Cost-Aware Node Ranking Algorithm for Embedding Virtual Networks in Internet of Vehicles

Khoa Nguyen(✉), Wei Shi, and Marc St-Hilaire

School of Information Technology, Carleton Univeristy, Ottawa, ON, Canada
{khoatnguyen,wei.shi,marc.sthilaire}@carleton.ca

Abstract. Internet of Vehicles (IoV), a subset of the Internet of Things (IoT), has been commonly considered as a primary paradigm for the anticipated success of the intelligent transportation. Network Virtualization (NV) enables flexible, cost-effective and on-demand services over the deployments of heterogeneous network service requests on a shared physical infrastructure. The most challenging problem of NV is Virtual Network Embedding (VNE) which involves embedding Virtual Network Requests (VNRs) into the substrate network efficiently and effectively, meeting several rigid resource constraints. In fact, the conventional VNE problem has been extensively investigated in the datacenter architecture in which the network topology is always fixed. Although recent studies have addressed the VNE problem considering IoV demands in datacenter networks, the development of VNE in IoV contexts, where connected and autonomous vehicles operate as substrate network nodes to handle incoming VNRs, is still in its early stages. This paper proposes a dual ranking-value and cost-aware heuristic algorithm, called CARA, for dealing with the online VNE problem in IoV. By considering vehicle mobility, our solution guarantees that the selected vehicles will remain within the preferable radius of the VNR while serving it. A thorough evaluation of our proposed VNE algorithm under the Random Waypoint (RWP) mobility model reveals that it accepts more than 40% VNRs while maintaining a drop-out ratio of almost zero and an execution time that is very practical.

Keywords: Network virtualization · Virtual network embedding · Vehicle ranking · Internet of vehicles · Heuristic algorithm

1 Introduction

Due to mobile traffic explosion, the growth of safety data, and the variety of vehicular applications in the field of intelligent transportation, extensive research on the Internet of Vehicles (IoV) becomes increasingly critical. In fact, connected vehicles and mobile infrastructure (e.g., roadside units, base stations) aren't always able to communicate with each other directly; thus, they will forward and receive messages/data through intermediate vehicles, forming irregular

Y. Kambayashi et al. (Eds.): AICON 2022, LNICST 477, pp. 3–19, 2023.
https://doi.org/10.1007/978-3-031-29126-5_1

self-organizing networks. In addition, migrating the computations from the core network to the edge can reduce the network loads of the core network, especially during the peak hours. Virtual Network Embedding (VNE) can become a propitious key for tackling the resource allocation problem in IoV. VNE enables the physical network resources to be shared between different Virtual Network Requests (VNRs) while maintaining an isolated coexistence of Virtual Networks (VNs) on the underlying Substrate Network (SN).

The VNE problem has been widely acknowledged as $\mathcal{NP}$-Hard either for Virtual Node Mapping (VNoM) or Virtual Link Mapping (VLiM) [1,2]. Although a number of optimization models (e.g., Integer Programming (IP)) are formulated to approach optimal VNE solutions, they are not likely to be deployed for solving online VNE problems due to scalability, time complexity, and impractical implementation issues. As a result, most VNE solutions focus on the efficient design of heuristic algorithms to deal with the impediments of the formulated optimization models. A typical VNE process comprises two stages: VNoM and VLiM. Most VNE approaches have attempted to address the VNoM phase followed by the VLiM phase which usually relies on the shortest path methods (e.g., Dijkstra's algorithm).

The VNE problem has been significantly studied in cloud computing in which the network topology is always fixed. Therefore, conventional VNE approaches cannot be used for IoV environments where vehicles move on roads dynamically and their locations keep changing over time. Vehicle communications is indeed based on wireless links (e.g., IEEE 802.11p standard), and the eligible vehicles from different vendors (e.g., Aspark, Apple, Audi, Tesla, Vinfast) can practically register with service providers to provide network services. When a VNR arrives and stays in the network for a random duration, it requires a particular topology with stringent network resources while the substrate topology keeps changing over time due to vehicle mobility. This dynamism complicates the design of VNE algorithms, and existing VNE approaches have not considered all these characteristics. Consequently, typical VNE approaches that work efficiently under static network topologies might not work in IoV environments since vehicles serving as service nodes are always changing their geographical locations due to their mobility. Thus, VNE algorithms should be sufficiently fast and should also consider vehicle mobility; otherwise, it could result in situations where some of the selected vehicles are no longer able to provide the requested services since they moved out of the preferable radius of the given VNR. Hence, we need new VNE algorithms that can handle online VNRs in IoV environments.

In this paper, we propose a dual ranking-value and cost heuristic algorithm, called CARA, for addressing the online VNE problem in IoV. The algorithm is aimed at dual objectives in which the vehicles that obtained the highest ranking values while producing the lowest costs are rapidly selected for mapping the VNR. First, it effectively ranks available vehicles based on network attributes, and then the vehicle producing the lowest cost value among a proportional number of the best ranked vehicles is greedily selected for mapping the virtual node. In fact, splittable mapping might theoretically support a better resource uti-

lization solution, but this approach could create abundant overheads through maintaining the consistency of the network state. Moreover, splittable embedding can lead to out-of-order deliveries which increases latency and becomes detrimental to delay-sensitive applications [3]. As a result, we merely consider unsplittable embedding solutions in this research. To the best of our knowledge, this is one of the very first papers that directly deals with the online VNE problem in IoV environments considering vehicle mobility. The major contributions of this paper can be summarized as follows:

