A Coordinated Heuristic Approach for Virtual Network Embedding in Cloud Infrastructure

Nahid Hamzehee Nia[1], Sepideh Adabi[1*] and Majid Nikougoftar Nategh [2]
[1] Department of Computer Engineering, North Tehran Branch, Islamic Azad University
Tehran 19696-33651, Iran.
[e-mail: n.hamzehee.se@gmail.com; adabi.sepideh@gmail.com]
[2] Department of Information and Technology, University of Qom
Qom 37161-46611, Iran
[e-mail: m.nikougoftar@qom.ac.ir]
*Corresponding author: Sepideh Adabi

Received October 15, 2016; revised February 9, 2017; accepted February 9, 2017; published May 31, 2017

Abstract

A major challenge in cloud infrastructure is the efficient allocation of virtual network elements on top of substrate network elements. Path algebra is a mathematical framework which allows the validation and convergence analysis of the mono-constraint or multi-constraint routing problems independently of the network topology or size. The present study proposes a new heuristic approach based on mathematical framework "paths algebra" to map virtual nodes and links to substrate nodes and paths in cloud. In this approach, we define a measure criterion to rank the substrate nodes, and map the virtual nodes to substrate nodes according to their ranks by using a greedy algorithm. In addition, considering multi-constraint routing in virtual link mapping stage, the used paths algebra framework allows a more flexible and extendable embedding. Obtained results of simulations show appropriate improvement in acceptance ratio of virtual networks and cost incurred by the infrastructure networks.

Keywords: Cloud infrastructure, virtual network embedding, allocation, paths algebra

1. Introduction

Cloud computing is one of the novel ideas towards the realization of the computing vision as a service which is presented to customers in different abstraction levels. According to its many advantages, the virtualization technology is an appropriate technology to implement a cloud [1]. Cloud computing provides basically three kinds of service: Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS).

IaaS can be supplied thanks to virtualization technologies. Network virtualization (NV) allows multiple heterogeneous virtual networks to cohabit on a shared substrate network. One of the main recognized challenges in NV is the efficient allocation of virtual network (VN) elements on top of substrate network (SN) elements which is known as the virtual network embedding (VNE) problem. In VNE, each virtual node is assigned to a substrate node and each virtual link is assigned to one or more substrate paths which connect the corresponding ending nodes. Due to path and node constraints, VNE is considered a NP-hard problem [2]. In order to reduce the execution time of VNE algorithms, the research community has focused on heuristic approaches. In this paper, in order to allocate resources in a substrate network, a heuristic approach is proposed which relies on a powerful and flexible mathematical framework and allows applying various policies to realize the goals of the substrate provider. To be realistic, the VNE algorithm has to handle the virtual network reuests as they arrive; hence the proposed approach considers the operation of virtual network as an online problem.

The rest of present paper is organized as follows. Section 2 presents the review of relate literature. The network model and the definition of virtual network embedding problem are presented in section 3. In section 4, proposed VNE approach based on paths algebra (i.e., CVNEPA) is described and section 5 shows the results of simulation. Finally, section 6 represents the conclusion to the study.

2. Review of Related Literature

Virtual network embedding problem refers to look for and optimize a case which is an NP-Hard problem. Therefore, for large scale problems the time to find the optimal solution becomes unaffordable. To avoid delay in the embedding of virtual network requests, VNE algorithm execution time should be minimized. Accordingly, heuristic-based VNE solutions were proposed which find acceptable solutions, compromising optimality for short execution time. For example, researchers in [3] proposed an approach to balance SN stress. Two heuristic algorithms were proposed in [3]: the ‘‘Basic VN assignment algorithm’’ to find the minimum sum of the node and link stress, and “Subdividing Algorithm” that splits each virtual network request (VNR) into a set of connected sub-VNs, each with a star topology. Lu and Turner [4] proposed a heuristic–based approach with uncoordinated node and link mapping, where their goal is to minimize the cost. The approach neglected resource constraints of virtual nodes and just considered bandwidth constraints. They limited VN topologies to spesefic backbone-star topology and evaluated their proposed approach in an off-line scenario. The approach proposed by Yu *et al.*[5] performed the VN embedding at seperated node and link mapping stages, with the purpose of maximizing long average revenue. Node mapping used a greedy algorithm to map the virtual nodes to the substrate nodes based on their amount of available resources. For link mapping, two different ways were proposed: utilize the k-shortest path algorithm for single path mapping, and map the virtual link to multiple paths by

