling VNFs while meeting end-to-end service requirements is a key challenge.

A central optimization problem in such infrastructures is SFC embedding (also referred to as the joint VNF placement and routing problem): deciding on (i) VNF placement (how many instances to instantiate and where) and (ii) SFC routing (how to route flows through the required VNFs to provide a service). These decisions directly impact end-to-end latency, bandwidth consumption, and resource efficiency, thereby determining how many SFC demands can be admitted (i.e., the demand admission ratio). On the one hand, multiple flows sharing VNF instances can improve utilization and reduce instantiation overhead; on the other hand, placing VNFs along each flow's shortest path reduces latency but may require additional instances. These objectives conflict, and the joint placement-and-routing problem with capacity and latency constraints is NP-hard [3]. A large body of work proposes mixed-integer linear programming (MILP) formulations and scalable heuristics for SFC embedding in networks [4], [5].

In parallel, regulations and policies are increasingly imposing constraints on *where* data and services may run. For operators, these requirements translate into hard constraints on where traffic may be *processed* (which sites may serve as hosts (i.e., stages) for particular VNFs) and constrain where traffic may *transit* (which sites, links, or administrative domains flows may traverse) [6], [7], [8]. This creates a distinct network challenge: the constraints couple *compute placement* with *traffic routing* across geographically distributed backbones and edge clouds, and can impact classic constraints (capacity, latency) and operator objectives (admission, efficiency). As a result, common virtualization narratives, such as *"spin up the VNF where capacity is available"*, no longer hold as service placement and routing must respect jurisdictional boundaries. These new constraints should be treated as hard constraints in SFC-embedding and traffic-engineering models.

In this paper, we use **data sovereignty** as an umbrella term for enforceable constraints on where traffic may be processed and transit. Research literature and policy discussions use related but distinct terms [6], [7], [8]: *Data residency* concerns the *physical location* where data is stored/processed (e.g., "within country X"). *Data localization* is a *policy requirement* that requires certain data and processing to remain within a territory, often restricting cross-border transfers. *Data sovereignty* focuses on *governance and legal control*: who can access data, under which jurisdiction(s), and under which administrative authority it is processed and operated.

Prior works provide strong foundations for joint placement and routing, scalable formulations, and high-level SFC embedding constraints. However, to the best of our knowledge, they have not made *data sovereignty* the primary modeling object.

ISBN 978-3-903176-82-9

We build on these foundations by treating data sovereignty as a specific class of processing and transit restrictions. This paper presents three contributions. First, we model data sovereignty as demand-specific feasibility constraints on processing locations and traffic transit, enabling granular control over where each SFC's VNFs execute. Second, we develop two MILP formulations for sovereignty-aware SFC embedding: an MILP baseline (solved to provable global optimality) and a *scalable* configuration–selection MILP that reduces the problem size via precomputed candidate embeddings. Third, in topologies of real backbone networks, we implement zone-based sovereignty policies, quantify the "cost of sovereignty", and report resource-efficiency trade-offs when sovereignty-constrained and unconstrained demands coexist.

The remainder of the paper is organized as follows: Section II reviews related work; Section III presents the network and sovereignty model. Section IV introduces the two MILP formulations. Section V describes the experimental methodology. Sections VI and VII report and discuss the results, while Section VIII concludes the paper.

II. Related Work

The increasing adoption of NFV has for years motivated how to embed SFCs, widely formulated as a joint optimization problem of how to instantiate VNFs and how to route traffic through them under capacity, latency, and operational constraints [3]. In parallel, regulatory and organisational requirements governing data traffic are emerging, as highlighted in ETSI's NFV use-case report [6]. In an increasingly fragmented geopolitical landscape, these requirements are becoming more stringent and prevalent. This shift towards data sovereignty, in turn, imposes additional strict constraints on the location of processing, storage, handling, and transit of data [8].

Early work, such as [3], captures the coupling between VNF placement decisions and network-level constraints and has served as a common baseline for later work. Practical scalability challenges of compact formulations are discussed in [9], including trade-offs and evaluations on realistic topologies. Beyond MILPs, the literature proposes algorithmic approaches with efficiency guarantees under structural assumptions [4].

Exact SFC embedding formulations rapidly become infeasible as the number of demands, chain lengths, and physical network sizes increase. To address this, [5] develops decomposition-based approaches using layered graphs and column generation. Complementarily, [10] proposes a *path-based* MILP that relies on candidate path enumeration, showing that restricting solutions to a carefully chosen candidate set can yield high-quality results when full enumeration is infeasible. This is extended in [11], which studies how alternative formulations can strengthen solution approaches for SFC embedding.

Several works extend SFC embedding beyond resource and latency constraints by incorporating higher-level placement constraints. In [12], [13], the authors introduce affinity and anti-affinity constraints for SFC requests and VNF placement, demonstrating how to encode higher-level placement policies in the optimization problem. Another structurally similar approach is [14], which derives feasibility constraints on nodes and paths from trust attributes and integrates them into a path-based MILP using k-shortest-path approximations.

Overall, existing work provides strong foundations for joint VNF placement and routing, scalable formulations, and constraints derived from higher-level policies such as affinity rules and trust. However, to the best of our knowledge, they do not consider data sovereignty as an explicit modeling dimension. At the same time, standardization efforts [6], policy discussions [8], and regulation [7] increasingly highlight constraints on where data may be processed and how it may transit across jurisdictions. These developments motivate treating data sovereignty as an explicit modeling dimension in SFC embedding. We therefore extend this line of work by modeling sovereignty as a first-class set of constraints on processing locations and traffic transit, and by studying its operational impact under realistic zone-based sovereignty policies on the topologies of real backbone networks.

III. Network and Sovereignty Model

A. Network Model

We model the infrastructure as a directed graph $G = (V, E)$, where V is the set of nodes and E the set of directed links. Nodes represent switches, servers, or data centers. Let $V^{\mathrm{c}} \subseteq V$ denote the subset of *VNF-hosting nodes* (i.e., nodes that can host VNFs). For each $v \in V^{\mathrm{c}}$, let C_v be its processing capacity measured in *CPU units*. For each link $e \in E$, let U_e be its bandwidth capacity (Mbps) and ℓ_e its propagation delay (ms).

