Adaptive Multiple Access and Service Placement for Generative Diffusion Models

Hamidreza Mazandarani[1], Mohammad Farhoudi[2], Masoud Shokrnezhad[3], and Tarik Taleb[1]
[1] *Ruhr University Bochum, Bochum, Germany;* { *hamidreza.mazandarani, tarik.taleb*}*@rub.de*
[2] *Oulu University, Oulu, Finland; mohammad.farhoudi@oulu.fi*
[3] *ICTFICIAL Oy, Espoo, Finland*; masoud.shokrnezhad@ictficial.com

***Abstract*—Generative Diffusion Models (GDMs) have emerged as key components of Generative Artificial Intelligence (GenAI), offering unparalleled expressiveness and controllability for complex data generation tasks. However, their deployment in real-time and mobile environments remains challenging due to the iterative and resource-intensive nature of the inference process. Addressing these challenges, this paper introduces a unified optimization framework that jointly tackles service placement and multiple access control for GDMs in mobile edge networks. We propose LEARN-GDM, a Deep Reinforcement Learning-based algorithm that dynamically partitions denoising blocks across heterogeneous edge nodes, while accounting for latent transmission costs and enabling adaptive reduction of inference steps. Our approach integrates a greedy multiple access scheme with a Double and Dueling Deep Q-Learning (D3QL)-based service placement, allowing for scalable, adaptable, and resource-efficient operation under stringent quality of service requirements. Simulations demonstrate the superior performance of the proposed framework in terms of scalability and latency resilience compared to conventional monolithic and fixed chain-length placement strategies. This work advances the state of the art in edge-enabled GenAI by offering an adaptable solution for GDM services orchestration, paving the way for future extensions toward semantic networking and co-inference across distributed environments.**

***Index Terms*—Generative Diffusion Model (GDM), Generative Artificial Intelligence (GenAI), inference, service placement, multiple access, mobile edge networks, Deep Reinforcement Learning**

I. Introduction

Generative Diffusion Models (GDMs) have recently emerged as a compelling framework within Generative Artificial Intelligence (GenAI), enjoying the distinctive capability of generating high-quality outputs by progressively denoising stochastic input data [1] (Fig. 1). Their inherent expressiveness and controllability make them particularly suitable for complex generative tasks, leading to their integration into communication systems for tasks such as channel-distortion-aware image reconstruction [2], image manipulation [3], and intelligent resource orchestration [4], [5]. Furthermore, GDMs are being recently explored for diffusion-based reasoning [6]. In vehicular networks, one of their applications involves the improvement of road intelligence to facilitate immersive in-vehicle experiences, which include the generation of real-time three-dimensional content [7]. Despite their versatility, GDM services present significant challenges for real-time and mobile deployments due to their computationally intensive inference process arising from their iterative nature [1]. Addressing this bottleneck has emerged as a central theme in recent literature, with approaches that include sampling step reduction and model compression, in conjunction with auto-regressive techniques [8], as well as more systemic solutions that leverage the substantial computing power of edge nodes [9]–[13].

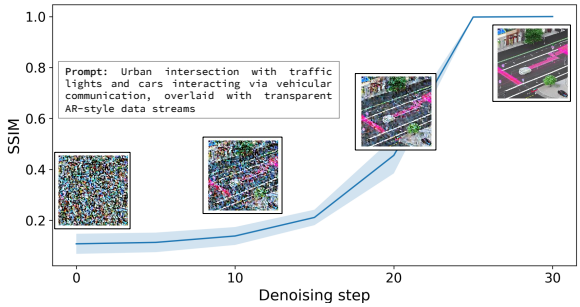

Fig. 1. An example of GDM inference utilizing Stable Diffusion. The image quality, quantified by the Structural Similarity Index Measure (SSIM), gradually enhances during the denoising process. The shaded area illustrates the standard deviation computed across 10 distinct prompts.

In the latter category, the inherent gradual nature of the denoising process is acknowledged, and edge computing is seen as a promising enabler, offering low-latency access to computational resources for mobile users. In this regard, Feng *et al.* [9] proposed an edge-user collaborative GDM inference framework in which a proportion of the denoising steps can be offloaded to the edge servers. Also, Quality of Experience (QoE) for users is considered, which is defined as image quality minus weighted latency and energy consumption. Xie *et al.* [7] introduced a framework for collaborative fine-tuning and distributed inference in vehicular networks, with a splitting strategy of inferences to optimize latency and content-generation capability. Similarly, in the proposal of Yang *et al.* [10], the inference process for each user was split into two phases with an optimizable split point: a shared model for low-level generation at the edge, followed by personalized user models. Zeng *et al.* [11] adopted a different approach by partitioning multi-modal content and offloading partial diffusion tasks to multiple servers. Lie *et al.* [12] proposed a reinforcement learning algorithm that leverages diffusion models and context-aware attention to improve multi-type task orchestration at the network edge. Finally, FlexGen [13]

sought to improve quality and cost adjustability by modulation of model width (i.e., the layer size).

