Network Abstraction with Provisioning Guarantees for Multi-Domain Virtual Network Embedding

Yanis Achaichia
Université Côte d'Azur, Inria, I3S UMR CNRS 7271
Sophia Antipolis, France
yanis.achaichia@inria.fr

Christelle Caillouet
Université Côte d'Azur, Inria, I3S UMR CNRS 7271
Sophia Antipolis, France
christelle.caillouet@univ-cotedazur.fr

Nicolas Huin
IMT Atlantique, IRISA UMR CNRS 6074
Rennes, France
nicolas.huin@imt-atlantique.fr

Geraldine Texier
IMT Atlantique, IRISA UMR CNRS 6074
Rennes, France
geraldine.texier@imt-atlantique.fr

Abstract

Network Function Virtualization and Software Defined Network have opened the way to network resources as market commodities. However, establishing slices over several Infrastructure Providers (InPs) implies a multi-domain approach and requires confidentiality. In particular, InPs are reluctant to expose their network topology and amount of resources (bandwidth and delay). We devise BLEND, a method that allows providers to construct an abstraction of their network and share a partial view of their resources while maintaining a high degree of confidentiality. Brokers can then aggregate these abstractions to solve the Multi-domain Virtual Network Embedding problem to embed slices with no back-and-forth between them and providers. Moreover, BLEND offers a way for the providers to generate abstractions with different amount of trade-off between delay and bandwidth constraints. We show on realistic and synthetic graphs that BLEND's parameterized approach is crucial as no universal value of α maximizes the service acceptance rate of heterogeneous requests.

1 Introduction

Network Function Virtualization (NFV) and Software Defined Networking (SDN) have fundamentally transformed the design and operation of network infrastructures, opening up the market of network commodities. One of the most significant challenges remains the placement of tailored slices requiring bandwidth, computing and storage capacities offered by Infrastructure Providers (InPs) [14]. This problem is often formalized as Virtual Network Embedding (VNE), which aims to map virtual network requests onto a physical infrastructure while optimizing resource utilization. The challenge is further exacerbated when slices are deployed across several network providers, in a multi-domain environment (MD-VNE), as network operators deliberately conceal their internal topologies and resource capabilities for confidentiality and security reasons [3, 19]. A solution can be to perform the slice placement on a combination of abstracted topologies exhibited by providers with a trade-off between the precision of the exposed resources and the opacity of the information disclosed. The exposed topology abstraction must represent the available resources accurately enough to calculate a good placement, but be sufficiently imprecise to prevent the reconstruction of the topology or the precise deduction of the resources available in the network.

INOC '26, Liège (Belgium)

The central problem in MD-VNE is thus designing an effective network abstraction that enables operators to collaborate with external tenants or partners without disclosing their full topology, while still ensuring that the acceptance rate of virtual network requests remains as high as possible, ideally, comparable to what would be achieved with full topology disclosure [4]. We propose BLEND, a network abstraction generator that exposes only the peering points of the network (which is public information) and advertises relevant aggregated bandwidth and delay resources. These abstractions are used to confidentially solve the MD-VNE problem and ensure the quality of the slice placement.

After reviewing existing work (Section 2), we propose an abstraction method (Section 3) for performing slice placement in a multi-domain environment, and show that *under-provisioned* abstracted topologies guarantee the confidentiality of InPs' resources. We define, in Section 4, BLEND a model to compute an InP topology abstraction solving the confidentiality issue inherent to the multi-domain context using public topology (a mesh of the peering points). It uses an adaptive parameter α to tune the exposed resources to the slice requests. We study, in Section 5, the influence of α on the relationship between bandwidth, delay and the acceptance rate.

2 Related Work

Since the introduction of NFV, slice placement has become a widely studied topic through the lens of multiple different objectives, from maximising Service Acceptance (SA) [17], to minimising energy consumption [16, 21] or embedding costs [4, 7]. However, most studies consider a complete view of the underlying network topology, which is not always available. In particular, in multi-domain environments, such a level of information disclosure cannot reasonably be expected from the participating providers.