B. Demand Model

Let $\mathcal{R}$ be the set of demands. Each demand $r \in \mathcal{R}$ is characterized by $r = (s_r, t_r, \mathbf{f}_r, b_r, L_r)$, where $s_r, t_r \in V$ are the source and destination nodes, b_r is the requested throughput (Mbps), L_r the end-to-end latency budget (ms), and $\mathbf{f}_r = (f_{r,1}, \ldots, f_{r,m_r})$ an ordered sequence of m_r VNF types specifying the required SFC stages. For each VNF type f, let α_f denote the processing requirement per unit throughput, measured in *CPU units per Mbps*. Thus, processing demand r at a VNF of type f consumes $\alpha_f\, b_r$ CPU units at the hosting node. In addition, let δ_f denote the fixed processing delay (ms) incurred when a VNF of type f processes traffic.

C. Sovereignty Model

We model data sovereignty as hard feasibility constraints on where traffic may be *processed* (VNF placement) and which links it may *transit* (routing). For each demand $r \in \mathcal{R}$ and VNF stage $k \in \{1, \ldots, m_r\}$, let $V^{\mathrm{proc}}_{r,k} \subseteq V^{\mathrm{c}}$ be the allowed subset of VNF-hosting nodes on which the VNF of type $f_{r,k}$ may be executed. Similarly, for each chain segment $k \in \{1, \ldots, m_r+1\}$, (from s_r to stage 1, between consecutive stages, and from the last stage to t_r), we define an allowed subset of transit links $E^{\mathrm{tr}}_{r,k} \subseteq E$. The embedding of demand r must place each VNF stage k on a node in $V^{\mathrm{proc}}_{r,k}$ and route each chain segment k using only links in $E^{\mathrm{tr}}_{r,k}$.

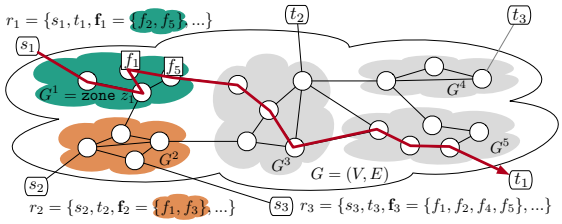


Fig. 1: Zone-based example of sovereignty-constrained demands where VNFs may only be executed within specific zones (colors).

Within a demand, these sets may vary across VNF stages and chain segments, enabling fine-grained policies. For example, a tenant may require that a deep packet inspection stage execute only in specific jurisdictions or facilities, while a network address translation stage is unconstrained; similarly, some segments may prohibit transit across certain administrative domains, while others are unrestricted.

In practice, sovereignty policies are often expressed through *zones* (e.g., states, operator domains, supranational regions). We treat zones as a general policy abstraction: each zone $z \in \mathcal{Z}$ is associated with a node subset $V(z) \subseteq V$ and a link subset $E(z) \subseteq E$. For processing constraints, the relevant hosting nodes in a zone are $V(z) \cap V^{\mathrm{c}}$. The mapping from zones to the allowed sets $V^{\mathrm{proc}}_{r,k}$ and $E^{\mathrm{tr}}_{r,k}$ is policy-dependent. The optimization models in Section IV treat these sets as inputs and enforce them directly. A concrete zone-based instantiation used in this paper is provided in Section V.

Figure 1 illustrates a zone-based policy example. A demand from s_1 to t_1 requires the ordered SFC of VNFs (f_2, f_5), both required to be deployed in zone z_1. The same holds for demand s_2 to t_2 in zone z_2, while demand s_3–t_3 has no sovereignty requirements and can be deployed at any VNF-hosting node. In this example, there are no constraints on transit, only on network-function processing (VNF execution).

IV. Problem Formulations

This section presents two MILP formulations for sovereignty-aware SFC embedding. Section IV-B introduces a baseline joint placement–routing (link-flow) MILP (LF-MILP), solved to *provable global optimality* under the stated assumptions. Section IV-C introduces a configuration-selection MILP (CS-MILP) that precomputes candidate embeddings for each demand and selects among them via a smaller MILP. These formulations offer complementary trade-offs between optimality and scalability: LF-MILP is exact but computationally expensive at scale, whereas CS-MILP scales better but may exclude the global optimum. Finally, Section IV-D introduces the constraints enforcing data sovereignty in both formulations.

A. Objective Structure

In real NFV deployments, operational constraints and capacity limits may prevent the operator from meeting all demand requirements simultaneously. To reflect this, we do not require all demands to be embedded; instead, we model *admission control* through binary variables $a_r \in 0, 1$ indicating whether demand r is served ($a_r = 1$) or rejected ($a_r = 0$).

We use a lexicographic objective: first maximize *admission value*, then minimize embedding cost among solutions with the maximum admission value. To support weighted admission, each demand r is assigned a nonnegative weight $w_r \geq 0$. Setting w_r equal for all r maximizes the *number* of admitted demands. Varying w_r values can encode priority classes, revenue-based weighting, or throughput-based weighting (e.g., $w_r = b_r$ prioritizes total admitted bandwidth).

We implement the lexicographic objective via a two-phase solve. In Phase 1, we solve the MILP with the objective

$$\max \quad \sum_{r \in \mathcal{R}} w_r a_r \tag{1}$$

and denote the optimal admission value by $A^\star$. In the second phase, we add the constraint

$$\sum_{r \in \mathcal{R}} w_r a_r = A^\star, \tag{2}$$

and re-solve the MILP with the objective of minimizing the embedding cost (formulated in Section IV-B3 and Section IV-C3). This ensures that cost minimization is performed only among solutions that achieve maximum admission.

B. Formulation I: Link-Flow MILP

This formulation decides jointly (i) where each VNF stage of each admitted demand is placed, and (ii) how traffic is routed between consecutive stages of the chain. The resulting MILP can be solved to provable global optimality under the stated assumptions, but can be large because it introduces per-demand, per-stage routing variables on links.

1) Decision Variables: To unify placement and routing, we index the chain by stages $k = 0, 1, \ldots, m_r + 1$, where $k = 0$ corresponds to the source, $k = m_r + 1$ to the destination, and $k = 1, \ldots, m_r$ to the *VNF stages*.

- Admission: $a_r \in \{0, 1\}$ equals 1 if demand r is admitted.
- Placement: $y_{r,k,v} \in \{0, 1\}$ equals 1 if stage $k \in \{0, \ldots, m_r + 1\}$ of demand r is at node $v \in V$.
- Routing: $x_{r,k,e} \in \{0, 1\}$ equals 1 if chain segment $k \in \{1, \ldots, m_r + 1\}$ of demand r uses link $e \in E$, where segment k connects stage $k - 1$ to stage k.

