EAI Endorsed Transactions
on AI and Robotics

Research Article EAI.EU

Designing Automation for Pickup and Delivery Tasks in Modern Warehouses Using Multi Agent Path Finding (MAPF) and Multi Agent Reinforcement Learning (MARL) Based Approaches

Shambhavi Mishra[1,*], Rajendra Kumar Dwivedi[2]

[1]Madan Mohan Malaviya University of Technology, Gorakhpur, India
[2] Madan Mohan Malaviya University of Technology, Gorakhpur, India

Abstract

A warehouse pickup and delivery problem finds its solution using multi agent path finding (MAPF) approach. Also, the problem has been used to showcase the capabilities of the multi agent reinforcement learning (MARL). The warehouse pickup and delivery work needs the agent to pick up a requested item and successfully deliver it to the intended location within the warehouse. The problem has been solved based on two approaches that include single shot and lifelong problem solution. The single shot solution has the delivery as the final goal and thus once it reaches the delivery address, it stops whereas in case of lifelong, the agent needs to deliver the item which it had picked, deliver it to the required place and then again pick up new item until requests are satisfied. The strategy used by multi agent path finding (MAPF) approach aims at constructing collision free paths to reach the delivery location but in case of multi agent reinforcement learning (MARL), the agents' decision-making tactics (or policies) are learned which are then used to help agents decide path to be followed based on environment state and agent's position. The results show that the lifelong conflict-based search (CBS) is a better option when the agents are less in number as in that case, the re-planning will take overall less time but when the agents are large in number then this re-planning can take very long to produce conflict free paths from source to goal nodes. In this case, shared experience action critic (SEAC) which is based on multi agent reinforcement learning (MARL) approach can be more efficient choice as it takes the current environment state to give the most suitable action for that time t. For this study the agents taken for learning are homogeneous in nature that can pickup and deliver any type of requested item. We can address the same pickup and delivery problem when the agents are not all same and differ in their capabilities and the type of item they can handle.

Keywords: Multi agent pickup and delivery problem, multi agent reinforcement learning (MARL), multi agent path finding (MAPF)

Received on 14 June 2023, accepted on 09 March 2024, published on 18 March 2024

doi: 10.4108/airo.3449

*Corresponding author. Email:mishra.shambhavi33@gmail.com

1. Introduction

The applications of multi agent path finding range from automating warehouse material movement to allocating schedule for airport operations. The warehouse movement is restricted by the large storage spaces that are usually located at the centre of the warehouse and are visited frequently for picking up the required item as demanded by stations that further process these items. Due to this occupancy the space which is used for travelling becomes narrow and can only allow single bot to pass at a time. To this end in a multiple agent system, it is required to guide these bots so that their movement is collision free and there is very less chance of item being damaged as a result of collision. In their way to the target node, the agents need to avoid obstacles and other agents and form a path through the warehouse to the intended position.

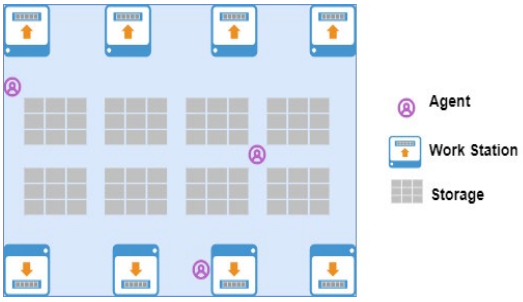

Figure 1. A Warehouse State with Multiple Workstations and Agents

Figure 1. shows the idea of a warehouse that contains several workstations, and agents that move in the corridors to pick items from storage area (located at the centre) and deliver it to the respective workstations.

1.1. Motivation

The multi agent pickup and delivery problem finds solution in both single shot and lifelong MAPF where the inner loops of lifelong strategy contain solution to single shot MAPF. With the increase in the number of agents, the environment size and other such factors, the number of possible collision free paths that can be traversed by an agent to reach the ultimate goal of dropping the requested item to the goal location also increases. To this end, the multi agent reinforcement learning can be put in use due to its capabilities of considering joint observation states and joint action space in order to reach from source to goal.

1.2. Problem Statement

The multi agent pickup and delivery warehouse problem aims at devising number of collision free paths using which the agents reach the goal through the start node (where the warehouse has several requests at a time t and different agents work simultaneously to deliver each of the item they picked up). This problem finds its application in various sectors where inventories are divided into small divisions and assembly of parts needs to be carried out to produce a final or even an intermediate item.

