# Broker-Coordinated Conflict-Based Search for Multi-Domain Virtual Network Embedding

Yi Zheng*, Erik Kline*, Sven Koenig†, and T. K. Satish Kumar*
*University of Southern California, Los Angeles, California, USA
†University of California, Irvine, Irvine, California, USA
Email: yzheng63@usc.edu, kline@isi.edu, sven.koenig@uci.edu, tkskwork@gmail.com

***Abstract*—Network virtualization enables multiple virtual networks to share physical network resources efficiently and flexibly. Realizing this capability requires solving the Virtual Network Embedding (VNE) problem, the task of assigning Virtual Network Requests (VNRs) to physical network resources while satisfying resource allocation constraints. Although VNE has been extensively studied in single-domain settings, practical deployments may span multiple administrative domains operated by independent Infrastructure Providers (InPs) that do not disclose their internal topologies or resource availability. In this paper, we study the multi-domain VNE problem and propose B-iVNE-CBS, a broker-coordinated framework that uses iVNE-CBS, an improved Conflict-Based Search (CBS) solver for VNE, to compute local embeddings within each InP. The broker assigns VNR vertices to suitable InPs based on InP-reported embedding costs and an abstract view of the inter-domain connectivity. Each InP computes its local embedding using iVNE-CBS and reports only the information required for coordination. When a local embedding fails or the resulting local embeddings cannot be combined into a feasible inter-domain embedding, the broker introduces constraints and revises the affected assignments through a CBS-style search. This design preserves the autonomy of each InP while limiting the disclosure of private network information. Experimental results show that, despite limited information disclosure, B-iVNE-CBS achieves the highest acceptance ratios for large multi-domain VNE instances, favorable cost/revenue ratios, and low runtimes compared with the evaluated baseline methods.**

***Index Terms*—network virtualization, multi-domain virtual network embedding, broker coordination, conflict-based search**

## I. Introduction

It is difficult to make large-scale architectural changes to the Internet, a limitation commonly referred to as Internet ossification. Network virtualization has been proposed as a means of overcoming this limitation by enabling shared physical network resources to support multiple customized virtual networks [1]. By decoupling network services from the underlying infrastructure, network virtualization allows Service Providers (SPs) to lease resources from Infrastructure Providers (InPs), thereby reducing capital and operational costs through resource sharing [2]. SPs can use these virtualized resources to offer network services, including Virtual Network Functions (VNFs) and network slices, with diverse Quality of Service (QoS) guarantees. This capability is particularly relevant to 5G network slicing and Network Slice as a Service (NSaaS), as specified by the 3rd Generation Partnership Project (3GPP) [3].

The physical resources on a network can be of many different types: processors, communication links, and storage devices, among others. While some of these resources are localized to individual network nodes, some other resources, such as communication bandwidth between network nodes, are spread across multiple network links. Network virtualization presents these physical resources as "logical" resources that can be allocated to virtual networks. However, doing so introduces an orchestration problem: The physical resources have to be managed efficiently and effectively in a shared environment that may have many users and many kinds of constraints. In essence, the physical resources of a network have to be properly allocated to satisfy requests for resources while meeting various kinds of constraints, ensuring the necessary QoS, and maximizing the overall resource utilization.

This resource-allocation task is commonly referred to as the Virtual Network Embedding (VNE) problem. The VNE problem allocates resources from a physical Substrate Network (SN) to satisfy the requirements of a Virtual Network Request (VNR) [4]. A VNR is represented by a graph with vertices representing logical compute nodes and edges representing logical communication links between pairs of logical compute nodes. A successful embedding[1] of a VNR onto an SN maps each VNR vertex to a physical compute node and each VNR edge to a physical communication path while allocating the required CPU and bandwidth resources and satisfying the corresponding capacity constraints.

In general, the VNE problem models a broad class of resource-allocation tasks in computer systems and networks. A customer may request a network slice to support 5G Ultra-Reliable Low-Latency Communications (URLLC), specifying a particular Radio Access Network (RAN) as the slice ingress and a particular Point of Presence (PoP) as the slice egress [5]–[7]. The network administrator must construct the slice across the physical network, satisfy its QoS requirements, and manage substrate resources so that existing slices remain supported while new requests are admitted.

In the conventional single-domain VNE setting, an SP submits a VNR to an InP, which embeds the request onto the SN that it operates. The InP has full knowledge of the SN topology and available resources. In practice, however, virtual networks

[1]The terms "embedding" and "mapping" are used interchangeably throughout this paper.

often need to be provisioned across heterogeneous administrative domains operated by multiple independent InPs [8], [9]. This gives rise to the multi-domain VNE problem, in which the overall SN is composed of multiple domains (subnetworks), each operated by an independent InP. Each InP preserves local autonomy over its domain and typically does not fully disclose its network topology or available resources. Therefore, single-domain VNE algorithms cannot be applied directly to the multi-domain setting because no participant necessarily has complete knowledge of the overall network. Instead, the multi-domain VNE problem requires coordination among independent InPs under limited information disclosure.

In this paper, we build on VNE-CBS [10], a Conflict-Based Search (CBS) solver for the single-domain VNE problem. VNE-CBS interprets the VNE problem as a constrained path-coordination problem, first proposed in [4], and exploits this interpretation by adapting CBS from the Multi-Agent Path Finding (MAPF) domain [11]. Its improved version, iVNE-CBS [12], further incorporates advanced MAPF search techniques, such as disjoint splitting and conflict avoidance tables. In particular, iVNE-CBS has been empirically shown to outperform widely used VNE algorithms and to scale to networks with several hundred vertices and thousands of edges, substantially larger than the instances commonly considered in prior VNE studies [13]. However, both VNE-CBS and iVNE-CBS are designed for the single-domain VNE setting.

To address this limitation, we propose Broker-coordinated iVNE-CBS (B-iVNE-CBS), a framework that extends CBS-style conflict resolution to multi-domain VNE under limited information disclosure. B-iVNE-CBS introduces a broker[2] between the SP and the InPs to coordinate the multi-domain embedding process. The broker performs a CBS-style search over assignments of VNR vertices to InPs and the inter-domain connections induced by these assignments.

The broker's search begins with a cost-based assignment of VNR vertices to InPs, constructed from InP-reported embedding costs and the broker's abstract view of inter-domain connectivity. Each involved InP then uses iVNE-CBS to compute the local embedding induced by this assignment. Using the resulting local embeddings and disclosed boundary information, the broker checks whether the local embeddings can be connected to form a feasible global multi-domain embedding. When a local embedding fails or the local embeddings cannot be connected feasibly, the broker branches its search by adding a constraint that excludes the conflicting vertex-to-domain assignment, border choice, or inter-domain SN edge. It then recomputes only the affected assignment decisions and local embeddings. This framework enables iVNE-CBS, originally designed for single-domain VNE, to be used as a local solver in the multi-domain setting while preserving limited information disclosure.

