Journal of Information Systems and Telecommunication
Vol.14, No.4, July-September 2026, 225-237
http://jist.acecr.org
ISSN 2322-1437 / EISSN:2345-2773

Soft and Safe Network Slicing with Hierarchical Multi-Agent Reinforcement Learning

Zeinab Sasan[1], Siavash Khorsandi[1*]

[1]. Computer Engineering Department, Amirkabir University of Technology (Tehran Polytechnic), Tehran, Iran

Received: 06 Oct 2025 / Revised: 19 Aug 2026 / Accepted: 26 Aug 2026

Abstract

Network slicing has emerged as a key enabler for next-generation mobile networks, allowing multiple virtual network slices to coexist over shared physical infrastructure. Soft network slicing can improve resource utilization by enabling controlled inter-slice resource sharing; however, increased resource sharing may impose a higher burden on donor slices and consequently increase slice dissatisfaction. Achieving efficient resource utilization while controlling slice dissatisfaction and maintaining feasible resource allocation remains challenging under dynamic and heterogeneous traffic demands. In this paper, we propose a Hierarchical Multi-Agent Reinforcement Learning (HMARL) framework for online soft and safe network slicing. The proposed architecture operates at two decision levels: slice-level agents make local admission decisions, while a higher-level meta-controller coordinates inter-slice resource sharing through adaptive borrowing limits. A deterministic safety layer validates the proposed actions before resource allocation and ensures compliance with resource, continuity, and borrowing constraints. The proposed framework is evaluated against learning-based, non-sharing, random, greedy, and heuristic resource-allocation approaches. The training results show that HMARL reaches a stable operating regime with high normalized resource utilization and a low request rejection ratio. Across different scenario classes, HMARL achieves a favorable trade-off between resource utilization and slice dissatisfaction, resulting in competitive overall objective values relative to the considered alternatives. These results indicate that hierarchical coordination combined with deterministic feasibility enforcement provides an effective approach for adaptive resource management in soft network slicing environments.

Keywords: Hierarchical Multi-Agent Reinforcement Learning (HMARL); Soft Network Slicing; Safe Network Slicing; Resource Allocation; Multi-Agent Systems; Safe Reinforment Learning.

1- Introduction

The advent of fifth-generation (5G) and beyond mobile networks has ushered in the era of network slicing, a paradigm that enables multiple logical networks, or slices, to be instantiated on top of a shared physical infrastructure [1]. Each slice is tailored to specific service requirements—such as enhanced mobile broadband, ultra-reliable low-latency communication, or massive machine-type communications—while sharing the same underlying hardware [2], [3]. This flexibility transforms network management but also introduces significant complexity: operators must allocate limited resources among slices dynamically, maintain service-level agreements (SLAs), and maintain an appropriate degree of slice isolation while adapting to heterogeneous and rapidly changing demands [4].

Conventional resource allocation strategies for network slicing typically rely on static provisioning, heuristic admission controls, or optimization models solved offline. While such approaches can be effective in relatively stable settings, they may have difficulty adapting to the variability and unpredictability of dynamic traffic conditions [5], [6]. They may underutilize network capacity, cause unnecessary request rejections, or degrade the quality of service (QoS) of sensitive services [7], [8]. Furthermore, soft and safe network slicing—where controlled resource sharing between slices is permitted while predefined resource-sharing constraints are enforced—poses an even more challenging problem: how to flexibly borrow and lend resources between slices without violating the corresponding resource and borrowing constraints [9], [10], [11].

In this context, reinforcement learning (RL) has emerged as a powerful data-driven approach to sequential decision making under uncertainty [12]. RL allows agents to interact

✉ **Siavash Khorsandi**
khorsandi@aut.ac.ir

with an environment, learn from feedback, and optimize long-term objectives without explicit modeling of the underlying dynamics. Extending RL to multiple cooperating entities leads to multi-agent reinforcement learning (MARL), which is naturally suited to distributed resource allocation problems like network slicing [13]. However, conventional flat MARL architectures—where all agents operate at the same decision layer—may face coordination challenges as the number of slices, actions, or resource constraints increases. They may also converge slowly or produce decisions that require additional constraint handling in dynamic resource allocation environments [14].

To address these challenges, this paper proposes a Hierarchical Multi-Agent Reinforcement Learning (HMARL) framework for online resource management in soft and safe network slicing. The proposed framework decomposes the decision-making process into two hierarchical levels. At the lower level, slice-specific agents evaluate eligible user requests based on their local observations and make admission decisions for individual requests. At the higher level, a meta-controller observes the system-level state and adaptively determines borrowing limits that regulate resource sharing between neighboring slices. This hierarchical structure combines local admission decisions with system-level coordination, allowing the resource-sharing policy to adapt to changing traffic conditions. Before execution, a deterministic safety layer validates the proposed decisions and performs resource allocation while enforcing the predefined resource, borrowing, and continuity constraints.

We evaluate the proposed HMARL framework against several representative alternatives, including HSS, a hierarchical reinforcement-learning baseline; a variant without the meta-controller (w/o Meta); No-Sharing; Random Admission; a greedy heuristic; and HRASS. The evaluation focuses on four key performance metrics: the overall objective value, normalized resource utilization, normalized slice dissatisfaction, and request rejection ratio. In addition, the training behavior of the considered learning-based approaches is examined to assess convergence over the training episodes. The results show that HMARL achieves high resource utilization and a low rejection ratio while maintaining competitive objective values across different scenario classes. Overall, the proposed hierarchical approach provides a favorable trade-off between efficient resource utilization, controlled inter-slice dissatisfaction, and request admission performance.

The contributions of this paper can be summarized as follows:

(1) We formulate online soft network slicing as a hierarchical multi-agent decision-making problem, where slice-level agents generate local admission proposals and a meta-controller coordinates inter-slice resource sharing through adaptive borrowing limits;
(2) We develop an HMARL framework that combines on-policy learning with a deterministic safety layer that validates the proposed decisions and enforces resource, borrowing, and continuity constraints before allocation;
(3) We evaluate the proposed framework against multiple learning-based, heuristic, random, and non-sharing alternatives under heterogeneous traffic scenarios, showing that HMARL achieves high resource utilization, low request rejection, and a favorable utilization–dissatisfaction trade-off.

The remainder of the paper is organized as follows. Section II reviews related work on soft network slicing and reinforcement learning–based resource management. Section III introduces the system model and problem formulation. Section IV describes the proposed HMARL framework together with the baseline methods. Section V presents the experimental setup and evaluation metrics. Section VI discusses the performance results. Finally, Section VII concludes the paper and outlines future research directions.

2- Related Works

In this section, we review the most relevant studies that have addressed network slicing from different perspectives. We first examine the literature on traditional and soft network slicing, highlighting approaches that aim to dynamically allocate resources among multiple virtual slices. We then discuss prior work that leverages multi-agent reinforcement learning (MARL) techniques to manage slice admission and resource sharing in a distributed manner. Finally, we focus on hierarchical MARL frameworks, which introduce additional layers of decision-making to improve coordination and adaptability. This overview positions our proposed HMARL framework within the context of existing research and clarifies how it advances the state of the art.