A multi agent path finding problem within a two-dimensional space can be represented as a graph. Given an undirected graph with k agents, acting in a directed graph $G = (V, E)$ for which every agent $a \in \{1,2. . . , k\}$ has a starting vertex $s_i \in V$ and a goal vertex $f_i \in V$. A path in G is a sequence of vertices $p = v_1v_2 ..... v_m$ so that $(v_j, v_{j+1}) \in E$ for all $1 \leq j < m$. Given paths $p\_1 = v_1v_2 . . . v_m$ and $p\_2 = v'_1v'_2 . . ... v'_m$ in graph for $m > 1$, the two paths p_1 and p_2 will be collision free iff:

(i) $v_j != v'_j$ (or no common vertex in the traversed path),

(ii) $(v_j, v_{j+1}) != (v'_{j+1}, v'_j)$ (or no common edge in path).

1.3. Contribution

In this study, we compare the two types of solutions, one, based on multi agent path finding and the other based on the multi agent reinforcement learning along with their benefits in different sized environment space. We compare the conflict-based path selection with the shared experience action critic reinforcement learning approach for multi agent pickup and delivery problem. We evaluate all the strategies and show that the changing environment states at every time stamp t can be handled easily in a multi agent reinforcement learning framework. We show that as the number of agents increase the conflict-based search approach shows lower chances for convergence and it can take very long to reach to a solution. In such cases, the shared experience approach fits well. Both mean makespan value and the mean flow time are used to evaluate the two strategies considering medium and large environment sizes and for different number of total agents present for training.

1.4. Organization

The remaining parts of the paper are arranged in the following manner. We discuss some necessary concepts required to better understand the study in section 2 which is background. We then do a comparative review of the existing previous works in the literature survey part in section 3. Section 4 contains details about the two different strategies used for multi agent pickup and delivery problem and their working process. In section 5 we evaluate the two strategies and compare the results obtained by the two methods based on few metrics. Section 6 presents the conclusion and existing challenges along with future scope.

2. Background

This section contains the background in which we have covered few concepts or the preliminaries that will be needed to understand the coming sections.

2.1. Single Agent Reinforcement Learning

Given an environment E the agent a learns to reach to the goal starting from the source based on rewards and penalties received for each step. The environment here is completely observable and the next action gets decided by the current environment state. The agents learn the policy that helps it maximize the cumulative reward at the end of the training.

2.2. Multi Agent Reinforcement Learning

Multi-agent reinforcement learning introduces the concepts of joint action and joint observations, and it studies how multiple agents [17] interact in a common environment. That is, when these agents interact with the environment and one another, can we observe them collaborate, coordinate, compete, or collectively learn to accomplish a particular task. Multi-agent reinforcement learning learns the agent behaviour in the same environment. The agents may work towards a common goal or against each other or a combination of the two. Based on the interaction and behaviour of agents in the environment with respect to other agents, the learning can be categorized as under:

Cooperative Agents

The agents work to accomplish some common goal and thus coordinate with other agents. The cooperation here is important because each agent present in the environment [11] has only a partial observation of the environment. In this case the agents coordinate [20-21] with each other to get an overall status of the environment and then taken action in order to accomplish the assigned work.

Competitive

Agents compete with other agents. They device different strategies based on actions of other agents to fulfil their respective goals. Hide and seek between two such agents is a suitable usecase to be accomplished with competitive agents. The seeker learns and updates its learning based on the actions or moves taken by the hiding agent. Similarly, the hiding agent further changes and refines its strategy to beat the seeker. Hence, both compete and learn to maximize their objective function.

Combination of the Two

This can be easily understood by an example of hockey where agents of one team coordinate among themselves while at the same time they compete with agents of the other team. Both teams learn and share strategy with similar agents and devise policy to win over the other team. This involves both, making better strategy for reward (a goal) and at the same time also make a strategy for countering the actions of the rival agents. These are the three types in which agent behave and learn the policies depending on the goal or the usecase for which they are used.

2.3. Single shot Multi Agent Path Finding

Here there are n agents (a1, a2, a3,..., an) in the environment and all these agents have source and goal as the attributes. The coordinates in the environment can be represented by an un-weighted and undirected graph and at each time step, the agent can either move to the next node to which its current position is directly connected, or it can remain it its current position without moving [13]. The decision of movement towards the adjacent node as well as to stay on the current position both take unit cost. In this case there can be two conflict types, one, that takes place when two agents decide to move to a common position i.e, ai and aj both lands up at same vertex v. The second conflict happens when the edge being used by two agents is same (i.e, both visiting the edge in opposite direction). These collisions and obstacles present in the path must be avoided to reach the target position. The request for the items is picked by each agent working in the same environment choosing paths leading them towards their goal.

Figure 2. An Environment Snippet at Time t with two Agents

In the figure 2.1, we can see two agents in yellow circle, the requested items are shown in red rectangular box while the agent that has picked up an item and moving towards goal (shown in blue at the bottom) are shown by red box containing yellow circle inside it.