Nevertheless, these efforts often fall short of addressing the dynamic and mobile nature of real-world users navigating through networks [14], [15]. Particularly, the potential for distributing the denoising steps of GDM services across various heterogeneous edge nodes, while considering the latent transmission costs and adaptable chain lengths, remains underexplored. In addition, existing works do not account for multiple access policies that coordinate how users share transmission channels to avoid collisions and ensure fairness under resource constraints [16], [17]. Building on our prior work on joint design of communications and computation [18]–[21], we propose an adaptable framework that jointly optimizes service placement and multiple access control for GDMs in mobile edge networks. We consider a group of Base Stations (BSs) equipped with computing resources, where mobile users intermittently request GDM-generated outputs. The framework aims to deliver the highest possible quality for users and is influenced by two primary cost factors: (i) placement costs, representing the energy and resource overhead of deploying services at edge nodes, and (ii) transmission costs, typically associated with latency requirements. Our system dynamically allocates GDM services to edge nodes while managing multi-user access to shared wireless channels. Our main contributions are outlined as follows:

- Unified optimization framework for joint channel allocation to users (for prompt/initial-condition transmission) and placement of GDM services on edge-computing-equipped BSs. Notably, GDMs are iterative and quality-progressive, produce large intermediate latents, and thus impose strict ordering, latency, and communication–compute coupling that require resource allocation.
- LEARN-GDM: a decision-making algorithm that partitions denoising blocks across heterogeneous nodes, incorporates latent transmission costs among them, and enables adaptive reduction of denoising steps to balance performance and resource consumption.
- Simulations demonstrating scalability and flexibility by evaluating (i) the impact of available access channels on user-request quality and (ii) quality maintenance under increasing numbers of simultaneous requests.

In the remainder of this paper, Section II introduces the system model and formulates the optimization framework. Section III details the proposed approach to service placement and multiple access control. The numerical results are presented in Section IV, while final remarks and directions for future research are presented in Section V.

II. Problem Definition

A. Glimpse into GDM

A GDM service can be characterized as a combination of forward and reverse processes. The forward process maps the initial state, denoted by x_0, to a noise vector $x_{\mathcal{B}}$ (that is sampled from a Gaussian prior)[1]. The reverse process denoises it to its original state ($x_{\mathcal{B}} \to x_{\mathcal{B}-1} \to \ldots \to x_0$), conditioned on a prompt or guiding signal c and utilizing learned transition kernels. This process, parameterized by θ, is

$$p_\theta(x_{t-1} \mid x_t, c) := \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t, c), \Sigma_\theta(x_t, t, c)), \tag{1}$$

where μ_θ and Σ_θ denote the learnable mean and variance of the reverse Gaussian transition kernel, respectively. Moreover, $\mathcal{B}$ signifies the number of denoising step blocks, with $\mathbb{B} = \{1, \ldots, \mathcal{B}\}$ representing the set of their blocks, with $\Omega^k(x)$ output quality of the k-th block for input data x. Notably, a higher number of blocks provides more degrees of freedom for service placement, and yet enlarges the problem size. In the example illustrated in Fig. 2-B, the output quality for various blocks is presented.

In this study, we assume that each block requires a single time frame for execution; however, this framework can be extended to scenarios in which the execution time of each block exceeds one time frame. Accordingly, it takes $\mathcal{B}$ time frames to fully complete a service, while executing only the first $K \leq \mathcal{B}$ blocks of service is possible, aiming to trade off quality for resource efficiency. In addition, it is possible to execute different blocks in distinct BSs, based on available resources and UEs' mobility patterns.

B. System Model

The system consists of a set of heterogeneous edge-computing-equipped BSs, denoted with $\mathbb{N} = \{1, \ldots, \mathcal{N}\}$, capable of providing GDM services to a collection of $\mathcal{U}$ mobile User Equipment (UE). This heterogeneity arises from various BS types, such as roadside units (RSUs), macro, and micro nodes, each with distinct computational capacities. UEs are uniquely labeled as u_i, where $i \in \mathbb{U} = \{1, \ldots, \mathcal{U}\}$. Fig. 2 depicts a modest scenario with $\mathcal{N} = 2$ and $\mathcal{U} = 4$. To execute each service with k blocks, it is necessary to specify its *execution path*, which is defined as the sequence of k nodes $(n_1, \ldots, n_k)$, with each node representing the executing BS of its corresponding block. This sequence is denoted by $\mathfrak{p} \in \mathbb{P} = \bigcup_{k \in \mathbb{B}} \mathbb{P}^k$, where $\mathbb{P}^k$ represents the set of all k-permutations of $\mathbb{N}$ with repetitions permitted[2]. In this regard, the inclusion of node n in path $\mathfrak{p}$ at step k is represented with $\mathcal{J}^k_{\mathfrak{p},n}$. In the scenario of Fig. 2 with $\mathcal{B} = 2$, set $\mathbb{P}^2 = \{(1,1), (1,2), (2,1), (2,2)\}$, where the first path executes two blocks on BS 1 and the second path executes on BS 1 and 2 respectively. For the second path (i.e., $(1,2)$), $\mathcal{J}^1_{2,1}$ and $\mathcal{J}^2_{2,2}$ are equal to 1. Table I consolidates the aforementioned symbols and the others discussed hereafter.

[1]The forward process is employed during the training phase and is not within the scope of this study [1], [4].

[2]Evidently, this set is composed of $\mathcal{N}^k$. For example, even in a small scenario with only four nodes and five blocks, $4^5 = 1024$ paths exist. Therefore, a subset of existing paths should be considered in practice.

TABLE I
NOTATIONS OF SYMBOLS USED IN THE SYSTEM MODEL.