Our main contribution is the B-iVNE-CBS solver, a broker-coordinated solver for the multi-domain VNE problem that employs a broker-level CBS-style search. Experimental results show that, despite limited information disclosure, B-iVNE-CBS achieves the highest acceptance ratios for large multi-domain VNE instances, favorable cost/revenue ratios, and low runtimes compared with the evaluated baseline methods. Moreover, building on our previous work that adapted the CBS framework to the single-domain VNE problem [13], these results demonstrate that CBS-style search can also be effective in the multi-domain VNE setting.

The remainder of this paper is organized as follows. Section II reviews related work, Section III defines the multi-domain VNE problem, Section IV presents B-iVNE-CBS, Section V reports the experimental evaluation, and Section VI concludes the paper.

## II. Background

In this section, we review literature on the single-domain and multi-domain VNE problems.

### A. Single-Domain Virtual Network Embedding

The single-domain VNE problem, also known as the intra-domain VNE problem, is a constrained resource allocation problem that maps VNR vertices to SN vertices and VNR edges to SN paths. The vertex mapping allocates compute resources, such as CPU, while the edge mapping allocates communication resources, such as bandwidth. A feasible VNE mapping must satisfy the CPU and bandwidth capacity constraints. Additional constraints, such as geographical location restrictions for VNR vertices, may also be imposed. Another commonly imposed constraint requires distinct VNR vertices from the same VNR to be mapped to distinct SN vertices [4], [15]. The VNE problem is NP-hard [16] and can be interpreted as a constrained path-coordination problem [4]. In the VNE literature, algorithms are commonly evaluated in an online setting, where VNRs arrive sequentially over time and stay in the network for finite lifetimes. Each accepted VNR holds its allocated SN resources until its departure, after which the resources are released.

Many algorithms have been proposed for solving the VNE problem and its variants [17], [18]. Optimization-based approaches typically formulate the VNE problem as a Mixed-Integer Program (MIP) or an Integer Linear Program (ILP). Examples include D-ViNE and R-ViNE, which use deterministic and randomized rounding, respectively, to derive integer solutions from a relaxed MIP formulation [4]. Other approaches greedily map VNR vertices to SN vertices and then map VNR edges to SN paths using shortest-path or Multi-Commodity Flow (MCF) computations, as in G-SP [19] and G-MCF [16], respectively. RW-MaxMatch-SP [20] uses a Markov Random Walk model, inspired by Google's PageRank, to rank VNR and SN vertices based on resource and topological attributes before first mapping VNR vertices to SN vertices and then mapping VNR edges to SN paths. More recently, inspired by the success of CBS in the MAPF domain [11], VNE-CBS [10] and its improved version, iVNE-CBS [12], [13], have been developed to solve the VNE problem using a two-level search framework.

[2]This broker is also referred to as a Virtual Network Provider (VNP) in the literature [14]. We prefer to use the term "broker" throughout this paper.

The high-level search resolves resource-allocation conflicts by branching and imposing constraints on the resulting high-level search nodes, while the low-level search computes vertex and edge mappings that satisfy these constraints.

### B. Multi-Domain Virtual Network Embedding

In the single-domain VNE setting, a single InP controls the entire SN and has full knowledge of its topology and available resources. However, in the multi-domain VNE problem, also known as the inter-domain VNE problem, an SP submits a VNR that may need to be provisioned across several domains of a multi-domain SN operated by independent InPs [8], [9], [21]. Each InP may embed part or all of the VNR according to its internal administrative policies while maintaining global connectivity through agreements with other InPs. Moreover, InPs typically do not fully disclose their SN topology, resource availability, or administrative policies to other InPs or coordinating entities. Therefore, single-domain VNE algorithms that assume full knowledge of the entire SN cannot be applied directly to the multi-domain VNE problem.

A common category of approaches to multi-domain VNE introduces a broker between the SP and the InPs [9], [21]–[23]. The broker collects limited information disclosed by the InPs, partitions the VNR across multiple domains, and coordinates the embedding process. Existing broker-based methods differ mainly in how they partition the VNR and coordinate the inter-domain embedding. The approach in [21] uses maxflow/mincut and MIP techniques; [22] uses traffic matrices rather than explicit VNR topologies; [23] uses an augmented network and Linear Programming (LP) relaxation; and [9] further considers the interactions between intra-domain and inter-domain edge mappings.

Another category comprises decentralized approaches that avoid a centralized broker. PolyViNE [8] is a policy-based distributed framework in which InPs recursively forward embedding requests and use bidding to obtain competitive costs without relying on a centralized broker. In the Consensus-based Auction mechanism for Distributed virtual network embedding (CAD) [24], each SN vertex acts as a distributed agent that participates in auction-based consensus to map the VNR vertices. VNR edges are then embedded separately using shortest-path algorithms.

Unlike prior broker-based approaches that primarily rely on optimization or heuristics to first compute a partition of the VNR and then attempt to solve the resulting subproblems, B-iVNE-CBS performs a systematic CBS-style search over the vertex-to-domain assignments and inter-domain connection decisions. When a local embedding fails or an inter-domain conflict is detected, the broker adds constraints that exclude the conflicting decision and explores alternative assignments.

## III. Problem Definition

In this section, we describe the multi-domain VNE problem considered in this paper, following the standard single-domain VNE formulation [4] and the prior multi-domain VNE formulations [8], [9].

### A. Substrate Network

The SN is composed of $d$ domains, each operated by an independent InP. The domain controlled by $\mathit{InP}_i$ ($1 \le i \le d$) is an undirected graph $G_i^s = (V_i^s, E_i^s)$, where $V_i^s$ is the set of SN vertices in the $i$-th domain and $E_i^s$ is the set of intra-domain SN edges whose endpoints both belong to $V_i^s$. The vertex sets of two different domains are disjoint, that is, $V_i^s \cap V_j^s = \emptyset$, for $i \neq j$. Each $\mathit{InP}_i$ has full knowledge of its own domain $G_i^s$. Moreover, the $i$-th domain has a set of border SN vertices $B_i^s \subseteq V_i^s$ that connect to border SN vertices in other domains through inter-domain SN edges. Let $E_{inter}^s$ denote the set of inter-domain SN edges that connect border SN vertices in different domains.

The overall multi-domain SN is denoted by $G^s = (V^s, E^s)$, where $V^s = \bigcup_{i=1}^{d} V_i^s$ and $E^s = \left(\bigcup_{i=1}^{d} E_i^s\right) \cup E_{inter}^s$. The attributes of an SN vertex $v^s \in V^s$ include its CPU capacity $\textsc{cpu}(v^s)$ and its location $\textsc{loc}(v^s)$. The attribute of an SN edge $e^s \in E^s$ is its bandwidth capacity $\textsc{bw}(e^s)$. An SN path is a path in $G^s$. An SN path whose edges all belong to the same $E_i^s$ is an intra-domain SN path, while an SN path that uses at least one edge in $E_{inter}^s$ is an inter-domain SN path. Fig. 1 shows an example of a multi-domain SN with 3 domains. The border SN vertices are shown as hexagons, with $B_1^s = \{5, 6\}$, $B_2^s = \{8\}$, and $B_3^s = \{11\}$.

