Computer Networks 93 (2015) 460–479

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

# Energy-aware node and link reconfiguration for virtualized network environments

Ebrahim Ghazisaeedi*, Changcheng Huang

*Department of Systems and Computer Engineering, Carleton University, Ottawa, Ontario, Canada*

ARTICLE INFO

*Article history:*
Received 5 March 2015
Revised 1 August 2015
Accepted 19 September 2015
Available online 19 October 2015

*Keywords:*
Virtualized network environment
Virtualized data center
Energy efficiency
Network reconfiguration
Network optimization

ABSTRACT

Energy consumption in information and communications technology (ICT) accounts for a large portion of the total energy consumed in industrial countries, which is rapidly increasing. Virtualized network environments (VNEs) have recently emerged as a solution in this technology to address the challenges of future Internet. VNEs also play a fundamental role toward virtualizing data centers. Consequently, it is essential to develop novel techniques for reducing the energy consumption of VNEs. In this paper, we discuss multiple energy-saving solutions that locally optimize the node and link energy consumption of VNEs during the off-peak period by reconfiguring the mapping of already allocated virtual nodes and links. The proposed reconfigurations enable the providers to adjust the level of the reconfiguration and accordingly control possible traffic disruptions. An integer linear program (ILP) is formulated for each solution. Because the defined ILPs are $\mathcal{NP}$-hard, a novel heuristic algorithm is also proposed. The proposed energy-saving methods are evaluated over random VNE scenarios. The results confirm that the solutions save notable amounts of energy in physical nodes and links during the off-peak period.



## 1. Introduction

Several reports from different information and communications technology (ICT) organizations from around the world have confirmed the increasing energy demands in this technology, which is a major concern for future Internet. In the case where no green technology will be deployed in communication networks, the Global e-Sustainability Initiative (GeSI) predicts an energy consumption of 35.8 TWh for European telecom operators in 2020 [1].

Recently, virtualization has emerged for communication networks in ICT. It shares the resources in the network environment [2]. A virtualized network environment (VNE) supports the coexistence of multiple virtual networks (VNs) over a single substrate network [3]. Virtual nodes and links are mapped onto physical nodes and links, respectively. Network virtualization decouples the functionality of the current network architecture into infrastructure providers (InPs) and service providers (SPs). In addition, virtual machines (VMs) traditionally virtualize physical servers' resources. VNs together with VMs underpin virtualized data centers (VDCs). Traditional data centers are moving toward virtualized data centers to address cloud computing limitations regarding network performance, security, and manageability [4]. Hence, network virtualization has been regarded as a promising technology to flexibly share the resources, and therefore, the corresponding solutions to energy savings in this type of network become essential.

In fact, virtual networks' traffic loads change over time. Virtual networks might be highly utilized during a period of time (peak time, e.g., day hours), whereas they are under-utilized during another notable period of time (off-peak time, e.g., night hours). Virtual networks' processing demands are likely related to their traffic loads. Traffic

* Corresponding author. Tel.: +1 6138188373.
*E-mail addresses:* eghazisaeedi@sce.carleton.ca (E. Ghazisaeedi), huang@sce.carleton.ca (C. Huang).

http://dx.doi.org/10.1016/j.comnet.2015.09.041
1389-1286/

variations in virtual networks correspondingly change the utilization of the substrate network. The reports for 40 North American and 25 European network providers reveal a 60% difference between peak and minimum off-peak traffic rates over their substrate networks [5]. However, today's substrate networks are provisioned to support VNs' peak time traffic demands, with some additional over-provisioning accommodating unexpected traffic rates [5]. The substrate network's elements are always switched on, neglecting the traffic behavior.

Cloud providers could determine the off-peak time period of the substrate network and traffic/processing demands of each VN in that period through information given by VNs' customers or using network traffic prediction techniques, e.g., [6,7], that estimate the future traffic by examining the current traffic state. During the off-peak period, it is possible to reduce the VNE's energy consumption by reconfiguring the mapping of the already embedded VNs according to their decreased demands. In this context, virtual networks are accepted and embedded onto the substrate network by a normal (not energy efficient) VNE embedding process to accommodate the peak traffic behavior. The cloud provider runs the reconfiguration technique during normal network operations on networks transitioning from the peak period to the off-peak period to save energy in the off-peak period. This is typically called elasticity in cloud terminology, and it can be performed in three ways: proactive cyclic scaling, proactive event-based scaling, and auto-scaling based on demand. However, when the traffic load changes from a peak level to an off-peak level, some traffic flows that last in both time periods might suffer from traffic disruptions imposed by applying the reconfiguration [5]. Moreover, reconfiguring the mapping of embedded VNs may require additional signaling traffic that is necessary for notifying all the involved routers [8]. This may introduce a significant work load for the signaling controller, particularly when the reconfiguration attempts to make changes to a large number of nodes at the same time. Consequently, it may not be a good practice to reconfigure the mapping of every embedded virtual node/link.

The reconfiguration process requires four steps. First, it learns the peak and off-peak periods and the associated demands. Second, it calculates the off-peak mapping. Third, at the beginning of each off-peak period, it reconfigures the mapping of all the affected VNs based on the result from the second step. Fourth, at the beginning of each peak period, it restores the original mapping.

The major cost of the reconfiguration consists of three components: collecting traffic information with each VN, calculating the off-peak mapping, and conducting the reconfiguration operation. Because the third item is supposed to be completed as fast as possible, its energy cost is insignificant. The first and second items depend on how often they are performed. For example, if all the traffic of all VNs are roughly periodic, the first and second items only need to be performed once whenever a new VN is added or a VN is leaving.

In this paper, we discuss multiple novel energy-saving solutions that optimize a VNE's node and link energy consumption during the off-peak time according to two defined power models. We approach the problem by reconfiguring the mapping for some already embedded virtual nodes and links. We assume that the substrate and virtual networks are homogeneous. All the substrate and virtual nodes are assumed to be switches/routers to reflect the network environment. This is the case in the majority of existing related research studies.

In this regard, we first formulated a mixed integer linear program (MILP) for off-peak node and link energy optimization through local node and link reconfiguration. A stress rate is defined for substrate nodes. Accordingly, the MILP *may* set less stressed substrate nodes and their adjacent substrate links into sleep mode for the off-peak time. We re-map a virtual link *if and only if* we sleep the substrate node that hosts its source/sink virtual node or at least one substrate node that relays its traffic over its embedded path. The energy consumption of a re-mapped virtual link is also minimized.

Because reconfiguring the mapping for both virtual nodes and virtual links may be expensive and cause interruptions to normal network operations, we derive another methodology that optimizes the node and link energy consumption of a VNE during the off-peak time by reconfiguring the mapping for only some of the virtual links. We do not reconfigure the mapping of the embedded virtual nodes in this case. We call the latter problem off-peak node and link energy optimization by local link reconfiguration. We define a different stress rate for intermediate substrate nodes, which do not host a virtual node and only relay the traffic. Accordingly, a solution is proposed to optimize the total energy consumption of the intermediate substrate nodes and substrate links during the off-peak time. This method *might* set the less stressed intermediate substrate nodes and their respective substrate links into sleep mode for the off-peak time. We re-map a virtual link *if and only if* we sleep at least one intermediate substrate node over its embedded path. The energy consumption of a re-mapped virtual link is also minimized. A MILP for splittable traffic and a binary integer linear program (BILP) for non-splittable traffic are formulated for this strategy.

These methods enable providers to change level of the reconfiguration by adjusting the stress rate's threshold. Therefore, they can control the possible traffic interruptions of the reconfiguration. Clearly, there is a trade-off between the energy-savings level and the possible traffic interruptions.

Because the formulated optimization programs are $\mathcal{NP}$-hard, we also propose a heuristic algorithm for off-peak node and link energy optimization through local link reconfiguration. The simulation results confirm that the heuristic algorithm can achieve points that are close to the optimum points set by the formulated optimization program, while it is considerably faster. The proposed heuristic is scalable to large network sizes.

The remainder of this paper is organized as follows. Most of the related works in the literature and our contributions in this paper are discussed in Section 2. Two power models for physical nodes and links are studied in Section 3. Multiple optimization programs are defined and formulated as ILPs in Section 4, and the proposed heuristic is discussed in Section 5. To evaluate the performances of the ILPs and the heuristic algorithm, multiple random scenarios have been simulated over randomly generated VNEs, and the results are analyzed in Section 6. The paper will conclude in Section 7.

## 2. Related works

The literature is rich in terms of network virtualization and energy-saving solutions for communication networks. However, these topics have been studied separately. There are few very recent works regarding energy consumption in VNEs. We review these works in this section.

Four papers [9–12] attempted to save energy in a VNE by making its embedding procedure energy-aware. This was performed by modifying the link weights based on the physical links' power consumption in [9] and consolidating VNs to the smallest number of substrate network elements in [10–12]. Modifying the VNE embedding algorithms to achieve an energy-efficient VNE suffers from a major difficulty. When VNE embedding algorithms are modified to map the resources energy-wise, several extra constraints will be added to the embedding procedure. Accordingly, the embedding algorithm has a smaller set of physical node and link candidates to choose from. This decreases the network's admittance ratio for new virtual network requests, which is not cost efficient for the providers. The main economic objective of providers is to reject the minimum number of virtual network requests. Thus, these solutions are not profitable for them in the long term.

Some other papers proposed heuristics that modify the already mapped VNs. Botero and Hesselbach [13] proposed a heuristic algorithm that reconfigures the mapping of accepted VNs at each embedding phase to save energy. This approach has the same problem of the energy-efficient embedding methods because reconfiguring the mapping of accepted VNs at each embedding phase for their life time might still cause capacity bottlenecks that decrease the network admittance rate for new VNs. Moreover, their heuristic assumes that each virtual link is only mapped onto a single physical link. However, the virtual links might be mapped onto physical paths. Finally, their reconfiguration problem is not formulated mathematically. An off-line heuristic reconfiguration algorithm was proposed in our previous work [14]. This algorithm attempts to maximize the number of sleep mode physical links during the off-peak period of VNEs. It reroutes the off-peak traffic of already embedded virtual links to other already allocated traffic capacities. It does not change the mapping of VNs. Assuming that fixed VN mapping prevents us from rerouting a VN's off-peak traffic to substrate links in which no traffic capacity is allocated in them to that particular VN, this decreases the level of energy that we could save. et al. [4] proposed a method for moving embedded virtual machines (VMs) on servers to other servers. Their solution is run over time periodically to consolidate the VMs. Nevertheless, moving allocated VMs and setting the servers into sleep mode is expensive, if not impossible, for two reasons. First, a large amount of data is generally distributed over a large number of servers, and it is not profitable/possible for the providers to move the data on one server to another one. Second, waking up servers from sleep mode (in the case of unexpected demand or returning to peak time) imposes hundreds of milliseconds of delay to the tasks, which might violate service level objectives (SLOs) [15]. Moreover, their solution does not enable the providers to adjust level of the reconfiguration and control the possible traffic disruptions.

*Our contributions*: (a) We discuss multiple novel optimization programs that formulate unique off-peak energy-saving reconfiguration strategies for nodes and links in VNEs. To the best of our knowledge, these problems have not been defined or formulated mathematically in any published study. (b) In contrast to related research studies, the proposed methods enable providers to control the level of reconfiguration and possible traffic interruptions. (c) Different from previous research studies [9–13], our method does not decrease the network admittance ratio for new virtual networks. This is because we reconfigure the mapping of the already accepted VNs only for the off-peak period, and they could be reconfigured back to their peak mapping in the case of unexpected new demands. (d) Our solution is not limited to a sub-topology as the case in [14], and thus, it has a larger degree of freedom to save energy. (e) We do not move VMs, so our method does not have the difficulties of [4]. (f) We define our solutions according to two power models. (g) Different programs are formulated for splittable and non-splittable traffic to study the impact of traffic splittability. (h) We present a scalable heuristic algorithm that can achieve results close to the optimum results but that is considerably faster than the optimization program. (i) We assess the impact of different parameters on the energy-saving capabilities of the discussed solutions through extensive simulations.

## 3. Power models

Power consumption is the rate at which the system performs work, and it is measured in watts, with units of joules/second. Energy consumption is the amount of work that has been performed by the system in a period of time, and it is measured in watt-hours. In this regard, power and energy are closely connected.

In a VNE, node and link energy consumption arise from physical nodes and links. To identify the factors that affect the energy consumption of physical nodes and links, it is necessary to study their power models. We review two power models for them.

We assume that substrate networks are homogeneous. All the substrate nodes are assumed to be switches/routers to reflect the network environment. Note that the proposed solutions are general, and they could also employ physical server power models. The power models for a server substrate node can be found in [16].

### 3.1. Fixed power model

The first power model defines a constant amount of power consumption for an active physical node or link, regardless of its traffic load. According to this power model, the entire power consumed by an active physical node or link is for keeping the device active, and the traffic load does not change its power consumption. We call this model the *fixed* power model.

Conforming to the *fixed* power model, the $i$th substrate node $v_s^i$ drains $\tilde{p}^m(v_s^i)$ if it is active, regardless of its traffic load. $\tilde{p}^m(v_s^i)$ is the maximum power consumption of $v_s^i$. Eq. (1) defines the actual power consumption $\tilde{p}(v_s^i)$ for a

physical node $v_s^i$ based on the *fixed* power model. $\alpha(v_s^i)$ indicates the status of $v_s^i$. If $v_s^i$ is active, then $\alpha(v_s^i)$ is 1. Otherwise, $\alpha(v_s^i)$ is 0.

$$\tilde{p}(v_s^i) = \alpha(v_s^i)\tilde{p}^m(v_s^i) \quad (1)$$

Similarly, according to the *fixed* power model, an active physical link consumes a constant amount of power. Eq. (2) defines the actual power consumption $\tilde{p}(l_s^{i,j})$ of a substrate link $l_s^{i,j}$ based on the *fixed* power model. $l_s^{i,j}$ represents the substrate link that connects the $i$th substrate node to the $j$th substrate node. If $l_s^{i,j}$ is active ($\alpha(l_s^{i,j}) = 1$), then the link consumes a fixed amount of $\tilde{p}^m(l_s^{i,j})$, which is the maximum power consumption of $l_s^{i,j}$.

$$\tilde{p}(l_s^{i,j}) = \alpha(l_s^{i,j})\tilde{p}^m(l_s^{i,j}) \quad (2)$$

### 3.2. Semi-proportional power model

The second power model considers a base amount of power for an active physical node or link, which is for keeping the device operational. However, in contrast to the first power model, the traffic load in the node or link changes its power consumption. We call this model the *semi-proportional* power model.

