Attention-Based Actor–Critic DRL for Online Service Function Chain Composition in 6G Networks

Solomon Fikadie Wassie*, Eric Samikwa*, Torsten Braun*, Sławomir Kukliński†, David Jia‡, Véronique Capdevielle‡
*Institute of Computer Science, University of Bern, Switzerland
†Warsaw University of Technology, Poland
‡Communications, Network & Systems (CNS), Thales SIX, France

***Abstract*—Flexible Network Service composition is a fundamental enabler for the design of 6G networks, where network services are modeled as ordered Service Function Chains (SFCs) with heterogeneous Virtual Network Functions (VNFs). However, dynamic traffic generated by end users and dynamic network resource infrastructure utilization make online context-aware and resource-efficient SFC composition challenging. While Deep Reinforcement Learning (DRL) has been explored for this task, the multimodal nature of traffic and the variable-length inputs limit achievable performance. To address these challenges, we propose an attention-based actor–critic framework that integrates Transformer self-attention and encoding to capture variable-length SFC states and inter-VNF dependencies. The learned representations are then used by an actor–critic policy to sequentially select resource-aware composition actions, enabling adaptive and efficient service chain construction under dynamic network conditions. Extensive simulations show that our proposed transformer-augmented actor-critic DRL achieves faster policy convergence, lower bandwidth and computational resource consumption, and higher deadline satisfaction rates compared to state-of-the-art baselines.**

***Index Terms*—6G Network, Flexible Network Service Composition, Attention-Based Deep Reinforcement Learning**

I. Introduction

The introduction of Network Function Virtualization (NFV) and Software Defined Network (SDN) [1] has enabled greater agility and flexibility in how network operators design, manage, and deploy network services. Fixed chains of VNFs requested by tenants result in suboptimal chain allocation, directly affecting revenue and incurring high operational costs [2], as network operators are unable to restructure the chains to fit their network infrastructure. Given the ordering of VNFs is not always entirely fixed, certain VNFs have strict functional dependencies (e.g., the network flow must be decrypted before it can be further processed), while others can be arranged more flexibly (e.g., there is no strict dependency between Intrusion Prevention System (IPS) and Traffic Monitoring (TM) VNFs); therefore, multiple chains can satisfy the same Network Service (NS). Consequently, determining the optimal chain configuration for each NS is very important for Internet Service Providers (ISPs).

Although several studies propose multiple Service Function Chain (SFC) structures for a single network service through VNF parallelization and dynamic ordering [3], determining which SFC structure is optimal is known as the *service chain composition problem*. Service chain composition refers to the dynamic integration, chaining, and ordering of multiple homogeneous or heterogeneous VNFs into sequential or parallel SFC structures [4]. However, constructing an optimal, customized, real-time SFC with multiple paths tailored to a specific application remains an NP-hard combinatorial optimization problem [5].

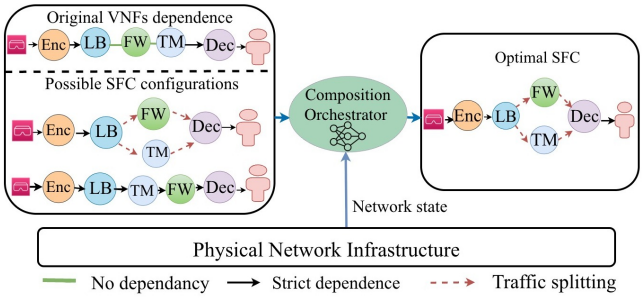


Fig. 1: High-level policy-driven optimal network service composition.

To illustrate flexible service composition, consider the scenario depicted in Figure 1. End-to-end service requests that follow Encryption (Enc), Load Balancer (LB), Firewall (FW), TM, and Decryption (Dec) arrive at the management plane with service specifications and logical VNF dependencies, shown as *Original VNFs dependence*. By exploiting the traffic-splitting capability of the LB VNF and the flexible ordering capability of TM, the composition orchestrator can derive multiple feasible SFC realizations (*Possible SFC configurations*) from the same logical service specification, where certain VNF dependencies are modeled as flexible rather than strictly sequential. The orchestrator selects the most resource-efficient configuration (*Optimal SFC*), while accounting for dependency constraints and the network state. The composition process is further detailed in Section III-A.

DRL methods [6], [7], [8] are widely adopted for SFC composition due to their adaptability to dynamic networks and their near-optimal decision-making capabilities. However, DRL models often suffer from low sample efficiency, unstable training, and limited robustness to abrupt changes in network state (e.g., resource bottlenecks, node failures, or traffic surges). Moreover, fixed-dimensional state and action representations hinder support for variable-sized and customized multi-path

ISBN 978-3-903176-82-9 © 2026 IFIP

SFCs, heterogeneous requests with varying numbers of VNFs, and unstructured data such as network intents, logs, and multi-modal traffic [9], ultimately leading to poor generalization.

Motivated by the above limitation, this work aims to address the following research question: **How can the order of VNFs be optimally decided in an online manner by adaptively selecting and chaining heterogeneous VNFs under dynamic service demands and resource utilization, to provision resilient End-to-End (E2E) network services while satisfying upper-layer application performance requirements?**

We propose a Transformer-Augmented Deep Reinforcement Learning (TDRL) framework for adaptive SFC composition decision leveraging the Transformer's capability to capture both local and global inter-VNF dependencies, enabling it to handle variable-length, high-dimensional network states and heterogeneous VNFs. Each VNF is represented as a token, and the entire SFC is modeled as a sequential input, allowing the framework to learn ordering constraints and long-range interactions among structured and unstructured multi-modal data flows, leveraging multi-head attention. The proposed approach dynamically selects resource-efficient, optimized SFC structures for informed E2E service composition decisions. The key contributions of this work are outlined as follows:

- We formulate the service chain composition problem as a multi-objective, constrained combinatorial optimization problem that considers functional dependencies while accounting for computing and network resources, and we further model it as a Markov Decision Process (MDP) to enable online optimal service chain composition decision.
- We propose a novel TDRL architecture for SFC composition in dynamic networks, augmenting both actor and critic networks with a decoder to capture temporally context-dependent inter-VNF traffic processing and unstructured multi-modal data within each VNF. Through multi-head attention, where each head learns complementary aspects of the optimization problem, including context-aware routing, long-range inter-VNF dependencies, position of VNF, and the topological suitability of the underlying network infrastructure for efficient resource allocation.
- We evaluate the performance of TDRL through extensive experiments, achieving faster convergence and lower bandwidth and compute resource consumption than state-of-the-art approaches.

The remainder of this paper is organized as follows. Section II explains the related works. Section III describes flexible network service composition in 6G, and Section IV presents the system model. Section V introduces the proposed TDRL approach. Section VI evaluates the proposed TDRL. Finally, Section VII concludes the paper.