### B. Virtual Network Request

A VNR is an undirected graph $G^r = (V^r, E^r)$, where $V^r$ is the set of VNR vertices and $E^r$ is the set of VNR edges. The demands of a VNR vertex $v^r$ include its CPU requirement $\textsc{cpu}(v^r)$, its preferred geographical location $\textsc{loc}(v^r)$, and the maximum allowed distance $D(v^r)$ from its preferred geographical location to the location of the SN vertex that it is mapped to. The demand of a VNR edge $e^r$ is its bandwidth requirement $\textsc{bw}(e^r)$. In the left panel of Fig. 1, the VNR contains 3 VNR vertices, $A$, $B$, and $C$, connected by 3 VNR edges.

### C. Broker and Information Disclosure

The broker coordinates the embedding of the VNR onto the multi-domain SN under limited information disclosure. Instead of exposing the full domain topology, each InP discloses only an abstract view of its domain to the broker. The abstract view contains the border SN vertices of the domain and the inter-domain SN edges incident on them. The broker combines these views into an abstract multi-domain SN $G^a$, which captures inter-domain connectivity while hiding each domain's internal topology. The right panel of Fig. 1 shows an example of an abstract multi-domain SN. In this example, the broker knows only the border SN vertices (hexagons 5, 6, 8, and 11) and the inter-domain SN edges connecting them, while the intra-domain topology of each domain remains hidden.

For each VNR edge $e^r = (v_1^r, v_2^r)$, each InP reports relevant cost information to the broker. The cumulative information provided by the InPs enables the broker to evaluate vertex-to-domain assignments. If $\mathit{InP}_i$ embeds both endpoints, $v_1^r$ and $v_2^r$, and chooses to embed the edge $e^r$ within its domain, it

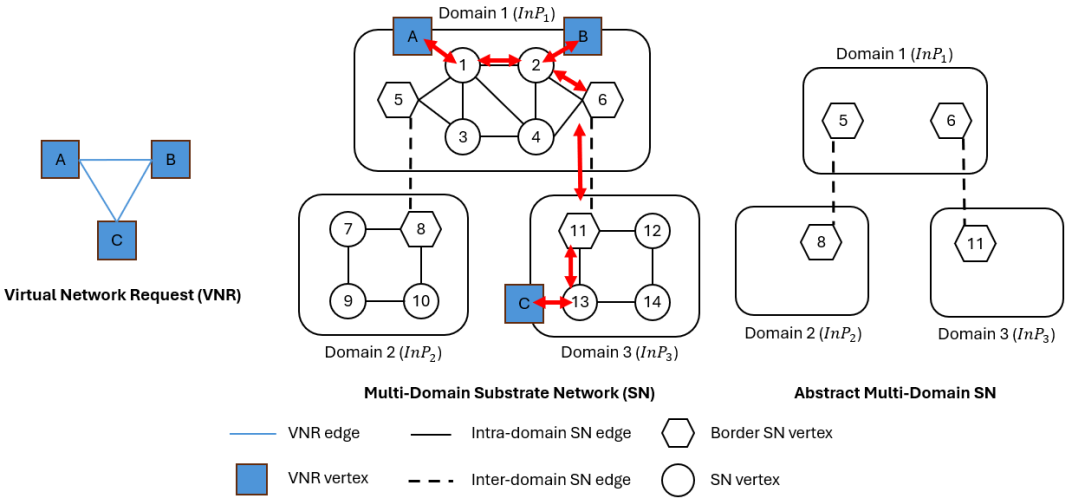


Fig. 1. An example multi-domain VNE instance and its abstract multi-domain SN. Red arrows indicate the multi-domain embedding. The VNR vertices $A$ and $B$ are assigned to domain 1, and the VNR vertex $C$ is assigned to domain 3. The VNR edge $A$–$B$ is mapped to an intra-domain SN path 1–2 in domain 1. The VNR edge $A$–$C$ is mapped to an inter-domain SN path 1–2–6–11–13. The VNR edge $B$–$C$ is mapped to an inter-domain SN path 2–6–11–13.

reports the total cost of doing so. However, if $InP_i$ embeds only one endpoint, it reports a cost vector containing the cost of embedding that endpoint and connecting it to each feasible border SN vertex within its domain. If $InP_i$ embeds neither endpoint but can serve as a transit domain for $e^r$, it reports a cost matrix containing the cost of connecting each feasible ingress border SN vertex to each feasible egress border SN vertex within its domain. The broker combines this InP-reported cost information with the costs of utilizing inter-domain SN edges to evaluate vertex-to-domain assignments and the inter-domain connections induced by them. For example, in Fig. 1, the VNR vertices $A$ and $C$ are assigned to domains 1 and 3, respectively. To evaluate the inter-domain VNR edge $A$–$C$, the broker combines the cost reported by $InP_1$ for connecting $A$ to border SN vertex 6, the cost reported by $InP_3$ for connecting $C$ to border SN vertex 11, and the cost of utilizing the inter-domain SN edge 6–11.

### D. Multi-Domain Virtual Network Embedding

In the multi-domain VNE problem, given a VNR $G^r = (V^r, E^r)$, a multi-domain assignment is a mapping $\alpha : V^r \to \{1, 2 \dots d\}$, where $\alpha(v^r) = i$ indicates that VNR vertex $v^r$ is assigned to the $i$-th domain controlled by $InP_i$. The assignment $\alpha(\cdot)$ determines which VNR vertices are embedded locally by each InP, which VNR edges have both endpoints assigned to the same domain and can therefore be embedded locally, and which VNR edges require inter-domain coordination.

Let $V_i^r = \{v^r \in V^r \mid \alpha(v^r) = i\}$ be the set of VNR vertices that are assigned to the $i$-th domain. $E_{intra,i}^r = \{(v_1^r, v_2^r) \in E^r \mid \alpha(v_1^r) = \alpha(v_2^r) = i\}$ is the set of intra-domain VNR edges that are embedded entirely within the SN $G_i^s$. $InP_i$ computes a local embedding that maps each VNR vertex in $V_i^r$ to an SN vertex in $V_i^s$ and each VNR edge in $E_{intra,i}^r$ to an intra-domain SN path in $G_i^s$.

