International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 12, No.1 (Jul-2022)

https://dx.doi.org/10.12785/ijcds/120129

A Cluster-Oriented Policy for Virtual Network Embedding in SDN-Enabled Distributed Cloud

Abderrahim Bouchair[1], **Belabbas Yagoubi**[1,2] **and Sid Ahmed Makhlouf**[1]

[1]*L.I.O. Laboratory, Department of Computer Science, University of Oran1 Ahmed Ben Bella, BP 1524 EL Mnaouer Oran, Algeria*
[2]*University of Mostaganem, Mostaganem, Algeria*

Received 30 Sep. 2021, Revised 6 Apr. 2022, Accepted 24 Jun. 2022, Published 20 Jul. 2022

Abstract: Resource allocation is a crucial challenge for network virtualization (NV) in a cloud environment. Virtual network embedding (VNE) approaches exemplify NV technologies' critical utility, which must efficiently deal with potential network issues. To promote cloud infrastructure flexibility, software-defined networking (SDN) has been adopted as a network practice to centralize the manageability of the data centre network (DCN) resources. This paper introduces a classification approach that ensures an accurate starting point for solving the VNE problem in a distributed system. The solution implementation is based on measuring the importance of each DCN using the spearman rank correlation coefficient. Afterward, we devise a constructive algorithm that classifies DCNs in clusters from unsupervised data learning. This DCN management allows us to direct the VNE process to a small number of DCNs, which will reduce the dimensionality of the search operation in a distributed environment. Ultimately, we adopt various metaheuristics as a VNE optimizer for the selected DCN. Numerical results verify that the Jenks natural breaks classification outperforms similar methods in terms of resource utilization and acceptance ratio.

Keywords: Cloud Computingt, Virtual Network Embedding, Data Center Network, Software Defined Networking, Classification

1. Introduction

Resource management remains one of the biggest challenges for cloud technologies, including operational efficiency and business flexibility [1]. Specifically, the traditional infrastructure provider (InP) was the main entity responsible for the service deployment (mostly creating virtual resources) and physical network resource management within an NV application. Besides that, service providers (SPs) have the role of synthesizing the deployed virtual resource in virtual networks (VNs).

Over the last decade, cloud computing's evolution has taken a substantial step towards a sustainable reckoning paradigm that entails a distributed accessing of services [2]. SDN is one of the most effective network technology that has been notably endorsed within cloud facilities, and this is stemmed from the gained benefits in architecture adaptability at a minimal cost. According to the open networking foundation (ONF) [3], the SDN concept establishes a thorough process of separating the forwarding function (InP tasks) from the network managing servers (SP tasks).

In networking, SDN establishes an orchestration design to centralize network provision at lower operating charges [4]. The modern concept of SDN-enabled focuses on offering innovative deployments for VNE policies, including resource allocation and utilization. The goal of VNE is to locate the ideal mapping approach from a set of virtual network requests (VNRs) to a substrate network (SN) [5]. So far in the literature, three different optimizing strategies were used to improve the performance of VNE:

- Exact approaches [6] address the small instance of the initial problem and set up a formulated solution that usually depends on an Integer Linear Programming (ILP) model.
- Heuristic methods [7] tend to solve the VNE problem in a low execution time while counting on the exact solution as a baseline strategy and extend it to large size problems.
- Metaheuristic methods are considered the most commonly exploited methods due to their suitability with real-world cloud scenarios, where near-optimal solutions are generated within a reasonable period.

When the cloud infrastructure is based on distributed datacenters, the perspective of solving a VNE problem is derived from an initial plan of choosing the appropriate DCN to meet the VNRs requirements; ergo, a higher level

E-mail address: bouchair.abderrahim@edu.univ-oran1.dz, byagoubi31@gmail.com, sidahmed.makhlouf@gmail.com of governance is potentially required. In this study, we take advantage of the SDN features such as programmability and controllability to apply a cluster-based method as a pre-embedding solution. The challenge of clustering-based DCN selection aims to determine a DCN cluster that can accommodate a maximum number of VNRs [8]. The main problem with DCN selection is identifying the resemblance property in which a collection of DCNs can provide the best possible physical resources in one group. The DCN clustering strengthens the pursuit of reducing resource usage and increasing acceptance ratio. Furthermore, because of the distributed environment, essential questions must be asked. Which DCN is the best to fulfill the VNs requirements? How to reduce the optimization runtime while solving the VNE problem? To tackle these problematics, we present the following main contributions:

- A two-part approach is proposed to solve the VNE problem. Substantially, it executes a data clustering method for DCNs classification, following with a metaheuristic-based embedding procedure associated with the physical and virtual properties.
- Adopt a topological information structure (logical heap) to classify the DCN clusters and speed up the search operations for optimal SNs. This structure-driven DCNs partitioning attempt to classify the DCNs based on a correlation analysis of their capacity and relative availability.
- Abstract the SDN control plane's impact in the multi-datacenter cloud by focusing fundamentally on the monitoring aspect to efficiently conduct the meta-heuristic solving process to the right DCN.
- Several comparisons of well-known metaheuristics have been conducted to solve the VNE problem. The results show the promising improvements of the cluster ranking correlation in increasing the VNRs acceptance ratio and while lowering the embedding cost.

This paper is arranged as follows: the most corresponding works to our approach are presented in Section 2. In Section 3, we explain the essential incentives behind our research. Next, Section 4 depicts the VNE problem within an SDN-enabled distributed cloud. Section 5 outlines the problem statement. A description of our solution proposal is discussed in Section 6. In Section 7, sufficient experiments are conducted to evaluate the effectiveness of the DCN-clustering method. Finally, we conclude in Section 8.

2. Related Work

Although recent research has addressed the consolidation of SDN technology in the cloud [9], little attention has been given to solving the VNE problem in a multi-datacenter cloud. Alaluna et al [10] present a novel approach for NV to avoid a cloud with a single datacenter. The proposed VNE technique (SecVNE) is formulated using a Mixed Integer Linear Program (MILP) to provide security assurance over a multi-cloud deployment. The authors have implemented a multi-cloud network virtualization platform called Sirius, which leverages the SDN structure to build the DCNs in private and public clouds. Furthermore, the data-paths in the data plane are configured via the SDN controller in the switches' forwarding rules. In [11], the authors proposed a VNE algorithm that maximizes the VNR acceptance ratio in a distributed cloud. This cloud revenue improvement is based on a novel metric that can model cloud resources' dynamic workloads. It provides accurate knowledge of periodic resource demands to the embedding algorithm by combining shared resources with a complimentary resource utilization ratio. The performance evaluation shows that the proposed policy approach outperforms similar heuristics by scoring a higher revenue by 31%.

Alhazmi et al [12] studied the resource provisioning for VNE in geographically distributed cloud datacenters with SDN-based VN management. The proposed method (OVNP) is formulated using the MILP model to optimize the VN provisioning, relying on the SDN controller's full awareness of the network's physical resources across datacenters. This provisioning procedure allows dealing with the VNE problem in an online fashion (i.e. no advanced information about the VNRs is available for the SDN controller) by integrating a flow-metrics-based approach to conducting the VNRs to the suitable SN in a single stage. The authors evaluated the proposed work against similar approaches; the OVNP approach showed improved results in terms of profits, computational resource utilization and the ratio of provisioned VNRs.

