IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received April 30, 2022, accepted May 22, 2022, date of publication May 27, 2022, date of current version June 3, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3178405

Proactive and Power Efficient Hybrid Virtual Network Embedding: An AWS Cloud Case Study

IKHLASSE HAMZAOUI[1,2,3], BENJAMIN DUTHIL[3,4], VINCENT COURBOULAY[4], AND HICHAM MEDROMI[1,2]

[1]Research Foundation for Development and Innovation in Science and Engineering (FRDISI), Casablanca 16469, Morocco
[2]Engineering Research Laboratory (LRI), System Architecture Team (EAS), National Higher School of Electricity and Mechanic (ENSEM), Hassan II University, Casablanca 20310, Morocco
[3]EIGSI, 17041 La Rochelle, France
[4]IT, Image and Interaction Laboratory (L3I), University of La Rochelle, 17000 La Rochelle, France

Corresponding author: Ikhlasse Hamzaoui (ikhlasse.h12@gmail.com)

ABSTRACT The sharp increase of multimodal cloud resources demand makes it more challenging to design rightsized virtual instances. Inefficient embedding of high sized instances into the substrate resource network has led to numerous resource underutilization issues, which further constitute a key driver to repetitive reallocations of virtual instances. Besides, repetitive reconfigurations of virtual network instances go through a process of intra- or inter-cloud migration that provokes additional increase in power consumption. This paper proposes to solve these mutual challenges through a proactive, power efficient and hybrid Virtual Network Embedding (VNE) approach. We first formulated a Mixed Integer Linear Programming (MILP) model purposing to maximize total power efficiency of intra Data Center (DC) and inter networking resources as a function of EC2 instances requests rates. Leveraging the AWS cloud as a primary case study for this paper, the suggested VNE combines a multi-stage hybrid Virtual Node Embedding (VNoE) policy with an adaptive multistep consolidated Virtual Link Embedding (VLiE). As a starting point, a Green-Location aware - Global Topology Ranking (GLA-GTR) is designed as a primary ranking process suggesting the greenest substrate DCs locations with their related delivery networks. After implementing our proposal on a real AWS backbone network topology, simulation results indicated the efficiency of the proposed VNE approach. The Stacked Denoising Auto Encoders - Bidirectional Gated Recurrent Unit - Resources Vector Matching VNoE (SDAE-BiGRU-RVM VNoE) policy achieved a power decrease of 14.61%, 14.95% and 17.21% compared to BiGRU-RVM-VNoE, BiGRU-BF-VNoE and BF-VNoE policies, respectively. Accordingly, the suggested policy has reached significant power efficiency and overall maximized resource utilization.

INDEX TERMS Proactive hybrid virtual node embedding, multistep virtual link embedding, global topology ranking, power efficiency, carbon emission, AWS cloud.

I. INTRODUCTION

Nowadays, the enormous multimodal cloud traffic, be it processing-intensive, memory-intensive, or storage-intensive, is further complicating the effective management of cloud resources. Poor resource planning management may exhibit repetitive allocation reconfigurations in response to resource underutilization issues and their chaotic demands [1]. Reconfiguring virtual instances provisioning incurs significant power costs, either through virtual instances migration within intra DCs or through inter-cloud migration [2]. These state how neither cloud resources capacity extension nor geo-distributed DCs installation, can keep pace with a non-greedy consumed power of cloud resources. The ''green'' or ''sustainable'' cloud concept is calling for an efficient processing of a wide spectrum of requests, while minimizing resource waste, power consumption and the cost of geo-distributed carbon emissions.

To keep in line with less Operational Expenditures (OPEX) and Capital Expenditures (CAPEX) in future distributed cloud industry [3], combining multi-level virtual resource consolidation in intra DCs and multi-domain virtual network,

The associate editor coordinating the review of this manuscript and approving it for publication was Xujie Li.

becomes one of the promising solutions. This jointure is also favored thanks to the emergence of Software Defined Network (SDN) and Network Function Virtualization (NFV) technologies [4]. In the case of AWS cloud, DCs are distributed following a redundancy hierarchy wherein clusters of DCs from different regions in the world are connected through a global AWS backbone network. From an Inter Cloud point of view, AWS backbone network is managed by the Equinix platform [5] that provides various Amazon Partners Networks (APNs) supplied with automated SD-WAN devices. As for intra AWS DCs nodes, each distributed computing node may hold multiple roles that may be initially grouped as central endpoints computing nodes, CloudFront or cache edge nodes, 5G wavelength nodes and AWS Direct Connect nodes [21]. AWS resources provisioning is therefore a process that mandatorily runs over inter AWS cloud backbone to intra endpoints DCs.

To dig deeper into details of provisioning AWS virtual instances, this process is executed in the Compute Optimizer recommendation engine of each DC. AWS recommendation engine deploys [6] a machine learning facility to select the optimal EC2 instance kind for a particular workload. More precisely, this recommendation process starts with a capture of the last 14 days data on resources utilizations for each AWS user account. Based on analyzed workload characteristics, the recommendation engine identifies resized resources among existing instances groups that best suit the workload requirement. Thereby, the Compute Optimizer projects the behavior of a particular user workload in terms of resource usage and execution time on the recommended instance option. This in turn gives a better insight on how a workload may operate, prior to implementing the recommendations. According to AWS, this Compute Optimizer's recommendations decrease costs by up to 25% [7].

In short, our key contribution resides in proposing a power efficient and hybrid VNE joining a multi-stage consolidated Virtual Node Embedding (VNoE) with a multistep consolidated Virtual Link Embedding (VLiE).

-The proposed hybrid VNoE, denoted as the SDAE-BiGRU-RVM-VNoE, is designed to be run in the AWS Compute Optimizer Engine within each of the four considered instances racks: General Purpose (GP), Compute Optimizer (CO), Memory Optimized (MO), and Storage Optimized (SO).

-Subsequently, the proposed multistep VLiE process is designed to be executed throughout the inter AWS cloud Equinix Fabric Engine. The first step in this process is an Adaptive Yens (AY-KSP) routing algorithm followed by a Best-Fit Spectrum Assignment (BF-SA) algorithm.

-Taken into account the Amazon's ambitious target of fostering green energy within their computing regions, a GLA-GTR algorithm is proposed to be executed as a starting point for the proposed framework. The GLA-GTR is designed as a ranking process recommending the greenest AWS DCs nodes given their computed Carbon Emission Rates (CERs).

The rest of this paper divided into seven sections. Section 2 reviews different VNE models while outlining our proposal's main contributions. Section 3 describes our global system architecture that contextualizes our proposed suggestions. Section 4 formalizes the VNE problem by defining its main parameters, constraints, and decision variables. The proposed flowchart solution design is synthetized in Section 5. Section 6 validates our proposal's in terms of considered performance metrics. Eventually, section 7 concludes this paper and hints at plausible future research perspectives.

II. RELATED WORKS

Elaborating a topical overview in (Table 1), it turns out how VNE techniques have been investigated differently over recent years. It is evident how previous VNE approaches were often treated in a static scheme with no consideration to the highly dynamic workload changes and resource fragmentation induced by incoming and outgoing requests. This shortcoming has been partially handled by probabilistic population-based heuristics then by some reinforced learning frameworks. Besides, these techniques were only tested on a small number of instance requests. Thus, their optimal convergence rates only evolved from the last observed states of the reward functions. To yield more decisive VNE solutions, our proposal bridges existing gaps by these terms:

-Unlike state-of-art studies, we treat the heterotopic aspect of VNE problem by ensuring realistic MILP constraints such as: multi-level embedding restrictions, reported dynamic instances arrival, and various canceled instances rates.

-Contrary to almost adopted ranking processes (table 1), our proposed VNE flowchart solution starts with a ranking process purposing to rank substrate computing and networks nodes not only according to their resources availabilities, but primarily based on their CER values.

-According to ranking classes results, the VNE proposal is constituted from a hybrid proactive VNoE denoted as SDAE-BiGRU-RVM-VNoE, then a multi-step VLiE.

-For each instance provisioning, the proposed hybrid VNoE integrates multivariate time series of EC2 instance attributes that are previously predicted by the SDAE-BiGRU model. Combined to the RVM rule-based allocation policy, a personalized instance size is recommended along with its optimal host's allocation index.

-Subsequently, the proposed multistep VLiE consists of a process that combines an adaptive routing algorithm (AYens-KSP) with a spectrum assignment (BF-SA) algorithm.

-Through this proposal, we aim to maximize total power efficiency of intra and inter AWS network resources, multi-level resources utilizations and geo-distributed carbon emission reduction.

III. SYSTEM DESCRIPTION

The distribution of AWS cloud DCs follows a redundancy hierarchy, in which clusters of DCs from different regions in the world are connected through a global AWS backbone

TABLE 1. Relevant state-of-art VNE techniques.

