2021 International Conference on Networking and Advanced Systems (ICNAS) | 978-1-6654-2821-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICNAS53565.2021.9628897

Dual Load Balancing Strategy for Virtual Network Embedding in SDN-Enabled Distributed Cloud

Abderrahim Bouchair
LIO Laboratory, Computer Science Department
University of Oran1 Ahmed Ben Bella,
PB 1524 El M'Naouer, Oran, Algeria
bouchair.abderrahim@gmail.com

Belabbas Yagoubi
LIO Laboratory, Computer Science Department
University of Oran1 Ahmed Ben Bella,
PB 1524 El M'Naouer, Oran, Algeria
byagoubi31@gmail.com

Sid Ahmed Makhlouf
LIO Laboratory, Computer Science Department
University of Oran1 Ahmed Ben Bella,
PB 1524 El M'Naouer, Oran, Algeria
sidahmed.makhlouf@gmail.com

***Abstract*—Virtual Network Embedding (VNE) has been acknowledged as an essential task in a functional cloud environment. Derived from the Network Virtualization (NV) concept, the VNE process involves assigning virtual resources from a given Virtual Network (VN) into the fittest physical resources from a substrate network. The integration of software-defined networking (SDN) technologies in the cloud are becoming a driven practise due to its centralized design that allows advanced resource management via several cloud policies. In this paper, we propose a systematic VNE policy based on a two-staged Load Balancing (LB) plan. Our approach covers firstly the Virtual Network Requests (VNRs) assignment onto multiple Data Center Networks (DCNs). Then, the servers in each DCN are classified in which they host the most compatible virtual resources. The effectiveness of our approach is finally validated by a set of numerical experiments.**

***Index Terms*—Cloud Computing, Virtual Network Embedding, Load Balancing, Software-Defined Networking, Resource Management.**

I. Introduction

Resource management is a critical routine imposed in any cloud system, which can have a far-reaching effect on the obtained performance. Load balancing is a popular cloud procedure operated by the cloud service providers to manage many applications where certain workloads are preferable to be dispersed across many resources [1]. Distributed cloud systems are distinguished as an ideal environment where LB is most likely to be used. The main objective of the LB process is to avoid overloading the available resources while exploiting the cloud scalability feature. Accordingly, the Virtual Network Embedding (VNE) procedure falls within this context. In this work, a VNE problem is solved using a two-level LB approach in SDN based cloud with multiple Data Center Networks (DCNs). A LB technique called Dynamic Weighted Round Robin (DWRR) based on [2] is applied to conduct a set of VNRs towards the available DCNs based on the latter importance values. At the second level, a categorical LB method is employed locally in every DCN in order to arrange servers and their attached links regarding the occupied capacity. The remainder of this article is planned as follows: the most related works are cited in Section II. Next, we outlined in Section III the principles of a VNE problem and its projection on a SDN-based distributed cloud. In Section IV, we described the implementation of our proposed solution based on a predefined context. Section V depicts the experimentation results. Finally, Section VI concludes the study along with perspective work.

II. Related Work

With the rapid evolution of cloud services, the amount of loads on the network resources increases progressively. Therefore, many researchers have devoted their works to solving the VNE problem where a breaking point is avoided in a Substrate Network (SN) by rationally distributing the arrived VNRs. From an energy-saving standpoint, Pyoung and Baek [3] propose an online iterative algorithm to solve the VNE problem. This algorithm is based on a penalty function that measures the embedding cost through an uncoordinated VN mapping process, with the aim to reduce energy consumption when the power is on. Similarly, Zhang [4] has incorporated the substrate node energy consumption and its utilization rate into the mapping computation ability of VNRs. This consolidation approach employs a load balance aware VNE algorithm based on node ranking mechanism, determining the node importance, so the embedding process is optimized.

Solving the VNE problem is typically addressed from multiple factors such as quality of service and resource equilibrium. Li et al. [5] proposed a load balancing plan to effectively solve a multi-objective VNE problem while improving the VNRs acceptance ratio and their revenue. The Group Search Optimizer (GSO) ensures this embedding scheme via two local search functions that target cost and node balance enhancement. The authors in [6] have employed a technique to solve a multi-objective virtual machine placement in the cloud. The solution is based on a reinforcement learning model to achieve an efficient virtual machine mapping and placement, which manages the loads in the hosts from an inter-load balancing and intra-load balancing stand point. Based on hybrid genetic algorithm, a VN mapping strategy is proposed by Zhang et al. [7] that improve its flexibility through pheromone-based selection and a dynamic cross-probability. Moreover, the mapping is reduced by using a load balancing strategy, which depends on a weight update mechanism.