$E_{inter}^r = \{(v_1^r, v_2^r) \in E^r \mid \alpha(v_1^r) \neq \alpha(v_2^r)\}$ is the set of inter-domain VNR edges that connect VNR vertices assigned to different domains. These VNR edges cannot be embedded entirely inside a single domain. Instead, the broker coordinates their embedding using the abstract multi-domain SN, which contains the disclosed border SN vertices and inter-domain SN edges. After the broker selects an inter-domain SN path to implement each edge in $E_{inter}^r$, let $E_{inter,i}^r \subseteq E_{inter}^r$ denote the set of inter-domain VNR edges whose SN paths originate, terminate, or transit through the $i$-th domain. For each VNR edge $e^r \in E_{inter,i}^r$, $InP_i$ embeds only the segment of $e^r$'s selected SN path that lies within the $i$-th domain. If one of the endpoints of $e^r$ is assigned to the $i$-th domain, $InP_i$ must find an intra-domain SN path from the mapped endpoint to the broker-selected border SN vertex. If $e^r$ only transits through the $i$-th domain, $InP_i$ must find an intra-domain SN path between the two broker-selected border SN vertices.

The VNR elements assigned to $InP_i$ for local embedding form a subproblem $G_i^r = (V_i^r, E_i^r)$, where $V_i^r$ is the set of VNR vertices assigned to $InP_i$ and $E_i^r = E_{intra,i}^r \cup E_{inter,i}^r$. For each edge in $E_{inter,i}^r$, only the segment that lies within the $i$-th domain must be embedded locally. Let $\text{VNE}_i(\cdot)$ denote the local VNE embedding computed by $InP_i$. The embedding $\text{VNE}_i(\cdot)$ maps each VNR vertex in $V_i^r$ to an SN vertex in $V_i^s$, each edge in $E_{intra,i}^r$ to an intra-domain SN path in $G_i^s$, and the relevant intra-domain segment of each edge in $E_{inter,i}^r$ to an intra-domain SN path in $G_i^s$. A feasible $\text{VNE}_i(\cdot)$ must satisfy the following constraints.

For the mapping of VNR vertices to SN vertices, the following constraints must hold:

1) each VNR vertex $v^r \in V_i^r$ is mapped to exactly one SN vertex $\text{VNE}_i(v^r) \in V_i^s$,
2) no two distinct VNR vertices $v_1^r \in V_i^r$ and $v_2^r \in V_i^r$ are mapped to the same SN vertex $v^s \in V_i^s$, and

3) for each VNR vertex $v^r \in V_i^r$:

$$\text{CPU}(v^r) \leq \text{CPU}(\text{VNE}_i(v^r))$$

and

$$\text{GEODIST}(\text{LOC}(v^r), \text{LOC}(\text{VNE}_i(v^r))) \leq D(v^r),$$

where $\text{GEODIST}(\cdot, \cdot)$ is a function that calculates the distance between two geographical locations.

For the mapping of local VNR edge segments to SN paths, the following constraints must hold:

1) the segment of each VNR edge $e^r \in E_i^r$ that lies within the $i$-th domain is mapped to an intra-domain SN path $\text{VNE}_i(e^r)$ in $G_i^s$, and
2) for each SN edge $e^s \in E_i^s$, the sum of the bandwidth requirements of all VNR edge segments that utilize it must not exceed its available bandwidth capacity:

$$\sum\nolimits_{e^r \in E_i^r:\ e^s \in \text{VNE}_i(e^r)} \text{BW}(e^r) \leq \text{BW}(e^s).$$

For each inter-domain VNR edge, the broker must select a sequence of inter-domain SN edges with sufficient bandwidth that is compatible with the local embeddings computed by the participating InPs. An inter-domain conflict occurs when the local embeddings and the broker-selected inter-domain SN paths cannot be combined into a single feasible global embedding. Such a conflict may arise from a vertex-to-domain assignment that cannot support the required connectivity, insufficient inter-domain bandwidth, or incompatible border SN vertex choices.

Therefore, a feasible multi-domain VNE embedding consists of a vertex-to-domain assignment $\alpha(\cdot)$, feasible local embeddings $\text{VNE}_i(\cdot)$ from all participating InPs, and feasible inter-domain SN paths for all inter-domain VNR edges $E_{inter}^r$.

### E. Objectives

Several metrics are commonly used to evaluate VNE algorithms, including revenue, embedding cost, cost/revenue ratio, and runtime [17]. The revenue of a successfully embedded VNR is defined as the sum of its requested CPU and bandwidth resources. If the VNR cannot be embedded, its revenue is 0. The embedding cost is the total amount of SN resources allocated to embed the VNR across all participating domains. It includes the CPU resources allocated on the SN vertices for the VNR vertices and the bandwidth resources allocated on the intra-domain and inter-domain SN edges for the VNR edges. Since every VNR edge is mapped to an SN path containing at least one SN edge, the embedding cost is at least as large as the revenue for any successfully embedded VNR. Another commonly used metric is the cost/revenue ratio, defined as the embedding cost divided by the revenue, which measures how efficiently the SN resources are allocated to a VNR. In this paper, the objective is to find a feasible multi-domain VNE mapping with a low embedding cost. In the online setting, VNE algorithms are also commonly evaluated using the acceptance ratio.

## IV. B-iVNE-CBS

In this section, we introduce B-iVNE-CBS, a broker-coordinated framework for solving the multi-domain VNE problem under limited information disclosure. It uses iVNE-CBS as the local embedding solver for each InP and introduces a broker that employs a CBS-style search to coordinate inter-domain embedding decisions and refine vertex-to-domain assignments. The following subsections first briefly describe VNE-CBS and its improved version, iVNE-CBS, then define the broker-level conflicts resolved by the broker, and finally present the B-iVNE-CBS algorithm.

### A. VNE-CBS and iVNE-CBS

VNE-CBS [10] adapts CBS, originally developed for the MAPF problem, to the single-domain VNE problem. It reformulates the single-domain VNE problem as a constrained path-coordination problem by constructing an augmented SN graph that combines the original SN with fictitious vertices representing the VNR vertices. Each fictitious vertex is connected to the SN vertices that satisfy its CPU and geographical location constraints via fictitious edges. Therefore, the fictitious edges encode the possible vertex mappings. Under this construction, a VNE mapping becomes a set of paths, each of which represents both the mapping of a VNR edge to an SN path and the mappings of its endpoint VNR vertices to SN vertices. For example, for the VNR edge $A$–$B$ in Fig. 1, the path from the fictitious vertex $A$ to the fictitious vertex $B$ through the SN vertices 1 and 2 simultaneously maps $A$ to the SN vertex 1, $B$ to the SN vertex 2, and $A$–$B$ to the SN path 1–2.