Sasan et al. [15] proposed the Heuristic Resource Allocation for Soft Slicing (HRASS) framework to overcome the inefficiencies of hard network slicing. Their work formulates soft slicing as a Mixed Integer Linear Programming (MILP) problem with the dual objective of maximizing resource utilization and minimizing slice dissatisfaction. To address the NP-hardness of the problem, the authors develop a heuristic algorithm that achieves near-optimal performance with significantly reduced computational complexity. Experimental results in their study demonstrate that HRASS effectively improves resource utilization while mitigating the limitations of hard slicing.

Li et al. [16] proposed a hierarchical soft-slicing framework for software-defined radio access networks supporting differentiated services with diverse QoS requirements. Their approach splits slicing into two levels: at the network

level, radio resources are pre-allocated to each gNodeB over a large time scale, while at the gNodeB level, these resources are dynamically scheduled at mini-slot granularity to handle short-term request variations. The framework enables time-varying traffic loads of differentiated services across multiple gNodeBs and allows dynamic inter-gNodeB resource sharing to increase multiplexing gains. A case study in their article demonstrates the effectiveness of the approach and outlines open research challenges.

Shan Zhang et al. [9] proposed a service-oriented hierarchical soft slicing framework for cache-enabled vehicular networks to meet diverse, multi-dimensional QoS requirements. In their approach, each slice supports a specific service with logical resource isolation while opportunistically reusing resources to exploit multiplexing gains. The framework is analysed for two representative on-road content services — a time-critical context information service (CIS) and a bandwidth-intensive infotainment service (IS) — with intra- and inter-slice resource reuse. Analytical and simulation results show that the proposed method can increase IS throughput by about 30 % while maintaining the same level of CIS content freshness and service delay.

Shi et al. [17] proposed a two-level soft slicing scheme for 5G-and-beyond radio access networks to simultaneously support ultra-reliable low-latency communications (URLLC) and enhanced mobile broadband (eMBB) services. At the network level, they determine the required radio resources for eMBB and analyse the delay-violation probability for URLLC, formulating an integer non-linear program for network-level preallocation. Given the NP-completeness of the problem, they develop a low-complexity heuristic to obtain near-optimal solutions, and at the gNB level they design a scheduling scheme enabling real-time resource sharing among URLLC services. Simulation results show that their soft-slicing approach satisfies stringent QoS requirements for both service types while achieving significantly higher resource-utilization efficiency than conventional hard slicing schemes.

Jingfang Ding et al. [18] proposed a dynamic switching soft slicing strategy for industrial mixed traffic in 5G networks. Their approach targets two traffic types — periodic delay-sensitive (PDS) and sporadic delay-tolerant (SDT) — and combines a traffic-QoS-aware soft slicing (TQASS) scheme with a resource-efficiency-aware soft slicing (REASS) scheme. The strategy guarantees the reliability of PDS traffic under strict delay constraints while dynamically allocating remaining resources to SDT traffic. Simulation results demonstrate that this dynamic soft slicing approach outperforms existing methods in satisfying the stringent QoS requirements of industrial mixed traffic.

Kaytaz et al. [19] addressed the RAN slicing problem in 5G vehicle-to-everything (V2X) communications and proposed an h-DQN-based Soft Slicing (HSS) method for model-free opportunistic slice management. HSS implements a hierarchical reinforcement learning framework in which a high-level meta-controller selects subgoals and low-level controllers choose actions conditioned on these subgoals and the system state. By decomposing performance constraints and applying deep reinforcement learning at different temporal scales, their approach improves sample efficiency and, according to their results, outperforms traditional and RL-based V2X RAN slice-management schemes in terms of Age of Information (AoI), service delay, and network utility maximization.

Yan Shao et al. [20] formulated inter-slice resource management in dense cellular networks as a multi-agent reinforcement learning problem and enhanced cooperation between base-station agents using graph attention networks (GAT), achieving superior performance over existing methods in extensive simulations. Nassima Toumi et al. [21] addressed the multi-domain service function chaining (SFC) placement problem by leveraging deep reinforcement learning. They proposed a hierarchical architecture with local and multi-domain agents trained on different DRL models to perform SFC and sub-SFC placement while meeting SLA requirements.

Xianfu Chen et al. [22] tackled two-timescale radio resource management in O-RAN slicing by proposing a meta Markov decision process formulation. They decomposed the problem into a long-timescale master and a short-timescale subproblem, solved via a hierarchical reinforcement learning mechanism combining deep RL and meta-RL. Their results showed superior performance over state-of-the-art baselines. Hasan Anıl Akyıldız et al. [23] proposed a multi-agent hierarchical reinforcement learning approach for joint radio resource allocation between URLLC and eMBB slices under random traffic arrivals. A main agent performs inter-slice allocation, while URLLC and eMBB sub-agents handle intra-slice allocation. The state space incorporates queue occupancy and channel quality, and actions correspond to user-level resource assignments. Their results show sub-millisecond URLLC delays with minimal eMBB throughput degradation.

Hnin Pann Phyu et al. [24] addressed the energy–QoS trade-off in sliced mobile networks. They formulated a joint slice activation/deactivation and user association problem aimed at minimizing energy consumption while maintaining QoS. Their proposed multi-agent cooperative framework (ICE-CREAM) uses agents at two granularity levels that share information with neighbors and are trained on a global reward. Evaluation on a real-world dataset of French mobile service consumption showed more than 30% energy efficiency improvement over always-active slices while preserving QoS. Tianlun Hu et al. [25] proposed a multi-agent deep reinforcement learning approach for dynamic, slice-aware radio resource partitioning across multiple cells. Their framework tackles inter-cell interference and inter-

slice constraints by introducing two distributed coordination schemes that allow agents to cooperate. System-level simulations demonstrate that the proposed method achieves over two-fold improvement in resource efficiency and better delay performance compared with centralized baselines.

Guorong Zhou et al. [26] addressed the challenge of multi-objective optimization in RAN slicing, where throughput, delay, and coverage of three slice types (high-bandwidth, low-delay, wide-coverage) must be jointly optimized over a shared infrastructure. They proposed an improved multi-agent deep deterministic policy gradient (IMADDPG) algorithm with centralized training and distributed execution to solve the NP-hard problem. Using a rank voting mechanism to approach near-Pareto optimal solutions, their method outperformed scalar-utility and benchmark algorithms and achieved a flexible tradeoff among competing slice objectives.

Federico Mason et al. [14] investigated dynamic resource allocation in network slicing using a distributed Deep Reinforcement Learning framework. Their approach deploys multiple cooperating agents trained with the Advantage Actor-Critic (A2C) algorithm to handle continuous action spaces. Through extensive simulations, they showed that the proposed DRL-based strategy outperforms efficient empirical algorithms, achieving higher adaptability to varying scenarios without requiring additional training. Muhammad Sulaiman et al. [13] addressed joint network slicing and slice admission control in 5G Cloud-RANs by proposing a multi-agent Deep Reinforcement Learning (DRL) approach. Their framework learns the dynamics of slice-request traffic and jointly optimizes slice placement and admission decisions. Compared with heuristic baselines and single-agent DRL, the proposed multi-agent DRL achieves up to 18% and 3.8% higher long-term revenue, respectively, and exhibits improved robustness when generalized to unseen scenarios.