- We study a centralized VNE model addressing the online VNE problem in IoV environments. Service time of vehicles, wireless communication links, vehicle locations and memory capacity are considered along with other typical resource constraints (e.g., CPU).
- We introduce a neighboring time factor to track the eligible vehicles for VNoM. This ensures that the selected vehicles will remain within the preferable radius of the VNR while serving it. Accordingly, we introduce three new performance metrics to testify this neighboring time: average initial acceptance ratio, actual acceptance ratio, and drop-out ratio. By introducing this neighboring time factor, we can remove a strong assumption that nodes remain available for the duration of the request, which is commonly used in most VNE papers.
- We propose an efficient heuristic VNE algorithm, called CARA, that ranks the available vehicles based on network attributes. The top ranked vehicles, driven by a cost function, form the best solution for mapping the corresponding virtual node. This cost function not only takes the cost of the preceding node mapping into account, but also considers those of all already-mapped node mappings to pre-minimize the cost values. By considering mapping cost in VNoM, it allows the solution to coordinate the link mapping in advance.
- We perform extensive evaluations under the Random Waypoint (RWP) mobility model. Our results indicate that the proposed algorithm performs better than five existing VNE algorithms, and it is also highly practical due to its fast execution time.

The remainder of this paper is organized as follows: the related work is presented in Sect. 2 while the network model is formulated in Sect. 3. Our proposed approach is described in Sect. 4. Performance evaluation is introduced in Sect. 5. Finally, Sect. 6 concludes the paper and presents future work.

2 Related Work

The VNE problem is commonly known as $\mathcal{NP}$-hard, which is intractable to solve by formulated optimization models due to high time complexity. Thus, the majority of the research papers in this field have focused on designing suboptimal heuristic algorithms.

Towards VNE research on conventional cloud computing, [2] proposed a node-link coordination relaxing the integer constraints for addressing the VNoM stage.

The rounding techniques were utilized to choose the most proper node mappings. Li et al. in [4] presented a VNE paradigm that jointly took both virtual resources and substrate resources of the core network into account in order to provide diversified services in 5G networks efficiently. Cao et al. in [5] and [6] introduced several network attributes and the global network resources for ranking substrate nodes and virtual nodes before mapping VNRs. Recently, [7] proposed three novel Genetic Algorithm (GA)-based approaches jointly coordinating virtual node and link mappings in a one-stage mapping where the VLiM stage was deployed on several different path searching methods. A novel conciliation mechanism was applied to deal with a set of infeasible link mappings in GA operations. Troia et al. in [8] proposed a VNE framework handling the admission control and resource allocation problems in 5G networks by deploying a deep reinforcement learning.

More specifically towards the VNE problem in IoV, the authors in [9] researched a bandwidth-aware VNE in multi-domain problems in the wireless communication mode. A Particle Swarm Optimization (PSO) algorithm was then introduced to enhance the performance of mapping VNRs. Recently, Fan et al. in [10] have investigated the hierarchy-based dynamic VNE problem in the static business environment considering IoV demands. The authors presented a node-evaluation model to sort and evaluate substrate nodes and then a global-resource capacity algorithm to solve the virtual node mappings.

However, all of those VNE papers merely concentrate upon addressing the VNE problem in conventional cloud computing scenarios. Even if [9] and [10] dealt with the VNE problem considering IoV requests in cloud computing, these papers are not solving the online VNE problem under IoV environments subject to vehicle movements. To the best of our knowledge, this is one of the very first papers tackling the online VNE problem directly, considering the vehicle mobility in IoV environment by a cost-aware vehicle ranking algorithm.

3 Network Model and Problem Description

3.1 Virtual Network Assignment

In this research, the software-defined edge computing paradigm in which the network is split into control plane and data plane is favored. The control plane is involved in the global view of the network while the data plane is related to the moving vehicles. Vehicles need to register with Service Providers (SPs) to provide network services, and periodically update their information to the controller such as state, location, velocity, and direction. With the aim of approaching a generic model for tackling the VNE problem in IoV environments, we model our network as a weighted undirected graph $G^s = (N^s, L^s)$, where N^s is a set of physical nodes (e.g., available vehicles) while L^s denotes the set of physical links. Vehicles can indirectly communicate with others via intermediate vehicles due to several issues such as blockages, security, capability, vendor preferences, etc. As a result, it is possible to have a physical path between a pair of vehicles that traverses several intermediate vehicles. A vehicular node $n^s \in N^s$ has multiple information