VNE-CBS uses a high-level search and a low-level search. The high-level search is a best-first search on a Constraint Tree (CT) that resolves conflicts caused by resource contentions. VNE-CBS defines several conflict types that capture invalid or over-capacity mappings, including the type-1 vertex conflict (a VNR vertex is mapped to two different SN vertices), the type-2 vertex conflict (two VNR vertices are mapped to the same SN vertex), and the bandwidth capacity conflict (an SN edge does not have sufficient bandwidth capacity to accommodate all VNR edges that utilize it).[3] A conflict is resolved in the high-level search by branching and creating child CT nodes with alternative constraints. Each CT node in the high-level search stores all the constraints accumulated via branching. The low-level search must respect the constraints imposed by a CT node. It finds shortest paths in the augmented SN graph that encode both the vertex and edge mappings. iVNE-CBS [12], [13] is an improved version of VNE-CBS that incorporates additional CBS enhancements, such as disjoint splitting, bypassing conflicts, and conflict avoidance tables.

iVNE-CBS for the single-domain VNE problem plays an important role in our proposed solver, B-iVNE-CBS, for the multi-domain VNE problem. In particular, in B-iVNE-CBS, each InP receives a local subproblem induced by the current

[3]The CPU constraints and the geographical constraints on the VNR vertices are directly enforced during the construction of the augmented SN.

vertex-to-domain assignments of the broker, including the endpoint-to-border demands associated with its inter-domain VNR edges. This subproblem qualifies as a single-domain VNE problem on the InP's private SN and is solved by iVNE-CBS. After invoking iVNE-CBS, the InP returns either a feasible local embedding with cost information or an indication that the local subproblem is infeasible.

### *B. Broker-Level Conflicts*

Compared to iVNE-CBS, B-iVNE-CBS must resolve three additional types of conflicts for coordination. These broker-level conflicts are incompatible border option conflicts, inter-domain bandwidth conflicts, and domain-assignment conflicts.

An incompatible border option conflict occurs when the endpoint InPs of an inter-domain VNR edge $e^r = (v_1^r, v_2^r)$ return two endpoint-to-border options that cannot be connected through the abstract multi-domain SN. Let $v_1^s$ and $v_2^s$ denote the selected border SN vertices returned by $InP_i$ for $v_1^r$ and by $InP_j$ for $v_2^r$, respectively. To resolve the conflict, the broker branches its search by creating two child broker CT nodes, one with the constraint $\overline{(e^r, v_1^s)}$ and the other with the constraint $\overline{(e^r, v_2^s)}$. Each constraint forbids the corresponding InP from reusing its selected option for $e^r$. In each child broker CT node, only the constrained InP reruns iVNE-CBS, while the other InP reuses its local embedding.

An inter-domain bandwidth conflict occurs when the total bandwidth requirement of the inter-domain VNR edges that utilize an inter-domain SN edge exceeds its available bandwidth capacity. For an inter-domain SN edge $e^s \in E_{inter}^s$, let $Q(e^s) = \{e^r \in E_{inter}^r \mid e^s \in \text{VNE}(e^r)\}$ denote the set of inter-domain VNR edges that utilize $e^s$.[4] A conflict occurs when $\sum_{e^r \in Q(e^s)} \text{BW}(e^r) > \text{BW}(e^s)$. To resolve this conflict, the broker branches its search by creating one child broker CT node for each $e^r \in Q(e^s)$, with the constraint $\overline{(e^r, e^s)}$ forbidding $e^r$ from utilizing $e^s$.

A domain-assignment constraint is of the form $\overline{(v^r, InP_i)}$. It forbids the assignment of $v^r$ to the $i$-th domain. It resolves a domain-assignment conflict that arises when there is no way to embed a VNR edge $e^r = (v_1^r, v_2^r)$ with $\alpha(v_1^r) = i$ and $\alpha(v_2^r) = j$ under the current broker-level constraints. There are two possible cases. If $i = j$ and there is no way to embed $e^r$ locally using iVNE-CBS, the broker branches its search by creating two child broker CT nodes, one with the constraint $\overline{(v_1^r, InP_i)}$ and the other with the constraint $\overline{(v_2^r, InP_i)}$. If $i \neq j$, the broker branches similarly with the constraints $\overline{(v_1^r, InP_i)}$ and $\overline{(v_2^r, InP_j)}$.

### *C. The B-iVNE-CBS Framework*

The B-iVNE-CBS framework consists of a broker and a set of InPs. Each InP has full knowledge of its own SN $G_i^s$, while the broker has access only to the VNR $G^r$, the abstract multi-domain SN $G^a$, and the cost and boundary information disclosed by the participating InPs. Algorithm 1 presents the pseudocode of B-iVNE-CBS. The broker maintains a priority queue *OPEN* of broker CT nodes. Each broker CT node $N$ stores a vertex-to-domain assignment $N.\alpha$, a set of accumulated broker constraints $N.$*constraints*, the current local and inter-domain embeddings $N.$*embedding*, its embedding cost $N.$*cost*, and the number of broker-level conflicts $N.$*num_conf*.

[4] $\text{VNE}(\cdot)$ denotes the complete multi-domain embedding obtained by combining the local embeddings $\text{VNE}_i(\cdot)$ with the inter-domain SN edges selected by the broker.

**Algorithm 1** B-iVNE-CBS

1: **Input:** VNR $G^r = (V^r, E^r)$, abstract multi-domain SN $G^a$, and the InPs $\{InP_1, InP_2 \ldots InP_d\}$
2: $N_R \leftarrow$ an empty broker CT node
3: $N_R.$*constraints* $\leftarrow \emptyset$
4: $N_R.\alpha \leftarrow$ initial assignment of VNR vertices to InPs based on InP-reported cost information and $G^a$
5: **if** there exists $v^r \in V^r$ with no vertex-feasible InP **then**
6: **return** "No Solution"
7: $N_R.$*embedding* $\leftarrow \emptyset$
8: **for** each $InP_i$ involved in the embedding induced by $N_R.\alpha$ **do**
9: $M_i \leftarrow$ local embedding computed by $InP_i$ using iVNE-CBS, given $N_R.\alpha$ and $N_R.$*constraints*
10: Register $M_i$ or a local failure in $N_R.$*embedding*
11: $N_R.$*embedding* $\leftarrow$ connect the local embeddings
12: $N_R.$*cost* $\leftarrow$ cost of $N_R.$*embedding*
13: $N_R.$*num_conf* $\leftarrow$ the number of broker-level conflicts in $N_R.$*embedding*
14: *OPEN* $\leftarrow \{N_R\}$
15: **while** *OPEN* $\neq \emptyset$ **do**
16: $N_T \leftarrow$ top broker CT node in *OPEN*
17: Remove $N_T$ from *OPEN*
18: **if** $N_T.$*num_conf* $= 0$ **then**
19: **return** $N_T.$*embedding* as the solution
20: *Conf* $\leftarrow$ first conflict in $N_T.$*embedding*
21: *Cons* $\leftarrow$ constraints for resolving *Conf*
22: **for** each $C \in$ *Cons* **do**
23: $N_C \leftarrow$ a copy of $N_T$
24: $N_C.$*constraints* $\leftarrow N_C.$*constraints* $\cup \{C\}$
25: Update affected elements of $N_C.\alpha$ and $N_C.$*embedding* and record any local failures
26: **if** the update succeeds **then**
27: $N_C.$*cost* $\leftarrow$ cost of $N_C.$*embedding*
28: $N_C.$*num_conf* $\leftarrow$ number of broker-level conflicts in $N_C.$*embedding*
29: Insert $N_C$ into *OPEN*
30: **return** "No Solution"