Eq. (3) defines the actual power consumption $\tilde{p}(v_s^i)$ of a substrate node $v_s^i$ that is a switch/router according to the *semi-proportional* power model [17,18]. If the node is active ($\alpha(v_s^i) = 1$), then it consumes the base amount of power $\tilde{p}^b(v_s^i)$. The traffic load $r(v_s^i)$ in the node increases its power consumption linearly between $\tilde{p}^b(v_s^i)$ and $\tilde{p}^m(v_s^i)$. $C_b(v_s^i)$ is the switching capacity of $v_s^i$. Note that there are also other power models for switches/routers that work based on the device configuration, e.g., [19]. Our solutions could be adapted to employ these models.

$$\tilde{p}(v_s^i) = \alpha(v_s^i)\tilde{p}^b(v_s^i) + \frac{r(v_s^i)}{C_b(v_s^i)}\left(\tilde{p}^m(v_s^i) - \tilde{p}^b(v_s^i)\right) \quad (3)$$

Similar to the node power model, Eq. (4) defines the actual power consumption $\tilde{p}(l_s^{i,j})$ of a physical link $l_s^{i,j}$ based on the *semi-proportional* power model. Hence, if the physical link $l_s^{i,j}$ is active, it consumes base power $\tilde{p}^b(l_s^{i,j})$, which is for keeping the link operational. However, the traffic load $r(l_s^{i,j})$ in the link also increases its actual power consumption linearly. $C_b(l_s^{i,j})$ is the bandwidth capacity of $l_s^{i,j}$.

$$\tilde{p}(l_s^{i,j}) = \alpha(l_s^{i,j})\tilde{p}^b(l_s^{i,j}) + \frac{r(l_s^{i,j})}{C_b(l_s^{i,j})}\left(\tilde{p}^m(l_s^{i,j}) - \tilde{p}^b(l_s^{i,j})\right) \quad (4)$$

Because switches/routers are considered to be the physical nodes, $\tilde{p}^b(v_s^i)$ and $\tilde{p}^m(v_s^i)$ could be determined using Eqs. (5) and 6, respectively [17,18]. $\tilde{p}^b(v_s^i)$ and $\tilde{p}^m(v_s^i)$ for a server physical node can be determined through calibration experiments, as described in [20].

$$\tilde{p}^b(v_s^i) = 0.85C_b(v_s^i)^{\frac{2}{3}} \quad (5)$$

$$\tilde{p}^m(v_s^i) = C_b(v_s^i)^{\frac{2}{3}} \quad (6)$$

Consequently, $\tilde{p}(v_s^i)$ for the case of the *semi-proportional* power model can be re-written as Eq. (7).

$$\tilde{p}(v_s^i) = 0.85\alpha(v_s^i)C_b(v_s^i)^{\frac{2}{3}} + \frac{0.15r(v_s^i)}{C_b(v_s^i)^{\frac{1}{3}}} \quad (7)$$

In addition, $\tilde{p}^b(l_s^{i,j})$ and $\tilde{p}^m(l_s^{i,j})$ for a physical link are dependent on the physical link bandwidth capacity $C_b(l_s^{i,j})$, its length, and the cable type. For example, [18] provides numerical values for the base and maximum physical link power consumption for some bandwidth capacity ranges.

Today's networks are designed based on the *fixed* power model; thus, it is a common model that is widely used [18]. Nevertheless, it is not efficient for an active physical node or link to consume a constant amount of power regardless of its traffic load. Therefore, physical nodes and links are expected to be modified such that their power consumption will be more adaptive to their traffic load. The *semi-proportional* power model provides a traffic-adaptive power model that formulates the power consumption of a physical node or link according to its traffic load.

The power model determines the energy-saving methodology. In this paper, we assume that all the substrate links are in the same bandwidth capacity range, and thus, they consume the same amount of power. First, considering the *fixed* power model, a physical node or link consumes a constant amount of power, regardless of the traffic load. Therefore, the best energy-saving strategy for minimizing a physical node or link's energy consumption is to set the device into sleep mode. However, because physical nodes consume a considerably larger amount of power compared to physical links, it is more important to set a physical node into sleep mode rather than a physical link. Second, in the case of the *semi-proportional* power model, the traffic load in the device changes its power consumption. Although a large amount of power is still consumed to keep the device operational, it might be possible to save some energy by rerouting the device's traffic to other active nodes and links, with the larger value for the division of the difference between the maximum and base power consumption by the bandwidth capacity. Therefore, in the case of the *semi-proportional* power model, it may be feasible to save energy by sleeping the network element or rerouting traffic. Consequently, the objective for an energy-saving solution based on the *fixed* power model is different from the objective for an energy-saving method based on the *semi-proportional* power model. In this regard, we formulate multiple node and link energy-saving solutions according to both power models.

## 4. Integer linear programs

During the off-peak time of a VNE, the virtual networks' demands in terms of traffic, processing, and so forth are decreased. Accordingly, it might be possible to save notable amounts of energy in substrate nodes and links by reconfiguring the mapping of the already allocated resources while still satisfying the off-peak demands. In this regard, we define a general reconfiguration problem that targets the mapping of both physical nodes and links to minimize the VNE's energy consumption during the off-peak period. The problem description is as follows:

Given:

- The homogeneous substrate network topology.
- Allocated virtual networks' topologies.
- For each substrate node: The physical bandwidth capacity, the processing capacity, and the allocated processing capacity to each virtual node in the substrate node.
- For each substrate link: The physical bandwidth capacity and the allocated bandwidth capacity to every virtual link in the substrate link.
- Off-peak traffic and processing demands of each virtual network, provided by virtual network customers or network traffic prediction techniques.

Find:

- Modified off-peak VNE mapping that leads to the minimum total node and link energy consumption during the off-peak time.

Constraints:

- Supporting off-peak traffic and processing demands.

We approach the defined energy-saving problem with two different local reconfiguration strategies. First, we reconfigure the mapping for some of the already allocated virtual nodes and links during the off-peak period to minimize the total node and link energy consumption of the network in that period. It is possible to minimize the VNE's total energy consumption by setting any capable physical node or link into sleep mode or by rerouting its traffic load. Rerouting traffic may also help save energy. However, we still need to accommodate the demanded off-peak processing and traffic rates. We call this problem off-peak node and link energy optimization by local node and link reconfiguration.

Reconfiguring the mapping for both virtual nodes and links might be expensive or cause interruptions to normal network operations. Moreover, in some cases, it is infeasible to reconfigure the mapping of virtual nodes. For example, moving an embedded access/edge virtual node to the other part of the substrate network might be impossible. An access/edge virtual node connects the VN to other networks (on different substrate networks). Hence, we derive the second reconfiguration method that optimizes the total node and link energy consumption in the VNE during the off-peak time by reconfiguring the mapping for only some of the already allocated virtual links. The second method does not reconfigure the already allocated virtual nodes, and therefore, it is not possible to sleep physical nodes in which one or multiple virtual nodes are allocated to them. However, it is possible to save energy in the capable intermediate substrate nodes or any qualified substrate link. Rerouting traffic loads over the network may also help to save energy based on some power models. We call the second problem off-peak node and link energy optimization by local link reconfiguration.

Both of the reconfiguration strategies are local optimization problems because they do not reconfigure the mapping for all of the virtual nodes or links. This helps to reduce possible interruptions to the network operations. Although the second reconfiguration strategy does not reconfigure virtual nodes and is therefore expected to save smaller amounts of energy compared to the first method, it causes fewer interruptions to the normal network operations.

Note that virtual network customers might have multiple constraints for their demanded virtual nodes and links. In this paper, we only consider the requested switching and processing capacity for switches/routers and the traffic demands.

In this regard, we first model VNE mathematically. Then, for both the *fixed* and *semi-proportional* power models, we formulate integer linear programs for off-peak node and link energy optimization by local node and link reconfiguration and off-peak node and link energy optimization by local link reconfiguration. Because the traffic type (splittable/non-splittable) has a major impact on the solution methodology, we formulate off-peak node and link energy optimization by local link reconfiguration for both splittable and non-splittable traffic.

### 4.1. Network model

The substrate network is modeled as a directed graph $G_s = (V_s, E_s)$, where $V_s$ is the set of substrate vertices and $E_s$ is the set of substrate edges. Vertices represent nodes and edges denote links in the network environment. Because the graph is directed, we have a higher level of flexibility in terms of rerouting traffic flows. $C_c(v_s^i)$ expresses the processing capacity of $v_s^i$.

Similar to the substrate network model, the $n$th virtual network, from the set of all the virtual networks $\Phi$, is also modeled as a directed graph $G_n = (V_n, E_n)$. $V_n$ and $E_n$ represent the $n$th virtual network's vertices and edges, respectively. $N_n = |V_n|$ and $L_n = |E_n|$ denote the total number of virtual nodes and the total number of virtual links in the $n$th virtual network, respectively. In addition, $v_n^k$ refers to the $k$th virtual node in the $n$th virtual network. $\acute{C}_c(v_n^k)$ represents the off-peak processing demand of $v_n^k$.

In the VNE embedding procedure, the requested virtual network $G_n$ is mapped onto the defined substrate network $G_s$: $G_n \rightarrow G_s$. Virtual nodes are allocated in the chosen physical nodes. $\phi(v_n^k, v_s^i)$ indicates whether the virtual node $v_n^k$ is allocated in the physical node $v_s^i$. $\phi(v_n^k, v_s^i)$ is 1 if $v_n^k$ is allocated in $v_s^i$. Otherwise, it is 0. A virtual link could be mapped onto a single physical link or onto multiple physical links that create a physical path. If traffic is splittable, a virtual link could be mapped onto multiple substrate paths. However, if traffic is non-splittable, each traffic demand is carried by only one path. The allocated virtual links of the $n$th VN are presented as the set of ordered allocated virtual node pairs ($a_m$, $b_m$), $m = 1, 2, \ldots, L_n$. $l_n^{a_m, b_m}$ represents $m$th virtual link, belonging to the $n$th VN, that connects the virtual node mapped onto the $a_m$th substrate node to the virtual node mapped onto the $b_m$th substrate node. $\acute{r}_n^m$ represents the off-peak traffic demand of $l_n^{a_m, b_m}$. In addition, $\acute{r}_n^{i,j}(m)$ denotes the off-peak traffic demand of the allocated traffic capacity to $l_n^{a_m, b_m}$ in $l_s^{i,j}$. During the off-peak period, the reserved processing and traffic capacities for virtual nodes and links in physical nodes and links, respectively, are equal to their particular off-peak demands, and the remainder the physical capacity could be shared.

### 4.2. Programs based on the fixed power model

In this section, we develop energy-saving solutions based on the *fixed* power model. It is assumed that all the substrate links are in the same bandwidth capacity range. Conforming to the *fixed* power model, the energy-saving strategy to reduce a physical node or link's energy consumption is to set the device into sleep mode. The following energy-saving methods are formulated according to this logic.

#### 4.2.1. Off-peak node and link energy optimization by local node and link reconfiguration (ONL-LNLs-F)

We define a stress rate for a substrate node. Accordingly, a MILP is formulated for the problem. The MILP *may* set less stressed substrate nodes and their adjacent substrate links into sleep mode for the off-peak time. We re-map a virtual link *if and only if* we sleep the substrate node that hosts its source/sink virtual node or at least one substrate node that relays its traffic over its embedded path. The program also sets the maximum number of physical links into sleep mode during the re-allocation process of virtual links. This problem is defined according to the *fixed* power model. The traffic is assumed to be splittable in this problem to deliver the highest level of energy savings. We call this problem ONL-LNLs-F.

In this problem, the stress rate $\tilde{s_1}(v_s^i)$ of a physical node $v_s^i$ reveals the intensity of the total off-peak processing and traffic demands in the node. Eq. (8) defines $\tilde{s_1}(v_s^i)$.

$$\tilde{s_1}\left(v_s^i\right) = \frac{\sum_{\{n|G_n\in\Phi\}}\sum_{k\in V_n}\phi\left(v_n^k, v_s^i\right)\acute{C}_c\left(v_n^k\right)}{C_c\left(v_s^i\right)} \times \left(\frac{\sum_{\{j|(i,j)\in E_s\}}\sum_{\{n|G_n\in\Phi\}}\sum_{m=1}^{L_n}\acute{r}_n^{i,j}(m)}{C_b\left(v_s^i\right)} + \frac{\sum_{\{j|(j,i)\in E_s\}}\sum_{\{n|G_n\in\Phi\}}\sum_{m=1}^{L_n}\acute{r}_n^{j,i}(m)}{C_b\left(v_s^i\right)}\right) \tag{8}$$

$\tilde{s_1}(v_s^i)$ considers two parameters. The first parameter is the fraction of total off-peak processing demands of allocated virtual nodes in the physical node over the processing capacity of the physical node. The second parameter considers the off-peak traffic by finding the fraction of total off-peak traffic passing the physical node over the physical bandwidth capacity of the node. Note that in this paper, we assume that the total traffic in a substrate node is the total traffic in its incoming physical links plus the total traffic in its outgoing physical links. This also defines the switching capacity of a substrate node. Accordingly, $\tilde{s_1}(v_s^i)$ is concerned with the off-peak traffic intensity by considering both incoming and outgoing off-peak traffic to the substrate node. However, if the node is the source/sink of traffic, it considers only one of the flows because the other one is zero.

Consequently, in this problem, the higher physical node's stress rate means that the node is utilized more in terms of off-peak processing and traffic demands. In this regard, we do not re-map virtual nodes that are mapped onto physical nodes with $\tilde{s_1}(v_s^i) \geq \mathcal{T}$ to decrease the traffic disruptions due to the reconfiguration. $\mathcal{T}$ is the stress rate's threshold, and it is a real number between 0 and 1. Providers could adjust $\mathcal{T}$. Decreasing $\mathcal{T}$ decreases the amount of energy that the programs could save, but it also reduces traffic interruptions due to the reconfiguration. This is because a smaller number of physical nodes and links are considered for energy saving.

We formulate ONL-LNLs-F as a MILP. ONL-LNLs-F is a combination of two sub-problems with the same objective. The first sub-problem is a node reconfiguration problem that reconfigures the mapping of virtual nodes. The second sub-problem is a link reconfiguration problem that reconfigures the mapping of virtual links. The link reconfiguration sub-problem can be formulated as a multi-commodity flow problem. In the context of this sub-problem, a virtual link is a commodity. Multiple virtual links are multiple commodities. To achieve the optimal result, it is necessary to formulate a single problem that combines both of the sub-problems.

The substrate network is known to the program, and therefore, the bandwidth capacity of every physical node and link and the processing capacity of each physical node are given. The allocated virtual networks are also identified, and thus, $\phi(v_n^k, v_s^i)$ for any virtual node in every physical node is specified. In addition, the off-peak processing demand $\acute{C}_c(v_n^k)$ of every virtual node is known to the program. In the context of the link reconfiguration sub-problem, we have $L_n$ commodities for the $n$th virtual network. For each $l_n^{a_m,b_m}$, as a commodity, we are given a scalar $\acute{r}_n^m$ that refers to the virtual link's off-peak traffic demand. Additionally, $\acute{r}_n^{i,j}(m)$ for every virtual link in each physical link is specified. Moreover, $\varrho'^m_n(v_s^i)$ is 1 if $v_s^i$ is a relay substrate node for $l_n^{a_m,b_m}$. $v_s^i$ is a relay node for $l_n^{a_m,b_m}$ if the source or sink virtual node of $l_n^{a_m,b_m}$ is not mapped onto $v_s^i$, while $v_s^i$ is on the allocated path for $l_n^{a_m,b_m}$ and forwards its traffic. Otherwise, $\varrho'^m_n(v_s^i)$ is 0.

