LEEV: Load-Aware and Energy-Efficient VNF Deployment in NFV-Enabled Networks

You-Chiun Wang and Pei-Chin Tsai
Department of Computer Science and Engineering,
National Sun Yat-sen University, Kaohsiung, 804, Taiwan
Email: ycwang@cse.nsysu.edu.tw; one23807659@gmail.com

***Abstract*—*Network function virtualization (NFV)* decouples network functions from dedicated hardware by implementing them as software-based *virtual network functions (VNFs)*, thereby enabling flexible network resource management. Network services are provisioned through *service function chains (SFCs)* composed of multiple VNFs. Given SFC requests and *physical machines (PMs)*, this paper investigates the VNF deployment problem with three objectives: maximizing the *service acceptance ratio (SAR)*, minimizing the *total energy consumption (TEC)*, and balancing the load among PMs. We then propose a *load-aware and energy-efficient VNF deployment (LEEV)* scheme that dynamically adapts its deployment strategy based on the system load. Under light-load conditions, LEEV deploys adjacent VNFs of a given SFC on one PM to reduce energy consumption, while under heavy-load conditions, it can shift toward load-balancing deployment without sacrificing energy efficiency. Simulation results reveal that LEEV achieves a higher SAR, lower TEC, and improved load balance.**

***Index Terms*—deployment, energy, load balance, network function virtualization (NFV), virtual network function (VNF).**

I. Introduction

In large-scale networks, most network services are provided by middleboxes—typically expensive, purpose-built hardware devices that can process and transform network traffic based on packet content. Service orchestration requires steering traffic through a sequence of selected middleboxes, forming *service function chains (SFCs)*. Such tight coupling between network functions and middleboxes not only increases the capital and operational expenditures of service providers but also makes network resource management rigid and complex [1].

To overcome the limitations, ETSI has developed standards for *network function virtualization (NFV)* [2]. NFV leverages virtualization technologies to replace traditional middleboxes with software-based *virtual network functions (VNFs)*, which facilitates their deployment and execution on commodity *physical machines (PMs)*. VNFs can migrate across PMs, and SFCs can be scaled by adding or removing VNFs as needed. As a core technology, NFV is crucial for 5G and 6G networks [3].

A key challenge in NFV is the efficient assignment of VNFs to PMs, known as the *VNF deployment problem*. This NP-hard problem significantly affects the performance and efficiency of NFV-enabled networks. To address this challenge, this paper studies the problem by jointly considering three objectives: 1) maximizing the *service acceptance ratio (SAR)*; 2) minimizing the *total energy consumption (TEC)*; and 3) balancing the load across PMs. The first objective is generally achieved by improving PM resource utilization [4], while the latter two aim to reduce operational expenditures and mitigate performance degradation caused by overloaded PMs [5].

However, these objectives often conflict. In particular, deploying VNFs on fewer PMs can improve resource utilization but may cause load imbalance, with some PMs heavily loaded and others idle. Conversely, distributing VNFs across multiple PMs improves load balancing, but VNFs belonging to the same SFC may reside on different PMs, increasing inter-PM traffic and energy consumption of network links.

Consequently, we propose a *load-aware and energy-efficient VNF deployment scheme*, referred to as LEEV, that dynamically adapts deployment decisions based on the system load. Under light-load conditions, LEEV employs a bundling-based deployment strategy to co-locate multiple adjacent VNFs on the same PM, which improves resource utilization and reduces energy consumption caused by inter-VNF communication. As the system load increases, LEEV shifts toward load balancing among PMs while preserving energy efficiency. Simulations demonstrate that the LEEV scheme can achieve a high SAR, reduce TEC, and balance the load across PMs.

The rest of this paper is organized as follows: Section II reviews related work, Section III presents the system model, Section IV describes the LEEV scheme, and Section V evaluates its performance. Finally, Section VI concludes the paper.

II. Related Work

The problem of VNF deployment for improving PM resource utilization has attracted considerable attention. An integer linear programming model is proposed in [6] to maximize resource utilization and provider revenue, while norm-based criteria are adopted in [7] to match PM resources with SFC demands. The work [8] proposes first-fit, best-fit, and segment-based algorithms to increase deployed SFC weight. The studies in [9]–[11] improve resource utilization by considering service latency, workload prediction, and link consumption. However, aggressive VNF consolidation tends to concentrate VNFs on a limited number of PMs, resulting in load imbalance.

Both studies [12], [13] improve energy efficiency by reducing the number of working PMs, potentially imposing heavy workloads on active PMs. The study [14] adopts a centrality-based ranking method for VNF-to-PM mapping. Zhang et al. [15] propose a near-optimal placement algorithm to decrease PM and link energy consumption. However, these approaches largely rely on VNF consolidation to achieve energy efficiency. Varasteh et al. [16] employ blocking islands to abstract PMs and links, reducing computational complexity at the cost of potentially suboptimal deployment decisions.

Load balancing among PMs has also been studied. In [17], a max-min allocation strategy is used to assign as many VNFs as possible to the least-loaded PM. If a PM becomes overloaded, the study [18] moves part of its VNFs to other PMs. The work [19] scores each PM based on its resource status and that of neighboring PMs, triggering VNF migration upon overload. Xuan et al. [20] apply a deep Q-network for VNF deployment to improve load balance. In [21], a utility function is used to evaluate PM suitability for hosting VNFs, with differentiated deployment methods for the first and subsequent VNFs in an SFC. Although these approaches enhance load balancing by distributing VNFs across PMs, they may incur increased inter-PM traffic and energy consumption of links.

In contrast, our work adapts VNF deployment decisions depending on the system load, with the aim of jointly considering resource utilization, energy consumption, and load balancing.

