Energy-Aware VNF-FG Placement with Transformer-based Deep Reinforcement Learning

Rania Sahraoui, Omar Houidi, Fetia Bannour

Energy-Aware VNF-FG Placement with Transformer-based Deep Reinforcement Learning

Rania Sahraoui*, Omar Houidi*, Fetia Bannour†
*Télécom SudParis, SAMOVAR, Institut Polytechnique de Paris, France
†ENSIIE, SAMOVAR, France
Email: rania.sahraoui@telecom-sudparis.eu, omar.houidi@telecom-sudparis.eu, fetia.bannour@ensiie.fr

***Abstract*—Although Network Function Virtualization (NFV) has introduced better flexibility and agility to the way network operators design, manage, and deploy their network services, it is still challenging to find the optimal real-time placement of network services which have evolved into complex dynamic graphs (or VNF-FGs) to satisfy the Quality of Service (QoS) requirements of their end-users and accommodate their dynamically changing service demands. Another crucial challenge that compounds the complexity of the online network service provisioning is to efficiently improve the utilization of the limited resources and reduce energy consumption and costs for service and infrastructure providers in large-scale networking environments such as 5G networks, edge computing, and Internet of Things (IoT). To meet both user and service provider needs, this paper proposes a novel Transformer-based Deep Reinforcement Learning (DRL) architecture, called TDRL (Transformer based-DRL), to address the dynamic energy-aware VNF-FG placement problem. Our intelligent encoder-decoder architecture leverages the power of both Graph Attention Networks (GAT) which extract the important features of the physical network, and sequence-to-sequence (seq2seq) models with Transformers which encode the ordered requirements of the complex VNF-FG service graphs. The main aim of these techniques is to improve the combined representation of the current state environment, and help our actor-critic DRL agent learn the optimal policy that achieves a "one-shot" placement decision of all VNFs in the service graph, thereby improving placement efficiency and resource utilization, especially in large-scale systems. Our extensive simulation results show that our TDRL approach significantly outperforms other state-of-the-art baseline learning algorithms in terms of achieving the optimal balance between acceptance ratio and energy efficiency.**

***Index Terms*—Energy Efficiency, Deep Reinforcement Learning, Transformer, Attention, VNF-FG Embedding, Scalability.**

I. Introduction

Network Function Virtualization (NFV) [1] is a promising networking paradigm that reduces network operating expenses by decoupling network functions from the underlying dedicated hardware, namely the traditional middle-boxes. This decoupling enables the hosting of network services, known as Virtualized Network Functions (VNFs), on commodity hardware (servers), which facilitates and accelerates service deployment and management by service providers, improves flexibility, leads to efficient and scalable resource usage, and reduces operational costs.

Network providers are confronted with the challenge of securing a stable power supply for their extensive large-scale networks. Due to the pivotal role of energy as an essential OPerating EXpenditure (OPEX) factor, its smart control is considered as necessary to facilitate network expansion. Therefore, Communications Service Providers (CSPs) are devoting much of their efforts to reduce energy consumption of their network infrastructures. In future communication networks, traditional management mechanisms, and centralized legacy solutions show their limitations in ensuring revenue for the infrastructure providers, the service providers, and a good Quality of Experience (QoE) for the end-users. The deployment of these services requires, typically, an efficient allocation of Virtual Network Function Forwarding Graphs (VNF-FGs) which is commonly known as the VNF-FG placement (or embedding) problem.

This paper focuses on the optimal placement of complex service graphs, or VNF-FGs, which implies the efficient placement of networking functions and their chaining to meet tenant requests while ensuring an efficient use of hosting infrastructure resources. A crucial challenge in this context is determining the optimal selection of physical servers to host the VNFs and the optimal set of physical links to interconnect the ordered VNFs. VNF placement algorithms may consider various network factors into account. Foremost among these factors are the resources available on the physical network, as their efficient utilization directly impacts the network provider's profitability. Additionally, it is crucial to focus on reducing the power consumption. By doing so, not only does it lower costs for the network provider, but it also contributes positively to environmental sustainability.

Machine Learning (ML) has recently piqued the interest of researchers all over the world. ML techniques are already widely used in networking [2]. However, the majority of these applications involve supervised learning. Many networking problems are unlabeled and can be viewed as decision-making issues in an uncertain environment. Reinforcement Learning (RL) approaches can be used to improve decision-making speed and accuracy when dealing with large service requests for dynamically constructing and updating end-to-end (E2E) slices (or VNF-FGs) across multiple infrastructures. Furthermore, Deep Reinforcement Learning (DRL), which combines conventional RL with neural networks, has recently been employed to efficiently solve complex, high-dimensional state spaces problems, such as the scalable VNF-FG placement challenge.

In this context, we propose TDRL, an intelligent energy-efficient VNF-FG embedding approach using Transformer-based DRL to address the NP-hard VNF-FG Embedding (VNF-FGE) problem in large-scale network environments.

The main contributions of this paper are the following:

This work was partially funded by the WIFIP project.

- We formulate the VNF-FGE problem as an Integer Linear Programming (ILP) optimization problem. Then, we model the VNF-FGE decision strategy as a Markov Decision Model (MDP) and establish a reward function that aims to achieve a trade-off between maximizing the acceptance rate (and thus the service providers' revenue) and minimizing the power consumption.
- We propose TDRL, a novel attention-based Transformer mechanism within an encoder-decoder framework that uses an Actor-Critic network structure, to address the online VNF-FGE problem.
- Our approach takes into consideration the features of physical hosts and links, and outputs the mapping decisions of VNFs on physical hosts and the steering of inter-VNF traffic across the hosts with verification of QoS constraints and consideration of energy efficiency.
- Through extensive simulation experiments, we demonstrate that our TDRL method performs better than existing algorithms. It achieves a higher acceptance rate, uses less power, and runs in a linear time frame.

The remainder of this paper is organized as follows. Section II discusses the related work and provides an overview of existing research on energy-efficient VNF-FG placement and MARL methods. Section III formulates the VNF-FGE problem and Section IV describes the proposed model. Section V outlines the performance evaluation and simulation results. Finally, Section VI summarizes the main findings.

II. Related Work

The VNF resource allocation problem in the single-domain scenario has received significant attention. While the majority of studies focused on the placement of simple single-VNF services, a few recent works addressed the realistic placement of complex service graphs involving multiple interconnected VNFs, or VNF-FGs. Different approaches have been followed in the literature to address the intra-domain VNF-FGE problem depending on the problem's characteristics and objectives.