solvingthe multi-commodity flow (MCF) problem. CPU and bandwidth constraints were considered in this proposal. An approach for coordinated node and link mapping in two stages was first proposed by Chowdhury *et al.*[6]. Authors first formulated the VNE problem as a mixed integer program through substrate network augmentation. Then, they used linear programming relaxation and devised two algorithms (DViNE and R-ViNE) using deterministic and randomized rounding techniques, respectively. After that, link mapping was performed following the same two solutions proposed in [5]. The proposed approach in [7] is a good example of heuristic-based VNE approaches in which node and virtual link mapping is performed in a single stage. Mapping is performed by reducing VNE to the Subgraph Isomorphism Detection (SID) problem, which includes finding an isomorphic subgraph (representing the VNR), fulfilling the VNR demands, inside the substrate network. In [8], after introducing a novel metric, called as global resource capacity (GRC), to quantify the embedding potential of all nodes in the SN, Gong *et al.* proposed an efficient heuristic VNE algorithm. The proposed algorithm applied a greedy loadbalance manner to map virtual nodes onto substrate nodes based on GRC. Subsequently, it adopted the shortest-path routing to embed each virtual link. Ding *et al.*[9] proposed a new two-stage VNE algorithm based on real-time topological attributes in the network, which introduces betweenness centrality into the virtual network embedding problem. The proposed algorithm embedded virtual nodes according to the node ranking results and then performed link mapping using the k-shortest path algorithm. VNE problem in [10] was formulated as a new multiple objective linear programming optimization program and then was solved in seperated node and link mapping phases. Furthermore, they proposed a heuristic algorithm based on an artificial inteligence source abstraction model (BI). Zhang *et al.*[11] proposed a node importance evaluating method based on the node degree and clustering coefficient information to measure the embedding potential of substrate nodes. Then, they proposed a two-stage VNE algorithm based on the metric of node importance, and utilized the breadth-first-search algorithm to embed the virtual nodes with the purpose of reducing the bandwidth utilization of substrate links. K-shortest path algorithm was used to map virtual links.Heuristic solutions usually suffer from getting stuck in a local optimum that could be far away from the real optimum. Meta-heuristic solutions improve the quality of results by escaping from local optimum in reasonable time. VNE approach based on the Max-Min Ant Colony meta-heuristic in [12] and based on particle swarm optimization (PSO) in [13] are good examples of metaheuristic-based solutions. None of these algorithms consider non-linear constraints. Most of these approaches perform the mapping of virtual links using k-shortest path or MCF algorithm. Virtual link mapping using k-shortest path algorithm is a greedy solution that allows finding the best path working in a mono-constraint basis, optimizing just one constraint or a function combining a set of constraints. Virtual link mapping using MCF algorithm provides a multi-path routing solution for each virtual link using optimal linear programming algorithms. Optimal linear solutions are not available when the considered constraints are not linear [14]. To deal with non-linear constraints, Botero *et al.* [14] proposed a novel strategy to solve the single-path virtual link mapping based on paths algebra which can find all the possible paths between each pair of nodes in the SN and organize them based on an unlimited number of constraints (or combination of constraints). In this strategy, virtual network is embedded in two separate stages. First, node mapping is performed greedy and nodes with greater demand are assigned with greater available resources. Then, virtual link mapping stage is performed using paths algebra-based multi-constraint routing algorithm (LADN) [15]. Therefore, Authors only considered the virtual link mapping stage; the node and link mapping are performed separately and in no coordination. Virtual node mapping without considering its relation with virtual link

mapping results in restricted solution space and decreases the overall performance of the embedding. In addition, greedy node mapping and continually choosing the node with more resources may cause an imbalance of stress on substrate nodes which creates congestion and hot spots. Authors in [14] evaluated their proposed strategy in an offline scenario, while in most real-world situations, VNE is needed to be tackled as an online problem. We propose a heuristc approach based on paths algebra to solve the online virtual network embedding problem that performs node and link mapping coordinately in one stage. We aim to show that coordination between node and link mapping stages and, also, considering bandwidth of connected links to the node while mapping the node can significantly both decrease the path costs and increase VNR acceptance ratio.

3. Network Model and Problem Description

Topology of the substrate network is modeled as a directed graph $G^S = (N^S, L^S)$, where N^S is the set of the substrate nodes, and L^S the set of substrate links. There is limited and defined availability to all physical resources (i.e. processing resources and bandwidth). In fact, G^S is not able to host an infinite number of VN requests. $R^S = (R_N^S, R_L^S)$ denotes the available resources in the G^S, where R_N^S and R_L^S denote the available processing resources of substrate node and the available bandwidth of substrate link, respectively. P^s denotes the set of all the simple substrate paths (i.e., no cycle) in the G^S. We made use of the notion of stress to determine the quantity the resource usage of the substrate network. It is important to maintain low stress in all substrate nodes to an efficient use of network resources. The stress of a substrate node n^s at time t, $S_{SN}(t, n^s)$, is defined as the number of virtual nodes assigned to that substrate node n^s which results from Eq. (1):

$$S_{SN}(t, n^s) = S_{SN}(t^-, n^s) + 1 \mid n^s = M_N(n^v), n^v \in N^V \quad (1)$$

where $M_N(n^v)$ denotes virtual node n^v mapped to a substrate node.

After Virtual network requests are modeled as a directed graph $G^V = (N^V, L^V)$, where N^V is the set of virtual nodes, and L^V the set of virtual links. The requested resource in the G^V is denoted by $R^V = (R_N^V, R_L^V)$, where R_N^V and R_L^V denote the processing resources requested by virtual node and the bandwidth requested by virtual link, respectively. The requests are dynamically arrived to the system and are available for a certain time in the network. When a request arrives, the substrate network allocates resources to the VN that satisfy the constraints of the virtual nodes and links. Exiting the accepted request, its allocated resources are released.

The virtual network embedding problem is defined by a mapping MAP: $G^V(N^V, L^V) \rightarrow G^S(N^{S'}, P^{S'})$ from G^V to a subset of G^S, where $N^{S'} \subset N^S$, $P^{S'} \subset P^S$. It can be decomposed into two mapping phases [16]:

- Node mapping that allocates virtual nodes in different substrate nodes to satisfy node resource constraints,
- Link mapping that maps virtual links to loop-free paths on the substrate, which satisfy the link resource demands.

4. The Proposed Heuristic to Embed Virtual Network Request

4.1 Objectives

The present study aimed to introduce a new approach that not only maximizes the acceptance ratio of the VN request, but also decreases the cost of the serving to the revenue of serving ratio.