Alzahrani and Shahin [13] proposed an energy-aware VNE approach (EA-VNE) for the multi-datacenters cloud. It adopts the Particle Swarm Optimization (PSO) algorithm to reduce energy consumption and embedding costs. The proposed system aims to deal with a distributed environment by generating a coarsened graph by partition each VNR into sub-graphs using Heavy Clique Matching (HCM) technique. Each node in the coarsened graph is allocated to the proper datacenter while initiating a modified PSO to find the nearest optimal result for the VNE problem. The EA-VNE approach was tested against various algorithms, which results in observing a higher acceptance ratio up to 70% scored by EA-VNE along with 9% less energy consumption compared with the other works. Xin et al [14] present a request portioning heuristic within a networked cloud. It proceeds by partitioning the VNR into k connected nodes (subgraph isomorphism) using the minimum k-cut algorithm. An initial cost function was proposed to balance the VNR loads across multiple cloud sites. The authors studied the VNE problem in a multi-provider cloud environment where they classified the VNR into bound and unbound. This classification supports the selection of a suitable InP while assuring a cost-effective embedding.

Several VNE approaches have employed the concept of clustering in a distributed environment to provide a proprietary picture to networks where similar resources are grouped in a single cluster. In [15], the authors proposed a VN splitting method based on a max-flow/min-cut theory in a multi-InP environment. This approach creates first a binary tree of the InPs to reduce the splitting complexity (two-domain splitting only) using a partitioning method based on probability. Following this, a capacity network constructing mechanism is presented to convert the two-domain splitting into a max-flow/min-cut problem and solve it via the shortest augmenting path algorithm. The simulation results show an efficiency improvement in VN splitting with a preserved embedding cost scored by the proposed compared versus other VN splitting methods.

Xue et al [16] proposed a method consists of node clustering and dynamic service balance awareness (NC-DSBA) based on a divide-conquer strategy. This approach maintains the quality of service (QoS) of each VNR requirement, where virtual nodes (VNods) have different embedding location preferences. A VNR is divided into several clusters of sub-requests and then dispatched to the nearest local SN, relying on a concurrence policy. The authors implemented a discrete event simulator to evaluate the employed algorithms with other existing methods. The results showed that NC-DSBA has a lower time complexity and a more balanced load distribution. In [17], heuristic and exact algorithms are proposed and combined with conceptual clustering techniques to find the optimal VNR splitting across multiple InPs. In addition, an exact embedding algorithm is presented to ensure that the VNE is performed in a single stage. The VNE problem is formulated as a mixed-integer program (MIP) with the aim of increasing the acceptance ratio while decreasing the provisioning cost for the InPs. The followed splitting and embedding algorithms outperform the existing approaches.

Prior mentioned studies mainly focus on designing a dedicated method for the VNE problem in a distributed environment, targeting only the VNR management including resource splitting and rating. However, in our work, we focus on delivering a VNE model that primary conduct multiple DCNs to enable a feasible policy of physical resource availability. Our VNE approach promote the cloud infrastructure supervision via SDN control plane, which will push a top-notch trusted SNs for the mapping of various VN types while improving VNE objectives.

3. Motivation

At present, a distributed cloud environment appears to be the most adopted real-life scenario versus a single datacenter cloud that tends to be abandoned over the course of the cloud's evolutionary vision. With that been said, SDN structures make it far simpler to build an operative multi-datacenter environment by connecting a cloud provider to a network structure that is already set up to incorporate cloud services. In this work, we focus on optimizing the VNE process as one of the essential types of NV technology offered by the modern cloud utilities. Our VNE model stems from the following motivations:

- An SDN multi-cloud environment allows cloud providers to make seamless decision-making and flexibility through an omniscient controller when managing VNR requirements.
- SDN structure solves many connectivity issues within distributed cloud by providing an automated networking that can eject the exigency of implementing a distributed algorithm. Through a single platform, InPs could easily interconnect datacenter locations globally as they grow their business and reach new customers.
- As many VNs need to be embedded across multiple DCNs, our approach seeks to maintain an affinity relationship between the VNRs and SNs while preserving the network design during the embedding process.
- A straightforward embedding of a whole VNR (no operation has been performed, such as splitting) will firmly uphold the user integrity and avoid redundant computational procedures for each VNR, which will eventually reduce the overall embedding runtime.
- To address the issue of avoiding the redundant cost of VNR management, we attempt to explore the existing DCNs in the interests of determining the most suitable DCN for any VNRs within a VNE framework. Consequently, this DCN management leads to formulating an optimal classification scheme for the data plane.
- Existing metaheuristic approaches provide the best results for a VNE problem with large SNs. But when the problem scale grows to multiple DCNs, these approaches' performance may degenerate due to the curse of dimensionality. Hence, the selected DCN from the exploring policy encloses the search space of metaheuristics in one DCN at a time for each VNR.

This paper introduces a cluster-based VNE method for a distributed cloud. Accordingly, the SDN structure enhances the DCNs management intelligently and automatically. Moreover, the SDN-enabled cloud offers a consistent orchestration to launch a metaheuristic algorithm that rationalizes the matching between the classified DCNs and the requirements of the received VNRs.

4. Virtual Network Embedding in SDN- Enabled Distributed Cloud

Among the cloud service models, Infrastructure as a Service (IaaS) permits the public tenants an opportunity to rent virtualized computing resources in a pay-as-you-go manner, which will accurately meet their exigencies. In this section, we briefly review the SDN structure's effect to cope with the VNE problem in a distributed cloud environment.

A. SDN-Enabled Cloud

Current cloud architectures rely on dole out the shared resources on many DCNs that may be found in different geographic locations. To improve cloud services' reliability, SDN technologies are adopted to promote more innovation in network design and management. In this regard, the deployment of VNs has been undeniably enhanced due to system monitoring that carries out efficiently the mapping process. Fig. 1 depicts an overview of the VNE plan in an SDN-enabled cloud with multiple DCNs.

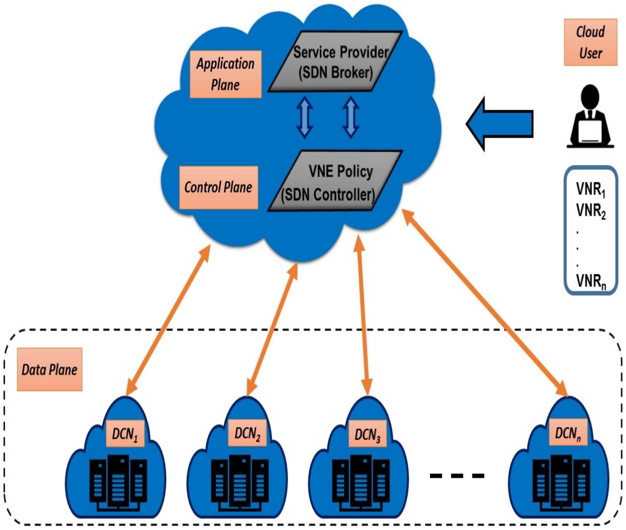

Figure 1. Overview of the SDN-enabled distributed cloud for VNE

A user submits a set of VNRs, and the system plans a low-cost embedding and orchestrates requests to multiple cloud DCNs. From Fig. 1, we can identify three standard SDN layers or plans:

1) *Application Plane*: A platform that describes the service-aware performance while receiving and handling the VNRs by the SDN broker. This layer cannot be remotely programmable to prevent any network cyberattacks. It separates the actual software operation panel, such as a firewall, from the firewall hardware.
2) *Control Plane*: It represents the critical source of network intelligence carried out by a management server noted as an SDN controller. This latter enables policy-based management of network resources. The controller's programmability feature sets out an automatized VNE solution that conducts the given VNR to the most suitable DCN in the data plane.
3) *Data Plane*: It accommodates all the networking equipment, including switches and routers, which handles the data forwarding rules from the SDN controller to a specific DCN.

