LitE: Load Balanced Virtual Data Center Embedding for Energy Efficiency in Data Centers

Preetham N
National Institute of Technology Karnataka
Surathkal, India
preetham.nagaraju@gmail.com

Sourav Kanti Addya
National Institute of Technology Karnataka
Surathkal, India
souravkaddya@nitk.edu.in

Keerthan Kumar T G
Siddaganga Institute of Technology
Tumkur, India
keerthanswamy@gmail.com

Saumya Hegde
National Institute of Technology Karnataka
Surathkal, India
saumya@nitk.edu.in

Abstract

Network virtualization (NV) enables efficient management of physical network (PN) resources by partitioning them into virtual data center requests (VDCRs), consisting of interconnected virtual machines (VMs) and virtual links (VLs). A key challenge in NV is virtual data center embedding (VDCE), which allocates PN resources to VMs and VLs and is $\mathbb{NP}$-hard problem. Existing VDCE strategies often fail to balance energy efficiency and resource distribution, leading to sub-optimal solutions with higher energy consumption in data centers (DCs). This work presents LitE, a load-balanced VDCE strategy focused on minimizing energy consumption in single-domain PN. LitE uses a resource management strategy that considers server utilization, overloading probability, and energy consumption to select suitable servers for VM embedding. It then applies Dijkstra's shortest path algorithm for VL embedding to optimize energy use. Experiments show LitE improves energy efficiency by 15% compared to baseline methods through better resource utilization.

CCS Concepts

• **Networks → Cloud computing**.

Keywords

Network virtualization, virtual data center embedding, energy efficiency, load balance, resource management.

ACM Reference Format:
Preetham N, Sourav Kanti Addya, Keerthan Kumar T G, and Saumya Hegde. 2025. LitE: Load Balanced Virtual Data Center Embedding for Energy Efficiency in Data Centers. In *26th International Conference on Distributed Computing and Networking (ICDCN 2025), January 04–07, 2025, Hyderabad, India.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3700838.3700849

This work is licensed under a Creative Commons Attribution International 4.0 License.

ICDCN 2025, January 04–07, 2025, Hyderabad, India
© 2025 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-1062-9/25/01
https://doi.org/10.1145/3700838.3700849

1 Introduction

Network virtualization (NV) is a crucial enabler for the growing demands of modern Internet technology. It offers benefits such as isolation, improved physical network (PN) utilization, and security [1, 16]. By leveraging NV, service providers (SPs) can logically partition PN resources into independent virtual data center requests (VDCRs), allowing for more efficient management of network resources. For instance, Fig. 1 represents the VDCR belonging to a sample real-time applications such as hosting websites, online games, and video streaming from geographically distributed users [10]. In Fig. 1, VDCR comprises four virtual machines (VMs) and four interconnected virtual links (VLs). The numerical value 4 associated with VM $v_{1,1}$ represents its resource demand regarding computational resource blocks (CRBs). One CRB unit equating to one CPU core and 512 MB of RAM [16, 18]. Similarly, the numerals associated with VLs represent the minimum communication bandwidth demand. One of the primary challenges in NV is allocating the required physical resources to VDCR components, i.e., VMs and VLs. This process is known as virtual data center embedding (VDCE). It comprises two sub-problems: first, VM embedding, which involves assigning server resources to VMs, and second, VL embedding, which maps physical paths to the VLs connecting VMs. Both of these sub-problems are proven to be $\mathbb{NP}$ hard [7] [17].

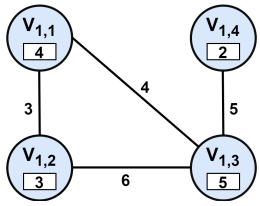

Figure 1: An instance of a virtual data center request.

To address this problem, many recent VDCE approaches focus on achieving objectives such as improving the revenue-to-cost ratio [16], increasing acceptance ratio [16], minimizing embedding costs [14], and energy minimization [8, 19]. However, these approaches overlook the importance of effective load distribution to achieve energy minimization in data centers (DCs), which is essential for modern Internet technology. In this regard, few research works have been conducted to address this objective of the VDCE problem. Fischer *et al.* in [4] introduced a strategy to map multiple VMs onto