II. Related Works

The efficient NFV resource allocation problem involves three main stages: service chain composition, SFC placement, and SFC scheduling. Several approaches, such as exact mathematical optimization [10], rule-based strategies [11], data-driven learning models [7], and Generative AI-assisted DRL approaches [12], have been proposed for optimal SFC composition optimization. *Ardagna et al.* [13] formulate service composition as an ILP problem; however, the problem's NP-hard nature leads to high computational complexity and limited adaptability in dynamic, large-scale networks. *Wang et al.* [11] adopt topology-based service composition methods with rule-based static policies modeling, which remains unsuitable for highly dynamic network environments.

Bian et al. [14] propose a game-theoretic learning framework to enhance adaptability via distributed decision-making; however, its reliance on prior knowledge of the environment and the game structure limits its applicability in highly dynamic networks and in inter-VNF modeling. *Ning et al.* [6] proposed a DRL-based framework that learns virtual link weights from dynamic traffic to optimize latency and resource utilization. Its fixed-state representations and limited temporal modeling constrain scalability and generalization; to solve these limitations, advanced variable-size handling learning-based approaches, such as the transformer, are needed.

Wang et al. [15] proposed graph-structured network representations, Graph Neural Networks (GNNs), with a pointer-network-based RL model for QoS-aware sequential composition, but graph aggregation and recurrent decoding limit long-range dependency modeling.*Heo et al.* [16] propose a topology-aware GNN-assisted DRL framework with strong generalization; however, it is unable to model the spatio-temporal dependence between VNF. *Sun et al.* [9] combine Generative AI with DRL to improve sample efficiency and generalization for adaptive SFC placement, but their scope is limited to placement. *Hsu et al.* [12] propose a Transformer-based actor-critic framework for sequence-aware SFC partitioning, improving acceptance rate via self-attention; however, it assumes predefined linear chains, whereas practical SFCs may form complex graphs with dynamic topologies.

Although mathematical optimization guarantees optimal solutions, it suffers from computational complexity and limited adaptability in dynamic, large-scale networks. Rule-based and heuristic methods are limited by rigidity, unable to learn, and struggle to adapt to complex network scenarios. Conventional data-driven models improve adaptability; however, they struggle to capture real-time network state information during sudden spikes or faults. Instead, the proposed TDRL leverages attention mechanisms to model complex traffic dependencies and learn adaptive service-composition policies in dynamic environments, enabling efficient service-chain optimization.

III. End-to-End Flexible Network Service Composition in 6G Network Architecture

Although Service-Based Architectures (SBAs) provide flexibility in modern network design, resource-efficient network service composition requires intelligent AI-based approaches that enable loosely coupled VNFs to discover and interact seamlessly, and adapt to dynamic network resource utilization and evolving service demand. We consider a generic system-level architecture for a 6G network over a distributed network infrastructure, as illustrated in Figure 2. End-user devices, such

as IoT devices, third-party applications, and Augmented Reality (AR) headsets, generate heterogeneous, dynamic traffic with application-specific requirements.

A. Dynamic Service Chain Composition: Illustrative Example

Dynamic traffic demands are triggered by application service requests and characterized by a set of network service descriptors for each VNF, as shown in the illustrative example in Figure 2 (Step ①). The service descriptors are summarized in the table and include the following information: (i) the initial service demand (Gbps); (ii) the set of VNFs, denoted by $\{V^{k_1}, V^{k_2}, \ldots, V^{k_4}\}$, with corresponding compute demand values of 40, 30, 50, and 60 [CPU cycles/Gbps] and data rate ratios of 120%, 40%, 60%, and 50%, respectively, and (iii) the functional dependency relationships among the VNFs, as illustrated by the dependency graph. The compute demand indicates the required processing effort per unit input traffic [CPU cycles/Gbps]. The data rate ratio is the traffic-volume increment factor for a VNF, defined as the ratio of its outgoing to incoming data rates [%]. The symbol "$\rightarrow$" indicates that the VNFs that follow can process the data only after the preceding VNFs have completed their processing, while the symbol "—" indicates possible flexibility in the ordering of the VNFs.

The composition orchestrator considers high-level network information and measures the resource requirements of each SFC structure (Step ②). For instance, we can drive two possible SFC structures for a single service request as depicted in Figure 2. The bandwidth requirement of the first structure with the link between V^{k_1} and V^{k_2} is 1.2 Gbps, obtained by multiplying the incoming load of V^{k_1} (i.e., 1 Gbps) by its data rate ratio 120%. The compute demand of V^{k_2} is 36 CPU cycles, obtained by multiplying its incoming traffic load (i.e., 1.2 Gbps) by its processing requirement of 30 (CPU cycles/Gbps). Using the same procedure, the total BandWidth (BW) and computational demands of candidate SFC chain 1 are calculated, resulting in a total BW requirement of 2.97 Gbps and a total compute demand of 113 cycles. Similarly, the total BW requirement and compute demand for candidate SFC chain 2 are 2.5 Gbps and 110 cycles, respectively. Finally, the composition orchestrator selects the least-resource-demanding chain (Step ③).

B. Service Composition Workflow in 6G Network Architecture

The complete service chain composition has been proposed as part of the 6G-Cloud architecture [17]. The service chain composition process in service-based architectures encompasses steps such as service request treatment, service discovery, network service descriptors (NSD) instantiation, VNF selection, ordering, and execution, while adapting to dynamic network conditions. This dynamic composition is based on the use of NSDs, which are standardized specifications that define the structural, functional, and lifecycle properties of network components. NSDs specify the topology of network services composed of multiple VNFs, including their chaining, access points, operational policies, service-flow dependencies, logical connectivity flows, and the processing and rate of traffic volume variation of individual VNFs to create complete E2E service chains. The overall procedure consists of the following steps.

1) Service request and discovery: A service is requested for a Network Service that specifies a Service Level Agreement (SLA). The Master Service Orchestrator (MSO) identifies high-level service requests with a given SLA. This SLA details critical service parameters, including maximum latency, minimum bandwidth, availability, security policies, and other quality attributes. The system consults the Assets Repository (ARep) and service registry to identify available VNFs and Network Functions (NFs) capable of fulfilling parts of the requested intent. This repository contains predefined catalogs of VNF, including RAN and Core VNFs. This results in a candidate pool of service components matching the intent's requirements.

2) Composition logic and NSD instantiation: The relationships among selected VNFs and components are analyzed considering dependencies between VNFs. Strong dependencies represent mandatory, tightly coupled relationships. Weak dependencies denote preferred but non-mandatory ordering or interaction. Using dependency analysis, the VNFs are arranged into an SFC, a linear or partially ordered sequence reflecting the flow of traffic/services through network functions. This SFC is then expanded into a composition structure, which may include branching, parallelism, or optional paths based on dependencies and intents. This results in a logical blueprint in the form of an NSD. This descriptor guides orchestrators and management systems on service instantiation.

