HELSA: Hierarchical Reinforcement Learning with Spatiotemporal Abstraction for Large-Scale Multi-Agent Path Finding

Zhaoyi Song[1], Rongqing Zhang[1], and Xiang Cheng[2]

***Abstract*— The Multi-Agent Path Finding (MAPF) problem is a critical challenge in dynamic multi-robot systems. Recent studies have revealed that multi-agent reinforcement learning (MARL) is a promising approach to solving MAPF problems in a fully decentralized manner. However, as the size of the multi-robot system increases, sample inefficiency becomes a major impediment to learning-based methods. This paper presents a hierarchical reinforcement learning (HRL) framework for large-scale multi-agent path finding, featuring applying spatial and temporal abstraction to capture intermediate reward and thus encourage efficient exploration. Specifically, we introduce a meta controller that partitions the map into interconnected regions and optimizes agents' region-wise paths towards globally better solutions. Additionally, we design a lower-level controller that efficiently solves each sub-problem by incorporating heuristic guidance and an inter-agent communication mechanism with RL-based policies. Our empirical results on test instances of various scales demonstrate that our method outperforms existing approaches in terms of both success rate and makespan.**

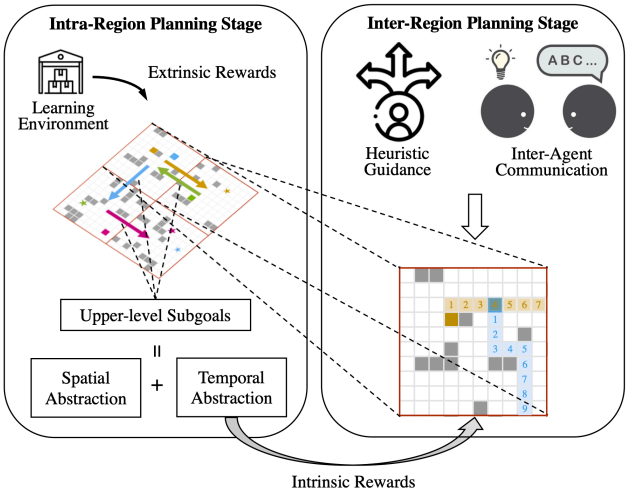


Fig. 1: Illustration of the HELSA framework

I. Introduction

In multi-robot systems, the ability to navigate effectively and efficiently is crucial. Such systems may entail thousands of automated mobile robots (AMRs) that have to find collision-free paths in a collaborative manner [1]. The Problem of multi-agent path finding is a challenging NP-hard problem [2] with numerous variants [3], and it has been of interest to researchers in the field of multi-agent systems ever since its inception.

Various solution-oriented approaches have been proposed for MAPF problems, which can be broadly classified as coupled and decoupled. Coupled approaches treat all robots as a single composite system, ensuring optimal and complete solutions at the cost of high computational complexity, making them less scalable as the number of agents increases. Decoupled approaches, in contrast, perform single-agent searches for each agent, reducing computational costs with the sacrifice of completeness and optimality. However, these solution-oriented approaches focus on computing complete paths for all agents based on global information, which inevitably requires either centralized coordination or sophisticated distributed protocols.

* This work is supported in part by the National Key R&D Program of China (Grant No. 2021ZD0112700), in part by the National Natural Science Foundation of China under Grant 62271351, 62125101, and 62341101, and the Fundamental Research Funds for the Central Universities. (Corresponding author: Rongqing Zhang)

[1]Zhaoyi Song and Rongqing Zhang are with the School of Software Engineering, Tongji University, Shanghai 200092, China. {2131509,rongqingz}@tongji.edu.cn

[2]Xiang Cheng is with the School of Electronics, Peking University, Beijing 100871, China. xiangcheng@pku.edu.cn

Real-time multi-robot applications require MAPF planners that can handle the challenges of coordinating massive AMRs and providing high response speed, scalability, and flexibility. In such environments, unexpected events, such as the sudden appearance of obstacles, can cause the planned paths to become invalid, and new paths must be quickly obtained to avoid collisions. Learning-based approaches with decentralized execution have become more and more popular in recent years since they are naturally suited to handling these challenges, allowing for fast and flexible adaptation to changing environments.

PRIMAL [4] proposed a novel approach that combines reinforcement learning and imitation learning, which share a common neural network to learn from demonstrations by an expert MAPF planner and encourages free exploration. Meanwhile, MAPPER [5] integrates a single-agent shortest path as heuristic guidance into each agent's observation space. However, these approaches incorporate either a MAPF planner or a single-agent path planner, which may contain misleading information and be more time-consuming, impairing their coordination learning and efficiency in the centralized learning phase.

Recent research has focused on learning inter-agent communication protocols to promote collaborative decision-making [6], [7]. These approaches eliminate the negative impact of independent decision-making under local information and strengthen the connections among agents. They provide an effective means of learning the coordination strategies that enables agents to collaborate effectively, without requiring a

priori knowledge of the environment. By learning how to communicate and coordinate with one another, agents can learn to work together and find optimal paths quickly and efficiently.

However, the use of learning-based MAPF planners for large-scale scenarios poses a significant challenge due to sample inefficiency. The data collection process of reinforcement learning is based on the interaction between the agents and the environment, usually following a epsilon-greedy exploration policy. As a result, the data collected may contain duplicate and uninformative samples, providing little feedback for policy and value learning, leading to slow and inefficient training process. This challenge is compounded by the fact that, in most learning-based MAPF methods, each agent has to make lots of attempts to find a feasible route to its goal before receiving any positive reward upon reaching it. This delayed and sparse reward further exacerbates the sample inefficiency, particularly as the scale of the training scenarios increases. Furthermore, the partial observability limits agents' perception range, giving rise to the convergence of RL models to oscillating or suboptimal policies, further impairing the training efficiency.