To find exact solutions, the VNF-FGE optimization problem has been mathematically formulated as an Integer Linear Programming (ILP) [3]–[6], a Mixed Integer Linear Problem (MILP) [7], or an Integer Non-Linear Program (INLP) [8]. Since the problem is NP-hard, the resolution was only possible for small network instances, and different strategies are considered to solve the ILP problem for larger instances [9].

Soualah *et al.* [10] proposed ILP-based algorithms that reduce the number of selected candidate servers/hosts from the infrastructure to control complexity and improve scalability. The proposed embedding algorithms, which work in both online and batch modes, relied on VNF sharing across tenants to optimize physical resource usage and power consumption and increase provider revenues. In the same spirit, Tomassilli *et al.* [11] approached the problem as a Set Cover Problem, offering a relaxed ILP formulation. The authors propose an efficient and near-optimal logarithmic factor approximation algorithm based on LP rounding with proven theoretical performance results for the placement of chains of VNFs with ordering constraints. Their goal was to minimize the total deployment cost while satisfying the SFC requirements. On the other hand, due to the scalability limitations of the above-mentioned strategies and their limited performance for real-world applications (associated assumptions), heuristic-based and meta-heuristics-based solutions with lower complexity have been proposed to obtain approximate solutions in a viable execution time [12], [13]. Authors in [12] propose "Holu", a fast heuristic algorithm to tackle the power-aware and delay-constrained joint VNF placement and chaining problem by dividing it into two sub-problems: a VNF placement based on a centrality-based ranking method, and a VNF routing based on a Delay-Constrained Least-Cost (DCLC) method. Khebbache *et al.* [13] formulated the SFC placement problem as a multi-objective optimization problem and proposed a meta-heuristics evolutionary algorithm based on the Non-Dominated Sorting Genetic Algorithm (NSGA-II) to minimize the total VNF mapping cost and maximize the physical link utilization. Although meta-heuristics are known to perform better than heuristics in solving complex optimization problems and adapting to changing real-life contexts and dynamic environments, they can encounter convergence issues. Given these limitations, Reinforcement Learning (RL)-based approaches have been explored [14] to solve the on-demand placement of virtualized services in complex and uncertain network environments. In [14], the authors leverage the RL benefits to solve the VNF-FG problem in dynamic networks, in which an agent learns from past experience an optimal policy to use resources more efficiently, maximize the number of accepted requests, and increase the long-term provider revenue. Their Q-Learning-based approach was enhanced with an expert knowledge mechanism to accelerate the training process and thus improve efficiency and performance.

More recently, Deep Reinforcement Learning (DRL) techniques have shown great potential in addressing the high-complexity associated with the VNF-FGE problem in large-scale networks [15]–[20]. Quang *et al.* [16] modeled the VNF-FG allocation problem as a Markov Decision Process (MDP) and implemented an efficient and lightweight DRL-based actor-critic approach to maximize the number of allocated VNF-FGs with their QoS requirements in the substrate networks. The authors enhanced the conventional Deep Deterministic Policy Gradient (DDPG) to provide feasible VNF-FGE solutions while improving the exploration of the large action space for the DRL agent.

VNF-FG deployment has been optimized through various objectives and aspects. However, the primary consideration of energy remains an open challenging topic, particularly in a graph-structured Service Function Chain (SFC) environment. In [18], authors modeled the energy-efficient graph-structured SFC as a combinatorial optimization problem, where a Graph Convolutional Network GCN-based DRL was proposed to minimize the energy consumption by autonomously selecting nodes with adequate resources. It is also notable that the work by Tajiki *et al.* [21] is one among the few attempts to address the problem of service function chaining with the goal of minimizing energy consumption. Several heuristics were proposed consisting in iteratively placing the VNFs in series using the nearest search procedure. In [22], a sampling-based Markov Approximation (MA) approach, using replications, is proposed

for the energy-efficient VNF placement problem. This method needs a long time to find a near-optimal solution which makes it impractical. Indeed, solutions that take into account multiple parameters as in [21], [22] seem to have limited applicability and are more efficient in the cases of a small number of user nodes, due to their complexity costs. Unlike these works, our approach introduces a "one-shot" methodology for VNF-FG placement, handling all VNFs in one single step rather than sequentially. This not only accelerates the response to VNF-FG requests, but also minimizes potential inefficiencies that may arise when VNFs are placed sequentially.

III. VNF-FG Embedding Problem Formulation

Our scenario involves a set of network service requests that need to be efficiently mapped on top of the physical network with the aim of maximizing service acceptance and minimizing the overall power consumption of the infrastructure. However, this optimization must adhere to the constraints imposed by the resource demand specified by each service. First, we present the physical network model and that of the network services. Then, we provide the formal definition of the VNF-FG embedding problem.

A. Physical network (NFV Infrastructure (NFV-I))

To model the physical network, we designate the servers as a set of nodes denoted as N^p and their connections as L^p. As a result, the physical network can be represented by a weighted undirected graph G^p, where $G^p = (N^p, L^p)$. Each node in the graph corresponds to a server that has resources such as Central Processing Units (CPUs) and Random Access Memory (RAM). The available resources are quantified as r_{n^p}. Additionally, we consider the available bandwidth for link resources, denoted as b_{l^p} with $l^p \in L^p$.

B. VNF-FG requests

We denote the nodes and links as N^v and L^v for service request (or VNF-FG) i with time arrival t_i. So, we can model the VNF-FG as a directed graph $G^v = (N^v, L^v)$. The requested resources of each VNF $n_i^v \in N_i^v$ are denoted as $r_{n_i^v}$ where r refers to the resource and i to the request. We can group the resources of each VNF n_i^v in SFC i as a vector $r_{n_i^v}$ = [$r_{n_i^v,1}$,..,$r_{n_i^v,k}$,..,$r_{n_i^v,|K|}$], where $r_{n_i^v,k}$ is the amount of resource $k \in K$ requested by VNF n_i^v. The bandwidth demand of virtual link $l_i^v \in L_i^v$ is represented by $b_{l_i^v}$.

C. Problem formulation

The objective of this paper is to minimize the power consumption (PC) and maximize the acceptance ratio (AR) while guaranteeing the success of VNF-FG deployment. For simplicity, we assume that the power consumption of all physical network nodes is directly proportional to the CPU utilization (i.e., depending also on the number of active/working servers in the physical network). The main aim is to minimize the overall power consumption in the whole physical network, by putting into sleeping mode all non-utilized physical nodes that are at idle power consumption while accommodating VNF-FGs' demands.

