Towards Deployment-Efficient and Collision-Free Multi-Agent Path Finding* (Student Abstract)

Feng Chen[1], **Chenghe Wang**[1], **Fuxiang Zhang**[1], **Hao Ding**[1],
Qiaoyong Zhong[2], **Shiliang Pu**[2], **Zongzhang Zhang**[1]

[1] National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China
[2] Hikvision Research Institute, Hangzhou 310051, China
{chenf, wangch, zhangfx, dingh}@lamda.nju.edu.cn, {zhongqiaoyong, pushiliang.hri}@hikvision.com, zzzhang@nju.edu.cn

Abstract

Multi-agent pathfinding (MAPF) is essential to large-scale robotic coordination tasks. Planning-based algorithms show their advantages in collision avoidance while avoiding exponential growth in the number of agents. Reinforcement-learning (RL)-based algorithms can be deployed efficiently but cannot prevent collisions entirely due to the lack of hard constraints. This paper combines the merits of planning-based and RL-based MAPF methods to propose a deployment-efficient and collision-free MAPF algorithm. The experiments show the effectiveness of our approach.

Introduction

Pathfinding is a fundamental form of many practical scenarios, such as robot routing and GPS navigation, and has been extensively studied. Multi-agent pathfinding (MAPF) occurs when multiple agents in a shared system find their way while avoiding collisions. We consider an extended setting called lifelong MAPF (Damani et al. 2021), where agents are assigned a new task immediately after completing one.

Planning-based approaches take inspiration from traditional search algorithms such as the A* algorithm (Hart, Nilsson, and Raphael 1968). RL-based methods, on the other hand, have the potential to solve the lifelong MAPF problem effectively due to their distributed decision-making approach. Current RL-based MAPF methods, such as PRIMAL$_2$ (Damani et al. 2021), introduce carefully designed observation spaces and reward functions to make the policy behave as we want. However, agents cannot completely avoid collisions with soft penalties in such rewards.

This paper proposes a novel framework that combines RL and planning to solve the lifelong MAPF problem. We train the agents with VDN (Sunehag et al. 2018) for explicit coordination. A classifier and a replanner are then introduced to detect and resolve collisions by local planning. We evaluate our approach in a realistic automated storage scenario and achieve excellent performance with complete collision avoidance. Test results in unseen maps further illustrate the generalization ability.

Approach

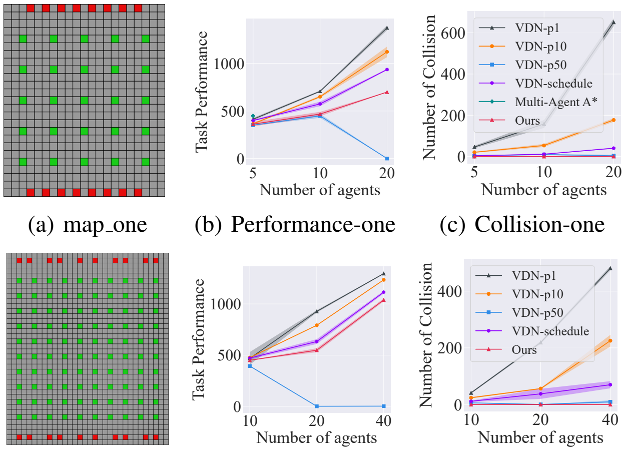
Our approach applies a hierarchical framework to balance efficiency and safety (collision avoidance). Most of the time, the agents are governed by an efficient policy obtained by the RL algorithm. In scenarios where collisions may occur, we hand over control to the planning module to avoid collisions.

RL Module VDN is a scalable MARL framework that allows decentralized execution and optimizes the sum of individual Q-values during training, explicitly facilitating coordination. To adapt VDN to our task, we must clearly define a Dec-POMDP for MAPF problems. Specifically, we define state $s \in S$ as global information of the map, including agents and carrying tasks. The o_i of each agent $i \in N$ represents its individual observation, typically the local information around it. In terms of reward function, we define a global reward function as $r = \lambda_1 \sum_{i=1}^{N} \mathrm{done}_i - \frac{\lambda_2}{N} \sum_{i=1}^{N} \mathrm{dis}_i - \lambda_3 \#\mathrm{coll} - \lambda_4 \#\mathrm{stag}$, where done_i represents whether agent i has reached its current goal at this timestep, dis_i represents agent i's distance to its current goal, $\#\mathrm{coll}$ represents the number of collision events that have just occurred, and $\#\mathrm{stag}$ represents the number of stagnant agents for a period of time. The first two terms guide agents to reach their goals fast. The third term can play a role in preventing agents from collisions, but may also curb agents' exploration and learning. To prevent possible negative effects, we devised the forth term to encourage exploration of agents.

Re-planning Module VDN policies can achieve relatively good coordination among agents, have the potential to solve general problems, and can be deployed efficiently. However, collision events are treated as penalties during the VDN's training phase, and it is almost impossible for the algorithm to avoid collisions 100% of the time, as the agents always learn to trade off task completion and collision avoidance. To address this limitation, we transfer the control to a planning-based algorithm when the probability of causing a collision is high. Therefore, the two key questions are: (1) How to judge whether a control switch is required? (2) How to make good plans for the local agents likely to collide?