the same physical server, preferring a server with lower power consumption and selecting energy-efficient paths. However, it failed to account for resource load dependency and scalability under overloaded conditions, leading to poor resource utilization and Quality of Service (QoS). To address this limitation, Rodriguez *et al.* in [13] proposed power-on-demand and live migration to minimize energy consumption while ensuring QoS and balancing the network load. However, it consumes high execution time and computational overhead due to frequent migrations. Further, Zhang *et al.* in [19] developed a VDCE strategy to improve energy by leveraging Gaussian distribution and diurnal patterns. Although it achieved energy savings, it faced challenges with modeling complexity and consumes more execution time for more extensive networks. The authors Lin *et al.* in [8] proposed a VDCE approach using an Integer Linear Programming (ILP) formulation to minimize costs. However, it is computationally complex and limited to smaller scenarios. Later, Pham *et al.* in [12] introduced a congestion-aware and energy-aware embedding strategy using the weighted constraint method. It tries to minimize the energy by putting inactive servers to sleep and mitigating congestion by dispersing traffic across multiple paths. However, it exhibited inefficiencies due to a fixed congestion ratio, resulting in degraded performance. On the other hand, the authors in [5] presented the embedding strategy by combining spectral clustering with field theory. This model enhanced network performance but faced clustering complexity, which led to underutilized resources. Additionally, the authors of [3] claim that the DC servers often operate at only 15–20% of CPU capacity despite consuming up to 70% of their peak energy when idle, leading to inefficiency and higher operational costs. Further, `Amazon` reports that 42% of its operational costs are due to DC energy use [15]. Globally, ICT infrastructure consumes 10% of the world's energy, with US DC alone consuming 1.4% of national electricity in 2010 and 1.8% in 2014. IT energy consumption could reach 13% by 2030, with DC electricity usage by 15–20% annually [11].

From the above literature, the following limitations still exist: (*i.*) *Existing works are less scalable* (*ii.*) *More energy consumption due to poorer embedding mechanism* (*iii.*) *Computationally complex model* (*iv.*) *Increased execution time* and (*v.*) *Lack of effective load distribution leads to poor PN utilization.* In order to tackle these limitations, this work introduces a greedy-based, two-stage heuristic VDCE approach called LitE. The key **contributions** of this work are as follows: (**1**.) This work introduces a framework called LitE for the VDCE problem. It generates a *spine-leaf* topology-based PN, typically called a full-meshed `Clos` architecture [16]. The proposed work aims to minimize the overall energy consumption in DC through effective load-balancing and thereby improve the overall performance of the network. (**2**.) LitE offers an efficient resource management (ERM) component to evaluate the VM embedding benefits of the server by considering server utilization, server overloading probability, and server energy consumption. Using this embedding data, VM embedding is carried out, followed by a VL embedding using Dijkstra's shortest path algorithm. (**3**.) To test LitE effectiveness, we integrated three state-of-the-art VDCE strategies (*i.*) Congestion-Aware, Energy-Aware Virtual Network Embedding (CEVNE) [12], (*ii.*) Dynamic Region of Interest (DROI) [5] and (*iii.*) First Fit algorithm. Simulation results show that LitE improves DC overall energy efficiency by minimizing up to 15% of energy consumption compared to baselines while improving PN resource utilization through load-balancing.

2 System Model

This section provides a detailed overview of the LitE architecture as depicted in Fig. 2 and its components, such as virtual data center request, physical network, and manager module.

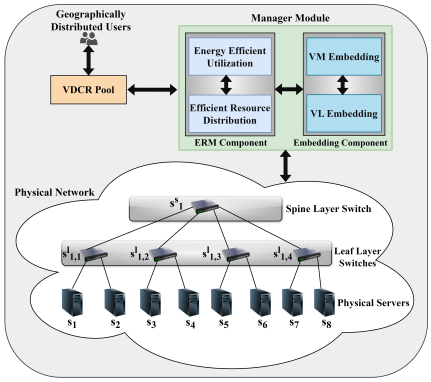

Figure 2: The proposed LitE architecture and its components.

Virtual Data Center Request (VDCR): A pool of VDCRs to be embedded is captured in the set as $\mathbb{G}_v = \{\mathbb{G}_1, \mathbb{G}_2, \ldots, \mathbb{G}_i, \ldots\}$. Each VDCR $\mathbb{G}_i \in \mathbb{G}_v$ is represented as an undirected weighted graph and captured in $\mathbb{G}_i = (\mathbb{N}_i, \mathbb{L}_i)$, where $\mathbb{N}_i = \{v_{i,1}, v_{i,2}, \ldots\}$ is the set of VMs and $|\mathbb{N}_i|$ captures the aggregate number of VMs. Similarly, $\mathbb{L}_i = \left\{e^i_{1,1}, e^i_{1,2}, \ldots, e^i_{j,j'}, \ldots\right\}$ is the set of VLs and $|\mathbb{L}_i|$ denote the total number of VLs. Let $r_{i,j}$ be the resource demand of a VM $v_{i,j} \in \mathbb{N}_i$ expressed in the form of CRB demand. On the other hand, the bandwidth resource demand of a VL $e^i_{j,j'} \in \mathbb{L}_i$ between VMs $v_{i,j}$ and $v_{i,j'}$ is captured as $r(e^i_{j,j'})$.