In the multi-domain context, the goal of the VNE problem equates to reaching a consensus on the distribution of a slice amongst the InPs, who will then embed their section of the slice without ever needing to reveal sensitive information about their topology. Such a consensus can be achieved through an auction system or, more generally, distributed algorithms. Zaheer et al. [20] present V-mart, a two-stage Vickrey auction protocol where InPs bid over slices to embed. Samuel et al. [15] present PolyViNE, a distributed tree searching protocol where a broker requests an embedding to InPs that can either accept and propose an embedding with its price, or reject if they are not interested. InPs that cannot fully embed a requested slice can, in their turn, request sub-slice embedding to other InPs, resulting in a tree searching structure. It allows InPs to partake in the search for

10.48786/inoc.2026.07

the best solution, making good use of their knowledge of their network as well as giving them the freedom of not needing to disclose topological information. However, such methods scales poorly due to messages overhead and long convergence times.

Centralized Algorithms gather, through an external coordinator, abstracted topological information from any relevant InP to construct an estimation of a global view of the substrate network. From this view, it can then compute an embedding to get an idea of the slice partitioning to then send each embedding request to the corresponding InP. Joshi and Kataoka [5] propose pSMART, a multi-step Service Function Chaining (SFC) algorithm. This paper also reintroduces a metric for measuring the privacy preserving qualities of different models in the form of a Privacy Index which was originally brought up in a paper by Nepali and Wang [9] on privacy monitoring in social networks. It also makes use of a biasing algorithm that memorises good candidates so as to reduce search time in future iterations. Such algorithms provide good scalability however the lack of precise information about the substrate topologies can result in poor solution quality.

The *"One Big Switch"* abstraction, widely adopted in software-defined networking, represents an entire network as a single logical node, thereby hiding internal topology, routing decisions, and resource constraints from the control plane [6]. While this abstraction simplifies network programmability, it offers limited support for addressing the VNE problem, which requires explicit knowledge of available network resources to map virtual nodes and links onto physical infrastructure. In contrast, a network abstraction that exposes only peering nodes preserves domain-level confidentiality while still necessitating the abstraction of available resources to enable feasible and efficient VNE across domains without revealing internal network structure.

Recently, there has been increasing interest within the IETF in network abstraction and the abstraction of computing resources, reflecting their importance for future programmable infrastructures. The Framework for Abstraction and Control of TE Networks (RFC 8453) defines a framework for abstracting and controlling traffic-engineered networks [2]. Complementary drafts, such as [13] and [8], further extend these principles to operational metrics and network slicing to highlight the importance of abstracting available resources in support of flexible, programmable virtual networks.

3 Virtual network embedding and topology abstraction

The Multi-domain virtual network embedding does not differ from the Virtual Network Embedding problem: the domains are aggregated into a single graph to be solved. To limit the information (topology and capacities) disclosed by the domains, we incur a pre-processing where each domain construct the topology abstraction it will expose. Note that the topology abstraction can also solve scalability issues since it offers an aggregated representation of the resources.

3.1 Virtual network embedding problem

The virtual network embedding problem aims to allocate resources on a substrate network for a given set of slice requests. It maximizes the number of accepted requests according to their resources requirements and the substrate network capacity.

We model the substrate network as a directed graph $G = (V, A)$. Each link $a \in A$ is characterized by its delay D_a and its bandwidth B_a. Each node $v \in V$ is characterized by its storage capacity S_v and its computing power capacity C_v.

Symbol	Definition
$G = (V, A)$	Substrate network graph
V	Set of substrate nodes
A	Set of substrate links
B_a	Bandwidth capacity of arc A
D_a	Delay of arc a
S_v	Storage capacity of node v
C_v	Computational capacity of node v
$r \in \mathscr{R}$	Service request
$H_r = (F_r, L_r)$	Virtual network corresponding to slice r
F_r	Set of virtual network function of slice r
L_r	Set of virtual links of slice r
β_l	Bandwidth requirement for virtual link l of a slice request
δ_l	Delay requirement of virtual link l
σ_f	Storage requirement of function f
γ_f	Computational requirement power for node f
Δ_r	Total delay requirement for the slice r
δ_v^-	Set of incoming arcs of v
δ_v^+	Set of outgoing arcs of v

Table 1: Summary of the notations

We model a slice request $r \in \mathscr{R}$ as a directed graph $H_r = (F_r, L_r)$. Each virtual link $l \in L_r$ requires an amount of bandwidth β_l and cannot exceed δ_l delay. Each virtual node $f \in F_r$ requires σ_f amount of storage, γ_f amount of computing power, and may be constrained to a set of substrate nodes $R^p = (f_1 : v_1, \quad ...)$ Finally, a slice request end-to-end delay cannot exceed Δ_r delay.