Yohan Kim et al. [12] proposed an RL-based dynamic resource allocation framework for end-to-end network slicing in hierarchical MEC environments. They modelled the problem as a Markov Decision Process (MDP) and used proximal policy optimization (PPO) to develop independently-collaborative and jointly-collaborative allocation algorithms. Their results show that the proposed methods can effectively recognize slice request characteristics and allocate resources efficiently while maintaining a high QoS satisfaction rate for heterogeneous 5G slices.

Hao Zhou et al. [27] developed a multi-agent reinforcement learning (MARL) approach for radio resource slicing in 5G networks. In their framework, each slice acts as an intelligent agent competing for limited radio resources, and a correlated Q-learning scheme (CO-QRA) is applied for inter-slice resource block allocation. They compare CO-QRA with Nash Q-learning (NQL), Latency-Reliability-Throughput Q-learning (LRTQ), and priority proportional fairness (PPF). Simulation results show that CO-QRA significantly reduces latency and packet drop rate while improving throughput compared with these baselines.

Despite the growing body of work on network slicing, the joint consideration of soft inter-slice resource sharing and hierarchical multi-agent decision making remains relatively limited. Existing soft-slicing approaches mainly rely on optimization, heuristic, or application-specific hierarchical mechanisms, while many MARL-based slicing methods focus on resource allocation without explicitly incorporating controlled resource sharing and slice dissatisfaction into the decision process. Moreover, ensuring that learning-based admission decisions remain feasible under resource, borrowing, and continuity constraints requires an explicit constraint-handling mechanism. To address these gaps, we propose an online hierarchical multi-agent framework in which slice-level agents perform admission decisions, a meta-controller coordinates inter-slice borrowing, and a deterministic safety layer validates and executes feasible resource allocations.

3- System Model and Problem Formulation

We consider a soft network slicing environment in which multiple logical slices share the available radio resources, with each slice assigned a predefined base resource share. User requests arrive dynamically with heterogeneous resource demands and service requirements. When its own available resources are insufficient, a slice may temporarily borrow unused resources from eligible neighboring slices, subject to predefined borrowing limits. While such controlled sharing can improve resource utilization, lending resources imposes a burden on donor slices and may increase slice dissatisfaction. Therefore, resource sharing must be adaptively controlled to balance resource utilization and slice dissatisfaction while maintaining feasible allocations.

3-1- System Entities

Slices: We denote the set of network slices by $\mathcal{M} = \{1, 2, \ldots, M\}$. Each slice m corresponds to a specific type of service or tenant with its own arrival processes, deadlines, and resource demands.

Jobs/Requests: Each incoming job for slice m is characterized by an arrival time a_i, a processing duration τ_i, a resource demand r_i, and a deadline d_i. In addition, a sensitivity indicator is included as part of the request information available to the corresponding slice-level agent when making admission decisions.

Resources: The system has a total resource capacity R, shared among all slices over discrete time slots. Each slice can primarily use its allocated base quota, while unused

resources may be temporarily borrowed by eligible neighboring slices. Inter-slice borrowing is permitted only between slices connected in the predefined sharing topology and is limited by the borrowing caps determined by the meta-controller.

Safety Layer: A deterministic safety layer validates admission decisions before resource allocation. For each accepted request, it first checks the available capacity of the request's own slice. If additional resources are required, borrowing is considered only from eligible neighboring slices and is constrained by the donor's available headroom and the remaining borrowing cap. The allocation is committed only when all resource and continuity constraints can be satisfied.

3-2- Decision Structure

At each time step t, the system must decide:

1. **dmission control:** For each slice, the corresponding lower-level agent determines which eligible user requests should be admitted based on the current request and system information.
2. **Meta-control**: At a slower decision timescale, the meta-controller determines the inter-slice borrowing caps that regulate how much resource sharing is permitted between neighboring slices.

These decisions form a two-level hierarchical control structure in which lower-level slice agents perform local admission decisions, while the meta-controller coordinates inter-slice resource sharing. The resulting decisions are subsequently validated and executed by the deterministic safety layer, which enforces the resource, borrowing, and continuity constraints.

3-3- Objective

The system-level objective is to efficiently utilize the available radio resources while controlling slice dissatisfaction arising from inter-slice resource sharing. At each time slot t, the instantaneous resource utilization U_t is defined as the ratio of the total amount of allocated resources to the overall available resource capacity:

$$U_t = \frac{R_t^{\text{used}}}{R} \qquad (1)$$

where R_t^{used} denotes the total amount of resources in use at time slot t, and R is the total available resource capacity.

To quantify the cost associated with inter-slice resource sharing, we define slice dissatisfaction according to the fraction of a slice's base resource quota that is temporarily lent to other eligible slices. Let $L_m(t)$ denote the amount of resources lent by slice m at time slot t, and let q_m denote its predefined base quota. The instantaneous system-wide dissatisfaction D_t is defined as:

$$D_t = \frac{1}{M}\sum_{m=1}^{M}\frac{L_m(t)}{q_m} \qquad (2)$$

A value of D_t close to zero indicates little or no inter-slice sharing, whereas a larger value represents a higher resource-sharing burden on donor slices.

Over an episode consisting of T time slots, the average resource utilization U is calculated as:

$$U = \frac{1}{T}\sum_{t=1}^{T}U_t \qquad (3)$$

while the average slice dissatisfaction D is given by:

$$D = \frac{1}{T}\sum_{t=1}^{T}D_t \qquad (4)$$

For performance evaluation, these two metrics are combined into the following system-level objective:

$$J_{\text{eval}} = \alpha U - \beta D \qquad (5)$$

where α and β are non-negative weighting coefficients representing the relative importance of resource utilization and slice dissatisfaction, respectively.

3-4- Problem Formulation

The resulting problem can be formulated as a hierarchical multi-agent decision-making task. At the lower level, each slice-specific agent evaluates eligible user requests using a fixed-dimensional observation that captures the request slack, resource demand, processing duration, sensitivity indicator, and the current system-level acceptance ratio, and then makes an admission decision for each request. At the higher level, a meta-controller operates at a slower timescale and observes compact system-level information for each slice, including its base resource share, pending-request ratio, accepted-request ratio, and remaining borrowing-cap ratio. Based on this information, the meta-controller adaptively determines the inter-slice borrowing caps that regulate resource sharing between eligible neighboring slices. Before execution, the proposed admission decisions are processed by a deterministic safety layer that checks resource availability, donor headroom, borrowing limits, and continuity requirements before committing an allocation.

This hierarchical formulation separates fine-grained admission decisions from system-level coordination of inter-slice sharing. By combining adaptive learning with deterministic feasibility enforcement, it provides a suitable foundation for achieving efficient resource utilization while controlling the cost of inter-slice borrowing. The resulting HMARL framework is described in the next section.