Physical Network (PN): The *spine-leaf* topology-based PN is modeled as an undirected weighted graph $\mathbb{G}_p = (\mathbb{N}_p, \mathbb{L}_p)$. The set of nodes captured in $\mathbb{N}_p = \mathbb{N}_s \cup \mathbb{N}_r$, where $\mathbb{N}_s$ is the set of servers, i.e., $\mathbb{N}_s = \{s_1, s_2, s_k, \ldots\}$, and $\mathbb{N}_r = \{\mathbb{N}_{sl}, \mathbb{N}_{ll}\}$ represents the set of routers/switches at each layer. Note that an entry $s^s_p \in \mathbb{N}_{sl}$ can be identified as a spine switch (from the superscript 's'), whereas the subscript 'p' indicates a number of spine switch. Similarly, an entry $s^l_{p,q} \in \mathbb{N}_{ll}$ specifies a leaf switch 'q' connected to the spine switch 'p'. Note that the superscript 'l' indicates that the switch corresponds to the leaf layer. The initial and residual available CRB capacity of a server $s_k \in \mathbb{N}_s$ is represented as c^{a}_k and c^{r}_k, respectively. It is assumed that the switches have very limited computing power, merely enough to handle packet forwarding. The set $\mathbb{L}_p = \{e_1, e_2, e_l, \ldots\}$ captures the physical links, wherein each link $e_l \in \mathbb{L}_p$ has an available bandwidth capacity $c(e_l)$. Successfully embedding a VL means finding a suitable physical path meeting bandwidth requirements. Let $P_{s_k,s_{k'}}$ be the set of all simple paths between servers s_k and $s_{k'}$ of which VMs $v_{i,j}$ and $v_{i,j'}$ mapped respectively. A specific path $p_{s_k,s_{k'}} \in P_{s_k,s_{k'}}$ which consists of multiple physical links, can successfully embed the VL $e^i_{j,j'}$, *iff*, $c(e_l) \geq r(e^i_{j,j'}), \forall e_l \in p_{s_k,s_{k'}}$.

Manger Module: This module comprises the ERM and embedding components. The ERM is responsible for determining the

most suitable server for each VM, focusing on two key factors: (*i.*) Energy-efficient utilization and (*ii.*) Efficient resource distribution. An approximate linear or sub-linear correlation exists between the energy consumption of a server and its resource utilization [2]. Consequently, we start with estimating the energy consumption of each server. This estimation is based on the server's idle and full energy levels, scaled by its resource utilization. ERM selects the server with the lowest computed energy consumption for VM embedding. The second aspect is assessed through the server overloading probability and is computed using the resource's mean and standard deviation across servers in the DC. The server with the lowest overloading probability is selected to balance the server's resource load. Hence, it ensures the best distribution of workloads across the DC. Finally, the ERM component considers a server with minimum energy consumption and minimum overloading probability to ensure that the most suitable server is chosen for VM placement. Subsequently, the embedding component leverages server information from the ERM to perform two-stage VM embedding followed by a VL embedding using Dijkstra's shortest path algorithm [6]. The manager module is also responsible for tracking the acceptance and rejection status of VDCRs, as well as monitoring the current status of available and used PN resources.

3 Problem Formulation

This section delivers the formulation of LitE objective and its associated constraints.