Symbol	Description
u_i	UE with index $i \in \mathbb{U} = \{1, \ldots, \mathcal{U}\}$
n	BS index $\in \mathbb{N} = \{1, \ldots, \mathcal{N}\}$
s	GDM service index $\in \mathbb{S} = \{1, \ldots, \mathcal{S}\}$
c	Communication channel index $\in \mathbb{C} = \{1, \ldots, \mathcal{C}\}$
t	Time frame index $\in \mathbb{T}$
$\mathbb{B}$	Set of blocks in a service, $\{1, \ldots, \mathcal{B}\}$
$\mathbb{P}$/ $\mathbb{P}^k$	Set of all / k-length execution paths
$\mathfrak{p}$	A specific execution path $\in \mathbb{P}$
$\mathcal{J}^k_{\mathfrak{p},n} \in \{0,1\}$	Indicator if n is used in path $\mathfrak{p}$ at step k
$\Psi = [\psi^t_{i,n}]$	Association graph: 1 if u_i connected to n at t
$\Lambda = [\lambda_{i,s}]$	UE to service assignment matrix
$r^t_{i,\mathfrak{p}} \in \{0,1\}$	UE i selects path $\mathfrak{p}$ at t
$e^t_{i,k,n} \in \{0,1\}$	block k of UE i's service, executed on BS n at t
$m^t_{i,c} \in \{0,1\}$	UE i uploads on channel c at t
$m^t_i \in \{0,1\}$	UE i uploads on any channel at t
W^t_n/ $\hat{W}_n$	Actual / Max blocks executed on BS n at t
$\Omega_s(k)$	Service s output quality with k blocks execution
Q^t_i/ $\overline{Q}_i$	Received / Min required output quality for UE i
$\hat{Y}_{n,n'}$	Cost of transmission from node n to n'
Y^t_i	Total transmission cost for UE i at t
ϵ_n	Execution cost of a single inference on BS n
α, β	Execution and Transmission cost trade-off

The system is divided into a total of $\mathcal{A}$ predefined service areas, which vary in size due to geographical factors such as obstacles. UEs move dynamically between areas, but for simplicity, they are assumed to stay within a single area *during* each time frame $t \in \mathbb{T}$. These mobilities create a dynamic association graph $\Psi = [\psi^t_{i,n}]_{\mathcal{U}\times\mathcal{N}\times\mathcal{T}}$, equals 1 if u_i is connected to BS n at time frame t, otherwise 0. In the example of Fig. 2, the association graph $\Psi^t = \big[[1,0],[1,0],[1,0],[0,1]\big]$. UEs in the same area are engaged in contention for access to a set of $\mathbb{C} = \{1, \ldots, \mathcal{C}\}$ perfectly time-slotted communication channels designated for uploading their data to BSs. Consequently, their simultaneous transmissions over a single channel lead to a collision. In contrast, service responses are delivered to UEs via collision-free downlink channels. It is also assumed that BSs have broadband channels between them.

The system offers a collection of GDM services represented by the set $\mathbb{S} = \{1, \ldots, \mathcal{S}\}$. We consider each service s as a trained and ready-to-use GDM model. UEs have been assigned to services through a predefined matrix $\Lambda = [\lambda_{i,s}]_{\mathcal{U}\times\mathcal{S}}$. This matrix is constructed through a scalable and dynamic service discovery mechanism [19]. In the example of Fig. 2, matrix $\Lambda = \big[[1,0,0],[1,0,0],[0,1,0],[0,0,1]\big]$.

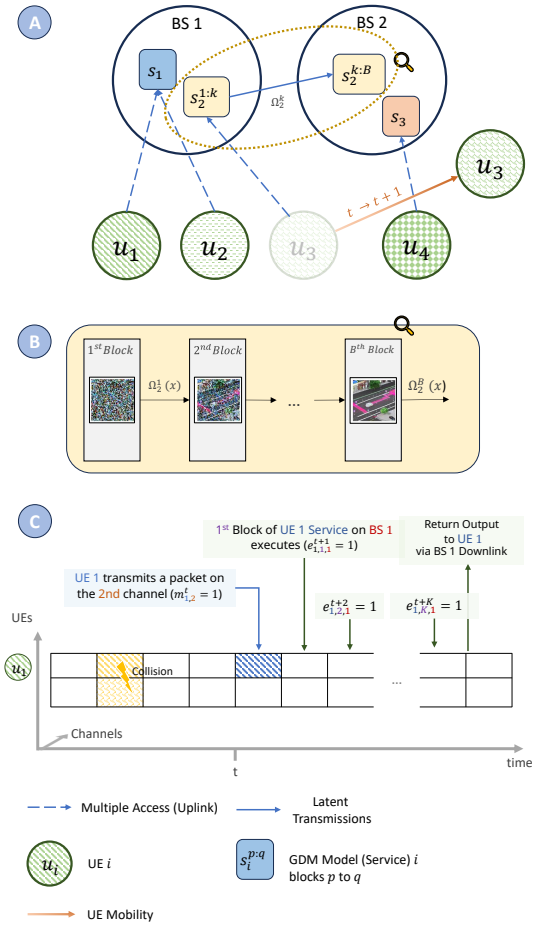

Fig. 2. Sample scenario with $\mathcal{N} = 2$, $\mathcal{U} = 4$ and $\mathcal{S} = 3$. UEs are fixed, except u_3, which changes its associated BS between two time frames. **A**) u_1 and u_2 are both registered to s_1, and u_3 and u_4 are registered to s_2 and s_3, respectively. **B**) GDM Model of s_2 with maximum $\mathcal{B}$ denoising step blocks. **C**) The first transmission attempt of u_1 resulted in a collision, while their next packets were transmitted successfully, resulting in the execution of the corresponding service. All blocks of s_1 are executed in BS 1. On the other hand, the last $(\mathcal{B}-k)$ blocks of s_2 are executed in BS 2 so that by generating the final synthetic data, u_3 is able to receive it from the associated BS.

C. Problem Formulation

To fulfill the services requested by UEs, the initial step is to determine the execution path for each UE. The binary decision variable $r^t_{i,\mathfrak{p}}$ indicates selection of execution path $\mathfrak{p}$ for u_i at the beginning of time frame t. Moreover, the support variable $e^t_{i,k,n}$ is responsible for indicating whether block k of UE i's service is executed on the BS n at time frame t. The variables r and e are subject to constraint C1, which specifies that the selection of a path of length k necessitates the execution of all k blocks of the associated service at the corresponding nodes over the subsequent k time frames. Hereafter, we assume out-of-bounds indices are 0 for simplicity. Moreover, each UE can select only one path per time frame (C2), and each BS n can execute maximum $\hat{W}_n$ blocks per time frame (C3)[3].