2.4. Lifelong Multi Agent Path Finding

In case of single shot multi agent path finding [12] approach, the agent stops when it has reached a delivery spot and dropped the requested item there. But in lifelong multi agent path finding approach, the agent again moves to the next goal location and has the task of delivery to next location in the warehouse or we can say that in the inner loop, the lifelong approach solves many single shot problems.

2.5. Types of Conflicts in Multi Agent Path Finding

In the process of navigating to solution, there can arise a number of non-acceptable moves that we call as conflicts

[22]. Since the corridors where the agent needs to work are very narrow, it gives rise to different types of conflicts namely edge conflict, vertex conflict, following conflict, cycle conflict and a swapping conflict.

Edge Conflict in MAPF

An edge conflict occurs if two agents at time step t are traversing the same edge $(v_i, v_j) \in V$ since this can result in a collision, so the constraint tree maintains a constraint at the top node to make sure this transition is not opted for when agent is in position A making a move towards B. If P_1 and P_2 are two paths formed by agent ai and aj then an edge conflict is present if $P_1[t] = P_2[t]$.

Vertex Conflict in MAPF

A vertex conflict is one in which two agents initially are at different vertices (shown by nodes A and B in figure (b)) and they end up reaching to a common vertex resulting in a collision (node C is the common node in example below). Thus, either of the two actions will be executed to avoid the conflict in the final solution to multi agent pickup and delivery problem.

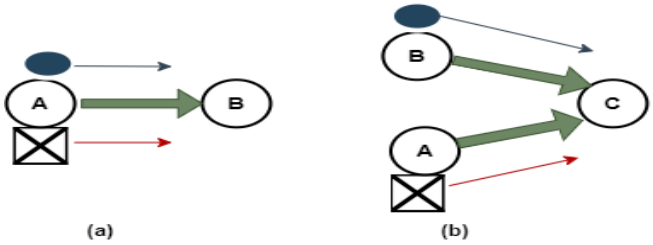

Figure 3. Edge Conflict (a) and Vertex Conflict Scenarios that May Arise in Conflict Based Search

Figure 3. shows the two collision types that we further consider in section 4 where we derive the solution using conflict-based search strategy. We do not consider other mentioned conflicts i.e, cycle conflict, following conflict and swapping conflicts in process of constructing collision free paths.
A Markov game is defined by the tuple $(N, S, \{O_i\}i\in N, \{A_i\}i\in N, P, \{R_i\}i\in N)$, where agents

ai ∈ N = {1, 2,. . . , N},
S = state space,
O = O1 × . . . × ON , combined observation space
A = A1×. . .×AN , combined action space,
Ri = individual agent's reward at time t.

The agents do not have full observation of the environment and take action based on their local observation. In case of multiple agents, the shared observation gives the agent an overall understanding of the environment. The agent works to maximize the reward and select the policy that results in maximum value of combined rewards.

2.6. Multi Agent Path Finding (MAPF) versus Multi Agent Reinforcement Learning (MARL)

MAPF and MARL are two approaches used widely to tackle multi agent pickup and delivery problem. In the proceeding sections we discuss and evaluate the conflict-based search (CBS) which is a MAPF based solution and works by constructing collision free paths. We then compare CBS with shared experience actor critic (SEAC) which is based on MARL approach.

3. Literature Review

A warehouse pickup and delivery problem finds its solution using multi agent path finding algorithm. Also, the problem has been used to showcase the capabilities of the multi agent reinforcement learning (MARL). The warehouse pickup and delivery work need the agent to pick up a requested item and successfully deliver it to the intended location within the warehouse.

The problem has been solved based on two approaches that include single shot and lifelong problem solution. The single shot solution has the delivery as the final goal and thus once it reaches the delivery address, it stops whereas in case of lifelong, the agent needs to deliver the item which it had picked, deliver it to the required place and then again pick up new item until requests are satisfied.

The strategy used by multi agent path finding algorithm aims at constructing collision free paths to reach the delivery location but in case of multi agent reinforcement learning, the agents' decision-making tactics (or policies) are learned which are then used to help agents decide path to be followed based on environment state and agent's position.

The solution to the multi agent path finding problem can be centralized where a single machine is used to guide agents towards their goal and agents needs to follow the directions. The other solution is decentralized meaning the agents have intelligence of their own and they cooperate with each other to reach to their assigned goals.

Frans et. al [1] used the concept of centralized training and decentralized execution where the agents could share each other's learning at the time of training but while performing the task after training was over, they need to take individual steps which they derived from all the combined learning which they gained.

Sinno et. al [2] used another paradigm where some agents act like guides and give their comments or remarks for actions taken by other agents. The agents thus act as teachers for other agents who work according to the feedback, they receive corresponding to the moves made by them in order to carry out and execute the task given to them.