Minimization of Energy: To minimize overall energy usage in the DC, the server with the lowest energy consumption is selected. The server s_k's energy consumption $\mathbb{E}_k(\mathbb{U}_k)$ is calculated based on its idle and full energy usage scaled by its resource utilization. The same is defined in Eq. (1), where $\mathbb{U}_k$ holds the resource utilization of a server s_k, and it is computed using Eq. (2). It is the ratio of the overall resource demand $r_{i,j}$ of VMs across all VDCRs embedded on the server s_k to the available resource capacity c_k^{a} of s_k.

$$\mathbb{E}_k(\mathbb{U}_k)=\mathbb{E}_k^{\text{idle}} + (\mathbb{E}_k^{\text{full}} - \mathbb{E}_k^{\text{idle}}) * \mathbb{U}_k \tag{1}$$

$$\mathbb{U}_k = \frac{\sum_i^{|G_v|} \sum_j^{|N_i|} r_{i,j}}{c_k^a}, \forall \mathbb{Y}(v_{i,j}, s_k) = 1 \; s_k \in \mathbb{N}_s \tag{2}$$

The energy consumption of a server s_k operating at full capacity of resource is denoted as $\mathbb{E}_k^{\text{full}}$=300$W$. It is important to highlight that a server continues to consume energy even when it is not handling any load. Hence this idle energy consumption is represented by $\mathbb{E}_k^{\text{idle}}$=150$W$. Therefore, the overall objective of LitE is represented in Eq. (3a), subject to diverse constraints in Eqs. (3b) to (3f).

$$\min \sum_{k\in\mathbb{N}_s}^{|\mathbb{N}_s|} \mathbb{E}_k(\mathbb{U}_k) \tag{3a}$$

$$\text{s.t.} \quad r_{i,j} \le c_k^{\mathrm{r}} \tag{3b}$$

$$r(e^i_{j,j'}) \le c(e_l); \forall e_l \in p_{s_k,s_{k'}} \tag{3c}$$

$$\mathbb{Y}(v_{i,j}, s_k) = \begin{cases} 1 & \text{if } v_{i,j} \text{ is assigned to } s_k \\ 0 & \text{otherwise} \end{cases} \tag{3d}$$

$$\mathbb{Y}(e^i_{j,j'}, p_{s_k,s_{k'}}) = \begin{cases} 1 & \text{if } c(e_l) \ge r(e^i_{j,j'}), \forall e_l \in p_{s_k,s_{k'}} \\ 0 & \text{otherwise} \end{cases} \tag{3e}$$

$$\mathbb{Y}(v_{i,j}, s_k) \wedge \mathbb{Y}(v_{i,j'}, s_k) \neq 1 \tag{3f}$$

Algorithm 1: Embedding Strategy (ES)

Input: $\mathbb{G}_v$, $\mathbb{G}_p$
Result: $\mathcal{M}$
Initialize: $c_k^{\mathrm{r}} = c_k^{\mathrm{a}}, \forall s_k \in \mathbb{N}_s; \lambda(v_{i,j}) = \Phi, \forall v_{i,j} \in \mathbb{N}_i;$
$\lambda(s_k) = \Phi, \forall s_k \in \mathbb{N}_s;$ free$[v_{i,j}] = T$

```
for each 𝔾_i ∈ 𝔾'_v do
    free[𝔾_i] = T
𝔾'_v = Arrival(𝔾_v)
while 𝔾'_v ≠ Φ do
    𝔾_i = First(𝔾'_v)
    while ∃v_{i,j} ∈ 𝔾_i | free[v_{i,j}] = T do
        Γ_min = ∞ ; // Tracks the server with minimum energy cost
        s_best = Φ ;  // Initialize best server as null initially
        for each s_k ∈ ℕ_s do
            if r_{i,j} ≤ c_k^r and 𝕐(v_{i,j}, s_best) ∧ 𝕐(v_{i,j'}, s_best) ≠ 1 then
                𝕌_k = (Σ_i^{|𝔾v|} Σ_j^{|ℕi|} r_{i,j}) / c_k^a
                μ = (1/|ℕs|) Σ_{i=1}^{|ℕs|} c_k^r
                δ = sqrt( 1/(|ℕs|−1) Σ_{i=1}^{|ℕs|} (c_k^r − μ)^2 )
                ℙ(s_k) = 1 − Φ( (c_k^r − r_{i,j} − μ) / δ )
                𝔼_k(𝕌_k) = 𝔼_k^idle + (𝔼_k^full − 𝔼_k^idle) * 𝕌_k
                Γ_k = 𝔼_k(𝕌_k) * e^{α·ℙ(s_k)}
                if Γ_k < Γ_min then
                    s_best = s_k
                    Γ_min = Γ_k
        if s_best ≠ Φ then
            Assign v_{i,j} to s_best
            c_k^r = c_k^r − r_{i,j}
            𝕐(v_{i,j}, s_best) = 1
        else
            reject[𝔾_i] = T
            Release_Resources(𝔾_i)
            break;
    foreach e^i_{j,j'} ∈ 𝕃_i do
        if Feasible_Path(ℳ(v_{i,j}), ℳ(v_{i,j'}), r(e^i_{j,j'})) then
            Reserve_Path(ℳ(v_{i,j}), ℳ(v_{i,j'}), r(e^i_{j,j'}))
        else
            reject[𝔾_i] = T
            free[𝔾_i] = F
            Release_Resources(𝔾_i)
            break;
return ℳ
```