Therefore, ONL-LNLs-F is formulated as a MILP as follows:

*Optimization variables:*

- $\alpha(v_s^i)$ is a binary variable. It denotes the status of $v_s^i$. $\alpha(v_s^i)$ is 1 in the case where $v_s^i$ is active; otherwise, it is 0.
- $\alpha(l_s^{i,j})$ is a binary variable. It refers to the status of $l_s^{i,j}$. $\alpha(l_s^{i,j})$ is 1 in the case where $l_s^{i,j}$ is active; otherwise, it is 0.
- $\tilde{\phi}(v_n^k, v_s^i)$ is a binary variable that expresses whether the virtual node $v_n^k$ is re-allocated in the physical node $v_s^i$ during the reconfiguration process. If the program re-allocates $v_n^k$ in $v_s^i$, then $\tilde{\phi}(v_n^k, v_s^i)$ is 1. Otherwise, $\tilde{\phi}(v_n^k, v_s^i)$ is 0.
- $f'^{i,j}_n(m)$ is a real-valued variable. It denotes the re-allocated off-peak traffic capacity to the $m$th virtual link of the $n$th virtual network in $l_s^{i,j}$. $f'^{i,j}_n(m)$ is the result of the node reconfiguration sub-problem.
- $\dot{\omega}_n^m$ is a binary variable. It is 1 if the program sets at least a relay node for $l_n^{a_m,b_m}$ into sleep mode. Otherwise, it is 0.
- $\ddot{\omega}_n^m$ is a binary variable. It is 1 if the program re-allocates the source virtual node $v_n^{a_m}$, the sink virtual node $v_n^{b_m}$ of $l_n^{a_m,b_m}$, or both of them to another physical node. Otherwise, it is 0.
- $f_n^{i,j}(m)$ is a real-valued variable. It denotes the re-allocated off-peak traffic capacity to the $m$th virtual link of the $n$th virtual network in $l_s^{i,j}$. $f_n^{i,j}(m)$ is the result of the link reconfiguration sub-problem.

*Objective function:* This program intends to minimize the VNE's total node and link energy consumption during the off-peak time according to the *fixed* power model. Eq. (9)

maintains this objective.

$$Minimize\left\{\sum_{i\in V_s}\alpha\left(v_s^i\right)\tilde{p}^m\left(v_s^i\right)+\sum_{(i,j)\in E_s}\alpha\left(l_s^{i,j}\right)\tilde{p}^m\left(l_s^{i,j}\right)\right\} \quad (9)$$

*Constraints:* A substrate node might host virtual nodes, and it may also relay the traffic of other virtual nodes. In this regard, setting a substrate node into sleep mode requires satisfying some constraints to maintain virtual networks' off-peak processing and traffic demands. In regard to the node reconfiguration sub-problem, if the program decides to set a physical node $v_s^i$ into sleep mode, it needs to suggest an alternative physical node for each of the allocated virtual nodes in $v_s^i$. Eq. (10) preserves this condition. If $v_n^k$ is already allocated in $v_s^i$, then $\phi(v_n^k, v_s^i)$ is 1. In the case where the program sets $v_s^i$ into sleep mode, $\alpha(v_s^i)$ is 0. Therefore, $\sum_{j\in V_s}\tilde{\phi}(v_n^k, v_s^j)$ must be 1, which means that $v_n^k$ is re-allocated into an alternative physical node. This constraint also ensures that the program re-allocates a virtual node only in a single alternative physical node. However, if $v_n^k$ is not allocated in $v_s^i$, then $\phi(v_n^k, v_s^i)$ is 0, and therefore, the constraint in Eq. (10) is always satisfied. Note that if the program leaves $v_s^i$ active, then none of the allocated virtual nodes in $v_s^i$ can be re-allocated.

$$\phi\left(v_n^k, v_s^i\right)\left(1-\alpha\left(v_s^i\right)-\sum_{j\in V_s}\tilde{\phi}\left(v_n^k, v_s^j\right)\right)=0,$$
$$\forall i\in V_s,\quad \forall n\in\{n|G_n\in\Phi\},\quad \forall k\in V_n \quad (10)$$

The alternative physical node needs to have some specifications. First, it has to support the off-peak processing demand $\acute{C}_c(v_n^k)$ of $v_n^k$. In this regard, as stated in the constraint in Eq. (11), the physical processing capacity $C_c(v_s^i)$ of every substrate node must be equal to or greater than the total off-peak processing rate of already allocated and not re-mapped virtual nodes in the substrate node $(\sum_{\{n|G_n\in\Phi\}}\sum_{k\in V_n}\phi(v_n^k, v_s^i)\acute{C}_c(v_n^k))$ plus the total off-peak processing demand of re-allocated virtual nodes in the substrate node $(\sum_{\{n|G_n\in\Phi\}}\sum_{k\in V_n}\tilde{\phi}(v_n^k, v_s^i)\acute{C}_c(v_n^k))$.

$$\sum_{\{n|G_n\in\Phi\}}\sum_{k\in V_n}\phi\left(v_n^k, v_s^i\right)\acute{C}_c\left(v_n^k\right)+\sum_{\{n|G_n\in\Phi\}}\sum_{k\in V_n}\tilde{\phi}\left(v_n^k, v_s^i\right)\acute{C}_c\left(v_n^k\right)\le C_c\left(v_s^i\right),\quad \forall i\in V_s \quad (11)$$

Second, the replaced physical node must be active. The constraint in Eq. (12) retains this concern. If $v_s^i$ was set into sleep mode, then $\alpha(v_s^i)$ is 0, and therefore, $\sum_{\{n|G_n\in\Phi\}}\sum_{k\in V_n}\tilde{\phi}(v_n^k, v_s^i)$ must be 0, which means that no virtual node can be re-allocated in $v_s^i$. This constraint, together with the constraint in Eq. (10), ensures that a virtual node cannot be re-allocated in the original substrate node, which is the physical node that the virtual node was previously allocated in. Note that $B_1$, $B_2$, $B_3$, $B_4$, and $B_5$ are integer numbers, and they must be large enough to be greater than the left-hand side of their respective equations.

$$\sum_{\{n|G_n\in\Phi\}}\sum_{k\in V_n}\tilde{\phi}\left(v_n^k, v_s^i\right)\le B_1\alpha\left(v_s^i\right),\quad \forall i\in V_s \quad (12)$$

Third, each virtual node is associated with one or multiple virtual links. Hence, as indicated in the constraints presented in Eq. (13), we do not re-allocate the source virtual node $v_n^{a_m}$ and the sink virtual node $v_n^{b_m}$ of the $m$th virtual link in a single substrate node. This is the case in the majority of the available VNE embedding methods.

$$\tilde{\phi}\left(v_n^{a_m}, v_s^i\right)+\tilde{\phi}\left(v_n^{b_m}, v_s^i\right)\le 1,$$
$$\phi\left(v_n^{a_m}, v_s^i\right)+\tilde{\phi}\left(v_n^{b_m}, v_s^i\right)\le 1,$$
$$\tilde{\phi}\left(v_n^{a_m}, v_s^i\right)+\phi\left(v_n^{b_m}, v_s^i\right)\le 1,$$
$$\forall i\in V_s,\quad \forall n\in\{n|G_n\in\Phi\},\quad m=1,2,\ldots,L_n \quad (13)$$

**Table 1**
Re-allocation combinations for virtual nodes of $l_n^{a_m,b_m}$ and the required off-peak traffic to be rerouted.

| $\phi(v_n^{a_m}, v_s^i)$ | $\tilde{\phi}(v_n^{a_m}, v_s^i)$ | $\phi(v_n^{b_m}, v_s^j)$ | $\tilde{\phi}(v_n^{b_m}, v_s^j)$ | Off-peak traffic to be rerouted |
|---|---|---|---|---|
| 0 | 1 | 0 | 1 | $\acute{r}_n^m$ |
| 0 | 1 | 1 | 0 | $\acute{r}_n^m$ |
| 1 | 0 | 0 | 1 | $\acute{r}_n^m$ |
| 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |

Fourth, it must be possible to re-map the off-peak traffic demand of all the virtual links belonging to the re-allocated virtual nodes. In this regard, we first find the off-peak traffic demand between virtual nodes in every pair of modified substrate nodes to consider the re-allocation of virtual nodes. Assuming a pair of substrate nodes such as $v_s^i$ and $v_s^j$, Table 1 summarizes the possible re-allocation combinations for the source virtual node $v_n^{a_m}$ and the sink virtual node $v_n^{b_m}$ of $l_n^{a_m,b_m}$ in these substrate nodes.

Table 1 determines the amount of off-peak traffic demand that needs to be re-allocated due to reconfiguring the mapping of virtual links between virtual nodes on every pair of substrate nodes. For instance, the first row indicates that the source virtual node of $l_n^{a_m,b_m}$ is re-allocated in $v_s^i$ and the sink virtual node of $l_n^{a_m,b_m}$ is re-allocated in $v_s^j$; thus, the off-peak traffic demand of $\acute{r}_n^m$ needs to be re-mapped from the virtual node in $v_s^i$ to the virtual node in $v_s^j$. However, in the second row, the source virtual node of $l_n^{a_m,b_m}$ is re-allocated in $v_s^i$, while its sink virtual node is already mapped onto $v_s^j$, and it is not modified. Therefore, the off-peak traffic demand of $\acute{r}_n^m$ needs to be re-mapped from the virtual node in $v_s^i$ to the virtual node in $v_s^j$. The program derives the amount of off-peak traffic capacity that is required to be re-allocated between virtual nodes in any pair of physical nodes for every virtual link.

Note that when both the source and sink virtual nodes of a virtual link are re-allocated in other physical nodes (first row), the off-peak traffic demand should be re-mapped only between the re-allocated source and sink virtual nodes. However, according to Table 1, in this case, the off-peak traffic demand might also be re-mapped between the original virtual node and the re-allocated virtual nodes (second or third row), which is not desirable.

Consequently, $d_n^{i,j}(m)\acute{r}_n^m$ is the off-peak traffic demand of $l_n^{a_m,b_m}$ that needs to be re-allocated from the virtual node in $v_s^i$ to the virtual node in $v_s^j$. $d_n^{i,j}(m)$ can be derived as expressed in Eq. (15).

Hence, by specifying the amount of every off-peak traffic demand that needs to be re-mapped, the constraint in Eq. (14) preserves one or multiple paths between the reconfigured source and sink virtual nodes. The constraint in Eq. (14) and the objective function force the program to choose the path with the minimum number of active physical links. In the case where there is no path between the modified virtual nodes, the program does not reconfigure the mapping of them.

$$\sum_{\{j|(i,j)\in E_s\}} f'^{i,j}_n(m) - \sum_{\{j|(j,i)\in E_s\}} f'^{j,i}_n(m) = \sum_{j\in V_s} d_n^{i,j}(m)\acute{r}_n^m - \sum_{j\in V_s} d_n^{j,i}(m)\acute{r}_n^m,$$
$$\forall i \in V_s, \quad \forall n \in \{n|G_n \in \Phi\}, \quad m = 1, 2, \ldots, L_n \tag{14}$$

where:

$$d_n^{i,j}(m) = \left(1 - \sum_{x\in V_s}\sum_{y\in V_s} \tilde{\phi}(v_n^{a_m}, v_s^x)\tilde{\phi}(v_n^{b_m}, v_s^y)\right) \times \left(\phi\left(v_n^{a_m}, v_s^i\right)\tilde{\phi}\left(v_n^{b_m}, v_s^j\right) + \phi\left(v_n^{b_m}, v_s^j\right)\tilde{\phi}\left(v_n^{a_m}, v_s^i\right)\right) + \tilde{\phi}\left(v_n^{a_m}, v_s^i\right)\tilde{\phi}\left(v_n^{b_m}, v_s^j\right) \tag{15}$$

As discussed, a substrate node $v_s^i$ might also forward the traffic of other virtual nodes and play the relay role. Therefore, it is essential to re-map the virtual links that $v_s^i$ relay their traffic before setting $v_s^i$ into sleep mode. This is the link reconfiguration sub-problem. In this regard, if one or multiple relay substrate nodes for $l_n^{a_m,b_m}$ are set into sleep mode, Eq. (16) sets the binary variable $\dot{\omega}_n^m$ to 1. Otherwise, $\dot{\omega}_n^m$ is set to 0. In addition, if the source or sink virtual nodes of $l_n^{a_m,b_m}$ or both of them are re-allocated, Eq. (17) sets the binary variable $\ddot{\omega}_n^m$ to 1. Otherwise, $\ddot{\omega}_n^m$ is 0. In this phase, we intend to re-map only the virtual links in which one or more of their relay nodes over their allocated paths are set into sleep mode, while their source and sink virtual nodes are not re-allocated. By finding the values of $\dot{\omega}_n^m$ and $\ddot{\omega}_n^m$ for every virtual link, $\dot{\omega}_n^m - \dot{\omega}_n^m\ddot{\omega}_n^m$ indicates whether the virtual link needs to be re-mapped ($\dot{\omega}_n^m - \dot{\omega}_n^m\ddot{\omega}_n^m = 1$) or not ($\dot{\omega}_n^m - \dot{\omega}_n^m\ddot{\omega}_n^m = 0$). The constraint in Eq. (18) re-maps the off-peak traffic demand of virtual links in which one or more of their relay substrate nodes are set into sleep mode and their source and sink virtual nodes are not re-allocated. Note that the constraint in Eq. (14) re-maps a virtual link, as a part of the node reconfiguration sub-problem, if its source or sink virtual node, or both of them, are re-allocated in other physical nodes. Therefore, the program does not consider the re-mapping of such a virtual link again.

$$\dot{\omega}_n^m \le \sum_{i\in V_s} \varrho'^m_n\left(v_s^i\right)\left(1 - \alpha\left(v_s^i\right)\right) \le B_2\dot{\omega}_n^m, \tag{16}$$

$$\ddot{\omega}_n^m \le \sum_{i\in V_s} \left(\tilde{\phi}\left(v_n^{a_m}, v_s^i\right) + \tilde{\phi}\left(v_n^{b_m}, v_s^i\right)\right) \le B_3\ddot{\omega}_n^m,$$
$$\forall n \in \{n|G_n \in \Phi\}, \quad m = 1, 2, \ldots, L_n \tag{17}$$

$$\sum_{\{j|(i,j)\in E_s\}} f_n^{i,j}(m) - \sum_{\{j|(j,i)\in E_s\}} f_n^{j,i}(m) = \begin{cases} (\dot{\omega}_n^m - \dot{\omega}_n^m\ddot{\omega}_n^m)\acute{r}_n^m & \text{if} \quad i = a_m \\ (\dot{\omega}_n^m\ddot{\omega}_n^m - \dot{\omega}_n^m)\acute{r}_n^m & \text{if} \quad i = b_m \\ 0 & \text{otherwise} \end{cases},$$
$$\forall i \in V_s, \quad \forall n \in \{n|G_n \in \Phi\}, \quad m = 1, 2, \ldots, L_n \tag{18}$$

