ESSO: An Energy Smart Service Function Chain Orchestrator

Md. Faizul Bari, *Member, IEEE*, Shihabur Rahman Chowdhury, *Student Member, IEEE*, and Raouf Boutaba, *Fellow, IEEE*

***Abstract*—The rapid development of technologies such as photo-intensive social networks, on-demand video streaming, online gaming, and the Internet of Things (IoT) is causing a tremendous growth of traffic volume. Such large-scale expansion is leading to higher energy consumption and carbon footprint for the telecommunication industry. Governments are trying to minimize the environmental impact by introducing regulations and taxes; driving companies to use renewable energy. However, renewable energy is still not as cost-effective compared to traditional sources of energy (*i.e.*, brown energy), and their availability varies significantly across time and geographic locations. Therefore, it is a challenge for telecommunication companies to comply with regulations and minimize carbon footprint without significantly increasing their operational cost. In this context, we propose an Energy Smart Service Function Chain Orchestrator called ESSO. ESSO reduces the overall carbon footprint of a telecommunication network by opportunistically adapting Service Function Chain (SFC) locations to utilize more energy at locations with surplus renewable energy. ESSO minimizes brown energy consumption by migrating SFCs across different locations. In addition, ESSO provisions SFC components in a manner that allows switches, switch ports, and servers to be put into low-power consumption state. Our trace-driven simulations on real ISP topologies show that considering the availability of renewable energy sources during SFC embedding even for a small-scale network can result in 2-3× reduction in carbon footprint.**

***Index Terms*—Energy-aware management, mathematical optimization, network function virtualization, orchestration, service function chaining.**

I. Introduction

THE RAPID proliferation of bandwidth intensive and latency sensitive applications such as on-demand video streaming, augmented and virtual reality, and Internet-of-Things is pushing the Information and Communication Technology (ICT) industry to expand continuously. For example, AT&T, one of the leading telecommunication service providers in the U.S., reportedly experienced 100,000 percent increase in traffic over a period of eight years (2008 to 2016) [1]. Such large-scale expansion is also increasing the energy requirement of ICT infrastructure and contributing to the global carbon emissions [2]. Statistics from 2013 show that telecommunication infrastructure alone accounted for 47% of the total energy consumed by the ICT sector [3]. When translated into carbon footprint, this sector produced 2.5% of global carbon footprint, which is projected to be double by 2020 [4]. Governments are trying to minimize the environmental impact of carbon emission by introducing regulations and taxes, driving companies to use renewable energy (*i.e.*, green energy) [5]. However, renewable energy is still not as cost-effective as traditional energy sources (*i.e.*, brown energy), and their availability varies significantly across time and geographic locations. Therefore, it is a challenge for telecommunication companies to comply with regulations and minimize carbon footprint [6] without significantly increasing their operational cost.

In order to address the challenge mentioned above, telecommunication network operators can employ several approaches such as reducing electricity consumption by optimizing resource usage, buying energy from green(er) energy vendors, and installing on-site renewable energy sources (*e.g.*, solar panels). For instance, Google achieved zero carbon emission for its data centers in 2017 by combining on-site renewable sources with energy bought from third-party renewable energy vendors [7]. In this paper, we particularly focus on reducing carbon emission of a telecommunication network by increasing resource efficiency through the following two techniques: (i) migration of network services between different geographic locations to maximize the utilization of on-site renewable energy sources, and (ii) intelligent topology aware placement and consolidation of network services to save energy by opportunistically switching off unused equipment. To this end, we propose an Energy-aware Smart Service function chain Orchestrator (ESSO) that employs these techniques to reduce the overall carbon footprint of a telecommunication network.

A major obstacle in implementing ESSO is the telecommunication operators' reliance on proprietary and vertically integrated hardware *middleboxes* for realizing different network services [8]. Telecommunication operators typically deploy middleboxes such as firewalls, load-balancers, WAN optimizers in Central Offices (COs) and Point-of-Presence (PoP) locations [1]. Depending on the type of traffic, network flows

Manuscript received April 22, 2019; revised August 12, 2019; accepted September 23, 2019. Date of publication September 27, 2019; date of current version December 10, 2019. This work was supported in part by the NSERC CREATE for Network Softwarization program and in part by an NSERC Discovery grant. This work also benefited from the use of CrySP RIPPLE facility at the University of Waterloo. The associate editor coordinating the review of this article and approving it for publication was G. Schembra. *(Corresponding author: Md. Faizul Bari.)*

The authors are with the David R. Cheriton School of Computer Science, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: faizul.bari@uwaterloo.ca; sr2chowdhury@uwaterloo.ca; rboutaba@uwaterloo.ca).

Digital Object Identifier 10.1109/TNSM.2019.2944170

at a CO are processed through different sequences of middleboxes or Service Function Chains (SFCs) to ensure fast, reliable, and secure access to the Internet [9]. Despite the telecommunication operators' extensive reliance on middleboxes, their closed and vertically-integrated nature pose a number of operational challenges [8]. Hardware middleboxes are attached to fixed locations in the network and offer little or no programmability and dynamicity [8]. As a result, it becomes a daunting task to dynamically migrate SFCs across COs to leverage on-site renewable energy or consolidate SFCs withing the same location to turn-off underutilized equipment.

Recently, telecommunication networks are going through a transformation known as *network softwarization* [10]. It provides the necessary flexibility and agility to dynamically control network elements and traffic flows that are required to build ESSO. Software Defined Networking (SDN), and Network Functions Virtualization (NFV) are two cornerstones of network softwarization [10]. SDN and NFV simplify network infrastructure by permitting the utilization of inexpensive commodity off-the-shelf (COTS) servers and networking hardware, transforming the COs and PoPs into mini-data centers; bringing economies-of-scale, agility, and flexibility of cloud computing to telecommunication networks. In this paper, we refer to COs, PoPs, and core data centers, which host the Virtual Network Functions, *i.e.*, the decoupled network functions from hardware middleboxes that is deployed on COTS servers, as NFV Point-of-Delivery or NFV-PoDs.

With SDN and NFV, VNFs are no longer restricted to fixed locations; they can be provisioned on any compute server within the network. Moreover, coupled with an SDN controller, VNFs and their associated traffic flows can be migrated within a short time-frame to a different compute server. The flexibility provided by SDN and NFV makes it possible to compose and reconfigure network services on the fly. ESSO builds on SDN and NFV and reduces the overall energy consumption and carbon footprint by (i) opportunistically utilizing more resources at locations with surplus renewable energy while minimizing consumption at locations where brown energy is the only option, and (ii) taking VNF placement and consolidation decisions in a manner that allows switches, switch ports, and servers to be put into low-power consumption state to reduce the overall power consumption. Specifically, this work consists of the following contributions:

- An ILP formulation for the SFC orchestration problem across multiple NFV-PoDs considering the availability of renewable energy.
- An architectural design for an SFC orchestrator that enables efficient resource and energy utilization for a geographically distributed infrastructure.
- A set of heuristic algorithms for taking SFC embedding, consolidation, and migration decisions across the NFV-PoDs of a telecommunication network.

The rest of the paper is organized as follows: Section II provides the required background to delineate the concepts applied in the paper. We discuss related work in Section III. The architecture of ESSO is presented in Section IV. Next, we provide a mathematical formulation for the network service orchestration problem in Section V. Section VI presents a set of heuristic algorithms for the placement, consolidation, and migration of network services based on fluctuations in the availability of renewable energy. After that, we present the results obtained from performance evaluation of the presented algorithms in Section VII, and finally, conclude with some future research directions in Section VIII.

II. Background and Motivation

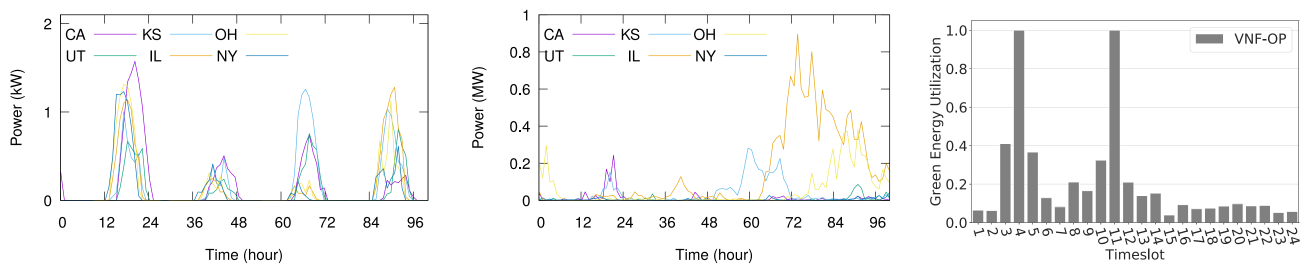
A. Background

1) Renewable Energy and Carbon Footprint: The amount of renewable energy generated from sources such as solar and wind significantly fluctuates across locations, time-of-day, weather conditions, and the technology used. Estimated energy generation potential for solar and wind in six U.S. states are shown in Fig. 1(a) and Fig. 1(b), respectively, according to the data collected by the U.S. Climate Reference Network and Regional Climate Reference Network [11]. As we can see, solar energy follows a diurnal pattern. In contrast, wind energy does not show any particular pattern. Moreover, based on the underlying technology, electricity grids can generate different amounts of carbon footprint per MWh. We incorporate this variation in our simulations by collecting data on different U.S. cities [12].

2) Energy Consumption States of Networking Equipment: A network switch consumes the major portion of power when it is turned on [13]. For example, the maximum wattage of the Dell PowerConnect 8024F switch is 160 W, and it consumes about 110 W without any traffic load [14]. When put into sleep mode, this switch consumes only 10 W power [15]. Ideally, consumed power should increase with the amount of traffic processed by the switch. In reality, switch power consumption depends on the settings of the switch ports, *i.e.*, the configured maximum link rate. For instance, a port in 1 Gbps and 10 Gbps settings consumes 1.2 W and 4.3 W, respectively, on the Dell PowerConnect 8024F switch [15]. For a given settings, power consumption of a switch port remains the same regardless of utilization. A number of works have focused on the energy efficiency of SDN switches [16], [17]. According to Vu *et al.* [17], for an OpenFlow switch, sleep mode for the switch ports and for the entire switch can save about 9.8% and 60% of total power, respectively.

B. Motivation

We perform a simulation to motivate the importance of energy-aware SFC orchestration. We run a simulation considering SFC request arrival and departure over time on AS-13129 topology from Rocketfuel dataset [18], and embed the SFCs using the ILP formulation proposed by VNF-OP [19]. VNF-OP addresses the optimal SFC embedding problem, however, does not take energy usage or the availability of renewable energy sources into account during embedding. We randomly chose three locations in the network (out of total seven locations) with renewable energy sources. We defer a full description of the simulation setting until Section VII for brevity. The utilization of renewable power over 24 hours is presented in Fig. 1(c). As we can see from the figure, the embedding was able to 100% utilize the available renewable



(a) Solar Energy (b) Wind Energy (c) Renewable Energy Unaware Embedding

Fig. 1. Renewable Energy and SFC Embedding.

energy only during two timeslots (timeslot 4 and 11). For the rest of the timeslots, the utilization remained less than 40%. This motivates the need for investigating an energy aware embedding scheme to better utilize much more renewable power. We will show in our evaluation that by maximizing the utilization of green energy over time, ESSO is able to reduce carbon footprint by 2-3× compared to VNF-OP.

III. Related Work

In recent years, the networking community has devoted significant attention to SFC orchestration [20], [21]. However, to the best of our knowledge, none of the previous works targeted the problem addressed here. Most existing SFC orchestration algorithms consider SFC placement without exploring SFC migration or consolidation of freed resources. The most related work to ours is the one by Eramo *et al.* [22] that addresses the problem of SFC provisioning and migration to reduce the operator's energy cost. However, the authors do not consider consolidation of freed resources. Moreover, they neither consider power consumption states of switches or switch-ports nor availability of renewable energy sources. Besides, they assume that each server will pre-activate a VNF of each type and then the proposed algorithm determines the number of cores for these pre-activated instances. However, this is not a suitable assumption as a VNF may have a preferred number of cores for its operations. For example, a VNF might only work with a single core, or it might not scale beyond a certain number of cores. Our work does not impose such restrictions and adopts a simplified model for VNFs that matches closely with the standard practices of cloud-based resource orchestrators.

