Decentralized multi-agent path finding framework and strategies based on automated negotiation

Keskin, M. Onur; Cantürk, Furkan; Eran, Cihan; Aydoğan, Reyhan

DOI
10.1007/s10458-024-09639-8

Publication date
2024

Document Version
Final published version

Published in
Autonomous Agents and Multi-Agent Systems

Autonomous Agents and Multi-Agent Systems (2024) 38:10
https://doi.org/10.1007/s10458-024-09639-8

Decentralized multi-agent path finding framework and strategies based on automated negotiation

M. Onur Keskin[1] · Furkan Cantürk[1] · Cihan Eran[1] · Reyhan Aydoğan[1,2]

Accepted: 8 February 2024

Abstract
This paper introduces a negotiation framework to solve the Multi-Agent Path Finding (MAPF) Problem for self-interested agents in a decentralized fashion. The framework aims to achieve a good trade-off between the privacy of the agents and the effectiveness of solutions. Accordingly, a token-based bilateral negotiation protocol and two negotiation strategies are presented. The experimental results over four different settings of the MAPF problem show that the proposed approach could find conflict-free path solutions albeit sub-optimally, especially when the search space is large and high-density. In contrast, Explicit Estimation Conflict-Based Search (EECBS) struggles to find optimal solutions. Besides, deploying a sophisticated negotiation strategy that utilizes information about local density for generating alternative paths can yield remarkably better solution performance in this negotiation framework.

Keywords Multi-agent path finding · Negotiation · Decentralized coordination

1 Introduction

Technological advancements in the last decades have enabled autonomous robots and vehicles to carry out various tasks, such as surveillance and transportation. They may need to navigate from one location to another to achieve their goal. Imagine an environment where hundreds of autonomous robots aim to reach specific locations. Such an environment requires a coordination mechanism to avoid some potential collisions. This problem,

✉ Reyhan Aydoğan
reyhan.aydogan@ozyegin.edu.tr; R.Aydogan@tudelft.nl

M. Onur Keskin
onur.keskin@ozu.edu.tr

Furkan Cantürk
furkan.canturk@ozu.edu.tr

Cihan Eran
cihan.eran@ozu.edu.tr

[1] Computer Science, Özyeğin University, Istanbul, Turkey

[2] Interactive Intelligence Group, Delft University of Technology, Delft, The Netherlands

Published online: 13 March 2024

allocating conflict-free paths to agents to navigate safely in an environment, is well-addressed in Multi-Agent Systems and known as the Multi-Agent Path Finding (MAPF) problem [27, 30]. Many studies tackle this problem; some propose centralized solutions while others focus on decentralized solutions [20, 29].

Centralized solutions rely on full access to all relevant information regarding the agents and properties of the given environment so that a global solution can be derived. In contrast, decentralized solutions decouple the problem into local chunks and address the conflicts locally [12]. Without any time constraints, centralized approaches can find optimal solutions if they exist. However, the performance of centralized solution approaches can suffer in high-density and complex environments [10, 22, 28]. Besides, complete information may not always be available due to communication, sensor, or privacy limitations. Decentralized approaches can deal with uncertainty and scalability issues and produce admissible solutions. However, they may overlook a potential optimal solution and end up with suboptimal solutions.

Accordingly, this paper introduces a negotiation-based decentralized conflict resolution approach to MAPF in which agents autonomously negotiate with each other to refine their paths to avoid possible collisions. One of the main challenges is dealing with the uncertainty about the environment due to the limited capacity of the sensors, communication, or some privacy concerns. Most real-world applications are partially observable, where the agents can perceive some relevant aspects of the environment. For instance, a robot may not perceive all objects far from its current location. Light detection and range-finding sensors can detect up to a certain distance. Similarly, wireless communication systems also have limited communication capabilities. Agents can only exchange information with each other if they are within a communication range. Besides, complete information is not always available due to the characteristics of the environment. For example, drivers in traffic do not know where other drivers are going. Furthermore, agents may be reluctant to share all information due to their privacy. For instance, they may not be willing to reveal their destination.

In the proposed approach, agents share their partial path information with only relevant agents close to them. If they detect any conflict on their partial path, they start a bilateral negotiation to allocate required locations for specific time steps. This study introduces a variant of alternating offers protocol enriched with token exchanges to govern this negotiation. By enforcing the usage of tokens, the protocol leads agents to act collaboratively and search for alternative optimal or suboptimal paths to avoid conflict. Two negotiation strategies are presented in line with the protocol: Path-Aware and Heatmap. While the Path-Aware negotiation strategy focuses on the agent's information on its paths and makes bids considering the length of its final path, the Heatmap strategy utilizes other agents' available path information in its field of view to avoid potential conflicts by making bids.