[3]While all blocks of a GDM service are implemented with the same neural network, they require separate inferences and thus separate processing power.

$$r_{i,\mathfrak{p}}^t \leq \sum_{n\in\mathbb{N}} e_{i,k,n}^{t+k-1} \cdot \mathcal{J}_{\mathfrak{p},n}^k \quad \forall i \in \mathbb{U}, \mathfrak{p} \in \mathbb{P}, k \in \mathbb{B}, t \in \mathbb{T} \tag{C1}$$

$$\sum_{\mathfrak{p}\in\mathbb{P}} r_{i,\mathfrak{p}}^t \leq 1 \qquad \forall i \in \mathbb{U}, \forall t \in \mathbb{T} \tag{C2}$$

$$W_n^t \triangleq \textstyle\sum_{i\in\mathbb{U},k\in\mathbb{B}} e_{i,k,n}^t \leq \hat{W}_n \qquad \forall n \in \mathbb{N}, t \in \mathbb{T} \tag{C3}$$

Another binary decision variable is $m_{i,c}^t$, which indicates the upload transmission of UE i on channel c at time frame t. Each UE can transmit only on one channel, and no more than one of the UEs connected to the same BS may transmit data simultaneously, as stated by channel constraints in C4 and C5, respectively.

$$m_i^t \triangleq \sum_{c\in\mathbb{C}} m_{i,c}^t \leq 1 \qquad \forall i \in \mathbb{U}, t \in \mathbb{T} \tag{C4}$$

$$\sum_{i\in\mathbb{U}} m_{i,c}^t \cdot \psi_{i,n}^t \leq 1 \qquad \forall n \in \mathbb{N}, c \in \mathbb{C}, t \in \mathbb{T} \tag{C5}$$

In order to establish an end-to-end transmission for each UE and its designated execution path, it is essential to correlate the variables e and m. Constraint C6 facilitates this linkage by stipulating that the initial block of the corresponding service may only be executed if a request is present in the preceding time frame $t-1$. Recall that the initial block starts the denoising process from noise, with the UE prompt serving as a required conditioning input.

$$\textstyle\sum_{\mathfrak{p}\in\mathbb{P}} r_{i,\mathfrak{p}}^t \leq \sum_{c\in\mathbb{C}} m_{i,c}^{t-1} \qquad \forall i \in \mathbb{U}, t \in \mathbb{T} \tag{C6}$$

Moreover, the end-to-end assignment of each UE must be ensured in terms of output quality. We define in C7 the quality of synthetic data generated for UE i at time frame t as the number of executed blocks applied to the function $\Omega_s(.)$, which generally increases with k. In practice, a minimum acceptable quality is set, represented by $\overline{Q}$ in C8. For example, in Fig. 1, if $\overline{Q} = 0.5$, delivering the output at step 10 offers no advantage. Notably, C8 must be satisfied if at least one path is selected for the UE.

$$Q_i^t \triangleq \sum_{s\in\mathbb{S},\mathfrak{p}\in\mathbb{P}} r_{i,\mathfrak{p}}^t \cdot \lambda_{i,s} \cdot \Omega_s(|\mathfrak{p}|) \tag{C7}$$

$$\geq \overline{Q}_i \cdot \sum_{\mathfrak{p}\in\mathbb{P}} r_{i,\mathfrak{p}}^t \qquad \forall i \in \mathbb{U}, t \in \mathbb{T} \tag{C8}$$

The final step is to establish a metric to measure the costs of intermediate latent data transfer between executing nodes, along with the transfer of UE data from the Point of Attachment (PoA) to the first execution node and the generated data from the last execution node to the final PoA (C9). Note that in this formulation, the execution path head and tail can differ from PoA nodes, but their negative impact on the transmission cost is considered. Here, $\hat{Y}_{n,n'}$ is the cost of transmitting data from node n to n'.

$$\begin{aligned} Y_i^t = & \sum_{\substack{\mathfrak{p},(n,n'),k \\ \in \mathbb{P},\mathbb{N}\times\mathbb{N},\mathbb{B}}} r_{i,\mathfrak{p}}^t \cdot \mathcal{J}_{\mathfrak{p},n}^k \cdot \mathcal{J}_{\mathfrak{p},n'}^{k+1} \cdot \hat{Y}_{n,n'} \\ & + \sum_{\substack{(n,n'),\mathfrak{p} \\ \in \mathbb{N}\times\mathbb{N},\mathbb{P}}} r_{i,\mathfrak{p}}^t \Big(\psi_{i,n}^{t-1} \cdot \mathcal{J}_{\mathfrak{p},n'}^1 + \psi_{i,n}^{t+|\mathfrak{p}|} \cdot \mathcal{J}_{\mathfrak{p},n'}^{|\mathfrak{p}|} \Big) \cdot \hat{Y}_{n,n'} \\ & \forall i \in \mathbb{U}, t \in \mathbb{T} \end{aligned} \tag{C9}$$

Now, we can set the objective of our problem to maximize the total quality of UEs over all time frames, subtracted by the scaled total cost of service execution and data transfer. Here, ϵ_n is the cost of a single inference on node n. Given that this objective exhibits non-linearity and certain parameters, such as $\Omega_s(.)$ (i.e., the quality per block function), are not practically known, traditional gradient-based optimization techniques may struggle to converge to an optimal solution.