2) Constraints:

a) Endpoint activation: We fix the source and destination positions when (and only when) the demand is admitted:

$$y_{r,0,s_r} = a_r, \qquad y_{r,0,v} = 0 \quad \forall v \in V \setminus \{s_r\}, \tag{3}$$

$$y_{r,m_r+1,t_r} = a_r, \qquad y_{r,m_r+1,v} = 0 \quad \forall v \in V \setminus \{t_r\} \tag{4}$$

b) Exactly one placement per admitted VNF: For each admitted demand r, each VNF (stage k) must be placed once at exactly one node; if r is rejected, no VNF is placed:

$$\sum_{v \in V^{\mathrm{c}}} y_{r,k,v} = a_r, \qquad \forall r \in \mathcal{R},\ \forall k = 1, \ldots, m_r \tag{5}$$

c) Placement domain: Internal VNF stages can only be located at VNF-hosting nodes:

$$y_{r,k,v} = 0, \qquad \forall r \in \mathcal{R},\ \forall k = 1, \ldots, m_r,\ \forall v \in V \setminus V^{\mathrm{c}} \quad (6)$$

d) Flow conservation: Let $\delta^+(v)$ and $\delta^-(v)$ denote the sets of outgoing and incoming directed links of node v, respectively. For each demand r, each chain segment $k = 1, \ldots, m_r + 1$, and each node $v \in V$, we enforce:

$$\sum_{e \in \delta^+(v)} x_{r,k,e} - \sum_{e \in \delta^-(v)} x_{r,k,e} \;=\; y_{r,k-1,v} - y_{r,k,v}, \qquad (7)$$

Flow conservation links routing and placement, enforcing a continuous route between nodes selected for stages $k-1$ and k. When $a_r = 0$, all corresponding y-variables are set to 0 by Eqs. (3)–(5). Setting $x_{r,k,e} = 0$ for all $e \in E$ satisfies Eq. (7).

e) Link capacity: For each link $e \in E$, the aggregate bandwidth consumed by admitted demands (summed over chain segments) must not exceed the link capacity:

$$\sum_{r \in \mathcal{R}} \sum_{k=1}^{m_r+1} b_r \, x_{r,k,e} \le U_e, \qquad \forall e \in E \qquad (8)$$

This constraint couples embedding decisions across demands and captures contention on the transport network.

f) Node processing capacity: For each VNF-hosting node $v \in V^{\mathrm{c}}$, the total load induced by VNFs placed on v must not exceed the available processing capacity:

$$\sum_{r \in \mathcal{R}} \sum_{k=1}^{m_r} \alpha_{f_{r,k}} \, b_r \, y_{r,k,v} \le C_v, \qquad \forall v \in V^{\mathrm{c}} \qquad (9)$$

Here $\alpha_{f_{r,k}} b_r$ is the required processing capacity (CPU units) to process throughput b_r at a VNF of type $f_{r,k}$.

g) E2E latency: The sum of routing delays across chain segments, plus per-VNF processing delays, must not exceed the latency budget for admitted demands:

$$\sum_{k=1}^{m_r+1} \sum_{e \in E} \ell_e \, x_{r,k,e} \;+\; \sum_{k=1}^{m_r} \sum_{v \in V^{\mathrm{c}}} \delta_{f_{r,k}} \, y_{r,k,v} \le L_r \, a_r, \forall r \in \mathcal{R} \qquad (10)$$

This couples the latency feasibility to admission: when $a_r = 1$, Eq. (10) enforces the end-to-end latency budget L_r; when $a_r = 0$, it forces zero latency contribution, consistent with rejected demands having no routed traffic.

3) Objective and Solution Procedure: We solve the LF-MILP using the two-phase lexicographic objective described in Section IV-A. In Phase 1, we solve the problem with the objective in Eq. (1) to obtain the maximum admission value $A^\star$. In Phase 2, we add the constraint in Eq. (2) to restrict the feasible set to only solutions that achieve maximum admission, and re-solve the MILP to minimize the embedding cost, defined as total bandwidth-weighted link cost:

$$\min \quad \sum_{r \in \mathcal{R}} \sum_{k=1}^{m_r+1} \sum_{e \in E} w_e \, b_r \, x_{r,k,e}, \qquad (11)$$

subject to the constraints in Eqs. (3)–(10) and Eq. (2), where $w_e \ge 0$ is a per-link weight.

Practical trade-off.: LF-MILP provides a globally optimal benchmark under the stated assumptions by optimizing over the full placement-and-routing space. Its size is dominated by $\Theta\big(\sum_{r \in R}(m_r + 1)|E|\big)$ routing variables, which becomes computationally expensive for large instances.

C. Formulation II: Configuration-Selection MILP

Our configuration–selection MILP (CS-MILP) is a two-step approach that (i) precomputes for each demand r a set of up to K end-to-end candidate SFC embeddings compliant with the end-to-end latency requirements, and next (ii) utilizes our MILP formulation to maximize the value of admitted demands subject to resource constraints. Similar to Formulation I, this configuration-selection scheme returns at most one end-to-end embedding for each demand r.

Step 1: Precomputation of end-to-end candidate embeddings. For each demand r, let Ω_r denote a set of candidate end-to-end embeddings, indexed by $p \in \Omega_r$. Each candidate $p \in \Omega_r$ is characterized by (i) hosts $h_{r,p,k} \in V^{\mathrm{c}}$ for VNF stages $k = 1, \ldots, m_r$, and (ii) link sets $P_{r,p,k} \subseteq E$ for chain segments $k = 1, \ldots, m_r + 1$.

For each demand r and VNF stage k, we define a host-candidate set $H_{r,k} \subseteq V^{\mathrm{c}}$ of possible locations of VNFs of type $f_{r,k}$. To include all possible candidates, for each stage k, set $H_{r,k} = V^{\mathrm{c}}$. However, enumerating candidates over all host combinations can be computationally expensive when $|V^{\mathrm{c}}|$ is large. For large networks, we reduce preprocessing time by restricting each stage to a host-candidate set $H_{r,k} \subseteq V^{\mathrm{c}}$ consisting of a limited number q of VNF-hosting nodes v with the smallest proximity score given by:

$$\mathrm{score}_r(v) = d_w(s_r, v) + d_w(v, t_r), \qquad (12)$$

where $d_w(\cdot, \cdot)$ is shortest-path distance under link weights w_e. This heuristic favors hosts closer to the demand endpoints, reducing latency while keeping a manageable candidate set.