In real cloud practices, the exchange of information between the three layers is achieved using several application programming interfaces (APIs); designated as northbound interfaces (NBIs) between the control plane and application plane, East-west API within the control plane in case of the existence of multiple SDN controllers; southbound interfaces (SBIs) are the link between the control plane and data plane. NBIs and SBIs are situated in the control plane in which they communicate with higher component via programmable network platforms, and forward data traffic to the data plane via protocols, respectively.

B. Virtual Network Embedding Problem

The process of VNE can be abstracted into an optimal assignment of VNs to an SN subject to resource constraints. In accordance with Fisher et al [5], the VNE optimization (VNEO) problem is related to two main functions, including *node embedding function* (NEF) and *link embedding function* (LEF).

VNEO is defined in Formula (1), where $SN = (SNod, SLin)$ includes a number of substrate nodes (SNods) and links (SLins); $VNR_i = (VNod_i, VLin_i)$ denotes the i-th VNR with VNods and virtual links (VLins).

$$VNEO : \begin{cases} NEF : VNod_i \rightarrow SNod \\ LEF : VLin_i \rightarrow SN' \subseteq SN \end{cases} \qquad (1)$$

The NEF ensures the mapping of every VNod to a particular SNod. Similarly, LEF will map each VLin into a SLin or more. We emphasize that an SN represents a single DCN, which is selected based on a VNE policy.

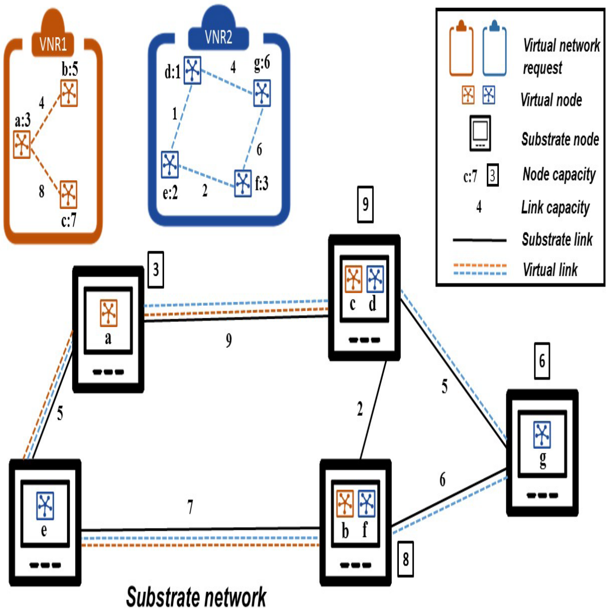

Figure 2. Example of a VNE solution

Fig. 2 shows an illustration of a VNE process that maps a pair of VNRs onto a single SN. A VNE solution typically features virtual resource allocation into a substrate resource where it should be used most economically. Indeed, a successful mapping of virtual resources depends on physical hardware's capacity. We note that an accepted VNR (all virtual resources can be mapped) must be mapped entirely onto one merely SN. A SNod can only host a single VNod from the same VN. Nevertheless, a VLin can be mapped onto multiple SLins, cross different SNods between two VNods, as shown in Fig. 2, between *a* to *b* and *d* to *e*.

5. Problem Statement

In this work, the VNE under study is based essentially on managing the interconnected DCNs. This management consists of classifying the DCNs in labeled clusters. Understanding that the NP-hard problem is a common characteristic for solving a VNE [18] and clustering optimizations [19] will enforce the difficulty of providing an adequate solution in these circumstances.

To address this challenge, a complexity reduction must be applied to one of the involved topics. As the VNE problem remains NP-hard even in reducing the unsplittable flow mapping of VLins into a single substrate path [20], we have only the option of simplifying the *data clustering problem*. Therefore, we focus on the dimensionality aspect of clustering by lowering the variables' number into one variable. This shift initiative relies on the definition of the employed metrics that will assess the similarity of variables. As a result, we deal precisely with a *one-dimensional (1D) partitioning problem* while the results are directed on hierarchical ranking design.

Commonly, a 1D partitioning is a data segmentation that can be done by partition it using eye-tracking (visualize the data distribution through a histogram) in case of a small dataset. However, it is not evident for a big dataset. Hence, many straightforward 1D partitioning methods have a low execution time, such as percentilization, quantiles, discretization, and minimum distance. Besides that, several advanced methods were employed to produce rational solutions (based on specific measurements) including, kernel density estimation (KDE), gaussian mixture model (GMM), bayes information criterion (BIC), expectation-maximization (EM) and k-means based heuristics such as *Ckmeans.1d.dp* [21].

We note that all mentioned methods (simpler and advanced) require initializing a desired number of clusters. In cluster analysis, choosing the right number of groups is a challenging task for users and considered a major drawback of these methods. However, in this work, we deal with a ranked data that allows us to fix a priori several clusters in organized classes. Indeed, we suggest clustering the DCNs into four classes based on their possible state of capacity including *very high*, *high*, *medium* and *low*. Thus, the proposed technique focuses on finding multiple cut-off thresholds in an unsupervised dataset, as we know its possible output. Fig. 3 summarizes the fundamental steps that address the VNE in a SDN-enabled distributed cloud.

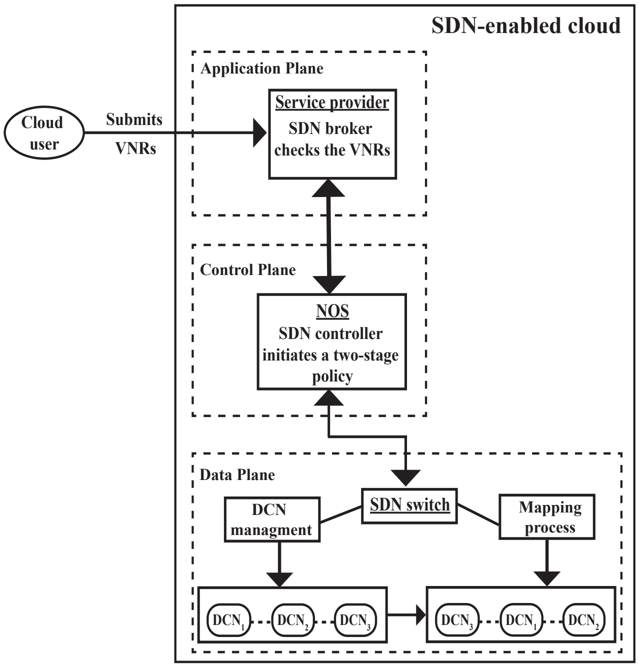

Figure 3. Flow diagram of the proposed two-staged VNE approach

6. Proposed Approach

With the expansion of the multi-datacenter cloud, the rate of the received VNRs has also been highly increased. This positive correlation drove the cloud SPs to establish an operations function in advance that will enhance the decision-making strategies. To diminish the cost management, we follow a throwaway strategy based on two parts:

1) Determine DCN's level of importance.
2) Conduct and apply the metaheuristic-based VNE method to the fittest DCN.

A. *Part One: DCN Management*

Data clustering algorithms rely critically on being given an effective metric over the provided inputs. For instance, the process of data clustering can be done in many tenable ways, and if a clustering algorithm fails to find meaningful clusters to a user, perhaps the only resort is to manually adjust the metric until sufficiently helpful clusters are found. Our main idea is based on two steps, as shown in Fig. 4.

The first step of the DCN management part consists of initiating a preliminary task that creates a sorted array of DCNs according to their importance values, which will be injected as input to the second step called adaptive data clustering. This latter relies on the Jenks natural break optimization(JNBO) [22] to partition and classify the DCNs dynamically, resulting in building a labeled heap.