A few recent works consider energy-efficient SFC orchestration. Authors in [23] propose a solution to jointly optimize energy utilization and resource allocation in a QoS-aware manner. They formulate the problem as an ILP and present a set of heuristic algorithms to solve realistic problem instances within an acceptable timescale. However, the proposed algorithms are offline. The authors do not consider the active/idle states of switches and switch-ports. Moreover, they do not consider geographically distributed data centers, and their solution is for a single administrative domain where the resource allocation status of the entire network is known to the orchestrator. In contrast, we propose an online embedding algorithm that considers a multi-domain network, geographically distributed infrastructure, active/idle states of switches and switch ports, and the availability of renewable energy.

The work in [24] considers a related but different problem. The authors assume that all required VNFs are already deployed on the servers. They address the problem of finding and scheduling traffic flows across the available VNF instances in an offline setting to reduce energy consumption. Authors in [25], [26] address the problem of SFC embedding with VNF sharing between tenants to optimize resource usage and maximize the operator's revenue. The authors provide an ILP solution to the problem that utilizes location constraints to reduce the running time of the ILP solution. The authors do not consider the availability of renewable energy or multiple administrative domains. In addition, they do not model the carbon-footprint of individual network equipment in active/idle states. The work in [27] address the problem of SFC orchestration across multiple domains; however, they assume that resource status within each domain is available to the global orchestrator. This assumption is not realistic for most telecommunication network operators. Furthermore, the authors only model servers and links; they do not model switches or switch ports for computing the total power consumption.

Green energy aware placement strategies are also explored in areas like Virtual Network (VN) and Virtual Data Center (VDC) embedding [28], [29]. The problem of VN embedding across multiple data centers to reduce the total energy consumption of switches and servers is considered in [30]. This work assumes that the workload of VNs can be predicted in the future and takes migration decisions based on the predicted values. It considers two time periods, workloads during the day and night while taking VN embedding decisions. However, this work does not consider the availability of renewable energy sources. In contrast, our algorithm is online and takes migration and consolidation decisions based on the fluctuation of renewable energy in real-time. Authors in [31], [32] propose VDC embedding algorithms across geographically distributed data centers while minimizing carbon footprint by utilizing green energy sources. However, the authors do not consider migration of virtual nodes to utilize green energy. They considered one-shot embedding of VDC requests across distributed data centers. Furthermore, they do not model the internal server and bandwidth of each data center separately. These works consider data centers with infinite capacity and only considered the bandwidth of the backbone network while taking embedding decisions. In

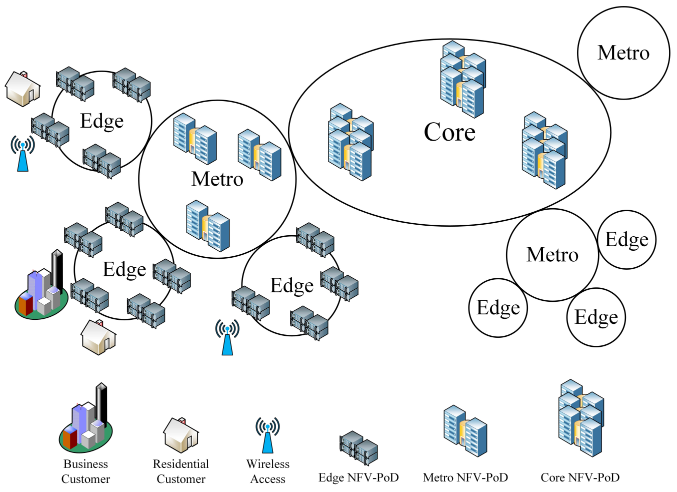


Fig. 2. Multi-Tiered NFV-PoD network.

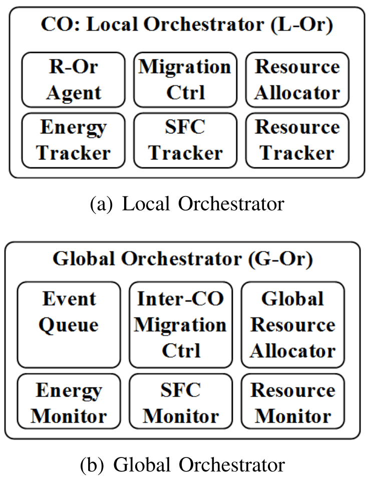


Fig. 3. Local and Global Orchestrators.

contrast, we consider both inter- and intra-NFV-PoD networks along with server and switch capacities inside each NFV-PoD while taking embedding, migration, and consolidation decisions.

IV. System Architecture

A. Assumptions

We make the following assumptions regarding the underlying infrastructure and energy usage policy. Without loss of generality, we assume a telecommunication network architecture similar to the one depicted in Fig. 2. NFV-PoDs, consisting of compute and networking resources, are geographically distributed and connected through edge, metro, and core transport networks. They serve residential, commercial, and mobile customers by providing last-mile and wireless connectivity. In this work, we assume that each NFV-PoD hosts a leaf-spine or fat-tree like network topology along with multiple servers or racks of servers for hosting VNFs [1].

The number of NFV-PoDs in a single city or metropolitan area depends on population density and can range from tens to hundreds [33]. Each NFV-PoD may or may not have on-site renewable energy sources and resorts to the traditional electricity grid when the amount of renewable energy is insufficient. Renewable energy sources are considered carbon free, while the energy drawn from the grid (brown energy) leaves a certain carbon footprint depending on the particular generation method used by local energy grid.

ESSO keeps track of the following metrics at each NFV-PoD: (i) amount of available compute and networking resources, (ii) lease duration of resources allocated to currently deployed SFCs, (iii) availability of renewable energy, and (iv) carbon footprint of brown energy. It also tracks the amount of available bandwidth on the inter-NFV-PoD backbone links. ESSO receives SFC requests and based on the aforementioned metrics, takes orchestration decisions with the objective to reduce the overall carbon footprint. Service chains are migrated based on the (i) fluctuation in availability of renewable energy and (ii) expiry of previously deployed service chains.

B. ESSO Architecture

Our proposed orchestration architecture has two components: a Global Orchestrator (G-Or) and one Local Orchestrator (L-Or) for each NFV-PoD. G-Or monitors the overall network infrastructure, receives service chaining requests, allocates resources across multiple NFV-PoDs to deploy service chains, and migrates service chains within or across NFV-PoDs. Each NFV-PoD has a Local Orchestrator (L-Or) that monitors and collects information about the intra-NFV-PoD network links, switches, and server resources and sends the data to the G-Or. The G-Or accumulates data received from the L-Ors and creates a global view of the entire infrastructure. The L-Ors collect data regarding the amount of renewable energy and track both renewable and brown energy usage, which are used to measure the total carbon emissions. Data provided by the L-Ors allow the G-Or to optimize resource allocation and migration decisions. According to the ETSI NFV MANO architecture, the L-Ors are simplified NFV Management and Orchestration (MANO) entities consisting of VNF managers and virtual infrastructure managers (VIMs) [34]. The G-Or is a full-fledged MANO entity that also includes ESSO, the NFV orchestrator. The VNF managers and VIM of the G-Or can interface with those of the L-Ors to collect information about the infrastructure and orchestrate network services accordingly. In the following, we describe the architecture of L-Or and G-Or in detail.

1) L-Or Architecture: An L-Or has six components (Fig. 3(a)): (i) Resource Tracker, (ii) SFC Tracker, (iii) Energy Tracker, (iv) Resource Allocator, (v) Migration Controller, and (iv) Resource Orchestrator (R-Or) Agent. As their names suggest, the first three components keep track of local resources (*e.g.*, intra-NFV-PoD servers, switches, and links), currently embedded SFCs, and energy usage, respectively. The resource allocator receives SFC embedding requests from the G-Or and determines the mapping of VNFs on physical servers and switches based on local resource availability. Then, it instantiates the required VMs or containers to deploy the VNFs and also creates the virtual links between them to instantiate the SFC. The migration controller is responsible for migrating a VNF along with its incident virtual links. The source and destination physical servers involved in the migration can reside either in the same NFV-PoD or different NFV-PoDs. The R-Or Agent facilitates the communication between an L-Or and the G-Or.

2) G-Or Architecture: The components of the G-Or are shown in Fig. 3(b). The Resource, SFC, and Energy Monitors collect usage data from the L-Ors. The Global Resource Allocator decides the placement of VNFs of an SFC upon receiving an SFC request. It considers the availability of renewable energy sources, the amount of free compute and networking resources at NFV-PoDs, and takes embedding decisions. We assume that each SFC request has an expiry time, after which the SFC is removed from the system. The Inter-NFV-PoD Migration Controller keeps track of SFC expiry events and changes in the availability of renewable energy at different NFV-PoDs. When an SFC expires and leaves the system, the freed resources can be used to consolidate existing SFCs for reducing power consumption. When a renewable energy source starts generating electricity, *e.g.*, during morning hours when the sun rises, the migration controller makes migration decisions to take advantage of surplus renewable energy. The migration controller decides a VNF migration only when the migration reduces the total carbon footprint of the SFC.

3) SFC Events and Interaction Between Components: The following two illustrative events describe how interactions are triggered and carried out between different ESSO components for managing power consumption and infrastructure resources.

a) New SFC embedding request: The G-Or event queue tracks new SFC requests. The global resource allocator listens for new SFC requests and initiates the embedding process. First, it accesses current network status from the resource monitor. Then, it selects a set of paths through the NFV-PODs as potential candidates for embedding. After that, the global resource allocator forwards the SFC request metadata to the selected L-Ors. Next, the L-Ors receive the data and invoke the local resource allocator to compute embedding cost. Finally, the L-Ors return the costs to the global resource allocator that takes the final embedding decision.

b) Migrating an existing SFC: Here, we will discuss the interaction for SFC migration. The energy tracker monitors for fluctuations in renewable energy and informs the G-Or's energy monitor. Next, the inter-CO migration controller periodically collects energy and resource data and triggers an SFC migration through the local migration controllers at the participating NFV-PoDs. Finally, the local migration controllers perform the migration, and update resource usage through the local resource tracker.

V. Problem Formulation

We first formally define the "Multi-Location SFC Orchestration" or *MLSO* problem and then construct an Integer Linear Program (ILP) formulation for it.

A. Multi-Location SFC Orchestration (MLSO) Problem

In the MLSO problem, the resource capacities of compute and networking resources, carbon footprint per unit of brown energy, availability of renewable energy for each NFV-PoD, and the link bandwidth of the inter-NFV-PoD backbone network are provided as input. A stream of SFC requests is also provided for which the ILP formulation will determine the optimal embedding. The amount of available renewable energy varies over time. The carbon footprint of brown energy is different at different locations. A solution to the MLSO problem must utilize these facts to reduce the overall carbon footprint of the telecommunication network.

Each SFC request consists of an ingress NFV-PoD, an egress NFV-PoD, and a sequence of Network Function (NF) types and flavors that the traffic needs to pass through. Types distinguish NFs into categories like firewall, NAT, IPS, proxy, *etc.* and flavors represent VNF variations in terms of resource requirements. For example, a firewall can have two flavors: (i) one uses 2 cpu cores and can process traffic at a maximum rate of 500Mbps and (ii) the other one uses 4 cpu cores and can process at a maximum rate of 900Mbps.