For each demand r, to determine the end-to-end candidate SFC embeddings, we:

(a) first build a *multi-stage graph* illustrated in Fig. 2 that includes s_r and t_r as the source and destination, respectively. In this graph:
  - each stage k consists of nodes $H_{r,k}$,
  - for each pair of consecutive stages $k-1$ and k, the weights of edges directed from vertices of $H_{r,k-1}$ to $H_{r,k}$ are equal to the costs of the cheapest paths in the physical network between the corresponding nodes. For each pair of consecutive stages $k-1$ and k, the sets $H_{r,k-1}$ and $H_{r,k}$ together with the related directed edges form complete bipartite subgraphs.

(b) next compute the top K shortest paths in that multi-stage graph. For each host sequence, we reconstruct the full embedding by routing each chain segment on a shortest path in G, compute the end-to-end cost, discard infeasible candidates ($lat_{r,p} > L_r$), and keep the top K.

Step 2: After obtaining for each demand r the candidate end-to-end embeddings p, we derive resource-consumption vectors

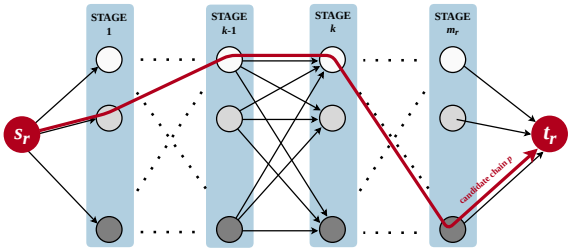


Fig. 2: Illustration of our multi-stage graph concept

used in the MILP. Link usage $u^{\text{link}}_{r,p}(e)$ is defined in Eq. (13), where $n_{r,p}(e)$ counts how many chain segments traverse link e under candidate p. Processing capacity usage $u^{\text{cpu}}_{r,p}(v)$, latency $lat_{r,p}$, and embedding cost $\kappa_{r,p}$ are defined in Eqs. (14)–(16).

$$u^{\text{link}}_{r,p}(e) = b_r \cdot n_{r,p}(e), \qquad \forall e \in E \tag{13}$$

$$u^{\text{cpu}}_{r,p}(v) = \sum_{k=1}^{m_r} \mathbf{1}\{h_{r,p,k} = v\}\, \alpha_{f_{r,k}}\, b_r, \qquad \forall v \in V^{\text{c}} \tag{14}$$

$$lat_{r,p} = \sum_{k=1}^{m_r+1} \sum_{e \in P_{r,p,k}} \ell_e + \sum_{k=1}^{m_r} \delta_{f_{r,k}} \tag{15}$$

$$\kappa_{r,p} = \sum_{e \in E} w_e\, u^{\text{link}}_{r,p}(e) \tag{16}$$

1) Decision Variables:

- Admission: $a_r \in \{0,1\}$ indicates if demand r is served.
- Embedding selection: $z_{r,p} \in \{0,1\}$ equals 1 if candidate embedding $p \in \Omega_r$ is selected for demand r, 0 otherwise.

Remark: The admission variable a_r is implied by the selection variables via Eq. (17) (i.e., $a_r = \sum_{p \in \Omega_r} z_{r,p}$) and is therefore not strictly necessary. However, we retain a_r to keep the notation and the lexicographic objective Eq. (1) consistent across both MILP formulations. It introduces one additional binary variable per demand, which is typically negligible compared to the number of selection variables $z_{r,p}$ (i.e., $\sum_{r \in \mathcal{R}} |\Omega_r|$).

2) Constraints:

a) Choose exactly one embedding if admitted:

$$\sum_{p \in \Omega_r} z_{r,p} = a_r, \qquad \forall r \in \mathcal{R} \tag{17}$$

This enforces that an admitted demand selects exactly one candidate embedding; if $a_r = 0$, no embedding is selected.

b) Link capacity:

$$\sum_{r \in \mathcal{R}} \sum_{p \in \Omega_r} u^{\text{link}}_{r,p}(e)\, z_{r,p} \le U_e, \qquad \forall e \in E \tag{18}$$

This aggregates the precomputed link usage of all selected embeddings and enforces physical link capacities.

c) Node processing capacity:

$$\sum_{r \in \mathcal{R}} \sum_{p \in \Omega_r} u^{\text{cpu}}_{r,p}(v)\, z_{r,p} \le C_v, \qquad \forall v \in V^{\text{c}} \tag{19}$$

This aggregates the processing of selected embeddings and enforces processing-capacity limits at VNF-hosting nodes.

3) Objective and Solution Procedure: We solve the CS-MILP using the two-phase lexicographic objective described in Section IV-A. In Phase 1, we solve the model with the objective in Eq. (1) to obtain the maximum admission value $A^\star$. In Phase 2, we add the constraint from Eq. (2) to restrict the feasible set to solutions with maximum admission, and re-solve with the objective of minimizing the embedding cost.

To ensure a fair comparison between the LF-MILP and CS-MILP, we define the cost of each candidate embedding $p \in \Omega_r$ as $\kappa_{r,p}$ in Eq. (16), consistent with the bandwidth-weighted objective used in the LF-MILP. The Phase 2 objective is then to minimize the total cost of selected candidate embeddings

$$\min \quad \sum_{r \in \mathcal{R}} \sum_{p \in \Omega_r} \kappa_{r,p}\, z_{r,p} \tag{20}$$

subject to the constraints in Eqs. (17)–(19) and Eq. (2). Since $\kappa_{r,p}$ is computed using the same link weights w_e as in Eq. (11), the two formulations minimize the same bandwidth-weighted cost. Performance gaps are thus attributable to the restricted candidate sets $\{\Omega_r\}_{r \in \mathcal{R}}$.

Practical trade-off.: CS-MILP improves scalability by selecting from candidate embeddings. For all r, if $|\Omega_r| \le K$, the MILP has $O(|R|K)$ selection variables; the trade-off is that solution quality depends on candidate coverage.

D. Sovereignty Formulation

We model *data sovereignty* as hard constraints that restrict (i) where traffic may be processed (VNF placement) and (ii) where traffic may transit (routing). For each demand $r \in \mathcal{R}$, we are given demand- and stage-specific allowed sets: for each VNF stage $k = 1, \ldots, m_r$, an allowed set of processing nodes $V^{\text{proc}}_{r,k} \subseteq V^{\text{c}}$, and for each chain segment $k = 1, \ldots, m_r + 1$, an allowed set of transit links $E^{\text{tr}}_{r,k} \subseteq E$. These sets are external policy inputs and may vary across stages and segments. Transit-node restrictions can be enforced by removing all incident links of a forbidden node from $E^{\text{tr}}_{r,k}$. A concrete zone-based instantiation is provided in Section V.