VNE technique / Year	Objective	Considered resources	Ranking process	Network topologies	Metrics	Limitations
Three-stage based polynomial time VNE heuristic [8] (2018)	Minimize energy consumption of federated SDN networks	CPU and bandwidth	No	GEANT & Nobel-EU topologies from SNDLib	-Energy (J) -Feasible solution ratio	-Abstract design of SDN network. -Static
Delay Sensitive Cross-Domain VNE (DSCD-VNE) (2020) -VLiE: Kruskal minimum spanning tree & Floyd algorithm [9]	Meet different Quality of service (QoS) requirements	CPU and bandwidth	No	GT-ITM tool	-Average time -Acceptance ratio	-Abstract network design -Static
AFBD-VNE -VNoE: Alternative Function Based on Degree (AFBD) ranking values [10] (2021)	Demonstrate the weak distinction capabilities of classical VNE evaluation functions	CPU and bandwidth	Yes	5 topologies from GT-ITM toolkit	-Acceptance ratio -Revenue -Cost-to-Revenue ratio	-Static
Priority-Location-VNE (PL-VNE) (2020) -VNoE: A breadth first search algorithm -VLiE: shortest path method [11]	Address VNE with large virtual resources requirement	CPU and bandwidth	No	BCube	-Average throughput -Average latency -Acceptance ratio -Packet hops	-Static
DVSDNE: Multi agent Distributed VNE for SDNs (4 agents) (2021) [12] -Multilevel k-way partitioning algorithm -VNoE: node ranking based on resources availabilities (minimum merit value first)	Ensure VNE problem for non-overlapping smaller substrate network partitions	CPU and bandwidth	Yes	Networkx tool	-Latency -Acceptance ratio -Revenue to cost ratio	-Static
VNE-Metamodel for virtual networks, substrate networks, and element mappings (2021) -ILP, Object Constraint Language [13]	Minimize the aggregated communication costs	-Switch -Bandwidth -Server (CPU, memory, storage).	No	Star topology	-Cost -Running time	-Static
Global Topology Ranking-VNE (GTR-VNE) Ant Colony Optimization-VNE (ANC-VNE) (2018) [14]	Minimize the total consumed power in software-defined optical data center networks	-DC computing -Optical network components	Yes	6 nodes and NSFNET	-Power in KW -Number of active DCs -Running time -Acceptance ratio	-Slow probabilistic convergence rate depending on random input solution
GA-VNE (2021) Genetic algorithm [15]	Improve network performance& reduce embedding process cost	CPU and bandwidth	No	Random	-Average total cost -Average embedding time	-Slow probabilistic convergence rate
GA-based distributed parallel VNE algorithm for link embedding stage (2021) [16] -Initial solution pool generated with Dijkstra's algorithm	Reduce VNE operations time	CPU and bandwidth	Yes	GT-ITM	-Revenue -Acceptance ratio -Revenue to cost ratio -Average cost -Average links utilization.	-High convergence speed under workload variations

TABLE 1. *(Continued.)* **Relevant state-of-art VNE techniques.**

VNE-SA-PSO Simulated annealing for initial solution pool generation and Particle Swarm Optimization for VNE process (2020) [17] - Linear differential decline strategy to calculate the inertia weight of PSO	Rationalize physical resources utilization to a higher degree	CPU and bandwidth	No	GT-ITM	-Embedding cost -Acceptance ratio -Revenue- to-cost-ratio -Running time	Varying convergence speeds depending on requests fluctuations
MUVINE (2020) [18] -Binary SVM classifier: (rejected / accepted VNs), -Maximum Likelihood Classifier (MLC): VMs features -Iterative SARSA RL for VNoE - Dijkstra algorithm for VLiE	Foresee future resource demand as fundamental shortcomings of traditional VNE	CPU Memory Bandwidth	No	Switch-centric network topology	-Average accuracy -Average resources allocation -Time	multistage costly solution
Full Information Disclosure (FID) and Limited Information Disclosure (LID) heuristics 2020 [19] 4 RL environments	Guarantee total privacy to both ISPs & customer	CPU and bandwidth	No	No topology	- Embedding cost -Data overhead	Abstract design of ISPs networks.
- Graph convolutional networks for network spatial features extraction - Asynchronous Advantage Actor-Critic (A3C) algorithm for training embedding policies (2021) [20][21]	Automatically detecting network status and dynamically providing solutions	CPU and bandwidth	No	CSTNET topology	-Acceptance Ratio -Long-term Average Revenue -Running time	Complex and time-consuming graph modeling
-RNN based seq2ses Encoder/Decoder to extract information -Policy Gradient VNE algorithm [22] (2020)	Obtain sufficient embedding data using unsupervised learning	CPU and bandwidth	No	GT-ITM tool	-Revenue to cost ratio -Long term average revenue -Long term acceptance ratio.	No sufficient learned representati ve information
Stacked Denoising Auto Encoder-Bidirectional Gated Recurrent-VNE (SDAE-BiGRU-RVM-VNE) (*)	Time series based VNE formulation that substantially exploit noisy historical substrate resources states	CPU Memory Storage Network bandwidth	Yes	AWS backbone network	-Total power efficiency -Acceptance ratio -CER ratio -Multilevel resources utilization	Communica tion overhead

network. Within a region, there exists a cluster of Availability Zones (AZs). An AZ is in turn represented by a cluster of closely spaced DCs, designed to minimize the risk of any downtime and ensure the highest availability of services [23].

AWS computing nodes located in different regions, or within the same region, and perhaps within an availability zone, can play various roles. We can initially group these roles as central endpoints computing nodes, CloudFront or cache edge nodes, 5G wavelength nodes and AWS Direct Connect nodes [24]. AWS resources provisioning by multi-tenant identified users, is a process that mandatorily runs through the inter AWS cloud backbone to the intra end-points DCs. Figure 1 describes the three phases that EC2 instances provisioning goes through.

A. AWS RESOURCES PROVISIONING ENGINE

In this phase, identified users receive the AWS MarketPlace catalog according to their specified region. Using Cloud-Formation, a user may choose either one Amazon Machine Image (AMI) to launch their specified EC2 instances with the same configuration or multiple AMIs when instances with divers configurations are needed. For more details about required EC2 instances, a user needs to specify first the required instance family type, then a specific instance name and size with dedicated vCPU, memory, storage, and network virtual resource capacities. In addition, a user must create a Virtual Private Network (VPN) connecting it to its instance infrastructure. Accessing resources from multiple regions mandates authorization checks through the AWS Services

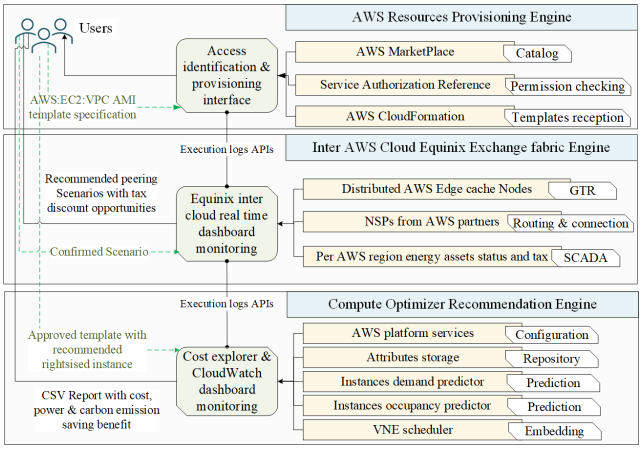

FIGURE 1. System description.

Authorization Reference. So far, all mentioned and coming users' actions are executed via APIs calls.

B. INTER AWS CLOUD EQUINIX EXCHANGE FABRIC ENGINE

In this phase, distributed AWS cache nodes are associated to AWS endpoints DCs nodes through a backbone network designed by popular AWS Partners Networks (APNs) along worldwide 1/2/3 Tier telecom. More precisely, the provided APN choices in Equinix platform includes Cisco, Juniper, Vmware, PaloNetworks, Aruba, Ckeck Point, Fortinet and Versa [25]. These entire APNs are supplied with automated SD-WAN devices connecting a single Hop-to-Hop (H2H) IP and optical network layer [26]. Each APN deploys its own routing strategies, policies, and priority rules. In this paper, we propose a multistep adaptive routing and spectrum assignment strategy for embedding virtual connections (Section 5). In addition, we assign another function to distributed AWS edge cache nodes, namely the Global intra and inter cloud resources Topology Ranking (GTR). Indeed, each edge cache node may recommend two peering connection scenarios to the identified users: a green aware scenario with tax discount opportunities and a delay-sensitive one. When choosing a green-aware scenario, a proposed GLA-GTR purposes to find the greenest substrates links and endpoints nodes (See more details in Section 5). For this target, we assume that distributed AWS cache nodes are informed about legislation status of AWS Renewable Energy (RE) assets via Equinix Portal. It is then up to users to confirm the peering connection scenario that best suit them.

C. COMPUTE OPTIMIZER RECOMMENDATION ENGINE

In this last engine, we deploy the proposed instances demand and resources occupancies predictor associated to the proposed proactive and hybrid VNE scheduler. This is intended to perform an early recommendation of rightsized instances with their embedding. Using predicted results, the VNE scheduler is executed in an offline fashion in order to compute anticipated cost, power, and carbon emission saving opportunities that may be achieved when recommending a downsized personalized instance. Depending on each instances' family historical occupancies and users behaviors, optimal rightsized EC2 instances are recommended in an early stage prior to instances underutilization occurrence. The recommendation benefits are sent to identified user through a CSV report for template confirmation. Once the recommended personalized template is validated, the proposed VNE scheduler is executed in an online fashion to link user to its endpoint infrastructure and associate a unique Amazon Resource Names (ARNs) for the identified template. An ARN is a combination of subnet regions partition in which resources are located, the AWS product, then user and resources IDs (See more details in Section 5).

IV. PROBLEM FORMULATION

Our model's parameters and variables are mostly dependent on AWS regions, seasonality time slots, EC2 instances' family types, then usage behaviors and provisioning natures. They include both discrete and continuous data and are noted as follows:

Parameters and Variables:

-$t \in T$: the set of time slots.

-$n \in N$: the set of AWS endpoint DCs nodes.