$$\begin{aligned} & \max \sum_{i\in\mathbb{U},t\in\mathbb{T}} Q_i^t - \alpha \cdot \sum_{n\in\mathbb{N},t\in\mathbb{T}} \epsilon_n \cdot W_n^t - \beta \cdot \sum_{i\in\mathbb{U},t\in\mathbb{T}} Y_i^t \\ & \text{s.t. (C1-C9)} \end{aligned} \tag{2}$$

III. Proposed Scheme

To address the previously mentioned challenges associated with problem complexities and incomplete knowledge, we propose muLtiplE Access and seRvice placemeNt for GDMs (LEARN-GDM), an intelligent resource allocation algorithm that allocates channels and places services per time frame. LEARN-GDM operates effectively despite uncertain system information since it incorporates user mobility prediction and proactively deploys services closer to where users are likely to move. LEARN-GDM consists of a greedy Multiple Access Algorithm (MAC) and Double and Dueling Deep Q-Learning (D3QL)-based service placement [14], [20], [22]. Deep Reinforcement Learning, including D3QL as a value-based method, has been extensively utilized to address complex network optimization challenges, even with limited knowledge of wireless networks, while continuously learning and enhancing understanding of the environment [23].

The MAC component in LEARN-GDM greedily assigns channels to UEs connected to the same BS, prioritizing those whose ongoing inference processes are closest to the quality threshold. For example, between two UEs with ongoing qualities 0.3 and 0.4, the UE with 0.4 has higher priority when the threshold is 0.5; if the threshold is 0.25, both would have equal priority.

D3QL enhances Deep Q-Learning (DQL) by decoupling action selection and evaluation [24] using the target value defined in (3), followed by integrating Wang *et al.*'s dueling approach [25]. This target incorporates reward ρ, observation O, real action $a \in \boldsymbol{\mathcal{A}}$, and predicted action a' to update DQL weights ($\mathcal{W}$) for each observation-action of time t (O^t, a^t). Here, $a' = \text{argmax}_{a\in\boldsymbol{\mathcal{A}}} Q(O^{t+1}, a; \mathcal{W}^t)$ with $\mathcal{W}$ representing evaluation weights updated at each step, and $\mathcal{W}^-$ as target

network weights, synchronized every $\hat{t} \gg 0$ step. Moreover, separate estimators calculate state values ($\mathcal{V}$) and action advantages ($\mathcal{AD}$), combining them to compute Q-values (4) and weights (5). This approach improves training stability, accelerates convergence, and mitigates overestimation issues.

$$Y^t = \rho^t + \gamma \cdot Q(O^{t+1}, a'; \mathcal{W}^{t-}) \tag{3}$$

$$Q(O^t, a^t; \mathcal{W}^t) = \mathcal{V}(O^t; \mathcal{W}^t) + \left(\mathcal{AD}(O^t, a^t; \mathcal{W}^t) - \frac{1}{|\boldsymbol{\mathcal{A}}|} \sum_{a' \in \boldsymbol{\mathcal{A}}} \mathcal{AD}(O^t, a'; \mathcal{W}^t) \right) \tag{4}$$

$$\mathcal{W}^{t+1} \leftarrow \mathcal{W}^t + \sigma \cdot [Y^t - Q(O^t, a^t; \mathcal{W}^t)] \nabla_{\mathcal{W}^t} Q(O^t, a^t; \mathcal{W}^t) \tag{5}$$

To define action space ($\boldsymbol{\mathcal{A}}$), it determines which BS (if any) should initialize or continue a process for the next time frame. A non-zero action for UE i signifies its continuity if it has already started but not yet finished. More concretely, if block $k < \mathcal{B}$ is executed in the current time frame, a non-zero value indicates that block $k+1$ will be executed in the next time frame; otherwise, the first block will be executed. Conversely, a zero value prevents starting a new inference process or stops an existing service. When a service stops, the generated image is delivered to the corresponding UE.

$$\boldsymbol{\mathcal{A}} = \left\{ a_i : \emptyset \cup \mathbb{N} \mid \forall i \in \mathbb{U} \right\} \tag{6}$$

Observation space must provide the resource allocator with sufficient information regarding the current and historical state of the allocated resources, as well as the ongoing state of each chain and multiple access information.

$$\begin{aligned} o^t &= \left\{ W_n / \hat{W}_n, \epsilon_n | n \in \mathbb{N} \right\} \cup \left\{ Q_i^t - \overline{Q}_i | i \in \mathbb{U} \right\} \cup \left\{ m_i^{t-1} | i \in \mathbb{U} \right\} \\ &\quad \cup \left\{ \psi_{i,n}^t | n \in \mathbb{N}, i \in \mathbb{U} \right\} \\ O^t &= \left\{ o^h | h \in \{ t - \mathcal{H}, \ldots, t \} \right\} \end{aligned} \tag{7}$$

To circumvent the problem of delayed rewards [26], we reward users based on their image quality increase resulting from successful allocations and constrained to exceed the quality threshold, subtracted by the scaled cost of allocations and transmissions.

$$\begin{aligned} \rho^t = &\textstyle\sum_{i \in \mathbb{U}} \left((Q_i^t - Q_i^{t-1}) \cdot \mathbb{1}(Q_i^t \geq \overline{Q}_i) \right) \\ &- \alpha \cdot \left(\textstyle\sum_{n \in \mathbb{N}} \epsilon_n \cdot W_n^t \right) - \beta \cdot \left(\textstyle\sum_{i \in \mathbb{U}} Y_i^t \right) \end{aligned} \tag{8}$$

Our approach, outlined in Algorithm 1, consists of several key components. Steps 4–8 implement the greedy MAC algorithm, while steps 10–14 apply an epsilon-greedy action selection strategy for service placement. For each UE, the placement procedure (steps 16–21) is executed as follows: the system first checks whether the maximum chain length has been reached. If it has, the inference process concludes, and the result is delivered to the UE. If not, the placement decision depends on the selected action–specifically, whether the action assigns a service placement to the UE–and the current node capacity status. If inference has already been initiated, the placement advances to the next block; if not, it begins at the first block. In cases where the action is null or the node's capacity is exhausted, any available latent data is sent to the UE. Finally, steps 23–28 address the D3QL agent training phase.