Re-allocating virtual links by either the constraint in Eq. (14) or the constraint in Eq. (18) also requires satisfying some other constraints. First, the program ensures in the constraint in Eq. (19) that no rerouted traffic passes through the sleeping physical nodes. The constraint in Eq. (19) ensures that the total outgoing re-allocated off-peak traffic capacity $\sum_{\{j|(i,j)\in E_s\}}\sum_{\{n|G_n\in\Phi\}}\sum_{m=1}^{L_n}(f_n^{i,j}(m) + f'^{i,j}_n(m))$ and the total incoming re-allocated off-peak traffic capacity $\sum_{\{j|(j,i)\in E_s\}}\sum_{\{n|G_n\in\Phi\}}\sum_{m=1}^{L_n}(f_n^{j,i}(m) + f'^{j,i}_n(m))$ in a sleeping physical node ($\alpha(v_s^i) = 0$) are 0.

$$\sum_{\{j|(i,j)\in E_s\}}\sum_{\{n|G_n\in\Phi\}}\sum_{m=1}^{L_n}\left(f_n^{i,j}(m) + f'^{i,j}_n(m)\right) + \sum_{\{j|(j,i)\in E_s\}}\sum_{\{n|G_n\in\Phi\}}\sum_{m=1}^{L_n}\left(f_n^{j,i}(m) + f'^{j,i}_n(m)\right) \le B_4\alpha\left(v_s^i\right), \quad \forall i \in V_s \tag{19}$$

Second, the total embedded off-peak traffic capacity in every substrate link must be equal to or less than its physical bandwidth capacity, as indicated in the constraint in Eq. (20). In this program, three types of traffic capacities form the total embedded off-peak traffic capacity in a substrate link $l_s^{i,j}$. The first type is the off-peak traffic capacity of the virtual links that are not re-allocated through the reconfiguration process. In this regard, none of the relay substrate nodes in the already allocated paths for the virtual link are set into sleep mode, and therefore, $1 - \dot{\omega}_n^m = 1$. Moreover, the source and sink virtual nodes of the virtual link are not re-allocated during the reconfiguration process; thus, $1 - \ddot{\omega}_n^m = 1$. Nonetheless, if the virtual link is re-allocated, this off-peak traffic is not considered. The second type is the re-allocated off-peak traffic capacity $f'^{i,j}_n(m)$ to a virtual link in the substrate link, as the result of the node reconfiguration sub-problem. However, the third type is the re-allocated off-peak traffic capacity $f_n^{i,j}(m)$ to a virtual link in the substrate link, as the result of the link reconfiguration sub-problem. Eq. (21) defines the total traffic in a substrate link.

$$r\left(l_s^{i,j}\right) \le C_b\left(l_s^{i,j}\right), \quad \forall (i,j) \in E_s \tag{20}$$

where

$$r\left(l_s^{i,j}\right) = \sum_{\{n|G_n\in\Phi\}}\sum_{m=1}^{L_n}\left(\left(1 - \dot{\omega}_n^m\right)(1 - \ddot{\omega}_n^m)\acute{r}_n^{i,j}(m) + f'^{i,j}_n(m) + f_n^{i,j}(m)\right) \tag{21}$$

Third, the total embedded off-peak traffic capacity in every substrate node must be equal to or less than its physical bandwidth capacity, as indicated in Eq. (22). Note that the total traffic in a substrate node is the total traffic in its incoming physical links plus the total traffic in its outgoing physical

links. Eq. (23) defines the total traffic in a substrate node.

$$r\left(v_s^i\right) \le C_b\left(v_s^i\right), \quad \forall i \in V_s \tag{22}$$

where

$$r(v_s^i) = \sum_{\{j|(i,j)\in E_s\}} r\left(l_s^{i,j}\right) + \sum_{\{j|(j,i)\in E_s\}} r\left(l_s^{j,i}\right) \tag{23}$$

Furthermore, we do not re-allocate the allocated virtual nodes in highly stressed substrate nodes, and the virtual links in all the physical nodes along their allocated paths are highly stressed. The constraint in Eq. (24) requires that a physical node $v_s^i$ with $\tilde{s_1}(v_s^i) \ge \mathcal{T}$ must stay active $\alpha(v_s^i) = 1$. Consequently, Eq. (10) does not allow the program to re-allocate any allocated virtual nodes in the physical node $v_s^i$. Moreover, Eqs. (16) and 17 ensure that a virtual link in all the physical nodes along their allocated paths are highly stressed is not re-allocated.

$$\forall i \in \{i|i \in V_s, \tilde{s_1}\left(v_s^i\right) \ge \mathcal{T}\}: \quad \alpha\left(v_s^i\right) = 1 \tag{24}$$

The constraint in Eq. (25) makes the program linear by setting the auxiliary binary variable $\alpha(l_s^{i,j})$ to 1 if there is any traffic in $l_s^{i,j}$. Otherwise, $\alpha(l_s^{i,j})$ is 0.

$$r\left(l_s^{i,j}\right) \le B_5\alpha\left(l_s^{i,j}\right), \quad \forall (i,j) \in E_s \tag{25}$$

The variables must also be in the following bounds:

$$f_n^{i,j}(m) \ge 0, \quad f'^{i,j}_n(m) \ge 0,$$
$$\forall (i,j) \in E_s, \quad \forall n \in \{n|G_n \in \Phi\}, \quad m = 1, 2, \ldots, L_n \tag{26}$$

$$\alpha\left(l_s^{i,j}\right) \in \{0,1\}, \quad \forall (i,j) \in E_s \tag{27}$$

$$\alpha(v_s^i) \in \{0,1\}, \quad \forall i \in V_s \tag{28}$$

$$\tilde{\phi}(v_n^k, v_s^i) \in \{0,1\}, \quad \forall i \in V_s,$$
$$\forall n \in \{n|G_n \in \Phi\}, \quad \forall k \in V_n \tag{29}$$

$$\dot{\omega}_n^m \in \{0,1\}, \ddot{\omega}_n^m \in \{0,1\}, \quad \forall n \in \{n|G_n \in \Phi\},$$
$$m = 1, 2, \ldots, L_n \tag{30}$$

There are some constraints in the program that include the product of binary variables. This issue makes the program nonlinear. However, a product of two binary variables can be replaced by one new binary variable, on which a number of constraints are imposed to the program [21], and therefore, the program remains linear. The extension to the products of more binary variables is straightforward. Considering the binary variables $x_1$ and $x_2$, their product $x_1x_2$ could be replaced by a new binary variable $y$, while the following constraints force $y$ to take the value of $x_1x_2$.

$$y \le x_1 \tag{31}$$

$$y \le x_2 \tag{32}$$

$$y \ge x_1 + x_2 - 1 \tag{33}$$

$$y \in \{0,1\} \tag{34}$$

As described, the formulated MILP for ONL-LNLs-F includes two sub-problems. The link reconfiguration sub-problem can be reduced to the problem discussed in [22], which is a simple two-commodity integer flow problem. It is proven in [22] that this simple two-commodity integer flow problem is $\mathcal{NP}$-hard. Hence, our formulated MILP for off-peak node and link energy optimization by local node and link reconfiguration is $\mathcal{NP}$-hard.

#### 4.2.2. Off-peak node and link energy optimization by local link reconfiguration

We define a different stress rate for intermediate substrate nodes in this section. Accordingly, a solution is proposed to optimize the total energy consumption of the intermediate substrate nodes and substrate links during the off-peak time. We do not re-map virtual nodes in this problem. This method *might* set the less stressed intermediate substrate nodes and their respective substrate links into sleep mode for the off-peak time. We re-map a virtual link *if and only if* we sleep at least one intermediate substrate node over its embedded path. The programs also consider the VNE's link energy consumption by setting the maximum number of physical links into sleep mode during the re-allocation of virtual links. A MILP for splittable traffic and a BILP for non-splittable traffic are formulated for this strategy.

The stress rate $\tilde{s_2}(v_s^i)$ of a physical node $v_s^i$ in this problem indicates the intensity of involved VNs and the total off-peak traffic demand in the substrate node. The definition of a physical node's stress rate in this problem is different than the definition of a physical node's stress rate in the previous problem because we do not reconfigure virtual nodes in this case. A VN is involved in a substrate node $v_s^i$ if at least one of its virtual nodes is mapped onto $v_s^i$ or at least one of its mapped virtual links passes through $v_s^i$. Assume that $\eta(v_s^i)$ is the number of virtual networks involved in physical node $v_s^i$; then, the following equation defines $\tilde{s_2}(v_s^i)$.

$$\tilde{s_2}(v_s^i) = \frac{\eta(v_s^i)}{|\Phi|}\left(\frac{\sum_{\{j|(i,j)\in E_s\}}\sum_{\{n|G_n\in\Phi\}}\sum_{m=1}^{L_n} \hat{r}_n^{i,j}(m)}{C_b\left(v_s^i\right)} + \frac{\sum_{\{j|(j,i)\in E_s\}}\sum_{\{n|G_n\in\Phi\}}\sum_{m=1}^{L_n} \hat{r}_n^{j,i}(m)}{C_b\left(v_s^i\right)}\right) \tag{35}$$

$\tilde{s_2}(v_s^i)$ considers two parameters. The first parameter is the fraction of the number of involved VNs in the node over the total number of active VNs. This parameter denotes the intensity of involved VNs in $v_s^i$. However, the second parameter considers the off-peak traffic demand by finding the fraction of total off-peak traffic that passes the substrate node over the bandwidth capacity of the node. Consequently, in this problem, a higher physical node's stress rate means that a larger number of VNs are involved in the node, while the substrate node is utilized more in terms of off-peak traffic. In this regard, we do not set the intermediate physical nodes with $\tilde{s_2}(v_s^i) \ge \mathcal{T}$ into sleep mode. Virtual links where the stress rate of all the physical nodes along their allocated paths are equal to or greater than $\mathcal{T}$ are also not re-mapped. These actions help to decrease traffic disruptions due to the reconfiguration. Note that the threshold value of $\mathcal{T}$ is adjustable. Providers can adjust the value of $\mathcal{T}$ and therefore manage the level of the reconfiguration. Decreasing the $\tilde{s_2}$ threshold decreases the amount of energy that the programs could save and reduces possible traffic interruptions because a smaller

number of physical nodes and links will be considered for energy saving. The impact of different values for the stress rate threshold on the energy-saving ability of the solutions is discussed in Section 6.

The traffic might be splittable or non-splittable. In the splittable case, the traffic demand of each virtual link could be carried by one or multiple paths in the substrate network. However, if the traffic is non-splittable, each virtual link's traffic demand may be required to follow the same path through the network rather than be divided among multiple paths. It is expected to save higher amounts of energy when traffic is splittable because we are more flexible in terms of finding alternative paths. This is a major restriction that has an important impact on the solution methodology. In this regard, we formulate off-peak node and link energy optimization by local link reconfiguration for both splittable and non-splittable traffic.

*4.2.2.1. Splittable traffic (ONL-LLs-F).* The defined local link reconfiguration problem, in the case of splittable traffic, can be formulated as a MILP in the category of multi-commodity flow problems. In the context of this problem, every virtual link is a commodity. The off-peak node and link energy optimization by local link reconfiguration that is formulated based on the *fixed* power model while traffic is splittable is called ONL-LLs-F.

Similar to the previous problem, the substrate network topology is specified. The physical bandwidth capacity of every substrate node and link is known. Allocated virtual networks are also identified. In this problem, we have $L_n$ commodities for the $n$th virtual network. For each virtual link $l_n^{a_m,b_m}$, as a commodity, we are given a scalar $\acute{r}_n^m$ that refers to the virtual link's off-peak traffic demand. $\acute{r}_n^{i,j}(m)$ for every virtual link in each physical link is also known. Moreover, $\varrho_n^m(v_s^i)$ indicates whether the allocated path for $l_n^{a_m,b_m}$ includes $v_s^i$. $\varrho_n^m(v_s^i)$ is 1 if the allocated path for $l_n^{a_m,b_m}$ contains $v_s^i$. Otherwise, $\varrho_n^m(v_s^i)$ is 0.

Therefore, ONL-LLs-F is formulated as a MILP as follows:

*Optimization variables:*

- $\alpha(v_s^i)$ is a binary variable. It denotes the status of $v_s^i$. $\alpha(v_s^i)$ is 1 in the case where $v_s^i$ is active; otherwise, it is 0.
- $\alpha(l_s^{i,j})$ is a binary variable. It refers to the status of $l_s^{i,j}$. $\alpha(l_s^{i,j})$ is 1 in the case where $l_s^{i,j}$ is active; otherwise, it is 0.
- $\omega_n^m$ is a binary variable. It is 1 in the case where at least one physical node over the allocated path for $l_n^{a_m,b_m}$ is set into sleep mode. Otherwise, it is 0.
- $f_n^{i,j}(m)$ is a real-valued variable. It denotes the re-allocated off-peak traffic capacity to the $m$th virtual link of the $n$th virtual network in $l_s^{i,j}$.

*Objective function:* This program aims to minimize the total energy consumption by physical intermediate nodes and physical links during the off-peak time according to the *fixed* power model. Eq. (36) and the constraint in Eq. (38) that allows only intermediate substrate nodes to be set into sleep mode preserve this objective.

$$Minimize\left\{\sum_{i\in V_s}\alpha\left(v_s^i\right)\tilde{p}^m\left(v_s^i\right)+\sum_{(i,j)\in E_s}\alpha\left(l_s^{i,j}\right)\tilde{p}^m\left(l_s^{i,j}\right)\right\} \quad (36)$$

*Constraints:* If the program places an intermediate physical node into sleep mode, it needs to suggest one or more alternative paths for every virtual link that was passing the substrate node. In this regard, the constraint in Eq. (37) sets the binary variable $\omega_n^m$ to 1 if at least one physical node $v_s^i$ that is on the allocated path for $l_n^{a_m,b_m}$ ($\varrho_n^m(v_s^i)=1$) is set into sleep mode ($\alpha(v_s^i)=0$). Otherwise, $\omega_n^m$ is 0. The program derives $\omega_n^m$ for every virtual link. $\omega_n^m$ determines whether $l_n^{a_m,b_m}$ is required to be re-mapped. Thus, the constraint in Eq. (38) preserves at least one alternative path from the source virtual node $v_n^{a_m}$ to the sink virtual node $v_n^{b_m}$ of $l_n^{a_m,b_m}$ if $\omega_n^m$ is 1. In the case where $\omega_n^m$ is 0, $l_n^{a_m,b_m}$ is not re-mapped. Note that the constraint in Eq. (38) does not allow a physical node in which at least one virtual node is allocated in it to be set into sleep mode. The program also considers the VNE's link energy consumption by re-allocating virtual links in the substrate paths with the minimum number of physical links. Nevertheless, it is also necessary to satisfy some other constraints when re-allocating a virtual link. Note that $B_1$, $B_2$, and $B_3$ are integer numbers, and they must be large enough to be greater than the left-hand side of their respective equations.