Our objective considers jointly the power consumption and the acceptance ratio. Let us denote the trade-off parameters as e_1 and e_2, where e_1 represents the weight for energy consumption and e_2 represents the weight for the acceptance ratio. These parameters satisfy the condition $e_1 + e_2 = 1$.

TABLE I: Main notations

Notation	Description
N^p	Set of nodes for physical network
L^p	Set of links for the physical network
r_{n^p}	Amount of resources r available in physical node n^p
$r_{n^p}^{\max}$	Maximum capacity of resources r of physical node n^p
b_{l^p}	Amount of bandwidth available in physical link l^p
$b_{l^p}^{\max}$	Maximum capacity of bandwidth of physical link l^p
N^v	Set of VNFs
L^v	Set of virtual links
$r_{n_i^v}$	Amount of resources r required by VNF n_i^v in SFC i
$b_{l_i^v}$	Bandwidth demanded by link l_i^v in SFC i

We consider the computing resources of physical nodes and the bandwidth of physical links as constraints in our optimization problem. These constraints are combined with the optimization objective to define the overall problem :

- Each VNF can be deployed at only one substrate node n^p; therefore :

$$\sum_{n^p} M_{n^p}^{n^v} \leq 1, \quad \forall n^v \in N^v \tag{1}$$

- A VNF is successfully deployed when its substrate node has sufficient resources, i.e. the total amount of the required resource k of VNF-FG i mapped on n^p should not exceed its residual amount of resources.

$$\sum_{n^v} M_{n^p}^{n^v} \times r_{n^v,k} \leq r_{n^p,k}, \quad \forall n^p \in N^p, \quad \forall k \in K \tag{2}$$

  Where $M_{n^p}^{n^v}$ is a binary variable indicating if VNF n^v is deployed at substrate node n^p and $r_{n^p,k}$ is the available amount of resource k in physical node n^p.
- In our case, we focus specifically on the bandwidth metric for the successful deployment of a virtual link l_i^v. This involves ensuring that the associated VNFs are deployed successfully and that the Quality of Service (QoS) requirements related to the bandwidth are met:

$$\sum_{l^v} M_{l^p}^{l^v} \times b_{l^v} \leq b_{l^p}, \quad \forall l^p \in L^p \tag{3}$$

- If the VNF-FG i is accepted, the path mapped in the physical network should traverse through VNFs following the order specified in the request. We denote $I(n^p)$ and $O(n^p)$ as the sets of incoming and outgoing links of physical node n^p. Additionally, $n_{i,s}^v$ and $n_{i,d}^v$ represent the source and destination VNFs of the virtual link l_i^v.

$$\sum_{l^p \in I(n^p)} M_{l^p}^{l_i^v} - \sum_{l^p \in O(n^p)} M_{l^p}^{l_i^v} = M_{n^p}^{n_{i,d}^v} - M_{n^p}^{n_{i,s}^v}, \forall l_i^v \in L_i^v \tag{4}$$

A VNF-FG is considered accepted if and only if all of its constituent VNFs and Virtual Links (VLs) are successfully

allocated and the Quality of Service (QoS) requirements are met. The objective is to maximize the number of accepted VNF-FGs and minimize the power consumption.

$$\begin{aligned} \text{Maximize:} \quad & e_1 \times \text{AR} - e_2 \times \text{PC} \\ \text{Adhering to:} \quad & (1), (2), (3), (4) \end{aligned}$$

IV. TDRL: Transformer-based Actor-Critic approach for VNF-FG Embedding

A. MDP model for DRL

Under the DRL model, the agent-environment interaction is formalized as a Markov Decision Process (MDP). Within this framework, an agent operates with an environment E in discrete time steps, aiming to enhance its performance through experiential learning. At each time step t, the agent perceives an observation o_t from the environment, processes this information to determine an action a_t, and subsequently obtains a reward r_t. We assert that the environment is fully perceptible, indicating that the state, denoted as s_t, at time step t aligns with the observation o_t. In the following section, we will provide an overview of the state, action, and reward.

1) State: To achieve a fully perceptible environment, it is essential to integrate the descriptions of both the physical network and the VNF-FG request. Accordingly, we define the state as $s_t = (s_t^p, s_t^v)$ which incorporates:

- The physical network state s_t^p: In the given context, the current state of the physical network is denoted as $s_t^p = (A, X)$. The adjacency matrix of the network, $A \in \mathbb{R}^{|N^p| \times |N^p|}$, represents the network's connections, while $X \in \mathbb{R}^{|N^p| \times F}$ illustrates the feature matrix of the physical nodes. Each row of X refers to a F-dimensional feature vector of physical node n^p.
  To describe the physical nodes, we consider various resources as features. These include the available CPU c_{n^p}, the maximum CPU capacity $c_{n^p}^{\max}$, the sum of available bandwidth $B_{n^p} = \sum_{l^p \in L(n^p)} b_{l^p}$, and the maximum bandwidth $B_{n^p}^{\max} = \sum_{l^p \in L(n^p)} b_{l^p}^{\max}$ of adjacent links $L(n^p)$ as features of each physical server n^p. To ensure consistency, we normalize these values into the [0, 1] range. This normalization process guarantees a uniform representation for the considered features.
- The VNF-FG request state s_t^v: This denotes the current state of the VNF-FG request i, which consists of the required amount of resources $r_{n_i^v}$ of VNF n_i^v, the required bandwidth $b_{l_i^v}$ of virtual link l_i^v, and the number of VNFs remaining to be placed.

2) Action: At time step t, the agent makes a selection of servers to simultaneously instantiate all the Virtual Network Functions (VNFs) of the current VNF-FG request. The action taken by the agent is thus represented as $A_t = \{n^p \in N^p | r_{n_i^v,k} \leq r_{n^p,k}, \forall k \in K\}$, indicating the servers responsible for deploying the VNF-FG. It is possible for a server to host multiple VNFs, as long as the available resources of the server can satisfy the deployment requirements and constraints.

3) Reward: We adopt both the acceptance rate and power consumption as the reward for an action. A VNF-FG is deployed when all VNFs and virtual links (VLs) are deployed successfully. The acceptance ratio (AR) has recently been adopted to assess the performance of VNF-FG embedding algorithms [23]. Unlike previous approaches, which just use the acceptance ratio as a simple reward function, we introduce in our proposed TDRL approach, as shown in Eq. 6, a constrained reward expression to penalize decisions if the global consumed power at time t, $PC_G(t)$ is very high.