4- Proposed HMARL Framework

4-1- Hierarchical Architecture and Decision Process

The proposed Hierarchical Multi-Agent Reinforcement Learning (HMARL) framework is designed for online resource management in a soft network slicing environment. The framework combines decentralized admission decisions at the slice level with system-wide coordination of inter-slice resource sharing. It consists of three main components: a set of low-level slice actors, a higher-level meta-controller, and a deterministic safety layer.

At the lower level, each network slice m is associated with an independent actor π_{θ_m}. The slice-level actor evaluates the currently eligible requests belonging to its own slice and determines whether each request should be proposed for admission. These actors operate at every time slot and focus on fine-grained admission control based on request-level and system-level information. Their outputs are admission proposals that are subsequently processed by the resource-allocation mechanism.

At the higher level, a meta-controller π_{θ_H} operates at a slower time scale, once every K time slots. The meta-controller observes a compact summary of the state of all slices and determines the inter-slice borrowing caps that regulate resource sharing between eligible neighboring slices. The predefined base resource quota of each slice remains unchanged during operation; therefore, the role of the meta-controller is limited to adapting the allowable degree of inter-slice borrowing according to the current network conditions.

The decisions generated by the learning policies are not executed directly. Instead, the proposed admission decisions and the borrowing limits determined by the meta-controller are provided to a deterministic safety layer. The safety layer verifies resource availability and checks the relevant borrowing, topology, and continuity constraints before committing an allocation. If the requirements of a proposed admission cannot be satisfied, the request is not allocated even when the corresponding low-level actor has proposed its admission.

This hierarchical design separates local admission control from global coordination of inter-slice resource sharing. The low-level actors can therefore focus on individual request decisions, while the meta-controller regulates the overall sharing policy at a slower time scale. At the same time, deterministic feasibility checking prevents learning-based actions from directly producing resource allocations that violate the predefined system constraints.

4-2- State and Action Spaces

Low-Level Slice Agents

Each network slice *m* is associated with a low-level actor that makes admission decisions for the requests belonging to that slice. To accommodate a variable number of eligible requests without changing the neural-network dimensions, each request is evaluated independently using a fixed-dimensional observation vector.

For an eligible request i of slice m at time slot t, the local observation is defined as

$$o_{m,i,t} = [\hat{s}_{i,t}, \hat{r}_i, \hat{\tau}_i, z_i, \rho_t^{acc}] \qquad (6)$$

where $\hat{s}_{i,t}$ denotes the remaining slack of request i, $\hat{r}_i$ is its resource demand, $\hat{\tau}_i$ represents its service duration, $z_i \in 0,1$ is the sensitivity indicator, and $\rho_t^{acc} \in [0,1]$ denotes the current system-level acceptance ratio. Thus, the low-level actor receives a five-dimensional observation for each eligible request. The slice actor π_{θ_m} maps the observation vector to an admission probability:

$$p_{m,i,t} = \pi_{\theta_m}(o_{m,i,t}) \qquad (7)$$

During training, the corresponding binary action is sampled from a Bernoulli distribution:

$$a_{m,i,t} \sim \text{Bernoulli}(p_{m,i,t}) \qquad (8)$$

With

$$a_{m,i,t} \in 0,1. \qquad (9)$$

Here, $a_{\{m,i,t\}} = 1$ indicates that the low-level policy proposes request i for admission, whereas $a_{\{m,i,t\}} = 0$ indicates that the request is not proposed for admission at the current time slot. The admission proposal does not itself guarantee resource allocation, since every proposed request must subsequently pass the deterministic feasibility checks performed by the safety layer.

This per-request formulation avoids constructing variable-length state and action vectors. Regardless of the number of eligible users at a given time slot, the same five-dimensional actor input and one-dimensional Bernoulli output are applied independently to each request.

Meta-Controller

The meta-controller operates at a slower time scale and makes one decision every *K* time slots. At each meta-decision instant, it observes a compact system-level representation that summarizes the condition of all *M* slices. For each slice *m*, the high-level observation vector is defined as

$$x_m^H = [\hat{q}_m, \rho_m^{pend}, \rho_m^{acc}, \rho_m^{cap}] \qquad (10)$$

where $\hat{q}_m$ denotes the base resource share of slice *m*, $\rho_m^{pend} \in [0,1]$ is its pending-request ratio, $\rho_m^{acc} \in [0,1]$ is its accepted-request ratio, and $\rho_m^{cap} \in [0,1]$ represents its remaining borrowing-cap ratio. The complete high-level

state is therefore represented by an $M \times 4$ matrix $X_t^H \in \mathbb{R}^{M \times 4}$.

Based on X_t^H , the meta-controller generates continuous latent actions using a parameterized Normal policy. These actions are transformed into the inter-slice borrowing-cap matrix

$$C_t = [c_{m \leftarrow n}(t)] \in \mathbb{R}^{M \times M}, \qquad (11)$$

where $c_{m \leftarrow n}(t)$ represents the maximum amount of resources that requesting slice *m* is permitted to borrow from donor slice *n* during the corresponding meta-control interval.

The meta-controller therefore controls only the allowable degree of inter-slice borrowing. The resulting borrowing caps are subsequently used by the deterministic safety layer when validating and executing the low-level admission proposals.

4-3- Deterministic Safety Layer

The admission decisions generated by the low-level actors are treated as admission proposals and are not directly translated into resource allocations. Before execution, all proposed requests are processed by a deterministic safety layer that verifies their feasibility with respect to the available resources, slice quotas, borrowing limits, sharing topology, and service requirements. Therefore, the learning policies determine which requests should be considered for admission, while the safety layer determines whether these proposals can actually be executed.

At each time slot *t*, the safety layer first collects all requests proposed for admission by the low-level actors. Since several requests may be proposed simultaneously, they are processed according to a deterministic priority rule. For each request *i*, the remaining slack is computed as

$$s_{i,t} = d_i - (t + \tau_i - 1), \qquad (12)$$

where d_i denotes the request deadline and τ_i is its required service duration. Requests with smaller slack are processed first because they have less scheduling flexibility. If two or more requests have the same slack, requests with larger resource demands are given higher priority, followed by those with shorter service durations. This ordering ensures that simultaneous admission proposals are handled consistently rather than being allocated in an arbitrary order.

For each request in the resulting priority order, the safety layer first checks whether the required resources can be supported over the complete service interval from time slot t to $t + \tau_i - 1$. It then attempts to satisfy the resource demand using the available capacity associated with the request's own slice. If sufficient local capacity is available throughout the required service interval, the request can be admitted without inter-slice borrowing.

When the available capacity of the requesting slice is insufficient, the remaining resource deficit is considered for inter-slice borrowing. Only slices connected to the requesting slice through the predefined sharing topology are considered as potential donors. For each eligible donor, the safety layer determines the amount of resource that can be provided according to two conditions: the donor must have sufficient available headroom, and the corresponding remaining borrowing cap must permit the requested transfer. Hence, the amount borrowed from a donor cannot exceed either its available resource headroom or the remaining cap associated with that slice pair $c_{m \leftarrow n}(t)$.

If multiple eligible donors are available, the safety layer examines the admissible borrowing opportunities until the remaining demand is satisfied or no additional feasible resource can be obtained. The request is admitted only when its complete resource demand can be supported over the entire required service interval. Partial allocations are not committed. If the complete demand cannot be satisfied using the combination of local and admissibly borrowed resources, the proposed admission is rejected and no tentative resource assignment is retained.