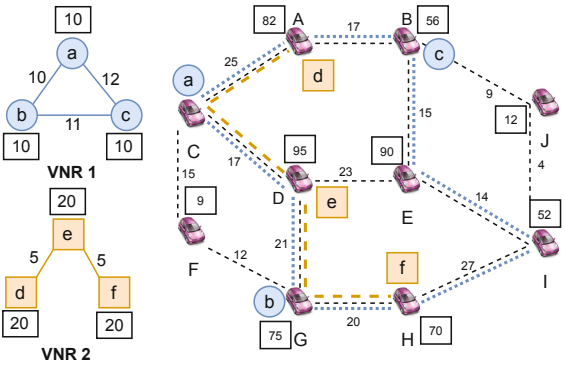

Fig. 1. An example embedding of VNRs onto a shared substrate network

including a current geographical location $loc(n^s)$ (i.e., GPS coordinates), the available CPU capacity $c(n^s)$, memory $m(n^s)$, and a registered service time $t_s(n^s)$ that the vehicle promises to deliver all assigned computational services while within the coverage of the corresponding base station at the registration. Furthermore, each vehicle also has a speed $s(n^s)$ and current direction $dir(n^s)$. On the other hand, each physical link $l^s \in L^s$ between a pair of vehicles possesses a $\zeta_s(l^s)$ link capacity. Without loss of generality, storage resources are excluded in this paper. Figure 1 shows a substrate network with two VNRs with node and link constraints. Similar to the substrate network, we model the i^{th} arrived VNR as a weighted undirected graph denoted as $G_i^v = (N_i^v, L_i^v)$, where N_i^v expresses the set of virtual nodes and L_i^v denotes the set of corresponding virtual links. Each virtual node $n_i^v \in N_i^v$ requires a CPU capacity $c(n_i^v)$, memory $m(n_i^v)$, whereas each virtual link $l_i^v(s_i^v, d_i^v) \in L_i^v$ between a virtual source node s_i^v and a virtual destination node d_i^v has a demanded data rate capacity $b(l_i^v)$. Each VNR has a preferred mapping radius $D(n_i^v)$ which states how far the virtual node n_i^v can be allocated from $loc(n_i^v)$, and it also has a lifetime $t_v(G_i^v)$. Similar to many VNE research paper, our main objective is to maximize the average acceptance ratio which has been widely considered as the most important factor in VNE, accompanying with average revenue, average cost and average path length whilst meeting node and link constraints imposed by VNRs. Given the above notation, the node and link constraints can be formulated as follows:

Node constraints:

$$c(n_i^v) \leq \mathcal{R}_{N_c}(E_N(n_i^v)) \tag{1}$$

$$m(n_i^v) \leq \mathcal{R}_{N_m}(E_N(n_i^v)) \tag{2}$$

$$\mathcal{D}(loc(n_i^v), loc(E_N(n_i^v))) \leq D(n_i^v) \tag{3}$$

$$t_v(G_i^v) \leq \mathcal{R}_{t_s}(E_N(n_i^v)) \tag{4}$$

$$E_N(n_i^v) \in N^s \tag{5}$$

$$\mathcal{R}_{N_c}(n^s) = c(n^s) - \sum_{n_i^v \to n^s} c(n_i^v) \tag{6}$$

$$\mathcal{R}_{N_m}(n^s) = m(n^s) - \sum_{n_i^v \to n^s} m(n_i^v) \tag{7}$$

$$\mathcal{R}_{t_s}(n^s) = t_s(n^s) - \sum_{n_i^v \to n^s} t_v(G_i^v), n_i^v \in G_i^v \tag{8}$$

where $E_N(n_i^v)$ denotes the mapping solution of the virtual node n_i^v. $\mathcal{D}(i^s, j^d)$ and $\mathcal{R}_{N_{c|m}}(n^s)$ are the distance between i^s and j^d, and the residual CPU or memory capacity of a substrate node, respectively. $\mathcal{D}(i^s, j^d)$ is computed by the Euclidean distance which is based on the current locations of nodes i^s and j^d. In addition, $\mathcal{R}_{t_s}(n^s)$ denotes the remaining service time of vehicular node n^s. In VNE, a substrate node is only permitted to allocate to a virtual node in the given VNR.

Link constraints:

$$\zeta(l^s) = Wlog(1 + \gamma(l^s)) \tag{9}$$

$$\zeta(e^s) = \min_{l^s \in e^s} \zeta(l^s) \tag{10}$$

$$\zeta(e^s) \geq b(l_i^v), \forall e^s \in \mathcal{E}^s(E_L(l_i^v)) \tag{11}$$

$$0 \leq \zeta(l^s) \leq \zeta_s(l^s) \tag{12}$$

where $\mathcal{E}^s(E_L(l_i^v))$ denotes the set of all available paths from source $E_N(s_i^v)$ to destination node $E_N(d_i^v)$. $\gamma(l^s)$, $\zeta(e^s)$ and W are the Signal to Interference and Noise Ratio (SINR) of the Vehicle-to-Vehicle (V2V) link l^s, the capacity of the path $e^s \in \mathcal{E}^s$ and channel bandwidth, respectively. To lessen the inter-system interference in massive multiuser Multiple-Input Multiple Output (MIMO) transmission environments, the zero-forcing pre-coder can be implemented to project the interference in the orthogonal space and gain high spectral efficiency of V2V links by re-utilizing the cellular spectrum [11]. To reflect the network loads, rather than being dependent on specific wireless technologies, each link capacity is configured as units varying in the specific range that can be easily converted to the real values. This is applicable to a generic model, and consistent with CPU and memory capacities. Considering that the channel state fluctuates during the embedding process, each substrate link capacity is randomly attenuated within $[0-\rho]$ subject to the distance between a pair of vehicles. An embedding solution is implied as feasible if and only if it satisfies the resource constraints (1–8) for node mapping or (9–12) for link mapping. Similar to most VNE approaches, the VNE algorithm in this work comprise two separate mapping stages for embedding a given VNR: VNoM followed by VLiM.

3.2 Performance Metrics

The most important objective when dealing with the VNE problem is to maximize the number of accepted VNRs along with optimizing the revenues and the

corresponding mapping cost. In fact, these aforementioned metrics have been widely used to evaluate the efficiency of VNE algorithms. The revenue is the sum of the total virtual resources embedded on the SN over time, whereas the embedding cost of the i^{th} VNR $C(G_i^v)$ is calculated as the sum of total substrate resources allocated to the i^{th} VN.