The revenue of serving a VN request in time t is defined as Eq. (2) [2, 5, 6]:

$$R(G^V, t) = \sum_{n^v \in N^V} R_N^V(n^v) + \sum_{l^v \in L^V} R_L^V(l^v) \quad (2)$$

and the cost of serving a VN request in time *t* (i.e. the sum of the total resources in substrate network allocated to the VN) is defined as Eq. (3):

$$C(G^V, t) = \sum_{n^v \in N^V} R_N^V(n^v) + \sum_{l^v \in L^V} \sum_{l^s \in L^S} a(l^s, l^v) \cdot R_L^V(l^v) \quad (3)$$

where $a(l^s, l^v) \in \{0,1\}$ and $a(l^s, l^v) = 1$ if substrate link l^s is allocated to virtual link l^v. If hidden hops, introduced in Botero *et al.* [17], are also considered, their cost in each path must be added to the above relation.

4.2. Steps of the CVNEPA

The steps of the proposed approach called CVNEPA, shown in **Fig. 1**, are as follows:

Step 1: ***Ordering the substrate network paths based on the paths algebra-based multi-constraint routing algorithm***

As we aimed to map each virtual link in the best compliant path, first we ordered existing paths between each pair of substrate nodes based on defined metric(s) in virtual network embedding request. In this step, the paths algebra-based multi-constraint routing algorithm (LADN) [15] is used to order substrate network paths. Paths algebra could be used to perform the virtual link mapping in a multi-constraint basis, that is, virtual link can be mapped to substrate paths characterized by infinite constraints (or combination of constraints). Metrics, syntheses and ordering relations employed should be determined before performing paths algebra. Considering the objectives of the present study, the defined routing metrics are:

$$M = \{Hops, BW, CPU\} \quad , F = M \quad (4)$$

The number of hops metric (Hops) is used to order the enumerate paths. The physical constraints of bandwidth (BW) and CPU indicate if the achieved mapping is feasible or not (if their values are negative, the mapping is infeasible).

The corresponding syntheses and ordering relations are represented in equations (5) and (6), respectively:

$$S = \{add, min, min\} \quad (5)$$

The Eq. (5) indicates that the number of hops along the path should be added to evaluate path length, and minimum available bandwidth and CPU capacity of links along the path are needed to evaluate available bandwidth and CPU capacities along the path, respectively.

$$\preccurlyeq_{ML} = \{\geq, \leq, \leq\} \quad (6)$$

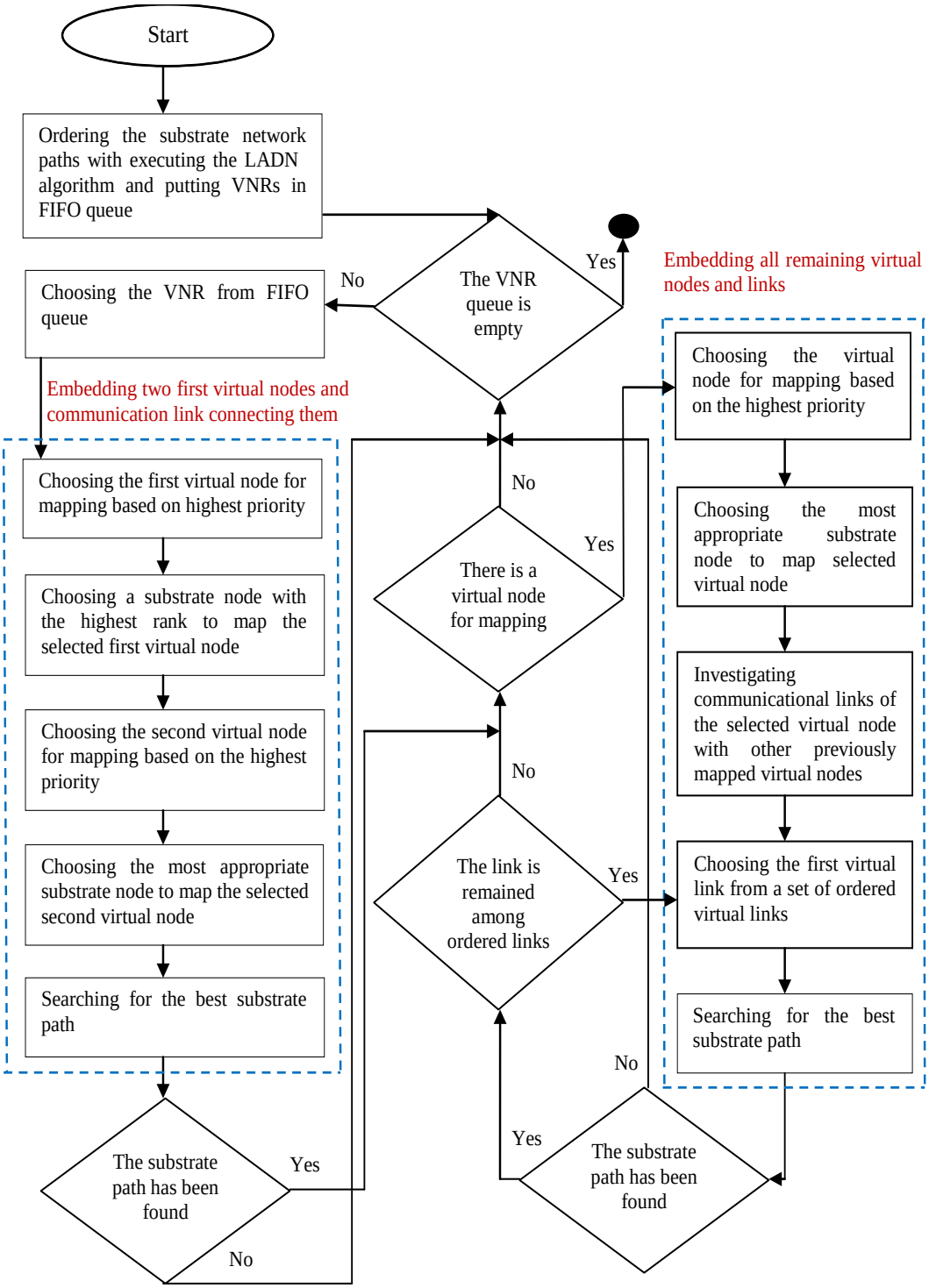


Fig. 1. Steps of proposed CVNEPA approach
(where LADN, VNR and FIFO shows the paths algebra-based multi-constraint routing algorithm, virtual network request and first-in first-out queue, respectively)