Condition (3b) ensures that a VM $v_{i,j} \in \mathbb{N}_i$ with resource demand $r_{i,j}$ is assigned to a server s_k only if the available residual resources c_k^{r} of the server s_k are sufficient to satisfy the demand. Condition (3c) ensures a VL $e^i_{j,j'} \in \mathbb{L}_i$, connecting VMs $v_{i,j}$ and $v_{i,j'}$, meets the bandwidth demand $r(e^i_{j,j'})$ over the physical path $p_{s_k,s_{k'}}$. These conditions prevent resource contention and embedding failures by ensuring VMs and VLs demand do not exceed resource limits. Additionally, we define the node indicator variable as in condition (3d), it is set to 1, in case VM $v_{i,j}$ is assigned to server s_k, otherwise, set to 0. Similarly, for $\mathbb{Y}(v_{i,j}, s_k)$=1 and $\mathbb{Y}(v_{i,j'}, s_{k'})$=1, we define a path indicator variable $\mathbb{Y}(e^i_{j,j'}, p_{s_k,s_{k'}})$ as in condition (3e). Condition (3f) ensures no VMs from the same VDCR are placed on the same server; it helps in preventing a single point of server failure and supporting distributed services [16].

4 Proposed Solution Approach

The proposed embedding strategy (ES) in LitE as illustrated in Algorithm 1, takes VDCRs $\mathbb{G}_v$ and PN $\mathbb{G}_p$ as an input. In the beginning, available server resource c_k^{a} is initialized to c_k^{r}. Mappings of all VM $\lambda(v_{i,j})$ and server $\lambda(s_k)$ are initialized to null. The algorithm marks all VMs of the VDCRs as free, i.e., free$[v_{i,j}]$ = $True$. Initially, the algorithm marks all the VDCRs are free by initializing free$[\mathbb{G}_i]$ = $True$ from Steps (1-2). The VDCR is processed in the order of their arrival, and the same is captured in $\mathbb{G}'_v$ from Step (3) of algorithm 1. The VDCR to be processed is not empty; then the first VDCR is picked and assigned to $\mathbb{G}_i$ from Steps (4-5) of algorithm 1. Initially, $\Gamma_{\min}$ is set to ∞ and s_{best} is set to Φ to represent that no server has been selected and no minimum energy cost has been determined yet. These values are updated as the algorithm evaluates each server's energy consumption and overloading probability from Steps (7-8) of algorithm 1. From Steps (9-15) of algorithm 1, for each server we compute server resource utilization $\mathbb{U}_k$, mean μ, standard deviation δ, server overloading probability $\mathbb{P}(s_k)$ and server energy consumption $\mathbb{E}_k(\mathbb{U}_k)$ is evaluated for the VM to be embedded. Here, the mean μ indicates the average amount of unused resource capacity available across all servers in the DC. The standard deviation δ of residual resource capacity at the DC measures how much the residual resource capacity deviates from the average μ. Server overloading probability $\mathbb{P}(s_k)$ is the likelihood that checks the server's resource demands will not exceed its available capacity, otherwise leading to potential performance degradation. These statistics assess server overload risk and its resource distribution, aiding in embedding the next VDCRs associated VMs. This approach is essential for effective resource allocation in DCs, enhancing performance, supporting load balancing, and minimizing performance degradation from resource contention. Using these key factors, Γ_{k} is determined from Step (16) of algorithm 1, which balances the energy consumption and the risk of server overloading. The server overloading probability $\mathbb{P}(s_k)$ for a server ranges from 0 to 1. When resource demand is low, $\mathbb{P}(s_k)$ is near 0, making it hard to distinguish between under-loaded servers. To address this, the probability range is modified using an exponential function, where any increase in $\mathbb{P}(s_k)$ leads to a rapid escalation in the objective function's value. The parameter $\alpha = 0.5$ is employed to control the impact of resource overloading probability on the objective function. This adjustment not only aids in accurately identifying the efficient server but also contributes to the objective of efficient energy minimization through load balancing. Next, the server s_k will be selected as the best server s_{best} for VM embedding, *iff*, Γ_{k} value is less than the initial $\Gamma_{\min}$ value. This will provide LitE to select the least energy consumed server $\mathbb{E}_k(\mathbb{U}_k)$ and lowest server overloading probability $\mathbb{P}(s_k)$ server, which helps to achieve the desired objective from Steps (17-19) of algorithm 1. The algorithm updates the server's resources accordingly and marks the VM as assigned by updating the embedding status from Steps (20-23) of algorithm 1. Otherwise, the VDCR is marked as rejected, and allotted resources are released from Steps (24-27) of algorithm 1. Eventually, we perform VL embedding by identifying the feasible path using Dijkstra's shortest path algorithm using Steps (28-30) of algorithm 1. Otherwise, mark the VDCR as rejected, release allocated resources, and move to the next VDCR for embedding from Steps (31-35) of algorithm 1. Finally, the mapping result is captured in $\mathcal{M}$ as in Step (36) of algorithm 1. The overall time complexity of the **LitE** is computed as $O(|\mathbb{N}_i| \times |\mathbb{N}_s| + |\mathbb{L}_i| \times |\mathbb{N}_s| \log |\mathbb{N}_s|)$.