1) Integration into the Link-Flow Formulation: In the LF-MILP (Section IV-B), sovereignty is enforced by restricting the placement and routing domains to the demand-specific allowed sets $V^{\text{proc}}_{r,k}$ and $E^{\text{tr}}_{r,k}$. All other constraints remain unchanged.

a) Sovereignty-compliant placement: For each demand $r \in \mathcal{R}$ and VNF stage $k = 1, \ldots, m_r$, we forbid placement on VNF-hosting nodes outside the allowed processing set:

$$y_{r,k,v} = 0, \quad \forall r \in \mathcal{R},\ \forall k = 1, \ldots, m_r,\ \forall v \in V^{\text{c}} \setminus V^{\text{proc}}_{r,k} \tag{21}$$

Together with the placement constraint (5), this ensures that admitted demands place each VNF stage on an allowed node.

b) Sovereignty-compliant transit: For each demand $r \in \mathcal{R}$ and chain segment $k = 1, \ldots, m_r + 1$, we forbid routing on links outside the allowed transit set:

$$x_{r,k,e} = 0, \quad \forall r \in \mathcal{R},\ \forall k = 1, \ldots, m_r + 1,\ \forall e \in E \setminus E^{\mathrm{tr}}_{r,k} \tag{22}$$

With (7), this guarantees that each segment is routed exclusively on allowed transit links. The domain restrictions in Eqs. (21)–(22) can be interpreted as removing disallowed variables from the model (equivalently, fixing them to zero).

2) Integration into the Configuration-Selection Formulation: During preprocessing, we retain only candidates $p \in \Omega_r$ that satisfy the processing and transit constraints; hence all candidates in Ω_r are sovereignty-compliant by construction:

$$h_{r,p,k} \in V^{\mathrm{proc}}_{r,k}, \qquad k = 1, \ldots, m_r, \tag{23}$$

$$P_{r,p,k} \subseteq E^{\mathrm{tr}}_{r,k}, \qquad k = 1, \ldots, m_r + 1. \tag{24}$$

Here, $h_{r,p,k}$ denotes the stage-k host and $P_{r,p,k} \subseteq E$ the set of links used by chain segment k in candidate p. In practice, we enforce sovereignty already during candidate generation by restricting shortest-path computations to the allowed link sets $E^{\mathrm{tr}}_{r,k}$ and by restricting host candidates to $H_{r,k} \subseteq V^{\mathrm{proc}}_{r,k}$. Therefore, no changes to the CS-MILP are needed.

V. Experimental Setup

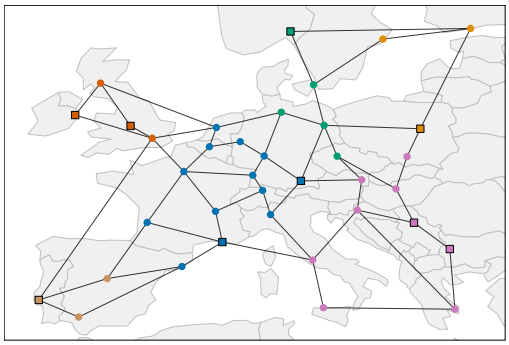
A. Topologies

In our experiments, we used topologies of real backbones from the Internet Topology Zoo [15] and SNDlib [16], depicted in Fig. 3. We converted all inputs to a directed graph representation (two directed arcs replaced undirected links) and set propagation delays (ℓ_e) based on geographic distances. Given the longitude and latitude coordinates, we computed the haversine distance and converted it to milliseconds assuming a fixed fiber-propagation speed of 2×10^8 m/s. For embedding-cost weights, we set $w_e = \ell_e$ so that the objective function focuses on minimizing bandwidth–delay. We assigned uniform bandwidth $U_e = U_0$ for all $e \in E$ and selected a set of VNF-hosting nodes $V^{\mathrm{c}} \subseteq V$ by sampling a fraction ρ_c of nodes uniformly at random. VNF-hosting node placements were resampled in each experimental iteration. For each compute node $v \in V^{\mathrm{c}}$, we assigned a uniform capacity $C_v = C_0$.

B. Sovereignty Policies

As outlined in Section IV-D, the MILPs support sovereignty policies for processing nodes, transit links, and indirectly, transit nodes. In the experiments, we focused on sovereignty in VNF-hosting nodes, as data sovereignty requirements are normally expressed as constraints on where data is stored and/or processed. Transit constraints are comparatively less harmonised [8]. For transit sovereignty policies or future studies, the MILPs support transit constraints in (22) and (24).

For VNF processing policies, the MILPs allow fine-grained constraints (e.g., forcing stage k of request r to specific nodes via $V^{\mathrm{proc}}_{r,k}$). In this paper, we defined sovereignty *zones* to emulate common data sovereignty policies: sovereignty is often formulated as requirements that data must be stored and/or processed within a given jurisdiction [8], [7]. We implemented zones as a non-overlapping partition of nodes via a node-to-zone mapping $\zeta : V \to \mathcal{Z}$: we clustered node locations into $|\mathcal{Z}|$ clusters (k-means) and assigned each node to exactly one cluster. For a zone $z \in \mathcal{Z}$, we set $V(z) = \{v \in V : \zeta(v) = z\}$.



(a) Cost266 from SNDlib

(b) Cogentco from Topology Zoo

(c) BtNorthAmerica from Topology Zoo.

Fig. 3: Topologies used in our evaluations. Round nodes are transit nodes, while square nodes are nodes capable of hosting VNFs.

C. Demand Generation

In NFV deployments, network resources are often shared by multiple *tenants* (e.g., distinct customers or slices) whose traffic is logically separated. Accordingly, we generated network traffic as *demands* grouped into *tenants*. Let $\mathcal{T}$ be a set of tenants, and let tenant $\tau \in \mathcal{T}$ be assigned a set of endpoint nodes $\mathcal{S}_\tau \subseteq V$. For each unordered endpoint pair $\{u, v\} \subseteq \mathcal{S}_\tau$ we generated one demand $r = (s_r, t_r, \mathbf{f}_r, b_r, L_r)$ between (u, v), yielding a per-tenant demand set with a maximum size of $\binom{|\mathcal{S}_\tau|}{2}$. For each generated unordered pair $\{u, v\}$, the demand direction $(s_r, t_r) \in \{(u, v), (v, u)\}$ was randomly chosen from an uniform distribution.