Without any confidentiality constraints, we can easily transform the basic Virtual Network Embedding problem to its Multi-domain variant: we define the substrate network G as the disjoint union of subgraphs $\bigcup_{k \in K} G^k = (V^k, A^k)$ that represents each provider's topology, and where domains intersect on the publicly known set of *peering nodes*.

All notations used in the paper are summarized in Table 1.

3.2 Topology abstraction

The centralized MD-VNE problem resolution can be performed by a broker or one of the involved domain. It requires each domain to share its internal topology and resources, which are highly confidential information. The alternative is to transform the information in order to expose a topology abstraction that is sufficiently representative to allow for good slice placements but abstract enough to prevent anybody from reconstructing the original data.

Topology abstractions can be built using either top-down or bottom-up methods. In the first category (and most prevalent in the literature [5, 16]) , i.e., legacy abstraction, we construct a graph by creating a full mesh between the peering nodes and value each of these abstracted edges with appropriate metrics. In the second category, i.e., recursive abstraction, we recursively build the abstraction from the original topology by applying obfuscating operations (e.g., cutting edges, merging nodes [12]) until it is sufficiently compact. We chose to explore legacy abstractions as they offer better confidentiality by only using public information about peering nodes. The main challenge lies in advertising the appropriate resource values onto the abstracted

edges between peering nodes, so that the resulting network abstraction accurately reflects the underlying capabilities without exposing internal topology.

3.3 Abstraction relevance

To evaluate the relevance of the generated abstraction, we compare the set of obtained feasible solutions when embedding slices on the abstracted topology $\mathcal{S}_{\texttt{ABS}}$ and on the original topology $\mathcal{S}_{\texttt{ORI}}$. A good abstraction would lead to the equality of both sets. The aim is to ensure that the broker will not overbook resources allocated to the slices, which would lead to financial losses due to Service Level Agreement (SLA) violation.

We label an abstraction according to the relation between the two sets of feasible solutions:

- if both sets are equal, the abstraction is *perfectly provisioned*, and the broker can find an optimal feasible embedding on the abstraction that is also optimal for the original topology.
- if $\mathcal{S}_{\texttt{ABS}}$ is a subset of $\mathcal{S}_{\texttt{ORI}}$, the abstraction is *under provisioned*, and the broker always provides a feasible embedding solution, but it may not be able to provide the optimal embedding, i.e., the original topology provides more resources than the abstraction.
- if $\mathcal{S}_{\texttt{ABS}}$ is a superset $\mathcal{S}_{\texttt{ORI}}$, the abstraction is *over provisioned*, and the broker may find better solution than in the previous case, but with no guarantee of feasibility in the original topology, i.e., the original topology provides less resources than the abstraction.

Over-provisioned abstractions would lead to messaging overhead as the InP would need to verify solutions against the original topology. Perfectly provisioned abstractions could expose detailed resource information, potentially allowing the underlying multi-domain topology to be inferred. Only under-provisioned abstractions can protect sensitive topology information while ensuring SLA compliance, as the broker only proposes embeddings that the topology can actually support.

4 BLEND – Bandwidth and deLay-awarE Network abstraction Design

We propose BLEND, a legacy abstraction method that generates links between the peering nodes by carefully crafting their bandwidth and their delay. We leverage the multicommodity flow problem to compute a fair bandwidth allocation to the crafted links, and then use the resulting subgraphs to determine the advertised delay.

4.1 Determining Bandwidth

Independent maximum-flows between peering nodes fail to capture the shared nature of bandwidth among the slices. We therefore propose to solve a Multi-Commodity Flow (MCF) problem, with one commodity per peering-node pair. The optimal flow of each commodity will then correspond to the bandwidth advertised on the abstracted link. The objective function for our MCF problem is paramount to obtain an abstraction reflecting the provider's concern. Having no information about forthcoming requests, they may want to maximise the flow traversing the network and, at the same time, to advertise bandwidth equally amongst each abstracted edge. This would share the load on the network links and may help to accept more requests.