We propose a novel and effective hierarchical multi-agent decision-making scheme, termed as HELSA (i.e., Hierarchical rEinforcement Learning with Spatiotemporal Abstraction), to address challenges stemming from the delayed and sparse reward in large-scale MAPF and the limited perception due to partial observability. Inspired by recent advancements in two-stage object detectors [8], the proposed framework decomposes an RL task into a hierarchy of subproblems, as illustrated in Fig. 1. The upper-level policy learns how to partition the task into several shorter-horizon subgoals over a long period, which is often referred to as *temporal abstraction* [9]. Specifically, vertices on the map are partitioned into rectangular and interconnected regions, with an upper-level controller producing optimal subgoals as the macro-actions for longer timescales. The subgoals are the next region that each agent will navigate towards, which is the temporally and spatially extended courses of actions, regarded as *spatial abstraction*. The spatiotemporal abstraction gives rise to more structured exploration and enlarges agents' perception range at a relatively small cost. It also enables efficient credit assignment over a long course of training. The lower-level controller then refines the coarse macro paths, planning a collision-free path to accomplish the subgoals within a shorter horizon. Each subgoal is formulated as a multi-goal, multi-agent pathfinding problem, with the temporal goals of agents clustered along the regions' boundaries instead of randomly scattered.

We then develop an effective hierarchical multi-agent reinforcement learning algorithm, comprising two MARL models, to tackle the challenging large-scale MAPF problem. The hierarchy of controllers collectively determines agents' behaviors in two stages, solving MAPF instances at different granularities and capturing intermediate rewards to enrich agents' feedback from exploration, making agents' policies more far-sighted. We conduct extensive experiments comparing our proposed approach with other SOTA learning-based approaches, namely PRIMAL [4], DHC [6], and DCC [7]. The results indicate that our approach plans shorter paths with higher success rates and fewer collisions in large-scale multi-agent scenarios. The solutions of our approach scale linearly with the problem scope due to the successful problem decomposition and interplay of the hierachical policies.

The main contributions of this work are three-fold:

- We propose HELSA, a novel and effective multi-agent HRL scheme, which leverages a hierachy of heuristics, making RL approaches more applicable to large scale routing and planning scenarios.
- HELSA decomposes the long-horizon scheduling problem into several easier-to-learn subproblems. To accomplish this, we design an upper-level intra-region planning algorithm that applies a fully cooperative MARL approach to optimize the routes of multiple robots towards globally better solutions based on partially observable regional information.
- As for the lower-level inter-region planning stage, we formulate a constrained multi-goal multi-agent pathfinding problem and introduce a two-stage attentional communication mechanism and heuristic guidance to simplify the constraints and facilitate collaborative decision-making.

II. Problem Formulation

In this section, we present the formulation of the MAPF problem, along with the two-stage decision-making scheme.

Problem. The problem we consider in this paper is the classic MAPF problem, as described in [10]. We are given a connected undirected graph $G = (V, E)$, where vertices V denote locations that agents can occupy, and edges E connect adjacent vertices along which the agents can move. Additionally, we are given a set of M agents, $\{i | i \in \{1, \dots, M\}\}$, each with a unique start vertex, $s_i \in V$, and a unique target vertex, $t_i \in V$. At each discrete time step, each agent, i, executes an action, $\tilde{\mathbf{u}}_t^i$, which involves moving to an adjacent vertex or staying at its current vertex, potentially giving rise to collisions with other agents or obstacles. The path for the i-th agent is a sequence of adjacent or identical vertices that starts at s_i and terminates at t_i. We formulate the MAPF problem as a sequential decision-making problem whose ultimate goal is to find collision-free paths for all agents, while minimizing the sum of costs (i.e., the sum of arrival times of all agents at their goal vertices), given the following assumptions.

Assumptions. 1) Each agent has limited access to the information in the grid world, confined to a square field of view (FoV) of size $l \times l$, where l is an odd number, ensuring that the agent is at the center of its FoV. 2) In contrast to MAPPER [5], where agents are removed from the map upon reaching their goals, this paper assumes that agents remain on the map until all agents have reached their goals. However, agents may not stay at their goal vertices in case they hinder other agents from reaching their goal vertices. 3) Agents can only communicate with their neighboring agents