We constructed tenants sequentially by sampling an endpoint set $\mathcal{S}_\tau$ for each new tenant τ and generated one demand per unordered endpoint pair $\{u, v\} \subseteq \mathcal{S}_\tau$. We repeated this process until the total demand set reached a target size $|\mathcal{R}|$.

Each tenant τ was assigned two independent attributes: (i) *zone span*: it is *single-zone* with probability $1 - \rho_{\mathrm{mz}}$ or *multi-zone* with probability ρ_{mz}, and (ii) a *sovereignty flag*: it is sovereignty-constrained with probability ρ_{sv}. Single-zone

TABLE I: Base parameters for experiments

Parameter	Value	Description
ρ_c	0.3	VNF-hosting node fraction
$\lvert\mathcal{Z}\rvert$	4	Num. of sovereignty zones
U_0	1000 Mbps	Per-link nominal capacity
C_0	2500 CPU units	VNF-hosting node capacity
w_e	ℓ_e	Per-link cost weight
$\lvert\mathcal{S}_\tau\rvert$	4	Endpoints per tenant
$\lvert\mathcal{R}\rvert/\lvert V\rvert$	3	Demand density (per node)
m_r	$\in \{2,3,4\}$	Service-chain length
$\mathcal{F}$	$\{\mathrm{FW}, \mathrm{NAT}, \mathrm{DPI}\}$	VNF-type set
α_f	$[1.0, 0.5, 2.0]$ (ordered as $\mathcal{F}$)	CPU/Mbps by VNF type
δ_f	$[0.10, 0.05, 0.20]$ ms (ordered as $\mathcal{F}$)	Proc. delay by VNF type
b_r	$\in \{50, 75, 100\}$ Mbps	Demand throughput
γ	2.5	Latency slack factor
ρ_{sv}	0.8	Sovereign-tenant probability
ρ_{mz}	0.75	Multi-zone-tenant probability

tenants chose a designated zone $z_\tau \in \mathcal{Z}$ and drew all endpoints from that zone, e.g., $\mathcal{S}_\tau \subseteq V(z_\tau)$. Multi-zone tenants chose a fixed set of zones $\mathcal{Z}_\tau \subseteq \mathcal{Z}$ with $|\mathcal{Z}_\tau| \geq 2$ and drew endpoints across these zones. This generation process yielded both intra- and inter-zone demands, and the tenant attributes induced per-demand, per-stage allowed processing-node sets $V^{\mathrm{proc}}_{r,k}$. In our experiments, transit was unrestricted for all demands, giving $E^{\mathrm{tr}}_{r,k} = E$ for all chain segments k of all demands r.

We instantiated one sovereignty policy per demand and applied it for all VNF stages of that demand. Non-sovereign demands used the common processing policy (V^{c}). Sovereignty-constrained single-zone demands were restricted to their designated zone: $V(z_\tau) \cap V^{\mathrm{c}}$, where z_τ is the request's designated zone. For each sovereignty-constrained multi-zone demand, we sampled either a source-zone or destination-zone policy: $z^\star_r \in \{\zeta(s_r), \zeta(t_r)\}$, with the admissible set $V(z^\star_r) \cap V^{\mathrm{c}}$.

Each demand $r = (s_r, t_r, \mathbf{f}_r, b_r, L_r)$ was generated as follows: Generate (s_r, t_r) from a tenant endpoint pair as described above. Sample the chain length m_r uniformly from $\{2,3,4\}$, and each VNF type $f_{r,k}$ i.i.d. from a VNF-type set $\mathcal{F}$. Sample throughput, b_r uniformly from $\{50, 75, 100\}$ Mbps. For the latency budget L_r, let $d_\ell(s_r, t_r)$ be the shortest-path delay from s_r to t_r under link delays ℓ_e. We set

$$L_r \;=\; \gamma \cdot d_\ell(s_r, t_r) \;+\; \sum_{k=1}^{m_r} \delta_{f_{r,k}}, \tag{25}$$

with slack factor $\gamma \geq 1$ and VNF-type processing delays $\delta_{f_{r,k}}$.

Unless stated otherwise, we used the base parameters in Table I. We chose values to put the system near the processing capacity threshold, so sovereignty constraints would meaningfully affect admission decisions. The latency slack factor γ was set to 2.5 to allow path diversity while still binding latency for longer detours. VNF processing coefficients α_f and delays δ_f were selected to represent heterogeneous VNF costs (light: Network Address Translation (NAT), moderate: firewall (FW), heavier: Deep Packet Inspection (DPI)).

VI. Results

All experiments were conducted on the three backbone topologies in Fig. 3. Since the qualitative trends we observed were consistent across topologies, we present only Cost266 results in the remainder of this section; the other two topologies exhibited the same trends in every experiment.

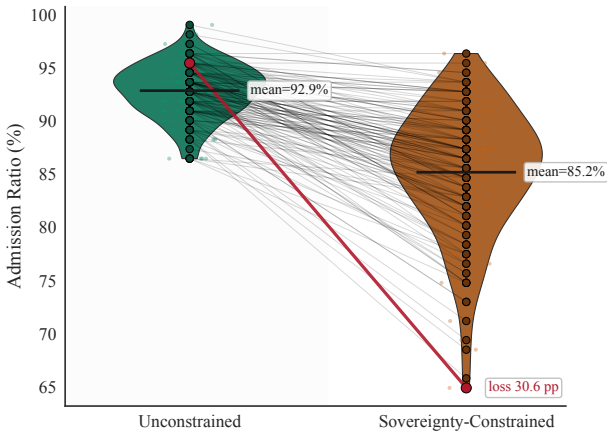


Fig. 4: Admission ratio across iterations, comparing unconstrained vs. sovereignty-constrained demands on identical instances. Lines connect paired iterations; violin plots show distributions. The red line marks the maximum observed per-iteration admission loss.

A. Cost of Sovereignty

We quantified the impact of sovereignty constraints with a paired design. Per iteration, we solved the same input instance (topology, VNF-host nodes, and demands) twice using the LF-MILP from Section IV-B to obtain globally optimal solutions: (i) with sovereignty constraints enabled (*sov.-constrained*) and (ii) with no sovereignty constraints (*unconstrained*).

Figure 4 shows demand admission ratios, comparing unconstrained vs. sovereignty-constrained demands for identical instances. Each grey line connects the paired solves for one iteration; violin plots show the distribution across iterations. Sovereignty constraints reduce the mean admission ratio from 92.9% to 85.2%, i.e., by 7.7 percentage points (pp), with a maximum observed per-iteration drop of 30.6 pp (highlighted in red). These results show that sovereignty restrictions can substantially reduce admitted demands even with unchanged underlying network and traffic instances.