The Eq. (6) indicates a more appropriate path is a path including fewer hops ($\geq$ ordering relation), higher bandwidth ($\leq$ relation) and higher CPU capacity ($\leq$ relation).

After performing the LADN algorithm, virtual network requests from FIFO queue are chosen to start mapping. As long as there is virtual network embedding request in queue, following steps are taken:

Step 2: ***Choosing the first virtual node for mapping based on the highest priority***

As it is difficult to map the virtual node with the largest degrees, first virtual node with the largest degrees is selected for mapping. Then its requested bandwidth is calculated. The amount of requested bandwidth is the sum of requested bandwidth by each virtual link connected to the virtual node. If the degree is equal, the virtual node with the highest requested bandwidth is selected. The first chosen virtual node with the highest priority is denoted by n^v.

Step 3: ***Choosing a substrate node with the highest rank to map the selected first virtual node***

All eligible substrate nodes, which can satisfy processing resources and bandwidth requested by selected virtual node from step 2, are nominated and the substrate node with highest rank is selected among them. If the appropriate substrate node is available, virtual node is mapped to it and substrate node resources are updated. Otherwise, virtual network request is rejected. The first chosen substrate node with the highest rank is denoted by n^s.

Ranking the substrate nodes, we follow the below proposed instruction:

We define notion of substrate node rank to calculate resource availability of a substrate node. The resource of a substrate node is defined as an available bandwidth of that substrate node which equals to the sum of available bandwidth in each link connected to that substrate node. As a substrate node having neighbors with more resources shows higher chance to achieve a successful mapping, this feature is considered as a parameter to calculate substrate node rank. $NB(x^s)$ denotes available bandwidth among neighbors to substrate node x^s:

$$NB(x^s) = \sum_{y^s \in Ne(x^s)} \sum_{l \in L(y^s)} BW(l) \tag{7}$$

where $Ne(x^s)$ is the set of neighbors to substrate node x^s, $L(y^s)$ is a set of links connected to substrate node y^s and *BW(l)* is the unoccupied bandwidth of link *l*. Here, only the effect of node's single hop neighbors is considered.

On the other hand, continually choosing the substrate node with more resources may cause in imbalance of stress on substrate nodes which in turn may cause congestion and hot spots. As the result, substrate node stress can be considered as an effective factor to determine substrate node rank. Recall that, the substrate node stress is calculated based on Eq. 1. Considering all said above, the following equation is proposed to determine substrate node rank:

$$NR(x^s) = \left(\sum_{l \in L(x^s)} BW(l) + NB(x^s)\right) / (S_{SN}(t, x^s) + \varepsilon) \tag{8}$$

where ε is a small positive constant to avoid dividing by zero in computing the substrate node rank. If the ranks are equal, a substrate node is selected randomly among substrate nodes having the same rank. Then, at the end of this step, the virtual node n^v (i.e., the selected first

virtual node) is mapped to the selected substrate node n^s and the resource of substrate node n^s is updated.

Step4: ***Choosing the second virtual node for mapping based on the highest priority***

The second virtual node n^v with the largest number of neighboring virtual nodes that are successfully mapped to substrate nodes previously, is selected. It insures that the second virtual node n^v is mapped to a substrate node that is the closest one to substrate nodes mapped previously. If the numbers of neighbors are equal, the virtual node with higher requested bandwidth is selected. Let M be a set of substrate nodes which are successfully mapped by the neighboring virtual nodes of n^v.

Step 5: ***Choosing the most appropriate substrate nodes to map the selected second virtual node***

All eligible substrate nodes, which can satisfy processing resources and bandwidth requested by selected virtual node from step 4, are nominated. For each selected eligible substrate node x^s the shortest path (i.e. a path with fewest hops) from x^s to each substrate node y^s in M is computed and the maximum length of these computed shortest paths is determined as the shortest distance between x^s and M. The length of shortest path between substrate nodes x^s and y^s, and the maximum distance (based on hop count) from substrate node x^s to M are denoted by $d^S: N^S \times N^S \to R$, $d^S(x^s, y^s)$ and $md^S: N^S \times 2^{N^S} \to R$, $md^S(x^s, M)$, respectively. $md^S(x^s, M)$ is defined as Eq. 9:

$$md^S(x^s, M) = max\{d^S(x^s, y^s) | \, y^s \in M\} \tag{9}$$

The substrate node with the highest rank among eligible substrate nodes with the minimum value of $md^S(x^s, M)$ is selected. In this step, substrate node rank is determined as Eq.10:

$$NR_d(x^s) = 1/(md^S(x^s, M) + S_{SN}(t, x^s)) \tag{10}$$

Here, considering node stress as a factor in substrate node ranking prevents usual selection of shorter paths, as such permanent selection causes imbalance stress on substrate nodes. On the other hand, it causes full assignment of these substrate node resources while resources of other parts of SN are still available. Full assignment of path resources spoils the possibility to choose paths as a part of longer paths among farther substrate nodes. It could result in fragmentation of SN's resources.