-$net \in Net$: the set of AWS APN nodes.

-$e \in E$: the set of AWS edge cache nodes.

-$s \in S_c$: the set of a rack servers.

-$c \in C_n$: the set of family racks in a DC.

-$AWS^u : EC2^k_{i,c} : VPC^t_p$: the template specification.

-$u \in U$: the set of identified users in a template.

-$i \in I$: the set of required EC2 instances in templates.

-$k \in \{1: dedicated/2: shared\}$: the tenancy type that might be either shared among many users or dedicated to one single user.

-$\rho \in \{1: green-aware/2: delay-sensitive\}$: the chosen peering connection scenario.

-$Loc(\cdot)$: the location region of a certain node $(\cdot)$.

-$CI_{Loc(\cdot)}$: the carbon intensity at energy assets locations.

-$P_{Static^{s,c}}$: the idle server's consumed power.

-$vCPU_i$, $vMemory_i$, $vStorage_i$: virtual CPU, memory and storage resources dedicated to an EC2 instance i.

-$UvCPU^t_i$, $UvMemory^t_i$, $UvStorage^t_i$: the utilized resources of an instance i.

-$ACPU^t_{s,c}(n)$, $AMemory^t_{s,c}(n)$, $AStorage^t_{s,c}(n)$: a server's capacities.

-$h \in H^{R,t}_{(e \to n)}$: the set of forwarded hops from a source cache node to endpoint DC, following a path R.

-$f^{o,t}_h$ = staring slot index of an optical spectrum o in the hop fiber link h.

-$W^{o,t}_h \in W$ = number of frequency slots dedicated to the optical spectrum o in the hop fiber link h.

-$G^{o,t}_h$ the guar band slot index of the optical spectrum o in the hop fiber link h.

-$p \in P$: the set of optical segments constituting a routing path R connecting the source edge cache node with the endpoint Virtual Private Cloud (VPC).

-(j, j'): two regenerated vertices delimiting an optical segment.

-C: a slot bandwidth capacity.

-$ESD^{Solar,T}_{n/net}$ /$ESD^{Wind,T}_{n/net}$: energy storage devices for solar and wind energies in a DC or APN node.

-$RS^{T-1}_{n/net}$/$RW^{T-1}_{n/net}$: remaining solar and wind energies from previous time interval.

Binary Decision Variables:

- $\gamma^{S,T}_{n/l}$ /$\gamma^{W,T}_{n/l}$: if DC or APN node's PDU respectively provide sufficient solar / wind energy at T.

- $\lambda^{i,t}_{s,c}(n)$: if required instance i is assigned to the server s in a cluster c within DC n.

- $\xi^{i,t}_{o,w,h}(R)$: $= 1$ if the provisioned instance i is assigned an optical spectrum channel o with a slot number w in the path R.

Constraints Formulation: The following introduced constraints represent the three main components of the studied system, i.e., the ranking process, the intra virtual nodes embedding and the inter virtual links embedding.

As a starting point, the proposed GLA-GTR ranking process aims to find the greenest substrate nodes and links. The corresponding ranking is supported by the Carbon Emission Rate (CER) computation, which relies on REs availabilities within DC or APN sites as well as the specific carbon intensities of their REs assets. The CER value is determined as follows, where C_R is a capacity regulator coefficient:

$$\begin{aligned} CER_{n/net} = [&(CI_{Loc(wa)} \cdot ESD^{Wind,T}_{n/net} \cdot \gamma^{W,T}_{n/net}) \\ &+ (CI_{Loc(sa)} \cdot ESD^{Solar,T}_{n/net} \cdot \gamma^{S,T}_{n/net})] \times C_R \\ &+ [CI_{Loc(ba)} \cdot AB^{T}_{n/net}] \cdot \gamma^{B,T}_{n/net} \end{aligned} \quad (1)$$

Each DC or APN node's related Power Distribution Unit (PDU) should provide only one type of energy per hourly interval.

$$\sum_{en=1}^{En} \gamma^{en,T}_{n/net} = 1 \quad (2)$$

Normally, the charging and discharging efficiencies of REs are often about 85%∼95% [27]. We set respectively $\alpha = 10\%$ and $\beta = 10\%$ as their energy losses, while the self-discharging energy loss is set to $\theta = 0.2\%$ [27]. The Purchased Active Solar (PAS) and Wind (PAW) energies by a DC or APN node during an interval T, are used to derive the Active Solar and Wind energies, as shown in (3-4):

$$AS^{T}_{n/net} = PAS^{T}_{n/net}(1 - \alpha - \beta - \theta) \quad (3)$$

$$AW^{T}_{n/net} = PAW^{T}_{n/net}(1 - \alpha - \beta - \theta) \quad (4)$$

At T_0, both solar and wind Energy Storage Devices (ESDs) within DC and APN nodes receive previous active energies that should be equivalent to their capacities.

$$ESD^{Solar,T_0}_{n/net} = AS^{T_0}_{n/net} = ESD^{Solar,Max}_{n/net} \quad (5)$$

$$ESD^{Wind,T_0}_{n/net} = AW^{T_0}_{n/net} = ESD^{Wind,Max}_{n/net} \quad (6)$$

In the following time intervals T, both ESDs types contain the remaining energies from the last time interval $T-1$ along with the available active energies received during the current time interval.

$$ESD^{Solar,T}_{n/net} = AS^{T}_{n/net} + RS^{T-1}_{n/net} \leq ESD^{Solar,Max}_{n/net} \quad (7)$$

$$ESD^{Wind,T}_{n/net} = AW^{T}_{n/net} + RW^{T-1}_{n/net} \leq ESD^{Wind,Max}_{n/net} \quad (8)$$

Remaining solar or wind energies in a DC related ESDs, are represented respectively by equations (9-10). δ^{T-1}_n is the efficiency rate of the cooling devices and fans, which varies in accordance with the computing loads of DC's servers on various racks. P^{T-1}_n is the total consumed power consumption of servers during previous time interval.

$$RS^{T-1}_{n} = AS^{T-1}_{n} - [(P^{T-1}_{n} + \delta^{T-1}_{n} AS^{T-1}_{n}) \cdot \gamma^{S,T-1}_{n}] \quad (9)$$

$$RW^{T-1}_{n} = AW^{T-1}_{n} - [(P^{T-1}_{n} + \delta^{T-1}_{n} AW^{T-1}_{n}) \cdot \gamma^{W,T-1}_{n}] \quad (10)$$

Remaining solar and wind energies in an APN related ESDs are represented respectively by equations (11-12).

$$RS^{T-1}_{net} = AS^{T-1}_{net} - (P^{T-1}_{net} \cdot \gamma^{S,T-1}_{net}) \quad (11)$$

$$RW^{T-1}_{net} = AW^{T-1}_{net} - (P^{T-1}_{net} \cdot \gamma^{W,T-1}_{net}) \quad (12)$$

Concerning intra virtual nodes embedding, a user must specify a template containing: the required EC2 instance name; the instanc's family typ; the tenancy typ; and the required VPC subnet with a preferred peering connection scenario. In this study, four types of instances' families corresponding to four computing racks are considered and namely: the compute-optimized, the memory-optimized, the storage-optimized and the general-purpose instances families.

The assigned instances provisioning templates shall not exceed available resources in the concerned DC within the involved time slot.

$$\sum_{i=1}^{I} (AWS^u : EC2^k_{i,c} : VPC^t_\rho \cdot vCPU_i) \times \lambda^{i,t}_{s,c}(n) \prec ACPU^t_{s,c}(n) \quad (13)$$

$$\sum_{i=1}^{I} (AWS^u : EC2^k_{i,c} : VPC^t_\rho \cdot vMemory_i) \times \lambda^{i,t}_{s,c}(n) \prec AMemory^t_{s,c}(n) \quad (14)$$

$$\sum_{i=1}^{I} (AWS^u : EC2^k_{i,c} : VPC^t_\rho \cdot vStorage_i) \times \lambda^{i,t}_{s,c}(n) \prec AStorage^t_{s,c}(n) \quad (15)$$

There should be only one instance request per template and that instance should not be assigned to more than one host and cluster.

$$\sum_{i=1}^{I} (AWS^u : EC2^k_{i,c} : VPC^t_\rho) = 1 \quad (16)$$

$$\sum_{s=1}^{S_c} (AWS^u : EC2^k_{i,c} : VPC^t_\rho) \times \lambda^{i,t}_{s,c}(n) = 1 \quad (17)$$

Since users only pay for resources used during their active time, nothing is paid during passive time even though inactive instances still consume a large amount of energy. Consequently, a user's instance may be fully terminated until the end of his agreement. Nevertheless, we adopt in this paper an instance stopping process regarding identified users with ''dedicated'' tenancy type. An assigned instance with ''dedicated'' tenancy type must be terminated once its user provisions new template (18). This process is not applicable to users with a ''shared'' tenancy type since they may provision various template at a same time.

$$\sum_{i=1}^{I} (AWS^{u} : EC2^{k}_{i,c} : VPC^{t}_{\rho}) \times \lambda^{i,t}_{s,c}(n) = 1, \quad \text{if } k = dedicated \tag{18}$$