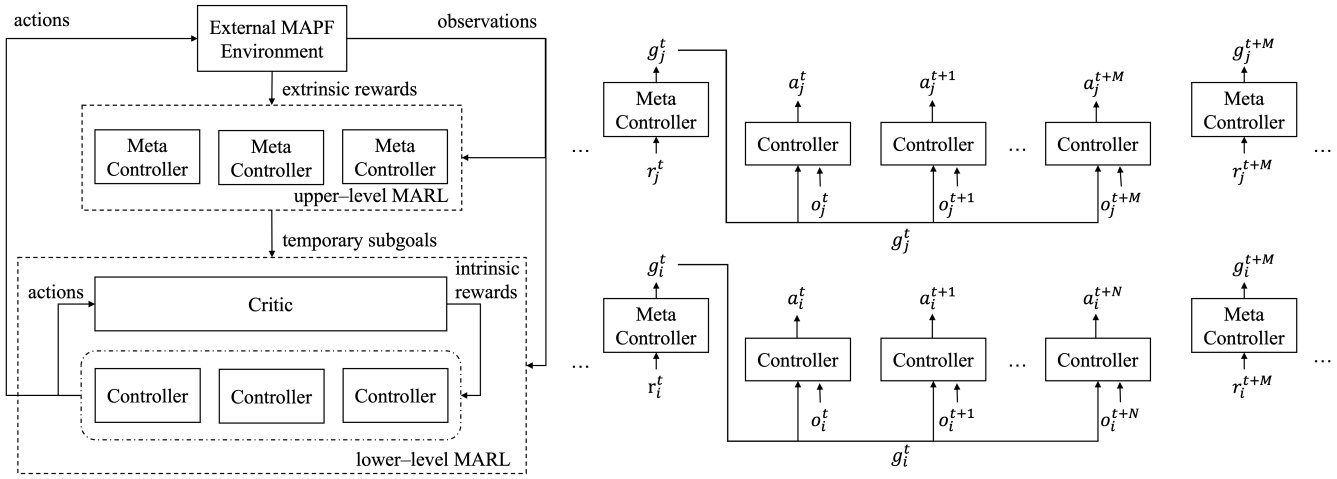


Fig. 2: Overview of the proposed framework. The meta controller perceives *region-wise observations* (denoted by r) and produces a *policy over subgoals*. The controller obtains *vertex-wise observations* (denoted by o) and produces a *policy over primitive actions* to accomplish the subgoal. The internal critic provides intrinsic reward to the controller based on the subgoals. The controller terminates either when the episode ends or when the subgoal is achieved. The meta controller repeatedly selects new subgoals until the task is completed or the episode ends.

within their communication range, which, in practice, is the same as the size of the agent's FoV, at a specific bandwidth. 4) The session between neighboring robots begins and ends instantly, without delay, in comparison to the time it takes for an agent to move. In other words, agents make decisions after sufficient exchanges of information.

A. Upper level: Intra-Region Path Planning

This paper focuses on MAPF instances in a 2D 4-neighbor grid world, where each agent or obstacle occupies precisely one grid. The framework partitions the grids into rectangular regions of equal size, determined by parameters W_r and H_r, which specify the width and height of each region, respectively.

We present a model for the hierarchical path planning using a factored Dec-POMDP model [11], which extends the standard Dec-POMDP model to long-horizon tasks where the transition time varies. The set of all regions, r, is $\mathcal{R}$, where each node represents a region, and edges denote connections between neighboring regions in a directed graph, $\tilde{G} = (\mathcal{R}, \tilde{E})$. Each region can accommodate multiple agents, allowing agents to travel through regions without collisions. Similar to standard MAPF, the i-th agent has a starting region, r_0^i, and a terminate region, r_g^i, while multiple agents are likley to share their starting and termination regions.

1) Upper-level Observation Representation: The upper-level observation features are characterizedbyatriad $\langle \mathcal{N}, \mathcal{T}, \tilde{\mathcal{H}} \rangle$, where $\mathcal{N}$ represents the current number of agents in the surrounding regions, $\mathcal{T}$ indicates the number of agents whose next target region is one of these regions, and $\tilde{\mathcal{H}}$ represents a four-layer one-hot encoding of the heuristic maps, which is illustrated in Fig. 4. Significantly, though no obstacles are present on the region-wise map, the upper-level heuristic maps take into account cases where there is no valid path connecting two adjacent regions. This is because the heuristic maps are computed based on the shortest valid single-agent paths, as detailed in Sec. III-B. The perceptions of the upper-level controller are bounded, and agents can only access information from nearby 5×5 regions with the agent's current region as the center.

2) Subgoals: As mentioned earlier, we use the notion *subgoals* (or *macro-actions*) to refer to our generalization of actions to include temporally extended courses of actions. We assume that $\mathcal{G}_i$ represents a finite set of subgoals for each agent, i, with $\mathcal{G} = \times_i \mathcal{G}_i$ as the set of joint subgoals. A *local subgoal* of agent, i, can be denoted by a triad:

$$\mathcal{G}_i = \langle \mathcal{I}_{g_i}, \beta_{g_i}, \pi_{g_i} \rangle, \tag{1}$$

consisting of an *initiation set* $\mathcal{I}_{g_i} \subseteq \mathcal{S}_i$, a *stochastic termination condition* $\beta_{g_i} : \mathcal{S}_i \to [0, 1]$, and a *policy* $\pi_{g_i} : \mathcal{S}_i \times \mathcal{A}_i \to [0, 1]$. In the context of HELSA, a subgoal can be described as identifying the next region that an agent plans to head for. In this case, a subgoal consists of a lower-level path finding policy, which is discussed in Sec. II-B, a termination condition for recognizing that the agent has reached the next region's boundary, and an initiation set containing states in which the agents are located at a newly arrived region.

The upper-level controller keeps track of when lower-level actions terminate, i.e., when the agent reaches the new region or its goal vertex, which is precisely when the agent needs to take a new macro-action.

3) Extrinsic Reward: We define the extrinsic reward function as $\mathcal{F} : \mathcal{R} \times \mathcal{G} \to \{0, 1\}$. The extrinsic reward signals received from the environment are positive if and only if the agent reaches its target vertex. The objective of the upper-level controller is to maximize cumulative extrinsic reward over long periods of time.