The multi-commodity flow problem is formulated as follows. Each pair of peering nodes defines a commodity $c \in \mathscr{C}$ on the original graph. Then, we define three sets of continuous variables

- $x_a^c \in \mathbb{R}^+$, the amount of flow going through the link a for commodity c;
- $f^c \in \mathbb{R}^+$, the total amount of flow passing through the network for commodity c;
- $y \in \mathbb{R}^+$, the minimum total flow over every commodity.

Constraints:

$$\sum_{a \in \delta_v^-} x_a^c = \sum_{v \in \delta_v^+} x_a^c \qquad \forall c \in \mathscr{C}, \forall v \in V \setminus \{s_c,\ t_c\} \qquad (1)$$

$$\sum_{a \in \delta_{s_c}^+} x_a^c = \sum_{a \in \delta_{t_c}^-} x_a^c \qquad \forall c \in \mathscr{C} \qquad (2)$$

$$\sum_{a \in \delta_{s_c}^-} x_a^c = 0 \qquad \forall c \in \mathscr{C} \qquad (3)$$

$$\sum_{c \in \mathscr{C}} x_a^c \leq B_a \qquad \forall a \in A \qquad (4)$$

$$y \leq f^c \qquad \forall c \in \mathscr{C} \qquad (5)$$

$$f^c = \sum_{a \in \delta_{s_c}^+} x_a^c \qquad \forall c \in \mathscr{C} \qquad (6)$$

Objective: Defining an objective function that maximizes advertised bandwidth whilst upholding an even distribution of resources requires a balance between multiple factors. We prioritize the fair distribution of advertised bandwidth before maximizing the total advertised resources within a two-step resolution. We first run the MCF with the goal of maximizing y to find the most even distribution of resources. In a second step, we run the MCF again, adding a new constraint ($y \geq y^*$ where y^* is the solution to the first MCF) to guarantee this even distribution, and with an objective function maximizing the total flow in the network.

$$\max \sum_c f^c$$

On top of allowing a fair advertising of resources, the MCF outputs a set of flows for each commodity. These flows, that we will call *commodity sub-graphs*, are sub-graphs of the original network. In the case where requests are splittable, there is a bijection between the mapping on the abstracted edges and the mapping on the commodity sub-graphs. As a result, the under-provisioning guarantee of the abstraction is preserved with respect to bandwidth. The computed flows on these commodity sub-graphs are used for determining the advertised delay of abstracted edges.

4.2 Determining Delay

Delay and bandwidth are tightly linked, even more when advertising a delay for an abstracted edge. Using the shortest delay between the two peering nodes, whether on the original topology or the previously computed sub-graphs, we can end up over-advertising the amount of bandwidth guaranteed with this delay. If the path guaranteeing this delay provides less bandwidth than the advertised bandwidth, we might use another path with a higher delay. We thus need to introduce the concept of flow-delay as a property combining delay and bandwidth.

To better grasp the idea of flow-delay, we introduce the concept of flow decomposition from graph theory [18]. In this problem, we want to decompose a flow assignment f on a directed graph $G = (V, A)$ into a set of paths $\mathscr{P} = \{p_1, p_2, ..., p_k\}$ with weights $(w_1, w_2, ..., w_k)$ such that the amount of flow on each edge a is equal to the sum of weights of the paths that this edge belongs

to, i.e.,

$$f(a) = \sum_{p_i \in \mathscr{P} | a \in p_i} w_i, \forall a \in P.$$

Considering that the slice requests are splittable over the substrate network, embedding a request amounts to finding a set of paths in the network. For this reason, path decomposition would allow for a better understanding of the actual delay of a given flow assignment.

Computing the delay for a given bandwidth is minimizing the the longest path used in a flow decomposition such that the amount of flow matches the amount of bandwidth advertised.

This problem has a lot of similarities to the Shortest Longest Path (SLP) problem that was shown to be NP-hard [10]. However, the SLP problem considers the flow assignment as an input, which is not our case. To test this, we first designed a simple k-shortest-path-based algorithm minimizing the delay. The idea behind this algorithm is to generate shortest paths until there exists an assignment of the load onto these paths such that the total load matches the advertised bandwidth.

k-shortest paths: Computing the k-shortest paths is solvable in polynomial time with regards to k. This allows us to use such an algorithm to generate paths one by one, checking for each new path whether our stopping criteria has been met. We keep a set of shortest paths $\mathscr{P}'$ that we update at each iteration. Once we reach the stopping criteria, the delay of the last path added to $\mathscr{P}'$ can be advertised as the flow-delay of the abstracted edge. This is because the final path added to $\mathscr{P}'$ is used in the set of paths that satisfy the stopping criteria and has the highest delay of all paths in $\mathscr{P}'$.