$$\omega_n^m \le \sum_{i\in V_s}\varrho_n^m(v_s^i)\left(1-\alpha(v_s^i)\right) \le B_1\omega_n^m,$$
$$\forall n\in\{n|G_n\in\Phi\},\quad m=1,2,\ldots,L_n \quad (37)$$

$$\sum_{\{j|(i,j)\in E_s\}} f_n^{i,j}(m) - \sum_{\{j|(j,i)\in E_s\}} f_n^{j,i}(m) = \begin{cases}\omega_n^m\acute{r}_n^m & \text{if} \quad i=a_m\\ -\omega_n^m\acute{r}_n^m & \text{if} \quad i=b_m\\ 0 & \text{otherwise}\end{cases},$$
$$\forall i\in V_s,\quad \forall n\in\{n|G_n\in\Phi\},\quad m=1,2,\ldots,L_n, \quad (38)$$

First, it is not feasible that the re-allocated off-peak traffic passes through the sleeping physical nodes. Thus, the constraint in Eq. (39) ensures that the total incoming and outgoing re-allocated traffic passing through a sleeping physical node ($\alpha(v_s^i)=0$) is 0.

$$\sum_{\{j|(i,j)\in E_s\}}\sum_{\{n|G_n\in\Phi\}}\sum_{m=1}^{L_n} f_n^{i,j}(m) + \sum_{\{j|(j,i)\in E_s\}}\sum_{\{n|G_n\in\Phi\}}\sum_{m=1}^{L_n} f_n^{j,i}(m) \le B_2\alpha(v_s^i),\quad \forall i\in V_s \quad (39)$$

Second, the total embedded off-peak traffic capacity in every substrate link must be equal to or less than its physical bandwidth capacity. Eq. (40) preserves this constraint. In this problem, two types of traffic capacities form the total embedded off-peak traffic capacity in a substrate link $l_s^{i,j}$. The first type is the embedded off-peak traffic capacity to the virtual links that are not re-allocated by the program. $1-\omega_n^m$ determines whether $l_n^{a_m,b_m}$ is re-allocated. If the program re-maps $l_n^{a_m,b_m}$, then $\omega_n^m$ is 1. Therefore, $1-\omega_n^m$ is 0, which means that $l_n^{a_m,b_m}$ is re-allocated, and thus, its off-peak traffic capacity does not exist in its original path. In contrast, if $l_n^{a_m,b_m}$ is not re-allocated, then $1-\omega_n^m$ is 1, and its off-peak traffic capacity is counted. The second type is the re-allocated off-peak traffic capacity $f_n^{i,j}(m)$ to virtual links in the substrate link. Consequently, $\sum_{\{n|G_n\in\Phi\}}\sum_{m=1}^{L_n}((1-\omega_n^m)\acute{r}_n^{i,j}(m)+f_n^{i,j}(m))$ is the

total embedded off-peak traffic capacity in $l_s^{i,j}$.

$$r(l_s^{i,j}) \le C_b(l_s^{i,j}), \quad \forall (i,j) \in E_s \tag{40}$$

where

$$r\left(l_s^{i,j}\right) = \sum_{\{n|G_n \in \Phi\}} \sum_{m=1}^{L_n} \left(\left(1-\omega_n^m\right)\acute{r}_n^{i,j}(m) + f_n^{i,j}(m)\right) \tag{41}$$

Third, the total embedded off-peak traffic capacity in a substrate node must be equal to or less than its physical bandwidth capacity $C_b(v_s^i)$, as expressed in (42). The total traffic in a substrate node is the total traffic in its incoming physical links plus the total off-peak traffic in its outgoing physical links. Eq. (43) defines the total traffic in a substrate node.

$$r\left(v_s^i\right) \le C_b(v_s^i), \quad \forall i \in V_s \tag{42}$$

where

$$r(v_s^i) = \sum_{\{j|(i,j)\in E_s\}} r(l_s^{i,j}) + \sum_{\{j|(j,i)\in E_s\}} r(l_s^{j,i}) \tag{43}$$

As discussed, we do not sleep the highly stressed physical nodes. The constraint in Eq. (44) ensures that a substrate node with $\tilde{s_2}(v_s^i) \ge \mathcal{T}$ remains active. Moreover, if all the physical nodes along the allocated path for a virtual link are highly stressed, the constraint in Eq. (37) ensures that the virtual link is not re-allocated.

$$\alpha\left(v_s^i\right) = 1, \quad \forall i \in \{i|i \in V_s, \quad \tilde{s_2}\left(v_s^i\right) \ge \mathcal{T}\} \tag{44}$$

The constraint in Eq. (45) makes the program linear, similar to the first program.

$$r\left(l_s^{i,j}\right) \le B_3\alpha\left(l_s^{i,j}\right), \quad \forall (i,j) \in E_s \tag{45}$$

The variables must also remain in the following bounds:

$$f_n^{i,j}(m) \ge 0, \quad \forall (i,j) \in E_s, \quad \forall n \in \{n|G_n \in \Phi\}, \quad m = 1,2,\ldots,L_n \tag{46}$$

$$\alpha\left(l_s^{i,j}\right) \in \{0,1\}, \quad \forall (i,j) \in E_s \tag{47}$$

$$\alpha\left(v_s^i\right) \in \{0,1\}, \quad \forall i \in V_s \tag{48}$$

$$\omega_n^m \in \{0,1\}, \quad \forall n \in \{n|G_n \in \Phi\}, \quad m = 1,2,\ldots,L_n \tag{49}$$

*4.2.2.2. Non-splittable traffic (ONL-LLns-F).* The traffic might be non-splittable, that is, each traffic demand has to be carried on the same path. The defined local link reconfiguration problem, in the case of non-splittable traffic, can be formulated as a BILP in the category of multi-commodity flow problems. Similar to ONL-LLs-F, each virtual link is a commodity in the context of this problem. A virtual link as a commodity generates traffic that flows through its allocated path. Different from when traffic is splittable, the traffic of every re-mapped virtual link must be carried in a single substrate path. The off-peak node and link energy optimization by local link reconfiguration that is formulated based on the *fixed* power model while traffic is non-splittable is called ONL-LLns-F. The input of this problem is the same as the input for the MILP formulated for ONL-LLs-F.

ONL-LLns-F is formulated as a BILP as follows:

*Optimization variables:*

- $\alpha(v_s^i)$ is a binary variable. $\alpha(v_s^i)$ denotes the status of $v_s^i$. $\alpha(v_s^i)$ is 1 in the case where $v_s^i$ is active; otherwise, it is 0.
- $\alpha(l_s^{i,j})$ is a binary variable. $\alpha(l_s^{i,j})$ refers to the status of $l_s^{i,j}$. $\alpha(l_s^{i,j})$ is 1 in the case where $l_s^{i,j}$ is active; otherwise, it is 0.
- $\omega_n^m$ is a binary variable. $\omega_n^m$ is 1 in the case where at least one physical node over the allocated path for $l_n^{a_m,b_m}$ is set into sleep mode. Otherwise, $\omega_n^m$ is 0.
- $z_n^{i,j}(m)$ is a binary variable. $z_n^{i,j}(m)$ is 1 in the case where the re-mapped path to $l_n^{a_m,b_m}$ includes $l_s^{i,j}$. Otherwise, $z_n^{i,j}(m)$ is 0.

*Objective function:* The same objective as Eq. (36).

*Constraints:* The constraints in Eqs. (37), (40), (42), (44), and (45) and the following: This program works almost the same as the program formulated for ONL-LLs-F. However, in contrast to the program developed for ONL-LLs-F and because traffic is non-splittable, the BILP must find only a single alternative path for any re-allocated virtual link. In this regard, if the program needs to re-map $l_n^{a_m,b_m}$, then $\omega_n^m$ is 1. Therefore, the constraint in Eq. (50) requires routing a single unit of data from $v_n^{a_m}$ to $v_n^{b_m}$. Because the variable $z_n^{i,j}(m)$ is binary, the unit of data could not be split. Moreover, Eq. (51) limits the program routing such that the maximum number of incoming and outgoing flows for any node is two to prevent a loop. Thus, the driven route will be a single path from node $v_n^{a_m}$ to $v_n^{b_m}$, which will be used as the replaced path for $l_n^{a_m,b_m}$.

$$\sum_{\{j|(i,j)\in E_s\}} z_n^{i,j}(m) - \sum_{\{j|(j,i)\in E_s\}} z_n^{j,i}(m) = \begin{cases} \omega_n^m & if\ i = a_m \\ -\omega_n^m & if\ i = b_m \\ 0 & otherwise \end{cases}, \quad \forall i \in V_s, \quad \forall n \in \{n|G_n \in \Phi\}, \quad m = 1,2,\ldots,L_n \tag{50}$$

$$\sum_{\{j|(i,j)\in E_s\}} z_n^{i,j}(m) + \sum_{\{j|(j,i)\in E_s\}} z_n^{j,i}(m) \le 2, \quad \forall i \in V_s, \quad \forall n \in \{n|G_n \in \Phi\}, \quad m = 1,2,\ldots,L_n, \tag{51}$$

The constraint in Eq. (52) ensures that any rerouted traffic flow does not pass through the sleeping nodes.

$$\sum_{\{j|(i,j)\in E_s\}} \sum_{\{n|G_n\in\Phi\}} \sum_{m=1}^{L_n} z_n^{i,j}(m) + \sum_{\{j|(j,i)\in E_s\}} \sum_{\{n|G_n\in\Phi\}} \sum_{m=1}^{L_n} z_n^{j,i}(m) \le B_2\alpha(v_s^i), \quad \forall i \in V_s \tag{52}$$

Note that the total embedded off-peak traffic capacity in any physical link is calculated similar to the previous program, but here, $z_n^{i,j}(m)\acute{r}_n^m$ determines the re-allocated off-peak traffic capacity to a virtual link $l_n^{a_m,b_m}$ in $l_s^{i,j}$. If the off-peak traffic demand $\acute{r}_n^m$ of $l_n^{a_m,b_m}$ is rerouted through $l_s^{i,j}$, then $z_n^{i,j}(m)\acute{r}_n^m$ is equal to $\acute{r}_n^m$. Otherwise, this amount is 0. This is reflected in Eq. (53).

$$r\left(l_s^{i,j}\right) = \sum_{\{n|G_n \in \Phi\}} \sum_{m=1}^{L_n} \left(\left(1-\omega_n^m\right)\hat{r}_n^{i,j}(m) + z_n^{i,j}(m)\hat{r}_n^m\right) \quad (53)$$

The variables must hold in the bounds in Eqs. 47–(49) and the following:

$$z_n^{i,j}(m) \in \{0,1\}, \quad \forall (i,j) \in E_s, \quad \forall n \in \{n|G_n \in \Phi\}, \\ m = 1,2,\ldots,L_n \quad (54)$$

The rest of the program works similar to the program formulated for ONL-LLs-F.

Both of the formulated integer linear programs for ONL-LLs-F and ONL-LLns-F can be reduced to the problem discussed in [22], which is a simple two-commodity integer flow problem. It is proven in [22] that this simple two-commodity integer flow problem is $\mathcal{NP}$-hard. Hence, the formulated ILPs for off-peak node and link energy optimization by local link reconfiguration, either in the case of splittable traffic demands or non-splittable traffic demands, are $\mathcal{NP}$-hard.

### 4.3. Programs based on the semi-proportional power model

The previous section developed energy-saving programs for the VNE's nodes and links conforming to the *fixed* link power model. According to the *fixed* link power model, an active physical node and link consumes a constant amount of energy, regardless of its traffic load. Therefore, the formulated programs reduce the VNE's total node and link energy consumption by setting physical nodes and links into sleep mode.

Nevertheless, considering a constant amount of power for an active physical node or link is not efficient. As discussed in Section 3, the *semi-proportional* link power model defines a traffic-adaptive power model for a physical node or link. Based on this power model, a large portion of the power consumed by a physical node or link is for keeping the device operational. Nonetheless, in contrast to the *fixed* link power model, the traffic load in the network element also changes its energy consumption. In this regard, it may be possible to reduce the node or link energy consumption by either setting the device into sleep mode or by rerouting its traffic load to the other active part of the network. However, we could save a larger amount of energy by sleeping the device compared to rerouting its traffic load.

Consequently, the energy-saving solutions for the *semi-proportional* power model are different from the *fixed* link power model solutions. The programs defined in the previous section are required to be modified to optimize the energy based on the *semi-proportional* power model. In this regard, it is essential to modify the objective functions defined in each of the formulated programs to minimize the node and link energy consumption based on the *semi-proportional* power model. The power consumption of a single physical node according to the *semi-proportional* power model is defined in Eq. (3). In addition, the power consumption of a single physical link based on this model is also defined in Eq. (4). Accordingly, it is necessary to replace the stated objective functions in Eqs. (9) and (36) by the new objective function defined in Eq. (55), which is formulated based on the *semi-proportional* power model.

$$Minimize \left\{ \sum_{i \in V_s} \left( \alpha\left(v_s^i\right)\tilde{p}^b\left(v_s^i\right) + \frac{r\left(v_s^i\right)}{C_b\left(v_s^i\right)}\left(\tilde{p}^m\left(v_s^i\right) - \tilde{p}^b\left(v_s^i\right)\right)\right) \right. \\ \left. + \sum_{(i,j)\in E_s} \left( \alpha\left(l_s^{i,j}\right)\tilde{p}^b\left(l_s^{i,j}\right) + \frac{r\left(l_s^{i,j}\right)}{C_b\left(l_s^{i,j}\right)}\left(\tilde{p}^m\left(l_s^{i,j}\right) - \tilde{p}^b\left(l_s^{i,j}\right)\right)\right) \right\} \quad (55)$$

By modifying the objective functions of the formulated programs and keeping the same constraints and bounds, the programs optimize the node and link energy consumption of the VNE during the off-peak period based on the *semi-proportional* power model.

Note that off-peak node and link energy optimization by local node and link reconfiguration, which is formulated based on the *semi-proportional* power model, while traffic is splittable is called ONL-LNLs-SP. Moreover, the off-peak node and link energy optimization by the local link reconfiguration problem that is defined according to the *semi-proportional* power model is called ONL-LLs-SP in the case of splittable traffic and ONL-LLns-SP in the case of non-splittable traffic.

## 5. Heuristic algorithm

The BILP for ONL-LLns-F discussed in Section 4.2.2.2 adjusts some of the already allocated virtual links to reduce the total node and link energy consumption in the VNE during the off-peak time while traffic is non-splittable. Nevertheless, the BILP formulated for ONL-LLns-F is $\mathcal{NP}$-hard, and therefore, the optimization solution is not scalable in the case of large network sizes due to the long execution time. Hence, in this section, we propose an energy-saving heuristic for ONL-LLns-F. The pseudocode of the heuristic is shown in Algorithm 1.

We know from *semi-proportional* power models that a very large part of the power consumption of a physical node or link is for its base power consumption rather than the traffic-related power consumption. Therefore, targeting only the base power consumption of a device still provides an effective energy-saving method. Hence, to simplify the heuristic and decrease its time complexity, it is defined based on the *fixed* power model. This means that the heuristic does not save energy by rerouting the traffic.

Accordingly, the physical nodes consume more energy than the physical links. In this regard, the algorithm first calculates some metrics and then sorts the intermediate physical nodes according to their stress rate to check the node removal possibility. Subsequently, it attempts to find an alternative path, with the minimum number of physical links, for the off-peak traffic of every virtual link that was passing the sleeping physical nodes.