Volodymyr et. al [4] uses distributed reinforcement learning where they work on off policy strategy where

target policy differs from the behavioural policy. The policy which the agents need to learn are thus different from the actions the agents take while working in the environment. It is just the opposite of on policy strategy where the learned policy is same as the actions taken by agent in the process of the completion of the assigned tasks.

Table 1. shows a comparative study between some of these approaches used for multi agent path finding problem.

S. No.	Author	Technique Used	Remarks
[1]	Frans et. al	Centralized training decentralized execution	Sharing of experience is limited to training time only and not at actual execution
[2]	Sinno et. al	Teacher learner based transfer learning approach	Teacher-learner protocol is followed
[3]	Stefan et. al	Demonstration of learning by other agents	Early policy is provided to agent which is then improved over time.
[4]	Volodymyr et. al	Distributed reinforcement learning	Target Policy not equal to Behavioural Policy (off policy)
[5]	Tonghan et. al	MARL with emergent roles (CTDE based)	Sub –tasks need to be identified to enable role wise learning
[6]	Kottinger et. al	Guided Conflict based search strategy is used	Here the segments of solution are first constructed which are then validated
[9]	Mehul et. al	Conflict based search strategy	Role based agents used to handle specific tasks
[9]	Ma et. al	Prioritized Planning	Agents are assigned priorities and the path planning is done according to the decreasing priority.
[10]	Christianos et. al	Selective Parameter Sharing for Scalability	The agents are grouped into clusters based on similar roles and abilities and the learned parameters are only shared among same cluster agents.
[19]	Wan et. al	Provides Solution to Dynamic MAPF	Use of outdated node and outdated constraint concept to smoothly include more number of agents over time.

Tonghan et. al [5] also used the same approach with an addition of the role specific or selected learning which the agents shared. This was done by first assigning the agents into groups based on the task they performed in the warehouse. Then the group learned the policies used by other agents belonging to the same group. It was found that the strategy worked well, and similar agents learned better even when not having similar rewards. Here the sub-tasks need to be identified before the agents are assigned with it.

Kottinger et. al [6] proposed solution to explainable multi agent path finding problem by representing solution with set of k segments each one denoting a path phase of all agents present in the environment. The segments are used to validate the solution and check for any collisions that the path may have led to in the given environment. Ma. et. al [9] have used a priority-based method to MAPF problem. The agents in the environment are first assigned some priority and the path planning is done according to these priorities. The drawback of the priority-based approach is its sub-optimal solution. Also, the planning is highly dependent on the assigned priorities to agents present in the system. This method may sometime lead to some agents not reaching the goal node as the necessary routes get blocked by the agents' paths with higher priorities.

As the goal node is reached the agents are assumed to be absent from the environment. Once these segments are generated for the path suggested for multiple agents, it becomes easier for the human supervisor to validate it. Between two-time steps, t1 and t2 vertices set (that is disjoint) is considered and segment is thus created. But it is crucial to determine the value of r which could be used where r denotes the number of segments required to represent the collision free solution.

Christianos et. al [10] have proposed selective parameter sharing strategy in order to incorporate more agents in the environment. The agents are represented using embedding based on their abilities and goals. After this some k clusters
are formed which depend on similarity between agents. Similar agents then share their learned parameters thus enabling goal-oriented learning.

Wan et. al [19] have applied conflict-based search withe concepts of outdated node and outdated constraints inorder to incorporate introduction of new agents in the mid-way of the solution. This enables the environment to allow inclusion of more agents and the dynamic path finding is handled based on current and older changes.

Stern et. al [22] provide a comprehensive review and have compared the existing solutions having different variations. They have also formally defined the conflicts that are taken into account while considering different possible solutions. The existing challenges of the non-stationarity of the environment is also addressed here and also suggestions are provided to overcome some of the existing challenges.

4. Proposed Model

In this section we first look at the working of each of the two solutions and then in next section we evaluate the two strategies based on two metrics. We have compared two different solution approaches, the conflict-based approach follows constraint tree at the higher level and traverses node as a graph in the lower level.

4.1. Conflict Based Search(CBS) for Single Shot MAPF

Conflict based search [14-16] is the strongest contender when it comes to solution to MAPF problem. It follows a tree-based approach where a condition tree is prepared, and nodes placed at high level contain conditions which are duly followed by lower-level nodes. This helps in maintaining the collision free from the source to goal. The condition or the constraint at a time t for an agent ai is denoted using tuple < ai, v, t> which means that ai cannot choose to move at vertex v at time t in order to meet the constraint. Node in constraint tree inherits constraints from its parent node along with new constraints.

The solution at a node is the sum of costs taken by each node individually. The figure 4.1. shows the procedure by which a constraint tree node is finally selected or rejected for the final path from source to the goal. The collision (both vertex and edge collision) is thus removed to traverse the path and reach the final goal. At each iteration, Conflict based search picks an unexplored node from the tree, based on some heuristic. Then, the conflicts (namely collisions) in the plan corresponding to that node are identified.