III. System Model

We model the underlying infrastructure network as a graph $(\mathcal{P}, \mathcal{L})$, where $\mathcal{P}$ denotes the set of PMs and $\mathcal{L}$ denotes the set of links between PMs. For each PM $p_x \in \mathcal{P}$, $\Lambda_x^{\mathtt{R}}$ represents its capacity of resource type $\mathtt{R}$, where $\Lambda_x^{\mathtt{R}} \in \mathbb{Z}^+$ and $\mathtt{R} \in \{\mathtt{C}, \mathtt{M}\}$. Specifically, $\mathtt{C}$ and $\mathtt{M}$ denote computing and memory resources, commonly required by VNFs [22]. Link $l_{x,y} \in \mathcal{L}$ exists if PMs p_x and p_y are directly connected.

Let $\mathcal{S}$ denote the set of SFCs to be served. Each SFC $\mathcal{S}_i \in \mathcal{S}$ is represented as an ordered sequence of VNFs, denoted by $\{v_{i,1}, v_{i,2}, \cdots, v_{i,n}\}$. For each $v_{i,j} \in \mathcal{S}_i$, $\lambda_{i,j}^{\mathtt{R}} \in \mathbb{Z}^+$ denotes its demand for resource type $\mathtt{R}$. If $v_{i,j}$ and $v_{i,j+1}$ are deployed on different PMs, data is transmitted over one or more links, consuming bandwidth $b_{i,j}$. We define $b_{i,j} = 0$ if $v_{i,j}$ and $v_{i,j+1}$ are deployed on the same PM or if $v_{i,j}$ is the last VNF in $\mathcal{S}_i$.

According to the energy model in [15], TEC is composed of *PM energy consumption (PEC)* and *link energy consumption (LEC)*. PEC represents the energy consumed by PMs for basic operations and VNF hosting. Given the set $\mathcal{V}_x$ of VNFs hosted on p_x, the PEC of p_x is defined as

$$e_x^{\mathtt{PEC}} = \phi_x^{\mathtt{base}} + \sum\nolimits_{\mathtt{R}\in\{\mathtt{C},\mathtt{M}\}} \sum\nolimits_{v_{i,j}\in\mathcal{V}_x} \phi_x^{\mathtt{R}} \lambda_{i,j}^{\mathtt{R}}, \tag{1}$$

where $\phi_x^{\mathtt{base}}$ is the baseline energy consumption of p_x without hosted VNFs, and $\phi_x^{\mathtt{R}}$ is the additional energy consumption per unit of resource type $\mathtt{R}$. LEC is the energy consumed by links for inter-PM data transmission. Let $\mathcal{V}_{x,y}^{'}$ be the set of VNFs whose traffic traverses $l_{x,y}$. The LEC of $l_{x,y}$ is estimated by

$$e_{x,y}^{\mathtt{LEC}} = \psi_{x,y}^{\mathtt{idle}} + \frac{\sum_{v_{i,j}\in\mathcal{V}'_{x,y}} b_{i,j}}{b_{x,y}^{\mathtt{max}}} \left(\psi_{x,y}^{\mathtt{full}} - \psi_{x,y}^{\mathtt{idle}}\right), \tag{2}$$

where $\psi_{x,y}^{\mathtt{idle}}$ and $\psi_{x,y}^{\mathtt{full}}$ denote the energy consumption of $l_{x,y}$ under idle and full-load conditions, and $b_{x,y}^{\mathtt{max}}$ is the maximum bandwidth of $l_{x,y}$. Then, TEC can be expressed as

$$\mathrm{TEC} = \sum\nolimits_{p_x\in\mathcal{P}} e_x^{\mathtt{PEC}} + \sum\nolimits_{l_{x,y}\in\mathcal{L}} e_{x,y}^{\mathtt{LEC}}. \tag{3}$$

On the other hand, the load of PM p_x can be characterized by the proportion of resources consumed by the VNFs hosted on it (i.e., $\mathcal{V}_x$), which is calculated as follows:

$$\xi_x = \frac{1}{|\mathtt{R}|} \sum\nolimits_{\mathtt{R}\in\{\mathtt{C},\mathtt{M}\}} \left(\sum\nolimits_{v_{i,j}\in\mathcal{V}_x} \lambda_{i,j}^{\mathtt{R}} / \Lambda_x^{\mathtt{R}} \right). \tag{4}$$

The degree of load balance among all PMs can be measured using *Jain's fairness index (JFI)* [23], computed as follows:

$$\mathrm{JFI} = \left(\sum\nolimits_{p_x\in\mathcal{P}} \xi_x\right)^2 / \left(|\mathcal{P}| \sum\nolimits_{p_x\in\mathcal{P}} \xi_x^2\right). \tag{5}$$

JFI ranges from $1/|\mathcal{P}|$ to 1, and a higher value indicates better load balance.

The VNF deployment problem seeks to assign PMs in $\mathcal{P}$ to host VNFs of each SFC in $\mathcal{S}$, with three objectives:

$$\text{maxmize} \quad \frac{\sum_{p_x\in\mathcal{P}} \sum_{\mathcal{S}_i\in\mathcal{S}} \sum_{v_{i,j}\in\mathcal{S}_i} z_{i,j}^x}{\sum_{\mathcal{S}_i\in\mathcal{S}} |\mathcal{S}_i|}, \tag{6}$$

$$\text{minimize} \quad \mathrm{TEC}, \tag{7}$$

$$\text{maxmize} \quad \mathrm{JFI}. \tag{8}$$

In particular, Eq. (6) maximizes SAR, where $z_{i,j}^x$ is a binary variable indicating whether VNF $v_{i,j}$ is deployed on PM p_x. Since the operation of an SFC depends on its deployed VNFs, SAR may alternatively be defined as the ratio of accepted SFCs to the total number of SFCs. However, Eq. (6) further accounts for variations in the number of VNFs among SFCs, assigning greater weight to SFCs with more VNFs and thus providing a more representative evaluation. Eq. (7) minimizes TEC, while Eq. (8) balances loads among PMs in $\mathcal{P}$.