There are a few attempts to solve the MAPF problem considering a negotiation-based approach [15, 16, 23, 24]. Either they require sharing complete path information with others, considering one-step decisions such as who will move in a certain direction at the time of conflict, or aiming to resolve the conflict in one shot (i.e., collaborate or reject). The following sections also provide a detailed analysis of the proposed approach. In contrast, our approach aims to reduce the complexity of the problem by resolving the conflicts in sub-path plans iteratively instead of the entire path plans, thereby respecting the agents' privacy to some extent.

We evaluate the proposed approach in various settings, considering agents' ability to wait or disappear at their final destinations. We also examine the impact of three types of commitments on the agreed paths regarding the success rate and solution quality.

Furthermore, we define a metric called *information sharing rates* to discuss the trade-off between privacy and solution performance for MAPF problems. Our experimental results show that our decentralized approaches can provide higher success rates in high-density scenarios (e.g., when the number of agents is 60 or 80) compared to the state-of-the-art centralized approach, namely EECBS. Although its variant, called EECBS with 0.1 suboptimality, could achieve a better solution rate and quality than the proposed approaches, our approach can achieve a better trade-off between privacy and performance. It is worth mentioning that centralized approaches entirely ignore the agents' privacy, whereas our approach prioritizes agents' privacy. Besides, we compared the performance of the proposed negotiation strategies empirically. Based on our experimental results, the Heatmap agent performs better than the Path-Aware agent in terms of success rate and information sharing rate across all settings. To sum up, this paper extends our previous study [11] by the following contributions:

- Introducing a novel negotiation strategy called Heatmap, which generates offers by avoiding dense areas to resolve the conflicts in MAPF problems,
- Presenting different commitment types and analyzing the performance of the proposed approaches in terms of various metrics,
- Studying the effect of field of view size and different problem settings where agents can take a wait action and/or disappear when they reach their destinations and
- Investigating the trade-off between success rate and privacy empirically by examining information sharing rate and success rate of centralized and decentralized solution approaches.

The rest of the paper is organized as follows. Section 2 presents the related work on decentralized approaches for MAPF problems. Section 3 describes the problem addressed in this paper, while Sect. 4 lays out the proposed negotiation approach. Section 5 presents our experimental setup and results. Finally, this paper's main contributions and planned future work are discussed in Sect. 6. The code repository of the paper is accessible at https://github.com/erancihan/NegotiationForMAPF.

2 Related work

Conflict resolution approaches for MAPF problems can be classified according to the centralization of solution mechanisms and cooperation of agents. Centralized solution approaches path-finding of cooperative agents provide optimal plans [13]. Suppose a trusted center with the information of all agents moving in a known environment or the ability to operate all of them is unavailable. In that case, negotiation can be used for a conflict resolution mechanism [1, 14, 15, 23, 24, 31]. One negotiation approach to allocating resources to multiple parties is Combinatorial Auction (CA). To resolve conflicts between self-interested agents, Amir et al. reduces the MAPF problem to CA and implements IBundle [21], an iterative CA algorithm [1]. One crucial factor for the effectiveness of CA algorithms is the trustworthiness of self-interested agents about their utilization. Considering this issue, Amir et al. propose a Vickrey-Clarke-Groves (VCG) auction for MAPF, a strategy-proof auction mechanism for manipulation attempts by the agents [1]. In this IBundle auction, the auctioneer is exposed to a computational burden as agents submit their all-desirable bundles, which requires impractical auction time. Addressing this limitation of IBundle, the auction mechanism proposed by Gautier et al. can provide a feasible allocation more likely by allowing agents to submit a limited number of bundles, and so terminates in less time [14]. Compared to these CA-based conflict resolutions for MAPF, our solution is free of trustworthiness and computational load concerns as it applies bilateral negotiation to resolve each conflict.

Critical challenges of addressing the MAPF problem within the decentralized method can be summarized as establishing a solution framework for agents to use while interacting with the environment, defining an interaction protocol between agents, and designing agents that can reach a solution. Purwin et al. propose a decentralized framework where cooperative agents allocate portions of the environment in which they can move [24]. Using these frameworks, agents send information to others via a network and keep information about all. Here, agents who collide update their trajectories with a pairwise agreement based on a priority rule. Similarly, agents broadcast information in our framework while seeking a conflict-free path, resolving conflicts bilaterally. However, the information revealed here is local and only achievable by agents in the broadcast range. The work of Pritchett et al. defines a bilateral negotiation structure to prevent collisions in air traffic [23]. Each round of the negotiation session offers cost increases until an agreement is reached. This offering mechanism forces agents to concede over time. In contrast, the Token-based Alternating Offers Protocol (TAOP) we use in our study does not expose a catalyst to reduce the number of offers. It is left open to be addressed by agent strategy designs. Another bilateral negotiation approach for air traffic is proposed for the pre-flight planning of unmanned air vehicles (UAVs) [15]. In this approach, agents bargain to undertake the cost of replanning to fix some of the path conflicts with counterparts in the negotiation. While this approach resolves path conflicts in pre-flight planning among all UAVs, our study addresses the real-time resolution of the MAPF problem. Besides, our negotiation approach depends on revealing sub-paths in the bilateral negotiations to resolve local conflicts so that the privacy of the agents' paths can be preserved better. Conversely, full-path information is revealed in the negotiation schema proposed by Ho et al..