Conflict based search attempts to resolve the conflicts by creating child nodes based on the conflicts, as follows: if agents i and j collide at time t in vertex v, then two children are created for the node, one with the constraint that agent i cannot be in vertex v at time t, and the other dually for agent j.

Then, in each child node, a low-level search is used to re-plan a path for the newly constrained agent, given the set of constraints obtained thus far along the branch of the constraint tree. This process repeats until either a non-colliding plan is found, or no new nodes are created in the constraint tree, at which point conflict-based search returns that there is no solution.

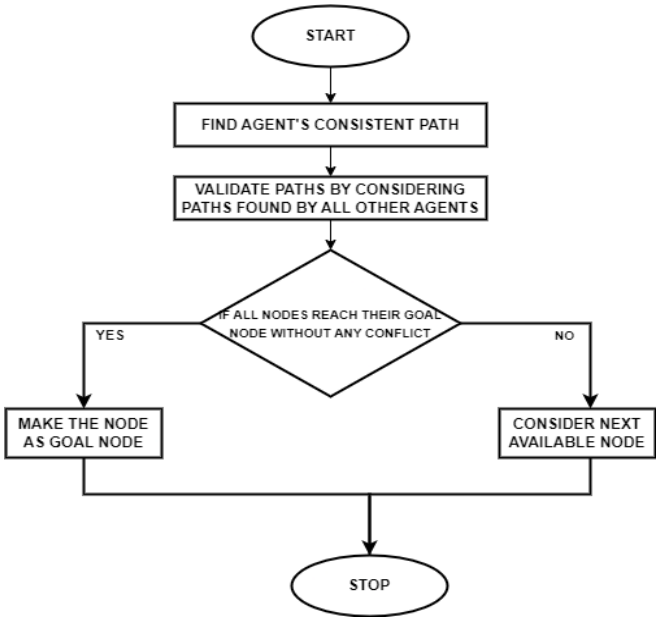

Figure 4. Flow chart depicting the process of including node in the solution

4.2. Shared Experience Action Critic (SEAC)

The shared Experience approach implies training of multiple agents together and then share the policies learned by individual agents to rest of the existing agents [18].

This technique enables agents without the same reward to learn more than one method of approaching and solving a problem.

The sharing of experience makes the agent more informed about the environment. The two components working together here are the actor that decides which action to be taken next while it is on some space s and the critic that tells the actor [18] about the action taken. How good was the choice made by actor?

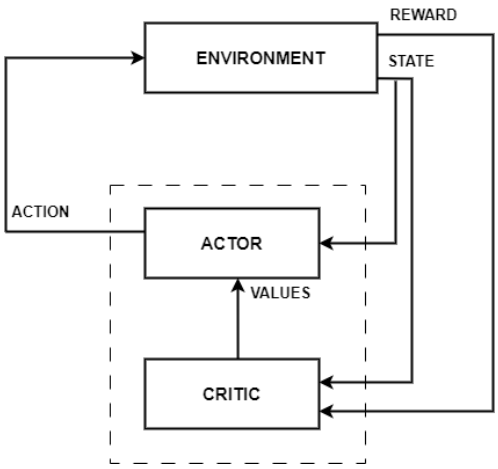

Figure 5. Action-Critic Architecture

Figure 4.2. shows an environment containing both the actor and the critic interacting with it and the state in which the environment will land is governed by what action the actor takes. Simultaneously there is a value function that enables the critic to provide feedback based on actor's move.

The figure 4.3 shows the environment containing two agents and a common goal. The first agent learns policy to reach the goal from the extreme left (bottom) and the second agent learns the policy to reach the goal from extreme right (top). Once they learn their own strategies, the learning is shared by each of them, and the two agents thus have two different ways to approach the goal node.

So even after not having same rewards, the learning is maximized. The same approach helps in the multi agent pickup and delivery problem by combining the paths learned by different agents while executing a requested pickup and drop.

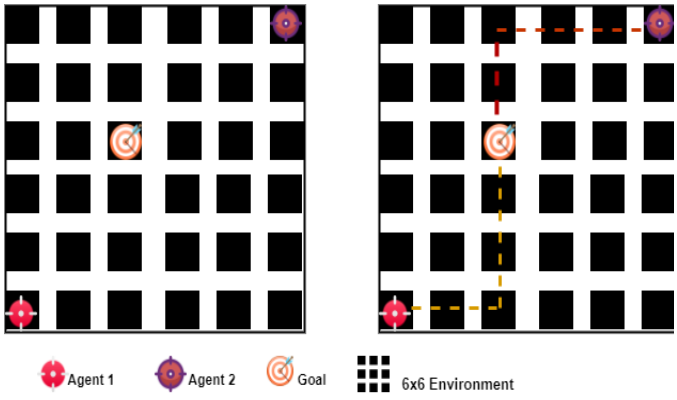

Figure 6. An instance before(left) and after(right) the model learns policy to reach to the goal node