Revenue ratio of i^{th} VNR G_i^v can be formulated as:

$$\Xi(G_i^v) = w_b * \sum_{l_i^v \in L_i^v} b(l_i^v) + w_n * \sum_{n_i^v \in N_i^v} c(n_i^v) + w_m * \sum_{n_i^v \in N_i^v} m(n_i^v) \tag{13}$$

Cost ratio of i^{th} VNR G_i^v can be formulated as:

$$\Theta(G_i^v) = \sum_{n_i^v \in N_i^v} c(n_i^v) + \sum_{n_i^v \in N_i^v} m(n_i^v) + \sum_{l_i^v \in L_i^v} \sum_{l^s \in L^s} f_{l^s}^{l_i^v} \tag{14}$$

where Ξ and Θ denote the generated revenue and network cost, respectively. $b(l_i^v)$ and $c(n_i^v)$ define the requested bandwidth of the virtual link l_i^v and the requested CPU of the virtual node n_i^v, whilst w_b, w_n and w_m are the unit weights of the bandwidth, CPU and memory resources, respectively. Likewise, $f_{l^s}^{l_i^v}$ defines the bandwidth of substrate link l^s that is allocated to the virtual link l_i^v. In this paper, we introduce three new metrics that are related to the number of accepted VNRs.

Initial Acceptance Ratio (IAR): defines a ratio between the number of accepted VNRs over the number of arrived VNRs during an interval time τ. IAR can be calculated as Eq. (15) below:

$$IAR = \left| \frac{\xi^a(\tau)}{\xi(\tau)} \right| \tag{15}$$

where $\xi^a(\tau)$ and $\xi(\tau)$ are the number of the embedded VNRs and the total number of VNRs respectively.

Actual Acceptance ratio (AAR): defines a ratio between the number of accepted and successfully completed VNRs over the number of arrived VNRs during an interval time τ. This is different from the IAR as a request can be initially accepted but as nodes are moving, it may not be able to successfully complete the request. AAR can be computed using Eq. (16):

$$AAR = \left| \frac{\xi^{a_n}(\tau)}{\xi(\tau)} \right| \tag{16}$$

where $\xi^{a_n}(\tau)$ is the number of the successfully embedded VNRs considering the neighboring time of the vehicles which is defined in Eq. 17 and 18.

$$\mathcal{T}_{\mathcal{NT}}(n^s) = \sum_{\kappa_i \in K_r} \frac{d_{\kappa_i}}{v_{\kappa_i}} \tag{17}$$

$$\mathcal{T}_{\mathcal{NT}}(n^s) \geq t_v(G_i^v) \tag{18}$$

where K_r is the number of trajectories of a vehicle when it reaches the borderline of the preferred radius of the VNR from its current location. K_r can be determined by the controller through communicating with vehicles while they are within its serving coverage.

Drop-out ratio (DR): defines a ratio between the differential number of the initial accepted VNRs and the successfully completed VNRs over the number of arrived VNRs during an interval time τ as calculated in Eq. (19):

$$DR = \left| \frac{\xi^{a}(\tau) - \xi^{a_n}(\tau)}{\xi(\tau)} \right| \tag{19}$$

4 Proposed Solution

This section presents a novel VNE algorithm consisting of three sub-stages: first the pre-processing stage chooses all available vehicles that meets the resource constraints including the neighboring time constraint to serve VNRs. Then, the vehicle ranking evaluates the selected vehicles from the previous stage by ranking them based on network attributes. Finally, a cost function is deployed to decide the vehicle that produces the lowest cost in a proportional number of the highest ranked-value vehicles for mapping the corresponding virtual node. Our approach handles a single virtual node at a time in the VNR until all virtual nodes are processed successfully. After achieving a mapping solution for all virtual nodes in a given VNR, the shortest path method is implemented for VLiM.

4.1 Pre-processing Stage

This operation is aimed at narrowing down the search space, in which the vehicles that satisfy several stringent resource constraints consisting of CPU, memory, service time, preferable radius of the VNR and the neighboring time as defined in Eqs. (17–18) are selected as potential node mappings. More precisely, vehicular nodes meeting Eqs. (1–8) are taken into account for mapping the virtual node. Vehicles must satisfy resource constraints demanded by the VNR, and their remaining service time must be long enough to provision services. Additionally, those vehicles must remain within the required radius of the VNR while serving it. Intuitively, vehicles that are closer to the source of the user request are more likely to be chosen for embedding the virtual node. At first glance, this strategy seems good enough for mapping a single virtual node. We also need to consider the vehicle's current speed and its direction to estimate the availability of the vehicle to ensure that the vehicle is always within the VNR's preferable radius while serving it. Otherwise, this can lead to a situation where the vehicle could move away from the radius of the request while providing its services, which would result in service disruption, allocated resource waste, and unexpected re-mappings. For this reason, existing VNE approaches are not well suited for IoV environments. To solve this problem, in this paper, we propose a function to estimate the remaining time that a vehicle is still within the VNR's preferable radius, called the neighboring time, to serve the given VNR.

Another challenge is that a VNR always includes a set of virtual nodes connected by virtual links to form a virtual network. Thus, the node mappings are admitted to create a considerable influence on the successive link mappings. Each node mapping does affect the successive embedding as well as the whole solution of the node mappings in general.