A server power consumption may be formulated as shown in (19). Therefore, the consumed power of a DC node at a certain time slot is mentioned in (20)

$$\begin{aligned} P^{t}_{s,c} &= P_{Static^{s,c}} + (P_{CPU_{s,c}} \cdot \sum_{i=1}^{I} U^{t}_{vCPU_{i,s}} \cdot \lambda^{i,t}_{s,c}) \\ &+ (P_{Memory_{s,c}} \cdot \sum_{i=1}^{I} U^{t}_{Memory_{i,s}} \cdot \lambda^{i,t}_{s,c}) \\ &+ (P_{Disk_{s,c}} \cdot \sum_{i=1}^{I} U^{t}_{Disk_{i,s}} \cdot \lambda^{i,t}_{s,c}) \end{aligned} \tag{19}$$

$$P^{t}_{DC}(n) = \sum_{c=1}^{C_n} \sum_{s=1}^{S_c} P^{t}_{s,c} \tag{20}$$

Embedding a customer dedicated VPN consists of peering its virtual connection in the inter backbone network, with a specified bandwidth depending on the provisioned instance template. In practice, a peering path may be constituted of one-to-many optical segments between regenerated network nodes (routers) (21). Each optical segment can be composed from one-to-many hops (links). No matter which routing hop is followed, a virtual connection should be assigned to an optical spectrum channel with a number of slots that satisfy the connection bandwidth (22).

$$H^{R,t}_{(e\to n)} = \sum_{p=1}^{P} H^{p,t}_{(j\to j')} \tag{21}$$

$$\begin{aligned} &AWS^{u} : EC2^{k}_{i,c} : VPC^{t}_{p} \cdot vBandwidth_{i} \\ &\leq \xi^{i,t}_{o,w,h}(R) \cdot \sum_{w\in W} W^{o,t}_{h} \cdot C \quad \forall h \in H^{R,t}_{(e\to n)} \end{aligned} \tag{22}$$

A virtual instance connection may be assigned only one optical spectrum per hop fiber link and their associated slots are not assigned to any other optical spectrum (23).

$$\sum_{o=1}^{O} \xi^{i,t}_{o,w,h}(R) = 1, \quad \forall h \in H^{R,t}_{(e\to n)} \cup \forall w \in W^{o,t}_{h} \tag{23}$$

Again, each assigned optical spectrum should be ended by an adjacent guar band slot to avoid any overlap between virtual connections (24).

$$f^{o,t}_{h} = W^{o,t}_{h} - G^{o,t}_{h}, \quad \forall h \in H^{R,t}_{(e\to n)}, \text{ where } G^{o,t}_{h} \leq W \tag{24}$$

Considering an optical segment between two regenerated routers (j, j'), the vertical slots continuity constraint (25) indicates how the assigned optical spectrum shall be the same along forwarded hop fiber links constituting this segment. In addition, the horizontal slots consecutiveness constraint mandates that sub-curries (slots) deployed over an optical spectrum must be consecutive in the frequency domain (26).

$$f^{o,t}_{h} \cdot \xi^{i,t}_{o,w,h}(R) = f^{o,t}_{h+1} \cdot \xi^{i,t}_{o,w,h+1}(R), \quad \forall h \in H^{p,t}_{(j\to j')} \tag{25}$$

$$\begin{aligned} &- |S| \cdot (\xi^{i,t}_{o,w,h}(R) - \xi^{i,t}_{o,w+1,h}(R) - 1) \\ &\geq \sum_{w\in[w+2,|W|]} \xi^{i,t}_{o,w,h}(R) \end{aligned} \tag{26}$$

The power consumption of network is formulated in (27), where refers to the typical power consumption of the router line card per Gbps.

$$P^{R,t}_{net(e\leftrightarrow n)} = \sum_{h=1}^{H^{R,t}_{(e\to n)}} \left[\sum_{o=1}^{O} (\xi^{i,t}_{o,w,h}(R) \cdot W^{o,t}_{h} \cdot C) \cdot P_{R} \right] \tag{27}$$

Objective Function: The objective function of the proposed MILP model is intended to maximize the total power efficiency, which involves intra AWS DCs nodes power efficiency and inter network power efficiency (28-30).

$$Max : TP_{eff} = P_{eff,DC}(n) + P_{eff,net(e\leftrightarrow n)} \tag{28}$$

$$P_{eff,DC(n)} = \frac{\sum_{c=1}^{C_n} \sum_{s=1}^{S_c} \sum_{i=1}^{I} (AWS^{u} : EC2^{k}_{i,c} : VPC^{t}_{p} \cdot \lambda^{i,t}_{s,c}(n))}{P_{DC}(n)} \tag{29}$$

$$P_{eff,net(e\leftrightarrow n)} = \frac{\sum_{h=1}^{H^{R,t}_{(e\to n)}} \sum_{i=1}^{I} (AWS^{u} : EC2^{k}_{i,c} : VPC^{t}_{p} \cdot \xi^{i,t}_{o,w,h}(R))}{P^{R,t}_{net(e\leftrightarrow n)}} \tag{30}$$

V. FLOWCHART SOLUTION DESIGN

The proposed flowchart depicted in (Figure 2) demonstrates the three main phases through which EC2 templates provisioning runs through. The first stage is a GTR Process followed by a two-stage VNE.

A. GREEN LOCATION AWARE-GLOBAL TOPOLOGY RANKING (GLA-GTR) STEP

The proposed GLA-GTR is designed to be performed at each AWS edge cache node to recommend the greenest endpoints substrate nodes with their related delivery APN.

Recommendations are released under three ranking classes. The first class Q1 englobes substrate nodes with

their associated network paths that have lower CER values *(Equation 1)* than their corresponding fixed thresholds (i.e., $CER_n < 4.73$ & $CER_{net} < 3.76$).

If this complete condition is not met, the algorithm extends the ranking process based on power consumptions metrics *(Equation 20 & 27)*. The second class Q2 incorporates then substrate nodes that are compensating the brown power of their related APNs nodes (i.e., $CER_{net} > 3.76$ & $P_{DC} > P_{net}$). Finally, the third class Q3 includes substrate nodes with their associated network paths that have higher CER values *(Equation 1)* than their corresponding fixed thresholds.

B. VIRTUAL NODE EMBEDDING (VNoE) STEP

The proposed proactive and hybrid VNoE is designed to be executed within the compute optimizer recommendation engine in each instances family rack. Combining, as input, future predicted template demand and resources occupancies with current received provisioning template, the VNoE may be performed using certain predefined rule-based policy.

In this study, future predicted templates inputs are released by the SDAE-BiGRU model. Therefore, the proposed VNoE adopt an RVM rule-based policy. As demonstrated in Figure 2, the proposed SDAE-BiGRU-RVM is a VNoE policy purposing to maximize the power efficiency of multilevel resources within entire instances family racks. More precisely, this policy makes use of predicted provisioning template to be received at $t+1$, current received provisioning template request at time t, and resources status of hosts inside a rack. The first step of this policy consists of checking the necessary instance shutdown, as stipulated in constraint (18), and thereby performing appropriate resources status updates. Next, the active hosts are sorted in ascending order of the dominant resource type in an instance family. For instance, in GP and CO families, active hosts are sorted in ascending order of available vCPU values. On the other side, active hosts are sorted in ascending order of available memory and storage values respectively in MO and SO families. The main target behind this sorting scheme is to maintain a multilevel resources consolidation. The second fundamental principle underlying this allocation policy consists of reserving the least available host resources to the provisioning template having the most similar required resources. The computed similarity in the RVM rule-based policy is performed using the Cosine function. As highlighted in Figure 2, the Cosine function was computed once between available resources and current received template required resources, then between available resources and next arrived template required resources (31), as shown at the bottom of the page.

If the similarity function between current template provisioning request and the hosts' least available resources is greater than the similarity function between next predicted provisioning request and hosts' least available resources, the host's least available resources will be dedicated to the next coming template. This implies that current template request will be assigned to the next active host index. At the end, the VNoE outputs the instance template allocation index, with its resulting servers consumed power, acceptance ratio and last resources status updates.

The key VNoE benchmark policies that are tested in this study are:

-The proposed SDAE-BiGRU-RVM: a proactive VNoE policy deploying multivariate instances attributes predictions using the SDAE-BiGRU model, combined to the RVM rule-based allocation policy.

-BiGRU-RVM: a proactive VNoE policy deploying the univariate instance demand predictions using the BiGRU model, combined to the RVM rule-based allocation policy.

-SDAE-BiGRU-BF: a proactive VNoE policy deploying the multivariate instances attributes predictions using the SDAE-BiGRU model, combined to the Best Fit (BF) rule-based allocation policy.

-BiGRU-BF: a proactive VNoE policy deploying the univariate instance demand predictions using the BiGRU model, combined to the Best Fit (BF) rule-based allocation policy. *-BF:* a reactive VNoE policy that does not deploy any prediction engine.

C. VIRTUAL LINK EMBEDDING (VLiE) STEP

Once the allocation index of the DC node is specified, the next multistep VLiE process is designed to be executed throughout the inter AWS cloud Equinix Fabric Engine.