The constraints are defined as follows:

$$z_{i,j}^x \in \{0,1\}, \qquad \forall p_x \in \mathcal{P}, \forall v_{i,j} \in \mathcal{S}_i, \forall \mathcal{S}_i \in \mathcal{S}, \tag{9}$$

$$\lambda_{i,j}^{\mathtt{R}} \le \min_{p_x\in\mathcal{P}} \Lambda_x^{\mathtt{R}}, \quad \forall \mathtt{R} \in \{\mathtt{C},\mathtt{M}\}, \forall v_{i,j} \in \mathcal{S}_i, \forall \mathcal{S}_i \in \mathcal{S}, \tag{10}$$

$$\sum\nolimits_{p_x\in\mathcal{P}} z_{i,j}^x \le 1, \qquad \forall v_{i,j} \in \mathcal{S}_i, \forall \mathcal{S}_i \in \mathcal{S}, \tag{11}$$

$$\sum_{\mathcal{S}_i\in\mathcal{S}} \sum_{v_{i,j}\in\mathcal{S}_i} z_{i,j}^x \lambda_{i,j}^{\mathtt{R}} \le \Lambda_x^{\mathtt{R}}, \quad \forall p_x \in \mathcal{P}, \forall \mathtt{R} \in \{\mathtt{C},\mathtt{M}\}, \tag{12}$$

$$\sum\nolimits_{v_{i,j}\in\mathcal{V}'_{x,y}} b_{i,j} \le b_{x,y}^{\mathtt{max}}, \qquad \forall l_{x,y} \in \mathcal{L}. \tag{13}$$

Specifically, Eq. (9) enforces $z_{i,j}^x$ to be binary. Eq. (10) ensures that a VNF's resource demands do not exceed PM capacity. Eq. (11) prevents a VNF from being split across multiple PMs. Eq. (12) guarantees that the aggregate resource demands of hosted VNFs remain within PM capacity. Eq. (13) accounts for link bandwidth required for VNF data transmission. Table I summarizes the acronyms and notations.

IV. The Proposed LEEV Scheme

The LEEV scheme dynamically adapts its deployment strategy according to the system load, defined as follows:

$$\Psi_{\mathtt{sys}} = \sum\nolimits_{\mathtt{R}\in\{\mathtt{C},\mathtt{M}\}} w_{\mathtt{R}} \times \left(\frac{1}{|\mathcal{P}|} \sum\nolimits_{p_x\in\mathcal{P}} \frac{\mu_x^{\mathtt{R}}}{\Lambda_x^{\mathtt{R}}} \right). \tag{14}$$

In particular, $w_{\mathtt{R}}$ denotes the weight associated with resource type $\mathtt{R}$, where $0 \le w_{\mathtt{R}} \le 1$ and $\sum_{\mathtt{R}\in\{\mathtt{C},\mathtt{M}\}} w_{\mathtt{R}} = 1$. Moreover,

TABLE I
SUMMARY OF ACRONYMS AND NOTATIONS.

(a) Summary of acronyms

acronym	full name
BIH	blocking island hierarchy
CLVS	cost-aware and load-balancing VNF scheduling
JFI	Jain's fairness index
LBVD	load-balanced VNF deployment
LEC/PEC/TEC	link/PM/total energy consumption
LEEV	load-aware and energy-efficient VNF deployment
NFV	network function virtualization
PM	physical machine
SAR	service acceptance ratio
SFC	service function chain
VNF	virtual network function

(b) Summary of notations

notation	definition
$\mathcal{P}$, $\mathcal{L}$, $\mathcal{S}$	sets of PMs, links, and SFCs ($\mathcal{S}_i$: an SFC in $\mathcal{S}$)
$\mathcal{P}_{\texttt{SS}}$, $\mathcal{P}_{\texttt{CAN}}$	search space and set of candidate PMs
$\Lambda_x^{\texttt{R}}$, $\mu_x^{\texttt{R}}$	capacity and usage amount of PM p_x's resource type $\texttt{R}$
$\lambda_{i,j}^{\texttt{R}}$	demand of VNF $v_{i,j}$ for resource type $\texttt{R}$
$z_{i,j}^x$	binary variable to check if PM p_x hosts VNF $v_{i,j}$
$b_{i,j}$	bandwidth required by VNF $v_{i,j}$ to send data
$e_x^{\texttt{PEC}}$, $e_{x,y}^{\texttt{LEC}}$	PEC of PM p_x and LEC of link $l_{x,y}$
$\Psi_{\texttt{sys}}$, $\delta_{\texttt{sys}}$	system load and its threshold

$\mu_x^{\texttt{R}} = \sum_{\mathcal{S}_i \in \mathcal{S}} \sum_{v_{i,j} \in \mathcal{S}_i} z_{i,j}^x \lambda_{i,j}^{\texttt{R}}$ represents the usage amount of resource type $\texttt{R}$ on PM p_x. The system load in Eq. (14) is calculated as the sum of the weighted average resource utilization across all PMs in $\mathcal{P}$. In the absence of any preference for a particular resource type, we recommend assigning equal weights to different resources, that is, $w_{\texttt{C}} = w_{\texttt{M}} = 0.5$. The value of $\Psi_{\texttt{sys}}$ ranges from 0 to 1, where a larger value indicates that most PMs have fewer remaining available resources.

For each SFC $\mathcal{S}_i \in \mathcal{S}$, LEEV deploys its VNFs sequentially. The deployment strategy is determined by comparing system load $\Psi_{\texttt{sys}}$ with a threshold $\delta_{\texttt{sys}}$ ($0 < \delta_{\texttt{sys}} < 1$). If $\Psi_{\texttt{sys}} < \delta_{\texttt{sys}}$, LEEV uses the *bundling-based deployment strategy* described in Section IV-A. When $\Psi_{\texttt{sys}} \geq \delta_{\texttt{sys}}$, LEEV switches to the *load-balancing deployment strategy* presented in Section IV-B. Further discussion of LEEV is provided in Section IV-C.