Previously mentioned contributions mainly focus on solving the VNE problem in a single cloud environment. However, in our work, we adopt a distributed cloud environment implemented on the basis of a layered structure of the SDN. Indeed, this system scheme promotes our VNE solution via a dual LB application (i.e., two different LB methods applied on two linked stages), which involves both virtual and substrate resources.

III. Theoretical Preliminaries

The VNE problem is considered a NP-Hard problem, solved from multiple perspectives [8]. Consequently, a set of standard agreements needs to be discussed.

A. Virtual Network Embedding Problem

Resource allocation is the main challenge in the embedding concept of VNs. Typically, a substrate resource is allocated during the VNE process when it can satisfy the virtual resource requirements. Equation (1) summarizes the general VNE problem with a basic network model of $SN = (N^S, L^S)$ and $VNR_i = (N_i^V, L_i^V)$; where it consists of two functions: Node Mapping Function (NMF) that relies on the available Substrate Nodes (SNods) in reference to the received Virtual Nodes (VNods) from the i-th VNR. Samely, the Link Mapping Function (LMF) is designated from the matching mechanism between the Substrate Links (SLins) and Virtual Links (VLins).

$$VNE : \begin{cases} NMF : VNod_i \rightarrow SNod \\ LMF : VLin_i \rightarrow SN' \subseteq SN \end{cases} \tag{1}$$

The NMF must embed every node from the VNR, where the LMP must find a Sub-SN (SN') designating a specific path or channel. Fig. 1 illustrates the mapping procedure of two VNRs onto a SN. We note that a successful mapping process depends on the underlying resource capacity. Moreover, a NMP must map the entire VNR where a SNod can host only one VNod from the same VNR. In LMF, a VLin can be assigned onto multiple SLins as shown in Fig. 1 (e.g., E to F).

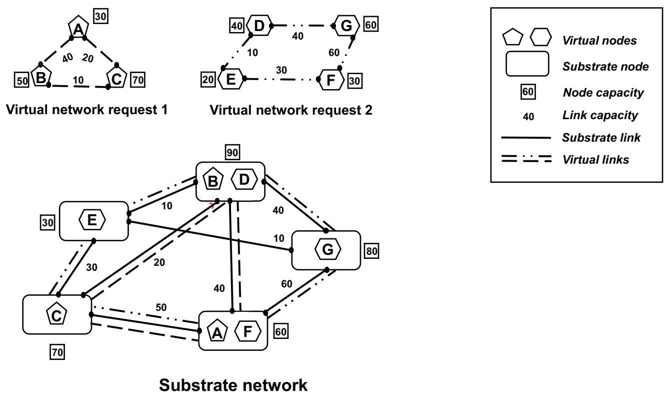

Fig. 1. An example of the VNE process.

B. Case Study

In recent years, cloud infrastructure has tended to combine modern components from different network technologies. To enhance the accuracy of the cloud services, SDN has been extensively incorporated to provide more innovation while the cloud is distributed in several geographic locations.

The Open Networking Foundation (ONF) [9] reported that the SDN structure provides advanced controllability and ensured automation based upon three primary planes. Following that, a cloud user submits a list of VNRs that will trigger the cloud system to run a low-cost embedding to guide the VNRs to the most suitable DCN among a set of distinguished DCNs as shown in Fig. 2. This VNE process is described through the SDN layers as follows:

1) *Application Plane*: A service-aware platform that describes the available products and services. This layer receives the VNRs via the SDN broker, who will decide whether to forward it or reject it.
2) *Control Plane*: It is considered a crucial source of the cloud paradigm where the network intelligence is centralized and carried out by the SDN controller. The prominent role of the SDN controller is to set up policy-based management based on its programmability feature, which will automate the solution of any network issue.
3) *Data Plane*: Known also as the infrastructure layer where the substrate network equipment is accommodated. These materials, including switches and routers, handle the sent data from the SDN controller to forward it to its destination (e.g., servers) passing by a specific DCN.

We point out that in a practical SDN-enabled cloud, the exchange of information between the planes is accomplished using three types of Application Programming Interfaces (APIs) known as Northbound Interfaces (NBIs), Southbound Interfaces (SBIs), and East-west APIs.