The stress rate $\tilde{s_2}(v_s^i)$ for a substrate node $v_s^i$ can be found in a way similar to that calculated in Eq. (35). $\tilde{s_2}(v_s^i)$ defines the intensity of involved VNs and the total off-peak traffic in a substrate node $v_s^i$. This helps us in sorting and decision-making matters in the following phases.

A virtual link might get mapped onto a single physical link or multiple physical links. In either case, the requested bandwidth will be allocated in the physical nodes and links. Because the virtual networks' traffic rates are decreased during

**Algorithm 1** Heuristic algorithm for ONL-LLns-F.

**for all** $i \in V_s$ **do**
  **if** $\tilde{s_2}(v_s^i) < \mathcal{T}$, and $v_s^i$ is an intermediate substrate node **then**
    place $v_s^i$ in *S_L*, in ascending order, based on $\tilde{s_2}$
  **end if**
  **if** $\tilde{s_2}(v_s^i) = 0$ **then**
    remove $v_s^i$ and all its respective physical links from $G_s^T$
  **end if**
**end for**
**for all** $i$ such that $v_s^i$ is the top unchecked physical node in *S_L* **do**
  remove $v_s^i$ and all its respective physical links from $G_s^T$
  **for all** virtual links $l_n^{a_m,b_m}$ pass through $v_s^i$ **do**
    find $K$-shortest path from $a_m$ to $b_m$ in $G_s^T$
    **for all** $K$ found paths **do**
      **for all** $(x, y)$ such that $l_s^{x,y}$ is on the alternative path **do**
        $\check{C}_b(l_s^{x,y}) = \check{C}_b(l_s^{x,y}) - \hat{r}_n^m$
        **if** $\check{C}_b(l_s^{x,y}) < 0$ **then**
          $\check{C}_b(l_s^{x,y}) = \check{C}_b(l_s^{x,y}) + \hat{r}_n^m$
          undo all the modifications respective to $v_s^i$
          break, and go for the next found path
        **else**
          $\hat{r}_n^{x,y}(m) = \hat{r}_n^{x,y}(m) + \hat{r}_n^m$
        **end if**
      **end for**
      **for all** $j$ such that $v_s^j$ is an intermediate node on the alternative path **do**
        $\check{C}_b(v_s^j) = \check{C}_b(v_s^j) - 2\hat{r}_n^m$
        **if** $\check{C}_b(v_s^j) < 0$ **then**
          $\check{C}_b(v_s^j) = \check{C}_b(v_s^j) + 2\hat{r}_n^m$
          undo all the modifications respective to $v_s^i$
          break, and go for next found path
        **end if**
      **end for**
      **if** the path is capable **then**
        break from checking the rest of the found paths
      **end if**
    **end for**
    **if** re-allocation was successful for $l_n^{a_m,b_m}$ **then**
      Remove the previously allocated virtual link capacities for $l_n^{a_m,b_m}$, on any physical link
    **else**
      place $v_s^i$ and all its respective physical links back to $G_s^T$
      undo all the modifications respective to $v_s^i$
      break, and go for next substrate node in *S_L*
    **end if**
  **end for**
**end for**
**return** $G_s^T$

the off-peak time, the entire allocated virtual link capacity is not used. During this period, the off-peak demand is the reserved capacity in a substrate node or link, and the remaining bandwidth capacity could be shared. The unused off-peak bandwidth capacity in $l_s^{i,j}$ is represented by $\check{C}_b(l_s^{i,j})$. $\check{C}_b(l_s^{i,j})$ is equal to the physical link's bandwidth capacity $C_b(l_s^{i,j})$ subtracted by the total off-peak traffic demand in $l_s^{i,j}$. Eq. (56) defines $\check{C}_b(l_s^{i,j})$. Moreover, as stated in Eq. (43), the total traffic in a physical node $v_s^i$ is equal to the total outgoing and incoming traffic of the node. Accordingly, the unused off-peak bandwidth capacity $\check{C}_b(v_s^i)$ of $v_s^i$ can be defined as in Eq. (57).

$$\check{C}_b(l_s^{i,j}) = C_b(l_s^{i,j}) - \sum_{\{n|G_n \in \Phi\}} \sum_{m=1}^{L_n} \hat{r}_n^{i,j}(m) \tag{56}$$

$$\check{C}_b(v_s^i) = C_b(v_s^i) - \left( \sum_{\{j|(i,j)\in E_s\}} \sum_{\{n|G_n\in\Phi\}} \sum_{m=1}^{L_n} \hat{r}_n^{i,j}(m) + \sum_{\{j|(j,i)\in E_s\}} \sum_{\{n|G_n\in\Phi\}} \sum_{m=1}^{L_n} \hat{r}_n^{j,i}(m) \right) \tag{57}$$

Moreover, the algorithm creates an auxiliary off-peak substrate topology $G_s^T$. Initially, $G_s^T$ is the same as the substrate network topology.

By knowing the calculated metrics, the algorithm sorts all the intermediate substrate nodes with a stress rate of less than $\mathcal{T}$. Because we do not reconfigure the mapping of virtual nodes in ONL-LLns-F, the heuristic sets only capable intermediate nodes into sleep mode. In this regard, because the substrate nodes with the larger number of involved VNs are more essential for connectivity and bandwidth demands, the algorithm starts setting intermediate physical nodes into sleep mode from the node that has the smallest number of involved VNs and lowest off-peak utilization. This occurs when the algorithm sorts the intermediate physical nodes with $\tilde{s_2}(v_s^i) < \mathcal{T}$ in ascending order based on $\tilde{s_2}$. The list is represented by *S_L*. Note that the $\tilde{s_2}(v_s^i)$ threshold amount is adjustable. We investigate the impact of different values for the $\tilde{s_2}$ threshold on the heuristic's outcome in Section 6

The next step checks the possibility of removing the intermediate physical nodes and their respective physical links. This step also ensures that the rearranged network accommodates the customer's off-peak traffic demands. In this regard, the algorithm removes the top unchecked physical node $v_s^i$ in *S_L* and its respective physical links from $G_s^T$. Nevertheless, there must be a single alternative path for every virtual link $l_n^{a_m,b_m}$ that was passing through the removed substrate node $v_s^i$. The algorithm finds $K$ loopless shortest paths from the source node of the virtual link to its sink node. Our preferred routing algorithm to find $K$ loopless shortest paths is Yen's algorithm [23], while the cost of every physical link is assumed to be 1. Note that the value of $K$ is adjustable, and the effects of different values of $K$ on the heuristic's result are discussed in Section 6. The alternative path must support the off-peak traffic demand $\hat{r}_n^m$ of the respective virtual link $l_n^{a_m,b_m}$. Therefore, a capable alternative path needs to satisfy some constraints.

First, the unused off-peak bandwidth capacity $\check{C}_b(l_s^{x,y})$ of every physical link on the alternative path must be equal to

or greater than the off-peak traffic demand $\acute{r}_n^m$ of the virtual link. Second, the unused off-peak bandwidth capacity $\check{C}_b(v_s^j)$ of every physical node on the alternative path also must support the off-peak traffic demand $\acute{r}_n^m$ of the virtual link. If the found path satisfies these constraints, then the algorithm updates $\check{C}_b(l_s^{x,y})$ and $\check{C}_b(v_s^j)$ of all the physical links and nodes on the path, respectively, and stops checking the remaining shortest paths. Moreover, it removes the previously allocated traffic capacities for the virtual link. However, if one or more physical nodes or links on the found path do not support the demanded traffic, the heuristic checks the next shortest path. In the case where there is no capable alternative path, the algorithm places the physical node and its respective physical links back to $G_s^T$, discards the modifications, and checks the removal possibility of the next substrate node in *S_L*.

After all of the physical nodes in *S_L* are checked, $G_s^T$ is returned as the energy-efficient off-peak substrate topology.

As previously mentioned, the BILP defined for off-peak node and link energy optimization by local link reconfiguration is $\mathcal{NP}$-hard. Therefore, it is not scalable in the case of large network scenarios due to the long execution time. Nevertheless, it is expected that the proposed heuristic for the same problem is considerably simpler and faster than the formulated BILP. The largest loop, which starts on line 9 and ends on line 44, determines the complexity of the proposed heuristic. In this regard, the main loop that starts on line 9 is run for each physical node, and thus, its complexity is $O(|V_s|)$. The first sub-loop that starts on line 11 is run for every virtual link belonging to all of the VNs, which passes through the physical node. Therefore, its complexity is $O(|\Phi||E_v^m|)$, where $E_v^m$ is the set of edges of the involved virtual network with the largest number of virtual links. The heuristic calls Yen's algorithm on line 12. The complexity of Yen's algorithm is $O(K|V_s|(|E_s|+|V_s|log|V_s|)$. The second sub-loop that starts on line 13 is run for $K$ found shortest paths, and consequently, its complexity is $O(K)$. The third sub-loop starting on line 14 checks the capability of every physical link on the found path. Therefore, its complexity is $O(|E_s|)$. The other sub-loop that runs alongside the previous one checks the eligibility of physical nodes on the found path, and therefore, its complexity is $O(|V_s|)$. The heuristic might need to check all the virtual links' capacities in every physical link, in the worst-case scenario, to discard the capacity and traffic modifications for each rerouted virtual link. Thus, the complexity of the undoing function on lines 18, 28 and 40 is $O(|E_s||\Phi||E_v^m|)$. Hence, the complexity of the proposed heuristic is $O(K|V_s||\Phi||E_v^m|(|V_s|^2log|V_s|+|E_s|^2|\Phi||E_v^m|+|V_s||E_s||\Phi||E_v^m|))$. Consequently, the proposed heuristic algorithm is considerably simpler than the BILP, and it could be solved in a polynomial time.

## 6. Evaluation

The proposed energy-saving solutions are supposed to reduce the total node and link energy consumption in VNEs during off-peak times. They need to guarantee the full connectivity, off-peak bandwidth requirements, and off-peak processing demands. Several random VNE setups have been assessed to study the impact of different parameters on the energy-saving capability of the discussed solutions.

Recently, the Waxman algorithm [24] has been widely used by researchers to generate random topologies for VNEs [12,13,25–27]. Therefore, in this paper, the substrate and virtual networks' topologies are generated using the Waxman algorithm. Waxman generates random network topologies based on two parameters, $\lambda$ and $\mu$. As $\lambda$ increases, the probability of having an edge between any nodes in the topology is increased. As $\mu$ increases, there is a larger ratio of long edges to short edges. In this paper, we choose the Waxman parameters for both the substrate and virtual networks' topologies as $\lambda = 0.5$ and $\mu = 0.5$ in an area size of $100 \times 100$. After creating random substrate and virtual networks' topologies, the substrate links' bandwidth capacity and virtual links' peak demand are generated randomly with a uniform distribution. The bandwidth capacity of each physical link is a random amount between 100 Mbps and 200 Mbps, but each virtual link's bandwidth demand is generated randomly between 50 Mbps and 100 Mbps. Therefore, according to [18], the base and maximum physical link power consumption for any physical link are 0.9 W and 1.00 W, respectively. The bandwidth capacity of every physical node is a constant amount of 1 Gbps. Moreover, the processing capacity of the physical and virtual nodes are generated randomly with a uniform distribution. The processing capacity of each substrate node is a random amount between 500 MHz and 800 MHz, but every virtual link's processing demand is generated randomly between 400 MHz and 500 MHz. The randomly generated substrate networks are symmetric; thus, if there is a physical link from node $i$ to node $j$ with a specific amount of bandwidth capacity, there is also a physical link from node $j$ to node $i$ with the same amount of bandwidth capacity. In the next step, each created virtual node is mapped onto a substrate node belonging to the set of substrate node candidates that have sufficient processing capacity for that virtual node with a uniform distribution. Subsequently, every generated virtual link's peak bandwidth demand is allocated in a substrate path through a state-of-the-art algorithm that does not consider energy efficiency.

As discussed, the formulated ILPs are $\mathcal{NP}$-hard, and thus, they are not scalable for large network sizes. Therefore, we assess the capability of the defined ILPs on small random simulation setups, similar to the other related works in [12–14,18]. The ILPs are solved using the CVX package [28], which uses the MOSEK solver [29]. Nonetheless, the theoretical complexity analysis reveals the proposed heuristic algorithm is considerably simpler, and therefore, it is scalable for large network sizes. Hence, the performance of the proposed heuristic is examined on medium random simulation setups.

Every small random simulation setup contains 10 randomly generated VNEs. Each VNE in a small random simulation setup has at least 2 random virtual networks that are allocated in a single random substrate network, while the substrate network has 15 nodes and each virtual network has 5 nodes. The average number of physical links in the small random simulation setups is 55. Furthermore, every medium random simulation setup includes 10 randomly generated VNEs. All the VNEs in a medium random simulation setup have at least 2 random virtual networks that are mapped onto a single random substrate network, while the substrate network has 50 physical nodes and each virtual network has