3) Composition implementation: The VNFs and components described in the Network Service Director are deployed onto the physical or virtual infrastructure. Simultaneously, service bus bindings are established to connect components via APIs, message buses, or event streams, ensuring they operate cohesively within the service chain. This results in a fully instantiated, interconnected, and operational network service. An advanced approach to composition involves dynamically selecting the ordering of VNFs based on real-time resource availability and variable resource demands, optimizing performance and resource utilization while satisfying dependencies and service constraints. This paper focuses on the final step in implementing service chain composition.

IV. System Model and Problem Formulation

A. System Model

An overview of our system model is depicted in Figure 2, where service demands are continuously generated by users and arrive at the orchestrator online. We consider a scenario involving an edge-cloud continuum distributed infrastructure modeled as a connected undirected graph $G = (V, E, W)$, where V and E denote the sets of physical network nodes and links, respectively. Each node $v \in V$ represents a physical network entity, such as an extreme-edge node, an edge server, or a central cloud server. The available CPU/GPU resources are described as a function of time t to indicate the physical server's dynamic resource utilization, with real-time capacity $C_i(t)$ [cycle/s]. The weight function $W : E \rightarrow \mathbb{R}^+$ represents the bandwidth

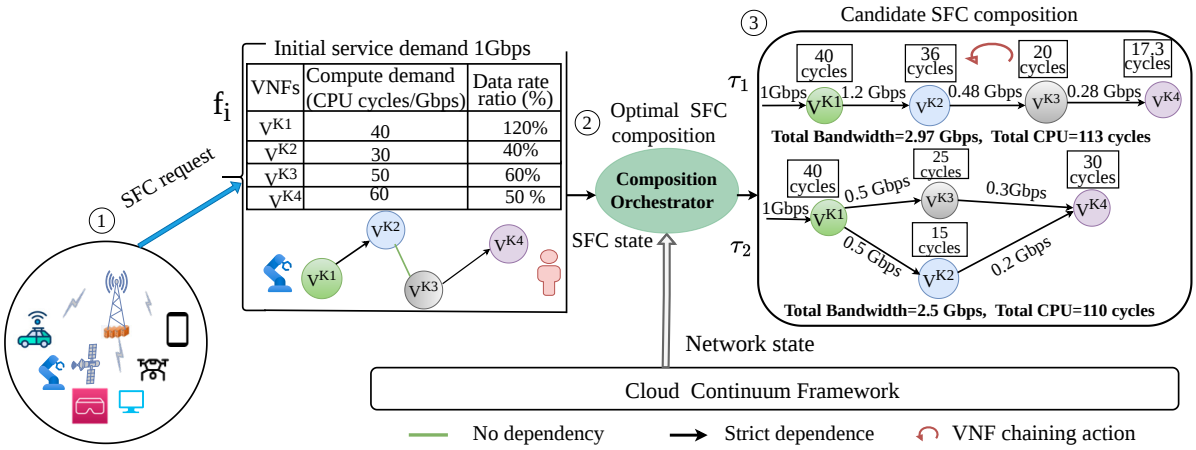


Fig. 2: System model for flexible network service composition with service requests descriptors, and their dependency graph.

capacity of each link. Each link $(v_i, v_j) \in E$ corresponds to a physical network link, representing a high-speed fiber link between nodes, whose available bandwidth varies over time and is described as a function of time $B_{ij}(t)$ [bit/s]. We consider a set of end-to-end service requests $F = \{f_1, f_2, \ldots, f_N\}$ arriving at the composition orchestrator over time follows a Poisson process with mean arrival rate λ, where N denotes the total number of SFC requests. We model a system in which a single request is selected at a given time. We further define a single SFC request f_i with its performance characteristics as follows.

$$f_i = (\mathcal{S}_i, \mathcal{D}_i, K_i, L_i, \zeta_i, \delta_i, B_i^{\min}, D_i^{\max}, \sigma_i), \tag{1}$$

where $\mathcal{S}_i$ and $\mathcal{D}_i$ denote the source and destination nodes, K_i denotes the set of VNFs, and L_i denotes the set of logical links between successive VNFs. ζ_i is the rate of traffic arrival intensity δ_i [s] is the service lifetime, $B_i^{\min}$ [bit/s] is the minimum throughput requirement, $D_i^{\max}$ [s] is the maximum tolerable end-to-end delay, and σ_i [cycle/s] represents the aggregate computational demand. The structure of f_i depends on the application it supports, and can be determined by the application type.

The i-th arriving SFC request specifies a set of VNFs $K_i = (k_{i,1}, k_{i,2}, \ldots, k_{i,u_i})$, where u_i denotes the maximum number of instantiated VNFs required to provide an end-to-end network service. Each VNF $k_{i,j} \in K_i$ represents a distinct softwareized virtual network function with specific resources and performance characteristics, and is capable of processing incoming packets. We further define M as the number of VNFs that do not have strict precedence (execution) dependencies, i.e., VNFs that can be freely reordered within the chain and thus arranged arbitrarily during composition.

B. Problem Formulation for E2E Service Composition

We formulate the problem of *optimal SFC composition* over a physical network, aiming to determine a resource-efficient SFC structure that minimizes the combined bandwidth and computing resource demands. The detailed mathematical formulations for the dependency modeling, the feasible composition space, and the SFC composition objectives are presented below.

a) Dependency Modeling: The functional dependencies among VNFs for request f_i are conceptually represented by a directed acyclic graph (DAG) $d_i = (K_i, L_i)$, where vertices correspond to VNFs in K_i and directed edges in L_i encode mandatory precedence relations. The degree of freedom (DoF) [18] of the dependency graph is defined as $\mathcal{D}(d_i) = |K_i|(|K_i| - 1) - |L_i|$, which quantifies the number of admissible ordering relations among VNFs that are not constrained by mandatory dependencies. A larger DoF indicates greater flexibility in constructing feasible SFC topologies and directly impacts the size of the feasible composition space. Despite these constraints, the feasible composition space grows combinatorially with a higher number of non-dependent VNFs M and the chain length u_i. In particular, the number of possible compositions is upper bounded by $\binom{u_i}{M} M! = u_i!/(u_i - M)!$ without repetition, or M^{u_i} with repetition. This rapid growth highlights the intrinsic combinatorial complexity of the SFC composition problem.