Algorithm 1: muLtiplE Access and seRvice placemeNt for GDMs (LEARN-GDM)

Input: $\mathcal{T}$, Set of Episodes ($\mathbb{E}$),
2 $\mathcal{W} \leftarrow \mathbf{0}$, $\mathcal{W}^- \leftarrow \mathbf{0}$, $\epsilon \leftarrow 1$, $memory \leftarrow \{\}$
foreach *ep in* $\mathbb{E}$ **do**
3 **foreach** t *in* $\mathbb{T}$ **do**
4 ⋆ Multiple Access ⋆
UE Priorities $\leftarrow \max\left\{1/(\overline{Q} - Q^t), 10^{-8}\right\}$
$\mathbb{U}_{sorted} \leftarrow \mathit{SORT}_u\{\mathbb{U}, \mathit{key} = \text{UE Priorities}\}$
foreach c, i *in* $\mathbb{C}, \mathbb{U}_{sorted}[1:\mathcal{C}]$ **do**
5 $m_{i,c}^t \leftarrow 1$
6 ⋆ Service Placement ⋆
$\zeta \leftarrow$ sample uniformly from *Uniform*$(0,1)$
if $\zeta > \epsilon$ **then**
7 $a^t \leftarrow \text{argmax}_{a' \in \boldsymbol{\mathcal{A}}} Q(O^{t+1}, a'; \mathcal{W}^t)$
8 **else**
9 select a random a^t from $\boldsymbol{\mathcal{A}}$
10 **foreach** i *in* $\mathbb{U}_{sorted}$ **do**
11 **if** *Maximum blocks ($\mathcal{B}$) has reached* **then**
12 Deliver the result to the UE
13 **else if** $(a_i^t = n \in \mathbb{N}) \wedge (W_n < \hat{W}_n)$ **then**
14 Deploy the first/next block on n
15 **else**
16 Deliver the result (if available) to UE
17 Observe ρ^t and construct O^{t+1} acc. to (7)
⋆ Training ⋆
$memory \leftarrow memory \cup \{(\rho^t, O^t, a^t, O^{t+1})\}$
Choose a batch of samples from $memory$
Train the agent according to (5)
if $\epsilon > \widetilde{\epsilon}$ **then**
18 $\epsilon \leftarrow \epsilon \cdot \epsilon'$

IV. Performance Evaluation

In this section, we conduct a numerical analysis of the D3QL-based solution using the parameters outlined in Table II. As a first step, we analyze convergence, followed by a comparison of performance. To illustrate the learning process involved in service placement, Fig. 3 displays the service placement reward (blue line) within learning episodes. The learning algorithms are trained over 200,000 time frames. The reward increases gradually, which illustrates the efficiency of DRL in placement. Additionally, rewards stabilize after the 175,000 time frame, indicating convergence.

For comparison, we explore two practical scenarios: 1) the impact of user numbers on scalability; 2) how the number of channels, indicative of a communications bottleneck in real-world applications, affects performance. Notably, in all scenarios, UEs are randomly distributed across the grid and

TABLE II
SYSTEM MODEL PARAMETERS.

Parameter	Value
Network area	4x4 grid
Node Processing Capacity ($\hat{W}$)	$\sim \mathcal{U}(1,3)$
Node Placement cost (ϵ)	$\sim \mathcal{U}(1,4)$ per inference
Quality Threshold ($\overline{Q}$)	$\sim \mathcal{U}(0.1,0.5)$
Number of Services ($\mathcal{S}$)	3
Max. blocks per service ($\mathcal{B}$)	4
Default number of UEs	15
Default number of channels	2
Scaling Factors (α, β)	0.1, 0.1
LSTM history size ($\mathcal{H}$)	3 experiences
Capacity of experience memory	5000 experiences
Batch size	32
Discount factor (γ)	0.9
Learning rate	0.0008
Exploration parameters $\widetilde{\epsilon}$, ϵ'	0.00001, 0.99995
Approximator model	LSTM with 128 units + fully-connected layers with 128, 64 and 32 units
Target network update frequency	Every 150 steps

move according to the Random Waypoint Model, with an average speed of 10 m/s and a pause time of 3 seconds. During comparison, we employ four benchmarks. First, the Monolithic Placement (MP) method, indicated by a purple line, operates with a single node per inference, placing a flexible number of blocks on that node. This method is a relaxed version of the approaches proposed by [12], representing an upper bound for their results in the present analysis. Second, a Fixed Chain Placement (FP), which is also based on the D3QL algorithm, lacks the flexibility of variable chain lengths that are helpful for tradeoffs. Third, a Greedy algorithm (GR), illustrated by a red line, serves as another benchmark by assigning each block to the PoA. Finally, the Optimization algorithm (OPT) solves the problem defined in (2) with full knowledge, utilizing the Gurobi optimization solver [27], and establishes a universal upper bound applicable to all approaches.

1) Number of Users: The number of users reflects the scalability of the system concerning growing demand. In this scenario, we put this under test by setting different values of $\mathcal{U}$. Fig. 4-(A) clearly shows the superiority of our approach over MP, FP, and GR methods. OPT possesses knowledge of the UE movements; therefore, they keep their performance despite heavy loads. Among MP and FP, the comparison depends on the problem scale. In a small number of UEs, MP outperforms, meaning that in such situations, having the option of variable chain length matters more. On the other hand, in a large number of UEs, distributing diffusion blocks on various nodes yields more benefit than setting a variable chain length.