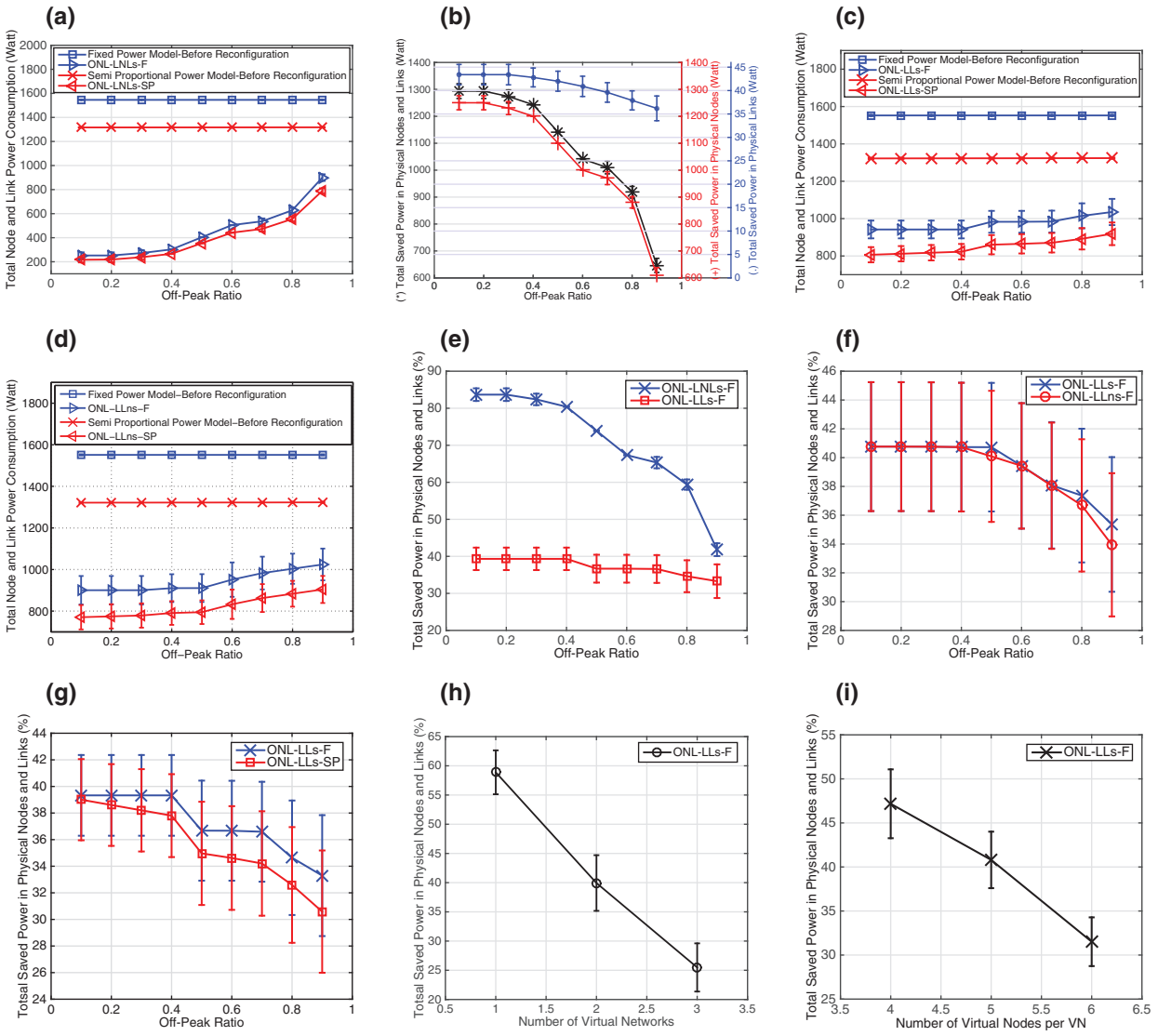


**Fig. 1.** (a) Total node and link power consumption based on off-peak ratio by MILPs for ONL-LNLs-F and ONL-LNLs-SP. (b) Total saved power in the network, total saved power in physical nodes, and total saved power in physical links based on off-peak ratio by MILP for ONL-LNLs-F. (c) Total node and link power consumption based on off-peak ratio by MILPs for ONL-LLs-F and ONL-LLs-SP. (d) Total node and link power consumption based on off-peak ratio by BILPs for ONL-LLns-F and ONL-LLns-SP. (e) Total saved power in physical nodes and links based on off-peak ratio by MILPs for ONL-LNLs-F and ONL-LLs-F. (f) Total saved power in physical nodes and links based on off-peak ratio by ILPs for ONL-LLs-F and ONL-LLns-F. (g) Total saved power in physical nodes and links based on off-peak ratio by MILPs for ONL-LLs-F and ONL-LLs-SP. (h) Total saved power in physical nodes and links based on the number of involved virtual networks by MILP for ONL-LLs-F. (i) Total saved power in physical nodes and links based on the number of virtual nodes per VN by MILP for ONL-LLs-F.

20 virtual nodes. The average number of physical links in the medium random simulation setups is 570. Note that the average results including confidence intervals with a confidence level of 90% are calculated for each setup.

### 6.1. ILPs

First, we study the power-saving capability of the MILPs formulated for ONL-LNLs-F and ONL-LNLs-SP on a small random simulation setup. We measured the network's total node and link power consumption for different off-peak ratios before and after applying the defined solutions. The off-peak ratio is the fraction of the network's off-peak traffic rate by its peak traffic rate. In the case where the off-peak ratio is $x$, the off-peak demands of every virtual network are $x$ times its peak demands. The measurement results for the off-peak ratio range of 0.1–0.9 are shown in Fig. 1a. Note that we measure power consumption rather than energy consumption because the amount of energy consumed depends on the total time we measured, and thus, it is not a good metric for comparison. The results reveal that the MILPs formulated for ONL-LNLs-F and ONL-LNLs-SP noticeably reduce the total node and link power consumption of the simulated virtualized networks. In addition, increasing the off-peak traffic ratio increases the amount of traffic that needs to be re-allocated; thus, the programs are more limited in terms of finding alternative paths for each removed virtual link. Consequently, the number of physical nodes and links in

sleep mode is decreased by increasing the off-peak traffic demand, and therefore, as Fig. 1a confirms, the programs could save lower amounts of power. Note that the total power consumption in the case of the *semi-proportional* power model is changed by varying the off-peak ratio, even before applying any energy-saving solution. However, because a larger portion of a node or link's power consumption in the *semi-proportional* power model is for keeping the device active ($\tilde{p}^b$), the variations in total power consumption due to the traffic fluctuations are small and could not be observed in the scale of Fig. 1a.

According to the power models defined in Section 3, physical nodes consume more energy than physical links. Thus, it is expected that the larger portion of energy that we could save in VNE is achieved by reducing the energy consumption of substrate nodes rather than substrate links. In this regard, we calculated the total amount of power saved in the network by the MILP formulated for ONL-LNLs-F. Moreover, we also calculated the total saved power in physical nodes and total saved power in physical links by the same MILP. Fig. 1b shows the results for these measurements. Fig. 1b confirms that the larger portion of total saved power is achieved by reducing the physical nodes' energy consumption rather than physical links. For instance, the formulated MILP for ONL-LNLs-F saved 1142.3 W in total when the off-peak traffic ratio was 0.5. Nonetheless, 1100.3 W was saved by reducing the physical nodes' power consumption, but only 42.0 W was saved by decreasing the power consumption in physical links.

Fig. 1c and d shows the network's total node and link power consumption before and after applying the off-peak node and link energy optimization by local link reconfiguration programs in the case of splittable traffic and non-splittable traffic, respectively. These measurements were performed for different off-peak ratios on a small random simulation setup. The programs decrease the VNE's node and link energy consumption by reconfiguring only some of the virtual links. The results prove that the formulated programs are able to save power effectively for either splittable traffic or non-splittable traffic based on both power models.

The off-peak node and link energy optimization by local node and link reconfiguration modifies the mapping for some of the virtual nodes and links. Nonetheless, the off-peak node and link energy optimization by local link reconfiguration adjusts the mapping for only some of the virtual links. Hence, it is expected that the formulated programs for the first strategy save more energy than the programs for the latter method. In this regard, we tested the MILPs formulated for ONL-LNLs-F and ONL-LLs-F on a small random simulation setup. Fig. 1e shows the percentage of power that the MILPs formulated for ONL-LNLs-F and ONL-LLs-F could save on the range of the off-peak ratio. The results confirm that the MILP formulated for ONL-LNLs-F could save larger amounts of power compared to the MILP formulated for ONL-LLs-F. Although the local link reconfiguration strategy does not reconfigure virtual nodes and therefore saves smaller amounts of power compared to the local node and link reconfiguration method, it causes fewer interruptions to the normal network operations. As shown in Fig. 1e, the energy-saving ability of ONL-LNLs-F is more sensitive to off-peak ratio fluctuations compared to the outcome of ONL-LLs-F. ONL-LNLs-F requires supporting both off-peak processing and traffic demands, but because ONL-LLs-F does not modify the mapping of virtual nodes, it only needs to handle off-peak traffic demands. Therefore, off-peak ratio variations have a greater impact on the energy-saving capability of ONL-LNLs-F than on the energy-saving capability of ONL-LLs-F.

It is expected that the programs save higher amounts of power when traffic is splittable because they are more flexible in terms of finding alternative paths. We checked this issue on the formulated ILPs for off-peak node and link energy optimization by local link reconfiguration. Fig. 1f shows the percentage of power that the ILPs formulated for ONL-LLs-F and ONL-LLns-F could save in the defined range of the off-peak ratio on a small random simulation setup. The results prove that the programs could save more power when traffic is splittable compared to when traffic is non-splittable. Note that ONL-LLs-F saves more power than ONL-LLns-F in high off-peak ratios. This is because when the off-peak ratio is low, it is easier for the programs to find the alternative paths, and therefore, ONL-LLs-F and ONL-LLns-F have almost the same outcome. However, when the off-peak ratio is high, it is more probable for the programs to find replaced paths by splitting high traffic loads into smaller loads.

Moreover, it is expected that higher levels of power will be saved when the *fixed* power model is applied than when the *semi-proportional* power model is employed, which is primarily due to two reasons. First, a physical node or link that is not fully utilized consumes a lower amount of power when it is designed based on the *semi-proportional* power model compared to when it is designed based on the *fixed* power model. Therefore, by setting the same physical nodes and links into sleep mode, the total amount of saved power is lower in the case of the *semi-proportional* power model. Second, if the energy-saving program sets a physical node into sleep mode, it re-allocates its virtual nodes to other substrate nodes and re-maps its respective virtual links to alternative paths. In the case of the *semi-proportional* power model, the re-allocated virtual node and the rerouted traffic increase power consumption over the alternative physical nodes and links. However, in the case of the *fixed* power model, because the traffic load does not affect the node or link power consumption, the reconfiguration does not increase the power consumption over the alternative physical nodes and links. We tested the MILPs formulated for ONL-LLs-F and ONL-LLs-SP on a small random simulation setup to verify this expectation. The results in Fig. 1g show the percentage of power saved by MILPs for ONL-LLs-F and ONL-LLs-SP. Fig. 1g confirms that the MILP for ONL-LLs-F could save higher amounts of power than the MILP for ONL-LLs-SP.

Furthermore, it is necessary to analyze the impact of different numbers of allocated virtual networks over the substrate network on the energy-saving ability of the proposed solutions. In this regard, we measured the percentage of power that the MILP formulated for ONL-LLs-F could save on a small random simulation setup when 1, 2 or 3 randomly generated virtual networks are allocated in the substrate network. The off-peak ratio is assumed to be 0.5. The results are presented in Fig. 1h. Fig. 1h reveals that increasing the number of allocated virtual networks in the substrate network decreases the total amount of power that the program could save. Allocating a new virtual network in the substrate

network imposes several constraints to the programs as new virtual nodes and links are mapped onto the physical nodes and links. For instance, if a program wants to set a physical node into sleep mode, it needs to re-allocate all the virtual nodes mapped onto the physical node, as well as every virtual link passing the physical node. By allocating a new virtual network, the program requires finding alternative physical nodes and paths for the newly mapped virtual nodes and links. Therefore, the program is more limited, and its energy-saving capability is decreased.

It is also important to study the effect of different numbers of virtual nodes per virtual network on the energy-saving ability of the proposed solutions. The number of virtual nodes per VN expresses the size of the virtual networks. Accordingly, we probed the percentage of power saved by the MILP formulated for ONL-LLs-F on a small random simulation setup while the number of virtual nodes per VN varied between 4 and 6. The off-peak ratio is assumed to be 0.5. The results are presented in Fig. 1i. Increasing the number of virtual nodes per VN increases the number of allocated virtual nodes in the physical nodes. Moreover, adding virtual nodes in a virtual network increases the number of virtual links in that virtual network. Consequently, increasing the number of virtual nodes per VN imposes several new constraints to the proposed energy-saving solutions and makes them more limited in terms of setting physical nodes and links into sleep mode. Hence, as confirmed in Fig. 1i for ONL-LLs-F, the program save lower amounts of power when the number of virtual nodes per VN is increased.

### 6.2. Heuristic

On the other hand, it is essential to study the effectiveness of the proposed heuristic algorithm for ONL-LLns-F. In this regard, we first tested the heuristic algorithm on a medium random simulation setup and measured network's total node and link power consumption before and after applying the heuristic algorithm on the defined range of off-peak ratio. As discussed in Section 5, $K$ affects the heuristic's outcome. $K$ is the number of shortest paths the algorithm examines to find a single capable alternative path for each re-mapped virtual link. The heuristic is simulated when $K$ is 1, 2, 3, 4 or 5. The results are presented in Fig. 2a. The results confirm that the proposed heuristic algorithm is able to effectively reduce the VNE's total node and link power consumption. In addition, Fig. 2a reveals that increasing the off-peak ratio decreases the amount of power that the heuristic could save. This is because increasing the off-peak ratio increases the traffic load that the algorithm needs to re-allocate, and therefore, the heuristic is more limited in terms of finding alternative paths.

In addition, Fig. 2a verifies that increasing $K$ increases the amount of power that the heuristic could save. If the heuristic needs to re-map a virtual link, it finds $K$ shortest paths from the source node to the sink node of the virtual link to provide a single qualified alternative path. However, the alternative path requires satisfying some capacity constraints. In this regard, the shortest path ($K = 1$) might not persuade the capacity constraints, but the second ($K = 2$) or third ($K = 3$) shortest path may. Consequently, the heuristic might save more power when $K$ is increased. Nevertheless, increasing $K$ increases the heuristic's run time because it needs to find a larger number of shortest paths. This is discussed theoretically in Section 5. Moreover, considering a specific off-peak ratio in Fig. 2a, the variations in the saved amount of power by the heuristic decreases when $K$ increases. This is because the number of available paths between two nodes in a substrate network is limited, and by increasing $K$, most of the possible paths are considered by the heuristic, and thus, it cannot find new replaced paths. Consequently, a large enough value of $K$, based on the substrate network size, makes it probable for the heuristic to save the maximum amount of power in a VNE.

It is vital to check the difference between the outcome of the BILP formulated for ONL-LLns-F as the optimum result and the proposed heuristic for the same problem. This is tested on a small random simulation setup, and results are presented in Fig. 2b. Fig. 2b shows the percentage of power that the BILP for ONL-LLns-F saves, as well as the percentage of power that the proposed heuristic for the same problem with different values of $K$ could save, on the defined range of off-peak ratio. Similar to the results of the previous simulation setup, increasing the off-peak ratio decreases the ability of the BILP and the heuristic in terms of saving power. Moreover, the heuristic works closely to the optimum points set by the BILP, whereas the heuristic is significantly faster than the BILP.

Additionally, we evaluated the impact of different numbers of mapped virtual networks over the substrate network on the heuristic's outcome when $K$ is 5 and the off-peak ratio is 0.5. The number of allocated VNs ranged from 1 to 3. The outcome is presented for a medium random simulation setup in Fig. 2c. Similar to the results shown in Fig. 1h, the results in Fig. 2c confirm that increasing the number of allocated virtual networks in the substrate network limits the heuristic in finding replaced paths and consequently decreases the amount of power it could save.

The influence of different numbers of virtual nodes per virtual network on the heuristic's outcome is also tested over a medium random simulation setup, and the results are presented in Fig. 2d. The off-peak ratio and $K$ are assumed to be 0.5, and 5, respectively. Similar to the results shown in Fig. 1i, the results in Fig. 2d verify that increasing the number of virtual nodes per mapped VN decreases the ability of the heuristic in terms finding alternative paths and therefore decreases the amount of power that the heuristic could save.

Moreover, it is explained in Section 5 that the proposed heuristic, similar to the BILP formulated for ONL-LLns-F, does not sleep the physical nodes with $\tilde{s_2}(v_s^i) \geq \mathcal{T}$ to decrease traffic disruptions due to reconfiguration. Fig. 2e shows the effect of changing the $\tilde{s_2}$ threshold on the capability of the heuristic when $K = 5$ over a medium random simulation setup. The off-peak ratio is assumed to be 0.5. Fig. 2e shows that decreasing the $\tilde{s_2}$ threshold amount decreases the amount of power that the heuristic saves because a smaller number of substrate nodes and links are considered for power saving. Although decreasing the $\tilde{s_2}$ threshold decreases the amount of power that the formulated solutions could save, it reduces traffic interruptions to the normal network operations due to reconfiguration.