b) Feasible Composition Space: We define T_i to denote the set of feasible SFC compositions for request f_i. Each $\tau_i \in T_i$ represents a candidate SFC topology encoding both VNF ordering and interconnection structure. For example, $\tau_i = [k_{i,1}, k_{i,2}, k_{i,3}, k_{i,4}]$ denotes a linear chain of four VNFs, whereas $\tau_i = [(k_{i,1}, k_{i,2}), k_{i,3}, k_{i,4}]$ represents a branched topology. To preserve valid functional dependencies among VNFs, we define a binary *dependency matrix* $A = [a_{mn}] \in \{0,1\}^{u_i \times u_i}$, where $a_{mn} = 1$ if VNF k_m must precede k_n. For service request f_i, let A_i denote the restriction of A to the selected VNFs K_i. Let $X_i = [x_{pq}^i]$ be the adjacency matrix of the composed chain, where $x_{pq}^i = 1$ if VNF $k_{i,p}$ precedes $k_{i,q}$ in τ_i. Feasibility requires $x_{pq}^i - a_{pq}^i \geq 0, \quad \forall p, q \in \{1, \ldots, u_i\}$ ensuring that all mandatory dependencies defined in A_i are preserved in the composition adjacency matrix X_i. In other words, if a dependency $a_{pq}^i = 1$ (i.e., VNF $k_{i,p}$ must precede $k_{i,q}$), then the corresponding entry in the composition must also satisfy $x_{pq}^i = 1$. Otherwise, when $a_{pq}^i = 0$, the condition remains valid for either $x_{pq}^i = 0$ or $x_{pq}^i = 1$, allowing

flexibility in the ordering of independent VNFs. This ensures that mandatory functional constraints are strictly enforced while preserving sufficient DoF for the optimizer to explore resource-efficient, performance-aware SFC configurations.

c) SFC Composition Objective: The SFC composition problem aims to determine the optimal service chain for each service request by minimizing a cost function subject to resource constraints. The objective balances bandwidth and computational resource consumption, where $\beta = (\beta_1, \beta_2)$ denotes weighting parameters determined by the network administrator according to the service provider's business objectives. Let $D(\tau_i)$ and $B(\tau_i)$ denote the E2E delay and achievable throughput of the composed chain τ_i, computed over all virtual links and VNFs in the selected sequence. For every possible SFC topology τ_i, the aggregate communication resource demand is defined as $\mathcal{B}(\tau_i) = \sum_{(k_p,k_q)\in\tau_i} b_{i,(p,q)}$, where $b_{i,(p,q)}$ [bit/s] denotes the bandwidth consumption on the virtual link between two consecutive VNFs $k_{i,p}$ and $k_{i,q}$ after VNF processing. Similarly, the aggregate computation demand of the chain is defined as $\mathcal{C}(\tau_i) = \sum_{k_p\in K_i} \mathcal{P}_{i,p}$, where $\mathcal{P}_{i,p}$ denotes the processing demand of a single VNF $k_{i,p}$.The optimal order of the service chain composition for request f_i is then given by $\tau_i^\star = \arg\min_{\tau_i\in T_i} [\beta_1 \cdot \mathcal{B}(\tau_i) + \beta_2 \cdot \mathcal{C}(\tau_i)]$, where $\tau_i^\star$ represents the optimized structure of the SFC that minimizes the aggregated value of communication and computation demands. Formally, this optimization can be expressed as

$$\underset{\tau_i\in T_i}{\text{minimize}} \quad \overbrace{\beta_1 \cdot \sum_{(k_p,k_q)\in\tau_i} b_{i,(p,q)}}^{\text{Bandwidth demand}} + \overbrace{\beta_2 \cdot \sum_{k_p\in K_i} \mathcal{P}_{i,p}}^{\text{Processing demand}} \tag{2}$$

subject to

$$D(\tau_i) \le D_i^{\max} \tag{2a}$$

$$B(\tau_i) \ge B_i^{\min} \tag{2b}$$

$$X_i - A_i \ge 0 \tag{2c}$$

The constraints ensure the feasibility and correctness of the composed SFCs. Constraint (2a) enforces the end-to-end latency requirement by limiting the cumulative processing and transmission delay of each composed chain to the maximum tolerable delay $D_i^{\max}$. Constraint (2b) guarantees service-level throughput by ensuring that the achievable end-to-end bandwidth of the composed chain meets the minimum requirement $B_i^{\min}$. Finally, Constraint (2c) enforces functional correctness by requiring the composition adjacency matrix X_i to be element-wise greater than or equal to the dependency matrix A_i, thereby preserving all mandatory VNF precedence relations and ensuring that the resulting SFC topology is semantically valid.

V. Transformer-Augmented DRL For Online Service Composition

In this section, we present a novel TDRL framework to solve the service chain composition optimization problem described in Section IV. The proposed framework leverages the variable-length sequence modeling capability of Transformers and the dynamic optimization capability of DRL through interaction with unknown network environments. The Transformer architecture, originally developed for natural language processing tasks, employs self-attention mechanisms to learn dependencies among input tokens [19]. We leverage these functionalities to embed the network state within a Transformer encoder, while Transformer decoders implement the actor and critic networks, enabling the learning of long-range VNF dependencies and supporting adaptive, real-time service composition. As illustrated in Figure 3, VNF requests include a source and destination VNF, a time-to-live (TTL), generic application performance requirements specified in a service-level agreement (SLA), and possible SFC structures τ_1 and τ_2. The replay buffer serves as temporary on-policy storage, storing trajectories (states, actions, rewards) within each update batch for gradient computation.

A. DRL Modeling of Service Chain Composition Optimization

We reformulate the service chain composition optimization problem described in Equation 2 as Markov Decision Process (MDP), given that the DRL agent interacts with an environment E at discrete time steps, aiming to learn an optimized, resource-efficient SFC structure through experience-driven learning. At each time step t, the agent observes the environment state o_t, which captures both the current SFC structure resource requirement and the underlying network infrastructure conditions. The detailed formulation of the state S_t, action A_t, and reward R_t is given below.

1) **State Space S_t:** Describes the current situation of the DRL agent in the environment. The system state is defined as the combination of the SFC request states and the physical network state. The system state at time t is given by $S_t = \left(V_{i,t}^{k_i}, V_{i,t}^{k_2}, \ldots, b_{V_\mathcal{S},\mathcal{D}}^{|K_i|}, f_i\right)$, where $V_t^{k_i}$ indicates that VNF k_i is composed over physical network V and $b_{V_\mathcal{S},\mathcal{D}}^{|K_i||K_j|}$ represents the bandwidth consumption on the physical link between source node $V_\mathcal{S}$ (hosting k_i) and destination node $V_\mathcal{D}$ (hosting $|k_j|$). The state information also includes the performance requirements of the SFC request f_i, the bandwidth requirement, the E2E delay requirement, the order dependence, and the traffic change ratios.
2) **Action Space A_t:** The action of the DRL agent is modeled as a multidimensional vector that generates an ordering of VNFs to construct an E2E SFC. At each time step t, the agent selects a feasible VNF composition that satisfies both dependency and resource constraints. Accordingly, the action space is defined as $A_t = \{\tau_1, \tau_2, \ldots, \tau_{|T|}\}$, where the feasible set T is constructed based on VNF dependency constraints. Each composition τ_i consists of an ordered (or partially ordered) set of VNFs, given by $\tau_i = \{V^{k_1}, (V^{k_2}, V^{k_3}), \ldots, V^{k_n}\}$, where V^{k_i} denotes the i-th VNF in the chain, and (V^{k_2}, V^{k_3}) represents a branched (parallel) execution of VNFs within the service chain.
3) **Reward Function R_t:** The reward function evaluates the impact of the agent's decisions on constructing