We assume that embedding, migration, or consolidation decisions are made at particular time-instances. These time-instances are identified by the occurrence of one or more of the following events: (i) arrival of a new SFC request, (ii) departure of one or more previously embedded SFC request(s), and (iii) changes in the amount of available renewable energy. When a new SFC request arrives, it needs to be embedded. After the departure of one or more previously embedded SFC request(s), resources at the corresponding NFV-PoDs can be consolidated. If the availability of renewable energy changes in a subsequent time-instance then already embedded VNFs can be migrated to a different NFV-PoD. A VNF is migrated only if the cost of migration is less than the savings in carbon footprint. The ILP makes embedding, migration, and/or consolidation decisions based on these events. The objective of the ILP is to minimize carbon footprint by determining the initial placement of new SFC requests, subsequent consolidation and migration of existing VNFs based on the availability of resources and renewable energy, while guaranteeing that the end-to-end latency of any SFC does not violate the maximum allowed delay constraint.

The MLSO problem is $\mathcal{NP}$-Hard. At each time-instance of the MLSO problem, we need to solve the general SFC orchestration problem [21] for the newly arriving SFCs, which is known to be $\mathcal{NP}$-Hard [19]. The general SFC orchestration problem can be reduced to the MLSO problem by considering one time-instance and assuming all SFC requests as new requests that need to be embedded. In this setting, a solution to the MLSO problem will also be a solution to the general SFC orchestration problem, and hence the MLSO problem is $\mathcal{NP}$-Hard. In fact, the MLSO problem is computationally much harder than the general SFC orchestration problem. Here, we need to keep track of active/sleep mode for each switch port, physical link, and server. In addition, we must track the amount of brown and renewable energy used at each NFV-POD to compute the final carbon footprint.

B. SFC Representation

An SFC request is composed of one of more VNFs belonging to various types, *e.g.*, `Firewall` $\rightarrow$ `IPS` $\rightarrow$ `Proxy`.

The nodes at the beginning and at the end represent the ingress and egress NFV-PoDs, respectively. For each VNF type $p \in P$, we have a set of flavors F_p, and the function $\phi(f)$ (for flavor $f \in F_p$) returns the type (p) of a VNF. The resource requirements of a VNF flavor *f*, is represented by $\varphi_f^r \in \mathbb{R}^+, \forall\, r \in R$, where *R* represents resource types, *e.g.*, CPU, memory, and disk. We also assume that a VNF of flavor f introduces a traffic processing delay of δ_f.

Let, $\mathcal{I}_n^t$ and $\mathcal{I}_p^t$ represent the set of incoming and pre-existing SFC requests in the system during timeslot $t \in \mathcal{T}$, respectively. Here, the set $\mathcal{T}$ represents all possible timeslots considered for the ILP. Each SFC request has an associated lifetime, after which it leaves the system and is removed from $\mathcal{I}_p^t$. Note that our model only needs to know the arrival of new SFCs and the departure of existing SFCs during a timeslot and SFC lifetime provides a convenient way to represent SFC departure events. The effectiveness of our solution does not depend on accurately knowing the lifetime of each SFC request in advance. At each timeslot, the ILP needs to take migration and/or consolidation decisions for the SFCs in $\mathcal{I}_p^t$ and at the same time determine the embedding of the SFCs in $\mathcal{I}_n^t$. We also define the set $\mathcal{I}^t = \mathcal{I}_n^t \cup \mathcal{I}_p^t$ to represent both new and pre-existing SFC requests at time *t*.

We define the following binary variables to establish the active duration, arrival, and departure events for an SFC:

$$\ddot{\nu}_i^t = \begin{cases} 1 & \text{if SFC } i \in \mathcal{I} \text{ arrives at timeslot } t \in \mathcal{T}, \\ 0 & \text{otherwise.} \end{cases}$$

$$\ddot{a}_i^t = \begin{cases} 1 & \text{if SFC } i \in \mathcal{I} \text{ is active at timeslot } t \in \mathcal{T}, \\ 0 & \text{otherwise.} \end{cases}$$

$$\ddot{d}_i^t = \begin{cases} 1 & \text{if SFC } i \in \mathcal{I} \text{ departs at timeslot } t \in \mathcal{T}, \\ 0 & \text{otherwise.} \end{cases}$$

An SFC request, $i \in \mathcal{I}^t$, is represented as a directed acyclic graph, $G_i = (N_i, L_i)$, where N_i is the set of nodes and L_i is the set of links. There are two types of nodes, namely placeholder nodes for the ingress and egress NFV-PoDs of the chain and VNF nodes. The embedding of the placeholder nodes are predetermined as they need to be embedded at particular NFV-PoDs. The embedding of VNF nodes needs to be determined. Each VNF node $n \in N_i$, is of a particular NF flavor denoted by the function *f*(*n*). The flavor of an NF determines its resource requirements. Each link $l \in L_i$ has a bandwidth requirement β_l. The maximum tolerable delay for the entire chain is δ_i. The functions *s*(.) and *d*(.) denote the source and destination nodes of each link $l \in L_i$, respectively.

C. Physical Infrastructure Representation

The physical infrastructure is represented by a graph $\bar{G} = (\bar{N} \cup \bar{S}, \bar{L})$, where $\bar{N}$ and $\bar{S}$ represent the servers and switches of all NFV-PoDs, respectively. $\bar{L}$ represents the inter- and intra-NFV-PoD links of the telecommunication network. The set $\mathcal{C}$, represents the NFV-PoDs in the network and an individual NFV-PoD is denoted by *c*. We define e_r^{tc} as the amount of available renewable energy at NFV-PoD *c* during timeslot *t*. The amount of carbon per watt of brown energy generated by the local grid is represented by ζ_c and the Power Usage Effectiveness (PUE) of a NFV-PoD is represented by μ_c.

TABLE I
GLOSSARY OF SYMBOLS

Service Function Chain (SFC)	
F_p	Set of VNF flavors of type $p \in P$
$\phi(f) \in P$	Returns the type p of a VNF of flavor f
R	Types of resources (*e.g.*, CPU, Memory, disk)
$\varphi_f^r \in \mathbb{R}^+, \forall r \in R$	Resource requirement r of VNF flavor f
δ_f	Processing delay for VNF flavor f
$\mathcal{I}_n^t$	New SFC requests during timeslot $t \in \mathcal{T}$
$\mathcal{I}_p^t$	Existing SFC requests during timeslot $t \in \mathcal{T}$
$\ddot{\nu}_i^t \in \{0,1\}$	$\ddot{\nu}_i^t = 1$ if SFC i arrives at timeslot $t \in \mathcal{T}$
$\ddot{a}_i^t \in \{0,1\}$	$\ddot{a}_i^t = 1$ if SFC i is active at timeslot $t \in \mathcal{T}$
$\ddot{d}_i^t \in \{0,1\}$	$\ddot{d}_i^t = 1$ if SFC i departs at timeslot $t \in \mathcal{T}$
$G_i = (N_i, L_i)$	A DAG representing an SFC.
δ_i	Maximum tolerable delay for SFC i
β_l	Bandwidth requirement of link $l \in L_i$
ϑ	Migration threshold
Physical Infrastructure	
$\bar{G} = (\bar{N} \cup \bar{S}, \bar{L})$	Physical network graph.
$\mathcal{C}$	Set of NFV-PoDs
e_r^{tc}	Renewable energy at $c \in \mathcal{C}$ during $t \in \mathcal{T}$
ζ_c	Carbon per unit of brown energy at $c \in \mathcal{C}$
μ_c	Power Usage Effectiveness (PUE) of $c \in \mathcal{C}$
$\kappa_{\bar{n}}^r \in \mathbb{R}^+, \forall\, r \in R$	Resource capacity of server $\bar{n} \in \bar{N}$
$m(\bar{\pi})$, $e_{m(\bar{\pi})}$	Operational mode and power usage of a switch port $\bar{\pi} \in \bar{\Pi}_{\bar{s}}$
$\beta_{\bar{l}} \in \mathbb{R}^+$, $\delta_{\bar{l}} \in \mathbb{R}^+$	Bandwidth and latency of physical link $\bar{l} \in \bar{L}$
ξ^β	Power consumed per unit bandwidth
$e_{\bar{n}}^s$, $e_{\bar{n}}^b$, $e_{\bar{n}}^m$	Server power usage: sleep, base, max. load
$e_{\bar{s}}^s$, $e_{\bar{s}}^b$	Switch power usage: sleep and active modes
$\ddot{a}_{\bar{n}}^t \in \{0,1\}$	1 if server $\bar{n}$ is active at timeslot $t \in \mathcal{T}$
$\ddot{a}_{\bar{s}}^t \in \{0,1\}$	1 if switch $\bar{s}$ is active at timeslot $t \in \mathcal{T}$
$\ddot{a}_{\bar{\pi}\bar{s}}^t \in \{0,1\}$	1 if port $\bar{\pi}$ of switch $\bar{s}$ is active at $t \in \mathcal{T}$
$\psi_{\bar{l}\bar{p}} \in \{0,1\}$	1 if link $\bar{l} \in \bar{L}$ belongs to path $\bar{p} \in \bar{P}$
$w_{\bar{\pi}\bar{s}\bar{p}} \in \{0,1\}$	1 if port $\bar{\pi}$ of switch $\bar{s}$ is on path $\bar{p} \in \bar{P}$
$\bar{N}^c$, $\bar{S}^c$, $\bar{L}^c$, and $\bar{P}^c$	Set of servers, switches, physical links, and paths belonging to NFV-PoD $c \in \mathcal{C}$
$\nu(.)$	NFV-POD of a server, switch, link, and path
$\mathcal{B}(\bar{p})$	Inter-NFV-POD links on path $\bar{p} \in \bar{P}$
$d_{\bar{n}p} \in \{0,1\}$	$d_{\bar{n}p} = 1$, if VNF of type $p \in P$ can be provisioned on server $\bar{n}$
$z_n^{ci} \in \{0,1\}$	$z_n^{ci} = 1$, if VNF $n \in N_i$ can be provisioned in NFV-PoD $c \in \mathcal{C}$
Decision Variables	
$\hat{x}_{n\bar{n}}^{ti} \in \{0,1\}$	$\hat{x}_{n\bar{n}}^{ti} = 1$, if VNF $n \in N_i$ is deployed on server $\bar{n} \in \bar{N}$ during timeslot $t \in \mathcal{T}$
$\hat{y}_{l\bar{p}}^{ti} \in \{0,1\}$	$\hat{y}_{l\bar{p}}^{ti} = 1$, if SFC link $l \in L_i$ is deployed on physical path $\bar{p} \in \bar{P}$ during timeslot $t \in \mathcal{T}$

R denotes the set of resources (*e.g.*, CPU, memory) offered by a server. The resource capacity of server $\bar{n} \in \bar{N}$ is denoted by $\kappa_{\bar{n}}^r \in \mathbb{R}^+, \forall\ r \in R$. The ports of a switch $\bar{s} \in \bar{S}$ is represented by the set $\bar{\Pi}_{\bar{s}}$ and individual ports are represented by $\bar{\pi} \in \bar{\Pi}_{\bar{s}}$. The operational mode of a port is denoted by the function $m(\bar{\pi})$. $e_{m(\bar{\pi})}$ denotes a port's energy consumption in a particular operational mode. Bandwidth and propagation delay of a physical link $\bar{l} \in \bar{L}$ is represented by $\beta_{\bar{l}} \in \mathbb{R}^+$ and $\delta_{\bar{l}} \in \mathbb{R}^+$, respectively. Energy consumed per unit bandwidth is represented by ξ^β.