The first step in this process is an adaptive Yens K-Shortest Path (KSP) algorithm which alternates between two SP routing functions for finding the initial SP. Yens algorithm outputs K required SP that are not necessarily completely different, thus may share a common edge link. In order to avoid any links premature blocking issue, while compromising link adaptability and execution time, the first initial SP in the Yens algorithm switches between two SP functions according to links loads. As a starting point, initial SP is determined by the Breath-First-Search function to find the first minimum hop path. Once the load of at least one edge link becomes greater than a predefined fixed threshold, the previous SP function is replaced by the Dijkstra function to find the first minimum load path. The sorted KSP are

$$Sim_{Cosine}(\text{Current}_i/\text{Predicted}_i, \text{available}_s) = \frac{(UvCPU_i \cdot AvCPU_s) + (UMemory_i \cdot AMemory_s) + (UStorage_i \cdot AStorage_s)}{\sqrt{UvCPU_i^2 + UMemory_i^2 + UStorage_i^2} \cdot \sqrt{AvCPU_s^2 + AMemory_s^2 + AStorage_s^2}} \quad (31)$$

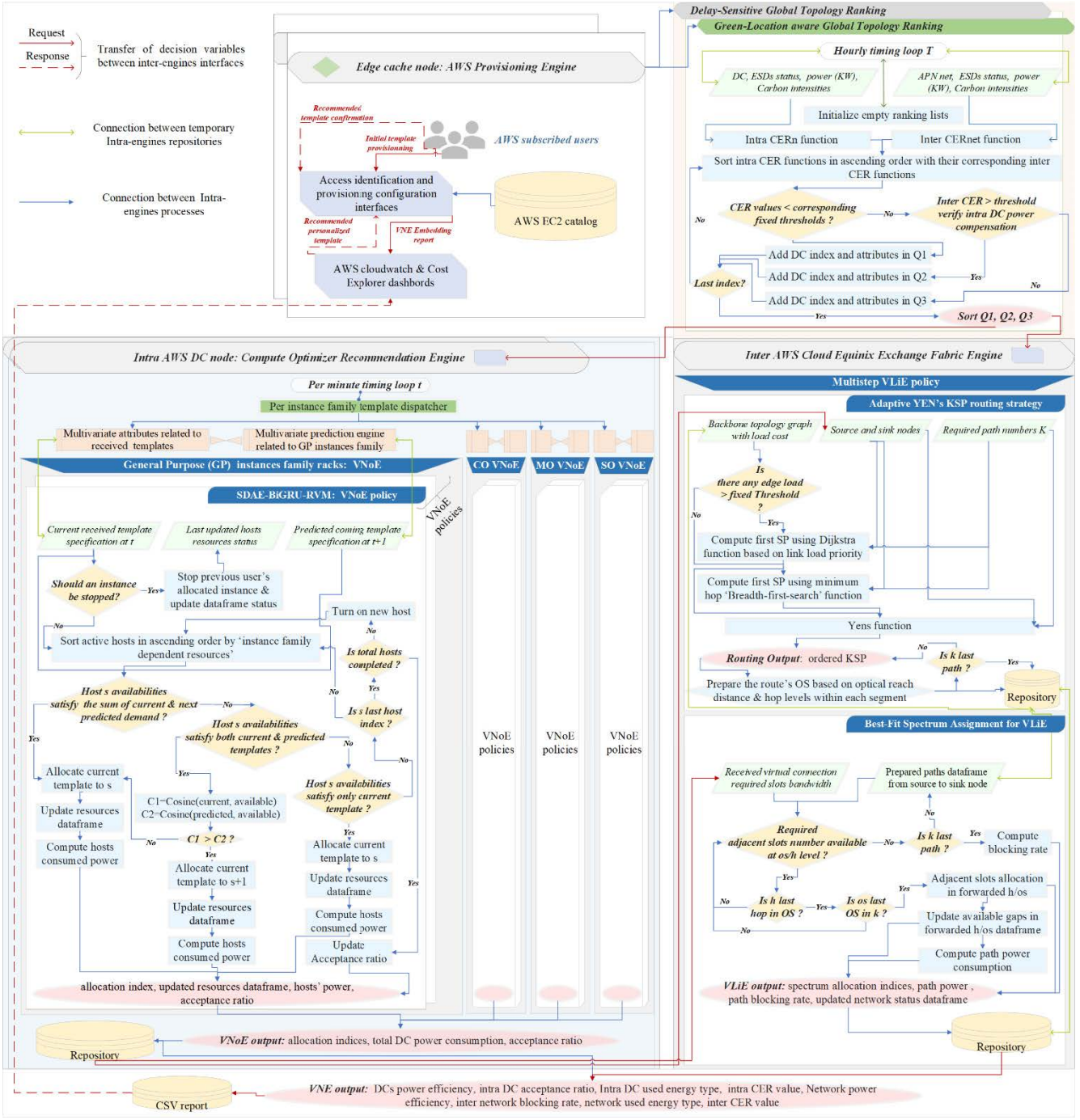

FIGURE 2. **The proposed flowchart solution design.**

successively deployed by the Best-Fit spectrum assignment stage, where received instance's virtual connection bandwidth is assigned a spectrum with the required slots number (Figure 2).

Likewise, after each connection assignment, the consumed network power is sorted with the allocated spectrum index, the resource state update, and the achieved blocking rate.

Once both VNE steps achieved, the VNE outputs the total power efficiency resulting from current embedding status, and entire resources utilizations.

VI. EXPERIMENTATION AND RESULTS

In this section, we provide separately an overview on:

-Simulation setups and supported hypothesis in subsection A;

-Instances input and energy input data in subsection B;
-Simulation results in subsection C.

A. SIMULATION SETUPS AND SUPPORTED HYPOTHESIS

All experiment scenarios were coded in Python 3.8 under Jupyter notebook using the NVIDIA GeForce RTX2070 GPU and 16 Go of RAM on an intel core 7 DELL G5-15 machine. The proposed architecture was restricted to AWS European and UK endpoints DCs (Figure 3) and simulated under the global AWS backbone network topology designed in (Figure 4).

We validated the proposed system under the following supported hypothesis:

-Initial networks and servers capacities are supposed to be unutilized and fully available at the first experiment.

-At the first-time interval, ESD status at DCs nodes and APNs nodes are equivalent to the received active energies. The remaining energies in ESD devices start to be computed after the first-time interval.

-Each Power Distribution Unit (PDU) of a DC or an APN node are supposed to provide only one type of energy per hourly interval, while prioritizing the greenest ones according to their carbon intensities.

-For more concise results regarding the proposed ranking system, the green-aware and delay-sensitive scenarios are treated separately since no chosen peering connection attribute is available in the deployed dataset.

-For the sake of simplicity, instance provisioning requests come from a single source.

As shown in Table 2, we sought to keep the configuration of DCs and hosts parameters setup as close as possible to the actual configuration of the AWS deployment, initially in terms of geographical locations as depicted in Figure 3, then in terms of computing capacities per various instances families (Table 2). Each family has a group of EC2 instances with a unique name. In addition, each instance could be available in different predefined sizes: starting from medium, large, to x/2x/4x/. . . /24xlarge dimensions. Table 3 provides characteristics values of deployed network resources.

B. AWS ENEGRY INPUT DATA AND EC2 INSTANCES INPUT DATA

For each computing or APN node within the AWS backbone network (Figure 3), data regarding hourly available renewable energies was gathered using the ElectricityMap platform [28]. As previously mentioned, we focused on UK and Europe countries given Amazon's ambitious target of fostering green energy in these regions.

We also leveraged the International Energy Agency (IEA) platform [29], to gain insight into dominant backup energy types within each region with their related carbon intensities depicted in (Figure 5). Furthermore, we proceeded with the following steps to get more tangible data:

-For each AWS solar and wind power plant (Figure 2), we estimated hourly energy availabilities using AWS plant's

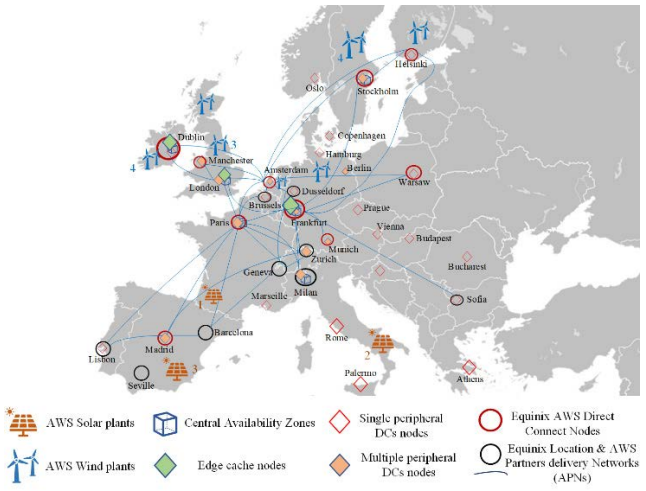

FIGURE 3. Simulated AWS nodes categories in the UK and Europe.

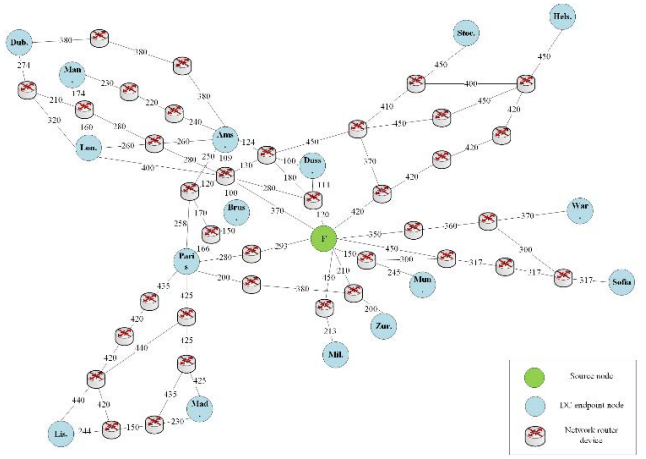

FIGURE 4. The simulated AWS backbone network.

installed capacities and occupied capacities percentages obtained from ElectricityMap platform.

-Obtained values were converted into KW unit for each AWS power plant.

-The hourly energy availabilities were then fairly distributed to the central and peripheral DC nodes and APNs nodes taking into account their numbers and ESDs capacities.