Here the trajectory of an agent is considered for its on-policy while the trajectory of other agents is included as off-policy data. The loss calculation of the off-policy policy gradient optimisation with respect to the behavioural policy β is given as

$$\nabla\varphi L(\varphi) = (-\pi(at|ot;\varphi)/\beta(at|ot))\,\nabla\varphi \log \pi(at|ot;\varphi)(rt + \gamma V(ot+1;\theta) - V(ot;\theta)) \quad (1)$$

Where,
π = represents the on-policy,
β = behavioural policy or off-policy
at = action taken by agent of environment at time t,
ot = observation of environment (as visible to agent) at time t,
V = value that the critic passes to the action module.

5. Performance Evaluation

In this section we discuss the environment simulation used, the metrics for evaluation and the results obtained for both models.

5.1. Environment Simulation

A multi robot warehouse environment with some enhancements is used to train the agents efficiently. The changes are as under: The number of delivery locations are increased, and the request queue is replaced with new requests every time an item is picked up from the queue by the agent. Also, the reward is modified to give a plus one once it picks up an item successfully and is credited with a plus two on a successful delivery. The changed environment consists of 6 agents working in warehouse of large and medium sizes [7]. The agents perform the pickup and delivery task till an episode is in operation.

5.2. Metrics Used

The two approaches used in the study are based on multi agent path finding and multi agent reinforcement learning respectively and thus have their own way of evaluation. But here we need to compare them on same metrics to analyze the result and draw some conclusion. The different metrics to evaluate the two strategies are discussed in this section. The metrics used are as under:

Mean Flow Time

Number of steps required by all individual agents to reach to their first drop off location. It is the total time required for the assigned task done for the very first time by an agent. It is the total time which the agents take to get to the first goal location starting from their initial position.

Mean Makespan

The traditional MAPF the objective function is maximized, and the evaluation is done using this metric. Once we know the time steps required by every agent to

reach its destination, we pick the agent which took the maximum of all durations. This is called the mean makespan[6]. In a solution for multi agent path finding $\pi = \{\pi 1, \pi 2 \ldots \pi\}$, where the makespan is given by:

$$\textit{Mean Makespan} = \max_{1 \le i \le k} |\pi_i|$$

Here πi denotes the cost in time step that is taken by agent ai to deliver the item from its start position to the target node.

Mean Episodic Cumulative Reward

Taking the reward as plus one for moving towards goal and plus two for correct delivery, the mean episodic cumulative reward is the rewards earned by every node individually after completion of the delivery task. The episode is nothing but the interaction between agent and the environment from start till the goal is reached. Here both the conflict-based search and the shared experience action critic have been given same episode time for evaluation.

5.3. Results

For lifelong conflict-based search, it is observed that the mean flow time value and makespan value both are directly proportional to the environment size and the total agents present in the environment. Increasing the environment size makes the number of steps to increase. The other two metrics decrease with the environment size and the number of agents.
In case of shared experience action critic, it is found that the frequency of items delivered is higher than lifelong conflict-based search. It is also noted that the values of mean makespan and mean flow time show the same relation as in case of lifelong conflict-based search.

Table 2. Comparison of SEAC and Lifelong CBS on Two Metrics

Approach Used	Metrics		
	Agents	Mean Flow Time	Mean Makespan
Shared Experience Action Critic	5	137.32	65.43
	10	223.34	101.51
	15	127.07	44.56
Conflict Based Search	2	17.68	10.29
	5	29.10	9.30
	8	37.71	9.89

Comparing the two approaches, getting same time for every episode, lifelong conflict-based search has comparatively smaller mean flowtime and mean makespan, whereas having large number of items successfully reaching the goal in an episode by each agent on an average.

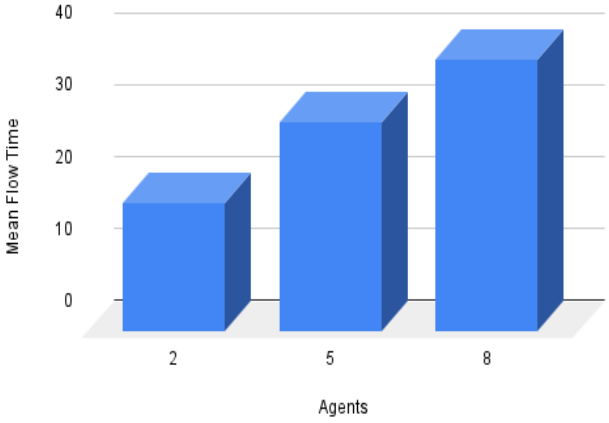

Figure 6. Results of Mean Flow Time for Conflict Based Search

The reason is that conflict-based search method plans the collision-free path for every agent from the start position to its goal position which should be more efficient than shared experience action critic which only gives the most probable action for each agent based on the current observation of the environment at every time step.
Figure 6. shows the mean flow value obtained for different number of agents present in the environment.