A. Bundling-Based Deployment Strategy

Let $v_{i,j}$ denote the VNF (belonging to SFC $\mathcal{S}_i$) currently to be deployed. We define $\mathcal{P}_{\texttt{SS}} \subseteq \mathcal{P}$ as the subset of PMs whose hop counts to the PM hosting the preceding VNF $v_{i,j-1} \in \mathcal{S}_i$ do not exceed a threshold ρ, where $\rho \in \mathbb{Z}^+$. If $v_{i,j}$ is the first VNF in $\mathcal{S}_i$, then $\mathcal{P}_{\texttt{SS}} = \mathcal{P}$. Hence, the set $\mathcal{P}_{\texttt{SS}}$ represents the search space of candidate PMs for deploying $v_{i,j}$.

Algorithm 1 shows the pseudocode of the bundling-based deployment strategy. In line 1, $\mathcal{P}_{\texttt{CAN}}$ denotes the set of candidate PMs, which is initially set to $\emptyset$, while $\mathcal{V}_{\texttt{CAN}}$ consists of $v_{i,j}$ and the subsequent $K-1$ VNFs, where K is a small integer with $K > 1$. In line 2, the Boolean variable *bundle* indicates whether it is feasible to deploy all VNFs in $\mathcal{V}_{\texttt{CAN}}$ on one PM.

Lines 3–8 search for candidate PMs for deploying the VNFs in $\mathcal{V}_{\texttt{CAN}}$ when it contains multiple VNFs. In line 4, $\overline{\lambda}_{\texttt{R}}$ denotes the aggregate demand of resource type $\texttt{R}$ required by all VNFs in $\mathcal{V}_{\texttt{CAN}}$. The for-loop in lines 5–8 checks whether each PM

Algorithm 1: Bundling-based deployment

1 $\mathcal{P}_{\texttt{CAN}} \leftarrow \emptyset$, $\mathcal{V}_{\texttt{CAN}} \leftarrow \{v_{i,j}, v_{i,j+1}, \cdots, v_{i,j+K-1}\}$;
2 *bundle* $\leftarrow$ false;
3 **if** $|\mathcal{V}_{\texttt{CAN}}| > 1$ **then**
4     $\overline{\lambda}_{\texttt{R}} \leftarrow \sum_{v_{i,m} \in \mathcal{V}_{\texttt{CAN}}} \lambda_{i,m}^{\texttt{R}}$, $\forall \texttt{R} \in \{\texttt{C}, \texttt{M}\}$;
5   **foreach** $p_x \in \mathcal{P}_{\texttt{SS}}$ **do**
6   **if** $\Phi(p_x, \overline{\lambda}_{\texttt{R}}, v_{i,j-1}) =$ true **then**
7             $\mathcal{P}_{\texttt{CAN}} \leftarrow \mathcal{P}_{\texttt{CAN}} \cup \{p_x\}$;
8   *bundle* $\leftarrow$ true;
9 **if** *bundle* = false **then**
10   **foreach** $p_x \in \mathcal{P}_{\texttt{SS}}$ **do**
11   **if** $\Phi(p_x, \lambda_{i,j}^{\texttt{R}}, v_{i,j-1}) =$ true **then**
12             $\mathcal{P}_{\texttt{CAN}} \leftarrow \mathcal{P}_{\texttt{CAN}} \cup \{p_x\}$;
13 **if** $\mathcal{P}_{\texttt{CAN}} \neq \emptyset$ **then**
14   **if** $|\mathcal{V}_{\texttt{CAN}}| > 1$ *and bundle* = false **then**
15         $\mathcal{V}_{\texttt{CAN}} \leftarrow \{v_{i,j}\}$;
16     $p_y \leftarrow \arg\min_{p_x \in \mathcal{P}_{\texttt{CAN}}} f_{\texttt{TEC}}(p_x, \mathcal{V}_{\texttt{CAN}})$;
17   Deploy all VNFs in $\mathcal{V}_{\texttt{CAN}}$ on p_y;
18   Update $\mu_y^{\texttt{R}}$, $\forall \texttt{R} \in \{\texttt{C}, \texttt{M}\}$;
19 **else**
20   Deny the deployment of $v_{i,j}$;

$p_x \in \mathcal{P}_{\texttt{SS}}$ can accommodate all VNFs in $\mathcal{V}_{\texttt{CAN}}$. To this end, we define a checking function $\Phi(p_x, \overline{\lambda}_{\texttt{R}}, v_{i,j-1})$, which returns true if the following two conditions are satisfied: 1) $\mu_x^{\texttt{R}} + \overline{\lambda}_{\texttt{R}} \leq \Lambda_x^{\texttt{R}}$, $\forall \texttt{R} \in \{\texttt{C}, \texttt{M}\}$, i.e., p_x has sufficient residual resources; and 2) a path with sufficient link bandwidth exists for the preceding VNF $v_{i,j-1}$ to transmit its generated data from its hosting PM to p_x. The second condition is not considered if $v_{i,j}$ is the first VNF. Otherwise, the checking function returns false. Once p_x passes the check, it is added to $\mathcal{P}_{\texttt{CAN}}$, and the variable *bundle* is set to true.

Lines 9–12 deal with the case where the variable *bundle* is false. This situation may occur under two conditions: 1) $\mathcal{V}_{\texttt{CAN}}$ contains merely $v_{i,j}$, that is, $v_{i,j}$ is the last VNF in $\mathcal{S}_i$; or 2) no candidate PMs are identified in lines 3–8, i.e., no PM can accommodate all VNFs in $\mathcal{V}_{\texttt{CAN}}$. In this case, candidate PMs for deploying $v_{i,j}$ are searched from $\mathcal{P}_{\texttt{SS}}$.