The power consumption, at time t, of a physical node $n^p \in N_p$ is estimated as defined in [24], [25] based on the following equation:

$$P_t(n^p) = P^{idle}(n^p) + \left[P^M(n^p) - P^{idle}(n^p)\right] \times U_t(n^p) \quad (5)$$

where $P^{idle}(n^p)$ is the power consumption where the machine n^p is at the idle state. $P^M(n^p)$ is the maximum power consumption. It is reached when the physical machine is fully used. $U_t(n^p)$ is the usage rate (i.e., a value between 0 and 1) of the initial CPU capacity.

The global consumed power at time t, $PC_G(t)$, equals the sum of the power consumption of the activated servers at time t. Formally, $PC_G(t) = \sum_{n^p \in N^p} P_t(n^p)$.

Equation 6 presents the reward expression used for performance evaluation.

$$\mathcal{R} = e_1 \times AR - e_2 \times PC_G(t) \quad (6)$$

where $PC_G(t)$ is the penalty term. The weights e_1 and e_2, which are assigned equal values, are respectively allocated to the acceptance ratio and power consumption. The weight e_1 emphasizes the importance of acceptance ratio, while e_2 penalizes excess power consumption. This equal weighting allows us to balance these crucial factors, reflecting the real-world necessity of optimizing multiple goals harmoniously.

B. TDRL model architecture

Our approach integrates Deep Reinforcement Learning (DRL) using Graph Attention Networks (GAT) [26], and a Transformer architecture [27] to optimize VNF-FG placement. In this framework, the encoder component of the Transformer is responsible for encoding the VNF-FG requests. Concurrently, the GAT network is employed to capture the features of the physical network. These two representations are then merged to form a full description of the environment state. Then, the decoder component of the Transformer makes "one-shot" placement decisions for the VNFs of a VNF-FG request. This approach enhances both placement efficiency and resource utilization in large-scale systems.

1) Network state embedding: The challenge of assigning an appropriate host in the physical network for each VNF within a VNF-FG is fundamentally based on the relevance of the environment's observation. Indeed, the accurate observation provides a comprehensive understanding of both the current state of the physical network and the demands of the VNF-FG enabling the agent to build a placement policy on the most relevant information. We learn separately the physical network embedding using a Graph Attention Network (GAT) and the VNF-FG embedding leveraging the encoder component of the Transformer model. GAT was selected for physical network modeling due to its suitability for environments with a constant topology but a changing resource utilization. Conversely, the Transformer's encoder was used for sequential VNF-FG

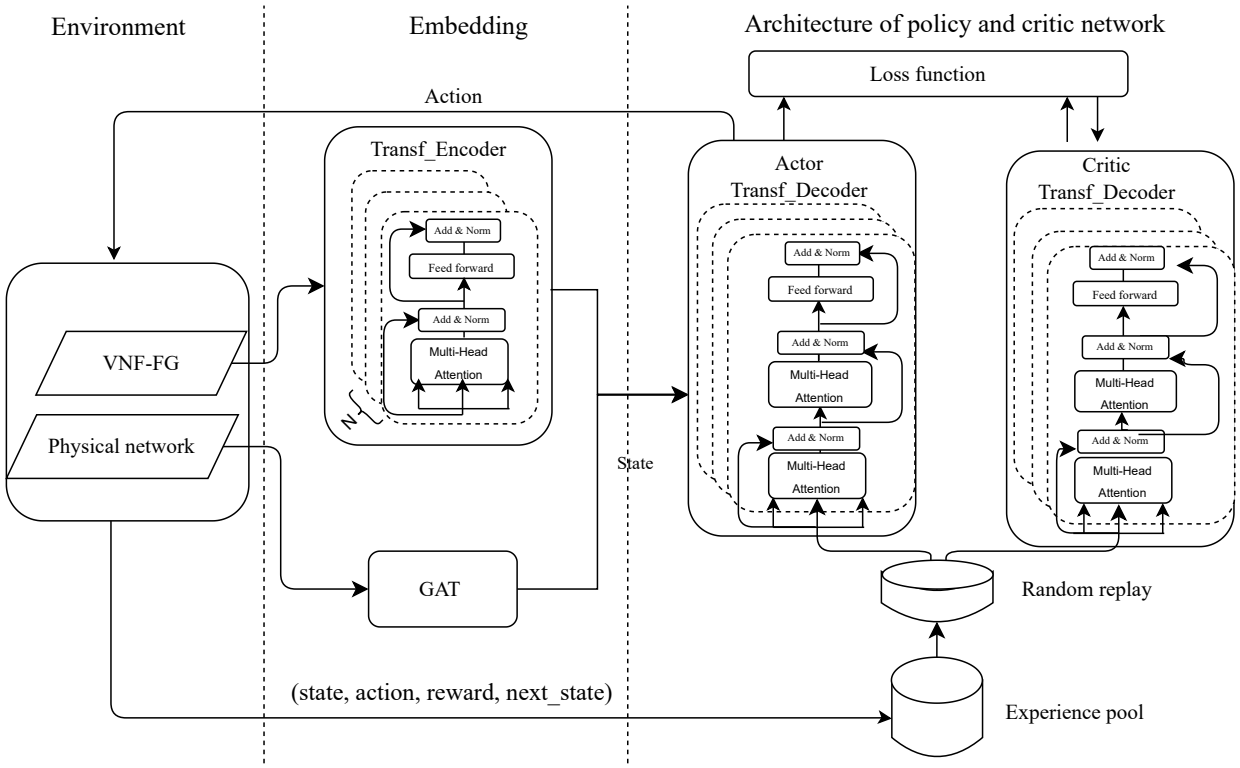

Fig. 1: Our TDRL architecture.

arrivals, as it excels in handling sequential data and capturing interdependencies among VNF-FGs.

a) GAT for physical network embedding: GATs are neural networks that use an attention mechanism to handle graph-structured data. They were first introduced by Veličković *et al.* in [26]. GATs are capable of focusing on the most informative parts of the input data.