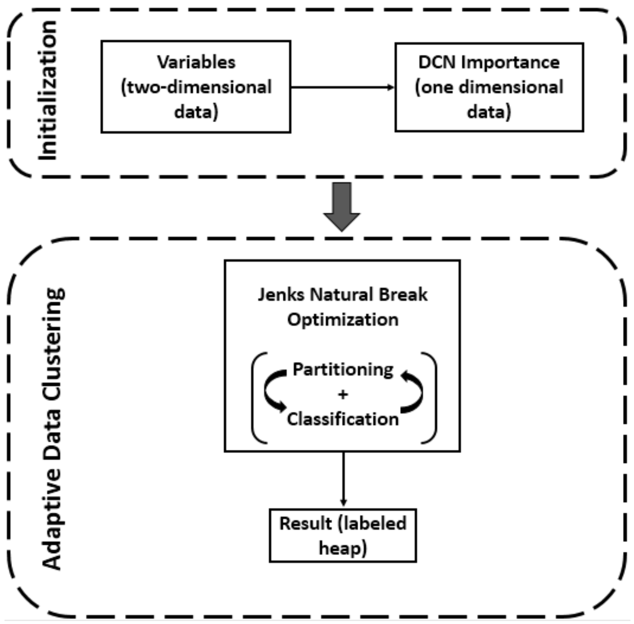
Initialization
Variables (two-dimensional data)
DCN Importance (one dimensional data)
Adaptive Data Clustering
Jenks Natural Break Optimization
Partitioning
+
Classification
Result (labeled heap)

Figure 4. Functional diagram of the overall DCN management

1) DCN importance

In our model, we provide a systematic way to cluster DCNs based on defining variables that are relevant to the VNE settings. Two main criteria must be satisfied for VNRs: processing requirements and the number of VNods. These criteria are often verified for the concerned SN. Accordingly, *capacity* and *availability* are the most common parameters from an operative DCN perspective in which they reinforce the VNE process.

$$DCN^{cap} = \sum_{i=1}^{n} SNod_i^{cap} + \sum DCN^{bw} \tag{2}$$

$$SNod^{cap} = \sum (CPU, RAM, BW, Storage) \tag{3}$$

$$DCN^{av} = \left| \frac{\alpha . \sum SNod^{ocp}}{SNod^{al}} - \beta . SNod^{fr} \right| \tag{4}$$

The DCN capacity (DCN^{cap}) is determined by summing the capacities of the SNods $SNod^{cap}$(each $SNod^{cap}$is determined by summing its local resource capacities) and DCN total bandwidth (DCN^{bw}). In this work, we assume that all the underlying physical resources are running regularly, with no possibility of downtime occurring (DCN is always responsive). Hence, the proposed DCN availability (DCN^{av}) variable is associated with the current state of the total number of SNods ($SNod^{al}$), including the number of occupied SNods ($SNod^{ocp}$) and the number of free SNods ($SNod^{fr}$). The relative factors α and β are used for elevating the values of $SNod^{ocp}$ and $SNod^{fr}$ where α < β. The greater the value of $SNod^{fr}$, the higher DCN is promoted.

Finally, DCN^{cap} and DCN^{av} are the two based variables entrench the importance of a given DCN. The importance attribute is defined through three steps. In the first step, we rank the data of $\text{DCN}^{\text{cap}}$and DCN^{av}, the highest value rank is equal to the data size. The second step consist of applying Spearman's rank correlation coefficient (ρ) in (5) to measure the similarity between all DCNs, which is defined as the following:

$$\rho = 1 - \left(\frac{6. \sum d_i^2}{(n^3 - 1)} \right) \tag{5}$$

Where n is the number of DCNs and d_i is the difference in the *i-th* rank when the values of the two variables are sorted. The value of ρ will always be between 1 (variables are positively correlated) and -1 (variables are negatively correlated). The third step is dedicated to define the importance value of every DCN (DCN^{imp}). This value is determined based on ρ^{all} (correlation coefficient for all DCNs) and ρ^{s} (correlation coefficient for all DCNs except the *k-th* DCN) by calculating the average similarity between the *k-th* DCN and other DCNs. The importance of a given DCN can be calculated as follows:

$$DCN_k^{imp} = \left| \frac{\rho^{all} - \rho^{s}}{(n-1).\rho^{all}} \right|, k = 1,\ 2,\ \ldots,\ n \tag{6}$$

The larger the value of $\text{DCN}_{\text{k}}^{\text{imp}}$ the more important the *k-th* DCN is. This process of defining the DCN importance served us to fulfill our purpose of reducing the dimension of the data clustering problem to one dimension. We note that every value in (6) is added to an array in ascending order.

2) Adaptive data clustering

The term clustering is often employed when facing a multivariate data issue. However, it requires sophisticated methods to generate desirable solutions. In this paper, we used a classification method called *natural breaks*, which is a kind of clustering technique but with a univariate dataset. JNBO applies a simple data-driven partitioning to improve the classification results by discovering hidden patterns in data.

Given an array $\text{A} = \{\text{x}_1, \text{x}_2, \ldots, \text{x}_\text{n}\}$ of positive values that represent the importance values of the DCNs, which are sorted in ascending order $\text{x}_1 < \text{x}_2, \ldots, < \text{x}_\text{n}$ where x_1 can be associated with any DCN (x_1 does not necessarily label the first DCN). The array A satisfies the constraint of the 1D partitioning problem that necessitates a single attribute (DCN importance) with a uniform representation of values. Our objective is to classify the DCNs into four ordinal classes (k) categorized as: *Low*, *Medium*, *High* and *Very High*. The JNBO objective function $\text{Z}:\ \text{A(x)} \rightarrow \text{k}$ is defined as the following:

$$Min\ Z = \sum_{i=1}^{k} SDCM\left(A_i^{sub}\right) \quad (7)$$

Where *SDCM* is the sum of squared deviations for each *subarray* (class) A^{sub} mean. The adopted method is statistically rigorous as it is designed to minimize the within-class variance (reduce the overlapping) while maximizing between-class differences (increase deviation). The iterative algorithm of DCN classification based on JNBO is instructed as follows:

1) Calculate the sum of squared deviations for the *A* mean (SDAM).
2) Divide the ordered data in *A* into four classes (k=4) and calculate the squared deviation of the generated combination of classes (SDCM). This step is repeated to investigate all the possible classes' combination through the splitting tasks.
3) Calculate the goodness of variance fit (GVF) to minimize the SDCM. While the value of GVF is always between 0 (worst fit) and 1 (best fit), the classes' combination with a GVF value closer to 1 will be selected as the optimal solution for *A* partition. GVF is calculated as:

$$GVF = 1 - \left(\frac{SDCM}{SDAM}\right) \quad (8)$$

4) Create a *Max-Heap* hierarchical design where each node represent a clustered class that is labeled by the highest DCN importance value.

We note that the partition solution has kept the ascending order for each class. The logical hierarchy of our partitioning approach is configured as a heap in step 4 (heap by a complete binary tree) where each node represents a cluster of DCNs. This additional step enriches the obtained natural classes by laying out a meaningful labeled-data representation, which will ultimately reduce the search time of substantial substrate resources during the VNE procedure.

Fig. 5 illustrates the DCN classification strategy, which ensures from the start the availability of the best DCN for any VNR received. Since the data plane is structured as Max-Heap topology, our approach consistently pushes the most important DCNs to the top. These DCNs are logically grouped in a cluster labeled by the highest importance value of a DCN. Therefore, the most frequent operation has the complexity of O(1) to identify the prominent DCN and O (log n) in case of reordering the heap clusters during the VNE process.