Regarding entered instances data, each instances family involves the following fourteen multivariate attributes: time attribute; required instance name and size; its related vCPU, memory, storage, and network capacities; user ID; a binary novelty attribute; tenancy; utilized vCPU, memory & storage resources; then next predicted instances demand and resources occupancies. The established future hourly intervals predictions were ensured by the BiGRU model and the SDAE-BiGRU for respectively instances demand prediction and multivariate instances resources occupancies prediction. In this paper, we worked with five hourly time intervals each containing 60 instances from various families.

TABLE 2. DCs and servers configurations.

Considered types of cluster/racks in all endpoints DCs nodes			2 general-purposes racks, 2 compute-optimized racks, 2 memory-optimized racks, 2 storage-optimized racks					
Number of servers per rack in central computing nodes			10					
Number of servers per rack in peripheral computing nodes			5					
Cluster/ rack types	**Server type and idle power in W**	**vCPU**	**Dynamic vCPU power W/vCPU**	**Memory Capacity in GB**	**Dynamic memory power W/GB**	**Storage Capacity in GB**	**Dynamic disk power W/GB**	**Network Capacity in Gbps**
General-purpose M5a, M5ad	AWS Graviton2 (2.5 GHz), NVMe SSD, 43 W / 11 W / 3 W	>64	0,97	>274.88	0,34	>2576.98	0,009	<50
Compute Optimized C5n, C6i	AWS Graviton2 (3.5 GHz), NVMe SSD, 60 W / 13,1 W / 0,19 W	>12	0,69	>274.88	0,33	>150	0,009	>100
Memory-Optimized R5dn, R5n	AWS Graviton2 (3.1 GHz), NVMe SSD 53 W/ 75,13 W / 8,7 W	>96	0,81	>824.63	0,32	>3865.47	0,008	>200
Storage-Optimized i3, i3en	AWS Graviton2 (3.1 GHz), NVMe SSD, 53 W/ 101,43 W / 180 W	>96	0,81	>824.63	0,28	>60000	0,007	>200

C. SIMULATION RESULTS

This section is dedicated to numerically evaluate our proposal by comparing VNoE and VLiE policies and peering scenarios proposed in Section 5. The performance evaluation is conducted according to the following metrics:

-*Power consumption*: evaluates intra DCs and inter network consumed power in (W) using respectively equations (20 & 27).

-Power efficiency: evaluates the total power efficiency resulting from intra DCs and inter network, in divers five experimented time intervals.

-*Multi level resources utilizations*: refer to hosts' occupied vCPU, memory and disk given each allocation policy.

-*Acceptance ratio*: represents the ratio of all completed instances on the total number of received instances.

-*CER values*: the total Carbon Emission Rate metric mentioned in equation (1), related to both considered peering scenarios.

Through intra DCs power consumptions summarized in (Figure 6), it is evident how the proposed SDAE-BiGRU-RVM-VNoE and the SDAE-BiGRU-BF-VNoE approaches outperformed other allocation policies given their anticipated personalized downsized instances allocation. More precisely, the proposed SDAE-BiGRU-RVM-VNoE policy achieved a power decrease of 14.61%, 14.95% and 17.21% respectively compared to BiGRU-RVM-VNoE, BiGRU-BF-VNoE and BF-VNoE. Again, the SDAE-BiGRU-BF-VNoE achieved a power decrease of 14%, 14.35% and 16.62% respectively compared to BiGRU-RVM-VNoE, BiGRU-BF-VNoE and BF-VNoE.

This mutual decrease in power related to the two proactive VNoE approaches is also related to the relatively smaller number of their occupied servers compared to the other policies. Figure 7 illustrates the total occupied servers number related to each VNoE allocation policy, in five-time intervals with divers instances inputs. Taken as an example the second time intervals, the total number of occupied servers in entire family racks is 28 for both SDAE-BiGRU-RVM-VNoE and SDAE-BiGRU-BF-VNoE policies, as compared to 39 occupied servers in other allocation approaches.

In the other hand, network power consumptions (Figure 8) are mainly dependent on network throughput rates requested by the arriving instances within a time interval, the routing congestion rates and, consequently, the transmitted network

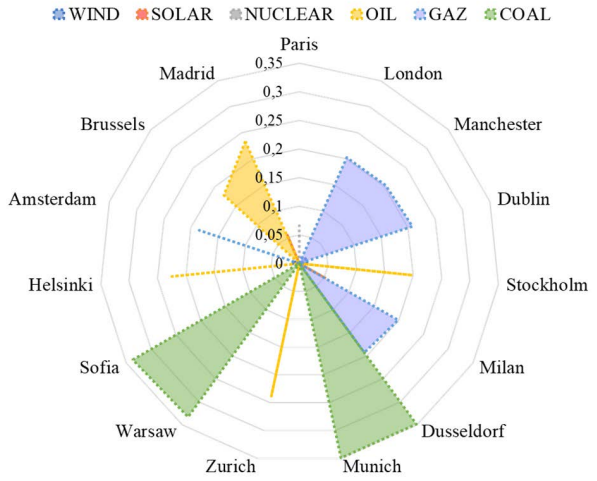

FIGURE 5. Multi energy sites related carbon intensities in KgeqCO2/KWh.

TABLE 3. Network configuration.

Network parameters	Values
Router line card power consumption per Gbps	0.16
Single line card port capacity in Gbps	400
Assumed active ports per router line card	14
Estimated optical reach distance in Km	500
Number of sliced slots per hop fiber link	448
Slot frequency in Ghz	6.4
Slot bandwidth capacity in Gbps	12.5
The range of per instance allocated spectrum slots	1-8

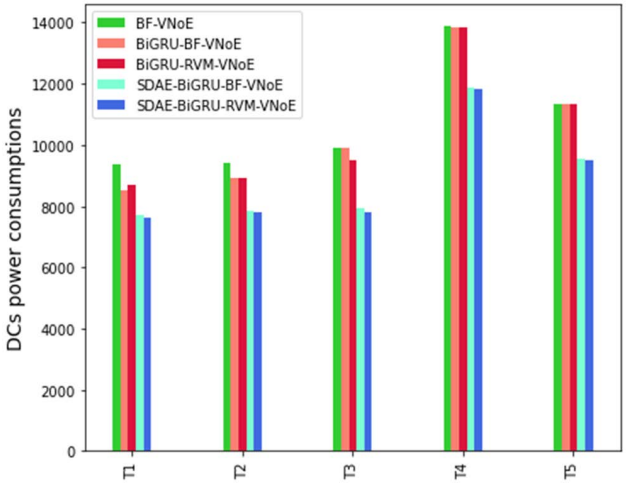

FIGURE 6. Power consumption results within a) intra DCs given the considered five VNoE policies.

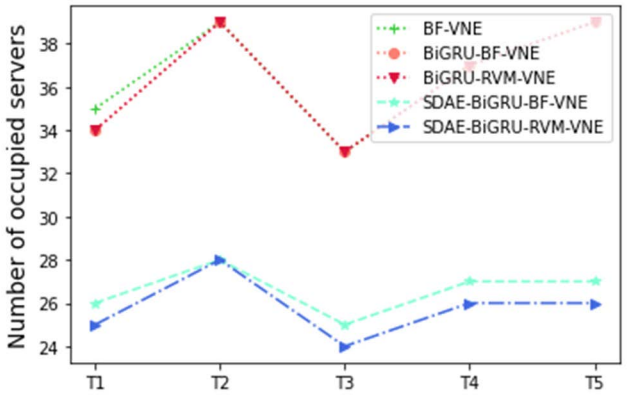

FIGURE 7. The total number of occupied servers in each time intervals.

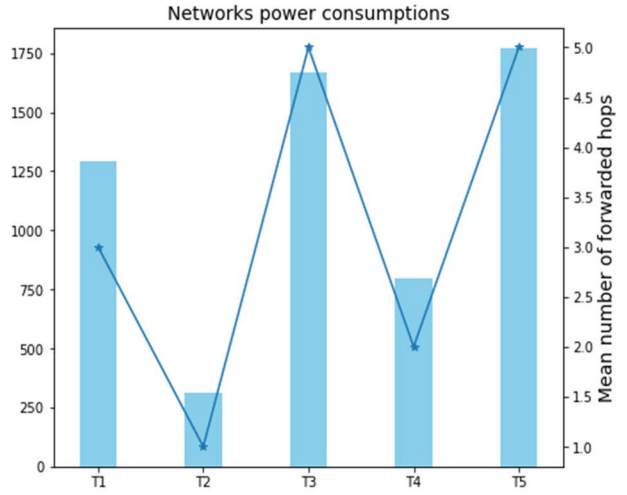

FIGURE 8. Power consumption results within inter AWS network topology using the proposed multistep VLiE.