A GAT consists of several layers, each performing the following computation steps:

- **Self-attention mechanism:** Each node i's feature vector, h_i, is first linearly transformed using a shared *weight matrix* $\mathbf{W}$. We then perform *self-attention* on the nodes (i.e., a shared attention mechanism a) by computing attention coefficients:

$$e_{ij} = a(\mathbf{W}h_i, \mathbf{W}h_j) \tag{7}$$

  Where the attention coefficients between node i and node j can be computed as:

$$e_{ij} = \text{LeakyReLU}\left(\mathbf{a}^{\intercal}[\mathbf{W}h_i \| \mathbf{W}h_j]\right) \tag{8}$$

  where $\cdot^{\intercal}$ represents transposition, $\|$ represents concatenation, and LeakyReLU (Leaky Rectified Linear Unit) is the activation function used in our case.
  The attention mechanism a is a single-layer feedforward neural network, parametrized by a weight vector "$\mathbf{a}$".
- **Normalization:** The attention coefficients are then normalized using the softmax function:

$$\alpha_{ij} = \text{softmax}_j\left(e_{ij}\right) = \frac{exp\left(e_{ij}\right)}{\sum_{k \in \mathcal{N}_i} exp\left(e_{ik}\right)} \tag{9}$$

- **Aggregation:** The next layer features are computed as a weighted sum of the neighbors' features, with weights given by the normalized attention coefficients:

$$h_i^{'} = \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} \mathbf{W} h_j\right) \tag{10}$$

  where σ denotes a non-linear activation function.

The attention mechanism in GATs enables the model to assess the influence of each neighboring node based on its features, leading to a more detailed representation of the network. This feature is particularly beneficial in physical networks, where the connections between nodes are key indicators of the network's overall state and dynamics.

b) Transformer Encoder for VNF-FG embedding: The host assigned to a VNF has an impact on the assignment of other VNFs and is also impacted by them. That is why considering the effective capture of inter-dependencies between nodes in the VNF-FG embedding task is perceived as essential.

To that end, we deploy a Transformer encoder in our TDRL architecture. This model, first introduced in the seminal paper "Attention is All You Need" by Vaswani *et al.* [27], is a kind of neural network architecture that counts heavily on attention mechanisms, ensuring equal attention to all parts of the input sequence, and that captures the relationships between their features. In contrast, traditional sequential models, such as recurrent neural networks (RNNs), analyze the data one element at a time. As the interval between the elements expands, their ability to capture and understand these distant dependencies diminishes, decreasing their performance in such tasks. The

Transformer encoder consists of a set of blocks, each block consists of the succession of two sub-layers: the self-attention mechanism and the feed-forward neural networks. The self-attention mechanism allows the model to attend to different positions within the input sequence and captures relevant contextual information. Here is the formula for self-attention:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{11}$$

In this equation, Q, K, and V represent the query, key, and value vectors, respectively. The softmax function is applied to the scaled dot-product of the query and key vectors, divided by the square root of the dimension of the key vector, d_k. Finally, the result is multiplied by the value vector.

The output of the self-attention mechanism is then passed through a feed-forward neural network (FFN) that consists of two linear transformations followed by a non-linear activation function, typically ReLU. Here is the formula for the FFN:

$$\text{FFN}(x) = \text{ReLU}(xW_1 + b_1)W_2 + b_2 \tag{12}$$

In this equation, x represents the input, W_1, b_1, W_2, and b_2 are the learnable weight and bias matrices.

By employing these components in the Transformer encoder, we can effectively capture and represent the requirements of each VNF in the VNF-FG, enabling more robust and efficient processing of the graph for various network functions.

We selected the Transformer model for three main reasons. First, it is computationally efficient for each layer. Second, it allows many operations to be done simultaneously, making them faster. Third, it is good at handling complex relationships in the data because the signals inside the network do not have to go through many steps.

Transformers process the entire input at once, giving equal attention to all parts of the input sequence. This capability makes them more suitable for tasks where inter-dependencies are crucial.

2) Policy learning with an Actor-Critic algorithm:

a) Transformer Decoder for the Actor network: Let s denote the state at any time step, which is represented as a sequence of VNFs, $\{v_1, v_2, \ldots, v_n\}$. We define an actor network A parameterized by θ that maps states to actions. In our model, the actor network is a Transformer decoder. Our choice of a Transformer decoder as the component to generate the placement policy, is motivated by:

$$A(s; \theta) = \text{TransformerDecoder}(s; \theta) \tag{13}$$

- Variable-length sequence handling: A VNF-FG is a sequence of VNFs which can vary in length. The Transformer Decoder inherently handles sequences of variable lengths, making it suitable for this task.
- Modeling inter-dependencies within the VNF-FG: The self-attention mechanism in the Transformer decoder allows it to model dependencies between different elements in the sequence. In the context of the VNF-FG, this means that the placement of one VNF can impact the placement of subsequent VNFs within the same graph. Through its self-attention mechanism, the Transformer decoder can learn these dependencies, which can lead to better placement decisions.
- Temporal dependencies across VNF-FGs: The ability of the Transformer decoder to use its previously generated outputs as inputs can also allow it to learn temporal dependencies across different VNF-FG placements.

The architecture of the Transformer decoder consists of a set of identical layers, each layer contains three sub-layers, as detailed below:

- Self-Attention Mechanism: This sub-layer helps the model focus on different parts of the input sequence. The output of the self-attention mechanism in layer l is computed as:

$$Z^l_{\text{self-attention}} = \text{Attention}(Q^l, K^l, V^l), \tag{14}$$

where Q^l, K^l, and V^l are the queries, keys, and values respectively. They are all derived from z^l, the output from the previous layer (or the input sequence for $l = 0$).

- Cross-Attention Mechanism: This sub-layer is traditionally used in the Transformer model to allow the decoder to focus on different parts of the input sequence. The output of the cross-attention mechanism in layer l is computed as:

$$Z^l_{\text{cross-attention}} = \text{Attention}(Q^l, K^l_{\text{enc}}, V^l_{\text{enc}}), \tag{15}$$

where Q^l is the same query from the self-attention mechanism, and K^l_{enc} and V^l_{enc} are the keys and values derived from the encoder output.

- Feed-Forward Network: This is a simple feed-forward neural network that consists of two linear transformations with a ReLU activation in between. The output of the feed-forward network in layer l is computed as:

$$Z^{l+1} = \text{FFN}(Z^l_{\text{cross-attention}}). \tag{16}$$

b) Transformer Decoder for the Critic network: The critic network is deployed as a Transformer decoder that estimates the Q-value using $V_\omega = (s_t, a_t)$. It plays a crucial role in evaluating the performance of an action by estimating the quality of the state-action pair. By analyzing the Q-value, the critic network assesses whether an action performs well or poorly in a given state.