We note that the DCNs at the same level have strictly higher importance values than those of the DCNs below. The communications between the SDN controller and DCNs are made via four SDN switches (programmable switches), one at each cluster. Practically, a real cloud application of our approach can simply instantiate the SDN switch ID by the simulated group ID, which allows the SDN switch to play the role of a cluster since it is connected directly to the DCNs.

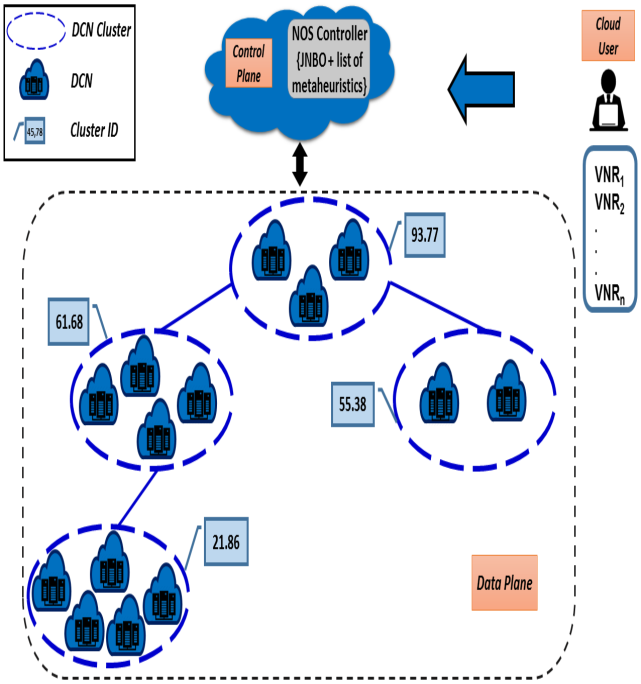

Figure 5. Proposed systyem architecture with a clustered data plane

B. Part Two: VNR Mapping

VNE is one of the key enablers for multi-tenancy procedures that require a full oversee of the system resources. Mainly, metaheuristics tend to balance the exploration and exploitation functions to achieve a near-optimal solution. However, in a large scale environment, the system will suffer from communication overhead triggered by the extension of complex operations. To avoid this matter, the first part of our approach (pre-processing) outlined a constructive algorithm that plays an essential role in boosting metaheuristics efficiency for solving the VNE problem.

This second part exhibits our primary embedding goal of reducing resource utilization while accommodating as many VNs as possible. The principal tasks of our proposed VNE policy are presented in Algorithm 1. To map each VNR, we employed a list of well-known population-based metaheuristics dedicated to solving the VNE problem. The metaheuristic exploitation and exploration search phases are restricted to the max-cluster size (subarray from H) targeting DCN with the highest importance value. We emphasize that each metaheuristic algorithm apply a mutual fitness function defined as:

$$Max\ S = \sum \left(SNod^{S} + SLin^{S} \right) \tag{9}$$

$$Subject\ to:$$

$$SNod^{cap} \geq VNod^{req} \tag{10}$$

$$SLin^{cap} \geq VLin^{req} \tag{11}$$

$$SLin^{cap}, SNod^{cap} \geq 0 \tag{12}$$

$$VLin^{req}, VNod^{req} > 0 \tag{13}$$

Where $\mathrm{SNod^{S}} = \mathrm{SNod^{cap}} / \sum \mathrm{VNod^{req}}$ and $\mathrm{SLin^{S}} = \mathrm{SLin^{cap}} / \sum \mathrm{VLin^{req}}$ are the stress rates of $SNods$ and $SLins$, respectively. The goal of this function is to maximize the local resource utilization in each SNod (Storage, RAM and CPU) and SLin (Bandwidth) until it cannot host any additional virtual resources. This maximization process depends on the virtual resource requirements $\mathrm{VNod^{req}}$ and $\mathrm{VLin^{req}}$ (similarly to substrate local resources) in each VNR, the smaller the fitness value the more DCN stress is maximized.

In essence, our VNE approach is executed in an offline mode where the user will receive a notification (accepted or rejected) after his VNRs got processed. This notification would take some time, especially when there are other prior VNRs arranged in a first-in-first-out (FIFO) queue. Algorithm 1 takes as inputs the Max-Heap array H and two lists of VNRs and metaheuristics (MHs). In line 2, we initialize the VNR requirement ($\mathrm{VNR^{rqm}}$) similar to the calculation of DCN capacity (as in (2) and (3)). In line 3, we extract the first cluster from H, which its ID is the importance value of the most valuable DCN. Based on the selected DCN, we initialized its capacity to establish a capacity verification in line 5. This test will allow us to avoid further computations as the condition is not verified.

A VNR can be scheduled for later mapping (line 14) if the current $\mathrm{DCN^{cap}}$ and initial DCN capacity $\mathrm{DCN^{cap}_{ini}}$ are respectively less than and greater than (or equal to) its requirements. This simple delay mechanism consists of passing a given VNR for mapping when one of the current mapped VNRs has completed its resource allocation duration while the importance condition is verified.

We note that Max-Heap is rebuilt when a DCN is updated (VNR got mapped or unmapped). The result of this rebuilding task is exclusively illustrated in two possible sorts:

1) Preserve the clusters' rank while only change the ID of the max-cluster.
2) Rearrange the clusters' position (adjustment in clusters' categories) when a bigger DCN^{imp} is found in lower levels. Essentially, the SDN controller decides what heap-rebuildment manner (line 11) the data plane should apply based on the DCN^{imp} values that are sent from the four SDN switches after each accepted VNR.

Algorithm 1: An overview of the VNE process.

```
   Input: H[ ], < VNRs >, < MHs >
   Output: DCN
1  foreach $VNR$ In < VNRs > do
2      Initialize $VNR^{rqm}$;
3      $DCN^{imp}$=max-cluster_ID;
4      Initialize $DCN^{cap}$;
5      if $DCN^{cap} \geq VNR^{rqm}$ then
6          foreach $MH$ In < MHs > do
7              Apply MH-VNE();
8              Update $DCN^{imp}$;
9              Return $DCN$;
10         end
11         Rebuild H[ ];
12     else
13         if $DCN^{cap}_{ini} \geq VNR^{rqm}$ then
14             Schedule $VNR()$;
15             Re-execute instructions from line 6 to 11;
16         else
17             Return ''Mapping failed'';
18         end
19     end
20 end
```

C. Evaluation Metrics

Since our primary objective is to lower the exploitation rate of substrate resources, we introduce the most accurate performance measures related to InP to assess work objectives effectively:

- Network utilization: The entire network usage is referred to as the total engagement of both node and link utilization for the *i-th* VNR regarding the total network capacity calculated in (2).

$$Net^{u} = \left[\frac{\sum_{i \in VNR} SNod_i^{cap} + \sum_{i \in VNR} SLin_i^{bw}}{DCN^{cap}} \right] .100\% \tag{14}$$

Where $\mathrm{SNod^{cap}}$ and $\mathrm{SLin^{bw}}$ are node capacity and link bandwidth, respectively.

- Embedding runtime: This metric describes the time of successfully mapping a given VNR regarding the runtime of Algorithm 1. Moreover, it is estimated by the CloudSimSDN simulator [23] through a set of events managed by the network operating system (NOS) class that has the role of SDN controller.

- Number of active nodes and links: This metric counts the average substrate resources that were involved in the embedding process by summing the active nodes and links. It manifest a crucial aspect of improving the embedding quality by saving the total energy cost of DCN.

$$Act^{Nod} = \left(\frac{\sum SNod^{ocp}}{\sum SNod^{SN}}\right).100\% \qquad (15)$$

$$Act^{Lin} = \left(\frac{\sum SLin^{ocp}}{\sum SLin^{SN}}\right).100\% \qquad (16)$$

Where $\mathrm{Nod}^{ocp}$and Lin^{ocp} are the number of occupied nodes and links, respectively. We note that the function defined in (9) promotes the reduction of active substrate resources.