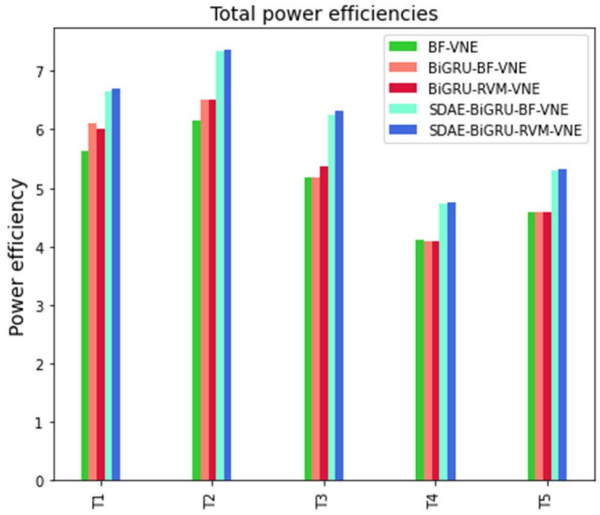

FIGURE 9. Total power efficiencies resulting from five VNE experimental intervals.

hops. Since the proposed routing strategy sorts feasible 3-SP adaptively according to network links states using two divers SP strategies, the network blocking rate was ultimately avoided under entire VNoE allocation policies. The same goes for intra DCs instances acceptance ratio which remained

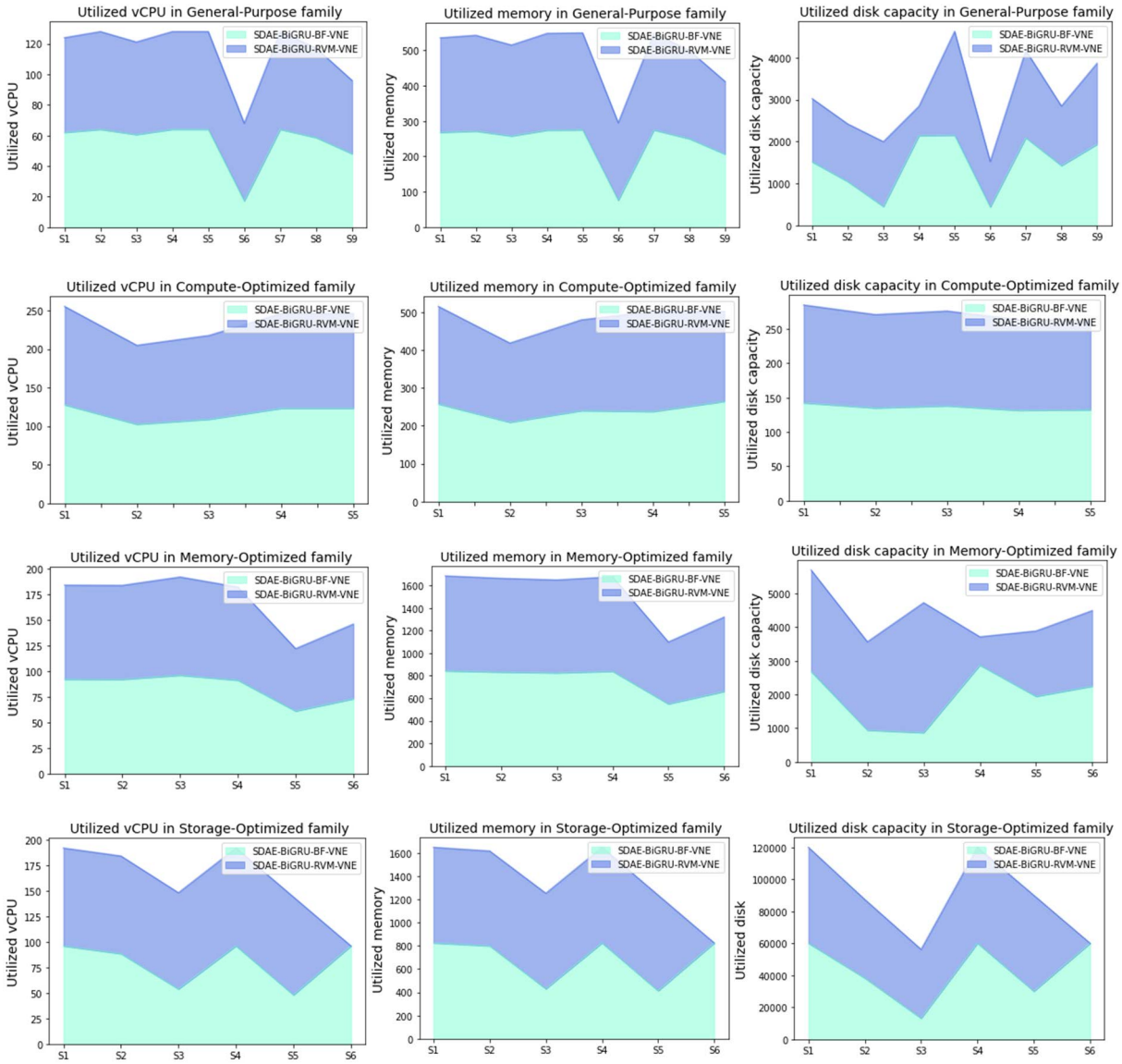

FIGURE 10. **Multi-level resources utilization in multimodal instances family racks.**

equivalent to 1 in all experiments, given the high considered availability of DCs locations and servers.

From a power efficiency point of view, total power efficiencies related to each experimental time intervals are depicted in (Figure 9). After converting power into KW, maximum power efficiencies were achieved during the second and first experimentation intervals. Indeed, this finding supports the model's dependence on both intra-DC power, given instance usage behaviors, provisioning natures in each time slot, variability in family types within a time slot, and then on inter-network power impacted by the considered destinations in the AWS topology.

It is obvious from Table 4 that the best method providing the most optimal assignments is the exact simplex MILP algorithm. While this is intended, we are also seeking speed and mixed-integer compatible processing with respect to the other two algorithms. When running 20 instances, the results provided by the SDAE-BiGRU-RVM-VNE are very near to the optimal values with an error not exceeding 0.21%. Besides, the proposed algorithm is significantly less time-consuming than simplex, even for large instances. In what follows, the discussion will concentrate on the analysis of the remaining experimental intervals by comparing only the two algorithms.

After demonstrating the effectiveness of the two multivariate proactive policies SDAE-BiGRU-RVM-VNoE and SDAE-BiGRU-BF-VNoE compared to the other univariate proactive policies and compared to the reactive policy, Figure 10 highlights some areas of divergence between these two policies in terms of multi-level resource exploitation.

TABLE 4. Experimental results comparison.

Instances number	Solution approach	P_{DC} in W	P_{net} in W	Running time in [s]
10	MILP	580.69	216	178.42
	SDAE-BiGRU-BF-VNE	580.69	216	1.43
	SDAE-BiGRU-RVM-VNE	580.69	216	1.95
20	MILP	1551.63	528	356.82
	SDAE-BiGRU-BF-VNE	1567.51	528	2.16
	SDAE-BiGRU-RVM-VNE	1554.91	528	3.21
30	MILP	-	-	-
	SDAE-BiGRU-BF-VNE	2146.26	800	3.81
	SDAE-BiGRU-RVM-VNE	2119.31	800	4.26

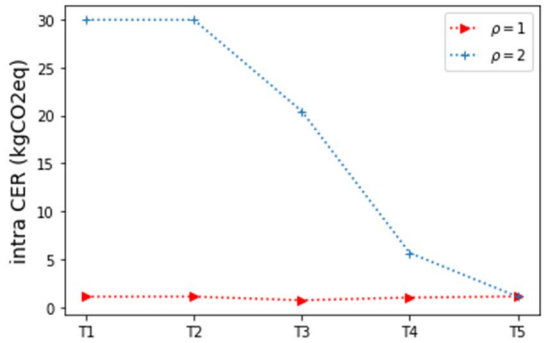

FIGURE 11. CER values obtained under GLA-GTR and DS-GTR peering scenarios.

It is worth reminding how both multivariate proactive policies deploy a multi-stage consolidation scheme depending on the instance family rack nature (Figure 2). Both proactive multivariate policies consider a vCPU resources consolidation in GP and CO families, a memory consolidation in MO family then a disk storage consolidation in SO family. While the SDAE-BiGRU-RVM-VNoE policy outputs optimal host index on the basis of a tri-dimensional resources investigation regardless of the envisaged instance family (equation 33), the SDAE-BiGRU-BF-VNoE solely relies on a single resource dimension that need to be consolidated.

Therefore, these differences explain the noticed disproportionate occupancy of host resources under the SDAE-BiGRU-BF-VNoE policy. These resources disproportions are particularly manifested over first occupied servers. Using the first three occupied servers examples (Figure 10), the proposed SDAE-BiGRU-RVM-VNoE shows an increase of 47.75% in occupied storage resources under the GP family, an increase of 27.09% of occupied storage resources under the MO family, then an increase of (20.15% - 20,08% - 36.73%) respectively in vCPU, memory, and storage under the SO family, respectively over the SDAE-BiGRU-BF-VNoE. These proportions explain the useless consumption of static powers beside unused resources when deploying the BF rule-based approach as a combination to the multivariate anticipated inputs.

In terms of CER metric, Figure 11 demonstrates obtained CER results in both GLA-GTR and DS-GTR peering scenarios. The sum of intra CER values resulting from the GLA-GTR algorithm during the five experimentation intervals accounts only for 6, 36% of the sum of CER values resulting from the DS-GTR algorithm.

VII. CONCLUSION

In this paper, we examined a jointure of a hybrid intra DCs VNoE policy with an inter cloud multistep VLiE policy, to solve the mismatch between the maximum resources utilization of personalized instances, overall power efficiency and the minimum geo-distributed carbon emission targets. Holistically, the proposed elastic and proactive VNE flowchart is founded on an AWS cloud case study to solve the formulated MILP optimization model. In the proposed system, green DCs and networking nodes are prioritized through a GLA-GTR algorithm.