The remaining code chooses a PM from $\mathcal{P}_{\texttt{CAN}}$ to deploy the VNFs in $\mathcal{V}_{\texttt{CAN}}$. In lines 14–15, if $\mathcal{V}_{\texttt{CAN}}$ contains multiple VNFs while the variable *bundle* is false, $\mathcal{V}_{\texttt{CAN}}$ is reduced to contain only $v_{i,j}$, because deploying multiple VNFs on the same PM is infeasible. Line 16 selects a PM, denoted by p_y, such that it yields the minimum value of $f_{\texttt{TEC}}(p_x, \mathcal{V}_{\texttt{CAN}})$. More specifically, $f_{\texttt{TEC}}(p_x, \mathcal{V}_{\texttt{CAN}})$ is a function that calculates the increase in TEC resulting from deploying all VNFs in $\mathcal{V}_{\texttt{CAN}}$ on PM p_x:

$$f_{\texttt{TEC}}(p_x, \mathcal{V}_{\texttt{CAN}}) = \sum_{\texttt{R} \in \{\texttt{C}, \texttt{M}\}} \sum_{v_{i,m} \in \mathcal{V}_{\texttt{CAN}}} \phi_x^{\texttt{R}} \lambda_{i,m}^{\texttt{R}} + \sum_{l_{\alpha,\beta} \in \mathcal{L}'} \frac{b_{i,j-1}}{b_{\alpha,\beta}^{\texttt{max}}} \left(\psi_{\alpha,\beta}^{\texttt{full}} - \psi_{\alpha,\beta}^{\texttt{idle}}\right), \tag{15}$$

Algorithm 2: Load-balancing deployment

1 $\mathcal{P}_{\texttt{CAN}} \leftarrow \emptyset$;
2 **foreach** $p_x \in \mathcal{P}_{\texttt{SS}}$ **do**
3   **if** $\Phi(p_x, \lambda^{\texttt{R}}_{i,j}, v_{i,j-1}) = \text{true}$ **then**
4         $\mathcal{P}_{\texttt{CAN}} \leftarrow \mathcal{P}_{\texttt{CAN}} \cup \{p_x\}$;
5 **if** $\mathcal{P}_{\texttt{CAN}} = \emptyset$ **then**
6   Deny the deployment of $v_{i,j}$ and return;
7 $e_{\texttt{min}} \leftarrow \min_{p_x \in \mathcal{P}_{\texttt{CAN}}} f_{\texttt{TEC}}(p_x, \{v_{i,j}\})$;
8 **foreach** $p_x \in \mathcal{P}_{\texttt{CAN}}$ **do**
9   **if** $f_{\texttt{TEC}}(p_x, \{v_{i,j}\}) > e_{\texttt{min}} + \tau$ **then**
10         $\mathcal{P}_{\texttt{CAN}} \leftarrow \mathcal{P}_{\texttt{CAN}} \setminus \{p_x\}$;
11 $p_y \leftarrow \arg\max_{p_x \in \mathcal{P}_{\texttt{CAN}}} \left(\sum_{\texttt{R} \in \{\texttt{C},\texttt{M}\}} w_{\texttt{R}} \times \frac{\Lambda^{\texttt{R}}_x - \mu^{\texttt{R}}_x}{\Lambda^{\texttt{R}}_x}\right)$;
12 Deploy $v_{i,j}$ on p_y and update $\mu^{\texttt{R}}_y$, $\forall \texttt{R} \in \{\texttt{C}, \texttt{M}\}$;

where $\mathcal{L}'$ is the set of links traversed by data flow from $v_{i,j-1}$ to $v_{i,j}$. If $v_{i,j}$ is the first VNF in $\mathcal{S}_i$, then $b_{i,j-1} = 0$. Lines 17–18 deploy all VNFs in $\mathcal{V}_{\texttt{CAN}}$ on the selected PM p_y and update its resource usage amount $\mu^{\texttt{R}}_y$ for each resource type.

However, if $\mathcal{P}_{\texttt{CAN}}$ is empty, i.e., no candidate PMs can be found, lines 19–20 are executed to deny the deployment of $v_{i,j}$. Two cases are then considered. If $v_{i,j}$ is a necessary VNF, the entire SFC is rejected, and previously deployed VNFs of $\mathcal{S}_i$ are removed (i.e., rollback). Otherwise, $v_{i,j}$ is skipped, and LEEV proceeds to deploy the next VNF in $\mathcal{S}_i$.

B. Load-Balancing Deployment Strategy

Unlike the bundling-based deployment strategy, which attempts to deploy multiple VNFs of SFC $\mathcal{S}_i$ on the same PM, the load-balancing deployment strategy selects a suitable PM for deploying VNF $v_{i,j}$. Algorithm 2 presents the pseudocode of this strategy. Lines 1–4 check whether each PM $p_x \in \mathcal{P}_{\texttt{SS}}$ can pass the checking function $\Phi(p_x, \lambda^{\texttt{R}}_{i,j}, v_{i,j-1})$. If so, p_x is added to the candidate set $\mathcal{P}_{\texttt{CAN}}$. However, when $\mathcal{P}_{\texttt{CAN}}$ remains empty, no feasible PM is found. Hence, the deployment of $v_{i,j}$ is rejected, as shown in lines 5–6.

Line 7 finds the minimum value of $f_{\texttt{TEC}}(p_x, \{v_{i,j}\})$ among all PMs in $\mathcal{P}_{\texttt{CAN}}$, denoted by $e_{\texttt{min}}$. The for-loop in lines 8–10 removes from $\mathcal{P}_{\texttt{CAN}}$ those PMs whose $f_{\texttt{TEC}}(p_x, \{v_{i,j}\})$ values exceed $e_{\texttt{min}} + \tau$. This procedure ensures that deploying $v_{i,j}$ on any PM in $\mathcal{P}_{\texttt{CAN}}$ results in a TEC increase that is bounded by τ, relative to the minimum achievable TEC increase.

Among all PMs in $\mathcal{P}_{\texttt{CAN}}$, line 11 selects the PM with the largest amount of remaining resources, denoted by p_y. Finally, line 12 deploys $v_{i,j}$ on p_y and then updates its resource usage amount accordingly.