We assume a linear model for server power consumption [35]. So, $e_{\bar{n}}^b$ and $e_{\bar{n}}^m$ represent the base and the maximum power consumption of a server, respectively. The base power is consumed when the server is active but there is no workload. Next, $e_{\bar{n}}^s$ represents the power consumption when the server is in sleep mode. We define the following binary variable to denote if a server is active or not.

$$\ddot{a}^t_{\bar{n}} = \begin{cases} 1 & \text{if } \bar{n} \in \bar{N} \text{ is active between } [t, t+1), \\ 0 & \text{otherwise.} \end{cases}$$

Next, we define an additional binary variable to denote whether a switch port is active or not.

$$\ddot{a}^t_{\bar{\pi}\bar{s}} = \begin{cases} 1 & \text{if port } \bar{\pi} \text{ is active between } [t, t+1), \\ 0 & \text{otherwise.} \end{cases}$$

A switch is considered to be in the active mode if at least one of its ports is active, otherwise it is in the sleep mode and consumes substantially less energy. The base energy of a switch $\bar{s}$ in active mode is represented by $e^b_{\bar{s}}$ and the energy consumption of a switch in sleep mode is denoted by $e^s_{\bar{s}}$. The total energy consumption of an active switch depends on the number of active ports and their operational modes, as discussed in Section II-A2. The following binary variable denotes whether a switch is in active mode or in sleep mode:

$$\ddot{a}^t_{\bar{s}} = \begin{cases} 1 & \text{if switch } \bar{s} \text{ is active between } [t, t+1), \\ 0 & \text{otherwise.} \end{cases}$$

All possible paths between the servers and switches in $\bar{N} \cup \bar{S}$ are represented by $\bar{P}$, and the following variable denotes whether a physical link $\bar{l} \in \bar{L}$ belongs to a path:

$$\psi_{\bar{l}\bar{p}} = \begin{cases} 1 & \text{if link } \bar{l} \in \bar{L} \text{ belongs to path } \bar{p} \in \bar{P}, \\ 0 & \text{otherwise.} \end{cases}$$

The following binary variable expresses the connectivity between physical paths and switch ports:

$$w_{\bar{\pi}\bar{s}\bar{p}} = \begin{cases} 1 & \text{if port } \bar{\pi} \text{ of switch } \bar{s} \text{ is on physical path } \bar{p} \in \bar{P}, \\ 0 & \text{otherwise.} \end{cases}$$

Next, we define $\bar{N}^c$, $\bar{S}^c$, $\bar{L}^c$, and $\bar{P}^c$ as the set of servers, switches, physical links, and physical paths belonging to NFV-PoD, $c \in \mathcal{C}$, respectively. The functions $\bar{s}(.)$ and $\bar{d}(.)$ denote the source node and the destination node, respectively, of both a physical link and a physical path. The function $\nu(.)$ maps a server or link to its corresponding NFV-PoD. We also define the function $\mathcal{B}(\bar{P})$ to denote the backbone links between NFV-PoDs on the physical paths. Next, $\beta_{\bar{p}}$ denotes the available bandwidth on path $\bar{p}$ and is equal to the minimum bandwidth over all physical links in a path: $\beta_{\bar{p}} = \min_{\bar{l} \in \bar{p}} \beta_{\bar{l}}$.

There can be certain hardware requirements (*e.g.*, hardware-accelerated encryption for Deep Packet Inspection (DPI)) that may prevent a server from running a particular NF type. Furthermore, the network operator may have preferences regarding provisioning a particular type of VNF on a particular set of servers, *e.g.*, Firewalls should be deployed close to the network edge. So, we assume that for each VNF type there is a set of servers on which it can be provisioned. The following binary variable represents this relationship:

$$d_{\bar{n}p} = \begin{cases} 1 & \text{if VNF of type } p \in P \text{ can be provisioned on } \bar{n}, \\ 0 & \text{otherwise.} \end{cases}$$

Moreover, each VNF in a chain can have location restrictions expressed by the following binary variable:

$$z^{ci}_n = \begin{cases} 1 & \text{if VNF } n \in N_i \text{ can be provisioned in NFV}-\text{PoD } c, \\ 0 & \text{otherwise.} \end{cases}$$

D. Decision Variables

Now, we define the following decision variables to denote the placement of a VNF and the routing path of a chain on the physical infrastructure:

$$\hat{x}^{ti}_{n\bar{n}} = \begin{cases} 1 & \text{if VNF } n \in N_i \text{ deployed on } \bar{n} \in \bar{N} \text{ in } [t, t+1), \\ 0 & \text{otherwise.} \end{cases}$$

$$\hat{y}^{ti}_{l\bar{p}} = \begin{cases} 1 & \text{if link } l \in L_i \text{ deployed on } \bar{p} \in \bar{P} \text{ in } [t, t+1), \\ 0 & \text{otherwise.} \end{cases}$$

E. ILP Formulation

The above mentioned variables must satisfy the following constraints of the optimization problem:

$$\sum_{\bar{n} \in \bar{N}} \hat{x}^{ti}_{n\bar{n}} = \ddot{a}^t_i, \ \forall\ t \in \mathcal{T}, i \in \mathcal{I}^t, n \in N_i \tag{1}$$

$$\sum_{\bar{p} \in \bar{P}} \hat{y}^{ti}_{l\bar{p}} = \ddot{a}^t_i, \ \forall\ t \in \mathcal{T}, i \in \mathcal{I}^t, l \in L_i \tag{2}$$

$$\hat{x}^{ti}_{n\bar{n}} \le d_{\bar{n}\phi(f(n))}, \ \forall\ t \in \mathcal{T}, i \in \mathcal{I}^t, n \in N_i, \bar{n} \in \bar{N} \tag{3}$$

$$\hat{x}^{ti}_{n\bar{n}} \le z^{\nu(\bar{n})i}_n, \ \forall\ t \in \mathcal{T}, i \in \mathcal{I}^t, n \in N_i, \bar{n} \in \bar{N} \tag{4}$$

$$\hat{y}^{ti}_{l\bar{p}} \le \hat{x}^{ti}_{s(l)\bar{s}(\bar{p})}, \ \forall\ t \in \mathcal{T}, i \in \mathcal{I}^t, s(l) \in N_i, \bar{p} \in \bar{P} \tag{5}$$

$$\hat{y}^{ti}_{l\bar{p}} \le \hat{x}^{ti}_{d(l)\bar{d}(\bar{p})}, \ \forall\ t \in \mathcal{T}, i \in \mathcal{I}^t, d(l) \in N_i, \bar{p} \in \bar{P} \tag{6}$$

$$\sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{I}^t} \sum_{n \in N_i} \hat{x}^{ti}_{n\bar{n}} \times \varphi^r_{f(n)} \le \kappa^r_{\bar{n}}, \ \forall\ \bar{n} \in \bar{N}, r \in R \tag{7}$$

$$\sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{I}^t} \sum_{n \in N_i} \hat{y}^{ti}_{l\bar{p}} \times \beta_l \le \beta_{\bar{p}}, \ \forall\ \bar{p} \in \bar{P} \tag{8}$$

$$\sum_{l \in L_i} \hat{y}^{ti}_{l\bar{p}} \times \delta_{\bar{p}} + \sum_{n \in N_i} \delta_{f(n)} \le \delta_i, \ \forall\ t \in \mathcal{T}, i \in \mathcal{I}^t \tag{9}$$

$$\ddot{a}^t_{\bar{n}} \le \sum_{i \in \mathcal{I}^t} \sum_{n \in N_i} \hat{x}^{ti}_{n\bar{n}}, \ \forall\ t \in \mathcal{T}, \bar{n} \in \bar{N} \tag{10}$$

$$\ddot{a}^t_{\bar{n}} \ge \frac{\sum_{i \in \mathcal{I}^t} \sum_{n \in N_i} \hat{x}^{ti}_{n\bar{n}}}{\sum_{i \in \mathcal{I}} |N_i|}, \ \forall\ t \in \mathcal{T}, \bar{n} \in \bar{N} \tag{11}$$

$$\ddot{a}^t_{\bar{s}} \le \sum_{\bar{\pi} \in \bar{\Pi}} \ddot{a}^t_{\bar{\pi}\bar{s}}, \ \forall\ t \in \mathcal{T}, \bar{s} \in \bar{S} \tag{12}$$

$$\ddot{a}^t_{\bar{s}} \ge \frac{\sum_{\bar{\pi} \in \bar{\Pi}} \ddot{a}^t_{\bar{\pi}\bar{s}}}{|\bar{\Pi}_{\bar{s}}|}, \ \forall\ t \in \mathcal{T}, \bar{s} \in \bar{S} \tag{13}$$

$$\ddot{a}^t_{\bar{\pi}\bar{s}} \le \sum_{i \in \mathcal{I}^t} \sum_{l \in L_i} \sum_{\bar{p} \in \bar{P}} w_{\bar{\pi}\bar{s}\bar{p}} \times \hat{y}^{ti}_{l\bar{p}}, \quad \forall\ t \in \mathcal{T}, \bar{s} \in \bar{S}, \bar{\pi} \in \bar{\Pi}_{\bar{s}} \tag{14}$$

$$\ddot{a}^t_{\bar{\pi}\bar{s}} \ge \frac{\sum_{i \in \mathcal{I}^t} \sum_{l \in L_i} \sum_{\bar{p} \in \bar{P}} w_{\bar{\pi}\bar{s}\bar{p}} \times \hat{y}^{ti}_{l\bar{p}}}{\sum_{i \in \mathcal{I}^t} \sum_{l \in L_i} \sum_{\bar{p} \in \bar{P}} \hat{y}^{ti}_{l\bar{p}}}, \quad \forall\ t \in \mathcal{T}, \bar{s} \in \bar{S}, \bar{\pi} \in \bar{\Pi}_{\bar{s}} \tag{15}$$

Constraints (1) and (2) ensure that all VNFs and virtual links for the active SFCs are embedded. Next, (3) and (4) ensure that VNFs are embedded on servers and NFV-PoDs without violating the placement constraints for the VNF type and NFV-PoD. Then constraint (5) and (6) make sure that the endpoints of virtual link embedding on physical path and VNF embedding on physical server match with each other. Next, constraint (7) and (8) denote the capacity constraints for the servers and links, respectively. Constraint (9) represents the end-to-end latency constraint for the SFC. Constraint (10) ensures that a server is not in the active state when any VNF is not embedded on it, and constraint (11) ensures that a server is activated when there is at least one VNF embedded on it. Similarly, (12) and (13) make sure that a switch is in the active state when any active path passes through it, otherwise it will be in sleep mode. Finally, (14), and (15) ensure that switch ports are activated only when there is a virtual link embedded on a physical path that passes through that port.

F. Objective Function

Here, we calculate the total carbon footprint of the network. First, we define the function d_t to denote the time-duration of timeslot *t*. Now, for any time-duration d_t and a NFV-PoD $c \in \mathcal{C}$, the total energy consumption is calculated as follows:

1) SFC Migration: We consider SFC migration to all possible destination locations. At the beginning of each timeslot, the optimization algorithm (optimal or heuristic) determines the possible reduction in carbon footprint that can be achieved by migrating an active SFC. An SFC is then migrated only when the new embedding reduces the cost by a certain proportion, specified by the migration threshold ϑ. If there are multiple migration options, then the minimal cost migration is performed. The migration threshold (ϑ) provides a tuning-knob to the network operator to control migration events. Migration overhead primarily depends on the deployment technology of an SFC. For instance, the migration cost will be different across SFCs that are realized with VMs or containers or mechanisms such as stateless-NFs [36], [37]. Therefore, we provide a control-knob for migration to handle a broad range of deployment scenarios. The network operator can adapt the value of ϑ to the underlying network infrastructure to achieve a better trade-off between the costs and benefits of migration.