5 Simulation Setup and Evaluation

We have performed simulations using `Mininet` [16], and the source code is available at [9]. A single-domain *spine-leaf* topology-based PN with 30 servers and 36 links is used in LitE. The resource capacities of the servers are uniformly distributed in the range U[200, 1000]. The bandwidth link capacities from server to leaf switches and from leaf switches to spine switches (in *Gbps*) are within the range U[500, 1000]. Each VDCR has VMs ranging from U[2, 10], with a link connectivity probability of 0.4, and VDCRs arrive at Poisson-distributed λ=0.4 [16]. The VM resource demand and VLs bandwidth demand are uniformly distributed within the range U[1, 10] and U[1, 5], respectively. These experimental settings are based on works in [16, 18]. We considered five scenarios of VDCRs [100, 200, 300, 400, 500], each running ten times. Experiments ensured a 100% acceptance ratio as in [5] to demonstrate the energy consumption across the test scenarios. To ensure an accurate and fair evaluation of its performance, LitE is compared against objective baselines such as (*i.*) CEVNE [12], (*ii.*) DROI [5], and (*iv.*) First Fit. Fig. 3 depicts the overall ***energy consumption*** by the servers to embed the VDCRs by different techniques. LitE consistently utilizes the least energy and outperforms the baselines such as First Fit, DROI, and CEVNE. This is due to LitE server selection mechanism based on resource utilization and energy metrics, which prevent high-energy spikes and attain effective load balancing. On the other hand, CEVNE consumes slightly more energy consumption. This is due to a lack of load balancing, resulting in more server consumption. DROI prioritizes dynamic regions over energy efficiency, which is the reason for its decreased performance than LitE and CEVNE. Alternatively, First-fit uses a greedy-based strategy, leading to poor server utilization and higher energy usage than all other approaches. Fig. 4 depicts the ***active node stress*** across different techniques. It is the fraction of the total consumed resources to the initially available resources at the server before embedding. In Fig. 4, LitE achieves moderate node stress by considering server utilization during VM embedding, which distributes load more evenly across servers. DROI and First Fit have higher node stress because they focus on specific regions or servers, leading to imbalance. CEVNE maintains the lowest stress but risks underutilizing server resources. Fig. 5 depicts the ***active link stress*** across the baselines. It is the fraction of the used bandwidth on physical links to the total available bandwidth on those links. It refers to the stress on physical links that are currently allocated VLs and serving VDCRs. LitE moderates link stress by balancing the network load. DROI has slightly higher link stress due to uneven distribution, while First Fit exhibits the highest stress from poor server selection. CEVNE has the least link stress but underuses physical links, leading to lower performance. Fig. 6 illustrates the ***number of servers*** utilized across various test cases. LitE uses fewer servers than CEVNE but slightly more than DROI and the First-fit approach. This is because LitE allocates server resources based on the resource overloading probability during VM embedding, optimizing for resource efficiency. Fig. 7 shows the ***number of links*** required to map VDCRs in different scenarios. LitE evenly distributes the network load, maintaining

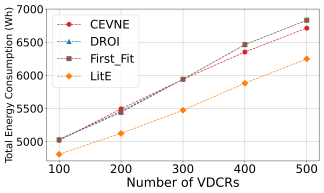

Figure 3: Energy Consumption *vs.* Number of VDCRs.