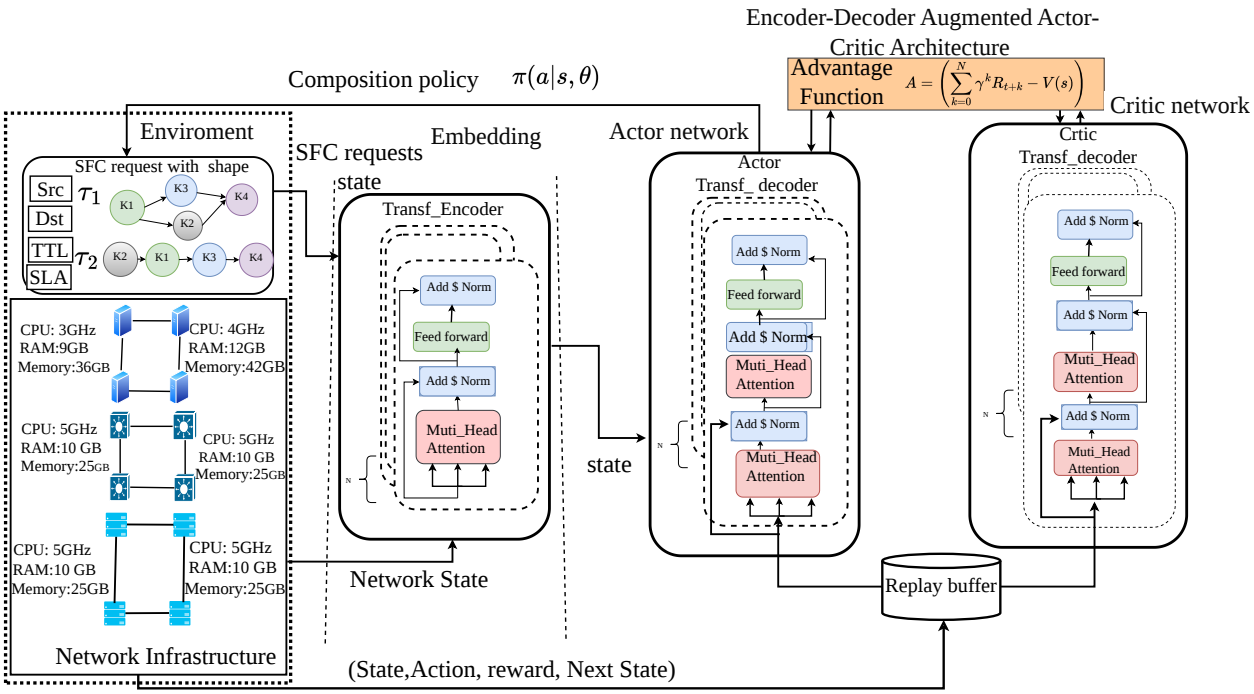


Fig. 3: Transformer augmented actor-critic proximal policy optimization for service composition.

an SFC structure. It is modeled as a constrained optimization objective given by the average path length to include branched SFC, given by $R = -\sum_{t=0}^{T} \gamma^t \left(\beta_1 \cdot \sum_{(k_p,k_q)\in\tau_i} b_{i,(p,q)} + \beta_2 \cdot \sum_{k_p \in K_i} \mathcal{P}_{i,p}\right)$, where β_1 and β_2 are optimization weights corresponding to BW consumption and computational resource consumption, respectively. The value of these weights is determined by policymakers based on the service provider's goals. The maximum time step over which the agent learns the optimal composition policy is denoted as T, and $\gamma \in (0, 1]$ is the discount factor.

B. Transformer Encoder for Network State Embedding

The proposed approach specifically uses a transformer-based Proximal Policy Optimization (PPO) for stable learning and captures both temporal dynamics and structural dependencies in service function chaining. While the Reinforcement Learning (RL) agent interacts with the environment to perceive network state information at each time step t, it constructs a contextual trajectory $\zeta_t = (x_i, x_{i+1}, \ldots, x_{t-1})$, where $x_i = (a_i, o_i, r_i)$, $\forall i \in \mathcal{I}$, represents historical action, observation, and reward encoded to tokens with a d-dimensional space with context length $N = t - 1 - i$. The historical tokens $x_i, \ldots, x_{t-1}$ are transformed into vector representations via positional encoding and embedding, followed by a linear transformation: $\hat{x}_i = \mathrm{ReLU}(x_i W_1 + b_1)W_2 + b_2$, where $\mathrm{ReLU}(\cdot)$ is the activation function, W_1, W_2 are learnable weight matrices, and b_1, b_2 are bias vectors. The attention layer then maps $\hat{x}_i, \ldots, \hat{x}_{t-1}$ into $\langle Q, K, V\rangle$ without altering dimensionality, where $Q_i = W_Q \hat{x}_i$, $K_i = W_K \hat{x}_i$, and $V_i = W_V \hat{x}_i$, with W_Q, W_K, W_V as learnable weights. The scaled dot-product attention module concatenates the token-wise representations across the transformed vectors to form $Q_{\hat{x}_i} = [Q_i, Q_{i+1}, \ldots, Q_{t-1}]$, $K_{\hat{x}_i} = [K_i, K_{i+1}, \ldots, K_{t-1}]$, $V_{\hat{x}_i} = [V_i, V_{i+1}, \ldots, V_{t-1}]$. Given the above vector matrices, the attention mechanism derives the contextual feature.

$$\boldsymbol{\alpha}_t = \mathrm{Attention}(Q, K, V) = \mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d_k}} + \mathcal{M}\right) V \quad (3)$$

where $\mathcal{M}$ is the mask matrix. The resulting attention feature $\boldsymbol{\alpha}_t$ is subsequently fed into the actor-critic network, where it serves as a contextual representation to facilitate the learning of the SFC composition policy.

C. Transformer-Augmented Actor-Critic Network

We employ a Transformer-based backbone shared by both the actor and critic to model variable-length VNF sequences, making it suitable for dynamic SFC composition. Conditioned on the attention representation $\boldsymbol{\alpha}_t$ generated by the encoder, the actor selects an action while capturing temporal dependencies across sequential composition decisions. The policy is defined as $a_i^t = \arg\max_{A_i^t} \; p_\theta(A_i^t \mid \boldsymbol{\alpha}_t)$, where θ denotes the actor parameters and p_θ represents the conditional probability over the feasible action set A^t. The actor parameters are updated by maximizing the clipped surrogate objective $\theta = \arg\max_\theta \; \mathbb{E}_{a_t \sim \pi_{\text{old}}}\left[r_\theta \cdot \hat{A}_t(o_t, a_t)\right]$, where the importance weight is defined as $r_\theta = \mathrm{clip}\left(\frac{\pi_\theta(o_i^t, a_i^t)}{\pi_{\text{old}}(o_i^t, a_i^t)}, 1-\epsilon, 1+\epsilon\right)$, with ϵ denoting the clipping parameter and π_{old} the previous policy. The advantage estimate is computed as $\hat{A}_t(s_t, a_t) = \sum_{x=i}^{t-1} \gamma^x r_i^x - V_\phi(s_t)$, where ϕ denotes the critic parameters and