- Acceptance ratio: In VNE research, the acceptance ratio (AR) is defined as a measure of the relationship between the number of accepted VNRs (VNR^{acp}) that are fully mapped, and the total processed VNRs (VNR^{pr}).

$$AR = \left(\frac{\sum VNR^{acp}}{\sum VNR^{pr}}\right).100\% \qquad (17)$$

7. Performance Evaluation

This section demonstrates our policy objectives through a set of performance evaluation simulated out of DCN and VNR designs.

A. *Simulation Settings*

For SN, we implemented a DCN called Diamond (Fig. 6) created by [24] that has only two layers (Core and edge) as an enhanced physical topology to the Fat-tree [25] DCN with an additional aggregation layer.

TABLE I. Servers and VMs requirements from the generated DCNs and VNRs

	Server	VM
Number of CPUs	4-20	2-8
RAM	2-16GB	512MB-8GB
Storage\Size	5GB-1TB	1-20GB
Bandwidth	2-100GB	100MB-5GB

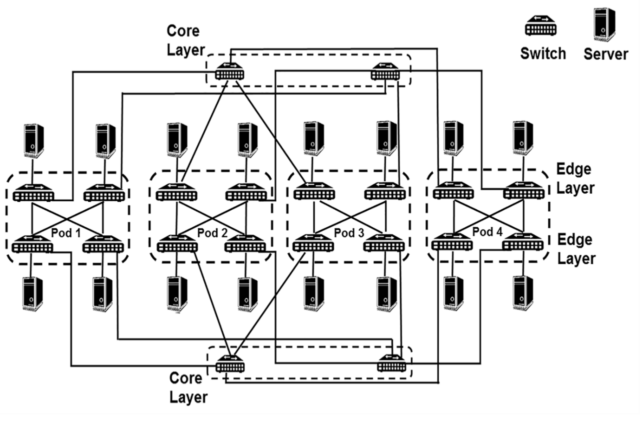

Figure 6. Diamond topology with 4 pods and 16 servers

We have generated 80 DCN instances of the Diamond topology that has a random number of servers from 16 to 128. On the virtual side, we have generated 300 VNRs with a random number of virtual machines (VMs) from 4 to 122. Further resource specifications are presented in Table 1.

We note that all algorithms are implemented using Java; simulations are performed on a computer with Intel(R) Core(TM) i5-6200U CPU up to 2.80GHz and 4GB of RAM.

B. *Comparison Metaheuristics and Data Classification Methods*

In our work, we have used the CloudSimSDN toolkit (based on object-oriented paradigm with Java) that simplifies the flow interactions between the three layers of SDN structure by integrating non-sophisticated traffic shaping methods. We recall that our policy benefits from the stronger optimization capability and acceptable time execution of VNE metaheuristic-based solutions. Table 2 presents all the compared metaheuristics in their basic implementation (no modification has been made).

To ensure that the VNE problem is solved in one stage coordinated process, we combine each metaheuristic with a breadth-first search (BFS) algorithm to embed VNods and VLins concurrently. The metaheuristic with the best performance will be selected as a VNE solution based on univariate data classification techniques (similar to JNBO). The compared classification schemes are listed in Table 3.

It is noteworthy that our policy has overcome critical limitations engendered in the previous metaheuristic-based studies for distributed VNE, including:

1) Solve the problem with no focus on implementing a prioritization scheme for the DCNs in the data plane, resulting in extra power consumption since no mechanism is incorporated to the search and select the appropriate DCN during the embedding process.
2) The exigency of employing a distributed algorithm in order to manage the traffic (i.e., additional forwarding mechanism to guide the network packets).
3) Lack of a baseline solution (initial checking) that assures conformity between the VNRs and the available DCNs, which will prevent unnecessary computing during the embedding process.

As a solution to the previously mentioned limitations, we suggest the following refinements:

1) Adjust the metaheuristic solution to be adapted with the diverse requirements of the arrived VNRs using a pre-processing procedure so that a VNR is rejected or scheduled for a later mapping round if it does not meet the DCN capacity.
2) Adopt an innovative network architecture such as SDN Software-Defined Data Centre (SDDC) to increase the metaheuristic solution accuracy (only focus on dealing with VNE problem) while a control

TABLE II. Compared algorithms

Notation	Algorithm description
PSO-VNE [26]	Particle swarm optimization-based solution where the operations and parameters of particles are redefined tosuit the VNE problem.
AC-VNE [27]	Inspired by the behavior of an ant colony to find the optimal path towards the food source. It split VNRs into sorted sub-VNRs to be assigned optimally.
GWO-VNE [28]	The VNE problem is solved based on an inspired behavior of the grey wolves during the hunting process which improves the search for the fittest resources.
GA-VNE [29]	A classic genetic algorithm to solve the VNE problem by taking advantage of the selection, crossover, mutation and feasibility checking operations.
HS-VNE [30]	Harmony search algorithm exploits an inspired improvisation process performed by musicians based on probabilistic rules to determine a pleasant harmony (set of DCN resources in VNE).

TABLE III. Compared univariate data classification methods

Unsupervised data classification	Description
Defined-intervals	A partitioning method that divides a normal distribution into four classes where each class can have a random number of data included in [5-40] while the total size of all classes equal 80.
Quantiles	An algorithm that computes three quartiles Q1(25th percentile), Q2(median or 50th percentile) and Q3(75th percentile,) which will divide a sorted distribution into four quarters where each will have 25% of the data.
Exponential interval	An algorithm that generates classes so that the number of observations in each class interval increases exponentially.
Standard deviation	A classification method that breaks the dataset based on how much it differs from the mean. In our case, we have four classes to define, which means two standard deviations must be calculated, one above the mean and one below the mean.

unit handles the networking operations.

C. *Results Analysis*

In this subsection, we investigate the impact of classified DCNs (using JNBO + Max-Heap design) on the VNE quality within a distributed system. The designed clustering policy has transformed the data plane plan into a multistage structure.

Fig. 7 illustrates the case where the pre-embedding process is excluded, including the importance computations and the heap tree construction. We can observe that despite the low rate of network utilization and active nodes and links, GWO-VNE has managed to provide the best acceptance ratio; this is due to the developed agents that improved the matching mechanism. Conversely, HS-VNE scored the worst performance resulting from the ability to reach a high number of iteration while not improving the final solution.

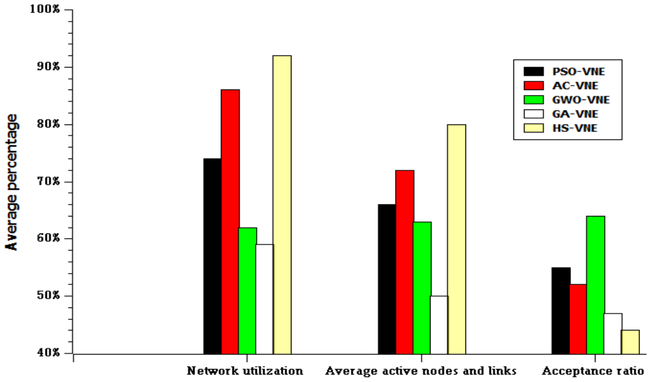

Figure 7. Performance comparison among metaheuristics without clustering integration

Fig. 8 depicts the result effects of applying a preliminary resource rearrangement (DCN management). Notably, the metaheuristics performance has been improved due to rate reduction for network usage and the number of active nodes and links. Moreover, an overall increasement (more than 10%) in acceptance ratio is captured, especially in the case of AC-VNE, where more VNRs got mapped due to