If the ranks of substrate nodes are equal, the substrate node with the most $\left(\sum_{l \in L(x^s)} BW(l) + NB(x^s)\right)$ is selected. The reason to the choice is that the substrate node, which its neighbors and it have connected links with more bandwidth, has a higher chance to achieve a successful mapping. At the end of this step, the substrate node with highest rank is chosen.

Step 6: ***Searching for the best substrate path***

Substrate node with the highest rank, which is selected in step 5, is considered along with the selected substrate node in step 3 to find the best substrate path. If the best substrate path is not found, virtual network request is rejected.

To find the best substrate path:

1. First all available paths between two selected nodes are considered.

2. The directed graph links' metrics (including bandwidth and CPU) are updated by subtracting the CPU capacity and bandwidth demands. Then the paths are synthesized and lexically ordered based on determined metrics and finally, the highest priority path is selected.
3. If a path is found, virtual nodes mapping and virtual link mapping to the selected path is performed. If the path is not found, the result is announced.

Finishing these steps, two virtual nodes with the highest priority and the connection links between them are mapped to the most appropriate substrate nodes and paths. If there are other virtual nodes and links in the virtual network embedding request, steps 7-11 are proposed. Otherwise, until there is a virtual network embedding request in FIFO queue, the proposed strategy repeats from step 2. If the queue is empty, the procedure stops and the results are recorded.

Mapping the two primary virtual nodes, each next selected virtual node could connect to each of the previously mapped virtual nodes. Therefore, a substrate node should be selected to map the virtual node which can satisfy links requirements. Hence, we separate relative steps to mapping such nodes from previous steps of proposed approach. As the result, after the embedding of two primary virtual nodes, we repeat the steps 7-11 until there is no other virtual node in virtual network request:

Step 7: ***Choosing the virtual node for mapping based on the highest priority***

The same procedure of step 4 is used here to choose the next virtual node.

Step 8: ***Choosing the most appropriate substrate node to map selected virtual node***

The same procedure of step 5 is used here to choose the appropriate substrate node (i.e., the substrate node with the highest rank).

Step 9: ***Investigating communicational links of the selected virtual node with other previously mapped virtual nodes***

In this step, first all links between virtual nodes selected in step 7 and previously mapped virtual nodes should be mapped. To do this, links are ordered based on requested bandwidth in descending order. At the end of this step, we have a set of ordered virtual links.

Step 10: ***Choosing the first virtual link from a set of ordered virtual links***

The first link is selected among ordered virtual links in step 9; the communicational link is eliminated from the set of ordered virtual links. The substrate node, which was selected as the host to the terminal node of the virtual link, is selected with a substrate node which is the host to other endpoint of this link.

Step 11: ***Searching for the best substrate path***

The host substrate node to the terminal node in virtual link selected in step 10 and the substrate node with the first rank achieved in step 8 is considered to look for the best substrate path. The procedure at this step is similar to step 6. If the best substrate path is found, virtual node and link are mapped and resources are updated.

Steps 10 and 11 continue until all ordered links in step 9 are mapped. If the algorithm is unsuccessful to find substrate path for each ordered link, virtual network request is rejected.

4.3 Computational time complexity of the CVNEPA

According to [14], the process of finding all the possible paths between any pair of nodes shows an increasing complexity for highly connected networks (not true for small and sparse networks, what is usually the case). Thus, considering big networks, the computational time complexity of the proposed CVNEPA strategy is dominated by the operations of its first step, in which all paths connecting all pairs of nodes in the SN should be discovered. To overcome the complexity and unaffordable execution time of the proposed algorithm besides not sacrificing the quality of the results for big networks, similar to [14] some modifications are introduced into the paths algebra-based multi-constraint routing algorithm to provide control over the number of operations performed by the algorithm and enabling the possibility to work with large networks (more details are described in [14]). In addition, the procedure of finding all paths connecting all pairs of nodes in the SN is done only once and can be performed off-line (before the embedding process starts). The other steps of the proposed CVNEPA strategy can be carried out in polynomial time. Thus, CVNEPA can be solved in polynomial time.

5. Simulation

ALEVIN simulation environment is used [18]. ALEVIN is a framework to develop, compare and analyze virtual network embedding algorithms. LADN algorithm which is adopted to solve the VNE problem (i.e., map virtual links to physical paths in a substrate network) is implemented in ALEVIN simulator. The Waxman algorithm is used to create both the substrate networks and virtual networks. We refer the readers to [14] for more information.

5.1 Simulation Parameters

Table 1 shows the parameters used in the simulations. In **Table 1**, a hidden hop addresses the intermediate node of a directed path in the SN that is mapping a specific virtual link of a VNR. Hidden hops entail a resource demand for forwarding the traffic that will pass through it and will consume CPU resources of the substrate network [14].

The substrate network will be equipped with resources by uniformly distributing each node resource X in a given interval $(0, NR_X^{max})$ for every substrate node (the interval for CPU resources is (0,100]) as well as each link resource Y in a given interval $(0, NR_Y^{max})$ for every substrate link (the interval for BW resources is (0,100]). BW and CPU, in the context of the VNE problem, are specified in terms of capacity units. In our work, CPU and bandwidth (BW) capacity units are scaled in a way that 100 CPU units = 2.66 GHz and 100 BW units = 1Gbps.

The arrival pattern of requests follows a Poisson process, where average arrival rate is 5 VNs per 100 time units. Lifetime of the requests is an exponential distribution with an average of μ =500 time units. As the Waxman topology generation is probabilistic, we perform N=20 runs for each value of load.