After a request is successfully admitted, the corresponding resource occupancy and remaining borrowing capacities are immediately updated. The next request in the priority order is therefore evaluated according to the updated system state. This sequential update is particularly important when several requests are proposed in the same time slot, since resources committed to an earlier request are no longer considered available to subsequent requests.

In addition, the safety layer prevents conflicting assignments by ensuring that the same physical resource is not simultaneously allocated to multiple requests in a given time slot. Resource availability is checked across the entire service duration of each admitted request, thereby preserving the required allocation for the requested processing interval.

Overall, the deterministic safety layer acts as a feasibility-enforcement mechanism between the learned policies and the actual resource allocator. It does not introduce an additional learned policy and does not modify the objectives of the low-level actors or the meta-controller. Instead, it ensures that only admission proposals satisfying the explicitly defined resource, borrowing, topology, and service-continuity constraints are executed. Thus, feasibility of the executed allocations is enforced deterministically, without assuming that the reinforcement-learning policies themselves learn or guarantee constraint satisfaction.

4-4- Reward Design and Advantage Estimation

The reward function used to train the HMARL policies is designed to encourage efficient resource utilization while discouraging excessive dependence on inter-slice

borrowing. At each time slot *t*, the training reward is defined as

$$r_t = \alpha_r \tilde{U}_t - \beta_r \tilde{D}_t + \eta_r B_t^{direct} - \xi_r P_t^{borrow}, \quad (13)$$

where $\tilde{U}_t$ denotes the normalized resource-utilization component and $\tilde{D}_t$ denotes the normalized slice-dissatisfaction component used during training. The term B_t^{direct} provides an additional positive signal when requests are served without requiring inter-slice borrowing, while P_t^{borrow} penalizes excessive use of borrowed resources. The coefficients α_r , β_r , η_r , and ξ_r control the relative contributions of these components.

4-5- PPO-Based Policy Optimization

The proposed framework follows centralized training with hierarchical execution and local admission decisions at the lower level. Two independent state-value critics are used during training. A shared centralized critic is used for the low-level slice actors, while a separate critic is associated with the high-level meta-controller. The two critics have independent parameters, although both operate on a compact global representation of the system state.

Let $V_{\phi_L}(g_t)$ denote the value function of the shared low-level critic, and let $V_{\phi_H}(g_t)$ denote the value function of the high-level critic, where $g_t \in \mathbb{R}^6$ represents the compact global state supplied to the critics. The critics are used only during training to estimate expected future returns and construct the advantage signals required for policy optimization.

For the low-level policies, the temporal-difference residual at time slot *t* is

$$\delta_t^L = r_t + \gamma V_{\phi_L}(g_{t+1}) - V_{\phi_L}(g_t), \quad (14)$$

where $\gamma \in [0,1]$ is the discount factor.

The corresponding low-level advantage is estimated using Generalized Advantage Estimation (GAE):

$$\hat{A}_t^L = \sum_{l=0}^{T-t-1} (\gamma\lambda)^l \delta_{t+l}^L, \quad (15)$$

where $\lambda \in [0,1]$ is the GAE parameter. The parameter λ controls the trade-off between the bias and variance of the advantage estimate, while γ determines the relative importance of future rewards.

Because the meta-controller operates at a slower time scale, a high-level learning sample corresponds to the interval between two consecutive meta-decisions rather than to a single time slot. Let t_j denote the time of the j-th meta-decision and let Δ_j denote the number of time slots until the next meta-decision. The reward associated with this high-level transition is obtained by aggregating the low-level rewards over the corresponding interval:

$$R_j^H = \sum_{l=0}^{\Delta_j - 1} \gamma^l r_{t_j + l}. \quad (16)$$

the high-level temporal-difference residual can then be expressed as

$$\delta_j^H = R_j^H + \gamma^{\Delta_j} V_{\phi_H}(g_{t_{j+1}}) - V_{\phi_H}(g_{t_j}). \quad (17)$$

The high-level advantage $\hat{A}_j^H$ is subsequently obtained using the same GAE principle over the sequence of meta-level transitions. In this way, the low-level actors receive advantage signals corresponding to fine-grained admission decisions, whereas the meta-controller receives feedback that reflects the cumulative effect of its borrowing-cap decisions over a longer control interval.

To improve numerical stability during policy optimization, the learning signals used for advantage estimation are normalized during training. This normalization is applied only to the learning procedure and does not change the definitions of the performance metrics reported in the evaluation results.

The policies at both levels of the proposed framework are trained using Proximal Policy Optimization (PPO). For a policy with parameters θ, the probability ratio between the current policy and the policy used to collect the training samples is defined as

$$\rho_t(\theta) = \frac{\pi_\theta(a_t \mid o_t)}{\pi_{\theta^{old}}(a_t \mid o_t)}. \quad (18)$$

PPO updates the policy using the clipped surrogate objective

$$L^{clip}(\theta) = \mathbb{E}_t\left[\min\left(\rho_t(\theta)\hat{A}_t, \text{clip}\left(\rho_t(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_t\right)\right], \quad (19)$$

where $\hat{A}_t$ is the advantage estimate obtained using GAE and ϵ is the PPO clipping parameter. The clipping operation limits excessive changes between successive policy updates and improves the stability of on-policy training.

The same PPO principle is applied separately at the two hierarchical levels. For the low-level actors, the policy corresponds to the admission decisions generated for individual requests, and the updates are based on the low-level advantages computed at the time-slot scale. For the meta-controller, the policy corresponds to the continuous actions used to determine the borrowing caps, and its updates are performed using the high-level advantages computed at the slower meta-decision scale.

5- Experimental Setup and Evaluation Metrics

5-1- Simulation Environment and Traffic Scenarios

The proposed framework is evaluated using a Python/PyTorch-based simulator developed for online soft network slicing. Multiple logical network slices share the available resources, while controlled inter-slice borrowing is permitted only between eligible neighboring slices according to a predefined sharing topology. Each slice is assigned a fixed base resource share, and temporary resource borrowing is regulated by the corresponding inter-slice borrowing caps.

At the beginning of each episode, a new set of user requests is generated for each slice. Each request is characterized by an arrival time, resource demand, service duration, deadline, and a binary sensitivity indicator. These request characteristics vary across users and episodes to represent heterogeneous and time-varying traffic conditions.

To evaluate the methods under different levels of scheduling difficulty, five traffic scenario classes are considered. The scenario classes differ in the temporal distribution of request arrivals and the tightness of their scheduling windows. They range from relatively dispersed traffic with greater scheduling flexibility to more concentrated traffic with tighter timing requirements. Accordingly, the considered scenarios expose the evaluated methods to different levels of temporal contention and resource-allocation difficulty.

For each evaluation instance, a fresh traffic realization is generated according to the corresponding scenario class. The performance of the considered methods is subsequently evaluated across all scenario classes to provide a representative assessment under heterogeneous traffic conditions.