In addition, rerouting the traffic to the other substrate paths causes changes to the link utilization. The heuristic algorithm re-maps the off-peak traffic onto the other paths to

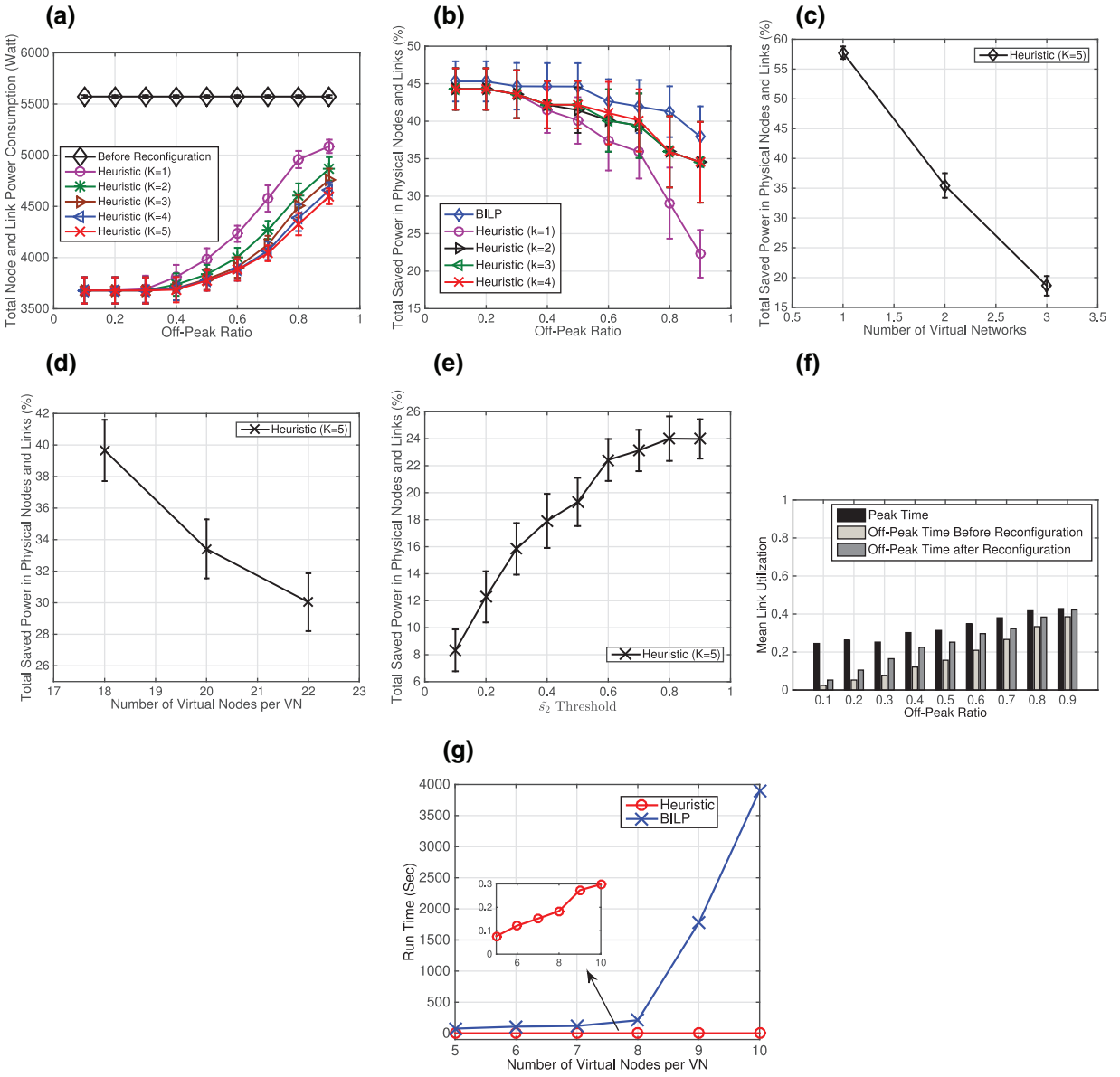


**Fig. 2.** (a) Total node and link power consumption based on off-peak ratio before and after applying the heuristic with different values of $K$ for ONL-LLns-F. (b) Total saved power in physical nodes and links based on off-peak ratio by the BILP and the heuristic with different values of $K$ for ONL-LLns-F. (c) Total saved power in physical nodes and links based on the number of involved virtual networks by the heuristic with $K = 5$ for ONL-LLns-F. (d) Total saved power in physical nodes and links based on the number of virtual nodes per VN by the heuristic with $K = 5$ for ONL-LLns-F. (e) The impact of different $\bar{s_2}$ thresholds on the total saved power in physical nodes and links by the heuristic when $K = 5$ for ONL-LLns-F. (f) Mean link utilization based on off-peak ratio over different configurations. (g) Required run time based on different number of virtual nodes per VN for the BILP and the heuristic with $K = 5$ for ONL-LLns-F.

save energy. Accordingly, it is necessary to ensure that the increased utilization is controlled and does not cause congestion. The link utilization for three different configurations is measured on a medium random simulation setup, and the average results are shown in Fig. 2f. The first configuration is for peak time when the allocated bandwidth is able to handle "worst-case" scenarios. The second configuration is for the off-peak period when no energy-saving algorithm is implemented. Over this period, the links are less utilized while the same bandwidth capacity is allocated and consume the same power as the peak time. After applying our suggested heuristic when $K = 5$ and the off-peak ratio is 0.5, the average link utilization is increased, but it is still less than the maximum utilization.

The BILP formulated for ONL-LLns-F is $\mathcal{NP}$-hard, while the complexity of the proposed heuristic algorithm for the same problem is $O(K|V_s||\Phi||E_v^m|(|V_s|^2 log|V_s| + |E_s|^2|\Phi||E_v^m| + |V_s||E_s||\Phi||E_v^m|))$. Therefore, it is expected that the proposed heuristic will require less run time compared to the formulated binary integer linear program for the same problem. We verified the run times for the BILP and the heuristic on a single random scenario. This scenario includes a random substrate topology with 15 physical nodes, while 2 random virtual networks are mapped onto the substrate

network, and the off-peak ratio is assumed to be 0.5. We probed the required run times for the BILP and the heuristic with $K = 5$ while the number of virtual nodes per virtual network changed from 5 to 10. The number of virtual nodes per VN expresses the size of the virtual networks. As the results in Fig. 2g show, the heuristic requires considerably less run time compared to the BILP. Additionally, Fig. 2g confirms that enlarging the size of mapped virtual networks increases the required run times for both the heuristic and the BILP. Note that the number of nodes per virtual network affects the number of virtual links per virtual network, and this impacts the time complexity of the heuristic. However, because the BILP is $\mathcal{NP}$-hard, its required run time changes almost exponentially by changing the number of virtual nodes per VN.

The simulation results prove that the proposed energy-saving solutions are able to effectively reduce a VNE's node and link energy consumption during the off-peak period. In addition, the proposed heuristic is a simple and fast algorithm that works closely to the optimum points.

Note that every simulation setup is quite large to cover a substantial number of random topologies to verify the effectiveness of the proposed solutions. Moreover, the calculated confidence intervals confirm that the results are precise enough to reveal the significances of the different energy-saving methods.

## 7. Conclusion

According to the latest published reports, ICT's energy consumption is rapidly increasing. Virtualized network environments have recently emerged in this technology. VNEs play a vital role toward virtualizing data centers. Therefore, it is essential to develop solutions that decrease energy consumption in VNEs. In this paper, we have formulated multiple novel ILPs that optimize node and link energy consumption in VNEs during the off-peak period by reconfiguring the mapping for some of the mapped virtual nodes and links. They are developed according to two power models and considering the impact of traffic splittability. Because the formulated ILPs are $\mathcal{NP}$-hard, we also proposed a novel energy-saving heuristic algorithm. The defined ILPs and the heuristic are tested over randomly generated VNEs. The simulation results show that the proposed solutions are notably effective and that the heuristic works closely to the optimum points. There are also some open areas for the future work. First, it is beneficial to consider the time space in the simulations and to use published real networks' traffic rates for different time periods to investigate the effectiveness of the solutions regarding saving energy over a period of time. Second, collocating multiple virtual nodes of a VN in a single substrate node has recently been suggested. Therefore, it is possible to extend our solutions to consider the collocation of virtual nodes to save more energy.

## References


[1] S. Ricciardi, D. Careglio, G. Santos-Boada, J. Sole-Pareta, U. Fiore, F. Palmieri, Saving energy in data center infrastructures, in: First International Conference on Data Compression, Communications and Processing (CCP), IEEE, 2011, pp. 265–270.

[2] J.S. Turner, D.E. Taylor, Diversifying the internet, in: Global Telecommunications Conference (GLOBECOM), 2, IEEE, 2005, pp. 6pp.–760.

[3] N. Chowdhury, R. Boutaba, A survey of network virtualization, Comput. Netw. 54 (5) (2010) 862–876.

[4] M.F. Zhani, Q. Zhang, G. Simon, R. Boutaba, Vdc planner: dynamic migration-aware virtual data center embedding for clouds, in: International Symposium on Integrated Network Management (IM), IFIP/IEEE, 2013, pp. 18–25.

[5] G. Rizzelli, A. Morea, M. Tornatore, O. Rival, Energy efficient traffic-aware design of on-off multi-layer translucent optical networks, Comput. Netw. 56 (10) (2012) 2443–2455.

[6] H. Feng, Y. Shu, Study on network traffic prediction techniques, in: International Conference on Wireless Communications, Networking and Mobile Computing, IEEE, 2005, pp. 1041–1044.

[7] A.S. San-Qi, A predictability analysis of network traffic, Comput. Netw. 39 (4) (2002) 329–345.

[8] F. Idzikowski, S. Orlowski, C. Raack, H. Woesner, A. Wolisz, Dynamic routing at different layers in IP-over-WDM networks-maximizing energy savings, Opt. Switch. Netw. 8 (3) (2011) 181–200.

[9] A. Fischer, M.T. Beck, H.D. Meer, An approach to energy-efficient virtual network embeddings, in: International Symposium on Integrated Network Management (IM), IFIP/IEEE, 2013, pp. 1142–1147.

[10] S. Su, Z. Zhang, X. Cheng, Y. Wang, Y. Luo, J. Wang, Energy-aware virtual network embedding through consolidation, in: Computer Communications Workshops (INFOCOM WKSHPS), IEEE, 2012, pp. 127–132.

[11] B. Wang, X. Chang, J. Liu, J.K. Muppala, Reducing power consumption in embedding virtual infrastructures, in: Globecom Workshops (GC Wkshps), IEEE, 2012, pp. 714–718.

[12] J.F. Botero, X. Hesselbach, M. Duelli, D. Schlosser, A. Fischer, H.D. Meer, Energy efficient virtual network embedding, Commun. Lett. 16 (5) (2012) 756–759.

[13] J.F. Botero, X. Hesselbach, Greener networking in a network virtualization environment, Comput. Netw. 57 (9) (2013) 2021–2039.

[14] E. Ghazisaeedi, N. Wang, R. Tafazolli, Link sleeping optimization for green virtual network infrastructures, in: Globecom Workshops (GC Wkshps), IEEE, 2012, pp. 842–846.

[15] D. Lo, L. Cheng, R. Govindaraju, L.A. Barroso, C. Kozyrakis, Towards energy proportionality for large-scale latency-critical workloads, in: Proceeding of the 41st Annual International Symposium on Computer Architecture, IEEE Press, 2014, pp. 301–312.

[16] X. Fan, W.-D. Weber, L.A. Barroso, Power provisioning for a warehouse-sized computer, in: ACM SIGARCH Computer Architecture News, 35, ACM, 2007, pp. 13–23.

[17] R. Tucker, J. Baliga, R. Ayre, K. Hinton, W. Sorin, Energy consumption in IP networks, in: ECOC Symposium on Green ICT, ITU, 2008, p. 1.

[18] A.P. Bianzino, C. Chaudet, F. Larroca, D. Rossi, J. Rougier, Energy-aware routing: a reality check, in: GLOBECOM Workshops (GC Wkshps), IEEE, 2010, pp. 1422–1427.

[19] J. Chabarek, J. Sommers, P. Barford, C. Estan, D. Tsiang, S. Wright, Power awareness in network design and routing, in: The 27th Conference on Computer Communications (INFOCOM), IEEE, 2008, pp. 457–465.

[20] D. Economou, S. Rivoire, C. Kozyrakis, P. Ranganathan, Full-system power analysis and modeling for server environments, in: Workshop on Modeling Benchmarking and Simulation (MOBS), 2006, pp. 13–23.

[21] J. Bisschop, AIMMS-Optimization Modeling, Lulu. com, 2006.

[22] S. Even, A. Itai, A. Shamir, On the complexity of time table and multi-commodity flow problems, in: 16th Annual Symposium on Foundations of Computer Science, IEEE, 1975, pp. 184–193.

[23] J.Y. Yen, Finding the k shortest loopless paths in a network, Manage. Sci. 17 (11) (1971) 712–716.

[24] B.M. Waxman, Routing of multipoint connections, Sel. Areas Commun. 6 (9) (1988) 1617–1622.

[25] A. Fischer, J.F.B. Vega, M. Duelli, D. Schlosser, X.H. Serra, H.D. Meer, Alevin—a framework to develop, compare, and analyze virtual network embedding algorithms, J. Electron. Commun. EASST (2011) 1–12 http://hdl.handle.net/2117/15170.

[26] J. Zhu, T. Wolf, Vnmbench: a benchmark for virtual network mapping algorithms, in: 21st International Conference on Computer Communications and Networks (ICCCN), IEEE, 2012, pp. 1–8.

[27] M.T. Beck, A. Fischer, H. de Meer, J.F. Botero, X. Hesselbach, A distributed, parallel, and generic virtual network embedding framework, in: International Conference on Communications (ICC), IEEE, 2013, pp. 3471–3475.

[28] M. Grant, S. Boyd, Cvx: Matlab software for disciplined convex programming, version 2.0 beta, September 2013, http://cvxr.com/cvx.

[29] E.D. Andersen, K.D. Andersen, The MOSEK interior point optimizer for linear programming: an implementation of the homogeneous algorithm, in: High Performance Optimization, Springer, pp. 197–232.

**Ebrahim Ghazisaeedi** received his M.Sc. degree in Mobile and Satellite Communications from the University of Surrey, England, in 2011. He is currently pursuing the Ph.D. degree in Electrical and Computer Engineering at the Department of Systems and Computer Engineering, Carleton University, Canada. His main research interests are in communication networks, network virtualization, and network optimization.

**Changcheng Huang** received his Ph.D. degree in Electrical Engineering from Carleton University, Canada, in 1997. Since July 2000, he has been with the Department of Systems and Computer Engineering at Carleton University, where he is currently a professor. His research interests are stochastic control in computer networks, resource optimization in wireless networks, reliability mechanisms for optical networks, network protocol design and implementation issues.