Table 1. Parameters chosen for the simulation scenarios [14].

Parameter description		Chosen values	Parameter description	Chosen values
Number of substrate nodes (|V|)		20	Number of VNRs (k)	10
Number of virtual nodes per virtual network (|V^k|)		10	Maximum CPU cycles in the substrate network ($\boldsymbol{NR_X^{max}}$)	100
Maximum bandwidth in the substrate network ($\boldsymbol{LR_Y^{max}}$)		100	Cost of hidden hops	0
loads' range	Low	{0.1, 0.2, 0.3}		
	Medium	{0.4, 0.5, 0.6}		
	High	{0.7, 0.8}		

5.2 Simulation Results

As proposed strategy in [14] made used of paths algebra for the virtual link mapping stage, it could be an appropriate option to compare the results and evaluate proposed approach CVNEPA. Proposed strategy in [14] is evaluated in an offline scenario; though, for logical comparison of CVNEPA and [14] approaches, we analyzed both scenarios online. From now and for the ease of reading the proposed strategy in [14] is named as PA-ViNE.

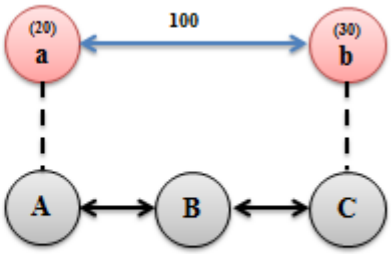
Results of the simulation were evaluated based on VNR acceptance ratio, mapped revenue ratio and Cost/Revenue (C/R) relationship criteria. While VNR acceptance ratio metric only considers the number of accepted VNRs without considering variations in VNR size, the mapped revenue ratio metric only considers ratio of accepted virtual resources (i.e., CPU and BW) without considering its VNRs. If an algorithm leads to SN's resource fragmentation, it may accept VNRs with small size and reject VNRs with large size that results in the probability of having high VNR acceptance ratio and low mapped revenue ratio are increased. By accepting VNRs with large size, the revenue of the algorithm will grow faster but the VNR acceptance ratio will be reduced. Depending on the two following optimization criteria in names the number of satisfied VNRs and mapped revenue ratio, the **Fig.s 2-a** and **2-b** are useful, respectively.

In order to understand the obtained results for C/R it is necessary to consider how Cost and Revenue are evaluated. **Fig. 2** shows an example of how evaluation of C/R is carried out. Considering a VN request (i.e., VNR) characterizaed by:

- virtual source node: a
- virtual destination node: b
- requested bandwidth: BW(a–b) = 100
- requested CPU: CPU(a) = 20; CPU(b) = 30

The associated revenue (i.e., R) is given by R = CPU(a) + BW(a–b) + CPU(b) = 20+100 +30 = 150. Let assume that the paths algebra algorithm finds one possible mapping summarized in **Fig. 2**. It can be seen that the best solution is achieved when a virtual link is directly mapped on a single substrate link. In this case, $C/R = 1.0$. Each time a hidden node has to be used, the cost increases by the cost of the CPU plus the cost of the bandwidth. As in this example the cost of the hidden nodes' CPU is taken equal to zero then for a mapping that uses k hidden nodes the cost(i.e., C) is given by C = CPU(a) + CPU(b) + (1+k)BW. As the revenue for a given VNR is a fixed number, then $C/R = 1 + k\,\frac{BW}{R}$, i.e., C/R increases linearly with increasing the number of hidden nodes.

Node mapping	a → A , b → C
Path	A;B;C
Cost	CPU(A)+CPU(B) + CPU(C) + BW(A–B)+BW(B-C) = 20 + 0 + 30 +100+100 = 250
Revenue	CPU(a) + CPU(b) + BW(a–b) = 20 + 30 + 100 = 150
C/R	250/150 = 1.67



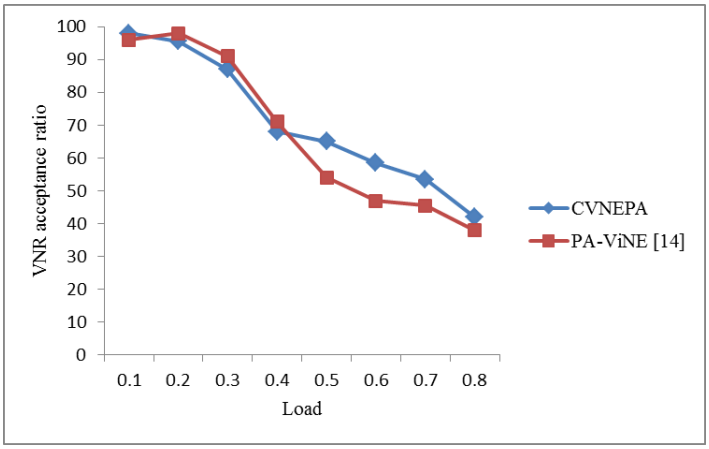
Note: CPU(B) = 0 because the cost of hidden hops is note being considered

Fig. 2. An example of how to evaluate Cost/Revenue

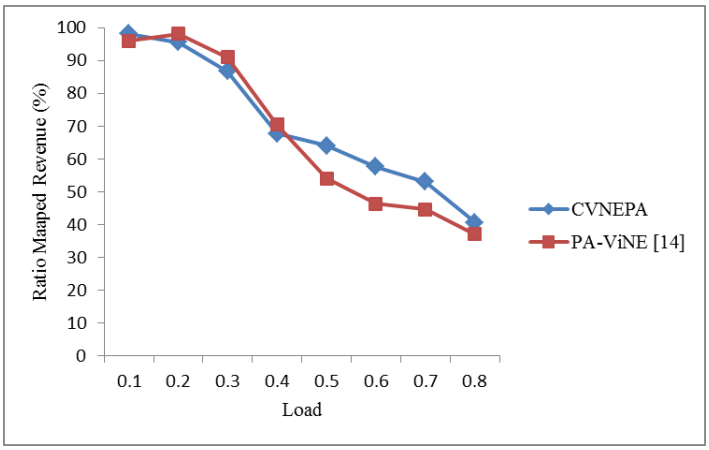
As is mentioned in [14], "the dimension of the sample space does not allow deriving quantitative conclusions with statistical significance but it is large enough to provide qualitative comparisons and obtain an understanding of the observed behaviors".