5-2- Training and Simulation Configuration

The main simulation and learning parameters are summarized in Table I. The same environmental settings and traffic-generation rules are used when evaluating all methods to provide a consistent comparison. For learning-based methods, the policy networks are independently initialized and trained using fresh training instances. Unless otherwise stated, the parameter values listed in Table I are kept fixed throughout the experiments.

Table 1: Simulation and Training Parameters

Parameter	Value
Number of slices (M)	5
Total resource capacity (R)	40
Episode horizon (T)	30 slots
Base resource proportions	[0.2, 0.2, 0.4, 0.1, 0.1]
Requests per slice	7
Meta-controller interval (K)	5 slots
Training episodes	2000
Learning rate	3×10^{-4}
Discount factor (γ)	0.99
GAE parameter (λ)	0.95
PPO clipping parameter (ε)	0.30
PPO optimization epochs	4
Evaluation-objective weights	$\alpha = 0.6$, $\beta = 0.4$

5-3- Baseline and Comparison Methods

To evaluate the proposed HMARL framework, we compare it with several learning-based, heuristic, and non-sharing approaches. All methods are evaluated under the same traffic scenarios and environmental settings to provide a consistent comparison.

h-DQN based Soft Slicing (HSS) is included as a hierarchical reinforcement-learning baseline [19]. In this approach, a high-level controller selects a slice as the current subgoal, while a lower-level controller performs the corresponding admission or allocation decisions. Unlike the proposed HMARL framework, HSS follows a sequential subgoal-based hierarchy rather than jointly coordinating all slices through adaptive inter-slice borrowing caps.

The w/o Meta variant removes the meta-controller from HMARL while retaining the low-level slice actors. Consequently, admission decisions are still learned, but inter-slice sharing is not adaptively coordinated at the higher level. This variant is used to assess the contribution of hierarchical meta-level coordination.

No-Sharing represents a hard-isolation configuration in which inter-slice borrowing is disabled by setting the borrowing caps to zero. Each slice can therefore use only its predefined resource share. This baseline illustrates the performance obtained when resource isolation is preserved at the expense of flexible inter-slice sharing.

Random Admission provides a simple non-learning reference. Eligible requests are proposed for admission according to a fixed Bernoulli probability, while the feasibility of the resulting proposals is handled by the same resource allocation constraints used in the simulation environment.

The Greedy method provides a lightweight heuristic reference. Requests are prioritized according to their scheduling urgency and resource requirements, and feasible requests are admitted sequentially whenever sufficient resources are available. Unlike the learning-based approaches, Greedy does not learn a policy from interaction with the environment.

Heuristic Resource Allocation for Soft Slicing (HRASS) is included as a heuristic resource-allocation reference that explicitly searches for feasible assignments while considering request timing, resource requirements, and admissible inter-slice resource sharing [15]. It uses broader request information to construct complete allocations and therefore provides a useful optimization-oriented reference for assessing the performance of the online learning approaches.

Together, these comparison methods cover different levels of decision-making complexity, ranging from random and isolated admission to heuristic and hierarchical learning-based resource management.

5-4- Evaluation Metrics

The performance of the considered methods is evaluated using four main metrics: the overall reward value, resource utilization, slice dissatisfaction, and request rejection ratio. Resource utilization and slice dissatisfaction are computed according to the definitions introduced in Section 3. Their normalized values are reported in the comparative results to facilitate consistent comparison across different traffic scenarios.

The request rejection ratio measures the fraction of generated requests that are not successfully admitted and allocated during an episode. A lower rejection ratio therefore indicates that a larger proportion of the offered traffic can be successfully served. These metrics are considered jointly, since a method may achieve low dissatisfaction by limiting inter-slice borrowing while simultaneously experiencing lower resource utilization or a higher request rejection ratio.

6- Performance Evaluation

This section presents and discusses the performance results of the proposed HMARL framework. The evaluation is conducted in two stages. First, the training behavior of HMARL is examined in terms of the evaluation objective, normalized resource utilization, normalized slice dissatisfaction, and request rejection ratio. Then, the performance of HMARL is compared with the considered learning-based, heuristic, and non-sharing methods across different traffic scenario classes using the same evaluation metrics.

6-1- Training Convergence

Fig. 1 illustrates the evolution of the main performance metrics over 2000 training episodes. The figure reports the training reward, normalized resource utilization, normalized slice dissatisfaction, and request rejection ratio. The trajectories show that HMARL undergoes a relatively short initial adaptation phase and subsequently reaches a stable operating regime, with only moderate fluctuations over the remaining training episodes.

As shown in Fig. 1(a), the reward value of HMARL increases rapidly during the early stage of training and stabilizes at approximately 0.23 after the initial adaptation period. It remains consistently above the other considered methods throughout most of the training process. This behavior indicates that the hierarchical policy progressively learns decisions that provide a favorable balance between resource utilization and the cost associated with inter-slice sharing. The relatively small fluctuations observed after stabilization suggest consistent performance under dynamically generated traffic instances.

The utilization trend in Fig. 1(b) provides further insight into this behavior. HMARL rapidly increases its normalized resource utilization to approximately 0.70 and maintains this level for most of the training process. The remaining methods stabilize at noticeably lower utilization levels. This result suggests that hierarchical coordination and adaptive borrowing allow HMARL to make more effective use of the available radio resources.

The higher utilization achieved by HMARL is accompanied by higher normalized slice dissatisfaction, as illustrated in Fig. 1(c). After the initial adaptation phase, the dissatisfaction of HMARL stabilizes at approximately 0.46–0.49. This behavior is associated with more active use of controlled inter-slice borrowing, which imposes a greater sharing burden on donor slices. In contrast, No-Sharing maintains a dissatisfaction value close to zero because inter-slice borrowing is disabled by construction.

Fig. 1(d) shows a complementary trend in the request rejection ratio. HMARL exhibits a sharp reduction in rejection during the early training stage, after which the ratio stabilizes at approximately 0.19. This is lower than the corresponding levels observed for the other methods in the figure. The simultaneous increase in utilization and reduction in rejection indicate that HMARL learns to serve a larger fraction of the offered requests while making greater use of controlled inter-slice sharing.