Suppose we have a set of eligible vehicles selected by the pre-processing stage that can allocate to a virtual node. We generally need to evaluate each vehicle in this set to decide the most appropriate one. Most of the recent VNE algorithms are using the node ranking approach to determine the importance of substrate nodes in a static topology based on network attributes, which attempt to exploit the possible relationships between these attributes and combine them in a single weighted formulation to recursively rank the nodes until obtaining a stable performance. These approaches are indeed inapplicable in IoV environments in which the network topology is extremely dynamic and the environment behaviors are nonstationary. Furthermore, finding out mutual relationships between different network attributes that possess very different characteristics and dimensions, and then combining them together are not indeed an easy task. Moreover, recent ranking approaches have not considered the embedding cost between the already-mapped nodes and each node in the set of potential nodes in order to select the best one. To solve this problem, we propose a ranking mechanism based on network attributes to rank the potential nodes determined from the pre-processing stage. A proportional number of the highest ranked vehicles are then chosen for the next stage where the most appropriate vehicle for mapping the virtual node is selected by the cost function. This cost function takes the cost of all already-mapped nodes into account and allows to directly coordinate the link mapping into the virtual node mapping in advance. Our solution can balance a trade-off between the vehicle producing a high ranked value while achieving a low embedding cost. Without considering the whole network topology, thanks to the pre-processing stage, our solution is not only scalable, but also adaptable to any sudden changes in the SN.

4.2 Vehicle Ranking

As described in the previous section, vehicle ranking will rank the available vehicles selected from the pre-processing stage based on network attributes. Instead of trying to obtain stable ranking values for all substrate nodes in a static SN, our approach simply needs to consider the previous node mappings, which allows to rapidly adapt to the uncertainty of vehicle mobility or scalability issues. The proposed VNE algorithm can quickly handle topology changes due to vehicle mobility, outage, exhausted lifetime, or any unexpected incidents. By taking all previous node mappings into account, the fragmentation problem can be reduced, improving the average acceptance ratio.

Node Degree defines the total number of adjacent edges of a node n^s in the network.

$$\Phi(n^s) = \sum_{m^s \in \nu_n} \sigma(l^s_{\overline{\mathrm{n^s m^s}}}) \tag{20}$$

where ν_n is a set of current neighbors of the vehicle n^s, and σ is a binary function that equals to 1 if there is a direct adjacent edge between n^s and m^s. Node degree, also acknowledged as degree centrality, implies connectivity, interaction, and the extensibility of a vehicular node. It indicates the capability to communicate with other vehicular nodes directly. A higher node degree is more favorable.

Node Intensity sums all adjacent link capacity of node n^s in the network.

$$I(n^s) = \sum_{m^s \in \nu_n} \zeta(l^s_{\overline{\mathrm{n^s m^s}}}) \tag{21}$$

The intensity attribute measures the strength of a vehicular node, which might have a big impact on the VLiM phase. Therefore, a high node intensity is more preferable.

Neighboring Distance defines the total distance of a node to all its neighbors. Vehicular nodes, that are closer to each other, can improve the resource fragmentation and somehow prevent service interruptions caused by uncertain vehicle mobility. Hence, smaller values of neighboring distance is desired.

$$\Pi(n^s) = \sum_{m^s \in \nu_n} \mathcal{D}(n^s, m^s) \tag{22}$$

where

$$\mathcal{D}(n^s, m^s) = \sqrt{(x_{n^s} - x_{m^s})^2 + (y_{n^s} - y_{m^s})^2} \tag{23}$$

Node Strength determines the total strength of a vehicular node considering the remaining resources such as CPU and memory, node degree and node intensity.

$$\mathcal{S}(n^s) = \mathcal{R}_{N_c}(n^s) * \mathcal{R}_{N_m}(n^s) * \Phi(n^s) * \frac{I(n^s)}{\Pi(n^s)} \tag{24}$$

The normalized $\mathcal{S}$ is calculated as follows:

$$\overline{\mathcal{S}}(n^s) = \frac{\mathcal{S}(n^s)}{\sum\limits_{n^s \in \mathcal{P}_v} \mathcal{S}(n^s)} \tag{25}$$

where $\mathcal{P}_v$ denotes the set of vehicles selected from the pre-processing stage.

4.3 Intermediate Node Cost

Although vehicle ranking can evaluate the importance of a vehicular node, it does not reflect the interactions between the node and the already-mapped nodes of the VNR, and the embedding cost if the node is chosen considering the previous knowledge of node mappings. Hence, we introduce a new attribute, called intermediate node cost Ψ, to estimate the potential mapping cost if a node is selected for embedding the corresponding virtual node subject to all already-mapped nodes as calculated in Eq. (26).

$$\begin{aligned} \Psi(n^s) = c(n_i^v) + m(n_i^v) + \sum_{u^v k^v \in G_i^v} b(u^v k^v) * \Upsilon(e^s)), \\ \forall n^s \in E_N(u^v), n^s \in \mathcal{P}_v,\ k^v \in \mathcal{M}_a,\ e^s \in \mathcal{E}^s(E_L(u^v k^v)). \end{aligned} \tag{26}$$

where $\Upsilon(e^s)$ denotes the hop-count of a substrate path e^s.

The normalized intermediate node cost is calculated as follows:

$$\overline{\Psi}(n^s) = \frac{\Psi(n^s)}{\sum\limits_{n^s \in \mathcal{P}_e} \Psi(n^s)} \tag{27}$$

where $\mathcal{P}_e$ is the elite vehicles that are sorted out from $\mathcal{P}_v$. After ranking the nodes in the node ranking stage, a proportional number of elite vehicles obtaining the highest ranking values are selected for the next stage. The vehicle producing the lowest intermediate cost, calculated from Eq. (26), is then greedily preferred as the node mapping for the given virtual node. This approach does not only balance the trade-off between the ranking values and the embedding cost, but also simplifies the difficulty of integrating different domains of network attributes having different characteristics.