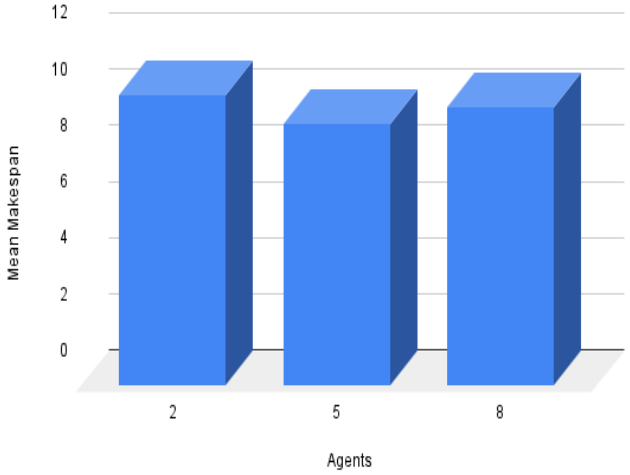

Figure 7. Results of Mean Makespan for Conflict Based Search

Figure 7. shows the mean makespan time obtained for different number of agents present in the environment for conflict-based search strategy.

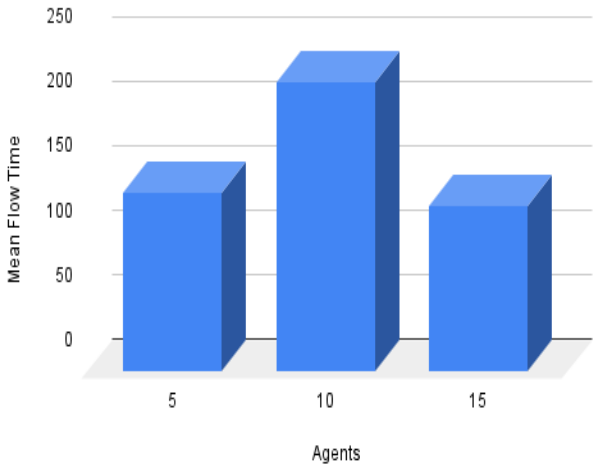

Figure 8. Results of Mean Flow Time for Shared Experience Action Critic

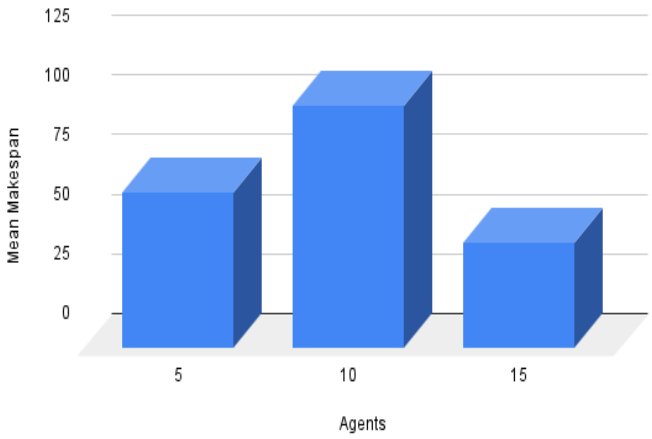

Figure 9. Results of Mean Makespan for Shared Experience Action Critic

Figure 8. and Figure 9. show mean flow time and makespan time taken by different number of agents in case of shared experience action critic approach.

This shows that the lifelong conflict-based search is better solver for the multi agent pickup and drop problem with lesser agents and when the environments are not much dense, whereas shared experience action critic should be used when agents are not small in number. Also, the training of shared experience action critic agents costs a good amount of time, which increases the number of agents.

6. Conclusion and Future Scope

The results show that ate lifelong Conflict based search is a better option when the agents are less in number as in that case, the re-planning will take overall less time but when the agents are large in number then this re-planning can take very long to produce conflict free paths from source to goal nodes. In this case, shared experience action critic can be more efficient choice as it takes the current environment state to give the most suitable action for that time t.

For this study the agents taken for learning are all similar type that can pick up and deliver any type of requested item i.e, the agents are homogeneous. We can address the same pickup and delivery problem when the agents are not all same and differ in their capabilities and the type of item they can handle. Also, there can be a role-based heterogeneity where a group of agents may perform both pickup and delivery while the others may perform only one of the two. To this end handling such agents is more complex and ways such as restricted sharing of learning helps the agents to work in the heterogeneous scenario.