*Corresponding author: Zongzhang Zhang. This work is supported by the NSFC (No. 62276126) and Hikvision Open Fund (CCF-HIKVISION OF 20210001).

To answer (1), we pre-train a collision detector, which takes local observations of two agents as input and predicts whether they will collide in the future. In practice, we roll out many trajectories of learned policies by VDN and collect a supervised learning dataset, where features are agents' observations and labels indicate whether they collide soon. With the collision predictor, we perform a control switch when the predicted probability is over a fixed threshold. Regarding (2), we can obtain the pair relationships about possible collisions between agents by the detector, which can be seen as edges connecting agents. For each connected agent set, we only plan actions locally. Specifically, we apply a heuristic-based planning method, where agents plan sequentially in a fixed order of priorities (e.g., id) to reduce their search space. Each agent prefers actions derived from the RL policy and searches for other actions when it encounters collisions with higher priority agents during planning.



(a) map_one (b) Performance-one (c) Collision-one

(d) map_two (e) Performance-two (f) Collision-two

Figure 1: Two storage scenarios and experimental results on them. Our method can have comparable performance to VDN baselines while achieving zero collision.

Experiments

To test the effectiveness of our approach, we design storage maps as grid worlds of different complexity in Fig. 1. We also designed map_three and its results are in the appendix[1]. In the beginning, n agents are initialized randomly on the paths (grey) with no loads and are assigned different tasks, including the loading port (red) and the goal shelf (green). A complete task consists of loading the package, delivering it to the target shelf, and unloading it.

In our experiments, we set all hyperparameters fixed except λ_3. For VDN baselines, we set $\lambda_3 = \alpha$ for VDN-pα and gradually tune λ_3 from 1 to 50 for VDN-schedule and our hybrid algorithm. For each map, we build scenarios with different numbers of agents, and the mean results of five random seeds are shown in Fig. 1. In Fig. 1(b) & 1(e), the algorithm's performance is quantified by $10000 \times \frac{\#\text{TasksDone}}{\#\text{Timesteps}}$, where $\#\text{TasksDone}$ counts the number of carrying tasks done within fixed timesteps, and $\#\text{Timesteps}$ indicates the total timesteps. In Fig. 1(c) & 1(f), collision is defined as the number of collisions occurred within fixed timesteps. In practice, we counted the results within 300 timesteps.

Training VDN with different λ_3 can make agents weigh between completing tasks and avoiding collisions. As we can see in Fig. 1, setting $\lambda_3 = 1$ makes agents pay little attention to collision avoidance and just get the transport done. If we set a large $\lambda_3 = 50$ for VDN-p50, collisions are mostly avoided, but the agents can hardly learn to complete tasks since the hefty penalty of collision intimidates agents from exploration and completing tasks. Thus, we come up with a schedule of λ_3 and train VDN policies with a minor penalty of collisions at first to encourage the agents to learn to complete carrying tasks as efficiently as possible, regardless of the collisions. Gradually, we increase the penalty of collisions, and agents can learn to adapt their policies and try to avoid collisions while still completing the tasks. However, since the penalty is a soft constraint, collisions still occur occasionally, which is intolerable in real-life applications. Planning-based algorithms like A* can achieve outstanding performance and zero collision, but frequent re-planning is inefficient. Our method introduces a re-planning module so we can entirely avoid collisions while maintaining competitive performance. As shown in Fig. 1, our method can be efficiently deployed with RL policies and guarantee safety with no collision. More results of map_three and a report of *computational time* can be found in the appendix. In addition, we train our MAPF policy for 20 agents on some extra generated scenarios. While keeping collision-free, our policy has the performance of 506 ± 3.66, 748.83 ± 5.17, and 425.66 ± 1.34 when testing on unseen map_one, map_two, and map_three, respectively. With little performance drop, it verifies the generalization ability of our approach.

Conclusion and Future Work

This paper proposes a novel framework combining RL and planning for solving lifelong MAPF problems and the experiments show the effectiveness of our framework. An interesting future work is to transfer policy to scenarios with different terrains and agent numbers without extra training.

References

Damani, M.; Luo, Z.; Wenzel, E.; and Sartoretti, G. 2021. PRIMAL$_2$: Pathfinding via Reinforcement and Imitation Multi-Agent Learning - Lifelong. *IEEE Robotics and Automation Letters*, 6(2): 2666–2673.

Hart, P. E.; Nilsson, N. J.; and Raphael, B. 1968. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2): 100–107.

Sunehag, P.; Lever, G.; Gruslys, A.; Czarnecki, W. M.; Zambaldi, V.; Jaderberg, M.; Lanctot, M.; Sonnerat, N.; Leibo, J. Z.; Tuyls, K.; and Graepel, T. 2018. Value-Decomposition Networks For Cooperative Multi-Agent Learning Based On Team Reward. In *AAMAS*, 2085–2087.

[1] https://www.lamda.nju.edu.cn/chenf/aaai23_appx.pdf