4.4 VNE-IoV Algorithm

Algorithm 1 illustrates our VNE solution, called **CARA**, in which the network input is the SN and a VNR as indicated in Line 2, whereas the output is the node and link mappings for the VNR. Our algorithm also adapts to this requirement. The proposed VNE algorithm begins with the node mapping stage in lines [5–19], followed by the link mapping phase in lines [20–23]. In terms of the node mapping, the pre-processing stage in lines [6–9] picks the available vehicles from the SN satisfying multiple resource constraints (Eqs. (1–8), (17), (18)) and adds to $\mathcal{P}_v$. Then the algorithm computes the normalized strength vector $\overline{\mathcal{S}}$ for all vehicles in $\mathcal{P}_v$ through Eqs. (20–25) and sorts the vector $\overline{\mathcal{S}}$ in descending order in lines [10–13]. Line 14 indicates that the proportional number of vehicles generating the highest ranking values are selected and then added to the elite set $\mathcal{P}_e$, where η denotes the candidate selection proportion. In lines [15–17], the algorithm calculates the normalized vector $\overline{\Psi}$ as Eqs. (26–27) and then selects the vehicle that produces the lowest cost value as the node mapping for the corresponding virtual node n_i^v in line 18. The VNoM is repeated until all virtual

Algorithm 1 : **CARA**

1: **Input**:
2: Network G^s and VNR G_i^v
3: **Output**:
4: The mapping solution for G_i^v.
▷ Virtual nodes and links in a given VNR are processed in order.
▷ Node Mapping:
5: **for** $n_i^v \in N_i^v$ **do**
▷ Pre-processing:
6: **for** $n_s \in N_s$ **do**
7: **if** n^s satisfies Eqs. (1)-(8), 17, 18 **add** n^s to $\mathcal{P}_v$
▷ $\mathcal{P}_v$ denotes a set of available vehicles after pre-processing stage.
8: **end for**
9: **if** $\mathcal{P}_v$ is empty **reject** the VNR.
▷ Vehicle Ranking:
10: **for** $j \in |\mathcal{P}_v|$ **do**
11: Calculate Eqs. (20)-(25) to achieve vector $\overline{\mathcal{S}}(\mathcal{P}_v[j])$
12: **end for**
13: Sort $\overline{\mathcal{S}}$ in descending order
14: $\mathcal{P}_e \leftarrow \eta \times |\mathcal{P}_v|$ ▷ η: candidate selection proportion
▷ Intermediate node cost:
15: **for** $k \in |\mathcal{P}_e|$ **do**
16: Calculate Eqs. (26)-(27) to obtain vector $\overline{\Psi}(\mathcal{P}_e[k])$
17: **end for**
18: Select $\mathcal{P}_e[k]$ producing the smallest values of $\overline{\Psi}(\mathcal{P}_e[k])$
19: **end for**
▷ Link Mapping:
20: **for** $l_i^v \in L_i^v$ **do**
21: Using the shortest path method to find the solution for l_i^v based on the information of achieved node mappings
22: **if** l_i^v fails to map **reject** the VNR.
23: **end for**

nodes are successfully handled. The VLiM is conducted in lines [20–23] to embed each virtual link to a feasible path. If any node or link is unsuccessfully to map, the whole VNR will be rejected (lines (9), (22)).

5 Performance Evaluation

We compare our proposed algorithm against several VNE algorithms consisting of BEST_FIT, FIRST_FIT, RANDOM, G-SP [2] and ETAGNR [5]. For BEST_FIT, the virtual node will be allocated to the vehicle possessing the least available resources (e.g., CPU, memory) to meet the VNR [12], whereas FIRST_FIT picks the first vehicle in the substrate nodes that satisfies the

resource constraints. In contrast, the RANDOM algorithm determines the available vehicle for virtual node mapping randomly. G-SP is broadly recognized as the most popular unsplittable VNE algorithm and also one of the fastest algorithms in this field, while ETAGNR exploits the topology attributes and the global network resources for ranking the substrate nodes.

Performance evaluations are divided into two parts. In the first part, we demonstrate the efficiency of our proposed cost-aware ranking algorithm by comparing the initial average acceptance ratio, average revenue, average cost, and average path length when processing VNRs at different traffic loads. In this evaluation, we remove the neighboring time from the proposed algorithm, called CARA_noNT, to justify the CARA algorithm itself. In the second part, we evaluate the effect of the neighboring time factor as described in Sect. 4.1. The neighboring time factor is evaluated by calculating the actual acceptance ratio and the drop out ratio. Furthermore, we also compare the average execution time between all algorithms at different network loads. Our simulation is conducted on a Ubuntu 22.04.1 LTS 64-bit platform with 16 GB memory and Intel Core i5-6200U CPU@2.30 GHz×4.

5.1 Simulation Setup

We have developed a discrete-event simulator to assess the proposed VNE algorithm and other rivals. The GT-ITM topology generator is utilized to create SN and VNs. SN has 50 nodes with 208 edges, and the physical CPU/link and memory capacities are uniformly distributed between [50–100] units and [100–200] units, respectively. The VNR arrivals follow the Poisson process with a mean arrival rate of λ requests per time unit, while their lifetime follows an exponential distribution with a mean $1/\mu = 1000$ time units. The miscellaneous loads of VNRs can be evaluated by $\lambda \times (1/\mu)$ Erlangs. Accordingly, the arrival rates of VNRs in this paper are varied between 1 and 8 requests per 100 time units. ρ and η are set out to 30% and 50%, respectively. With arbitrary SN and VNs generated for extensive evaluations, it indicates that our proposed algorithm can adapt to any topologies with diverse traffic patterns. Each simulation runs for 50,000 time units, which is 50 times longer than the average lifetime of a VN. This allows to generate a large number of independent samples. All performance figures were based upon average values with 95% confidence interval. The error bars are tiny due to the large number of samples used, which shows that our simulation results are reliable.