C. Discussion

We discuss the rationale behind the LEEV scheme. When system load $\Psi_{\texttt{sys}}$ is below threshold $\delta_{\texttt{sys}}$, most PMs in $\mathcal{P}$ are considered to have ample available resources. Under this condition, LEEV adopts the bundling-based deployment strategy, which opportunistically attempts to deploy K adjacent VNFs in SFC $\mathcal{S}_i$ on the same PM. This approach not only improves the resource utilization of the selected PM but also reduces LEC incurred by $\mathcal{S}_i$. Note that the value of K should remain small; otherwise, the selected PM may become heavily loaded. Based on this consideration, we recommend setting $K = 2$.

When $\Psi_{\texttt{sys}} \geq \delta_{\texttt{sys}}$, the system operates under a relatively high-load condition. The available resources of PMs become limited, and thus bundling multiple VNFs on the same PM may lead to performance degradation. Therefore, LEEV switches to the load-balancing deployment strategy, which deploys each VNF individually while considering both energy efficiency and load distribution. By filtering candidate PMs based on the TEC increase threshold τ and subsequently selecting the PM with the largest amount of remaining resources, LEEV can prevent excessive load concentration on just a few PMs. This strategy effectively balances the trade-off between minimizing energy consumption and maintaining a balanced resource utilization across PMs under high system load.

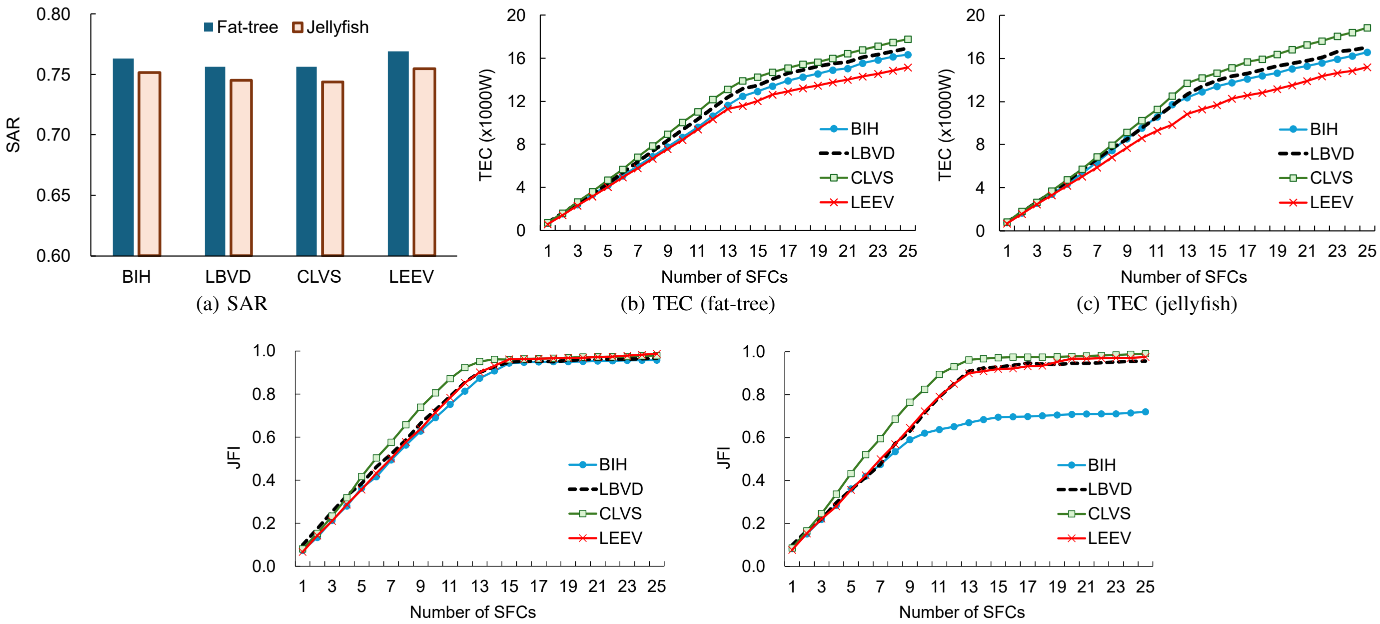
V. Performance Evaluation

To evaluate system performance, a simulator is implemented in C++. Two common large-scale network topologies, namely *fat-tree* and *jellyfish*, are used. The fat-tree topology organizes PMs into a 3-layer hierarchy, providing multiple shortest paths between edge PMs [24]. The jellyfish topology is modeled as a random regular graph, where non-edge PMs have identical degrees but are connected arbitrarily [25]. The network comprises 99 PMs with the maximum node degree of 6, resulting in 162 and 142 links for the fat-tree and jellyfish topologies.

For each PM, the computing and memory capacities ($\Lambda^{\texttt{C}}_x$ and $\Lambda^{\texttt{M}}_x$) are 64 units and 32 units. The baseline energy consumption of a PM ($\phi^{\texttt{base}}_x$) is 5 W, while the additional energy consumption per unit of computing and memory resources ($\phi^{\texttt{C}}_x$ and $\phi^{\texttt{M}}_x$) is set to 4 W and 2 W. For each link, the maximum bandwidth ($b^{\texttt{max}}_{x,y}$) is uniformly drawn from the range [800, 1000] Mbps. Its energy consumption under idle and full-load conditions ($\psi^{\texttt{idle}}_{x,y}$ and $\psi^{\texttt{full}}_{x,y}$) is 0.44 W and 15.4 W [26].

The number of SFCs is increased from 1 to 25, where each SFC consists of 12 VNFs. For the middle six VNFs in each SFC, the computing and memory demands ($\lambda^{\texttt{C}}_{i,j}$ and $\lambda^{\texttt{M}}_{i,j}$) are uniformly drawn from [18, 20] units and [14, 16] units. For other VNFs, the computing and memory demands are drawn from [8, 10] units and [4, 6] units. In addition, the bandwidth required for data transmission between two consecutive VNFs ($b_{i,j}$) is uniformly drawn from the range [1, 5] Mbps.