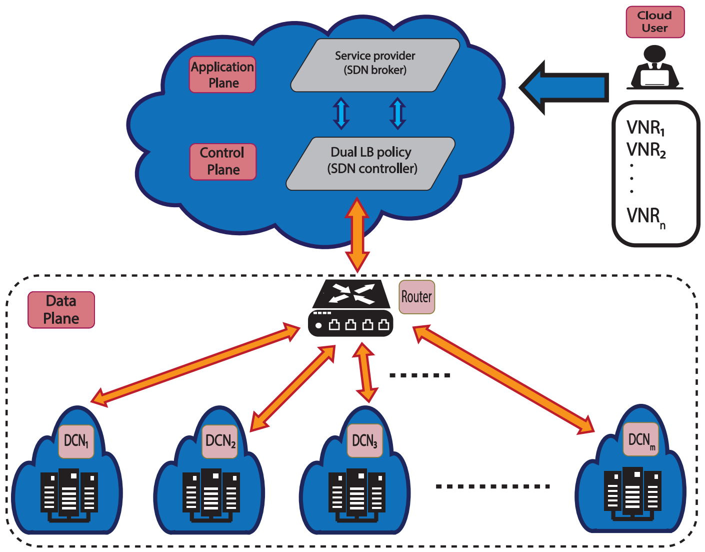

Fig. 2. An overview of the VNE process in a SDN-enabled distributed cloud.

In our work, the simulation was performed via the CloudSimSDN toolkit [10] that has a pre-implemented class called Network Operating System (NOS), which has the role of a SDN controller (i.e., it undertake the functions of the APIs in addition to its main task of executing the designated network policy).

Substantially, we emphasize that the distributed cloud based on SDN architecture illustrated in Fig.2 has been adopted as our proposed system model to solve the VNE problem. Also, the exigency of implementing a distributed algorithm is ejected by the automated networking aspect of SDN (i.e., the existence of programmable SDN switches).

IV. Solution Methodology

Maintaining a solid analogy between the arrived VNs and SNs is an essential aspect when we focus on optimizing the VNE process. In conformity with the classification of VNE approaches in [8], we have endorsed the following preferences during the VNE optimization:

- **Redundant:** Similar component execute an identical functions.
- **Static:** The number of physical resources is fixed with no VN migration.
- **Offline:** The VNRs are managed in First-In-First-Out (FIFO) waiting list.

This work proposes a heuristic VNE solution that explores the distributed cloud (consolidated with the SDN structure) through a load balancing policy, divided into two parts: VNR $assignment$ and VNR $mapping$.

A. Part One: VNR Assignment

At this stage, we focus on directing a set of VNRs across all the SDN layers. Initially, the SDN broker examines the received VNRs possibility to be mapped by executing a simple verification function, which will permit us to gain more time and prevent the cloud system from consuming unnecessary energy. This function compares the total available DCN capacity (calculated by (2) in terms of total SNods capacity and the available DCN bandwidth) with the total VNR requirement (using (3) in terms of total VNods requirements and VNR required bandwidth).

$$DCN^{c} = \sum_{i=1}^{n} SNod^{c} + DCN^{bw} \tag{2}$$

$$VNR^{r} = \sum_{i=1}^{n} VNod^{r} + VNR^{bw} \tag{3}$$

A VNR is accepted for further processing when the value of VNR^{r} is less or equal to the biggest value of DCN^{c}. The SDN controller received the accepted VNRs and then established a DWRR as an orchestration policy of VNR assignment. Our version of the DWRR relies on the importance value of each DCN (i.e., it changes based on the current load in the DCN). This value is determined by calculating at first the $Spearman's\ rank\ correlation\ coefficient$ (ρ) in (4), which takes DCN^{c} and K (Total number of servers in a given DCN) as two variables to find the difference ranks d_i between them for each DCN. Eventually, the DCN importance value DCN^{imp} in (5) can be calculated by measuring the j-th DCN average similarity versus other DCNs.

$$\rho = 1 - \frac{6\sum d_i^2}{n.(n^2-1)} \tag{4}$$

$$DCN_j^{imp} = \left| \frac{\rho - \rho^t}{(n-1).\rho} \right| \tag{5}$$

Where n is the sum of DCNs and ρ^t is the correlation coefficient produced by excluding the j-th DCN. The process of VNRs assignment illustrated in Fig. 3 is handled by the SDN controller (acts as a load balancer) that follows these descriptive steps:

1) Define the importance value for each DCN (DCN_1 has the higher DCN^{imp} equals to 59.46).
2) The accepted VNRs via the SDN broker (7 VNRs) are forwarded to the load balancer. This latter launches the DWRR procedure by attributing the VNRs into the DCNs in a rotation mode while updating the DCN^{imp} value. For instance, the 7^{th} VNR is assigned to the 3^{rd} DCN rather than the 2^{nd} indicating an inability to fulfill the VNR requirements.
3) A VNR is scheduled for a later round of assignment when the current status of all the DCNs is incapable of hosting the VNR (i.e., $VNR^{r}$is greater than every DCN^{imp}). The delayed VNR can be finally assigned when a given VNR has completed its mapping period, releasing sufficient resources.

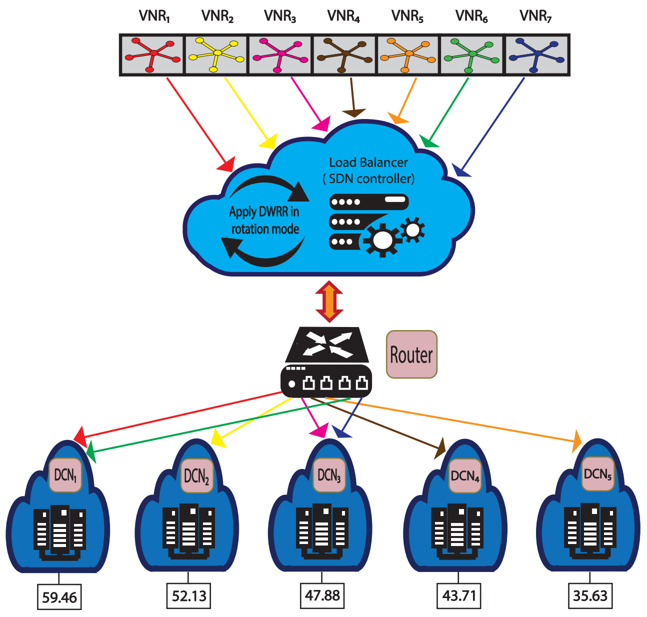

Fig. 3. A VNR assignment process based on DWRR load balancing.

B. Part Two: VNR Mapping

Our primary concern is to select the SNods focusing on minimizing its *stress* Ψ compared with the overall hosted load in the DCN. Therefore, this second part proposes a greedy technique for load balancing the VNods based on the following mathematical formulation.

$$Min\ Z = \sum_{i=1}^{n} \Psi_i \tag{6}$$

$$Subject\ to:$$

$$VNod^r \leq SNod^c \tag{7}$$

$$VLin^r \leq SLin^c \tag{8}$$

$$SNod^c, SLin^c \geq 0 \tag{9}$$

$$VNod^r, VLin^r > 0 \tag{10}$$

Where Ψ is defined in (11), which depends on the number of VNods embedded in the SNods and their gained revenue R (defined in (12) in reference to the $VNod^r$ that has been satisfied and the bandwidth that in use for all its attached links).

$$\Psi = \frac{\sum_{i=1}^{n} VNod_i}{\sum_{i=1}^{n} R_i} \tag{11}$$

$$R_i = VNod^r + \sum_{i=1}^{n} VLin_i{}^r \tag{12}$$

The objective function introduced in (6) ensures that the amount of the incoming loads are embedded efficiently into a set of SNods. Hence, a Resource-Constrained (RC) load balancing policy is established through classifying the current state of a SNod in three classes. We define a *threshold* τ that will assist in distinguishing the class that a given SNod belongs to, and then we measure the *residual capacity* Ω for the i-th SNod in (13).

$$\Omega_i = SNod^c - \sum_{i=1}^{n} VNod_i{}^r \tag{13}$$

To perform this classification, we choose two threshold values, τ^+ = 0.7 and τ^- = 0.4 based on the generated residual capacities. Thereby, the SNod classes are created as follows:

- **Underloaded SNods:** Contains the list of SNods that satisfy this condition: $\theta > \tau^+$ where $\theta = \Omega_i/\Omega^s$ and Ω^s denotes the total residual capacity of all SNods in the DCN except the i-th SNod.
- **Overloaded SNods:** It has similar aspect as the previous class but with this condition: $\theta < \tau^+$.
- **Balanced SNods:** The server list that belong to this class must respect the following constraint: $\theta \in [\tau^-, \tau^+]$.