Unlike bilateral conflict resolution approaches, Sujit et al. focuses on solving a task allocation problem in their work, using a multilateral negotiation structure [31]. Besides, in their work, agents utilize the presented token to determine whose offer to accept in a deadlock situation that might happen, in which the agent with the least number of tokens is selected. Whereas in our framework, token usage is dedicated to penalizing repeated offers. Inotsume et al. demonstrates a bilateral negotiation-based approach to MAPF [16]. In this approach, agents submit their desired paths to an area manager before moving their initial positions. The area manager is responsible for keeping control of a path reservation table and responding to path requests by agents based on this table to prevent any intersecting paths. The requests are accepted on a first-come, first-serve basis. Agents can request already reserved areas from corresponding agents by offering some tokens. Using a path reservation system deviates from the decentralized MAPF approach. In our framework, agents aim to minimize path costs and maximize tokens. In contrast, our solution uses tokens to manage the negotiation behavior of any agent designed to reach an agreement.

The MAPF problem requires all of the agents to reach their destination. Therefore, they must behave in a way that complies with the social welfare to some extent because allowing agents to act selfishly usually leads to a Nash Equilibrium [26]. To motivate self-interested agents in a network to behave in a socially desirable way, taxation is a classical approach to diminish congestion in the network [8]. Considering the MAPF problem for self-interested agents, Bnaya et al. propose an Iterative Taxing Framework to detect where, when, and how much additional cost is imposed on agents to change their paths so that they are discouraged from using paths that would collide with others [5]. Their results show a substantial total cost decrease compared to a solution scheme depending on a selfish replanning procedure. However, they do not benchmark with optimal solution cost. Thus, the efficiency of the proposed framework needs to be investigated further. Aiming for system efficiency, Ramos et al. introduce a toll-based multi-agent reinforcement learning approach to minimize total congestion and can achieve system-efficient results [9]. In this study, tolls are used to divert drivers from congested routes. Similar to these two studies, our proposed system for MAPF aims to urge agents to prefer alternative optimal paths or individual sub-optimal paths when they face a density of other agents in their next sub-paths requiring more tokens to resolve conflicts.

Besides providing a scalable multi-agent planning framework for path-finding problems among self-interested agents, our study also addresses the privacy aspect in the multi-agent planning domain. In the privacy-concerned planning concept, agents have the task of searching and are enforced to preserve the disclosure of sensitive data while resolving conflicts of interest. MA-STRIPS [7] model provides a privacy-preserving planning scheme to control what part of the information is shared with all agents. However, this planning paradigm reveals information among all agents instead of transmitting data only between the agents subject to a conflict of interest. Bonisoli et al. provides a generalized form of MA-STRIPS by establishing pairwise privacy among agents [6]. In their setting, agents can specify a subset of agents intended to share information. Their approach is well proper for multi-agent problems, including individual interest protection. Similarly, in our situation, agents need to collaborate with other agents in case of future conflicts, although they are all self-interested. Therefore, path-seeker agents have to reveal local information. Our decentralized MAPF framework follows a geographical manner to condition what part of the information is shared with which agents. Such local information revealed with close agents is needed because the addressed problem requires no collision and proactive responses (i.e., replanning) by the stakeholder agents in a reasonable time before potential path conflicts.

3 Problem definition

MAPF is the problem of assigning conflict-free paths to agents from their respective starting locations to their destinations [30]. There are k agents $A = \{ A_1, A_2, ..., A_k \}$ navigating in a grid $G = (V, E)$ where the starting and destination locations for each agent A_i are denoted by $s_i \in V$ and $g_i \in V$, respectively. A path is a sequence of states $\pi_i = [(s_i, 0), \cdots, (g_i, T)]$, where a state is a tuple of vertex $v \in V$ and time step $t \in \{0, \cdots, T\}$, and T is the time step when A_i reaches g_i. The cost of a path is calculated as $|\pi| - 1$. At any time step, agents cannot be located in the same vertex – $\pi_i^t \neq \pi_j^t \; \forall \; i \neq j$ or traverse the same edge $\pi_i^t = \pi_j^{t+1} \wedge \pi_i^{t+1} = \pi_j^t, \forall i \neq j$ (i.e., swapping conflict), where π_i^t is the location of A_i at t.