We compare the proposed LEEV scheme with three representative methods introduced in Section II. The *blocking island hierarchy (BIH)* method [16] uses blocking islands to abstract PMs and links for VNF placement. The *load-balanced VNF deployment (LBVD)* method [19] scores PMs according to local and neighboring resource status and performs VNF migration upon overload. The *cost-aware and load-balancing VNF scheduling (CLVS)* method [21] employs a utility-based approach with differentiated placement strategies for the first and subsequent VNFs in an SFC.



(a) SAR (b) TEC (fat-tree) (c) TEC (jellyfish)

(d) JFI (fat-tree) (e) JFI (jellyfish)

Fig. 1. Performance evaluation of BIH, LBVD, CLVS, and LEEV under different network topologies.

A. Performance Comparison

Fig. 1(a) compares SAR after deploying VNFs of 25 SFCs. As the fat-tree topology contains more links than the jellyfish topology, and all methods restrict the search space of candidate PMs to those near the PM hosting the previous VNF, better performance is observed under the fat-tree topology. Moreover, all VNFs in an SFC are mandatory, i.e., if any VNF cannot be successfully deployed, all previously deployed VNFs in the same SFC are removed. Thus, the SAR values of all methods fall within the range of 0.74 to 0.77. As observed, the LEEV scheme consistently achieves a higher SAR than others.

Fig. 1(b) and Fig. 1(c) present the TEC achieved by each method. Both LBVD and CLVS distribute VNFs across PMs to improve load balancing, which significantly increases inter-PM traffic and consequently results in higher TEC compared to the other methods. In contrast, LEEV adopts both bundling-based and load-balancing deployment strategies that explicitly consider the impact on TEC when placing VNFs. As a result, LEEV achieves the lowest TEC among all methods. In the fat-tree topology, LEEV reduces TEC by 6.12%, 10.41%, and 14.73% compared to BIH, LBVD, and CLVS, respectively. In the jellyfish topology, LEEV reduces TEC by 9.93%, 12.63%, and 18.52% compared to BIH, LBVD, and CLVS, respectively.

Fig. 1(d) and Fig. 1(e) compare the JFI achieved by each method. JFI generally increases as more SFCs are added, since more PMs are assigned VNFs and their loads become more balanced. In the fat-tree topology, all methods achieve similar JFI values due to the regular structure and high link connectivity. In contrast, BIH achieves significantly lower JFI than the other methods under the jellyfish topology, indicating that the blocking island abstraction provides limited information for balancing loads across PMs. The LEEV scheme performs comparably to LBVD and merely slightly worse than CLVS, demonstrating its effectiveness in terms of load balancing.

B. Effect of Threshold δ_{sys}

The LEEV scheme determines the used deployment strategy according to whether the system load is below threshold δ_{sys}. Therefore, we evaluate the impact of δ_{sys} on the performance of LEEV, as shown in Fig. 2.

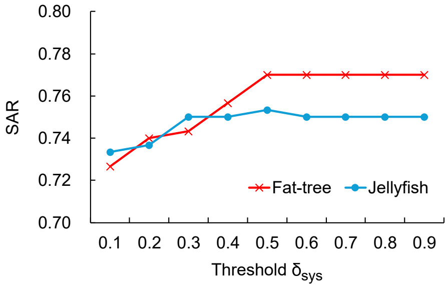
We first investigate the case where $\delta_{\mathrm{sys}} \leq 0.5$. As δ_{sys} rises, the likelihood of using the bundling-based deployment strategy under light-load conditions also increases. This leads to higher PM resource utilization (reflected by an increased SAR), lower TEC (due to reduced inter-PM traffic), and a decrease in JFI (as some PMs tend to host more VNFs).

When $\delta_{\mathrm{sys}} > 0.5$, the system load becomes relatively high, meaning that most PMs have limited remaining resources. Under such conditions, the bundling-based deployment strategy cannot place multiple VNFs on the same PM in most cases and thus behaves similarly to the load-balancing deployment strategy. Hence, SAR, TEC, and JFI remain nearly unchanged as δ_{sys} continues to increase beyond 0.5.

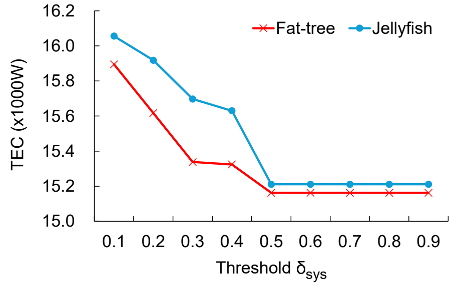
Based on the results in Fig. 2, we recommend setting $\delta_{\mathrm{sys}} = 0.5$, as this value achieves a favorable trade-off by improving SAR, reducing TEC, and maintaining a high JFI.

VI. Conclusion

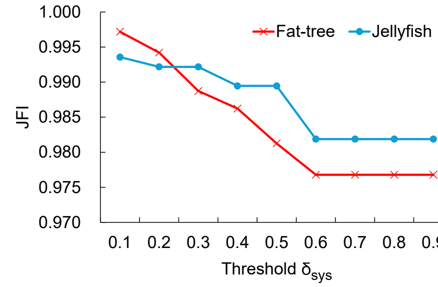
In this paper, we propose the LEEV scheme to efficiently address the VNF deployment problem by jointly considering resource utilization, energy consumption, and load balancing. LEEV dynamically adapts deployment decisions according to the system load. Under light-load conditions, LEEV employs



(a) SAR



(b) TEC



(c) JFI

Fig. 2. Effect of threshold $\delta_{\mathtt{sys}}$ on the performance of LEEV.

the bundling-based deployment strategy to flexibly place adjacent VNFs of an SFC on the same PM. As the system load exceeds threshold $\delta_{\mathtt{sys}}$, LEEV switches to the load-balancing deployment strategy to distribute VNFs across PMs. Simulation results demonstrate that LEEV consistently outperforms BIH, LBVD, and CLVS in terms of SAR and TEC while maintaining effective load balancing among PMs.

Acknowledgment

This work was supported by National Science and Technology Council, Taiwan under Grant 113-2221-E-110-056-MY3.