A sample event of our proposed policy is illustrated in Fig. 4, where two VNRs are mapped into a single SN based on the server's load status (this policy is identical in every DCN within the cloud).

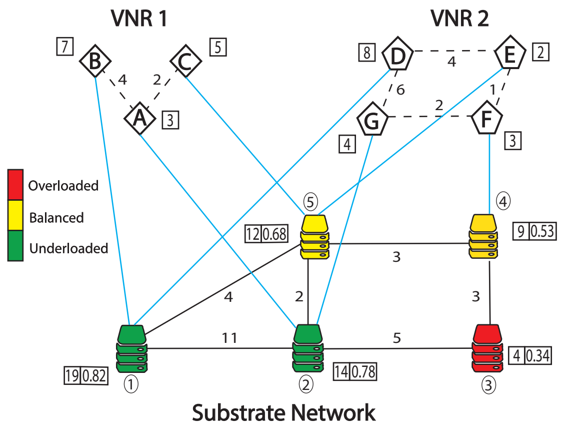

Fig. 4. A VNR mapping process based on a RC load balancing.

Mainly, the embedding process conducts each VNod to the server with the highest θ and then check if the constraints in (7), (8), (9), and (10) are satisfied in order to finalize the allocation of the required physical resources.

For example, the server with a marked ID of 1 is considered underloaded since it has θ equals 0.82 (highest value) and confirms the class condition while available capacity equals 19 (favorable SNod selection).

V. Performance Evaluation

The dual load balancing strategy has been implemented and evaluated using the CloudSimSDN (extended from the CloudSim simulator [11]), a Java-based toolkit for modeling network policies in SDN-enabled cloud datacenters. The implementation is carried out by extending the NOS class to run the logic of our proposed strategy regarding a VNE context. We have used the physical topology and virtual topology classes to generate the DCNs and VNRs, respectively, in a JSON format files. Each of the produced files contains all the information needed for mapping, including the capacity of substrate resources and the load requirements for virtual resources.

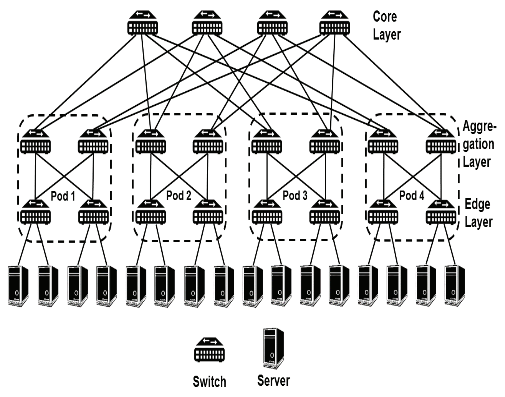

Fig. 5. Fat-tree architecture with 16 servers.

TABLE I
CONFIGURATION SETUP FOR DCNS AND VNRS

	DCN	**VNR**
Number of CPUs	4-20	2-8
Storage\Size	5GB-1TB	1-20GB
RAM	4-16GB	512MB-8GB
Bandwidth	4-100GB	220MB-5GB

We note that we have used the Fat-tree [12] topology as a DCN architecture due to its recursive scalable design, path diversity access with low-cost commodity switches in contrast to the conventional data centers.

Fig. 5 depicts an instance of the Fat-tree topology with 16 servers connected to a hierarchical structure that consists of the edge layer (attached directly to the servers). The aggregation layer forms four pods with the edge layer (both layers have eight switches) to reduce network congestion. The core layer has 4 switches connected to the control plane through a router to manage different traffic requests. Further DCN and VNR specifications are introduced in Table I, including the range capacity of SNods (servers), SLins, VNods (Virtual Machins (VMs)), and their attached VLins.

We note that all the simulations were performed on a computer with 8GB of RAM and an I5-4300U CPU up to 2.5 GHz.