Fig. 3 shows the results of comparison between the new approach CVNEPA and PA-ViNE [14]. **Fig. 3-(a)** shows when the load ρ is low, as the ratio of request to delivery is small, both approaches seem to perform well and acceptance ratio is high. The more the request, the less the capacity of substrate network to accept the requests.

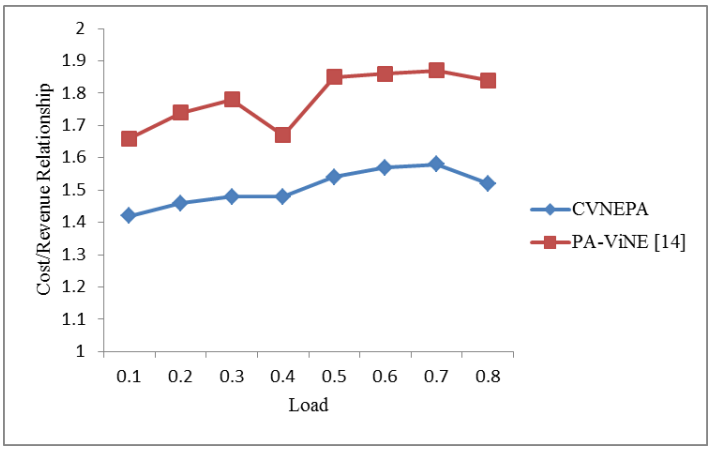
As seen in the figure, CVNEPA approach has partly increased requests acceptance ratio. Due to coordinated mapping of virtual node and link, embedding cost (C) for each request decreases and as the result, there will be higher network capacity to accept the requests. In addition, embedding in one stage and mapping the virtual link immediately after mapping its end nodes leads to faster discovery of lack of enough resources to embed current request, asvirtual nodes with more requested resources (including CPU and sum of bandwidth of connected links to the node) first are mapped and immediately the link between them is mapped. The early recognition helps to decrease request waiting time in queue through rejecting the current request, while more requests are investigated in a given time and there is a higher chance to embed more requests to infrastructure provider.



a. VNR acceptance ratio



b. Ratio mapped revenue



c. Cost/revenue (C/R)

Fig. 3. The comparison between the proposed approach CVNEPA and virtual link mapping strategy based on paths algebra (in name PA-ViNE) [14].

As the revenue is in proportional to the number of accepted VNRs, the ratio mapped revenue shows the same behavior of accepted VNRs ratio, shown in **Fig. 3-(b)**.

Both approaches studied here considered minimizing the costs as a priority, which means they both aimed to map a VNR using the shortest available path; the shortest paths are used first. When the load increases, longer path have to be used to satisfy the VNR requests, while it increases the cost. Therefore, it is expected that C/R shows a constant behavior or increases slowly with load increase. **Fig. 3-(c)** summarizes the results obtained for Cost/revenue (C/R) relationship. Recall that, the best solution is achieved when a virtual link is directly mapped on a single substrate link. In this case, $C/R = 1.0$. Also the value of the worst experienced C/R is smaller than the numerical value 2. Thus, the numerical values of y-axis are bounded between 1 to 2. The obtained results in agreement with expected results and CVNEPA showed better performance in comparison to PA-ViNE [14]. As more substrate network resources must be allocated to the requests, the more the rate of requested resource (or load), the higher the cost. Two main reasons for improving the performance of CVNEPA approach are: first, each substrate node selected to be mapped should have the smallest load stress and be the closest

node to previously mapped substrate nodes. It decreases the number of intermediate hops and, as the result, the cost. The other reason is that in step 5, if the ranks of substrate nodes are equal (according to Eq. 10), the proposed node mapping algorithm selects the substrate node which has neighbors with more bandwidth. This increases the probability of selecting the node's single hop neighbors in the next steps and, thus, will decrease the length of the substrate path.

Although proposed approach CVNEPA has not greatly improved requests acceptance ratio, but it is more powerful, since it has decreased the cost of serving and has tried to balance stress on the substrate nodes. Offering a solution to improve acceptance ratio of VN requests is left for future work.

6. Conclusion

The present study aimed to introduce a heuristic approach called CVNEPA to solve the virtual network embedding problem in cloud. In proposed approach, we performed the virtual node and link mapping stages coordinately, and in one stage while supporting multiple criteria thanks to the paths algebra routing framework. The advantage of proposed approach was that all eligible nodes and paths were ordered. Therefore, not only the best solutions were introduced but also all possible solutions could be accessible to. This feature is valuable to implement additional strategies such as survivability or planning.

Simulation results indicated that proposed approach made more efficient use of cloud resources, increased revenue of cloud provider and improved acceptance ratio of VN requests by decrease cost of serving.