The investigated VNE stipulated two instances provisioning scenarios: a supply capacity in excess over demand, and a supply capacity equaling demand. Future works could focus on improving our proposal's through incorporating other factors influencing the proposed VNE decisions. The first factor concerns the addressing of a third provisioning scenario in which demand exceeds resources capacity supply. Then, the other factors may concern the provisioning of multi-sources requests destined to multi-destination computing nodes. This later concern should be investigated within the proposed system in a parallel scheme, in which an effective ordering approach is therefore desirable to assign routing priority to the source/destination pairs where path finding is prone to be more problematic.

NOMENCLATURE

APN	Amazon Partners Networks.
AWS	Amazon Web Services.
AY-KSP	Adaptive Yens K-Shortest Path.
BF	Best Fit.
CER	Carbon Emission Rate.
CO	Compute Optimizer.
DCs	Data Centers.
ESD	Energy Storage Devices.
GLA-GTR	Green Location Aware Global Topology Ranking.
GP	General Purpose.
ISPs	Internet Service Providers.
MO	Memory Optimizer.
PDU	Power Distribution Unit.
REs	Renewable Energies.
RVM	Resources Vector Matching.
SDAE-BiGRU	Stacked Denoising Auto-Encoders Bidirectional Gated Recurrent Unit.

SD-WAN	Software Defined Wide Area Network.
SO	Storage Optimizer.
VLiE	Virtual Link Embedding.
VNE	Virtual Network Embedding.
VNoE	Virtual Node Embedding.
VPC	Virtual Private Cloud.
VPN	Virtual Private Network.

ACKNOWLEDGMENT

Hamzaoui Ikhlasse would like to thank EIGSI, ENSEM, FRDISI, and The Moroccan Ministry of Higher Education (CNRST) for their supports.

REFERENCES

[1] V. Eramo, E. Miucci, and M. Ammar, "Study of reconfiguration cost and energy aware VNE policies in cycle-stationary traffic scenarios," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 5, pp. 1281–1297, Apr. 2016, doi: 10.1109/JSAC.2016.2520179.

[2] N. Godinho, H. Silva, M. Curado, and L. Paquete, "A reconfigurable resource management framework for fog environments," *Future Gener. Comput. Syst.*, vol. 133, pp. 124–140, Aug. 2022, doi: 10.1016/J.FUTURE.2022.03.015.

[3] C. Ren, H. Li, Y. Li, Y. Wang, H. Xiang, and X. Chen, "On efficient service function chaining in hybrid software defined networks," *IEEE Trans. Netw. Service Manage.*, early access, Oct. 27, 2021, doi: 10.1109/TNSM.2021.3123502.

[4] M. Otokura, K. Leibnitz, Y. Koizumi, D. Kominami, T. Shimokawa, and M. Murata, "Evolvable virtual network function placement method: Mechanism and performance evaluation," *IEEE Trans. Netw. Service Manage.*, vol. 16, no. 1, pp. 27–40, Mar. 2019, doi: 10.1109/TNSM.2018.2890273.

[5] Equinix. (2022). *Amazon Web Services | Alliance Equinix*. Accessed: Mar. 21, 2022. [Online]. Available: https://www.equinix.fr/partners/aws

[6] B. Smith and G. Linden, "Two decades of recommender systems at Amazon.com," *IEEE Internet Comput.*, vol. 21, no. 3, pp. 12–18, May/Jun. 2017, doi: 10.1109/MIC.2017.72.

[7] AWS. (2022). *AWS Compute Optimizer*. Accessed: Mar. 22, 2022. [Online]. Available: https://aws.amazon.com/fr/compute-optimizer/

[8] M. H. Dahir, H. Alizadeh, and D. Gözüpek, "Energy efficient virtual network embedding for federated software-defined networks," *Int. J. Commun. Syst.*, vol. 32, no. 6, p. e3912, Apr. 2019, doi: 10.1002/DAC.3912.

[9] P. Zhang, X. Pang, Y. Bi, H. Yao, H. Pan, and N. Kumar, "DSCD: Delay sensitive cross-domain virtual network embedding algorithm," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 4, pp. 2913–2925, Oct. 2020, doi: 10.1109/TNSE.2020.3005570.

[10] C. Aguilar-Fuster and J. Rubio-Loyola, "A novel evaluation function for higher acceptance rates and more profitable Metaheuristic-based online virtual network embedding," *Comput. Netw.*, vol. 195, Aug. 2021, Art. no. 108191, doi: 10.1016/j.comnet.2021.108191.

[11] W. Fan, F. Xiao, X. Chen, L. Cui, and S. Yu, "Efficient virtual network embedding of cloud-based data center networks into optical networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 11, pp. 2793–2808, Nov. 2021, doi: 10.1109/TPDS.2021.3075296.

[12] A. A. Nasiri, F. Derakhshan, and S. S. Heydari, "Distributed virtual network embedding for software-defined networks using multi-agent systems," *IEEE Access*, vol. 9, pp. 12027–12043, 2021, doi: 10.1109/ACCESS.2021.3050922.

[13] S. Tomaszek, R. Speith, and A. Schürr, "Virtual network embedding: Ensuring correctness and optimality by construction using model transformation and integer linear programming techniques," *Softw. Syst. Model.*, vol. 20, pp. 1299–1332, Aug. 2021, doi: 10.1007/s10270-020-00852-z.

[14] Y. Zong, Y. Ou, A. Hammad, and K. Kondepu, "Location-aware energy efficient virtual network embedding in software-defined optical data center networks," *IEEE J. Opt. Commun. Netw.*, vol. 10, no. 7, pp. 58–70, Jul. 2018, doi: 10.1364/JOCN.10.000B58.

[15] P. Zhang, X. Pang, G. Kibalya, N. Kumar, S. He, and B. Zhao, "GCMD: Genetic correlation multi-domain virtual network embedding algorithm," *IEEE Access*, vol. 9, pp. 67167–67175, 2021, doi: 10.1109/ACCESS.2021.3076916.

[16] K. T. D. Nguyen and C. Huang, "Distributed parallel genetic algorithm for online virtual network embedding," *Int. J. Commun. Syst.*, vol. 34, no. 4, Dec. 2020, Art. no. e04691, doi: 10.1002/DAC.4691.

[17] P. Zhang, Y. Hong, X. Pang, and C. Jiang, "VNE-HPSO: Virtual network embedding algorithm based on hybrid particle swarm optimization," *IEEE Access*, vol. 8, pp. 213389–213400, 2020, doi: 10.1109/ACCESS.2020.3040335.

[18] H. K. Thakkar, C. K. Dehury, and P. K. Sahoo, "MUVINE: Multi-stage virtual network embedding in cloud data centers using reinforcement learning-based predictions," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 6, pp. 1058–1074, Jun. 2020, doi: 10.1109/JSAC.2020.2986663.

[19] D. Andreoletti, T. Velichkova, G. Verticale, M. Tornatore, and S. Giordano, "A privacy-preserving reinforcement learning algorithm for multi-domain virtual network embedding," *IEEE Trans. Netw. Service Manage.*, vol. 17, no. 4, pp. 2291–2304, Dec. 2020, doi: 10.1109/TNSM.2020.3022278.

[20] Z. Yan, J. Ge, Y. Wu, L. Li, and T. Li, "Automatic virtual network embedding: A deep reinforcement learning approach with graph convolutional networks," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 6, pp. 1040–1057, Jun. 2020, doi: 10.1109/JSAC.2020.2986662.

[21] P. Zhang, C. Wang, N. Kumar, W. Zhang, and L. Liu, "Dynamic virtual network embedding algorithm based on graph convolution neural network and reinforcement learning," *IEEE Internet Things J.*, early access, Jul. 6, 2021, doi: 10.1109/JIOT.2021.3095094.

[22] H. Yao, S. Ma, J. Wang, P. Zhang, C. Jiang, and S. Guo, "A continuous-decision virtual network embedding scheme relying on reinforcement learning," *IEEE Trans. Netw. Service Manage.*, vol. 17, no. 2, pp. 864–875, Jun. 2020, doi: 10.1109/TNSM.2020.2971543.

[23] AWS. (2022). *Régions et Zones de Disponibilité de l'Infrastructure Mondiale*. Accessed: Mar. 22, 2022. [Online]. Available: https://aws.amazon.com/fr/about-aws/global-infrastructure/regions_az/

[24] AWS. (2022). *Fonctionnalités Principales d'Amazon CloudFront*. Accessed: Mar. 22, 2022. [Online]. Available: https://aws.amazon.com/fr/cloudfront/features/?whats-new-cloudfront.sort-by=item.additionalFields.postDateTime&whats-new-cloudfront.sort-order=desc

[25] M. Gene, K. Vivek. (2020). *3 Steps Toward Cloud WAN Optimization for AWS Interconnection—Interconnections—The Equinix Blog*. Accessed: Mar. 22, 2022. [Online]. Available: https://blog.equinix.com/blog/2020/03/18/3-steps-toward-cloud-wan-optimization-for-aws-interconnection/

[26] *IP Optimized Optical Transport—The Cisco Routed Optical Network–Cisco*, Cisco, San Jose, CA, USA, 2020.

[27] C. Gu, L. Fan, W. Wu, H. Huang, and X. Jia, "Greening cloud data centers in an economical way by energy trading with power grid," *Future Gener. Comput. Syst.*, vol. 78, pp. 89–101, Jan. 2018, doi: 10.1016/j.future.2016.12.029.

[28] (2022). *ElectricityMap, Electricitymap*. Accessed: Mar. 22, 2022. [Online]. Available: https://app.electricitymap.org/?lang=fr

[29] IEA. (2022). *IEA—International Energy Agency*. Accessed: Mar. 22, 2022. [Online]. Available: https://www.iea.org/