A. Evaluation Metrics

To test the accuracy of our proposed strategy, we have elected a set of the most relevant metrics for the adopted VNE context. Besides the SNod stress metric and the result obtained from the VNR assignment stage (i.e., number of assigned VNRs per DCN), we took up these additional measurements:

1) $Acceptance\ Ratio\ (AR)$: The VNR acceptance rate is given by (14), and it defines the number of VNRs successfully accepted VNR^{apt} (i.e., apt to be embedded) over the assigned VNRs VNR^{asg} for each DCN instance.

$$DCN_i{}^{AR} = \frac{\sum VNR^{apt}}{\sum VNR^{asg}} \tag{14}$$

2) $Revenue\ to\ Cost\ Ratio\ (R/C)$: This performance metric measures the rate efficiency between the total obtained revenue (defined in (12)) and the total issued cost (defined in (16)). The cost embedding of VLins is related to the Path Link (PL), which is the number of SLins used by a single VLin.

$$R/C = \frac{\sum R_i}{\sum C_i} \tag{15}$$

$$C_i = VNod^r + VLin^r.PL \tag{16}$$

3) $Network\ Utilization$: Resource utilization has always been a target factor for LB policies. The metric in (17) calculates the entire network usage for the i-th DCN with regards to the mapped VNods and VLins into the utilized SNods (N^u) and SLins (L^u), respectively.

$$DCN^u = \frac{\sum_{i\in N^u}^{n} VNod_i{}^r + \sum_{j\in L^u}^{n} VLin_j^r}{DCN_i{}^c} \tag{17}$$

B. Evaluation Methods

To assess the efficiency of the proposed LB strategy, we compared the obtained results to the most convenient LB methods for our case study.

- **Randomized Static (RS):** As its name implies, the virtual resource attribution is simply distributed by random using a number generator (applied in this work) or a specified method like generating random results based on $Poisson\ distribution$.
- **Least Connection (LC):** This method consists of assign/embed the virtual resources to the substrate resource with less loads. For example, if we take the VNRs assignment stage (same logic in the embedding stage), the next VNR is assigned to the DCN with fewer VNRs.
- **Weighted Round Robin (WRR):** This method guides the incoming virtual resources based on a static weight given to every physical resource while respecting the basic concept of the round robin algorithm.
- **Dynamic Priority Weighting (DPW):** The SDN controller always picks an object (DCN, SNod, and SLin) with high capacity, which will be changed when an assignment/embedding occurs.

C. Simulation Results

To initiate the simulation scenarios, we have generated 300 VNRs with 4 to 122 VMs and 50 DCN instances of the Fat-tree topology with 16 to 128 servers. The number of iteration for each LB method is fixed at 2000 iterations. These parameter values have been chosen to reinforce the scalability aspect of our approach and to validate the results based on a real world scenarios.

The results in Fig. 6 and Fig. 7 are affiliated to a ranked DCNs based on their importance values (i.e., descending order where the 50^th DCN has the lowest importance value). Fig. 6 shows a higher VNRs assignment for the adopted DWRR method due to its tracking quality that updates the DCN^{imp} value for every received engaged VNR. In particular, the DPW method provides the closest performance to DWRR, where some DCNs were fully loaded before treating all the VNRs (i.e., decreased the possibility of assigning more VNRs).

Notably, the RS method in Fig. 7 attained the lowest AR across the DCNs (i.e., the last 5 DCNs did not map any VNR). Overall, The applied RC policy scored the highest AR within almost all the DCNs, owing that to the selective mechanism applied on the class types regarding the SNod load state. The methods LC and WRR produce a similar performance with a slight advantage of LC in the case of mapping VNRs with massive requirements. Overall, the obtained results in Figs. 6 and 7 are not linear due to the fact that the VNRs are

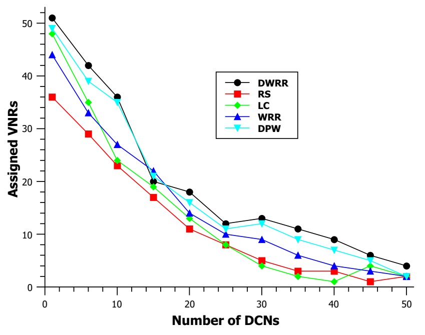

Fig. 6. VNRs assignment outcome over distributed DCNs.

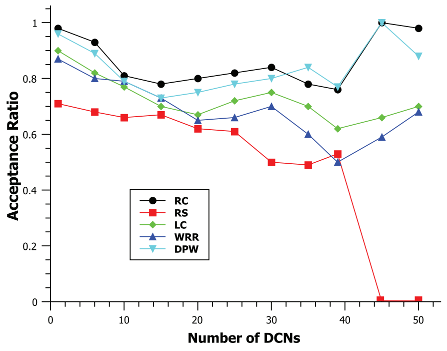

Fig. 7. VNRs acceptance rate over distributed DCNs.

not ranked (i.e., diverse VNR requirements in every VNE initiation).