Algorithm 1: Adaptive Network Service Composition

Input: $F = (f_1, f_2, \ldots, f_N)$, $T_{\max}$, G^{topo}, $b(i,j)$,
// Initialize policy, value, replay buffer
1 **Initialization:** ϕ_0, π_0, D_t
// Initially randomly structure SFC
2 $S_t = \left(V_t^{K_i}, V_t^{K_2}, \ldots, b_{V_{\mathcal{S}},\mathcal{D}}^{|K_i|}, G_{\text{topo}}, \tau_i\right)$
// Initialize the reward to zero
3 $R_t \leftarrow 0$
4 **for** $t \in T_{max}$ **do**
// RL Agent selects SFC topology τ_i
5 $f_i \leftarrow Select(T, \tau_i)$
6 $\bar{c} \leftarrow$ `MeasureComputeDemand`(τ_i)
7 $\bar{b} \leftarrow$ `MeasureBandwidthDemand`(τ_i)
8 $\bar{l} \leftarrow$ `MeasureDelayRequirment`(τ_i)
9 $\bar{q} \leftarrow$ `MonitorsubsetofPhysicallink`(G^{topo})
10 $\overline{\boldsymbol{\alpha}_t} \leftarrow \text{CalculateAttention}(\boldsymbol{\alpha}_t)$
// Translate the attributes of τ_i and attention $\boldsymbol{\alpha}_t$ as the state
11 $S_t \leftarrow (\bar{l}, \bar{b}, \bar{c}, \bar{q}, G^{\text{topo}}, \overline{\boldsymbol{\alpha}_t})$
// Execute $A_t \sim \pi_\theta$ following current policy
12 $S_t \xrightarrow{A_t, \pi_\theta(A_t|S_t,A_t)} S_{t+1}, R_{t+1}$ // Rebuild the structure of τ_i
// Store each s_t, a_t, R_t in replay buffer
13 $D_t \leftarrow \{s_t^0, a_t^0, r_t^0, \ldots, s_t^1, a_t^1, r_t^1 .. s_t^n, a_t^n, r_t^n\}$
14 **if** $B^{min}(\tau_i) < b(i,j)$ **then**
15 $A_t \sim \pi_\theta(A_t \mid S_t, A_t) \xrightarrow{\text{Execute}} R_{t+1}, S_{t+1}$
// calculates advantage estimate
16 $A^{\pi_{\theta_t}}(s_t, a_t) \leftarrow \sum_{x=i}^{t-1} \gamma^x r_i^x - V_\phi(s_t)$
// Update θ and ϕ parameters
17 $\theta_{t+1} = \arg\max_\theta \frac{1}{|\mathcal{D}_t|T} \sum_{\tau \in \mathcal{D}_t} \sum_{t=0}^{T}$
$\min\left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_t}(a_t|s_t)} A^{\pi_{\theta_t}}(s_t, a_t), g(\epsilon, A^{\pi_{\theta_t}}(s_t, a_t))\right)$
18 $\phi_{t+1} = \arg\min_\phi \frac{1}{|\mathcal{D}_t|} \sum_{\tau \in \mathcal{D}_t} \sum_{t=0}^{T} (V_\phi(s_t) - R_t)^2$
19 **else**
// rebuild the topology of SFC
20 $S_t \xrightarrow{A_t, \pi_\theta(A_t|S_t,A_t)} S_{t+1}$
// Select optimal structure of τ_i optimally
21 $\tau_i^\star = \arg\min_{\tau_i \in T_i} [\mathcal{B}(\tau_i))]$,
22 **return** $\phi(s_t), \pi_\theta(s_t)$

$V_\phi(o_t)$ is the value function. The critic is trained by minimizing the squared value loss $\phi = \arg\min_\phi \left(\sum_{x=i}^{t-1} \gamma^x r_i^x - V_\phi(s_t)\right)^2$. The overall training procedure is summarized in Algorithm 1.

D. Adaptive Network Service Composition Algorithm

The step-by-step workflow of Algorithm 1 is given as follows. The inputs include the set of VNFs K_i with their attributes, the SFC request requirements f_i, and the physical network topology G_{topo}. The policy and value network parameters θ_0 and ϕ_0, and the replay buffer, are initialized (line 1). An initial reward is set to $R_t = 0$, and a random logical chaining structure τ_i is then generated, and the i (lines 2–3). Second, the algorithm evaluates the bandwidth and computational demand of τ_i, observes the available network resources, and computes the attention parameter $\overline{\boldsymbol{\alpha}_t}$ (lines 4–10). Based on the current policy π_θ, a composition action is selected, yielding the new state S_{t+1}; the transition (s_t, a_t, R_t) is stored in D_t (lines 10–14). Thirdly, the BW and computational demands are compared with the available resources (i.e., $B^{\min}(\tau_i) < b(i,j)$), and the reward R_t. Third the advantage function $A^{\pi_{\theta_t}}(s_t, a_t)$ is computed (lines 13–17). Finally, the parameters θ_{t+1} and ϕ_{t+1} are updated, and the procedure iterates until convergence to the optimal SFC composition policy (lines 15–22).

TABLE I: Simulation Parameters

Parameter	Value
Number of nodes ($\lvert V\rvert$)	$\{10, 20, \ldots, 50\}$
Transformer layers (ℓ)	6
Embedding dimension (d_{model})	512
Number of attention heads (h)	4
Feedforward dimension (d_{ff})	2048
Dropout rate (P)	0.1
Layer norm epsilon (ϵ_{norm})	10^{-6}
Maximum sequence length ($\lvert K_i\rvert$)	10
Discount factor (γ)	0.99
Clip ratio (ϵ)	0.2
Number of epochs (N_{epoch})	50
Batch size (N_{batch})	$\{64, 128, 256\}$
Learning rate (α)	$\{5 \times 10^{-4}, 10^{-3}, 3 \times 10^{-3}\}$