B. Lower level: Inner-Region Path Planning

The subtask assigned to the lower level is to achieve the subgoal derived from the upper level, which is to move to the boundaries between the current region and the next region as soon as possible while simultaneously avoiding collisions with other agents and obstacles. Suppose that the agent is at the goal region. In that case, the subgoal is precisely the final goal vertex of the agent, which is a particular case where only one subgoal vertex exists, and it does not necessarily locate at the boundary of the region. We now formulate the lower-level inner-region path planning problem with a Dec-POMDP model.

1) Lower-level Observation Representation: As previously mentioned, agents observe the environment inside their field of view. The observation information is grouped into three parts, which can be denoted by a triad $\langle \mathcal{A}, \mathcal{O}, \mathcal{H} \rangle$. Specifically, $\mathcal{A}$ and $\mathcal{O}$ are two binary matrices representing the locations of other agents and obstacles within the FoV, respectively. In addition, unlike in PRIMAL [4], where agents' targets are represented by a normalized vector pointing to their goals, we add 4-layer heuristic representations, $\mathcal{H}$, to the observation encoder, as illustrated in Fig. 4. The heuristic maps are more succinct and provide rich rational knowledge.

2) Primitive Actions: Agents move cardinally or stay still in the grid world at each discrete time step. During training and execution, agents may run into obstacles or collide with other agents from time to time, which would be fatal in a real-world deployment. Since the action space is relatively small, we do not mask out invalid actions, as as is done in most current approaches. Instead, we give negative rewards for these invalid actions during training. Once a collision occurs, we recursively recover the affected agents to their previous states until no collision exists. The experimental results in Sec. IV-D show that the number of collisions are under controll compared with other baselines.

3) Intrinsic Reward: Agents are subject to a penalty of -0.025 for each time step they do not remain at their designated goal vertices. This incentivizes agents to seek shorter paths to their goals. An agent receives a large positive intrinsic reward of 5 only upon accomplishing a subgoal, motivating them to work towards achieving intermediate objectives. When no collision occurs, the reward for the team is the average of each agent's reward. However, to discourage collisions, the team of agents receives a negative reward of -5 when any agents collide with an obstacle or other agents.

III. The Proposed Approach

In this paper, considering the challenges mentioned above, we propose the HELSA scheme to decouple the large-scale MAPF problems with spatiotemporal hierarchies. The proposed framework contains two levels of decentralized planners, the architecture of which is illustrated in Fig. 2.

A. Upper-level Subgoal Planning Method

Given the discrete action spaces of the upper-level subgoal planning sub-problem, we adopt the dueling DQN to solve the sub-problem for each agent with fast convergence. The detailed designs of the IQL-based algorithm are specified as follows.

1) IQL with Multi-Step TD Learning: To avoid confusion with lower-level states, we use the notion of r to refer to upper-level states. The independent Q-learning (IQL) estimates the following optimal Q-value function:

$$Q^*(r, g) = \max_{\pi_g} \mathbb{E}\left[\sum_{t'=t}^{t+N} f_{t'} + \gamma \max_{g'} Q^*\left(r_{t+N}, g'\right)\right] \quad (2)$$

where N denotes the number of time steps until the lower-level controller halts given the current goal. It is noteworthy that the subgoal is not a triad as previously mentioned, but a moving action in cardinal directions that indicates the relative direction of the next region.

To improve the sample efficiency, we adopt a multi-step bootstrapping technique, where the loss function is a multi-step TD error,

$$\mathcal{L}(\theta) = \text{Huber}\left(y_i(t) - Q_i\left(r_i(t), g_i(t); \theta_i\right)\right) \quad (3)$$

with the multi-step TD target,

$$y_i(t) = \sum_{j=0}^{m-1} \gamma^j f_i(t+j) + \gamma^m \max_{g_i} \tilde{Q}_i\left(r_i(t+m), g_i; \tilde{\theta}_i\right), \quad (4)$$

where $\sum_{j=0}^{m-1} \gamma^j f_i(t+j)$ is the expected extrinsic return of the i-th agent in the following m steps, $r_i(t)$ and $g_i(t)$ are the upper-level state and subgoal representation of the i-th agent in the t-th time step, and $\tilde{\theta}$ denotes the parameters of the target network, which is a periodical copy of the evaluation network parameters θ.

2) Distributed Priority Experience Replay: To increase the probability of learning high-reward sparse behavior, the HELSA adopts the distributed priority experience replay (PER) technique [12]. Instead of randomly selecting state-action pairs from the global experience replay buffer, the PER stores past experiences in a priority tree, where state-action pairs with higher step rewards have a higher probability of being sampled. This approach helps to improve the sample efficiency of the algorithm.

The observation encoder in the upper level follows the same structure as that in the lower level, as descirbed in Sec. III-B.

B. Communication-based Inter-Region Planning Method

Inspired by the heuristic guidance used in DHC [6] and graph attention networks used in MAGAT [13], for the lower-level planning sub-problem, we propose a communication-based method that incorporates the heuristic guidance to indicate goal locations and the two-stage attention mechanism to build a dynamic communication topology and determine the relative importance of the information exchanged.