The results obtained in Fig. 8 wind up the LB methods performance through three assessments. The outcome of solving the stress objective function shows a better average percentage across all DCNs for the RC policy while taking into account that the AR results. Therefore, even that RS scored the lowest stress rate (35% as best performance by notice), it is still considered a poor performance resulting from a low AR. A small cost with a higher AR can improve the R/C ratio. Hence, the best R/C performance scored by the RC method promotes better revenue. The RC method has scored the highest result of network utilization, employing almost every SNods/SLins during the mapping process. This is deemed as an essential feature for an operative LB policy.

Based on the prior result analysis, we have deduced the following claims (a method outperforms another using the sign $>$): DWRR/RC $>$ DPW $>$ LC $>$ WRR $>$ RS.

VI. Conclusion

NV is a fundamental key in cloud computing. The VNE process takes the lead as one of the most studied concepts in the NV core technologies. In this paper, we have investigated the trade-off between load balancing policies and the VNE process in a two-staged strategy. From this perspective, our work is devoted to solving the VNE problem in a distributed cloud architecture while collaborating with the SDN structure

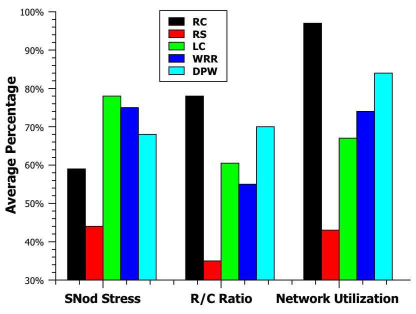

Fig. 8. Comparison of LB methods over VNE metrics.

(mainly through the SDN controller). The impact of the SDN integration is exhibited through the provision of complete network monitoring and network traffic handling. Overall, the proposed dual LB policy outperforms the LB techniques in comparison by spreading the loads over all of the substrate resources in the most rational way.

Perspectively, we plan to adjust the proposed LB policy into an energy-saving oriented LB technique within a distributed Virtual Data Centers (VDCs).

References

[1] K. Mishra and S. Majhi, "A state-of-art on cloud load balancing algorithms", *International Journal of computing and digital systems*, 9(2), pp.201-220, 2020.

[2] M. Katevenis, S. Sidiropoulos and C. Courcoubetis, "Weighted round-robin cell multiplexing in a general-purpose ATM switch chip," in *IEEE Journal on Selected Areas in Communications*, vol. 9, no. 8, pp. 1265-1279, Oct. 1991.

[3] C. K. Pyoung, S. J. Baek, "Joint load balancing and energy saving algorithm for virtual network embedding in infrastructure providers," *Computer Communications*, pp. 1-8, May. 2018.

[4] P. Zhang, "Incorporating energy and load balance into virtual network embedding process," *Computer Communications*, pp. 80-8, Sep. 2018.

[5] Z. Li, F. Yuan, L, Ma, "A load balancing algorithm for solving multi-objective virtual network embedding," *Transactions on Emerging Telecommunications Technologies*, Aug. 2020.

[6] A. Ghasemi, A.T. Haghighat, "A multi-objective load balancing algorithm for virtual machine placement in cloud data centers based on machine learning". *Computing*, 102(9), pp.2049-2072, 2020.

[7] P. Zhang, F. Liu, C. Jiang, A. Benslimane, J.L. Gorricho and J. Serrat-Fernández, "A Multi-Domain VNE Algorithm Based on Load Balancing in the IoT Networks". *Mobile Networks and Applications*, pp.1-15, 2021.

[8] A. Fischer, J. F. Botero, M. T. Beck, H. de Meer and X. Hesselbach, "Virtual Network Embedding: A Survey," in *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 1888-1906, 2013.

[9] "Software-Defined Networking (SDN) Definition," Accessed on: Aug. 23, 2021. [Online]. Available: https://www.opennetworking.org/sdn-definition/

[10] 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," 2015 15th *IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, pp. 475-484, 2015.

[11] R. N. Calheiros, R. Ranjan, A. Beloglazov, D. A. De Rose, R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software : Practice and experience*, pp. 23-50, Jan. 2011.

[12] M. Al-Fares, A. Loukissas, A. Vahdat, " A scalable, commodity data center network architecture," *ACM SIGCOMM computer communication review*, pp. 63-74, Aug. 2008.