VI. Performance Evaluation

A. Simulation Setup

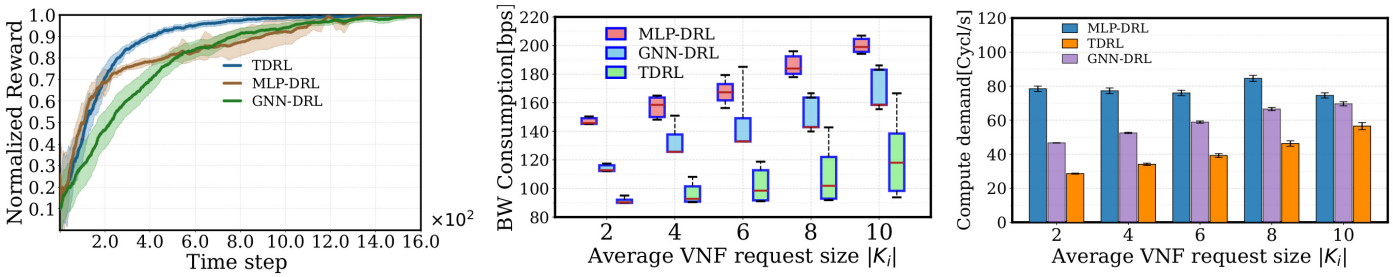
We developed a Python-based customized simulation environment that uses NetworkX[1] to generate and manage the realistic USA-NET physical network topology, consisting of 50 nodes and 200 links. For the DRL implementation, we adopt the OpenAI Gymnasium framework[2]to develop a customized RL environment and use Stable Baselines3 [3]to train and evaluate a customized actor-critic RL agent. The computing and bandwidth capacities of physical nodes and links are randomly generated within the range of $[50, 500]$ CPU cycles and $[100, 1000]$ [bit/s], respectively. We randomly generate SFC requests composed of a heterogeneous number of VNFs in set of $\{2, 3, 4, 6, 8, 10\}$ with the requests are subject to E2E delay and BW constraints, with customized execution dependencies among the VNFs. The computational demand of each VNF and the bandwidth demand between consecutive VNFs in an SFC are generated uniformly within the ranges $[5, 50]$ CPU cycles and $[5, 10]$ [bit/s], respectively. Proper hyperparameter tuning balances exploration and exploitation. The detailed simulation settings are summarized in Table I.

B. Compared Baseline Algorithms

To evaluate the performance of the proposed TDRL framework, we compare it against competitive learning-based baseline approaches, including a conventional DRL model that encodes network states using a standard neural network and a GNN-assisted DRL approach that leverages graph neural networks to effectively extract features from graph-structured data, model network topology, and VNF dependencies. Although heuristic and optimization methods perform well for static problems, we exclude them as baselines due to their limited adaptability to dynamic environments

- **MLP-DRL** [6]: A DRL-based SFC composition approach that uses an MLP with an Advantage Actor Critic (A2C)

[1]https://networkx.org/
[2]https://gymnasium.farama.org/index.html
[3]https://stable-baselines3.readthedocs.io/en/master/

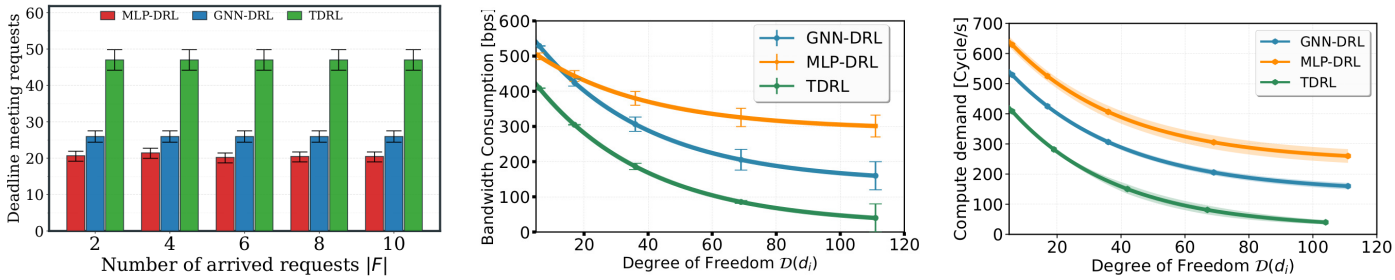


(a) Learning convergence of TDRL compared with baselines.

(b) Impact of average VNF size on bandwidth consumption.

(c) Impact of average VNF size on compute resource demand.

Fig. 4: Overall performance comparison of the proposed approach, in terms of learning convergence, bandwidth consumption, and computational demand compared with baselines.



(a) Impact of composition decisions on deadline satisfaction.

(b) Impact of degree of freedom on bandwidth consumption.

(c) Impact of the degree of freedom on compute resources demand.

Fig. 5: Performance comparison of the number of deadline-satisfied requests and the impact of degrees of freedom on bandwidth and computational resource consumption.

algorithm to generate routing weights, but lacks explicit modeling of VNF ordering, service chain topology, and long-range dependencies.

- **GNN-DRL** [15]: A GNN-assisted DRL approach for service chain composition, where GNN provides topology-aware representations of service dependencies, and a pointer network–based DRL agent sequentially selects and orders services to construct feasible and QoS-aware chains.

C. Performance Metrics

To evaluate the proposed TDRL approach, we consider three metrics: **Policy Learning Performance**, which reflects the ability to learn effective service chain composition decisions as defined in Equation 2; **Bandwidth Consumption**, defined as the total bandwidth used by accepted SFC virtual links relative to the selected physical link capacity; and **Compute Resource Consumption**, defined as the aggregate VNF processing demand allocated during service chain composition relative to the available infrastructure compute capacity.

D. Evaluation Results

We examine the convergence behavior and learning efficiency of the attention-based actor–critic architecture compared to two learning-based baselines, as shown in Figure 4a. The TDRL model converges more rapidly and achieves higher rewards. Although the baselines exhibit similar behavior during the early training episodes, GNN-DRL improves performance by capturing network topological features, achieving results comparable to TDRL. Moreover, the TDRL framework enhances learning by modeling temporal context and parallelizing the processing of tokenized VNF via an attention mechanism, thereby improving sample efficiency.

Figure 4b presents the average BW consumption as compared with the average VNF request size for three different methods. The BW consumption increases steadily as the number of VNF sizes grows. However, the proposed TDRL method consistently outperforms the baseline methods by reducing BW consumption through sample-efficient learning of optimal SFC composition decisions. This performance gain is attributed to the Transformer network's attention mechanism, which learns inter VNF dependency for optimal composition decision over GNN- and MLP-based feature extractors in DRL. As shown in Figure 4c, the compute demand grows approximately linearly with the average VNF size $|K_i|$ for all schemes. When average size of VNF $|K_i| = 2$, the compute demand is about 25 Cycle/s for TDRL, 40 Cycle/s for MLP-DRL, and 45 Cycle/s for GNN-DRL; at $|K_i| = 10$, it increases to roughly 75 Cycle/s, 100 Cycle/s, and 110 Cycle/s, respectively. Hence, over $|K_i| \in [2, 10]$, the total increase is approximately 50 Cycle/s (TDRL), 60 Cycle/s (MLP-DRL), and 65 Cycle/s (GNN-DRL), corresponding to average per-VNF increments of about 6.25, 7.5, and 8.1 Cycle/s, respectively. Therefore, TDRL achieves the smallest marginal compute growth per additional VNF,

thanks to the transformer's attention mechanism.