2) Number of Channels: With the proliferation of GenAI services and in particular, GDMs on the edge, it is expected that the communications factors of the system play an important role in service provisioning. By varying the number of available channels (denoted as $\mathcal{C}$), we can assess how

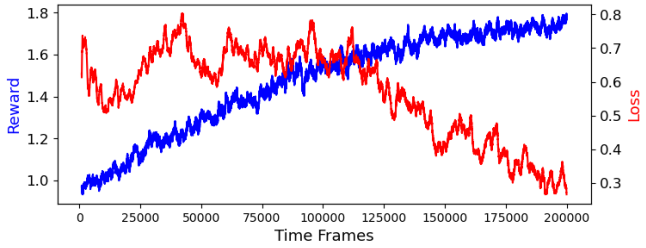

Fig. 3. Reward and Mean Squared Error (MSE) Loss evolution of the service placement learning algorithm over 5,000 training episodes (each with 40 time frames), showing increased rewards and decreased MSE loss over time.

the communications bounds affect system performance. As illustrated in Fig. 4-(B), a reduced number of channels leads to slightly increased collisions, which hinders the users' capacity to transmit their data containing prompts and initial conditions to the executing nodes. However, our approach demonstrated a considerably diminished negative impact in comparison to other methods. This resilience is attributed to its flexibility of variable chain lengths and executing nodes, the lack of which makes the MP, FP, and GR methods struggle.

V. CONCLUSION

This work introduced a unified problem formulation for channel allocation to GenAI users for transmitting their prompts or conditioning inputs, alongside the placement of GDM services on edge-computing-enabled BSs. The problem considered the distribution of denoising blocks across multiple nodes, accounting for latent transmission costs, and potentially reducing the number of denoising steps to balance performance with resource consumption. Moreover, we proposed and evaluated LEARN-GDM, a decision-making algorithm built upon D3QL, which is an enhanced version of Deep Q-Learning. To this end, the state and action spaces, as well as the reward mechanism, were thoroughly customized for the problem at hand. Our analysis demonstrates that the proposed algorithm enhanced overall QoS compared to conventional approaches.

While this study focuses on GDMs, the proposed framework is modifiable to other gradual inference processes, such as DNN partitioning use cases [28] or Large Language Models (LLMs) deployments over space-air-ground integrated networks comprising numerous heterogeneous nodes [15]. Additionally, our research aligns with the ongoing semantic revolution in communication systems, which emphasizes the role of data semantics in achieving task-specific goals [29]. Future work includes extending this approach to semantic networking that enables co-inference—sharing computation blocks across GenAI services for different users. This can be facilitated by incorporating tasks with similar intents into a unified knowledge graph [7]. Last but not least, there is potential to explore GDM resource allocation in a quantum internet in the future, yielding greater sustainability and applicability [30].

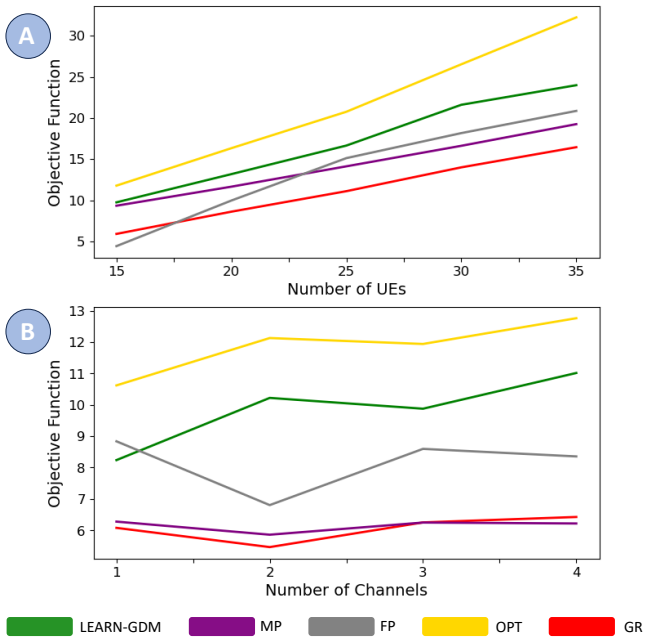

Fig. 4. Performance comparison of LEARN-GDM against baseline methods: Monolithic Placement (MP), Fixed Chain Placement (FP), Optimization algorithm (OPT), and Greedy algorithm (GR) under varying system settings where (A) the number of UEs increases. (B) The number of channels increases.

ACKNOWLEDGMENT

This work is partially conducted at ICTFICIAL Oy. It is partially supported by the European Union's Horizon Europe programme for Research and Innovation through the 6G-SANDBOX project (Grant No. 101096328) and the 6G-Path project (Grant No. 101139172). The studies of M. Farhoudi are partially supported by the Business Finland 6Bridge 6Core project (Grant Number 8410/31/2022). The paper reflects only the authors' views. The European Commission and the Spanish Ministry are not responsible for any use that may be made of the information it contains.

REFERENCES

[1] H. Cao *et al.*, "A survey on generative diffusion models," *IEEE Trans. Knowl. and Data Eng.*, vol. 36, no. 7, pp. 2814–2830, 2024.

[2] M. Letafati *et al.*, "Conditional denoising diffusion probabilistic models for data reconstruction enhancement in wireless communications," *IEEE Trans. on Mach. Learn. Commun. Netw.*, vol. 3, pp. 133–146, 2025.

[3] A. Salar, Q. Liu, Y. Tian, and G. Zhao, "Enhancing facial privacy protection via weakening diffusion purification," in *Proc. IEEE/CVF Conf. Computer. Vis. and Pattern Recognit.*, June 2025, pp. 8235–8244.