Stopping criteria: To figure out if the set of currently generated paths can contain the necessary flow to match the advertised bandwidth, the shared capacities between paths needs to be taken into account. For this reason, we maintain an *interference matrix* M. This matrix provides for each edge in the graph, a list of the generated paths that use this edge. To formalise this, given the current set of shortest paths $\mathscr{P}'$, then for all $(u, v) \in E$, $M_{u,v} = \{p \subset \mathscr{P}' \mid (u, v) \in p\}$.

With this information, the model to check if the stopping criteria is met is defined as follows. Let $G' = (V', E')$ be the current commodity sub-graph considered, and $\mathscr{P}'$ the current set of shortest paths.

We define the set of continuous variables $x_p \in \mathbb{R}^+$ as the amount of flow to assign to path $p \subset \mathscr{P}'$.

The objective is to maximize the amount of flow

$$z^* = \max \sum_{p \in \mathscr{P}'} x_p,$$

while respecting the bandwidth of the edges:

$$\sum_{p \in M_{u,v}} x_p \leq b_{u,v} \qquad \forall (u, v) \in E'$$

where $b_{u,v}$ denotes the bandwidth associated with arc (u, v) in the corresponding commodity sub-graph, this bandwidth is defined as the flow value x_a^c obtained from the model presented in Section 4.1. Then, if z^* is greater or equal to the advertised flow, the stopping criterion is met, and the delay metric to be advertised is equal to the longest delay of a path used in the final solution, in other words, the last path generated in $\mathscr{P}'$.

4.3 Trade-off between bandwidth and delay

Advertising a flow-delay based on the full capacity of the commodity sub-graph will result in the worst case delay metric. It will enable the embedding of slices with a loose delay requirement but it will prevent the embedding of slices that have stronger constraints on delay while requiring less bandwidth. The flow-delay being parameterized by a quantity of necessary flow means that by holding back on advertised bandwidth, we can advertise a lower delay metric. We thus introduce a scalar $\alpha \in [0, 1]$ that determines the amount of flow advertised on the abstraction and therefore also determines the delay on the edges.

For example, when creating an abstraction of the network with a value of $\alpha = 0.5$, each abstracted edge will only advertise 50% of the max flow computed with the MCF. In exchange, the stopping criteria for the flow-delay algorithm based on the k-shortest paths will be reached sooner, as the sum of the load on the generated paths will only need to match 50% of the same maximum flow, thus giving better advertised delays on the abstracted links.

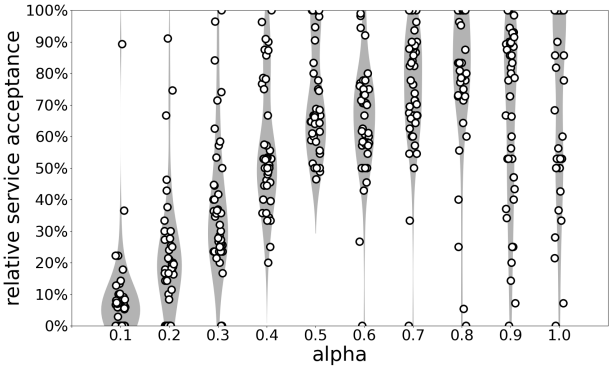

Figure 1: Distribution of the SA of 31 instances of the nobel-germany network topology, relative to their maximal SA when varying α.

5 Results

We evaluate BLEND by first analyzing the impact of the value of α on the quality of the abstraction. We run a slice placement optimal solver proposed by Masoud et al. [17] on the original network (in order to obtain our baseline) and on the abstracted topologies. We compute the ratio of slices embedded on the abstraction over the ones embedded on the original graph. For both resolutions, the objective is to maximize the total number of embedded slices using the same set of requests. slice requests are randomly generated routing demands defined by a source and a destination node, each associated with a peering node, and characterized only by link metrics. The values of requested bandwidth and delay are sampled within a range that suits the metrics of the original network.