References

[1] B. Yi, X. Wang, K. Li, S. K. Das, and M. Huang, "A comprehensive survey of network function virtualization," *Computer Networks*, vol. 133, pp. 212–262, 2018.

[2] ETSI. ISG NFV. [Online]. Available: https://www.etsi.org/technical-groups/nfv/

[3] W. Rafique, J. R. Barai, A. O. Fapojuwo, and D. Krishnamurthy, "A survey on beyond 5G network slicing for smart cities applications," *IEEE Communications Surveys & Tutorials*, vol. 27, no. 1, pp. 595–628, 2025.

[4] W. Attaoui, E. Sabir, H. Elbiaze, and M. Guizani, "VNF and CNF placement in 5G: Recent advances and future trends," *IEEE Transactions on Network and Service Management*, vol. 20, no. 4, pp. 4698–4733, 2023.

[5] G. Liu, Y. Liu, Q. Meng, B. Wang, K. Chen, and Z. Shen, "Traffic load balancing in data center networks: A comprehensive survey," *Computer Science Review*, vol. 57, pp. 1–30, 2025.

[6] O. Soualah, M. Mechtri, C. Ghribi, and D. Zeghlache, "Online and batch algorithms for VNFs placement and chaining," *Computer Networks*, vol. 158, pp. 98–113, 2019.

[7] C. Pham, N. H. Tran, S. Ren, W. Saad, and C. S. Hong, "Traffic-aware and energy-efficient VNF placement for service chaining: Joint sampling and matching approach," *IEEE Transactions on Services Computing*, vol. 13, no. 1, pp. 172–185, 2020.

[8] D. H. P. Nguyen, Y. H. Lien, B. H. Liu, S. I. Chu, and T. N. Nguyen, "Virtual network function placement for serving weighted services in NFV-enabled networks," *IEEE Systems Journal*, vol. 17, no. 4, pp. 5648–5659, 2023.

[9] L. Liu, S. Guo, G. Liu, and Y. Yang, "Joint dynamical VNF placement and SFC routing in NFV-enabled SDNs," *IEEE Transactions on Network and Service Management*, vol. 18, no. 4, pp. 4263–4276, 2021.

[10] R. Mohamed, M. Avgeris, A. Leivadeas, I. Lambadaris, J. Chinneck, T. Morris, and P. Djukic, "Service function chain network planning through offline, online and infeasibility restoration techniques," *Computer Networks*, vol. 242, pp. 1–17, 2024.

[11] D. Niu, G. Yi, and Z. Tang, "Toward a resource-efficient service function chain mapping mechanism: A heuristic method," in *IEEE Vehicular Technology Conference*, 2024, pp. 1–5.

[12] S. Qi, S. Li, S. Lin, M. Y. Saidi, and K. Chen, "Energy-efficient VNF deployment for graph-structured SFC based on graph neural network and constrained deep reinforcement learning," in *Asia-Pacific Network Operations and Management Symposium*, 2021, pp. 348–353.

[13] R. Lin, L. He, S. Luo, and M. Zukerman, "Energy-aware service function chaining embedding in NFV networks," *IEEE Transactions on Services Computing*, vol. 16, no. 2, pp. 1158–1171, 2023.

[14] A. Varasteh, B. Madiwalar, A. V. Bemten, W. Kellerer, and C. Mas-Machuca, "Holu: Power-aware and delay-constrained VNF placement and chaining," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1524–1539, 2021.

[15] X. Zhang, Z. Xu, L. Fan, S. Yu, and Y. Qu, "Near-optimal energy-efficient algorithm for virtual network function placement," *IEEE Transactions on Cloud Computing*, vol. 10, no. 1, pp. 553–567, 2022.

[16] A. Varasteh, M. D. Andrade, C. M. Machuca, L. Wosinska, and W. Kellerer, "Power-aware virtual network function placement and routing using an abstraction technique," in *IEEE Global Communications Conference*, 2018, pp. 1–7.

[17] C. You and L. M. Li, "Efficient load balancing for the VNF deployment with placement constraints," in *IEEE International Conference on Communications*, 2019, pp. 1–6.

[18] J. Fu and G. Li, "An efficient VNF deployment scheme for cloud networks," in *IEEE International Conference on Communication Software and Networks*, 2019, pp. 497–502.

[19] Y. C. Wang and S. H. Wu, "Efficient deployment of virtual network functions to achieve load balance in cloud networks," in *Asia-Pacific Network Operations and Management Symposium*, 2022, pp. 1–6.

[20] H. Xuan, L. Lin, Z. Liu, and X. Zhao, "Load balance and time delay efficient algorithm for VNF service chain deployment based on DQN," in *International Conference on Computational Intelligence and Security*, 2022, pp. 440–444.

[21] Y. C. Wang and C. Y. Tsai, "Deployment of virtual network functions for load balancing and cost efficiency in NFV systems," in *IEEE International Conference on Communication, Networks and Satellite*, 2025, pp. 195–202.

[22] A. Laghrissi and T. Taleb, "A survey on the placement of virtual resources and virtual network functions," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1409–1434, 2019.

[23] Y. C. Wang and D. R. Jhong, "Efficient allocation of LTE downlink spectral resource to improve fairness and throughput," *International Journal of Communication Systems*, vol. 30, no. 14, pp. 1–13, 2017.

[24] Y. C. Wang and S. Y. You, "An efficient route management framework for load balance and overhead reduction in SDN-based data center networks," *IEEE Transactions on Network and Service Management*, vol. 15, no. 4, pp. 1422–1434, 2018.

[25] Z. Alzaid, S. Bhowmik, and X. Yuan, "Multi-path routing in the jellyfish network," in *IEEE International Parallel and Distributed Processing Symposium Workshops*, 2021, pp. 832–841.

[26] IEEE Computer Society, *IEEE Std 802.3af-2003: Data Terminal Equipment (DTE) Power via Media Dependent Interface (MDI)*, Institute of Electrical and Electronics Engineers Std., June 2003.