2) Energy Consumption of an SFC: The energy consumption for active SFCs constitutes three components: energy consumption by (i) physical servers, (ii) physical switches and ports, (iii) inter-NFV-PoD or backbone links. We measure the total energy consumption for the active SFCs as follows:

$$
\begin{aligned}
E_a^{tc} = \ddot{a}_i^t \times \sum_{i \in \mathcal{I}_n^t} \Bigg(\sum_{\bar{n} \in \bar{N}^c} \Bigg(& e_{\bar{n}}^s \times \left(1 - \ddot{a}_{\bar{n}}^t\right) + \ddot{a}_{\bar{n}}^t \\
& \times \Bigg(e_{\bar{n}}^b + \left(e_{\bar{n}}^m - e_{\bar{n}}^b\right) \times \sum_{n \in N_i} \sum_{r \in R} \hat{x}_{n\bar{n}}^{ti} \\
& \times \varphi_{f(n)}^r \times \xi^r \Bigg)\Bigg) \\
+ \sum_{\bar{s} \in \bar{S}^c} \Bigg(& e_{\bar{s}}^s \times \left(1 - \ddot{a}_{\bar{s}}^t\right) + \ddot{a}_{\bar{s}}^t \\
& \times \Bigg(e_{\bar{s}}^b + \sum_{\bar{\pi} \in \bar{\Pi}_s} \ddot{a}_{\bar{\pi}\bar{s}}^t \times e_{m(\bar{\pi})} \Bigg)\Bigg) \\
+ \sum_{l \in L_i} \sum_{\bar{p} \in \mathcal{B}(\bar{P}^c)} & \hat{y}_{l\bar{p}}^{ti} \times \beta_l \times \xi^\beta \Bigg) \times d_t
\end{aligned}
\tag{16}
$$

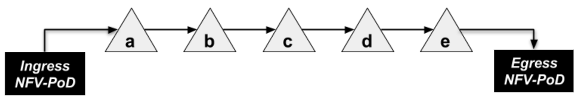


Fig. 4. Example of a Service Chain.

Now, the total brown energy consumption for time-duration $[t, t+1)$ and NFV-PoD c is equal to:

$$E^{tc} = \max\left(E_a^{tc} - e_g^{tc},\ 0\right)$$

Finally, the carbon footprint during a time-duration $[t, t+1)$ for the whole network is:

$$K^t = \sum_{c \in \mathcal{C}} E^{tc} \times \zeta_c \tag{17}$$

Our objective is to minimize (17) over all timeslots:

$$\textit{minimize} \sum_{t \in \mathcal{T}} K^t \tag{18}$$

Computing the optimal solution for large networks will require a substantial amount of time since the MLSO problem is NP-Hard. Solving for the optimal solution is not a suitable approach in an online setting, where SFC requests must be embedded within seconds. In the following, we present three heuristic algorithms that decompose the MLSO problem into three subproblems and provide fast near-optimal solutions.

VI. Heuristics for Orchestration

In this section, we describe three heuristic algorithms for SFC embedding, consolidation, and migration. After receiving a new SFC request, the heuristic for embedding is used to embed the VNFs and the inter-VNF links in the SFC. SFC migration and consolidation decisions are taken based on the following events: (i) changes in the availability of renewable energy; and (ii) departure of one or more SFCs from the system. We first describe the heuristic for SFC embedding. After that we present the heuristic for taking migration decisions, followed by the heuristic for resource consolidation.

A. SFC Embedding

The SFC embedding algorithm is a three-stage heuristic. Stages one and three are executed by the G-Or, and stage two is executed by the L-Or(s) who are selected by the G-Or to host the SFC. The G-Or receives the SFC request as input and determines a set of candidate NFV-PoDs to eventually host the chain. An example chain is shown in Fig. 4. The output of the embedding algorithm is a mapping for each VNF and inter-VNF links in the chain to a set of NFV-PoDs and physical links within and across these NFV-PoDs. The three stages of the embedding heuristic are as follows:

Stage-1: The G-Or calculates a set of potential paths between the ingress and egress NFV-PoDs of the SFC. Paths with a higher delay than the maximum allowed latency of the chain are immediately discarded. Among the remaining paths, the one with the maximum aggregate renewable energy is chosen. Then the G-Or sends the details of the SFC to each NFV-PoD (L-Or) on this path.

TABLE II
COST-MATRIX COMPUTED BY L-OR

	a	b	c	d	e
a	*c*(a)	*c*(a-b)	*c*(a-b-c)	*c*(a-b-c-d)	*c*(a-b-c-d-e)
b	-	*c*(b)	*c*(b-c)	*c*(b-c-d)	*c*(b-c-d-e)
c	-	-	*c*(c)	*c*(c-d)	*c*(c-d-e)
d	-	-	-	*c*(d)	*c*(d-e)
e	-	-	-	-	*c*(e)

TABLE III
EMBEDDING TABLE FOR TABU SEARCH

	a	b	c	d	e
U	0	0	0	0	0
V	1	0	0	0	0
W	0	1	1	0	0
X	0	0	0	1	0
Y	0	0	0	0	1
Z	0	0	0	0	0

Stage-2: The L-Ors (chosen by the G-Or in Stage-1) run the second stage of the heuristic. Each L-Or computes a cost-matrix representing the embedding cost of sub-chains of the original SFC. An example cost-matrix for the SFC in Fig. 4 is shown in Table II. Each cell of this matrix represents the cost of embedding a specific sub-chain of the original chain. The L-Or also computes and stores the embedding locations of these partial chains. The L-Ors return the cost-matrix and embedding locations to the G-Or for further processing.

Stage-3: The final stage of the heuristic is executed by the G-Or. Upon receiving the cost-matrices from the NFV-PoDs selected in Stage-1, the G-Or constructs an embedding table as shown in Table III. Each cell of this table represents the mapping of a VNF (on the column) to an NFV-PoD (on the row). The cell values are binary, *i.e.*, 0 or 1. In the final mapping, each column will contain exactly one 1. A row can contain multiple 1s, denoting the embedding of more than one VNF on the same NFV-PoD. There is an exponential number of possibilities with respect to the number of rows. Therefore, the final embedding is determined by using *Tabu Search* to selectively explore the search-space of all possible embeddings within a reasonable amount of time.

Algorithm 1, outlines the pseudo-code for path selection in Stage-1. The algorithm takes as input the SFC *s* and *k*, the number of paths to consider between the ingress and egress NFV-PoDs. The parameter *k* is used for finding the *k*-shortest paths between the NFV-PoDs (Line 1). A small *k* reduces the running time of the algorithms, but restricts the search space of embedding paths. A large *k* increases the chances of finding a better embedding at the cost of increased processing time. During our performance evaluation, the value of *k* is varied between 3 to 7. Next, line 2 removes any path with a higher delay than the maximum tolerable delay of the chain. Finally, the path with the maximum aggregate renewable energy is computed and returned (Line 3–7).

The computation performed in Stage-2 is shown in Algorithm 2. It is a Dynamic Program (DP) to calculate the embedding cost of sub-chains (chain suffixes) in a bottom-up manner. Line 3 works as the base case for the DP. Each *cm*[*i*][*i*] entry of the cost-matrix is initialized with the cost of embedding the *i*–th VNF in the chain. Next, Line 5 is the recursive call for computing embedding cost of a longer chain using the already computed costs of shorter chains. The embedding cost is computed based on a first-fit strategy, where the servers are pre-sorted according to their ids. Then, servers are considered in increasing order of id and the first server with enough capacity to host a VNF is chosen as the target server. The size of the cost-matrix is $n \times n$, where n is the length of the service chain. Algorithm 2 fills the upper diagonal of the matrix, so there are $\frac{n(n+1)}{2}$ entries in total. For each embedding, the above-mentioned process might need to check all servers. Assuming that there are *k* servers within a single NFV-PoD, the running time of Algorithm 2 is $O(kn^2)$.

Algorithm 1 Embedding: Stage-1 (Runs @G-Or)

```
Input: service chain, s; number of considered paths between ingress
    and egress NFV-PoDs, k
Output: path p for embedding s
1: $P \leftarrow kShortestPaths(s.ingress, s.egress, k)$
2: Remove any path $p \in P$ s.t. $p.\delta > s.\delta$
3: for all path $p \in P$ do
4:     for all NFV-PoD $c \in p$ do
5:         $p.e_r \leftarrow p.e_r + c.e_r$
6: sort(P) {In decreasing order of renewable energy}
7: return P[1]
```

Algorithm 2 Embedding: Stage 2 (Runs @L-Or)

```
Input: service chain, s;
Output: cost-matrix for s
1: initialize cm
2: for $i = 1$ to s.size do
3:     $cm[i][i] = embeddingCost(s[i])$
4:     for $j = i + 1$ to s.size do
5:         $cm[i][j] \leftarrow cm[i][j-1] + embeddingCost\ (s[j])$
6: return cm
```

The third and final stage of the heuristic is presented in Algorithm 3. In this stage, we assume that chain embedding always progresses in the *forward direction* on the selected path. For the path `U->V->W->X->Y->Z`, if the second VNF `b` is embedded on NFV-PoD `W`, then the subsequent VNFs (*i.e.*, `c, d, e`) cannot be embedded on any NFV-PoD that comes before `W` on the path (*i.e.*, `U` and `V`); they must be embedded on either `W` or NFV-PoDs that come after `W`, *i.e.*, `X, Y,` and `Z`. Algorithm 3 first initializes an embedding-table (*e.g.*, Table III) by filling all the cells with zeros. Then, line 3, fills the embedding table with sub-optimal solution based on the first-fit approach. Here, each VNF is embedded on the first NFV-PoD that has enough capacity maintaining the ordering constraint mentioned above. After that, we perform tabu search on the embedding-table as follows to find a better solution:

1) Initial Solution: The initial solution is generated based on the first-fit approach (Line 3). Algorithm 3 attempts to embed the next VNF in the next NFV-PoD on the path. It maintains the constraint that an SFC's links always move in the forward direction. If first-fit fails to generate a valid solution, then the initial solution is generated by randomly embedding VNFs on any NFV-PoD with enough capacity (line 5).

Algorithm 3 Embedding: Stage 3 (Runs @G-Or)

```
Input: cost-matrix for $s$ from each NFV-PoD on path $p$;
Output: mapping of VNFs and inter-VNF links in $s$ to NFV-PoDs on path $p$
1: $S \leftarrow \phi$ {$S$: current solution}
2: if firstFitSolution() is valid then
3:     $S \leftarrow$ firstFitSolution()
4: else
5:     $S \leftarrow$ randomSolution()
6: $S^* \leftarrow S$ {$S^*$: best solution so far}
7: while stopping criteria not met do
8:     $SS \leftarrow \phi$
9:     for $S^c \in N(S)$ do
10:        if $S^c \notin T$ then
11:            $SS \leftarrow SS \cup S^c$
12:    $S \leftarrow best(SS)$
13:    $add(T, S, S^*)$
14:    if $cost(S) < cost(S^*)$ then
15:        $S^* \leftarrow S$
16:    update(T)
17: return $S^*$
```

2) *Solution Neighborhood:* Each iteration of the tabu search moves from the current solution to a neighboring solution. Here, solutions are defined by the values in the embedding-table, hence, one solution is the neighbor of another solution if they differ in exactly one column, *i.e.*, the embedding location of one VNF. The key element for the performance of tabu search is the fast computation of costs for a solution. Here, we can calculate the cost of an embedding quickly from the cost-matrices calculated in the previous stage. The cost computation reduces to at most *m* cost-matrix lookups where *m* is the number of VNFs in the service chain.

3) *Tabu List:* A list of solutions or rules which are deemed tabu (forbidden) for a specific amount of time, enables tabu search to avoid getting stuck at locally optimal solutions. If a VNF *v* is moved away from a server *s*, then we add the pair $\langle v, s\rangle$ to the tabu list *T* (line 13 or Algorithm 3). The item stays in the list for $m - 1$ iterations, which gives other VNFs the opportunity to change their embedding. The amount of time each rule stays in the tabu list is updated at line 16.

4) *Stopping Criteria:* If the best solution (S^*) does not improve for $m + n$ iterations, where *m* and *n* are the number of VNFs and NFV-PoDs, then the tabu search is terminated.

B. SFC Migration

SFC migration decisions are also made using the tabu search algorithm described above. The algorithm first builds a list of potential paths between the ingress and egress NFV-PoDs and then orders them according to the number of overlapping NFV-PoDs with the current path. These paths are monitored for changes in renewable energy and available resource. Whenever there is an increase in embedding cost or availability of new resource at one or multiple NFV-PoDs on these paths, the embedding-table with updated costs is used to perform another tabu search for the candidate path. This approach restricts migrations on a subset of the possible paths but provides a fast and simple algorithm for taking migration decisions.