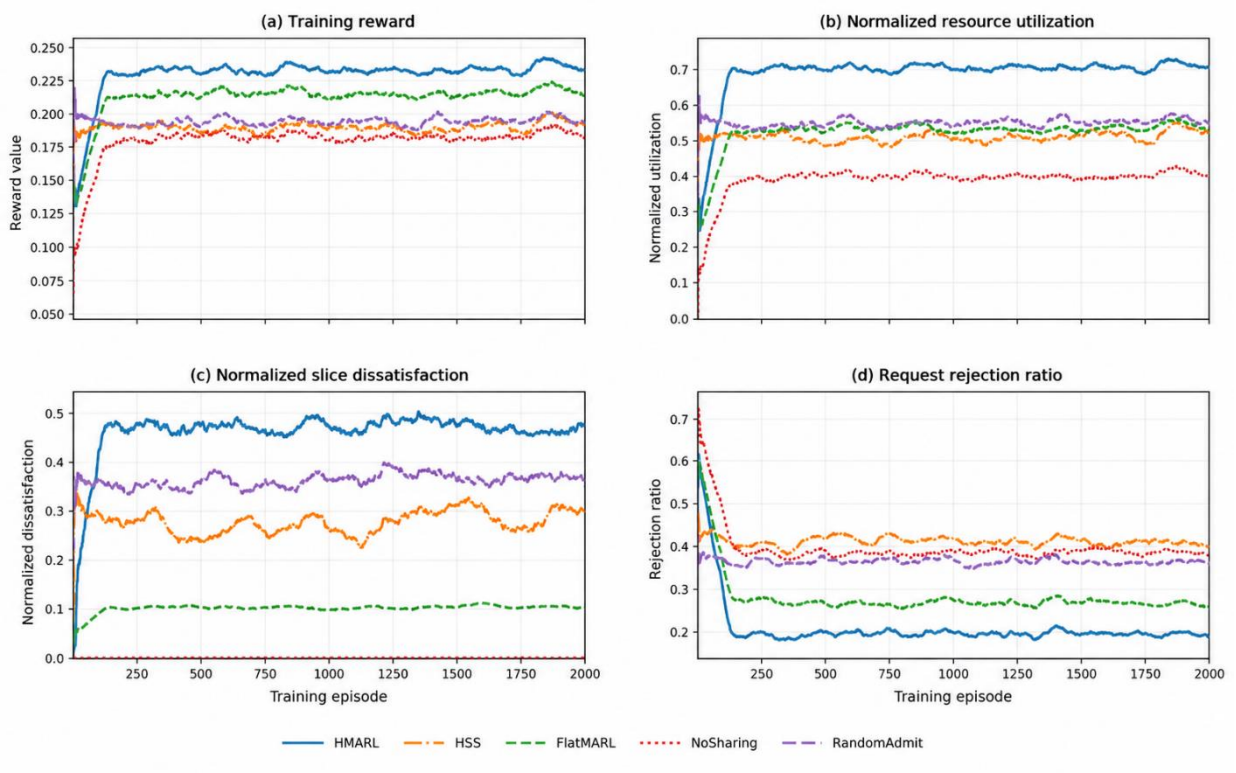


Fig. 1 Training convergence of the evaluated online methods over 2000 training episodes.

Taken together, the four metrics indicate that HMARL reaches a stable operating regime after the initial training phase. The observed performance does not result from minimizing slice dissatisfaction alone; rather, HMARL accepts a higher sharing burden while achieving substantially higher resource utilization and a lower request rejection ratio. This trade-off is reflected in the higher and stable training reward observed during the later stages of training.

6-2- Comparative Performance Across Traffic Scenarios

Fig. 2 compares HMARL with the considered learning-based, heuristic, random, non-sharing, and offline reference methods. The reported values are aggregated across the evaluated traffic scenarios and include the objective value, normalized resource utilization, normalized slice dissatisfaction, and request rejection ratio.

As shown in Fig. 2(a), HRASS achieves the highest objective value at approximately 0.25, followed closely by HMARL at about 0.24 and the w/o Meta variant at about 0.23. The small difference between HMARL and HRASS is notable because HRASS operates offline with broader knowledge of the requests over the scheduling horizon, whereas HMARL makes its decisions online using only currently available information.

Fig. 2(b) shows a similar trend in resource utilization. HRASS achieves the highest normalized utilization, approximately 0.76, followed by HMARL at about 0.73 and Greedy at about 0.68. The lower utilization of the w/o Meta variant highlights the contribution of adaptive meta-level borrowing coordination. As shown in Fig. 2(c), the higher utilization of HMARL is accompanied by a higher normalized slice dissatisfaction of approximately 0.51, reflecting its more active use of inter-slice borrowing. In contrast, No-Sharing maintains near-zero dissatisfaction because borrowing is disabled, but this is accompanied by substantially lower utilization and a higher rejection ratio.

Figure 2(d) further shows that HRASS achieves the lowest rejection ratio, approximately 0.32, followed by Greedy at about 0.35 and HMARL at about 0.37. The w/o Meta variant reaches approximately 0.44, while the remaining methods exhibit higher rejection levels.

Overall, HMARL provides a favorable trade-off among resource utilization, slice dissatisfaction, and request rejection, while achieving an objective value close to the offline HRASS reference. These results also show that adaptive meta-level coordination improves resource utilization and request admission at the cost of a controlled increase in inter-slice sharing.

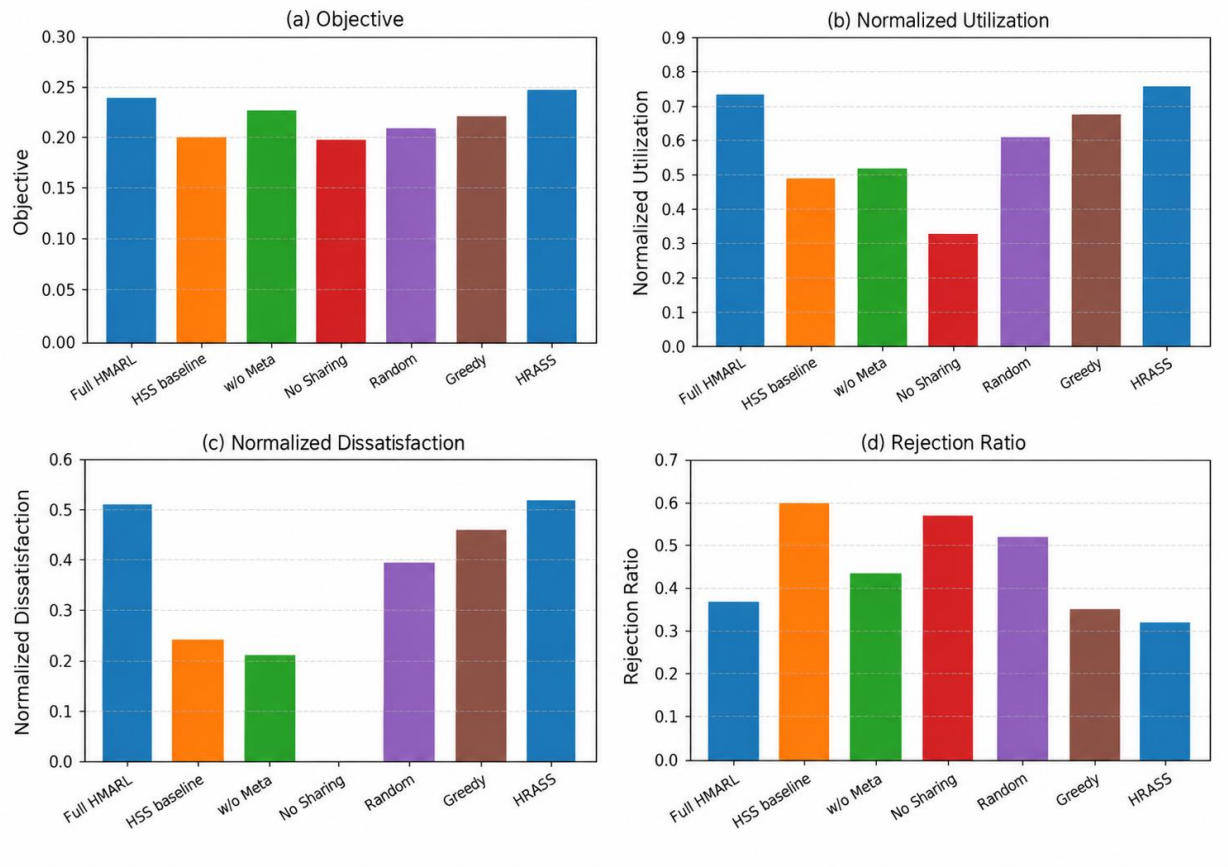


Fig. 2 Comparison of the evaluated methods across the traffic scenarios.