Lines 2–14 construct the root CT node $N_R$. On Line 4, the broker greedily generates an initial vertex-to-domain assignment using the InP-reported cost information and $G^a$. The broker greedily assigns the VNR vertices in non-increasing order of their total resource demand, defined as their CPU requirement plus the bandwidth requirements of their incident VNR edges. Assigning the most resource-demanding vertices first preserves more placement options for them and reduces the risk that they become difficult to accommodate later.

An InP is vertex-feasible for a VNR vertex $v^r$ if there exists at least one SN vertex satisfying the CPU and geographical constraints of $v^r$. The broker assigns each VNR vertex $v^r$ to the vertex-feasible InP with the smallest total estimated cost of connecting it to its already assigned neighbors. For each VNR edge $(v_1^r, v_2^r)$ connecting $v_1^r$ to an already assigned neighbor $v_2^r$, the broker obtains the estimated connection cost from the cost information reported by the candidate and neighboring InPs, together with $G^a$.[5] If two or more InPs have the same total estimated cost for accommodating a VNR vertex, the broker prefers the InP with fewer already assigned VNR vertices to balance the workload. Any conflicts resulting from the initial assignment are detected and resolved later by the broker-level search. Therefore, the initial assignment fails only when some VNR vertex has no vertex-feasible InP, in which case the broker returns "No Solution" on Lines 5–6.

On Lines 8–10, each InP involved in the embedding induced by $N_R.\alpha$ constructs and solves its local subproblem using iVNE-CBS. The local subproblem includes the assigned VNR vertices, the VNR edges embedded entirely within the domain, and the relevant intra-domain segments of inter-domain VNR edges. If successful, the InP returns a complete local embedding containing one selected endpoint-to-border option for each incident inter-domain VNR edge and, when applicable, one selected ingress-to-egress border option for each transiting inter-domain VNR edge. If iVNE-CBS fails, the failed local edge or endpoint-to-border segment is recorded as a domain-assignment conflict.

On Lines 11–13, if all required local embeddings are available, the broker connects them to construct a complete candidate embedding. For each inter-domain VNR edge, the broker attempts to connect the two selected endpoint-to-border options returned by its endpoint InPs through the abstract multi-domain SN. The resulting path consists of the endpoint-to-border segments in the two endpoint domains, the selected inter-domain SN edges, and any required border-to-border segments in intermediate domains. If the two selected options cannot be connected while also satisfying $N_R.\textit{constraints}$, the broker records an incompatible border option conflict. When constructing an inter-domain SN path, the broker considers only inter-domain SN edges whose available bandwidth is sufficient for that VNR edge and selects the lowest-cost path between the two selected border SN vertices. After all inter-domain paths are selected, the broker checks their combined bandwidth usage for inter-domain bandwidth conflicts. The cost of a broker CT node is the actual cost of its embedding.

The root broker CT node is inserted into *OPEN* on Line 14. The broker CT nodes are ordered lexicographically by embedding cost and then by the number of broker-level conflicts. In each iteration of Lines 15–29, the broker removes the top broker CT node $N_T$ from *OPEN* according to this ordering. If $N_T$ contains no broker-level conflict, its complete embedding is returned on Lines 18–19 as a solution. Otherwise, Lines 20–21 check $N_T.\textit{embedding}$ for conflicts, select the first conflict, and generate its resolving constraints as described in Section IV-B. For each resolving constraint, Lines 22–29 create a child broker CT node and update only the affected assignment, local embedding, or routing decision. To resolve an incompatible border option conflict, only the constrained endpoint InP reruns iVNE-CBS to select another endpoint-to-border option, while the other endpoint InP's local embedding is reused. To resolve an inter-domain bandwidth conflict, the vertex-to-domain assignment and local embeddings are preserved, and only the affected inter-domain path is recomputed. To resolve a domain-assignment conflict, the broker reassigns the constrained VNR vertex and reruns iVNE-CBS only for the affected InPs. All other unaffected local embeddings are reused. After updating the child broker CT node's embedding cost and number of broker-level conflicts, the broker inserts it into *OPEN*. The algorithm returns "No Solution" on Line 30 if *OPEN* becomes empty.

Overall, B-iVNE-CBS coordinates private local embeddings through a broker-level CBS-style search while requiring each InP to disclose only limited information. It decomposes the multi-domain VNE problem into smaller local subproblems, systematically revises conflicting assignment, border-option, and routing decisions through constraint branching, and reuses unaffected local embeddings during conflict resolution. However, because the broker makes greedy assignment and reassignment decisions using estimated costs derived from limited cost information, B-iVNE-CBS does not guarantee completeness or global optimality.

## V. Experimental Evaluation

In this section, we empirically evaluate B-iVNE-CBS in the online setting of the multi-domain VNE problem commonly considered in the existing literature. We implemented B-iVNE-CBS and the baseline algorithms in C++. All experiments were run on an AWS instance with 8 CPUs and 16 GB of RAM. We set a timeout of 60 seconds for embedding each VNR.

We compare B-iVNE-CBS with the multi-domain baseline solver TF [23] and two centralized VNE baseline solvers, iVNE-CBS [12] and RW-MaxMatch-SP [20]. TF is the "Traffic-Fraction" variant of the broker-based multi-domain VNE algorithm proposed in [23]. It uses an LP relaxation to derive traffic-fraction scores for mapping the VNR vertices and then maps the VNR edges to SN paths with sufficient residual bandwidth. The VNR is accepted only if all participating InPs successfully embed their assigned local subproblems. We implemented TF and adapted it to the problem definition in Section III. Thus, TF and B-iVNE-CBS both operate under limited information disclosure, but TF relies on LP relaxation and greedy rounding rather than a broker-level CBS-style search. The centralized iVNE-CBS solver, referred to as "iVNE-CBS" in this section, provides a full-information CBS reference, while RW-MaxMatch-SP provides a full-information heuristic reference based on a Markov Random Walk ranking of the VNR vertices and shortest-path mapping of the VNR edges.

[5] If no estimated connection is available, the broker uses a large finite estimate for that edge. The candidate InP for accommodating $v_1^r$ remains eligible and may still be selected if its resulting total estimated cost is the smallest among all possible candidates.