1) CNN-based Observation Encoder: We extend the heuristic guidance proposed in [6] to multi-goal scenarios, as illustrated in Fig. 4. In cases where no obstacle exists on the map, there is always a unique shorted path toward the region's border. However, agents may have multiple

Algorithm 1 IQL-based Subgoal Planner

Initialize the shared replay buffer M and the exploration probability $\epsilon = 1$.
Initialize random parameters $\{\theta, \tilde{\theta}\}$ for the evaluation DQN $Q(\mathbf{r}, \mathbf{g}; \theta)$ and the target DQN $\tilde{Q}(\mathbf{r}, \mathbf{g}; \tilde{\theta})$, respectively.
for each episode **do**
  Initialize the lower-level state descirption s_i^0 and t_i ($i \in \{1, \ldots, M\}$) as the start and terminate states, respectively.
  Initialize the upper-level state description r_i and termination condition β_i ($i \in \{1, \ldots, M\}$).
  for each time step t **do**
    for each agent i **do**
      if $s_i^t \in \beta_i$ **then** ▷ With a subgoal accomplished, the DQN parameters are updated.
        Obtain extrinsic reward $f_i(r_i, g_i, r_i')$, and store transition (r_i, g_i, f_i, r_i') in the global replay buffer M.
        Compute TD Target using $\tilde{\theta}$, and perform gradient descent on θ to minimize multi-step TD error.
        Anneal ϵ and repalce $\tilde{\theta}$ with θ periodically.
        if $t_i \notin \beta_i$ **then** ▷ If the target region has not yet been reached, assign a new subgoal.
          $g_i \leftarrow EpsGreedy\,(r_i, \mathcal{G}_i, \epsilon, Q)$
          $\beta_i, r_i' \leftarrow ExpandGoal(g_i)$
      Sample a primitive action a_i^t via the lower-level controller.
      Execute a_i^t and observe next state s_i^{t+1}.
      Obtain intrinsic reward $\tilde{f}_i(s_i^t, a_i^t, s_i^{t+1})$, and update low-level actor and critic parameters.

potential optimal choices when taking into consideration the existence of obstacles. To obtain intuitive heuristics, the algorithm performs breadth-first search to obtain the lengths of shortest paths from each vertex to the given subgoals. Then the four channels illustrated in Fig. 4 are obtained, with each element representing a boolean value indicating a shorter distance away from the subgoals after executing the candidate action. Since all potential subgoals can be determined after the partition of the map, both the upper-level and lower-level heuristic maps can be computed in advance. During execution, a fraction of these maps can be sampled based on the agents' locations and FoV. We stack the heuristic channels along with the obstacle map and agent map and extract features using a CNN-based encoder. Between the stacked residual blocks, a skip connection joins together the features before and after the block. This structure is of great benefit to the generalization capability of feature extractors.

2) Two-Stage Attentional Communication: Each agent has a local share of the graph convolutional layers and communicates its observation encodings e_i^t with neighboring agents within its communication radius. Through this the agent enlarges its perception range and reinforces the connection between other agents. The fused observation features are then fed into the next module to perform decentralized decision-making with enriched information.

Inspired by G2ANet [14], we adopt a 2-stage attention mechanism to model the interplay of agents, where each agent does not have to cooperate with all nearby agents, and the extent to which each pair of agents is associated varies. As shown in Fig. 3, we model the relationship between agents using a graph, where each agent is a vertex and each relationship between neighboring agents is an edge. For each agent, i, we merge embeddings of agent i and j using an LSTM model:

$$h_{i,j} = FC(LSTM(h_i, h_j)), \tag{5}$$

where $FC(\cdot)$ represents a fully connected layer. We also employ a BiLSTM model to eliminate the input order's influence and capture all agents' information equally.

The hard attention model, whose value is calculated by Equ. 6, is a binary operator that specifies which agents each agent needs to interact with and discards the connection with the less relevant agents entirely, through which the agents focus on essential elements and the preliminary game abstraction can be achieved.

$$W_h^{i,j} = Gumble(h_{i,j}), \tag{6}$$

where $Gumble(\cdot)$ represents the gumble-softmax function employed to facilitate back propagation, which cannot be achieved by hard attention.

We also train a soft attention model to determine the weight $W_s^{i,j}$ of each edge in the graph G using a query-key mechanism, which represents the contributions from other agents to the learning agent a_i.

$$W_s^{i,j} \propto \exp\left(e_j^T W_k^T W_q e_i W_h^{i,j}\right) \tag{7}$$

with W_k transforming e_j into a key as well as W_q transforming e_i into a query. The final term $W_s^{i,j}$ represents the weight of each edge.

3) Training via COMA: Unlike previous learning-based MAPF approaches, which are usually based on independent Q-learning and suffer from insufficient information sharing, we adopt an effective policy-gradient-based learning scheme counterfactual multi-agent policy gradients (COMA). In this scheme, a centralized critic assesses the contribution of each agent's actions to the whole team's global return, promoting globally optimum policies. While the centralized critic is only required during training, only the actor net is needed during execution.

The key insight underlying COMA is for each agent i to compute an advantage function by subtracting the estimated