Algorithm 4 Resource Consolidation

```
Input: resource allocation state of the NFV-PoD
1: $S \leftarrow$ active servers
2: sortDecreasingID($S$)
3: for all server $s \in S$ do
4:     $canMoveAllVNFs \leftarrow$ true
5:     $vnfDests \leftarrow \phi$
6:     for all VNF $v$ hosted on $s$ do
7:         $minPower \leftarrow MAX$
8:         $selectedServer \leftarrow NULL$
9:         for all server $t \in S$ and $t.id < s.id$ do
10:            if $t.residual >= v.req$ and $t.power(v) < minPower$ then
11:                $minPower \leftarrow t.power(v)$
12:                $selectedServer \leftarrow t$
13:        if $selectedServer = NULL$ then
14:            $canMoveAllVNFs \leftarrow$ false
15:            break
16:        else
17:            $vnfDests \leftarrow vnfDests \cup \{v, selectedServer\}$
18:    if $canMoveAllVNFs =$ true then
19:        moveVNFs($s$, $vnfDests$)
20:        put $s$ in sleep mode
```

C. SFC Consolidation

The consolidation heuristic is run locally by the L-Or at each NFV-PoD after an SFC's lifetime expires and resources allocated for that SFC are freed. Consolidation is performed by moving VNFs towards one side (represented by sequential numbering of the servers) of the server racks of the NFV-PoD. We assume that each server in a NFV-PoD is numbered and the numbering starts from one side and increases towards the other side of the server racks. The consolidation algorithm moves VNFs towards lower numbered servers and updates their routing paths accordingly. This problem is similar to the bin-packing problem, where VNFs on a candidate server to be turned off are considered to be the items and the rest of the active servers are considered as bins with different capacities. Here, we use a modified version of the Best Fit Decreasing (BFD) [38] algorithm to find a solution. The BFD algorithms is known to perform reasonably well for bin packing problems [39]. The algorithm is shown in Algorithm 4. The consolidation algorithm takes the current resource allocation state as input and determines a sequence of VNF moves for resource consolidation. If all VNFs from a server can be moved to other active servers, then that server is put in sleep mode.

VII. Performance Evaluation

We perform trace-driven simulations using real ISP network topologies and SFC request patterns to compare and contrast the performances of our proposed algorithms. In the following, we first describe the datasets in Section VII-A, then the performance metrics in Section VII-B, followed by the simulation setup in Section VII-B. Finally, a detailed discussion of the simulation results is presented in Section VII-D.

A. Datasets

1) *SFC Data:* We generate SFCs with lengths between three and six consisting of VNFs from Table IV. Due to the

TABLE IV
VNF CPU REQUIREMENT AND PROCESSING DELAY [44], [45]

Network Function	CPU Required	Processing Delay
Firewall 1	4	0.5 ms
Firewall 2	2	0.8 ms
Proxy 1	4	0.025 ms
Proxy 2	8	0.001 ms
NAT 1	8	0.1 ms
NAT 2	16	0.05 ms
IDS	4	0 ms

TABLE V
SERVER, SWITCH, AND SWITCH PORT POWER CONSUMPTION

Server Data [47]		
Physical CPU Cores	Idle Energy	Peak Energy
16	80.5W	2735W
Switch Data [14], [15]		
Dell PowerConnect	Sleep Mode Energy	Active Mode Energy
	10W	110W
Switch Port Data [15]		
Port Link Rate	1 Gbps	10 Gbps
Energy Consumption	1.2W	4.3W

lack of any published real-world data on the type and length of SFCs, we resort to a synthetic generation method. We surveyed the relevant research literature [19], [40]–[42] and Internet drafts [43] to determine the set of VNFs, and the length of SFCs to use in our simulations. VNF characteristics listed in Table V were obtained by studying the relevant literature and product data sheets [41], [44], [45]. SFCs are generated between random source-destination locations according to a Poisson process with an arrival rate of λ. The lifetime of each SFC is determined based on an exponential distribution with a mean of $1/\mu$. The specific values of λ and μ are determined based on the objectives of a particular simulation. The bandwidth demands of the SFCs are generated according to a sine-cyclostationary traffic matrix sequence generated according to the process described in [46]. The mean and standard deviation of traffic volume is set to 500 Mbps and 0.9, respectively. The maximum delay for an SFC is uniformly distributed between 100 ms to 1 s.

2) Power Consumption Data: The power consumption data for servers, switches, and VNFs are presented in Table V.

3) Renewable Energy Data: The publicly available renewable energy data from the U.S. Climate Reference Network and Regional Climate Reference Network [11] provide an estimate of energy generation potential from solar and wind. NFV-PoDs are assumed to host solar panels with sizes varied randomly between 4 to 8 square meters. For wind energy, the number of wind turbines is varied randomly between 1 to 3.

4) Network Topology: We use PoP-level topologies of four Autonomous Systems (ASs) obtained from the RocketFuel topology dataset [18] (Table VI). The PoP locations in these topologies are assumed to represent potential NFV-PoD locations. Inside each PoP, we assume the presence of a fat-tree data center network consisting of switches, gateway router, and servers. For the sake of simplicity, we assume a linear power consumption model for the servers, *i.e.*, power consumption increases linearly with the number of CPUs used [48].

TABLE VI
NETWORK TOPOLOGIES

Topology	Node Count	Edge Count	Maximum Edge Latency
AS-13129	7	9	3 ms
AS-7170	18	60	43 ms
AS-3549	61	486	66 ms
AS-3561	92	329	60 ms

B. Performance Metrics

1) Carbon Footprint: Carbon footprint is measured for all nodes and physical links in the network. We report the carbon footprint by summing the carbon footprint for each NFV-PoD and the inter-NFV-PoD links. So, even if an NFV-PoD is not participating in embedding a particular SFC, its contribution is summed to calculate the overall carbon footprint (18). This approach ensures a fair comparison between the optimal and heuristic solutions.

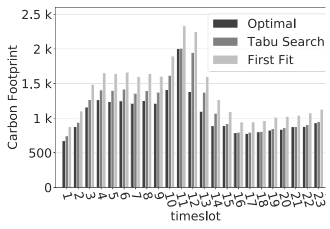
2) Renewable Energy Usage: This metric quantifies how much renewable energy is utilized at each time-instance by the solutions. Even though the utilization of renewable energy is not a direct objective of our problem, this metric is indirectly optimized since more renewable energy leads to less carbon footprint. We report the proportion of renewable energy usage over total energy consumption for measuring the effectiveness of the proposed algorithms in utilizing renewable energy.

3) Server, NFV-PoD, and Path Spread: Server-spread is the ratio of the number of servers occupied by an SFC to the number of VNFs present in that SFC. This metric measures the distribution of an SFC over servers. Similar to server-spread, NFV-POD-spread is measured as the ratio of the number of distinct NFV-PODs occupied by an SFC to the number of VNFs in that SFC. Finally, path spread is defined as the ratio of the number of physical edges used to embed an SFC to the number of virtual links in the SFC. These metrics help to determine the importance of computing switch and switch-port power while taking embedding decisions.

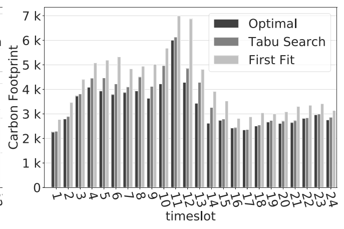
4) SFC Acceptance Ratio: We measure SFC acceptance ratio as the ratio of the number of embedded SFCs to the total number of SFC requests. Acceptance ratio is correlated with the overall carbon footprint. For instance, a lower acceptance ratio will certainly reduce the overall carbon footprint. Therefore, during the simulations, datasets are pre-processed to avoid the impact of one of these metrics on the other. While measuring carbon footprint, datasets are per-processed to ensure that both optimal and heuristic solutions can embed 100% of the SFC requests, thereby, nullifying the impact of acceptance ratio. In turn, when comparing acceptance ratios of optimal and heuristic solutions, it is ensured that an SFC request is not rejected when there are adequate resources available in the network.

C. Simulation Setup

We perform extensive simulations to show the effectiveness of the proposed algorithms in terms of overall carbon footprint, utilization of available renewable energy, and acceptance ratio. In addition, the impact of migration on the overall carbon footprint is also reported by running two identical



(a) Carbon Footprint (AS-13129)



(b) Carbon Footprint (AS-7170)

Fig. 5. Carbon footprint per timeslot.

TABLE VII
SOLUTION EXECUTION TIME (MS) PER SFC REQUEST

Topology	Tabu Search			Optimal		
	5^{th}	Mean	95^{th}	5^{th}	Mean	95^{th}
AS-13129	2	4.4	7	2100	4400	7500
AS-7170	3	4.5	7	27500	88300	175000
AS-3549	4	5.5	9	-	-	-
AS-3561	4	5.9	13	-	-	-

simulations with and without enabling migration. In all experiments, we simulate 24 hours of new and expired SFC requests. Renewable energy at the NFV-PoDs is simulated by assigning them to different U.S. cities obtained from the renewable energy data [11]. The ILP formulation of the MLSO problem is implemented using the IBM CPLEX Optimizer v12.5. All simulations are conducted on a machine with a 4×8 core 2.4Ghz Intel Xeon E5-4640 CPU and 512GB of memory. We developed an in-house discrete event simulator for conducting simulations involving SFC arrival and departures over time. We used a Poisson distribution for new SFC arrivals with the mean arrival rate (λ) varied between 0.01 to 0.10 per time unit (*i.e.*, a simulated second). SFC lifetime (μ) was drawn from an exponential distribution with a mean of one simulated hour.

D. Results

1) Carbon Footprint: We present the results on carbon footprint over time in Fig. 5(a) and Fig. 5(b) for AS-13129 and AS-7170, respectively. For these results, we ensured that all the SFCs were embedded by all three algorithms for a fair comparison. The increase and decrease in the overall carbon footprint for the optimal solution and tabu-search heuristic exhibit a similar pattern, which demonstrates the adaptability of the heuristic algorithms to traffic fluctuations.

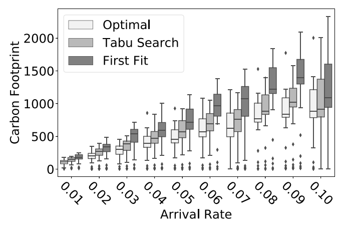
The next set of plots show the impact of load, *i.e.*, arrival rate of new SFC requests, on carbon footprint. Fig. 6(a) and Fig. 6(b) presents the carbon footprint obtained from running all three solutions on AS-13129 and AS-7170 topology, respectively. For AS-13129, the ratio of carbon footprint between tabu-search and the optimal solution varies between 1.1 and 1.3 across all arrival rates. Compared to AS-13129 topology, AS-7170 has a higher proportion of physical links, which increases the number of possible paths for SFC embedding. The optimal solution always chooses the least-cost path at the expense of higher running time. Our heuristic solution exhibits significant speedup in running time for AS-7170 without significantly deviating from the optimal solution. For the other two AS topologies, the optimal solution failed to find embedding for a single SFC even after running for an hour. Hence, we refrained from running the optimal solution on these topologies.

2) Renewable Energy Usage: Fig. 6(c) and Fig. 6(d) show the renewable energy proportions for AS-13129 and AS-7170, respectively. The optimal solution achieves better renewable energy utilization than the tabu-search heuristic. The primary reason behind this is the way tabu-search chooses a path to embed an SFC. It chooses a path that offers the highest aggregate renewable energy. However, the total renewable energy might be concentrated on a few NFV-PoDs on the path. Choosing such a path can reduce the carbon footprint of the servers but significantly increases the carbon footprint for embedding the SFC's links. This behavior is discovered by closely examining the embedding locations of the optimal and heuristic solutions. The performance of the tabu-search heuristic can be improved by examining more paths. However, this approach increases the running time and does not improve performance in all cases. Therefore, we opted for a faster and simpler heuristic algorithm by choosing the current path selection approach. Tabu-search's mean renewable energy utilization is within 0.72 to 0.84 of the optimal, while the first-fit heuristic's utilization is only within 0.4 to 0.6 of the optimal.