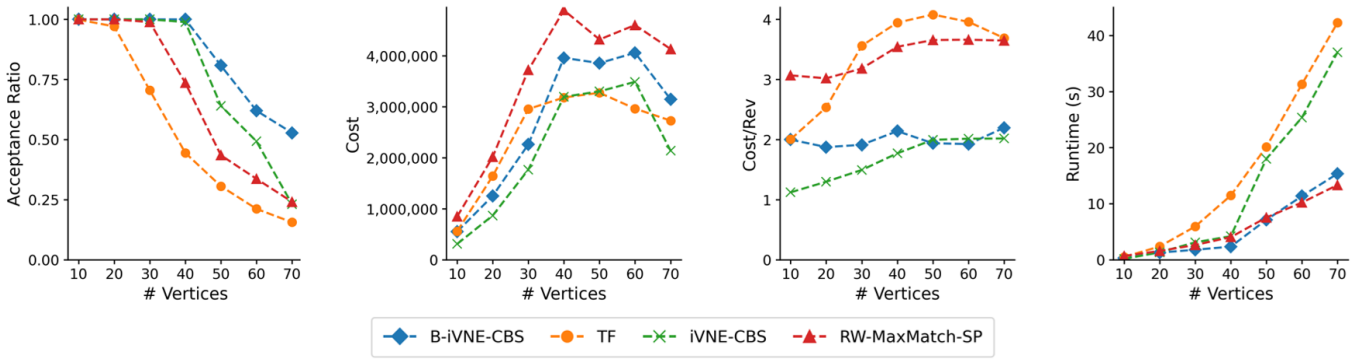


Fig. 2. Experimental results comparing B-iVNE-CBS with TF, iVNE-CBS, and RW-MaxMatch-SP in the online setting of the multi-domain VNE problem.

Because these centralized baselines have access to strictly more information about the SN, they provide strong reference points for evaluating B-iVNE-CBS under limited information disclosure. We choose $w = 2.0$ as the suboptimality factor for both the local iVNE-CBS solvers used by B-iVNE-CBS and the baseline iVNE-CBS solver.

We use Waxman graphs to generate the SN topologies in our VNE problem instances. Waxman graphs [25] are commonly used in the VNE literature to simulate communication networks. The SN topologies are generated as random Waxman graphs with parameters $\alpha = 0.1$ and $\beta = 0.3$. To simulate the multi-domain environment, we generate five SNs, each with 100 vertices and approximately 800 edges, corresponding to five InPs. Approximately 30 vertices in each domain are randomly selected as border SN vertices. For each pair of border SN vertices belonging to different domains, we independently add an inter-domain SN edge with probability 0.1. The CPU and bandwidth capacities of SN vertices and edges are real numbers drawn uniformly at random from $[50, 100]$. For the centralized baselines, the complete multi-domain SN is provided as a single SN with 500 vertices, including all intra-domain and inter-domain SN edges.

We also use the Waxman model with $\alpha = 0.2$ and $\beta = 0.3$ to generate VNR topologies. The number of VNR vertices is varied from 10 to 70 in increments of 10. We generate $1,000$ VNRs for each setting. The VNR vertices are placed in the same $100 \times 100$ grid as the SN vertices. The maximum allowed Euclidean distance for the geographical constraint on each VNR vertex is set to 10. The CPU requirements of VNR vertices and the bandwidth requirements of VNR edges are real numbers drawn uniformly at random from $[0, 20]$ and $[0, 50]$, respectively.

In the online setting, VNRs arrive dynamically over time, and each accepted VNR holds its allocated SN resources until the end of its lifetime. VNR arrivals follow a Poisson process with an average rate of 4 VNRs per 100 timesteps, while VNR lifetimes follow an exponential distribution with a mean of $1,000$ timesteps. Previously embedded VNRs cannot be reconfigured when a new VNR arrives. If an algorithm cannot embed a VNR within 60 seconds, the VNR is rejected and the algorithm proceeds to the next request. We generate three independent multi-domain SN instances and, for each VNR size, embed the same set of $1,000$ VNRs on each SN. We report the average performance across the three runs.

Figure 2 compares the algorithms using four metrics. "Acceptance Ratio" is the fraction of VNRs that are successfully embedded, averaged over the three runs. "Cost" is the total embedding cost of all successfully embedded VNRs, averaged over the three runs. "Cost/Rev" is the average cost/revenue ratio of all successfully embedded VNRs across the three runs. "Runtime" is the average runtime over all successfully embedded VNRs across the three runs.

All algorithms achieve high acceptance ratios for small VNRs, but their acceptance ratios decrease as the VNR size increases. B-iVNE-CBS maintains a higher acceptance ratio over a wider range of VNR sizes and performs best for the largest VNRs. Compared with the broker-based baseline TF, B-iVNE-CBS achieves a higher acceptance ratio, suggesting that systematically exploring alternative vertex-to-domain assignments and revising conflicting decisions improves its ability to find feasible embeddings. B-iVNE-CBS also outperforms the centralized baselines for larger VNRs. One possible explanation is that B-iVNE-CBS decomposes each VNR into smaller local subproblems that are solved independently by the participating InPs, rather than embedding the entire VNR directly on the complete 500-vertex SN, which may make feasible embeddings easier to find within the timeout.

The total embedding cost generally increases with VNR size because each accepted request requires more SN resources. However, this metric also depends on how many VNRs an algorithm accepts, so a higher total cost may partly reflect a higher acceptance ratio rather than lower resource efficiency. Cost/Rev therefore provides a more direct comparison of resource efficiency. B-iVNE-CBS maintains a relatively stable Cost/Rev ratio because its broker-level search is guided by embedding cost, while each InP uses iVNE-CBS to construct low-cost local embeddings. Its ratio is substantially lower than those of TF and RW-MaxMatch-SP and remains close to that of centralized iVNE-CBS for larger VNRs. These results suggest that B-iVNE-CBS preserves much of the resource efficiency of the full-information iVNE-CBS solver despite operating under limited information disclosure.

The average runtime of all algorithms increases with VNR size. TF becomes the slowest method for large VNRs, which is consistent with the increasing size of its LP relaxation, while centralized iVNE-CBS becomes increasingly expensive as it searches over the complete 500-vertex SN. B-iVNE-CBS remains substantially faster than both methods for large VNRs, although RW-MaxMatch-SP is slightly faster because it uses a greedy procedure. These observations are consistent with B-iVNE-CBS decomposing the problem into smaller local subproblems and reusing unaffected local embeddings during conflict resolution. Overall, B-iVNE-CBS achieves the highest acceptance ratio for large VNRs while maintaining favorable resource efficiency and low runtime with limited information.