The Q-value represents the expected cumulative reward that an agent can achieve by taking a particular action a_t in a specific state s_t, considering the future rewards and possible future states. Through its deep neural architecture, the critic network learns to approximate this Q-value by processing the state and action inputs.

By estimating the Q-value, the critic network provides valuable feedback to the reinforcement learning agent. This feedback guides the agent's decision-making process, enabling it to select actions that maximize the expected cumulative reward. In essence, the critic network serves as a powerful tool for evaluating and assessing the performance of actions in the context of an RL setting.

$$Q(s_t, a_t; \omega) = \mathbb{E}[R_{t+1} + \gamma \max_a Q(s_{t+1}, a; \omega) | s_t, a_t], \tag{17}$$

where: $Q(s_t, a_t; \omega)$ is the Q-value function, parameterized by ω, which estimates the expected return for taking action a_t

in state s_t, R_{t+1} is the immediate reward after taking action a_t in state s_t, s_{t+1} is the state following s_t, γ is the discount factor, which determines how much future rewards contribute to the Q-value. The expectation $\mathbb{E}[\cdot|s_t, a_t]$ is conditioned on the current state-action pair (s_t, a_t) and averages over the possible next states and rewards and the $\max_a$ operation selects the action that yields the highest Q-value in the next state s_{t+1}.

V. Performance Evaluation

This section presents the evaluation methodology and the simulation results. The algorithms are compared using extensive simulations where both the service requests and the hosting network infrastructure are generated using standard graph generation tools (such as the Waxman topology model [28]) and real infrastructure topologies.

A. Simulation setup

We randomly generate a physical network with 100 nodes and 500 links, following the Waxman topology model [28], which imitates a medium-sized infrastructure. The CPU resources of nodes and the bandwidth of links in the physical network are uniformly distributed with 50 to 100 units. In each episode, 1000 VNF-FG requests arrive at the system sequentially according to a Poisson process with an average arriving rate of 20 per 100 time units.

Specifically, each VNF-FG request consists of different numbers of VNFs from 5 to 15 according to the uniform distribution and has a lifetime exponentially distributed with an average of 1000. The node and link resources demand of VNF-FG requests are generated from the uniform distribution of 2 to 30. Our model architecture is built with PyTorch. Adam optimizer is employed to update the parameters of neural networks. Our simulation experiments are executed on a computer with 2.60 GHz Intel Core i7-9750H CPU and 16 GB of RAM. During the testing phase, only the actor network of the trained agent works to place 1000 VNF-FG requests. The simulation parameter values are given in Table II.

TABLE II: Simulation parameters

Learning rate of actor and critic	10^{-1}
Batch size	32
τ delayed network update rate	0.1
γ discount factor	0.95
Number Transformer layers	4
Number of attention heads	4
Optimizer	*Adam*

B. Baseline algorithms

To evaluate the efficacy of our proposed TDRL method, we choose the following two methods for comparative analysis:

- **GRU-A3C** [29]: A methodology that employs the Asynchronous Advantage Actor-Critic (A3C) [30] algorithm for concurrent training of both policy and value function estimations. To capture the orderly requirements of VNF-FG requests, the approach utilizes an encoder from a Seq2Seq model, implemented using a Gated Recurrent Unit (GRU) network.
- **GRC** [31]: A heuristic algorithm that maps VNFs into physical nodes based on the global capacity of resources.

C. Performance metrics

The metrics used for performance evaluation are typical indicators for placement algorithms and are defined below:

*1) **Acceptance ratio**:* is the ratio of incoming service requests that have been successfully deployed on the network to all incoming requests. In other terms, the rate of VNF-FG requests that have been accepted due to physical resource limitations (i.e., available CPU). Maximizing this quantity is equivalent to minimizing the rejection rate of requests.

*2) **Power consumption**:* The global consumed power at time t, $PC_G(t)$, is equal to the sum of the power consumption of the activated servers at this time t.

Formally, $PC_G(t) = \sum_{n^p \in N^p} P_t(n^p)$.

*3) **Execution time**:* is the time needed to find an embedding solution for each VNF-FG request. This metric reflects the ability of algorithms to scale with problem size. It is a decisive measure that assesses algorithm scalability. Service providers favor efficient and fast algorithms to quickly serve clients, especially in the context of large-scale network environments.

D. Results and discussion

We analyze the training loss to assess the performance of our TDRL-DDPG approach. As depicted in Figure 2, the loss functions for both TDRL-DDPG and GRU-A3C decrease after a few initial steps. However, the critic loss for TDRL-DDPG is notably more stable compared to that of GRU-A3C. This enhanced stability, along with the convergence to a steady state, contributes to a more consistent output policy for online VNF-FG placement. The observed rapid convergence is a result of extensive hyper-parameter tuning, a critical aspect of our training process.

Figure 3 depicts the mean acceptance rate of VNF-FG requests achieved by each algorithm. The acceptance rate of our TDRL-DDPG is higher than that of the GRU-A3C by

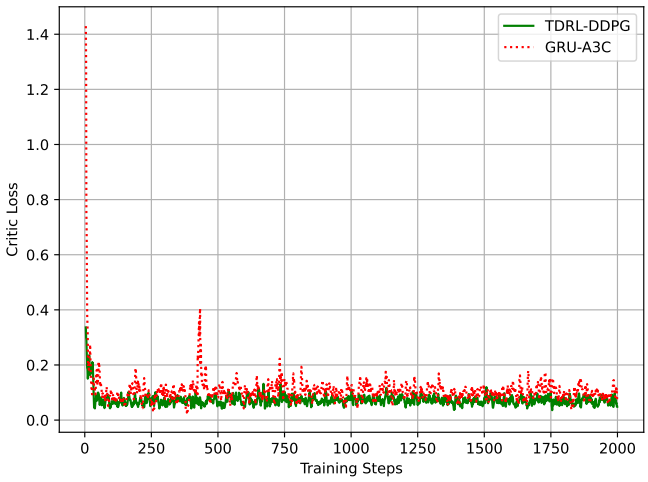

Fig. 2: TDRL-DDPG Critic Loss during training

23.38% and better than GRC by 40.46%. The TDRL approach employs the Transformer architecture's encoder to capture the ordered requirements and inter-node dependencies within the VNF-FG, and uses Graph Attention Networks (GAT) to obtain a nuanced representation of the physical network. The merged representation of the two high-dimensional representations gives a comprehensive view of the environment, facilitating more informed decision-making. This integration empowers the Transformer's decoder to execute "one-shot" placements for the entire VNF-FG, thereby enhancing efficiency and resource utilization.