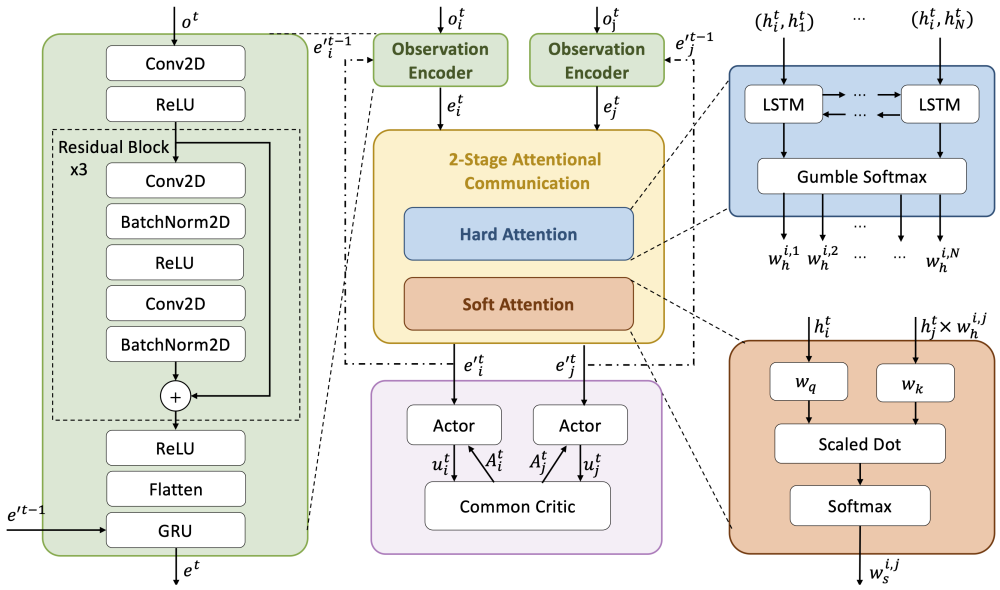


Fig. 3: The proposed lower-level planning framework.

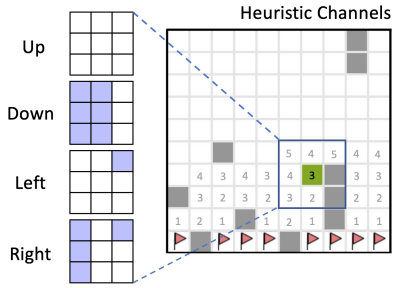


Fig. 4: Heuristic map calculated in the perspective of the green agent. The red flags represent the temporary subgoals, and the gray grids represent obstacles. The four channels indicate whether the agent gets closer to its subgoals by taking the certain action at these locations within agent's FoV respectively.

return for the current joint action with a counterfactual baseline that marginalizes out the agent's current action a_i:

$$A_i(\mathbf{s}, \mathbf{a}) = Q(\mathbf{s}, \mathbf{a}) - \sum_{a_i'} \pi_i\left(a_i \mid \tau_i\right) Q\left(\mathbf{s}, \left(\mathbf{a}_{-i}, \mathbf{a}_i'\right)\right), \quad (8)$$

where $\mathbf{s}$ and $\mathbf{a}$ represent the joint action and state of all agents, respectively, and τ_i represents the observation-action trajectories. The subtrahend constitutes the counterfactual baseline iterating over all possible actions of agent i except its current action while keeping all other agents' actions fixed.

With the help of the advantage function, the centralized critic reasons the contribution of each agent to the team's success and performs a reasonable multi-agent credit assignment while avoiding potentially expensive simulations. To further compress the output space of the network, the representation of the critic, of which the size scales linearly with the number of agents and actions, is adopted, allowing for efficient baseline evaluation and enabling great generalization.

TABLE I: Test scenario sets for experiments. Each scenario contains 100 randomly generated cases with the robot density fixed at 0.005 and the obstacle density fixed at 0.2.

map_size	n_robots	n_obstacles	max_epi_len
40 × 40	8	320	256
80 × 80	32	1280	512
160 × 160	128	5120	1024
240 × 240	288	11520	1536
320 × 320	512	20480	2048
400 × 400	800	32000	2560

It should be noted that partitioning the map into regions does not imply that each region maintains an independent MARL environment. The upper level only serves as a way to provide subgoals and simplify the task, making the overall learning process more efficient. The interactions between agents are still modeled at the lower level, within and across regions, through the lower-level controllers, comprising a common critic and M actors with parameter sharing.

IV. Empirical Evaluation

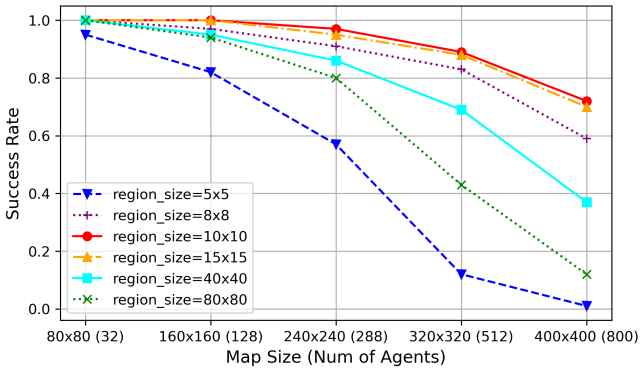
A. Metrics

The metrics of evaluation are listed as follows.

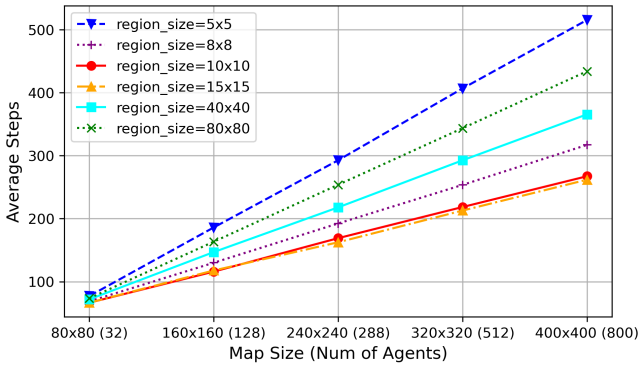
1) *Success Rate (SR):* The proportion of successful cases over the total number of tested cases. A case is considered *successful* or *complete* when all robots reach their goals before exceeding the maximum allowed steps, which is specified in Tab. I.
2) *Average Steps (AS):* The average number of time steps that all agents take to reach their respective goals, i.e., the average length of all agents' paths.