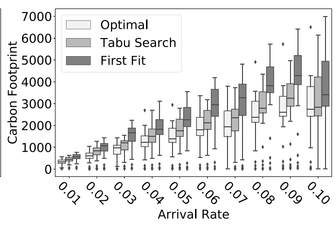
3) Execution Time: Execution time statistics of the optimal solution and tabu-search for a single SFC is reported in Table VII. AS-13129 (7 nodes and 9 edges) is the smallest topology used in the simulations. The optimal solution requires a 5^{th}-percentile of 2.1 seconds to embed an SFC, while the mean and 95^{th}-percentile time required are 4.4 and 7.5 seconds, respectively. In contrast, tabu-search requires a 95^{th}-percentile of only 7 ms to embed an SFC, which is a two orders of magnitude improvement compared to even the minimum time required for the optimal solution. For AS-7170 (18 nodes and 60 edges) the optimal solution requires 1 minute and 28 seconds on average, whereas the heuristic requires only 4.5 ms, which is four orders of magnitude faster. The CPLEX-based optimal solution was unable to solve problem instances involving AS-3549 and AS-3561 topologies. In contrast, the heuristic took only 5.5 and 5.9 ms on average for AS-3549 and AS-3561, respectively. The execution time of the baseline first-fit heuristic is less than 2 ms irrespective of the topology, hence, we do not report them in the table.

4) Server, NFV-PoD, and Path Spread: Fig. 7 presents the histogram of server and NFV-POD spread for AS-13129 with and without considering networking equipment (*e.g.*, switch and switch-port) power consumption during embedding. When we are not considering networking equipment power consumption, there is no additional cost for distributing an SFC across servers or NFV-PoDs. Consequently, the optimization algorithms try to embed VNFs without considering the length of the embedding path, resulting in higher carbon footprint.

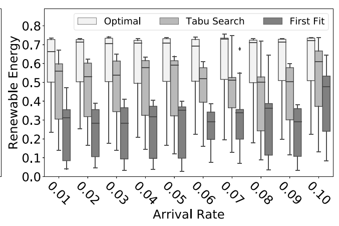
We present a histogram of server-spread values for AS-13129 in Fig. 7(a). A server-spread value of 0.5 means that an SFC with four VNFs was embedded on two servers. Whereas, a server-spread value of 1.0 indicates that an SFC with four VNFs was embedded on four servers. As we can see, if we do not consider networking equipment power consumption,



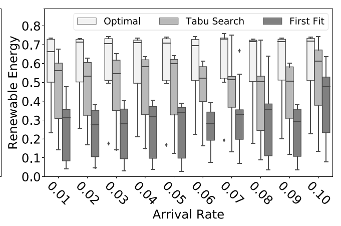
(a) Carbon Footprint (AS-13129)



(b) Carbon Footprint (AS-7170)

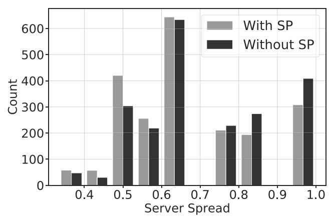


(c) Renewable Energy (AS-13129)

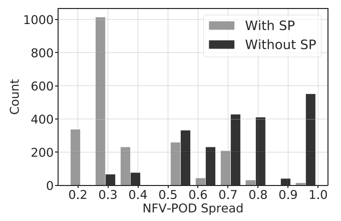


(d) Renewable Energy (AS-7170)

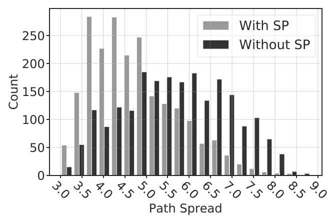
Fig. 6. Carbon footprint and renewable energy.



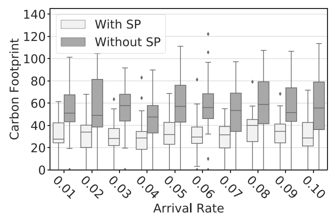
(a) Server Spread Histogram



(b) NFV-POD Spread Histogram



(c) Path Spread Histogram



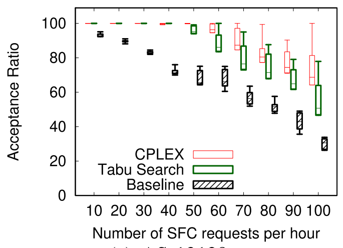
(d) Impact on Carbon Footprint

Fig. 7. Impact of considering switching power.

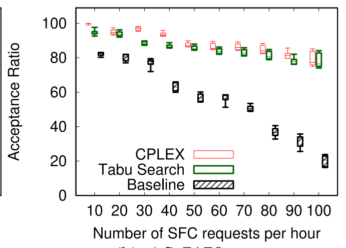
then the distribution of server-spread values moves closer to 1.0. We also observe similar impact of considering networking equipment power consumption on path-spread in Fig. 7(c) as demonstrated by the shifting of the darker bars towards the right. A similar trend, however, with a more profound impact is observed in the case of NFV-PoD spread in Fig. 7(b). It is a direct consequence of the fact that without considering networking equipment power consumption, the optimizer tries to embed more VNFs at NFV-PODs with surplus renewable energy. However, the cost of sending traffic to a distant NFV-PoD can be much higher compared to the savings in carbon footprint by utilizing renewable energy at that NFV-PoD.

The impact of considering networking equipment power consumption on carbon footprint becomes more evident with increased load. For instance, as illustrated in Fig. 7(d), for a mean arrival rate of 10 SFCs per 100 time units, the mean value of carbon footprint almost doubles when power consumption of networking equipments is not considered.

5) *Acceptance Ratio:* In these simulations, we vary the mean arrival rate of new SFC requests between 10 and 100 per simulated hour with a mean lifetime of one simulated hour. We run the simulation for 24 simulated hours and track the percentage of embedded SFCs within each hour. The acceptance ratio is calculated hourly, and then the min, 5^{th}-percentile, mean, 95^{th}-percentile, and the max values are presented for the optimal, tabu-search heuristic, and the baseline heuristic in Fig. 8(a) and Fig. 8(b) for AS-13129 and AS-7107, respectively. We observe that the acceptance ratios of the heuristic solution are very close to those of the optimal in both cases. For AS-13129, the acceptance ratio of the heuristic is within 12% of the optimal, except for the 100 request per hour setting. For AS-7170, the heuristic performs much better as there are more paths than AS-13129, which offers more optimization opportunities to the heuristic. For AS-7170, the acceptance ratio of the heuristic remains within 7% of the optimal. The

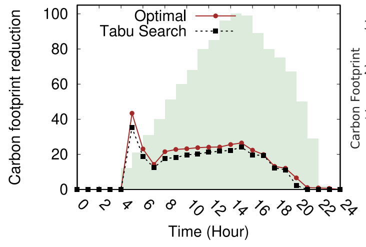


(a) AS-13129

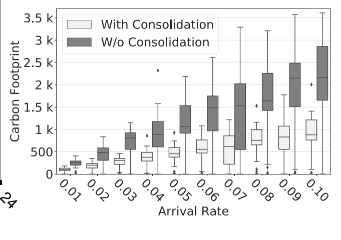


(b) AS-7170

Fig. 8. Acceptance Ratio.



(a) Impact of Migration

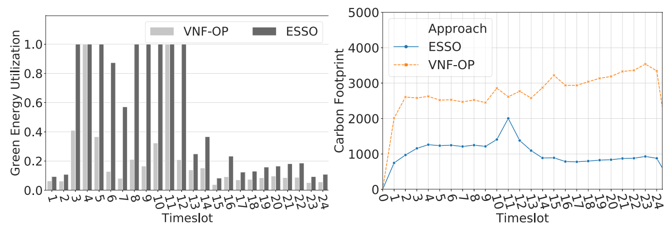


(b) Impact of Consolidation

Fig. 9. Impact of Migration and Consolidation (AS-13129).

acceptance ratio of the baseline heuristic is much lower than the tabu-search heuristic for both topologies.

6) *Impact of Migration:* To measure the impact of migration, we run simulations with and without enabling SFC migration on AS-12129 and present the results in Fig. 9(a). We pre-processed the renewable energy data to start SFC embedding at a time when there is no renewable energy, and then the amount of renewable energy rises, and finally, it decreases. This approach represents a renewable energy source that exhibits diurnal pattern (*e.g.*, solar panel), and captures the impact of migration in three possible scenarios. The normalized value of available renewable energy per hour is plotted as the shaded (background) bar chart in Fig. 9(a). The amount of



(a) Green Energy Utilization (b) Carbon Footprint

Fig. 10. Impact of Renewable Energy Awareness.

available renewable energy is calculated by summing the available renewable energy at all NFV-PoDs. From the figure, we can see that peak renewable energy is observed during the 14th hour and around 50% of the peak energy is observed around 8th and 20th hours. We observed ≈20% reduction in carbon footprint due to migration for both optimal and heuristic solutions.

The differences between the percentage reduction in carbon footprint for the optimal and heuristic solutions are very small. The initial rise in carbon footprint reduction is because renewable energy becomes available for the first time at hour 4, and a significant number of SFCs are migrated to reduce carbon footprint. However, the effect subsides after hour 5 as the only candidate SFCs that can reduce overall carbon footprint are the ones that arrived during hour 4. After that, the percentage reduction increases with increasing renewable energy. However, when the amount of renewable energy starts decreasing, the percentage reduction in carbon footprint also keeps decreasing. At this stage, the only opportunity to reduce carbon footprint occurs when some SFCs expire, and resources powered by renewable energy become available. We observed similar behavior for AS-7170.

7) Impact of Consolidation: The impact of resource consolidation is shown in Fig. 9(b). Here, we have plotted carbon footprint against arrival rate for two scenarios: (i) with consolidation and (ii) without consolidation. We have kept the SFC average lifetime constant (one simulated hour) for this simulation to avoid its impact on the result. Impact of consolidation is more profound for shorter lifetime as there is more churn, hence, more resource fragmentation in the network. Here, we focus on the impact of consolidation for different SFC arrival rates. The carbon footprint of the network will increase with the increase in arrival rate. However, as we can see from the figure, without consolidation, carbon footprint rises at a much higher rate compared to the case when consolidation is enabled. The takeaway from this simulation is that consolidation reduces resource fragmentation in the network and can lead to more efficient resource utilization.

8) Impact of Renewable Energy Awareness: We revisit our motivational experiment from Section II to demonstrate the effectiveness of ESSO in better utilizing renewable energy sources, in this way reducing overall carbon footprint. Fig. 10 shows the impact of renewable energy aware embedding on the utilization of available renewable energy (Fig. 10(a)) and overall carbon footprint (Fig. 10(b)). From Fig. 10(a), we can see that ESSO utilizes significantly more renewable energy compared to VNF-OP. ESSO utilizes 100% renewable energy for eight timeslots, while VNF-OP can reach 100% utilization only for two timeslots. Fig. 10(b) shows the overall carbon footprint for VNF-OP and ESSO. As we can see from the figure, ESSO can reduce carbon footprint by two to three times by considering the availability of renewable power.

VIII. Conclusion

Telecommunication networks are rapidly expanding to cope with the traffic growing at an unprecedented rate. Such expansion is substantially increasing the carbon footprint generated from powering the telecommunication infrastructure. Government regulations and environmental concerns are pushing network operators to devise innovative solutions for minimizing carbon footprint. Given the variable cost and availability of renewable energy sources, it is becoming challenging for network operators to comply with such regulations without significantly increasing their operational cost. In this perspective, we propose ESSO, an SFC orchestrator that reduces the overall carbon footprint of a telecommunication network by intelligently embedding and migrating SFCs across different locations and consolidating resources whenever possible. Our simulations on real network topologies demonstrate that even for a small network with seven locations, considering the availability of renewable energy sources during SFC embedding can result in 2-3× reduction in carbon footprint. We hope that ESSO will stimulate further research in energy-aware network service orchestration. One interesting future research direction is to investigate pricing models for network services, where the network operator profits from embedding an SFC but pays for electricity and penalties for carbon emission and SLA violations.