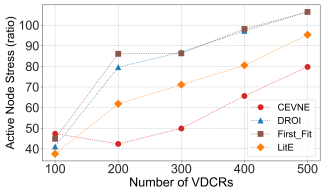

Figure 4: Active Node Stress *vs.* Number of VDCRs.

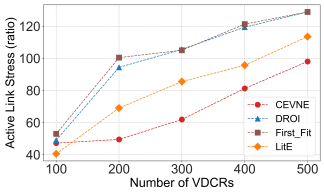

Figure 5: Active Link Stress *vs.* Number of VDCRs.

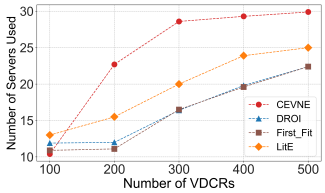

Figure 6: Servers used *vs.* Number of VDCRs.

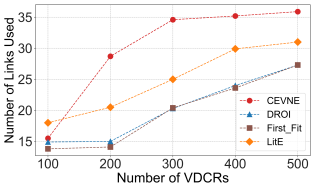

Figure 7: Links used *vs.* Number of VDCRs.

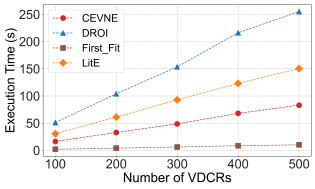

Figure 8: Execution Time *vs.* Number of VDCRs.

stable physical link usage and improving performance. In contrast, CEVNE uses more links, leading to inefficiencies, while DROI and First-fit use fewer links, increasing node stress and energy consumption and potentially lowering performance. The overall ***execution time*** consumed across the baseline is captured in Fig. 8. LitE takes slightly longer than CEVNE and First-fit. This is due to its focus on finding load-balanced and energy-efficient servers during the VM embedding process. DROI, however, experiences a sharp increase in execution time because of its complex clustering mechanism. At the same time, First-fit benefits from a faster execution time due to its straightforward server selection strategy.

6 Conclusion and Future Work

This work introduces a VDCE framework LitE for minimizing energy consumption by effectively balancing load in a PN. LitE leverages key factors such as server resource utilization and the probability of server overloading, thereby trying to reduce DC overall energy consumption. Consideration of server overloading probability during VM embedding, followed by a VL embedding using the shortest path algorithm, significantly enhances both energy efficiency and resource management. The simulation results demonstrate that the LitE considerably minimizes the energy consumption in DC by 15% compared to baselines. In the future, we plan to integrate LitE with software-defined networking-specific features to validate the model's performance in more realistic and dynamic environments and also to test and validate the working of LitE over commercial cloud platforms such as AWS and Azure.

References

[1] Iqbal Alam, Kashif Sharif, Fan Li, Zohaib Latif, Md Monjurul Karim, Sujit Biswas, Boubakr Nour, and Yu Wang. 2020. A survey of network virtualization techniques for Internet of Things using SDN and NFV. *ACM Computing Surveys (CSUR)* 53, 2 (2020), 1–40. https://doi.org/10.1145/3379444

[2] Anton Beloglazov and Rajkumar Buyya. 2012. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. *Concurrency and Computation: Practice and Experience* 24, 13 (2012), 1397–1420. https://doi.org/10.1002/cpe.1867

[3] Md Hasanul Ferdaus, M. Manzur Murshed, Rodrigo N. Calheiros, and Rajkumar Buyya. 2017. Multi-objective, Decentralized Dynamic Virtual Machine Consolidation using ACO Metaheuristic in Computing Clouds. *CoRR* abs/1706.06646 (2017), –. http://arxiv.org/abs/1706.06646

[4] Andreas Fischer, Michael Till Beck, and Hermann De Meer. 2013. An approach to energy-efficient virtual network embeddings. In *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*. IEEE, Ghent, Belgium, 1142–1147.