TABLE II: Overall performance comparison. All results are the average values of the 100 randomly generated test cases. In each column, ↑ indicates that higher values are preferred while ↓ indicates the opposite. The best result in each column is shown in bold, with results generated by our approach additionally highlighted in red.

Model	8 agents, 40-sized map, 0.2 density					32 agents, 80-sized map, 0.2 density					128 agents, 160-sized map, 0.2 density				
	SR ↑	AS ↓	MS ↓	CA ↓	CO ↓	SR ↑	AS ↓	MS ↓	CA ↓	CO ↓	SR ↑	AS ↓	MS ↓	CA ↓	CO ↓
PRIMAL [4]	**1.0**	56.49	98.90	0.42	**0.0**	0.88	164.39	305.73	4.12	**0.0**	0.07	356.51	1007.08	113.06	4.27
DHC [6]	**1.0**	31.40	55.77	0.38	**0.0**	0.98	69.18	139.77	3.20	**0.0**	0.87	132.31	399.19	29.38	0.06
DCC [7]	**1.0**	**28.84**	**50.49**	0.40	**0.0**	0.98	**64.47**	**134.34**	5.91	0.01	0.67	149.50	567.41	37.48	**0.0**
HELSA	**1.0**	29.71	52.29	**0.21**	**0.0**	**1.0**	65.85	136.17	**0.54**	**0.0**	**0.97**	**126.51**	**296.14**	**3.69**	**0.0**
Model	288 agents, 240-sized map, 0.2 density					512 agents, 320-sized map, 0.2 density					800 agents, 320-sized map, 0.2 density				
	SR ↑	AS ↓	MS ↓	CA ↓	CO ↓	SR ↑	AS ↓	MS ↓	CA ↓	CO ↓	SR ↑	AS ↓	MS ↓	CA ↓	CO ↓
PRIMAL [4]	0.0	530.06	1536.0	593.59	34.48	0.0	736.50	2048.0	1498.20	173.49	-	-	-	-	-
DHC [6]	0.70	193.13	804.55	99.52	**0.01**	0.53	252.62	1304.48	236.22	0.30	0.40	315.08	1906.36	468.61	0.71
DCC [7]	0.19	235.32	1375.04	151.88	12.97	0.04	300.78	2020.76	423.40	57.41	-	-	-	-	-
HELSA	**0.93**	**175.56**	**629.58**	**49.41**	0.03	**0.87**	**221.17**	**935.99**	**101.78**	**0.04**	**0.74**	**268.83**	**1211.15**	**269.67**	**0.37**



(a) success rates



(b) average steps

Fig. 5: Simulation results of our approach with different region sizes at test scenarios described in Tab. I.

3) *Makespan (MS):* The maximum number of time steps for all agents to reach their goals, i.e., the length of the longest path among all agents.
4) *Collisions with Agents (CA)*: The number of edge and vertex conflicts with other agents.
5) *Collisions with Obstacles (CO)*: The number of collisions with obstacles.

B. Experimental Setup

To compare the generalization capability of our proposed approach with other baselines, we prepare test datasets according to Tab. I. We train our models with 40×40 maps and 8 robots with a fixed obstacle density of 20% and then test on all scenarios shown in Tab. I.

The size of FoV is 11×11 for the proposed lower-level planner as well as for all other baselines, where the agent is at the center of the FoV. The maximum allowed step size during training is 256. The size of each region is 10×10 and the effect of the region size is studied in Sec. IV-E.

As for the training settings, the upper-level model is trained with a batch size of 32. The discount factor in the calculation of multi-step TD error in Equ. 4 is set to $\gamma = 0.99$, and the length of the steps is set to 2. The lower-level planner is trained by optimizing the actor and critic networks, both of which are trained via the RMSProp optimizer with a learning rate of 0.001 and a decay rate of $\alpha = 0.99$.

C. Baselines

in Sec. IV-D, we compare the HELSA with one of the most distinguished approaches named PRIMAL [15], and two most recent communication-based planners named DHC [6] and DCC [7].

In Sec. IV-F, we build two ablations with simpler lower-level controllers to evaluate whether the two-stage attentional communication mechanism in the proposed lower-level controller works.

D. Results

Tab. II shows the comparison between our proposed HELSA framework and other baselines in various scales of test scenarios with fixed robot and obstacle density. Our approach outperforms PRIMAL in all the test cases in terms of success rate, average steps, and makespans, especially in the larger maps. This comparison indicates that it is challenging for the fusion of imitation learning and reinforcement learning in PRIMAL to strike a balance between exploration and exploitation, which makes PRIMAL hard to extend to large-scale and long-horizon tasks. The two communication-based approaches yield excellent results in relatively small-scale scenarios, primarily due to sufficient information exchange among agents and informative observation encodings. Considering the inevitable information loss

TABLE III: Evalutaion of the adopted lower-level controller with other ablations in terms of success rates and average steps.