Figure 5a illustrates the number of deadline-meeting requests versus the number of arrived requests. As traffic arrivals increase from 2 to 10, all approaches maintain nearly stable performance, since requests are processed sequentially and the arrival rate is independent of individual performance requirements. TDRL improves deadline satisfaction by approximately 57%-over MLP-DRL and 45% over GNN-DRL, respectively. The performance gains can be attributed to Transformer self-attention global dependency modeling and enhanced exploration.

Figures 5b and 5c examine the impact of VNF dependencies on composition optimality, measured by the average DoF. As described in Section IV, lower DoF values indicate stronger interdependencies among VNFs, limiting SFC composition flexibility, whereas higher DoF values imply weaker dependencies (i.e., greater flexibility in the composition space). Figure 5b shows that BW consumption varies significantly with DoF; as dependency decreases (higher DoF), the proposed method consistently achieves lower BW consumption than the baselines, demonstrating superior composition decisions under varying dependency constraints. Figure 5c also demonstrates that, as the degree of freedom increases, compute resource consumption decreases, and vice versa, along with the corresponding 95% confidence intervals.

An ablation study comparing the conventional PPO DRL model and the TDRL can be derived from the performance comparison metrics in Figures 4 to 5, which demonstrate the contribution of multi-head attention to improved decision quality. The results show reductions of 35–45% in bandwidth consumption and 30–38% in computational demand, indicating that attention-based approaches are robust to inter-VNF dependencies modeling and sample-efficient learning.

The time complexity of GNN-DRL is $O\big(\ell(|V| \cdot d_{\text{ff}}^2 + |E| \cdot d_{\text{ff}})\big)$, while TDRL incurs $O\big((|V|^2 h + |V|)d_h\big)$ due to attention-based pairwise interactions. Although this introduces additional overhead, it is offset by faster convergence and improved policy performance, as shown in Figure 4a. In practice, the inference stage involves a single forward pass through the trained actor network, resulting in low decision latency per SFC request.

VII. Conclusion

This paper addresses the problem of network-state-adaptive, context-aware Network Service composition in dynamic 6G environments, where SFCs are constructed under evolving traffic demands and network infrastructure constraints. We proposed a TDRL framework that captures inter-VNF dependencies and variable-length service structures to enable context-aware, resource-efficient composition decisions. The experimental results demonstrate that the proposed method improves bandwidth efficiency, increases the number of SFC requests that meet deadline requirements, and achieves more efficient resource utilization compared to baseline approaches. For future work, we plan to extend to distributed generative multi-agent RL for scalable, efficient composition decisions across edge-cloud continuum network infrastructures.

References

[1] H. Hawilo, A. Shami, M. Mirahmadi, and R. Asal, "Nfv: state of the art, challenges, and implementation in next generation mobile networks (vepc)," *IEEE network*, vol. 28, no. 6, pp. 18–26, 2014.

[2] A. F. Ocampo, J. Gil-Herrera, P. H. Isolani, M. C. Neves, J. F. Botero, S. Latré, L. Zambenedetti, M. P. Barcellos, and L. P. Gaspary, "Optimal service function chain composition in network functions virtualization," in *IFIP international conference on autonomous infrastructure, management and security*. Springer International Publishing Cham, 2017, pp. 62–76.

[3] K. Ning, H. Wang, Z. Zhang, Z. Xu, and X. Shu, "Parallel deployment of vnfs in service function chain: Benefit or not?" in *2022 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)*. IEEE, 2022, pp. 628–635.

[4] S. Bian, X. Huang, Z. Shao, X. Gao, and Y. Yang, "Service chain composition with resource failures in nfv systems: A game-theoretic perspective," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 224–239, 2020.

[5] I. Al Ridhawi, M. Aloqaily, Y. Kotb, Y. Al Ridhawi, and Y. Jararweh, "A collaborative mobile edge computing and user solution for service composition in 5g systems," *Transactions on Emerging Telecommunications Technologies*, vol. 29, no. 11, p. e3446, 2018.

[6] Z. Ning, N. Wang, and R. Tafazolli, "Deep reinforcement learning for nfv-based service function chaining in multi-service networks," in *2020 IEEE 21st International Conference on High Performance Switching and Routing (HPSR)*. IEEE, 2020, pp. 1–6.

[7] A. Moustafa and T. Ito, "A deep reinforcement learning approach for large-scale service composition," in *PRIMA 2018: Principles and Practice of Multi-Agent Systems: 21st International Conference, Tokyo, Japan, October 29-November 2, 2018, Proceedings 21*. Springer, 2018, pp. 296–311.

[8] S. F. Wassie, A. Di Maio, and T. Braun, "Deep reinforcement learning for context-aware online service function chain deployment and migration over 6g networks," in *Proceedings of the 40th ACM/SIGAPP Symposium on Applied Computing*, 2025, pp. 1361–1370.

[9] G. Sun, W. Xie, D. Niyato, F. Mei, J. Kang, H. Du, and S. Mao, "Generative ai for deep reinforcement learning: Framework, analysis, and use cases," *IEEE Wireless Communications*, 2025.

[10] A. F. Ocampo, J. Gil-Herrera, P. H. Isolani, M. C. Neves, J. F. Botero, S. Latré, L. Zambenedetti, M. P. Barcellos, and L. P. Gaspary, "Optimal service function chain composition in network functions virtualization," in *IFIP international conference on autonomous infrastructure, management and security*. Springer International Publishing Cham, 2017, pp. 62–76.

[11] C. Huang and N. Crespi, "A service composition model for dynamic service creation and update in ims/web 2.0 converged environment," in *Proceedings of the 6th Asian Internet Engineering Conference*, 2010, pp. 95–102.

[12] C. S.-H. Hsu, A. Dalgkitsis, C. Papagianni, and P. Grosso, "Transformer-empowered actor-critic reinforcement learning for sequence-aware service function chain partitioning," *arXiv preprint arXiv:2504.18902*, 2025.

[13] D. Ardagna and B. Pernici, "Adaptive service composition in flexible processes," *IEEE Transactions on Software Engineering*, vol. 33, no. 6, pp. 369–384, 2007.

[14] S. Bian, X. Huang, Z. Shao, X. Gao, and Y. Yang, "Service chain composition with resource failures in nfv systems: A game-theoretic perspective," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 224–239, 2020.

[15] X. Wang, H. Xu, X. Wang, X. Xu, and Z. Wang, "A graph neural network and pointer network-based approach for qos-aware service composition," *IEEE Transactions on Services Computing*, vol. 16, no. 3, pp. 1589–1603, 2022.

[16] D. Heo, S. Lange, H.-G. Kim, and H. Choi, "Graph neural network based service function chaining for automatic network control," in *2020 21st Asia-Pacific Network Operations and Management Symposium (APNOMS)*, 2020, pp. 7–12.

[17] "6g-cloud project: Deliverables," https://www.6g-cloud.eu/deliverables/, accessed: 2026-02-28.

[18] D. J. Klein and M. Randić, "Innate degree of freedom of a graph," *Journal of Computational Chemistry*, vol. 8, no. 4, pp. 516–521, 1987.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.