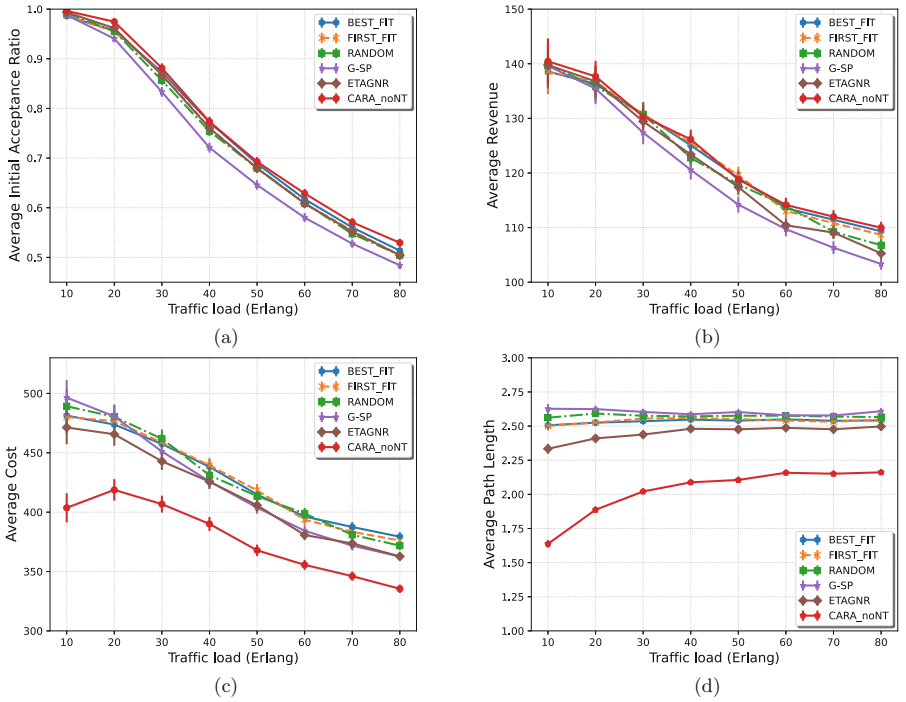

Fig. 2. (a) Average IAR (b) Average revenue (c) Average cost (d) Average path length

5.2 Vehicle Mobility Model

With the aim to approach a generic model for the VNE problem in IoV environments and to primarily concentrate on investigating the resource allocation, we used the Random Waypoint (RWP) model [13] to evaluate all VNE algorithms in this paper. Each vehicle randomly moves within a simulation area (1000×1000) with a random direction and a random speed in a range of [5–60 km/h], and the pausing time is set to 100 time units. We have used the BonnMotion software [14] to generate the vehicle mobility patterns under tracing files. When running simulations, the simulator reads the tracing files and updates the vehicles' information into the substrate network.

5.3 Evaluation Results

Performance results are illustrated in Figs. 2 and 3. For the first part of the evaluation, as we want to emphasize the efficiency of the proposed algorithm, we do not consider the neighboring time in the pre-processing stage. The proposed algorithm in this evaluation is called CARA_noNT. Figure 2a indicates that our algorithm accepts more VNRs than all rivals. The reason is that the proposed algorithm has an efficient cost-aware ranking mechanism with effective network

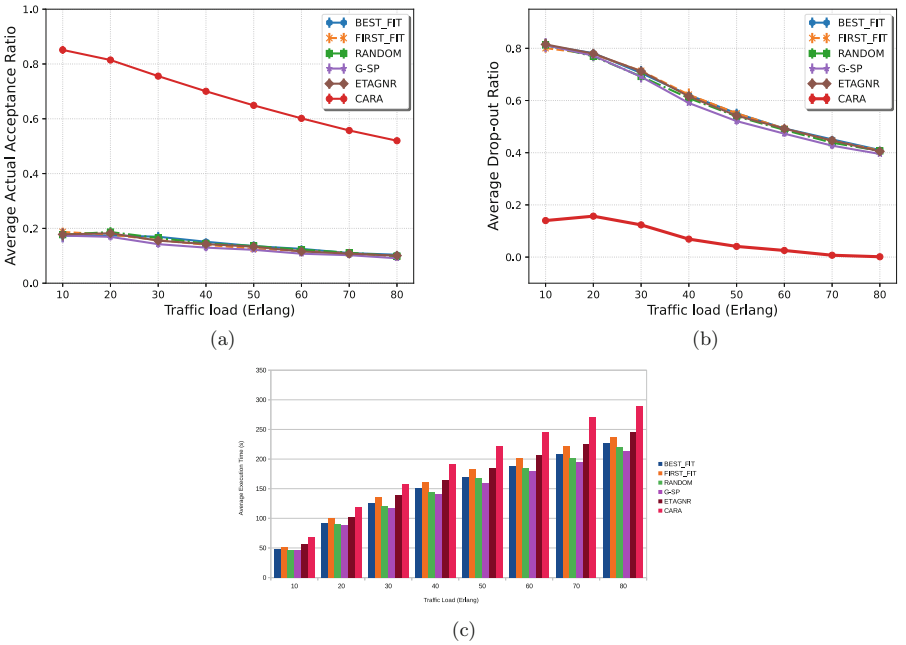

Fig. 3. (a) Average AAR (b) Average DR (c) Average execution time

attributes. Among the rivals, ETAGNR performs better than others when the traffic load is low, but when the traffic load increases, BEST_FIT, FIRST_FIT, and RANDOM perform better than ETAGNR while G-SP performs worst. Figures (2b–2c) confirm these results since CARA_noNT achieves the highest average revenue while producing the lowest average embedding cost. The results of Fig. 2c is validated by Fig. 2d when the proposed algorithm CARA_noNT obtains the shortest path length.