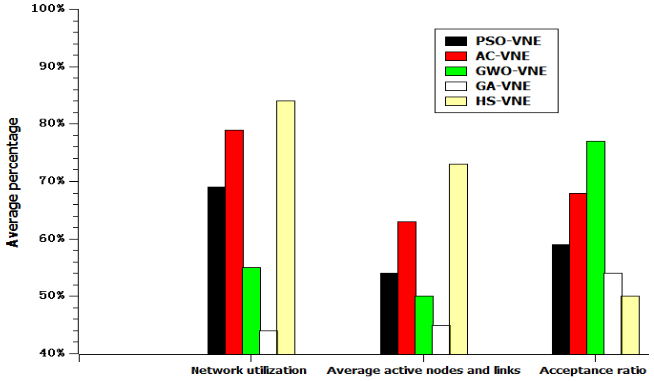

Figure 8. Performance comparison among metaheuristics with clustering integration

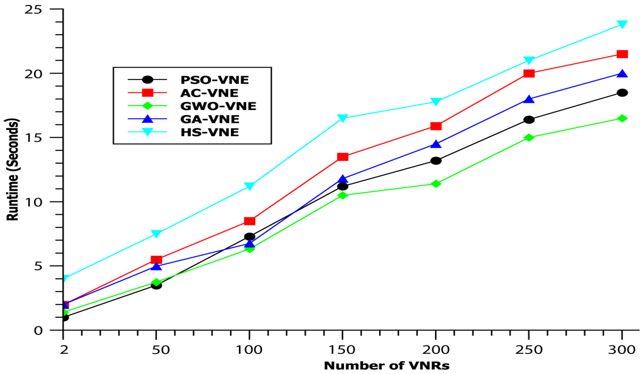

Figure 9. Total VNE runtime comparison between metaheuristics without clustering integration

the ability to adapt with the DCN capacity updates (i.e., assured convergence in low-dimensional problem) during the reconstruction of Max-Heap.

Figs. 9 and 10 show the time execution of VNE-based metaheuristics over a set of processed VNRs. In Fig. 9, we notice that GA-VNE, PSO-VNE and GWO-VNE yield a convergent runtime performance in the first half of VNRs. Nevertheless, GWO-VNE has scored lower time execution at the end of the simulation. Via a classification framework, the runtime results in Fig. 10 show a positively correlated performance as Fig. 9, yet with a significantly reduced time running (up to 6 seconds). Particularly, PSO-VNE outperforms GWO-VNE by providing a short computational time due to the few parameters to adjust.

Overall, the initiative classification has enhanced the objective model of VNE prospects in a distributed cloud environment. This improvement is supported via the SDN centered management plan. A good VNE algorithm should essentially engender a balanced performance in reducing the cost of the engaged physical resources while increasing the long-term profit from the accepted VNRs. Accordingly, the previous results from Figs. 7, 8, 9 and 10 have yielded the following conclusion (the symbol > indicates that the prior algorithm outperforms the next one): GWO-VNE > AC-VNE > PSO-VNE > GA-VNE > HS-VNE.

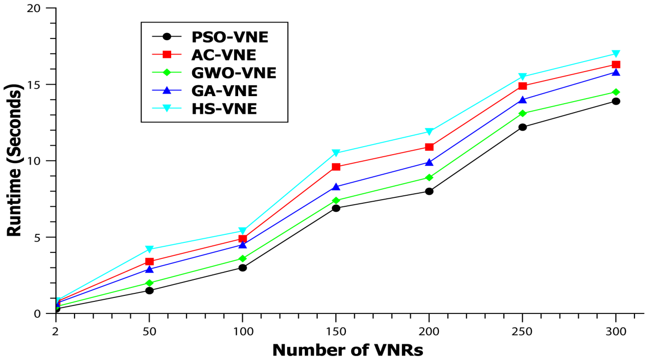

Figure 10. Total VNE runtime comparison between metaheuristics with clustering integration

Based on the latter result, we ran conclusive tests to validate the efficiency of JNBO compared with similar unsupervised techniques. Figs. 11 and 12 represent the compared results of the classification techniques using GWO-VNE as a metaheuristic-based solution.

Fig. 11 shows dissimilar results between the compared methods caused by our policy orientation, which targets for each VNR the max-cluster (with very high capacity of DCNs) only. Hence, we observe a tremendous amount of network utilization and high average of active nodes and links for the exponential interval, which led to deliver almost the same acceptance ratio as JNBO. The defined-intervals method scored the lowest VNE performance due to the randomization argument that requires a prior knowledge of the VNRs condition.

Fig. 12 shows that the exponential interval algorithm scored the highest runtime at an increasing rate. Quantiles and standard deviations methods show a near-matched performance with a lesser runtime for the quantiles method.

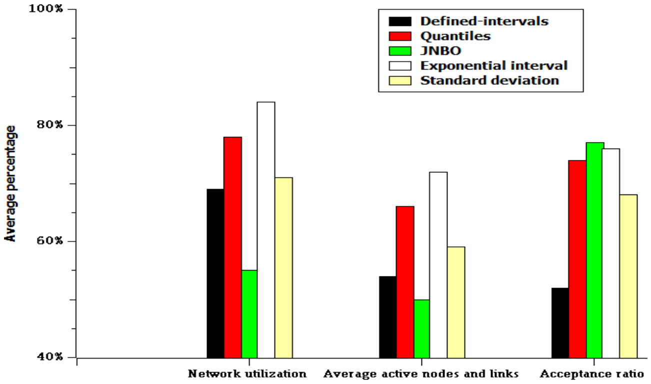

Figure 11. Performance comparison of classification methods within a VNE framework

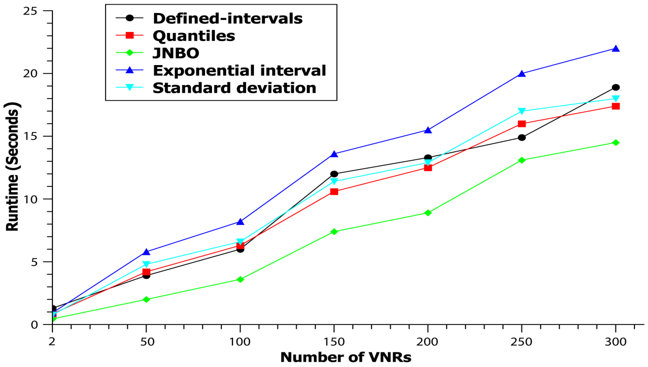

Figure 12. Total VNE runtime comparison between different univariate data classification methods

The defined-intervals algorithm has attained an unstable performance due to the frequent change in the class size with each VNR. Based on these remarks, we deduce the following main factors that affect the policy results: (i) Number of DCNs (class size) at the Max-cluster; (ii) Rational offline mode where the VNRs are processed in a FIFO manner (no predefine order-based) with different resources requirements.

To sum up, the VNE policy based on JNBO classification and GWO-VNE has provided the most convenient performance for all VNE objectives with a reasonable runtime.

8. Conclusion and Futer Work

Cloud resource management becomes more diligent with the rapid development of network technologies. Thus, InPs have established dedicated network policies that rely on innovative task automation and advanced accessibility. In this paper, we adopt a distributed cloud datacenters based on an SDN structure to solve the VNE problem. Indeed, our policy guarantees the delivery of resource discovery, control and VN mapping.

The VNE process is conducted through a DCN management based on data classification (importance identification) and metaheuristic-based solution. In the sense of resource allocation, the mapping solution is undertaken by static substrate resource capacity with no mechanism of virtual resource migration. Simulation results indicate that the proposed DCN-clustered strategy achieved a favorable impact compared with similar methods.