References

[1] A. Khan, A. Zugenmaier, D. Jurca, and W. Kellerer, "Network virtualization: a hypervisor for the Internet?," *IEEE Communications Magazine*, vol. 50, no.1, pp. 136-143, January, 2012. Article (CrossRef Link)

[2] X. Cheng, S. Su, Z. Zhang, K. Shuang, F. Yang, Y. Luo, et al., "Virtual network embedding through topology awareness and optimization," *Computer Networks*, vol. 56, no. 6, pp. 1797-1813, April, 2012. Article (CrossRef Link)

[3] Y. Zhu and M. Ammar, "Algorithms for Assigning Substrate Network Resources to Virtual Network Components," in *Proc. of 25th* International Conference *on Computer Communications (INFOCOM)*, pp. 1-12, April 23-29, 2006. Article (CrossRef Link)

[4] J. Lu and J. Turner, "Efficient mapping of virtual networks onto a shared substrate," *Tech. Rep. WUCSE-2006-35*, Washington University in St. Louis, January, 2006. Article (CrossRef Link)

[5] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking virtual network embedding: Substrate support for path splitting and migration," *ACM SIGCOMM CCR*, vol. 38, no. 2, pp. 17–29, April, 2008. Article (CrossRef Link)

[6] N. M. M. K. Chowdhury, M. R. Rahman, and R. Boutaba, "Virtual Network Embedding with Coordinated Node and Link Mapping," in *Proc. of IEEE INFOCOM 2009*, pp. 783-791, April 19-25, 2009. Article (CrossRef Link)

[7] J. Lischka and H. Karl, "A virtual network mapping algorithm based on subgraph isomorphism detection," in *Proc. of the 1st ACM workshop on Virtualized infrastructure systems and architectures*, pp. 81-88, August, 2009. Article (CrossRef Link)

[8] L. Gong, Y. Wen, Z. Zhu and T. Lee, "Toward profit-seeking virtual network embedding algorithm via global resource capacity," in *Proc. of IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, pp. 1-9, April, 2014. Article (CrossRef Link)

[9] J. Ding, T. Huang, J. Liu, and Y. Liu, "Virtual network embedding based on real-time topological attributes," *Frontiers of Information Technology & Electronic Engineering*, vol. 16, no. 2, pp. 109-118, February, 2015. Article (CrossRef Link)
[10] T. Wang and M. Hamdi, "Presto: Towards efficient online virtual network embedding in virtualized cloud data centers," *Computer Networks*, vol. 106, pp. 196–208, September, 2016. Article (CrossRef Link)
[11] P. Zhang, H. Yao and Y. Liu, "Virtual Network Embedding Based on the Degree and Clustering Coefficient Information," *IEEE Access*, vol. 4, pp. 8572 – 8580, January, 2017. Article (CrossRef Link)
[12] I. Fajjari, N. Aitsaadi, G. Pujolle, and H. Zimmermann, "VNE-AC: Virtual Network Embedding Algorithm Based on Ant Colony Metaheuristic," in *Proc. of IEEE International Conference on Communications (ICC)*, pp. 1-6, July , 2011. Article (CrossRef Link)
[13] Z. Zhang, X. Cheng, S. Su, Y. Wang, K. Shuang, and Y. Luo, "A unified enhanced particle swarm optimization-based virtual network embedding algorithm," *International Journal of Communication Systems*, vol. 26, no. 8, pp. 1054-1073, August, 2013. Article (CrossRef Link)
[14] J. F. Botero, M. Molina, X. Hesselbach-Serra, and J. R. Amazonas, "A novel paths algebra-based strategy to flexibly solve the link mapping stage of VNE problems," *Journal of Network and Computer Applications*, vol. 36, no. 6, pp. 1735-1752, November , 2013. Article (CrossRef Link)
[15] W. de Paula Herman and J. R. de Almeida Amazonas, "Hop-by-hop Routing Convergence Analysis Based on Paths Algebra," in *Proc. of Electronics, Robotics and Automotive Mechanics Conference (CERMA* 2007), pp. 9-14, Sept. 25-28, 2007. Article (CrossRef Link)
[16] A. Fischer, J. F. Botero, M. Till Beck, H. de Meer, and X. Hesselbach, "Virtual Network Embedding: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 1888-1906, February, 2013. Article (CrossRef Link)
[17] J. F. Botero, X. Hesselbach, A. Fischer, and H. De Meer, "Optimal mapping of virtual networks with hidden hops," *Telecommunication Systems*, vol. 51, no. 4, pp. 273-282, December, 2012. Article (CrossRef Link)
[18] A. Fischer, J. F. Botero, M. Duelli, D. Schlosser, X. Hesselbach, and H. de Meer, "ALEVIN- a framework to develop, compare, and analyze virtual network embedding algorithms," *Electronic Communications of the EASST*, vol. 37, pp 1-12, 2011. Article (CrossRef Link)

Nahid Hamzehee Nia received the B.S degree in Software Engineering from Payame Noor University Kermanshah Branch, Iran, and the M.S. degree in Computer Engineering from Islamic Azad University North Tehran Branch, Iran, in 2010 and 2014, respectively. Her research interests include Cloud Computing, Distributed Systems and Resource Management.

Sepideh Adabi received the BS, MS and PhD degrees in software engineering from Department of computer Engineering, Islamic Azad University in 2006, 2008 and 2012 respectively. She is currently an assistant professor in the Department of Computer Engineering at Islamic Azad University in Tehran, Iran. Her research interests include distributed systems, grid computing, cloud computing and resource management. She is a member of the IEEE.

Majid Nikougoftar Nategh received the B.S degree in Information Technology from Payame Noor University Qom Branch, Iran, and the M.S. degree in Information Technology from University of Qom, Iran, in 2011 and 2014, respectively. His research interests include cloud computing, data mining, analysis of large-scale data sets and mining patterns from high dimensional databases.