We ran tests on the nobel-germany network topology from the SNDlib collection [11], on which we apply BLEND with 10 separate values for α, $[0.1, 0.2, ..., 1.0]$. For each set of slice requirements, the solver attempts to embed them onto 10 variations of the original network parameterized by α. For a given set of requirements, we denote α^* as the value of α that maximizes service acceptance (SA). We can then define the relative service acceptance to be the ratio of an instance SA over its maximal SA, (i.e., $SA_\alpha / SA_{\alpha^*}$). Figure 1 depicts the distribution of relative service acceptance for each set of requests. For low values of α, the SA does not reach its maximal value, and for high values of α, there is a clear dropoff from optimality for many of our instances. This

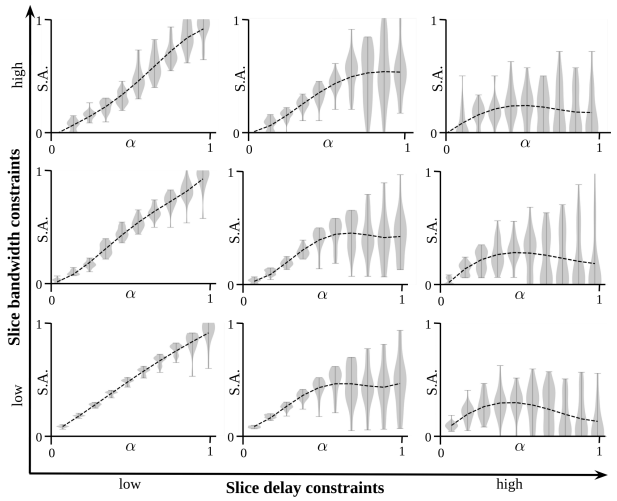

Figure 2: Comparison of the strain on SA caused by bandwidth and delay requirements on the abstraction

supports the idea that introducing an abstraction parameter such as α allows for better fine-tuning of the embedding by the solver.

We then analyze how α^* evolves depending on the types of requests received. In a routing context, the bandwidth and the delay are the two metrics that determine whether a slice may be embedded or not. The tighter the requirements of the requests are on these metrics, the higher the strain it will cause on the solver. We created multiple network instances from the nobel-germany network and added variance on the edge metrics by choosing the bandwidth and delay of each edge from an interval. Service requests were then generated following either a low, a medium, or a high level of strain. We determined these levels of strain to fit within the same order of magnitude as the substrate network's capacities.

Figure 2 shows SA over the previously defined values of α for different levels of strain on both metrics: delay along the x axis and bandwidth along the y axis. The noticeable trend here is that the increase of bandwidth strain has very little impact on SA. The strain applied to delay requirements however, is much more visible. This is because the bandwidth metric of a network dictates the amount of slice requests that can be embedded onto it. Increasing the strain on bandwidth will therefore simply reduce the number of embeddable slices, both for the abstractions and for the substrate network. The delay metric, on the other hand, dictates whether an entire class of slice requests may be embedded onto the network. Increasing the strain on delay can stop whole sets of requests from having feasible embeddings. Figure 2 therefore shows us that the best value for α depends on the type of requests received.

BLEND has a glaring weakness as of now, being the delay computation phase. The algorithm does not provide any meaningful guarantee on its stopping time. We examine the computational time increase for the abstraction process based on the size of the substrate graph. The testing instances are randomly generated substrate networks based on the Barabási-Albert model [1]. This model takes two arguments to construct a network, n representing the number of nodes in the graph, and m representing in a more general way its level of connectivity, a graph created this way will end up with nm edges.