[4] H. Du *et al.*, "Enhancing deep reinforcement learning: A tutorial on generative diffusion models in network optimization," *IEEE Commun. Surveys Tuts.*, vol. 26, no. 4, pp. 2611–2646, 2024.

[5] N. C. Luong *et al.*, "Diffusion models for future networks and communications: A comprehensive survey," *arXiv preprint arXiv:2508.01586*, 2025.

[6] J. Ye, S. Gong, L. Chen, L. Zheng, J. Gao, H. Shi, C. Wu, X. Jiang, Z. Li, W. Bi *et al.*, "Diffusion of thought: Chain-of-thought reasoning in diffusion language models," *Adv. Neural Information Process. Syst.*, vol. 37, pp. 105 345–105 374, 2024.

[7] G. Xie *et al.*, "GAI-IoV: Bridging generative AI and vehicular networks for ubiquitous edge intelligence," *IEEE Trans. Wireless Commun.*, vol. 23, no. 10, pp. 12 799–12 814, 2024.

[8] H. Tang *et al.*, "HART: Efficient visual generation with hybrid autoregressive transformer," 2024.

[9] W. Feng *et al.*, "Exploring collaborative diffusion model inferring for AIGC-enabled edge services," *IEEE Trans. on Cogn. Commun. Netw.*, pp. 1–1, 2024.

[10] W. Yang, Z. Xiong, S. Guo, S. Mao, D. I. Kim, and M. Debbah, "Efficient multi-user offloading of personalized diffusion models: A DRL-convex hybrid solution," *IEEE Trans. Mobile Comput.*, 2025.

[11] W. Zeng *et al.*, "Generative AI-aided multimodal parallel offloading for AIGC metaverse service in IoT networks," *IEEE Internet Things J.*, pp. 1–1, 2025.

[12] Y. Liu *et al.*, "QoS-aware multi-AIGC service orchestration at edges: An attention-diffusion-aided DRL method," *IEEE Trans. on Cogn. Commun. Netw.*, pp. 1–1, 2025.

[13] P. Li *et al.*, "Flexgen: Efficient on-demand generative AI service with flexible diffusion model in mobile edge networks," *IEEE Trans. on Cogn. Commun. Netw.*, pp. 1–1, 2024.

[14] H. Mazandarani, M. Shokrnezhad, T. Taleb, and R. Li, "Self-sustaining multiple access with continual deep reinforcement learning for dynamic metaverse applications," in *Proc. IEEE Int. Conf. Metaverse Comput., Netw. and Appl. (MetaCom)*. IEEE, 2023, pp. 65–70.

[15] M. Shokrnezhad and T. Taleb, "An autonomous network orchestration framework integrating large language models with continual reinforcement learning," *IEEE Commun. Mag.*, vol. 63, no. 8, pp. 78–84, 2025.

[16] M. Shokrnezhad, H. Mazandarani, and T. Taleb, "Fairness-utilization trade-off in wireless networks with explainable kolmogorov-arnold networks," in *Proc. IEEE Virtual Conf. Commun. (VCC)*. IEEE, 2024, pp. 1–6.

[17] H. Mazandarani, M. Shokrnezhad, and T. Taleb, "A novel multiple access scheme for heterogeneous wireless communications using symmetry-aware continual deep reinforcement learning," *IEEE Trans. Mach. Learn. Commun. and Netw.*, 2025.

[18] M. Shokrnezhad *et al.*, "Towards a dynamic future with adaptable computing and network convergence (ACNC)," *IEEE Netw.*, 2024.

[19] M. Farhoudi *et al.*, "Discovery of 6G services and resources in edge-cloud-continuum," *IEEE Netw.*, vol. 39, no. 3, pp. 223–232, 2025.

[20] H. Mazandarani *et al.*, "A semantic-aware multiple access scheme for distributed, dynamic 6G-based applications," in *Proc. IEEE Wireless Commun. and Networking Conf.*, 2024, pp. 1–6.

[21] M. Farhoudi *et al.*, "QoS-aware service prediction and orchestration in cloud-network integrated beyond 5G," in *Proc. IEEE Global Telecommun. Conf.*, Dec. 2023, pp. 369–374.

[22] M. Farhoudi, M. Shokrnezhad, S. Kianpisheh, and T. Taleb, "Deep learning based service composition in integrated aerial-terrestrial networks," in *IEEE Int. Conf. on Net. Soft.*, 2025, pp. 204–208.

[23] A. Alwarafy, M. Abdallah, B. S. Ciftler, A. Al-Fuqaha, and M. Hamdi, "The frontiers of deep reinforcement learning for resource management in future wireless hetnets: Techniques, challenges, and research directions," *IEEE Open Journal of the Communications Society*, vol. 3, pp. 322–365, 2022.

[24] H. v. Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," *Proc. of the AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, Mar. 2016.

[25] Z. Wang *et al.*, "Dueling network architectures for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, vol. 48, Jun. 2016, pp. 1995–2003.

[26] H. Li *et al.*, "From hype to reality: The road ahead of deploying DRL in 6G networks," 2024.

[27] Gurobi Optimization, LLC, "Gurobi Optimizer Reference Manual," 2023. [Online]. Available: https://www.gurobi.com

[28] P. Kayal *et al.*, "DNNSplit: Latency and cost-efficient split point identification for multi-tier DNN partitioning," *IEEE Access*, vol. 12, pp. 80 047–80 061, 2024.

[29] M. Shokrnezhad *et al.*, "Semantic revolution from communications to orchestration for 6G: Challenges, enablers, and research directions," *IEEE Netw.*, 2024.

[30] Prados-Garzon *et al.*, "Deterministic 6GB-assisted quantum networks with slicing support: A new 6GB use case," *IEEE Netw.*, vol. 38, no. 1, pp. 87–95, 2023.