B. Load Sensitivity of the "Sovereignty Penalty"

To quantify how the impact of sovereignty constraints varies with network load, we ran a sensitivity analysis, sweeping the demand-load multiplier $|\mathcal{R}|/|V|$ from 1 to 6. Per load, we solved paired instances across 50 iterations: once with sovereignty constraints enabled and once without. We kept topology, VNF-host nodes, and demands identical per pair.

Figure 5 reports the difference in admission ratio together with the average load of the VNF-hosting nodes in the network for variable load multipliers, with 95% confidence intervals. We show the admission ratio for each load and the admission penalty (unconstrained − sov.-constrained), and observe a clear non-monotone pattern. The difference increases from low load and peaks around load multiplier 3.0, where admission was 93.7% without sovereignty and 86.4% with sovereignty (7.24

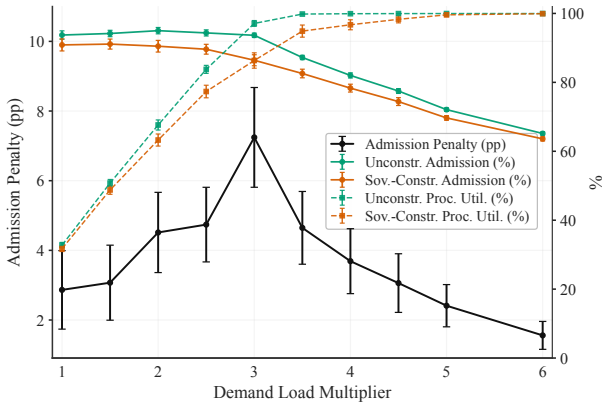


Fig. 5: Demand-load sensitivity: The black line shows the admission-ratio gap between runs with/without sovereignty constraints. Solid lines show admission ratio (left y-axis), and dashed lines show mean utilization (right y-axis) of VNF-hosting nodes. Red denotes runs with sovereignty constraints; green denotes runs without. Points are paired means over 50 iterations; error bars show 95% confidence intervals.

pp difference). Notably, this peak occurred near the start of processing capacity saturation, not at maximum utilization: at a load multiplier of 3.0, mean VNF-hosting node utilization was 97.1% versus 86.3%, while at a load multiplier of 6.0 both were saturated, and the difference shrank to 1.6 pp.

C. Runtime and Performance Trade-offs between LF-MILP and CS-MILP formulations

Here, we compared the runtime and admission trade-offs of LF-MILP (Section IV-B) and CS-MILP (Section IV-C). As a baseline, we also implemented a random-order greedy configuration-selection heuristic (CS-Greedy) using the same precomputed candidate set as CS-MILP: demands were processed sequentially in random order and assigned their lowest-cost feasible candidate embedding under the residual link and node capacities, or rejected if no feasible candidate existed.

We evaluated two scenarios: **(i) Static**, where each method computed an embedding for the initial demand set $\mathcal{R}$ from scratch; and **(ii) Dynamic**, where between re-optimization epochs, a fraction ρ of active demands were replaced. We set

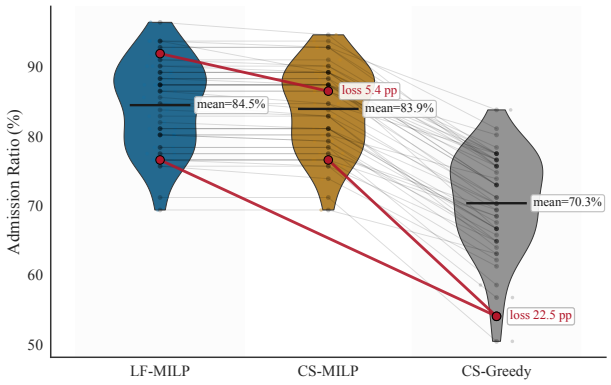


Fig. 6: Admission ratio across iterations under LF-MILP, CS-MILP, and a greedy baseline. Red lines show the maximum observed loss.

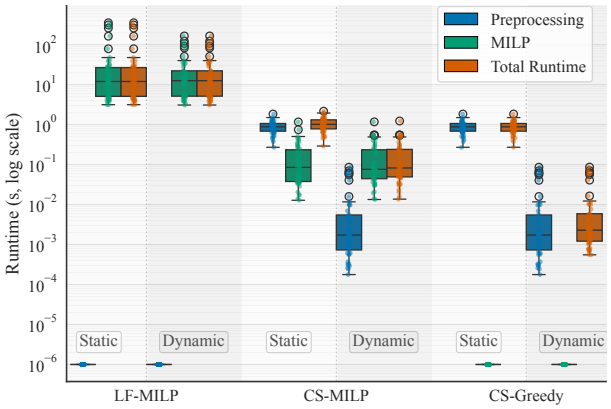


Fig. 7: Runtimes of LF-MILP, CS-MILP, and the greedy heuristic in the *Static* scenario (initial solve from scratch) and the *Dynamic* scenario (re-optimization after demand churn).

$\rho = 1\%$ so that the number of active demands (and hence the problem size) remained comparable across epochs.

Figure 6 shows the admission ratio in the Static scenario. Red annotations indicate the maximum per-iteration admission loss. CS-MILP achieved admission comparable to LF-MILP: the mean admission differs by 0.6 pp (84.5% for LF-MILP vs. 83.9% for CS-MILP). In contrast, the greedy baseline incurred a substantial admission loss (mean 70.3%).

Figure 7 compares runtimes in the Static and Dynamic scenarios. In the Static scenario, LF-MILP spent all runtime in MILP solving, while CS-MILP was dominated by candidate preprocessing and thus achieved much shorter solve times. In the Dynamic scenario, CS-MILP was 12.35× faster than in Static because preprocessing was required only for newly added demands. LF-MILP, however, had similar runtimes in both scenarios because it could not reuse preprocessing and had to re-solve the full problem after churn. Consequently, LF-MILP was 11.82× slower than CS-MILP in Static and 152.67× slower in Dynamic, underscoring CS-MILP's benefit in online settings. The greedy baseline was fastest, but at the cost of lower admission (Fig. 6).