References

[1] L. Peterson *et al.*, "Central office re-architected as a data center," *IEEE Commun. Mag.*, vol. 54, no. 10, pp. 96–101, Oct. 2016.
[2] T. Joyce, T. A. Okrasinski, and W. Schaeffer, "Estimating the carbon footprint of telecommunications products: A heuristic approach," *J. Mech. Design*, vol. 132, no. 9, 2010, Art. no. 094502.
[3] *G.W.A.T.T.—Global 'What If' Analyzer of Network Energy Consumption*. Accessed: Apr. 10, 2019. [Online]. Available: http://gwatt.net/
[4] H. S. Dunn, *GISW Thematic Report: The Carbon Footprint of ICTs*, Glob. Inf. Soc. Watch, Assoc. Progressive Commun., Johannesburg, South Africa, 2010, pp. 15–16.
[5] *Federal Renewable Energy Projects and Technologies*. Accessed: Apr. 10, 2019. [Online]. Available: http://energy.gov/eere/femp/federal-renewable-energy-projects-and-technologies
[6] H. Ikebe, N. Yamashita, and R. Nishii, "Green energy for telecommunications," in *Proc. IEEE INTELEC*, 2007, pp. 750–755.
[7] *100% Renewable Is Just the Beginning*. Accessed: Apr. 10, 2019. [Online]. Available: https://environment.google/projects/announcement-100/
[8] J. Sherry, S. Hasan, C. Scott, A. Krishnamurthy, S. Ratnasamy, and V. Sekar, "Making middleboxes someone else's problem: Network processing as a cloud service," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 4, pp. 13–24, 2012.
[9] V. Sekar, S. Ratnasamy, M. K. Reiter, N. Egi, and G. Shi, "The middlebox manifesto: Enabling innovation in middlebox deployment," in *Proc. ACM HotNets-X*, 2011, pp. 1–6.
[10] H. Freeman and R. Boutaba, "Networking industry transformation through softwarization [the president's page]," *IEEE Commun. Mag.*, vol. 54, no. 8, pp. 4–6, Aug. 2016.
[11] *USCRN/USRCRN Quality Controlled Datasets*. Accessed: Apr. 10, 2019. [Online]. Available: https://www.ncdc.noaa.gov/crn/qcdatasets.html

[12] *State Electricity Profiles*. Accessed: Apr. 10, 2019. [Online]. Available: https://www.eia.gov/electricity/state/
[13] B. Heller *et al.*, "ElasticTree: Saving energy in data center networks," in *Proc. NSDI*, vol. 10, 2010, pp. 249–264.
[14] *PowerConnect 8024F Switch*. Accessed: Apr. 10, 2019. [Online]. Available: http://www.dell.com/us/business/p/powerconnect-8024f/pd
[15] S. Fang, H. Li, C. H. Foh, Y. Wen, and K. M. M. Aung, "Energy optimizations for data center network: Formulation and its solution," in *Proc. IEEE GLOBECOM*, 2012, pp. 3256–3261.
[16] T. H. Vu, T. Thanh, V. Q. Trong, P. N. Nam, and N. H. Thanh, "NetFPGA based OpenFlow switch extension for energy saving in data centers," *REV J. Electron. Commun.*, vol. 3, nos. 1–2, pp. 77–83, 2013.
[17] T. H. Vu, V. C. Luc, N. Quan, T. Thanh, N. Thanh, and P. Nam, "Sleep mode and wakeup method for OpenFlow switches," *J. Low Power Electron.*, vol. 10, no. 3, pp. 347–353, 2014.
[18] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson, "Measuring ISP topologies with rocketfuel," *IEEE/ACM Trans. Netw.*, vol. 12, no. 1, pp. 2–16, Feb. 2004.
[19] F. Bari, S. R. Chowdhury, R. Ahmed, R. Boutaba, and O. C. M. B. Duarte, "Orchestrating virtualized network functions," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 4, pp. 725–739, Dec. 2016.
[20] J. G. Herrera and J. F. Botero, "Resource allocation in NFV: A comprehensive survey," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 3, pp. 518–532, Sep. 2016.
[21] Y. Xie, Z. Liu, S. Wang, and Y. Wang, "Service function chaining resource allocation: A survey," *eprint arXiv:1608.00095*, 2016. [Online]. Available: http://arxiv.org/abs/1608.00095
[22] V. Eramo, E. Miucci, M. Ammar, and F. G. Lavacca, "An approach for service function chain routing and virtual function network instance migration in network function virtualization architectures," *IEEE/ACM Trans. Netw.*, vol. 25, no. 4, pp. 2008–2025, Aug. 2017.
[23] M. M. Tajiki, S. Salsano, L. Chiaraviglio, M. Shojafar, and B. Akbari, "Joint energy efficient and QoS-aware path allocation and VNF placement for service function chaining," *IEEE Trans. Netw. Service Manag.*, vol. 16, no. 1, pp. 374–388, Mar. 2019.
[24] C. Assi, S. Ayoubi, N. El Khoury, and L. Qu, "Energy-aware mapping and scheduling of network flows with deadlines on VNFs," *IEEE Trans. Green Commun. Netw.*, vol. 3, no. 1, pp. 192–204, Mar. 2019.
[25] O. Soualah, M. Mechtri, C. Ghribi, and D. Zeghlache, "A green VNFs placement and chaining algorithm," in *Proc. IEEE/IFIP NOMS*, Apr. 2018, pp. 1–5.
[26] O. Soualah, M. Mechtri, C. Ghribi, and D. Zeghlache, "A green VNF-FG embedding algorithm," in *Proc. IEEE NetSoft*, Jun. 2018, pp. 141–149.
[27] G. Sun, Y. Li, H. Yu, A. V. Vasilakos, X. Du, and M. Guizani, "Energy-efficient and traffic-aware service function chaining orchestration in multi-domain networks," *Future Gener. Comput. Syst.*, vol. 91, pp. 347–360, Feb. 2019.
[28] N. M. K. Chowdhury and R. Boutaba, "A survey of network virtualization," *Comput. Netw.*, vol. 54, no. 5, pp. 862–876, 2010.
[29] M. F. Bari *et al.*, "Data center network virtualization: A survey," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 2, pp. 909–928, 2nd Quart., 2013.
[30] X. Guan, B.-Y. Choi, and S. Song, "Energy efficient virtual network embedding for green data centers using data center topology and future migration," *Comput. Commun.*, vol. 69, pp. 50–59, Sep. 2015.
[31] A. Amokrane, M. F. Zhani, R. Langar, R. Boutaba, and G. Pujolle, "Greenhead: Virtual data center embedding across distributed infrastructures," *IEEE Trans. Cloud Comput.*, vol. 1, no. 1, pp. 36–49, Jan.–Jun. 2013.
[32] A. Amokrane, R. Langar, M. F. Zhani, R. Boutaba, and G. Pujolle, "Greenslater: On satisfying green SLAs in distributed clouds," *IEEE Trans. Netw. Service Manag.*, vol. 12, no. 3, pp. 363–376, Sep. 2015.
[33] *AT&T Domain 2.0 Vision White Paper*. Accessed: Apr. 10, 2019. [Online]. Available: https://www.att.com/Common/about_us/pdf/AT&T Domain2.0 Vision White Paper.pdf
[34] *Network Functions Virtualisation (NFV): Architectural Framework*, document ETSI GS NFV 002, ETSI, Sophia Antipolis, France, 2013.
[35] J. A. Aroca, A. Chatzipapas, A. F. Anta, and V. Mancuso, "A measurement-based characterization of the energy consumption in data center servers," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 12, pp. 2863–2877, Dec. 2015.
[36] S. Rajagopalan, D. Williams, H. Jamjoom, and A. Warfield, "Split/merge: System support for elastic execution in virtual middleboxes," in *Proc. NSDI*, vol. 13, 2013, pp. 227–240.
[37] M. Kablan, A. Alsudais, E. Keller, and F. Le, "Stateless network functions: Breaking the tight coupling of state and processing," in *Proc. USENIX NSDI*, 2017, pp. 97–112.
[38] *Bin Packing Problem*. Accessed: Apr. 10, 2019. [Online]. Available: https://en.wikipedia.org/wiki/Bin_packing_problem
[39] R. E. Korf, "A new algorithm for optimal bin packing," in *Proc. AAAI/IAAI*, 2002, pp. 731–736.
[40] Z. A. Qazi, C.-C. Tu, L. Chiang, R. Miao, V. Sekar, and M. Yu, "SIMPLE-fying middlebox policy enforcement using SDN," in *Proc. ACM SIGCOMM*, 2013, pp. 27–38.
[41] X. Li and C. Qian, "An NFV orchestration framework for interference-free policy enforcement," in *Proc. IEEE ICDCS*, 2016, pp. 649–658.
[42] S. Ayoubi, S. R. Chowdhury, and R. Boutaba, "Breaking service function chains with Khaleesi," in *Proc. IFIP Netw.*, 2018, pp. 64–72.
[43] S. Kumar, M. Tufail, S. Majee, C. Captari, and S. Homma, *Service Function Chaining Use Cases in Data Centers*, IETF, Fremont, CA, USA, 2015.
[44] *pfSense Hardware Sizing Guide*. Accessed: Apr. 10, 2019. [Online]. Available: https://www.pfsense.org/hardware/#sizing
[45] J. Martins *et al.*, "ClickOS and the art of network function virtualization," in *Proc. USENIX NSDI*, 2014, pp. 459–473.
[46] A. Nucci, A. Sridharan, and N. Taft, "The problem of synthetically generating IP traffic matrices: Initial recommendations," *SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 3, pp. 19–32, Jul. 2005.
[47] *Power Efficiency Comparison of Enterprise-Class Blade Servers and Enclosures*. Accessed: Apr. 10, 2019. [Online]. Available: https://www.dell.com/downloads/global/products/pedge/en/BladePowerStudy WhitePaper_08112010_final.pdf
[48] D. Meisner and T. F. Wenisch, "Peak power modeling for data center servers with switched-mode power supplies," in *Proc. ACM/IEEE Int. Symp. Low Power Electron. Design (ISLPED)*, 2010, pp. 319–324.

Md. Faizul Bari (S'12–M'19) received the B.Sc. and M.Sc. degrees in computer science and engineering from the Bangladesh University of Engineering and Technology, and the Ph.D. degree in computer science from the University of Waterloo, where he is a Post-Doctoral Research Fellow with the School of Computer Science. His research interests include network softwarization, SDN, NFV, and cloud computing. He was a recipient of the several scholarships and awards, including the David R. Cheriton Dissertation Award, the Ontario Graduate Scholarship, and the President's Graduate Scholarship at the University of Waterloo.

Shihabur Rahman Chowdhury (S'13) received the B.Sc. degree in computer science and engineering from the Bangladesh University of Engineering and Technology in 2009. He is currently pursuing the Ph.D. degree with the School of Computer Science, University of Waterloo. His research interest includes virtualization and softwarization of computer networks. He was a recipient of several recognitions, including the Best Paper Awards at the IEEE NetSoft 2019 Conference and IEEE/ACM/IFIP CNSM 2017 Conference, and the MITACS Globalink Research Award.

Raouf Boutaba (F'12) received the M.Sc. and Ph.D. degrees in computer science from University Pierre and Marie Curie, Paris, in 1990 and 1994, respectively. He is currently a Professor of computer science and the University Research Chair with the University of Waterloo. His research interests include resource and service management in networks and distributed systems. He was a recipient of the several best paper awards and recognitions, including the Premier's Research Excellence Award, the IEEE ComSoc Hal Sobol Award, the Fred W. Ellersick Award, the Joe LociCero Award, the Dan Stokesbury Award, the Salah Aidarous Award, and the IEEE Canada McNaughton Gold Medal. He is the Founding Editor-in-Chief of the IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT from 2007 to 2010 and on the editorial boards of other journals. He is a fellow of the Royal Society of Canada, the Engineering Institute of Canada, and the Canadian Academy of Engineering.