7- Conclusion

In this paper, we proposed a Hierarchical Multi-Agent Reinforcement Learning (HMARL) framework for online soft and safe network slicing. The framework combines slice-level admission decisions with a higher-level meta-controller that adaptively regulates inter-slice borrowing, while a deterministic safety layer ensures that only feasible resource allocations are executed. The experimental results show that HMARL converges to a stable policy and achieves a favorable balance among resource utilization, slice dissatisfaction, and request rejection. Compared with the considered learning-based, heuristic, random, and non-sharing alternatives, HMARL provides high resource utilization and competitive objective values while maintaining a relatively low rejection ratio. The comparison with the w/o Meta variant further demonstrates the benefit of meta-level borrowing coordination.

Future work will investigate more heterogeneous and time-varying traffic conditions, richer slice-specific service requirements, and adaptive multi-objective resource-management strategies. Further extensions may also consider larger and more dynamic network environments and mechanisms for improving the interpretability of hierarchical decisions in practical network slicing deployments.

References

[1] Chowdhury, M.Z., et al., 6G wireless communication systems: Applications, requirements, technologies, challenges, and research directions. IEEE Open Journal of the Communications Society, 2020. 1: p. 957-975.

[2] Afolabi, I., et al., Network slicing and softwarization: A survey on principles, enabling technologies, and solutions. IEEE Communications Surveys & Tutorials, 2018. 20(3): p. 2429-2453.

[3] Vassilaras, S., et al., The algorithmic aspects of network slicing. IEEE Communications Magazine, 2017. 55(8): p. 112-119.

[4] Khan, L.U., et al., Network slicing: Recent advances, taxonomy, requirements, and open research challenges. IEEE Access, 2020. 8: p. 36009-36028.

[5] Jiang, W., et al., The road towards 6G: A comprehensive survey. IEEE Open Journal of the Communications Society, 2021. 2: p. 334-366.

[6] Sasan, Z., et al. Joint network slicing, routing, and in-network computing for energy-efficient 6g. in 2024 IEEE Wireless Communications and Networking Conference (WCNC). 2024. IEEE.

[7] Gonzalez, A.J., et al. The isolation concept in the 5G network slicing. in 2020 European Conference on Networks and Communications (EuCNC). 2020. IEEE.
[8] Sasan, Z. and S. Khorsandi. Slice-aware resource calendaring in cloud-based radio access networks. in 2022 30th International Conference on Electrical Engineering (ICEE). 2022. IEEE.
[9] Zhang, S., et al., Hierarchical soft slicing to meet multi-dimensional QoS demand in cache-enabled vehicular networks. IEEE Transactions on Wireless Communications, 2020. 19(3): p. 2150-2162.
[10] Wang, T., et al., LinkSlice: Fine-grained network slice enforcement based on deep reinforcement learning. IEEE Journal on Selected Areas in Communications, 2022. 40(8): p. 2378-2394.
[11] Zhang, H., et al. A hard and soft hybrid slicing framework for service level agreement guarantee via deep reinforcement learning. in 2022 IEEE 95th Vehicular Technology Conference:(VTC2022-Spring). 2022. IEEE.
[12] Kim, Y. and H. Lim, Multi-agent reinforcement learning-based resource management for end-to-end network slicing. IEEe Access, 2021. 9: p. 56178-56190.
[13] Sulaiman, M., et al. Multi-agent deep reinforcement learning for slicing and admission control in 5G C-RAN. in NOMS 2022-2022 IEEE/IFIP network operations and management symposium. 2022. IEEE.
[14] Mason, F., G. Nencioni, and A. Zanella. A multi-agent reinforcement learning architecture for network slicing orchestration. in 2021 19th Mediterranean Communication and Computer Networking Conference (MedComNet). 2021. IEEE.
[15] Sasan, Z. and S. Khorsandi, Balancing resource utilization and slice dissatisfaction through dynamic soft slicing for 6G wireless networks. Scientific Reports, 2025. 15(1): p. 22987.
[16] Li, J., et al., A hierarchical soft RAN slicing framework for differentiated service provisioning. IEEE Wireless Communications, 2020. 27(6): p. 90-97.
[17] Shi, W., et al., Two-level soft RAN slicing for customized services in 5G-and-beyond wireless communications. IEEE Transactions on Industrial Informatics, 2021. 18(6): p. 4169-4179.
[18] Ding, J., M. Zheng, and H. Yu, Soft Resource Slicing for Industrial Mixed Traffic in 5G Networks. IEEE/CAA Journal of Automatica Sinica, 2025. 12(2): p. 463-465.
[19] Kaytaz, U., F. Sivrikaya, and S. Albayrak. Hierarchical deep reinforcement learning based dynamic RAN slicing for 5G V2X. in 2021 IEEE Global Communications Conference (GLOBECOM). 2021. IEEE.
[20] Shao, Y., et al., Graph attention network-based multi-agent reinforcement learning for slicing resource management in dense cellular network. IEEE Transactions on vehicular Technology, 2021. 70(10): p. 10792-10803.
[21] Toumi, N., M. Bagaa, and A. Ksentini. Hierarchical multi-agent deep reinforcement learning for SFC placement on multiple domains. in 2021 IEEE 46th Conference on Local Computer Networks (LCN). 2021. IEEE.
[22] Chen, X., et al. Hierarchical Meta-Reinforcement Learning for Resource-Efficient Slicing in O-RAN. in GLOBECOM 2023-2023 IEEE Global Communications Conference. 2023. IEEE.
[23] Akyıldız, H.A., et al., Hierarchical reinforcement learning based resource allocation for RAN slicing. IEEE Access, 2024. 12: p. 75818-75831.
[24] Phyu, H.P., D. Naboulsi, and R. Stanica, ICE-CREAM: multI-agent fully CooperativE deCentRalizEd frAMework for Energy Efficiency in RAN Slicing. IEEE Transactions on Network and Service Management, 2025.
[25] Hu, T., et al. Inter-cell slicing resource partitioning via coordinated multi-agent deep reinforcement learning. in ICC 2022-IEEE International Conference on Communications. 2022. IEEE.
[26] Zhou, G., et al., Joint multi-objective optimization for radio access network slicing using multi-agent deep reinforcement learning. IEEE Transactions on Vehicular Technology, 2023. 72(9): p. 11828-11843.
[27] Zhou, H., M. Elsayed, and M. Erol-Kantarci. RAN resource slicing in 5G using multi-agent correlated Q-learning. in 2021 IEEE 32nd Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC). 2021. IEEE.