In the second part of the evaluation, we reevaluated the node mappings that are part of all VNE algorithms. If the vehicles selected by the algorithms moved away from the preferred mapping radius of the VNRs, they are marked as unsuccessful VNRs. In this evaluation, our proposed algorithm, called CARA, considers the neighboring time in the pre-processing stage. We compare the VNE algorithms towards the average actual acceptance ratio and the drop-out ratio. The results show that CARA significantly outperforms all other algorithms as its initial acceptance ratio (Fig. 2a) and its actual acceptance ratio (Fig. 3a) are relatively close. CARA with neighboring time consideration performs best among others, especially when the network becomes congested. As shown in Figs.(3a–3b), CARA keeps high acceptance ratios ranging from 52.82% to 85.64% while remaining the drop-out ratio almost zero at the highest traffic loads. In contrast, the initial acceptance ratios of the rivals in Fig. 2a are now significantly reduced from 39.52% up to 82% as illustrated through the average drop-out ratio in Fig. 3b, leading to their very low actual acceptance ratios varying from 9.1% up

to only 18.7% as shown in Fig. 3a. All rivals perform quite similar in general, and BEST_FIT and RANDOM perform slightly better than the others in particular.

Finally, we also measured the execution time of all VNE algorithms to identify the practicality of our proposed approach. As shown in Fig. 3c, the execution time of the proposed algorithm (CARA) is competitive since it rapidly ranks the available vehicles that have been sorted out from the pre-processing stage. Therefore, the proposed VNE algorithm is not only efficient in embedding VNRs, but also practical and scalable.

6 Conclusion and Future Work

IoV is foreseen to become a major research in intelligent transportation systems. This paper directly deals with the online VNE problem in IoV environments taking vehicle mobility into account. Vehicular service time, wireless links, current vehicle locations, and memory capacity are also considered in the proposed VNE model. Furthermore, we also proposed a novel cost-ware vehicle-ranking algorithm that strikes a balance between the high ranking-values of vehicles and the interactions with already-mapped nodes through the intermediate cost to select the most appropriate vehicles for mapping online VNRs. Extensive simulations illustrate that our proposed VNE approach is not only efficient and scalable in the VN embedding, but also practical due to its rapid execution time. For future work, we plan to investigate extra mobility-related metrics and also to evaluate the proposed solutions under different mobility models.

References

1. Gil Herrera, J., Botero, J.F.: Resource allocation in NFV: a comprehensive survey. IEEE Trans. Netw. Serv. Manage. **13**(3), 518–532 (2016)
2. Chowdhury, M., Rahman, M.R., Boutaba, R.: Vineyard: virtual network embedding algorithms with coordinated node and link mapping. IEEE/ACM Trans. Netw. **20**(1), 206–219 (2012)
3. Paschos, G.S., Abdullah, M.A., Vassilaras, S.: Network slicing with splittable flows is hard. In: 2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), pp. 1788–1793 (2018)
4. Li, J., Zhang, N., Ye, Q., Shi, W., Zhuang, W., Shen, X.: Joint resource allocation and online virtual network embedding for 5G networks. In: 2017 IEEE Global Communications Conference (GLOBECOM 2017), pp. 1–6 (2017)
5. Cao, H., Yang, L., Zhu, H.: Embedding virtual networks using a novel node-ranking approach via exploiting topology attributes and global network resources. In: 2017 9th International Conference on Wireless Communications and Signal Processing (WCSP), pp. 1–6 (2017)
6. Cao, H., Yang, L., Zhu, H.: Novel node-ranking approach and multiple topology attributes-based embedding algorithm for single-domain virtual network embedding. IEEE Internet Things J. **5**(1), 108–120 (2018)
7. Nguyen, K., Huang, C.: Towards adaptive joint node and link mapping algorithms for embedding virtual networks: a conciliation strategy. In: IEEE Transactions on Network and Service Management, p. 1 (2022)

8. Troia, S., Vanegas, A.F.R., Zorello, L.M.M., Maier, G.: Admission control and virtual network embedding in 5G networks: a deep reinforcement-learning approach. IEEE Access **10**, 15860–15875 (2022)
9. Zhang, P., Wang, C., Aujla, G.S., Kumar, N., Guizani, M.: IoV scenario: implementation of a bandwidth aware algorithm in wireless network communication mode. IEEE Trans. Veh. Technol. **69**(12), 15774–15785 (2020)
10. Fan, W., et al.: Dynamic virtual network embedding of mobile cloud system based on global resources in internet of vehicles. IEEE Trans. Veh. Technol. **70**(8), 8161–8174 (2021)
11. Liu, X., Li, Y., Xiao, L., Wang, J.: Performance analysis and power control for multi-antenna V2V underlay massive MIMO. IEEE Trans. Wireless Commun. **17**(7), 4374–4387 (2018)
12. Light, J.: Green networking: a simulation of energy efficient methods. Procedia Comput. Sci. **171**, 1489–1497 (2020)
13. Kezia, M., Anusuya, K.V.: Mobility models for internet of vehicles: a survey. Wirel. Pers. Commun. **125**, 1857–1881 (2022). https://doi.org/10.1007/s11277-022-09637-7
14. Aschenbruck, N., Ernst, R., Gerhards-Padilla, E., Schwamborn, M.: Bonnmotion: a mobility scenario generation and analysis tool. In: ICST (2010)