References

[1] Frans A. Oliehoek, Matthijs T. J. Spaan, and Nikos Vlassis. “Optimal and Approximate Q-Value Functions for Decentralized POMDPs”. In: Journal of Artificial Intelligence Research 32 (2008), pp. 289–353.

[2] Sinno Jialin Pan and Qiang Yang. “A Survey on Transfer Learning”. In: IEEE Transactions on Knowledge and Data Engineering 22.10 (2009), pp. 1345–1359.

[3] Stefan Schaal. “Learning from Demonstration”. In: Advances in Neural Information Processing Systems.1997, pp. 1040–1046.

[4] Volodymyr Mnih, Adria P. Badia, Lehdi Mirza, Alex Graves, Tim Harley, Timothy P. Lillicrap, David Silver, and Koray Kavukcuoglu. “Asynchronous Methods for Deep Reinforcement Learning”. In: International Conference on Machine Learning. Vol. 4. 2016, pp. 2850–2869.

[5] Tonghan Wang, Heng Dong, Victor Lesser, and Chongjie Zhang. “ROMA: Multi-Agent Reinforcement Learning with Emergent Roles”. In: International Conference on Machine Learning. 2020.

[6] Kottinger, J., Almagor, S. and Lahijanian, M. (2022) ‘Conflict-based search for explainable multi-agent path finding’, Proceedings of the International Conference on Automated Planning and Scheduling, 32, pp. 692–700. doi:10.1609/icaps.v32i1.19859.

[7] Filippos Christianos, Lukas Schafer, and Stefano V. Albrecht. Shared experience actor-critic for multiagent reinforcement learning. In Advances in Neural Information Processing Systems (NeurIPS), 2020.

[8] Mehul Damani, Zhiyao Luo, Emerson Wenzel, and Guillaume Sartoretti. PRIMAL2: Pathfinding via reinforcement and imitation multi-agent learning - Lifelong. IEEE Robotics and AutomationLetters, 6(2):2666–2673, 2021.

[9] H. Ma, D. Harabor, P. Stuckey, J. Li, and S. Koenig. Searching with consistent prioritization for multi-agent path finding. In Proceedings of the AAAI Conference on Artificial Intelligence, pages 7643–7650, 2019.
[10] Filippos Christianos, Georgios Papoudakis, Arrasy Rahman, and Stefano V. Albrecht. Scaling multi agent reinforcement learning with selective parameter sharing. In Proceedings of the International Conference on Machine Learning (ICML), 2021.
[11] Georgios Papoudakis, Filippos Christianos, Lukas Schafer, and Stefano V. Albrecht. Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks. arXiv preprint arXiv:2006.07869, 2021.
[12] Oren Salzman and Roni Stern. Research challenges and opportunities in multi-agent path finding and multi-agent pickup and delivery problems. In Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2020.
[13] Guillaume Sartoretti, Justin Kerr, Yunfei Shi, Glenn Wagner, T.K. Satish Kumar, Sven Koenig, and Howie Choset. Primal: Path finding via reinforcement and imitation multi-agent learning. IEEE Robotics and Automation Letters, 4(3):2378–2385, 2019.
[14] G. Sharon, R. Stern, A. Felner, and N. Sturtevant. Conflict-based search for optimal multi-agent path finding. Artificial Intelligence, 219:40–66, 2015.
[15] G. Sharon, R. Stern, A. Felner, and N. Sturtevant. Conflict-based search for optimal multi-agent path finding. In Proceedings of the AAAI Conference on Artificial Intelligence, pages 563–569, 2012.
[16] Kottinger, J. 2021. Explanation-Guided Conflict-Based Search for Explainable MAPF. https://github.com/ariasystems-group/Explanation-Guided-CBS.
[17] Littman, M. L. Markov Games as a Framework for Multi Agent Reinforcement Learning. In Proceedings of the Eleventh International Conference on Machine Learning, 1994.
[18] Lowe, R., Wu, Y., Tamar, A., Harb, J., Pieter Abbeel, O., and Mordatch, I. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In Advances in Neural Information Processing Systems, volume 30, pp. 6379–6390, 2017.
[19] Wan, Q. et al. (2018) 'Lifelong multi-agent path finding in a dynamic environment', 2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV) doi:10.1109/icarcv.2018.8581181.
[20] R. Luna and K. E. Bekris, "Push and swap: Fast cooperative pathfinding with completeness guarantees," in IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011, 2011, pp. 294–300.
[21] C. Wei, K. V. Hindriks, and C. M. Jonker, "Altruistic coordination for multi-robot cooperative pathfinding," Appl. Intell., vol. 44, no. 2, pp. 269–281, 2016.
[22] Stern, R. et al. (2021) 'Multi-agent pathfinding: Definitions, variants, and benchmarks', Proceedings of the International Symposium on Combinatorial Search, 10(1), pp. 151–158. doi:10.1609/socs.v10i1.18510..