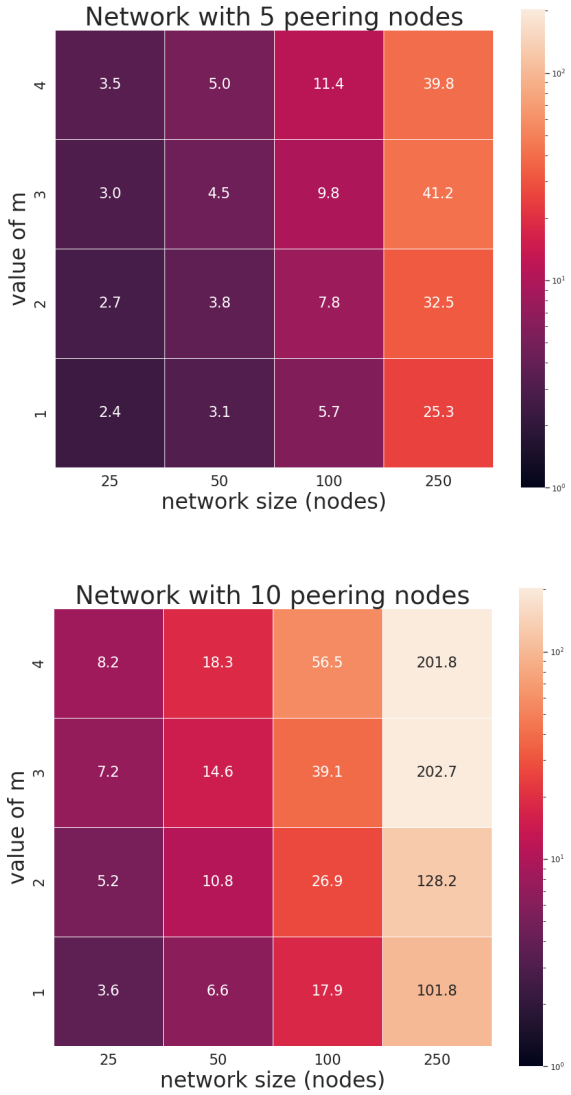

Figure 3: Average computing time in seconds of BLEND on Barabási-Albert graphs with regards to graph size, connectivity, and the number of peering nodes.

Figure 3 is a heatmap representing the average computing time (in seconds) of the abstraction parameterized by these values of m and n. Although the abstraction is computed reasonably fast, we can see that the depicted computing times increase quite drastically due to the high theoretical complexity of the current delay computation. Indeed, it is bounded by the number of simple paths in the network, which scales exponentially with the size of the network.

6 Conclusion

In this paper, we tackle the challenges of designing an abstraction process for multi-domain VNE preserving the confidentiality of each domain while minimizing messaging overhead. We propose BLEND, a method to derive network abstractions exposing only the peering nodes interconnected in a full mesh. The difficulty lies in the computation of both bandwidth and delay advertised on the abstracted topology since it has a direct impact on the service acceptance when embedding slices. We focus the analysis on the relationship between bandwidth and delay and introduce

a configurable parameter α that can help InPs to generate under-provisioned abstractions suited for different bandwidth and delay requirements. Thanks to its commodity-based subgraph representation, BLEND naturally extends to an online setting, as it enables the abstraction to maintain and update a consistent abstract topology over time as requests arrive.

A next step will be to address the scalability of BLEND by considering advanced decomposition methods to further enhance the flow-delay computation. Additionally, we plan to further extend BLEND to include node-based metric abstraction.

Acknowledgments

This work was funded by the French government under the France 2030 ANR program "PEPR Networks of the Future" (ref. ANR-22-PEFT-0002).

References

[1] Albert-László Barabási and Réka Albert. 1999. Emergence of Scaling in Random Networks. *Science* 286, 5439 (1999), 509–512.

[2] Daniele Ceccarelli and Young Lee. 2018. Framework for Abstraction and Control of TE Networks (ACTN). RFC 8453. https://www.rfc-editor.org/info/rfc8453

[3] Josué Castañeda Cisneros, Sami Yangui, Saul E. Pomares Hernández, and Khalil Drira. 2022. A Survey on Distributed NFV Multi-Domain Orchestration from an Algorithmic Functional Perspective. *IEEE Communications Magazine* 60, 8 (2022), 60–65.

[4] David Dietrich, Amr Rizk, and Panagiotis Papadimitriou. 2013. Multi-domain virtual network embedding with limited information disclosure. In *IFIP Networking Conference*. 1–9.

[5] Kalpana D. Joshi and Kotaro Kataoka. 2020. pSMART: A lightweight, privacy-aware service function chain orchestration in multi-domain NFV/SDN. *Computer Networks* 178 (Sept. 2020), 107295.

[6] Nanxi Kang, Zhenming Liu, Jennifer Rexford, and David Walker. 2013. Optimizing the "one big switch" abstraction in software-defined networks. In *9th ACM Conference on Emerging Networking Experiments and Technologies* (Santa Barbara, California, USA) *(CoNEXT)*. 13–24.