[5] Mengyang He, Lei Zhuang, Shuaikui Tian, Guoqing Wang, and Kunli Zhang. 2020. DROI: Energy-efficient virtual network embedding algorithm based on dynamic regions of interest. *Computer Networks* 166 (2020), 106952. https://doi.org/10.1016/j.comnet.2019.106952

[6] Priyanka Kamboj and Sujata Pal. 2022. Energy-Aware Routing in SDN Enabled Data Center Network. In *2022 IEEE 21st International Symposium on Network Computing and Applications (NCA)*, Vol. 21. IEEE, Boston, MA, USA, 123–130. https://doi.org/10.1109/NCA57778.2022.10013588

[7] TG Keerthan Kumar, Sourav Kanti Addya, and Shashidhar G Koolagudi. 2024. InDS: Intelligent DRL Strategy for Effective Virtual Network Embedding of an Online Virtual Network Requests. *IEEE Access* 12 (2024), 94843 – 94860. https://doi.org/10.1109/ACCESS.2024.3424474

[8] Rongping Lin, Shan Luo, Haoran Wang, and Sheng Wang. 2017. Energy-aware virtual network embedding in flexi-grid networks. *Optics Express* 25, 24 (2017), 29699–29713. https://doi.org/10.1364/OE.25.029699

[9] LitE. 2024. LitE. https://anonymous.4open.science/r/LitE-D90B/ Accessed: 2024-08-08.

[10] Meilian Lu and Meng Li. 2024. A multiple QoS metrics-aware virtual network embedding algorithm. *Comput. J.* 67, 3 (2024), 1171–1186. https://doi.org/10.1093/comjnl/bxad050

[11] Sudipto Mondal, Fashat Bin Faruk, Dibosh Rajbongshi, Mohammad Masum Khondhoker Efaz, and Md Motaharul Islam. 2023. GEECO: Green Data Centers for Energy Optimization and Carbon Footprint Reduction. *Sustainability* 15, 21 (2023), 15249. https://doi.org/10.3390/su152115249

[12] Minh Pham, Doan B Hoang, and Zenon Chaczko. 2019. Congestion-aware and energy-aware virtual network embedding. *IEEE/ACM Transactions on Networking* 28, 1 (2019), 210–223. https://doi.org/10.1109/TNET.2019.2958367

[13] Esteban Rodriguez, Gustavo P Alkmim, Nelson LS da Fonseca, and Daniel Macêdo Batista. 2015. Energy-aware mapping and live migration of virtual networks. *IEEE Systems Journal* 11, 2 (2015), 637–648. https://doi.org/10.1109/JSYST.2015.2467159

[14] Anurag Satpathy, Manmath Narayan Sahoo, Arun Kumar Sangaiah, Chittaranjan Swain, and Sambit Bakshi. 2022. CoMap: An efficient virtual network re-mapping strategy based on coalitional matching theory. *Computer Networks* 216 (2022), 109248. https://doi.org/10.1016/j.comnet.2022.109248

[15] Fei Teng, Lei Yu, Tianrui Li, Danting Deng, and Frédéric Magoulès. 2017. Energy efficiency of VM consolidation in IaaS clouds. *The Journal of Supercomputing* 73 (2017), 782–809. https://doi.org/10.1007/s11227-016-1797-5

[16] Keerthan Kumar TG, Sourav Kanti Addya, Anurag Satpathy, and Shashidhar G Koolagudi. 2023. NORD: NOde Ranking-based efficient virtual network embedding over single Domain substrate networks. *Computer Networks* 225 (2023), 109661. https://doi.org/10.1016/j.comnet.2023.109661

[17] Keerthan Kumar TG, KB Aneesh, Anagha Siddheshwar, Amulya Marali, Anupama Kamath, Shashidhar G Koolagudi, and Sourav Kanti Addya. 2024. DeepVNE: Deep Reinforcement and Graph Convolution Fusion for Virtual Network Embedding. In *2024 16th International Conference on COMmunication Systems & NETworkS (COMSNETS)*. IEEE, Bengaluru, India, 633–636. https://doi.org/10.1109/COMSNETS59351.2024.10426879

[18] Keerthan Kumar TG, Shivangi Tomar, Sourav Kanti Addya, Anurag Satpathy, and Shashidhar G Koolagudi. 2024. EFraS: Emulated framework to develop and analyze dynamic Virtual Network Embedding strategies over SDN infrastructure. *Simulation Modelling Practice and Theory* 134 (2024), 102952. https://doi.org/10.1016/j.simpat.2024.102952

[19] Zhongbao Zhang, Sen Su, Junchi Zhang, Kai Shuang, and Peng Xu. 2015. Energy aware virtual network embedding with dynamic demands: Online and offline. *Computer Networks* 93 (2015), 448–459. https://doi.org/10.1016/j.comnet.2015.09.036