Method	w/ hierarchy?	80-sized map		160-sized map		240-sized map		320-sized map		400-sized map		Avg.	
		SR ↑	AS ↓	SR ↑	AS ↓	SR ↑	AS ↓	SR ↑	AS ↓	SR ↑	AS ↓	SR ↑	AS ↓
COMA+Comm +Attention	✓	**1.0**	**65.85**	**0.97**	**126.51**	**0.93**	**175.56**	**0.87**	221.17	0.74	268.83	**0.90**	**171.58**
		0.98	67.25	0.76	141.95	0.41	219.03	0.07	287.99	0.0	347.63	0.44	212.77
COMA+Comm	✓	**1.0**	66.78	0.95	130.20	0.90	182.13	0.86	**219.75**	**0.77**	**245.00**	0.90	172.97
		0.98	69.89	0.72	147.77	0.35	233.98	0.09	311.93	0.0	387.54	0.43	230.22
COMA	✓	0.95	96.30	0.83	193.39	0.44	323.95	0.04	433.73	0.0	615.19	0.45	332.51
		0.90	139.13	0.43	248.67	0.12	477.53	0.01	633.55	0.0	883.10	0.29	476.40

caused by problem decomposition, it is not surpirsing that our approach leads to longer steps for agents to reach their goals. It is essential to note that though DCC performs better than DHC and HELSA in smaller scenarios due to its focus on important relations, it experiences rapid deterioration of performance as the scenarios become larger. This indicates that sufficient information exchange is a key factor in large-scale planning. Through the hierarchical decision-making framework, each agent makes decisions based on information beyond its lower-level perception and communication range and collaborate with more distant agents in an indirect way through intra-region cooperation. The hierarchy of policies is more robust and scales linearly with the scope of the problem, which can be inferred from the simulation results of larger-scale scenarios.

E. Study on the Effects of Partitioning Granularity

The configuration of region size is an important hyperparameter that presents a dilemma: a small region size puts to much burden on the upper-level controllers, aggravating the performance degradation caused by problem decomposition, while a large region size can make a solution gradually degenerate into one without hierarchical coordination. We conduct extensive experiments with different region sizes in the test scenarios with various scopes, and the results are shown in Fig. 5. Our finding suggest that when region size of the execution phase is close to the that of the training phase, which is 10 in practice, the algorithm yields the highest success rates. Conversely, the solutions get worse as the granularity gets finer or coarser, i.e., the region size gets smaller or lager, resulting in a mountain valley distribution.

F. Ablation Studies

We evaluated the effectiveness of the two-stage communication mechanism in the lower-level controller by introducing two ablations with simpler structures. The results, shown in Tab. III, demonstrate that the communication mechanism dramatically improves success rates in all of the test scenarios. However, we also observed that while the two-stage attention mechanism can facilitate collborative decision making in relatively small scenarios, the discard of some messages or connections may harm coordination in larger scenarios, which confirms our previous conclusion. We also noticed that no matter which lower-level controller is adopted, the hierarchical framework can improve success rates in a great measure.

V. Conclusions

This paper addresses the large-scale MAPF problem. We propose the HELSA framework to tackle the problem of sparse reward and long horizon. Experiments show that our approach performs significantly better in large-scale multi-robot routing tasks in success rates, makespans, and collision rates than state-of-the-art learning-based planners.

References

[1] P. R. Wurman, R. D'Andrea, and M. Mountz, "Coordinating hundreds of cooperative, autonomous vehicles in warehouses," *AI magazine*, vol. 29, no. 1, pp. 9–9, 2008.

[2] J. Yu and S. M. LaValle, "Structure and intractability of optimal multi-robot path planning on graphs," in *Twenty-Seventh AAAI Conference on Artificial Intelligence*, 2013.

[3] R. Stern, N. R. Sturtevant, A. Felner, *et al.*, "Multi-agent pathfinding: Definitions, variants, and benchmarks," in *Twelfth Annual Symposium on Combinatorial Search*, 2019.

[4] G. Sartoretti, J. Kerr, Y. Shi, *et al.*, "Primal: Pathfinding via reinforcement and imitation multi-agent learning," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2378–2385, 2019.

[5] Z. Liu, B. Chen, H. Zhou, *et al.*, "Mapper: Multi-agent path planning with evolutionary reinforcement learning in mixed dynamic environments," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 11 748–11 754.

[6] Z. Ma, Y. Luo, and H. Ma, "Distributed heuristic multi-agent path finding with communication," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 8699–8705.

[7] Z. Ma, Y. Luo, and J. Pan, "Learning selective communication for multi-agent path finding," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1455–1462, 2021.

[8] S. Ren, K. He, R. Girshick, *et al.*, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015.

[9] R. S. Sutton, D. Precup, and S. Singh, "Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning," *Artificial intelligence*, vol. 112, no. 1-2, pp. 181–211, 1999.

[10] E. Boyarski, A. Felner, R. Stern, *et al.*, "Icbs: Improved conflict-based search algorithm for multi-agent pathfinding," in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[11] C. Amato, G. D. Konidaris, and L. P. Kaelbling, "Planning with macro-actions in decentralized pomdps," in *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-Agent Systems*, 2014, p. 12731280.

[12] D. Horgan, J. Quan, D. Budden, *et al.*, "Distributed prioritized experience replay," in *International Conference on Learning Representations*, 2018.

[13] Q. Li, W. Lin, Z. Liu, *et al.*, "Message-aware graph attention networks for large-scale multi-robot path planning," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5533–5540, 2021.

[14] Y. Liu, W. Wang, Y. Hu, *et al.*, "Multi-agent game abstraction via graph attention neural network," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, 2020, pp. 7211–7218.

[15] M. Damani, Z. Luo, E. Wenzel, *et al.*, "Primal _2: Pathfinding via reinforcement and imitation multi-agent learning-lifelong," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2666–2673, 2021.