[7] Godfrey Kibalya, Joan Serrat-Fernandez, Juan-Luis Gorricho, Doreen Gift Bujjingo, and Jonathan Serugunda. 2021. A Multi-Stage Graph Aided Algorithm for Distributed Service Function Chain Provisioning Across Multiple Domains. *IEEE Access* 9 (2021), 114884–114904.

[8] Daniel King, John Drake, Haomian Zheng, and Adrian Farrel. 2024. *Applicability of Abstraction and Control of Traffic Engineered Networks (ACTN) to IETF Network Slicing*. Internet-Draft. IETF. https://datatracker.ietf.org/doc/draft-ietf-teas-applicability-actn-slicing/10/

[9] Raj Kumar Nepali and Yong Wang. 2013. SONET: A SOcial NETwork Model for Privacy Monitoring and Ranking. In *IEEE 33rd International Conference on Distributed Computing Systems Workshops*. 162–166.

[10] Jan Peter Ohst. 2016. *On the construction of optimal paths from flows and the analysis of evacuation scenarios*. Ph. D. Dissertation. Universität Koblenz-Landau.

[11] S. Orlowski, R. Wessaly, M. Pioro, and A. Tomaszewski. 2007. SNDlib 1.0: Survivable Network Design Library. In *International Network Optimization Conference (INOC)*.

[12] Stanislas Pedebearn, Slim Abdellatif, Pascal Berthou, Dariusz Nogalski, and Dallal Belabed. 2024. Virtual Link Embedding in Collaborative Sliced Multi-Administrative Multi-Domain Networks. In *39th ACM/SIGAPP Symposium on Applied Computing (SAC)*. New York, NY, USA, 1088–1095.

[13] Sabine Randriamasy, Luis M. Contreras, Jordi Ros-Giralt, and Roland Schott. 2024. *Joint Exposure of Network and Compute Information for Infrastructure-Aware Service Deployment*. Internet-Draft. IETF. https://datatracker.ietf.org/doc/draft-rcr-opsawg-operational-compute-metrics/08/

[14] Farah Aït Salaht, Frédéric Desprez, and Adrien Lebre. 2020. An Overview of Service Placement Problem in Fog and Edge Computing. *ACM Computing Surveys* 53, 3, Article 65 (June 2020), 35 pages.

[15] Fady Samuel, Mosharaf Chowdhury, and Raouf Boutaba. 2013. PolyViNE: policy-based virtual network embedding across multiple domains. *Journal of Internet Services and Applications* 4, 1 (March 2013), 6.

[16] Gang Sun, Yayu Li, Hongfang Yu, Athanasios V. Vasilakos, Xiaojiang Du, and Mohsen Guizani. 2019. Energy-efficient and traffic-aware service function chaining orchestration in multi-domain networks. *Future Generation Computer Systems* 91 (Feb. 2019), 347–360.

[17] Masoud Taghavian, Yassine Hadjadj-Aoul, Géraldine Texier, Nicolas Huin, and Philippe Bertin. 2023. An Approach to Network Service Placement Reconciling Optimality and Scalability. *IEEE Transactions on Network and Service Management* 20, 3 (2023), 2218–2229.

[18] Benedicte Vatinlen, Fabrice Chauvet, Philippe Chrétienne, and Philippe Mahey. 2008. Simple bounds and greedy algorithms for decomposing a flow into a minimal set of paths. *European Journal of Operational Research* 185, 3 (2008), 1390–1401.

[19] Christoph Werle, Panagiotis Papadimitriou, Ines Houidi, Wajdi Louati, Djamal Zeghlache, Roland Bless, and Laurent Mathy. 2011. Building virtual networks across multiple domains. In *ACM SIGCOMM conference*. 412–413.

[20] Fida-E Zaheer, Jin Xiao, and Raouf Boutaba. 2010. Multi-provider service negotiation and contracting in network virtualization. In *IEEE Network Operations and Management Symposium (NOMS)*. 471–478.

[21] Chuangchuang Zhang, Xingwei Wang, Anwei Dong, Yong Zhao, Qiang He, and Min Huang. 2020. Energy efficient network service deployment across multiple SDN domains. *Computer Communications* 151 (2020), 449–462.