## VI. Conclusions and Future Work

In this paper, we studied multi-domain VNE under limited information disclosure and proposed B-iVNE-CBS, a broker-coordinated framework that uses iVNE-CBS as the local embedding solver for each InP. The broker operates on the abstract multi-domain SN and InP-reported costs and performs a CBS-style search to revise VNR assignments, endpoint-to-border choices, and inter-domain SN paths when local embedding failures or inter-domain conflicts occur.

Experimental results in the online VNE setting show that B-iVNE-CBS achieves a favorable balance among acceptance ratio, resource efficiency, runtime, and information disclosure. Compared with the broker-based multi-domain baseline TF and the centralized baselines iVNE-CBS and RW-MaxMatch-SP, B-iVNE-CBS achieves the highest acceptance ratios for large VNRs, favorable Cost/Rev ratios, and low runtimes while preserving the autonomy and privacy of individual InPs.

Future work will evaluate B-iVNE-CBS on real-world or more realistic multi-domain network topologies and workloads and investigate its performance under practical conditions such as heterogeneous domains, communication overhead, and incomplete or stale InP-reported information.

## VII. Acknowledgment

This research was supported by DARPA under grant number HR001120C0157 and by NSF under grant numbers 1817189, 1935712, 2544613, 2321786, and 2112533. The views, opinions, and/or findings expressed are those of the authors and should not be interpreted as representing the official views or policies of the sponsoring organizations, agencies, or the U.S. government.

## References

[1] N. M. M. Chowdhury and R. Boutaba, "Network Virtualization: State of the Art and Research Challenges," *IEEE Communications Magazine*, 2009.

[2] N. Feamster, L. Gao, and J. Rexford, "How to Lease the Internet in Your Spare Time," *Computer Communication Review*, 2007.

[3] The European Telecommunications Standards Institute. (2020) 5G; Management and Orchestration; Concepts, Use Cases and Requirements. [Online]. Available: https://www.etsi.org/deliver/etsi_ts/128500_128599/128530/16.02.00_60/ts_128530v160200p.pdf

[4] M. Chowdhury, M. R. Rahman, and R. Boutaba, "Virtual Network Embedding with Coordinated Node and Link Mapping," in *Proceedings of the Twenty-Eighth Joint Conference of the IEEE Computer and Communications Societies*, 2009.

[5] M. Richart, J. Baliosian, J. Serrat, and J. Gorricho, "Resource Slicing in Virtual Wireless Networks: A Survey," *IEEE Transactions on Network and Service Management*, 2016.

[6] P. Popovski, K. F. Trillingsgaard, O. Simeone, and G. Durisi, "5G Wireless Network Slicing for eMBB, URLLC, and mMTC: A Communication-Theoretic View," *IEEE Access*, 2018.

[7] A. A. Barakabitze, A. Ahmad, R. Mijumbi, and A. Hines, "5G Network Slicing Using SDN and NFV: A Survey of Taxonomy, Architectures and Future Challenges," *Computer Networks*, 2020.

[8] M. Chowdhury, F. Samuel, and R. Boutaba, "PolyViNE: Policy-Based Virtual Network Embedding across Multiple Domains," in *Proceedings of the Second ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures*, 2010.

[9] S. Li, M. Y. Saidi, and K. Chen, "Multi-Domain Virtual Network Embedding with Coordinated Link Mapping," in *Proceedings of the Twenty-Fourth International Conference on Software, Telecommunications and Computer Networks*, 2016.

[10] Y. Zheng, S. Ravi, E. Kline, S. Koenig, and T. K. S. Kumar, "Conflict-Based Search for the Virtual Network Embedding Problem," in *Proceedings of the Thirty-Second International Conference on Automated Planning and Scheduling*, 2022.

[11] G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant, "Conflict-Based Search for Optimal Multi-Agent Pathfinding," *Artificial Intelligence*, 2015.

[12] Y. Zheng, S. Ravi, E. Kline, L. Thurlow, S. Koenig, and T. K. S. Kumar, "Improved Conflict-Based Search for the Virtual Network Embedding Problem," in *Proceedings of the Thirty-Second International Conference on Computer Communications and Networks*, 2023.

[13] Y. Zheng, E. Kline, L. Thurlow, S. Ravi, S. Koenig, and T. K. S. Kumar, "Adapting the Conflict-Based Search Framework for the Virtual Network Embedding Problem," *Journal of Artificial Intelligence Research*, 2026.

[14] G. Schaffrath, C. Werle, P. Papadimitriou, A. Feldmann, R. Bless, A. Greenhalgh, A. Wundsam, M. Kind, O. Maennel, and L. Mathy, "Network Virtualization Architecture: Proposal and Initial Prototype," in *Proceedings of the First ACM Workshop on Virtualized Infrastructure Systems and Architectures*, 2009.

[15] M. Chowdhury, M. R. Rahman, and R. Boutaba, "ViNEYard: Virtual Network Embedding Algorithms with Coordinated Node and Link Mapping," *IEEE/ACM Transactions on Networking*, 2012.

[16] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking Virtual Network Embedding: Substrate Support for Path Splitting and Migration," *Computer Communication Review*, 2008.

[17] A. Fischer, J. F. Botero, M. T. Beck, H. de Meer, and X. Hesselbach, "Virtual Network Embedding: A Survey," *IEEE Communications Surveys and Tutorials*, 2013.

[18] H. Cao, S. Wu, Y. Hu, Y. Liu, and L. Yang, "A Survey of Embedding Algorithm for Virtual Network Embedding," *China Communications*, 2019.

[19] Y. Zhu and M. H. Ammar, "Algorithms for Assigning Substrate Network Resources to Virtual Network Components," in *Proceedings of the Twenty-Fifth Joint Conference of the IEEE Computer and Communications Societies*, 2006.

[20] X. Cheng, S. Su, Z. Zhang, H. Wang, F. Yang, Y. Luo, and J. Wang, "Virtual Network Embedding through Topology-Aware Node Ranking," *Computer Communication Review*, 2011.

[21] I. Houidi, W. Louati, W. B. Ameur, and D. Zeghlache, "Virtual Network Provisioning across Multiple Substrate Networks," *Computer Networks*, 2011.

[22] D. Dietrich, A. Rizk, and P. Papadimitriou, "Multi-Domain Virtual Network Embedding with Limited Information Disclosure," in *Proceedings of the International Federation for Information Processing Networking Conference*, 2013.

[23] M. Shen, K. Xu, K. Yang, and H.-H. Chen, "Towards Efficient Virtual Network Embedding across Multiple Network Domains," in *Proceedings of the Twenty-Second IEEE International Symposium on Quality of Service*, 2014.

[24] F. Esposito, D. D. Paola, and I. Matta, "On Distributed Virtual Network Embedding with Guarantees," *IEEE/ACM Transactions on Networking*, 2016.

[25] B. M. Waxman, "Routing of Multipoint Connections," *IEEE Journal on Selected Areas in Communications*, 1988.