The average power consumption of all three compared algorithms is presented in Figure 4. The proposed TDRL approach achieves an average power consumption around 31×10^3 watts versus 38×10^3 watts for GRC. Notably, our approach involves the optimization of two primary criteria: maximizing the acceptance ratio and minimizing power consumption. The pursuit of this dual optimization objective adds some complexity, further intensifying the challenges we encountered. However, our TDRL-DDPG method excels in achieving the optimal balance between these two contrasting objectives. It is worth noting that the baseline GRU-A3C approach achieves lower energy consumption but at the expense of a reduced acceptance ratio.

This observation strongly implies the presence of a compromise between the two optimization criteria. Consequently, we manage to achieve an optimal acceptance ratio while simultaneously maintaining a reasonable level of power consumption.

The aspect that deserves more attention is the algorithms' execution time as reported in Table III which clearly shows that the TDRL-DDPG and GRU-A3C methods have the best performance for this metric since they find solutions for the VNF-FG requests placement in a few milliseconds for 100 nodes, a few tens of milliseconds for 500 nodes. In addition, the reported results indicate that the execution times of the compared approaches grow linearly.

TABLE III: Execution time (ms)

Algorithms \ NFV-I	100	200	300	400	500
TDRL-DDPG	9.71	16.92	25.13	33.84	42.07
GRU-A3C	6.77	14.17	21.32	29.35	36.54
GRC	14.98	28.95	43.75	59.81	76.24

Table III reports the execution time performance of the algorithms to gain insight into their scalability and complexity with problem size. To evaluate this metric, we generate 1000 VNF-FG requests and vary the physical network (NFV-I) size from 100 to 500 nodes. These results indicate that the GRU-A3C algorithm has significantly better execution time when compared to our TDRL-DDPG algorithm and GRC's execution time. However, the faster execution times of the GRU-A3C algorithm are accomplished at the expense of the number of accepted requests. Indeed, for the scenario where NFV-I is equal to 100 nodes, our TDRL-DDPG achieves 78.1% as acceptance rate, versus 63.3% for the GRU-A3C, while the GRC accepts only 55.6%. The extra time taken by the TDRL-DDPG is due to the ability of our TDRL approach to find more feasible solutions and hence accept more requests.

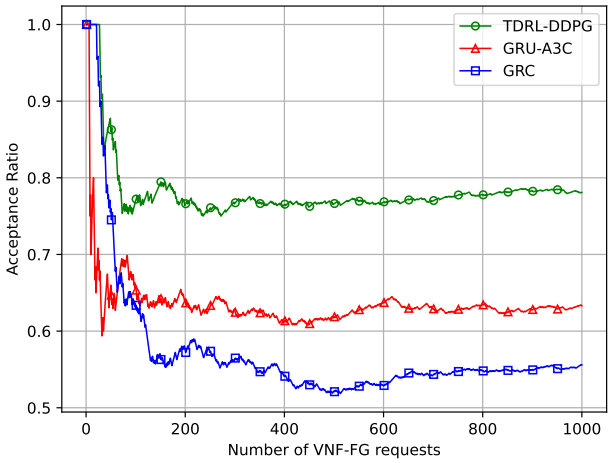

Fig. 3: Acceptance ratio for different VNF-FG requests.

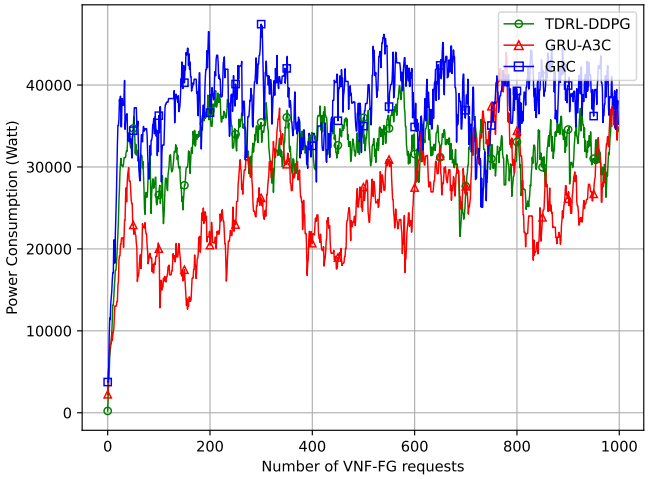

Fig. 4: Power consumption for different VNF-FG requests.

VI. Conclusion

This paper proposes a smart energy-efficient algorithm for VNF-FG placement and chaining in NFV-enabled infrastructures. Using a Transformer-based DRL mechanism, our solution reduces the global data-center power consumption through consolidation at the hardware level and optimizes the QoS for clients by optimally sharing the VNFs across multiple tenants at the service level. Simulation results of our approach confirm the applicability and the energy efficiency of our TDRL architecture. For large-scale topologies, our results highlight the scalability of our proposal which is ensured by an attention-based mechanism that extracts the relevant features from the underlying physical network. That said, our approach guarantees a balanced trade-off between service acceptance, energy savings, and scalability. In future work, we will address in-depth topics like VNF scaling, efficient migration methods, and strategies for fault-tolerant placement.

REFERENCES

[1] Bo Yi, Xingwei Wang, Keqin Li, Sajal K. Das, and Min Huang. A comprehensive survey of Network Function Virtualization. *Computer Networks*, 133:212–262, 2018.

[2] Junfeng Xie, F Richard Yu, Tao Huang, Renchao Xie, Jiang Liu, Chenmeng Wang, and Yunjie Liu. A survey of machine learning techniques applied to software defined networking (SDN): Research issues and challenges. *IEEE Communications Surveys & Tutorials*, 21(1):393–430, 2018.

[3] Marcelo Caggiani Luizelli, Leonardo Richter Bays, Luciana Salete Buriol, Marinho P. Barcellos, and Luciano Paschoal Gaspary. Piecing together the NFV provisioning puzzle: Efficient placement and chaining of virtual network functions. In *IFIP/IEEE IM 2015, Ottawa, ON, Canada, 11-15 May, 2015*, pages 98–106, 2015.

[4] Md. Faizul Bari, Shihabur Rahman Chowdhury, Reaz Ahmed, and Raouf Boutaba. On orchestrating virtual network functions. In *CNSM 2015, Barcelona, Spain, November 9-13, 2015*, pages 50–56, 2015.