VII. Discussion

A central takeaway is that *data sovereignty changes the nature of virtual resource allocation*. NFV orchestration often assumes that VNFs can be placed wherever spare capacity exists. However, with sovereignty, we must take into account hard, domain-specific constraints on *where* traffic may be processed and which domains it may traverse. Modeling sovereignty as per-demand *allowed sets* over hosting nodes and transit links turns policies into directly enforceable feasibility constraints inside the embedding problem, rather than an external post-processing step. This abstraction allows regulatory or contractual policies to map to allowed sets without changing the embedding formulations.

Quantitatively, we observed a clear *sovereignty penalty*: enforcing sovereignty reduced the feasible embedding space and substantially lowered admission. The penalty was strongest

during the transition-to-congestion phase: unconstrained demands could exploit geographic/administrative flexibility to avoid bottlenecks, whereas sovereignty-constrained demands could not. At very high load, when processing capacity was saturated, both cases became dominated by the same scarce resources, and the incremental penalty of sovereignty was reduced. Operationally, this suggests that mitigation lies in *local* provisioning and traffic steering: (i) ensuring sufficient processing capacity within each sovereignty zone, and (ii) placing unconstrained demands outside those zones to preserve the scarcer sovereign capacity. Evaluating such planning and steering strategies is a natural direction for future work.

A second key point is *how to enforce sovereignty at scale*. The LF-MILP provides a provably globally optimal benchmark, valuable for planning, but can be costly for larger instances or frequent re-optimizations. The CS-MILP addresses this by splitting the problem into two steps: first generating sovereignty-compliant end-to-end *candidates* on a layered multi-stage graph, and then solving a smaller resource-aware selection MILP over those candidates. This is especially attractive in practice because topologies are usually stable and demand patterns change slowly: candidate generation can be performed offline and reused across re-optimizations, adding candidates only for new demands under churn. The trade-off is that solution quality depends on candidate coverage; too small candidate sets can exclude good embeddings. Since the CS-MILP depends on the quality of the precomputed candidate set, an important direction for future work is to systematically study how candidate-generation and parameter choices affect runtime gains, robustness, and solution quality.

Finally, our evaluation isolated the impact of sovereignty under a specific set of modeling choices. Extending the framework to more varied policies (e.g., link-level transit sovereignty, overlapping jurisdictions), richer VNF resource models (multi-resource, stateful functions), and explicit reconfiguration costs would improve realism and help quantify the full operational cost of sovereignty-aware orchestration.

VIII. Conclusion

We reframed SFC embedding to explicitly account for *data sovereignty* by modeling sovereignty as per-demand allowed sets over VNF hosting nodes and transit links. We developed two *sovereignty-aware* optimization tools: (i) a link–flow MILP benchmark (solved to provable global optimality), and (ii) a scalable configuration–selection MILP that generates sovereignty-compliant end-to-end candidates on a layered multi-stage graph and then solves a smaller MILP problem. The latter exploits operational stability: candidate generation can be performed offline on stable topologies and reused across re-optimizations, requiring only the addition of new candidates for new demands under churn.

Using topologies from real backbone networks and mixed-traffic workloads, we quantified the resulting "sovereignty penalty", showing that sovereignty constraints can substantially reduce feasibility and admission. We observed the strongest effects in the "transition-to-congestion" phase, where placement flexibility is most valuable. Overall, sovereignty is a structural factor that should be integrated into orchestration and capacity planning. Future work should study more varied sovereignty policies (including transit link and node constraints), richer VNF resource models (multi-resource and stateful functions), and scalable online methods with adaptive candidate generation and refresh under demand churn.

Acknowledgment

This work has received funding from the Research Council of Norway through the SFI Norwegian Centre for Cybersecurity in Critical Sectors (NORCICS) project no. 310105.

The contribution of Jacek Rak to this paper was provided in part during his research fellowship stay in 2025 at the Dept. of Information Security and Comm. Tech. of NTNU – Norwegian University of Science and Technology, Trondheim, Norway.

References

[1] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, "Network Function Virtualization: State-of-the-Art and Research Challenges," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 236–262, 2016.

[2] 3rd Generation Partnership Project (3GPP), "System architecture for the 5G system (5GS)," 3GPP, Technical Specification (TS) 3GPP TS 23.501, Dec. 2025, version number: 20.0.0.

[3] F. Bari, S. R. Chowdhury, R. Ahmed, R. Boutaba, and O. C. M. B. Duarte, "Orchestrating Virtualized Network Functions," *IEEE Transactions on Network and Service Management*, vol. 13, no. 4, pp. 725–739, Dec. 2016.

[4] A. Tomassilli, F. Giroire, N. Huin, and S. Pérennes, "Provably Efficient Algorithms for Placement of Service Function Chains with Ordering Constraints," in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, Apr. 2018, pp. 774–782.

[5] N. Huin, B. Jaumard, and F. Giroire, "Optimal Network Service Chain Provisioning," *IEEE/ACM Transactions on Networking*, vol. 26, no. 3, pp. 1320–1333, Jun. 2018.

[6] N. F. V. E. I. Specification, "Group ETSI GR NFV 001 V1. 3.1 network functions virtualisation (NFV); use cases. 2021."

[7] European Commission, "Adequacy decisions: How the EU determines if a non-EU country has an adequate level of data protection," 2026.

[8] J. L. González, F. Casalini, and J. Porras, "A Preliminary Mapping of Data Localisation Measures," *OECD Trade Policy Papers*, Jun. 2022.

[9] B. Addis, D. Belabed, M. Bouet, and S. Secci, "Virtual network functions placement and routing optimization," in *2015 IEEE 4th International Conference on Cloud Networking (CloudNet)*, Oct. 2015, pp. 171–177.

[10] A. Mouaci, E. Gourdin, I. Ljubic, and N. Perrot, "Virtual network functions placement and routing problem: Path formulation," in *2020 IFIP networking conference (networking)*. IEEE, 2020, pp. 55–63.

[11] ——, "Two extended formulations for the virtual network function placement and routing problem," *Networks*, vol. 82, no. 1, pp. 32–51, 2023.

[12] 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, Jun. 2017.

[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] N. Torkzaban and J. S. Baras, "Trust-Aware Service Function Chain Embedding: A Path-Based Approach," in *2020 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, Nov. 2020, pp. 31–36.

[15] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The Internet Topology Zoo," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 9, pp. 1765–1775, Oct. 2011.

[16] S. Orlowski, R. Wessäly, M. Pióro, and A. Tomaszewski, "SNDlib 1.0—Survivable Network Design Library," *Networks*, vol. 55, no. 3, pp. 276–286, May 2010.