Since the DCN architectures are much broader and more divergent, we strive to develop a distributed network-aware policy that manages heterogeneous cloud DCNs while defining the VNR types (tailored to their diverse needs) in favor of ensuring an efficient VNE solution with low power consumption.

References

[1] M. A. Shahid, N. Islam, M. M. Alam, M. Mazliham, and S. Musa, “Towards resilient method: An exhaustive survey of fault tolerance methods in the cloud computing environment,” *Computer Science Review*, vol. 40, p. 100398, 2021.

[2] A. L. S. Muhammad *et al.*, “Cloud computing enabled data center infrastructure development and deployment by it firms,” *International Journal of Computing and Digital Systems*, vol. 9, no. 1, 2020.

[3] “Sdn-definition page,” https://www.opennetworking.org/sdn-definition, accessed: 2022-03-08.

[4] O. Yousuf and R. N. Mir, “A survey on security enhancements in the internet of things using software-defined-networking (sdn),” *International Journal of Computing and Digital Systems*, vol. 9, no. 4, pp. 591–606, 2020.

[5] A. Fischer, J. F. Botero, M. T. Beck, H. De Meer, and X. Hesselbach, “Virtual network embedding: A survey,” *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 1888–1906, 2013.

[6] H. Cao, L. Yang, Z. Liu, and M. Wu, “Exact solutions of vne: A survey,” *China communications*, vol. 13, no. 6, pp. 48–62, 2016.

[7] H. Cao, H. Hu, Z. Qu, and L. Yang, “Heuristic solutions of virtual network embedding: A survey,” *China Communications*, vol. 15, no. 3, pp. 186–219, 2018.

[8] J. Zhang, X. Wang, H. Huang, and S. Chen, “Clustering based virtual machines placement in distributed cloud computing,” *Future Generation Computer Systems*, vol. 66, pp. 1–10, 2017.

[9] F. Nzanywayingoma and Y. Yang, “Efficient resource management techniques in cloud computing environment: a review and discussion,” *International Journal of Computers and Applications*, vol. 41, no. 3, pp. 165–182, 2019.

[10] M. Alaluna, L. Ferrolho, J. R. Figueira, N. Neves, and F. M. Ramos, “Secure virtual network embedding in a multi-cloud environment,” *arXiv preprint arXiv:1703.01313*, 2017.

[11] Z. Xu, W. Liang, and Q. Xia, “Efficient embedding of virtual networks to distributed clouds via exploring periodic resource demands,” *IEEE Transactions on Cloud Computing*, vol. 6, no. 3, pp. 694–707, 2016.

[12] K. Alhazmi, A. Shami, and A. Refaey, “Optimized provisioning of sdn-enabled virtual networks in geo-distributed cloud computing datacenters,” *Journal of Communications and Networks*, vol. 19, no. 4, pp. 402–415, 2017.

[13] A. S. Alzahrani and A. A. Shahin, “Energy-aware virtual network embedding approach for distributed cloud,” *arXiv preprint arXiv:1710.11590*, 2017.

[14] Y. Xin, I. Baldine, A. Mandal, C. Heermann, J. Chase, and A. Yumerefendi, “Embedding virtual topologies in networked clouds,” in *Proceedings of the 6th international conference on future internet technologies*, 2011, pp. 26–29.

[15] T.-H. Lee, S. Tursunova, and T.-S. Choi, “Graph clustering based provisioning algorithm for virtual network embedding,” in *2012 IEEE Network Operations and Management Symposium*. IEEE, 2012, pp. 1175–1178.

[16] J. Xue, J. You, J. Wang, and F. Deng, “Nodes clustering and dynamic service balance awareness based virtual network embedding,” in *2013 IEEE International Conference of IEEE Region 10 (TENCON 2013)*. IEEE, 2013, pp. 1–4.

[17] I. Houidi, W. Louati, W. B. Ameur, and D. Zeghlache, “Virtual network provisioning across multiple substrate networks,” *Computer Networks*, vol. 55, no. 4, pp. 1011–1023, 2011.

[18] D. G. Andersen, “Theoretical approaches to node assignment,” 2002.

[19] T. F. Gonzalez, “On the computational complexity of clustering and related problems,” in *System modeling and optimization*. Springer, 1982, pp. 174–182.

[20] S. G. Kolliopoulos and C. Stein, “Improved approximation algorithms for unsplittable flow problems,” in *Proceedings 38th Annual Symposium on Foundations of Computer Science*. IEEE, 1997, pp. 426–436.

[21] H. Wang and M. Song, “Ckmeans. 1d. dp: optimal k-means clustering in one dimension by dynamic programming,” *The R journal*, vol. 3, no. 2, p. 29, 2011.

[22] G. F. Jenks, “The data model concept in statistical mapping,” *International yearbook of cartography*, vol. 7, pp. 186–190, 1967.

[23] J. Son, A. V. Dastjerdi, R. N. Calheiros, X. Ji, Y. Yoon, and R. Buyya, “Cloudsimsdn: Modeling and simulation of software-defined cloud data centers,” in *2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*. IEEE, 2015, pp. 475–484.

[24] Y. Sun, J. Cheng, Q. Liu, and W. Fang, “Diamond: An improved fat-tree architecture for large-scale data centers.” *J. Commun.*, vol. 9, no. 1, pp. 91–98, 2014.

[25] M. Al-Fares, A. Loukissas, and A. Vahdat, “A scalable, commodity data center network architecture,” *ACM SIGCOMM computer communication review*, vol. 38, no. 4, pp. 63–74, 2008.

[26] X. Cheng, Z.-b. Zhang, S. Su, and F.-C. Yang, “Virtual network embedding based on particle swarm optimization,” *Dianzi Xuebao(Acta Electronica Sinica)*, vol. 39, no. 10, pp. 2240–2244, 2011.

[27] I. Fajjari, N. Aitsaadi, G. Pujolle, and H. Zimmermann, “Vne-ac: Virtual network embedding algorithm based on ant colony metaheuristic,” in *2011 IEEE international conference on communications (ICC)*. IEEE, 2011, pp. 1–6.

[28] A. Bouchair, S. A. Makhlouf, and Y. Belabbas, “Grey wolf optimizer for virtual network embedding in sdn-enabled cloud environment,” in *International Conference Europe Middle East & North Africa Information Systems and Technologies to Support Learning*. Springer, 2019, pp. 321–330.

[29] L. Davis, “Handbook of genetic algorithms,” 1991.

[30] Z. W. Geem, J. H. Kim, and G. V. Loganathan, “A new heuristic optimization algorithm: harmony search,” *simulation*, vol. 76, no. 2, pp. 60–68, 2001.

Abderrahim Bouchair Ph.D candidate in computer science at the University of Oran1 Ahmed Ben Bella (Algeria). His research revolves mostly around the areas of Cloud Computing, Virtualization, Metaheuristics and Network management.

Belabbas Yagoubi PhD in Computer Science. He has been a full professor at the University of Oran1 Ahmed Ben Bella (Algeria). His field of interests is Parallel and Distributed Systems including Security, Fault Tolerance, Replication, Load Balancing and Task Scheduling. He occupied the function of Dean of the Faculty of exact and applied science since 2009. He is currently the rector of University of Mostaganem (Algeria) and a member of the LIO laboratory (Algeria).

Sid Ahmed Makhlouf Ph.D in computer science. His main research interests include Distributed System, Cluster, Grid and Cloud Computing, Load Balancing, Task and Workflow Scheduling, and Machine Learning. He is currently an associate professor at the University of Oran1 Ahmed Ben Bella (Algeria).