[5] Hendrik Moens and Filip De Turck. VNF-P: A model for efficient placement of virtualized network functions. In *CNSM 2014 and Workshop, Rio de Janeiro, Brazil, November 17-21, 2014*, pages 418–423, 2014.

[6] Sevil Mehraghdam, Matthias Keller, and Holger Karl. Specifying and placing chains of virtual network functions. In *IEEE CloudNet 2014, Luxembourg, Luxembourg, October 8-10, 2014*, pages 7–13, 2014.

[7] Venkatarami Reddy Chintapalli, Vishal Siva Kumar Giduturi, Bheemarjuna Reddy Tamma, and A Antony Franklin. Ravin: A resource-aware vnf placement scheme with performance guarantees. In *NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium*, pages 1–9, 2023.

[8] Xiangqiang Gao, Rongke Liu, Aryan Kaushik, and Hangyu Zhang. Dynamic resource allocation for virtual network function placement in satellite edge clouds. *IEEE Transactions on Network Science and Engineering*, 9(4):2252–2265, 2022.

[9] Ákos Recse, Nattakorn Promwongsa, Amin Ebrahimzadeh, Seyedeh Negar Afrasiabi, Carla Mouradian, Wubin Li, Róbert Szabó, and Roch H. Glitho. Look-ahead vnf-fg embedding framework for latency-sensitive network services. *IEEE Transactions on Network and Service Management*, pages 1–1, 2023.

[10] Oussama Soualah, Marouen Mechtri, Chaima Ghribi, and Djamal Zeghlache. Online and batch algorithms for vnfs placement and chaining. *Computer Networks*, 158:98–113, 2019.

[11] A. Tomassilli, F. Giroire, N. Huin, and S. Pérennes. Provably efficient algorithms for placement of service function chains with ordering constraints. In *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, pages 774–782, 2018.

[12] Amir Varasteh, Basavaraj Madiwalar, Amaury Van Bemten, Wolfgang Kellerer, and Carmen Mas-Machuca. Holu: power-aware and delay-constrained vnf placement and chaining. *IEEE Transactions on Network and Service Management*, 18(2):1524–1539, 2021.

[13] Selma Khebbache, Makhlouf Hadji, and Djamal Zeghlache. A multi-objective non-dominated sorting genetic algorithm for vnf chains placement. In *2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, pages 1–4, 2018.

[14] Omar Houidi, Oussama Soualah, Wajdi Louati, and Djamal Zeghlache. An enhanced reinforcement learning approach for dynamic placement of virtual network functions. In *2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 1–7, 2020.

[15] Yikai Xiao, Qixia Zhang, Fangming Liu, Jia Wang, Miao Zhao, Zhongxing Zhang, and Jiaxing Zhang. NFVdeep: Adaptive online service function chain deployment with deep reinforcement learning. In *Proceedings of the International Symposium on Quality of Service*, pages 1–10, 2019.

[16] Pham Tran Anh Quang, Yassine Hadjadj-Aoul, and Abdelkader Outtagarts. A deep reinforcement learning approach for vnf forwarding graph embedding. *IEEE Transactions on Network and Service Management*, 16(4):1318–1331, 2019.

[17] Nan He, Song Yang, Fan Li, Stojan Trajanovski, Fernando A. Kuipers, and Xiaoming Fu. A-ddpg: Attention mechanism-based deep reinforcement learning for nfv. In *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*, pages 1–10, 2021.

[18] Siyu QI, Shuopeng LI, Shaofu LIN, Mohand Yazid SAIDI, and Ken CHEN. Energy-efficient vnf deployment for graph-structured sfc based on graph neural network and constrained deep reinforcement learning. In *2021 22nd Asia-Pacific Network Operations and Management Symposium (APNOMS)*, pages 348–353, 2021.

[19] Nan He, Song Yang, Fan Li, Stojan Trajanovski, Liehuang Zhu, Yu Wang, and Xiaoming Fu. Leveraging deep reinforcement learning with attention mechanism for virtual network function placement and routing. *IEEE Transactions on Parallel and Distributed Systems*, 34(4):1186–1201, 2023.

[20] Omar Houidi, Oussama Soualah, Ines Houidi, and Djamal Zeghlache. Energy Efficient VNF-FG Embedding via Attention-Based Deep Reinforcement Learning. In *2023 19th International Conference on Network and Service Management (CNSM)*, pages 1–7. IEEE, 2023.

[21] Mohammad M Tajiki, Stefano Salsano, Luca Chiaraviglio, Mohammad Shojafar, and Behzad Akbari. Joint energy efficient and QoS-aware path allocation and VNF placement for service function chaining. *IEEE Transactions on Network and Service Management*, 16(1):374–388, 2018.

[22] Chuan Pham, Nguyen H Tran, Shaolei Ren, Walid Saad, and Choong Seon Hong. Traffic-aware and energy-efficient vNF placement for service chaining: Joint sampling and matching approach. *IEEE Transactions on Services Computing*, 13(1):172–185, 2017.

[23] Roberto Riggio, Abbas Bradai, Davit Harutyunyan, Tinku Rasheed, and Toufik Ahmed. Scheduling wireless virtual networks functions. *IEEE Transactions on network and service management*, 13(2):240–252, 2016.

[24] Massoud Pedram and Inkwon Hwang. Power and performance modeling in a virtualized server system. In *2010 39th International Conference on Parallel Processing Workshops*, pages 520–526, 2010.

[25] Khaled Hejja. Power aware resource allocation and virtualization algorithms for 5g core networks. 2019.

[26] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, Yoshua Bengio, et al. Graph attention networks. *stat*, 1050(20):10–48550, 2017.

[27] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[28] Zhongxia Yan, Jingguo Ge, Yulei Wu, Liangxiong Li, and Tong Li. Automatic virtual network embedding: A deep reinforcement learning approach with graph convolutional networks. *IEEE Journal on Selected Areas in Communications*, 38(6):1040–1057, 2020.

[29] Tianfu Wang, Qilin Fan, Xiuhua Li, Xu Zhang, Qingyu Xiong, Shu Fu, and Min Gao. Drl-sfcp: Adaptive service function chains placement with deep reinforcement learning. In *ICC 2021-IEEE International Conference on Communications*, pages 1–6. IEEE, 2021.

[30] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR, 2016.

[31] Long Gong, Yonggang Wen, Zuqing Zhu, and Tony Lee. Toward profit-seeking virtual network embedding algorithm via global resource capacity. In *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*, pages 1–9. IEEE, 2014.