Agents are located in a $M \times M$ grid-like environment where each cell corresponds to a vertex as illustrated in Fig. 1a. Initially, each agent has an optimal path to reach their destination, shown by dashed lines in the grid. There are three agents, A, B, and C, whose planned paths are colored in their respective colors (i.e., red, blue, and green). All agents can only move to their vertical and horizontal adjacent neighboring cells (i.e., cardinal directions) or wait. The example includes a conflict between agent A and C at $t = 2$ in cell (3, 3). They need to resolve this conflict to achieve their goals. In the classical MAPF problems, the system aims to minimize the sum of individual costs $\sum_{i=1}^{k}(|\pi_i| - 1)$ where the

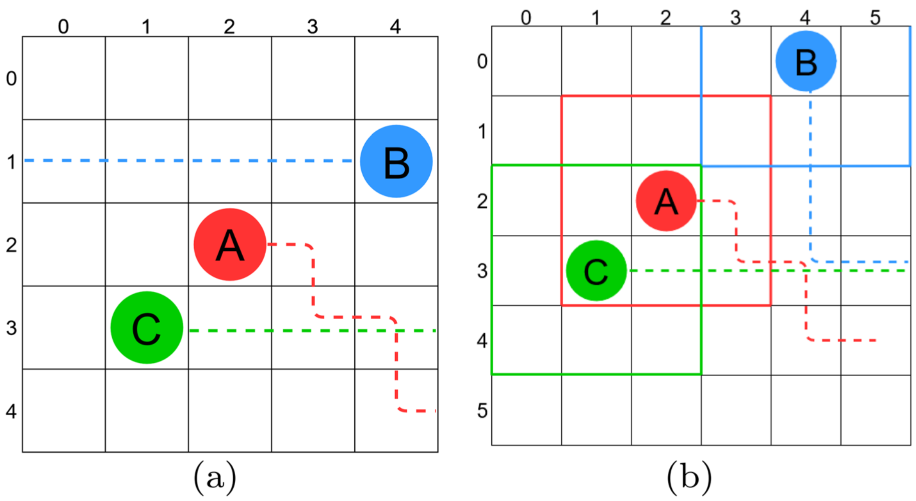


Fig. 1 Example environment and field of view size = 3

individual cost of each agent i corresponds to their path length denoted by $|\pi_i|$. Note that each individual aims to minimize their cost in decentralized MAPF problems.

This work examines four different MAPF problem settings. In *disappear-at-target* MAPF problem, agents can disappear from the environment when they reach their destinations. In contrast, agents stay at the destination in *stay-at-target* MAPF problem (i.e., $\pi_i^t \neq \pi_i^{t+1}$ whereas $\pi_i^t \neq g_i$.). In another problem setting, agents can either wait or move. Accordingly, this work studies four different variants of the MAPF problem as follows:

- Setting-1: Agents cannot wait until reaching their destinations. They stay at their target destinations and act as obstacles to other agents.
- Setting-2: Agents can wait at any time step until reaching their destinations. They stay at their target destinations and act as an obstacle for other agents.
- Setting-3: Agents cannot wait at any time step and disappear at their target destinations.
- Setting-4: Agents can wait at any time step until reaching their destination and disappear at their target destinations.

4 Decentralized MAPF framework based on automated negotiation

In the proposed framework, agents are located in a grid as shown in Fig. 1b. Initially, each agent knows only their starting location, destination, and a path plan to reach their destination. Those planned paths are shown as the colored dashed lines in the grid for each agent. We adopt the concept of *field of view* (FoV) to simulate the partially observable environment mentioned above [11]. The framework enables agents to access a limited portion of other agents' planned paths within a certain proximity and share their own. In other words, an agent's FoV determines the scope of its communication and perception capacity. An agent can only observe and communicate with other agents within its FoV. That is described as a square at size $2d + 1$, where d is the distance of agent location from the view boundary. For instance, when d equals 1, a red rectangle for agent A shows the

FoV boundary. In such a case, agent A can receive/send information from/to only agent C located in its FoV and vice versa. However, in the given snapshot, agent B cannot communicate with other agents at that time.

In this framework, each agent broadcasts its planned subpath. Agents can determine to what extent the path will be shared with other agents in FoV. In our experiments, agents share their current subpath with a length of $2d$. If any conflict is detected by one of the agents, they can engage in a negotiation session. For example, when d equals 1, agents will share their current subpath with a length of 2 (i.e., its next two moves) with the agents located in the scope of their FoV. agent A broadcasts its current subpath as $Broadcast : [\pi_A^{t=1} = (3,2), \pi_A^{t=2} = (3,3)]$ while agent C shared its own as $Broadcast : [\pi_C^{t=1} = (2,3), \pi_C^{t=2} = (3,3)]$. Since agents would detect a conflict in the vertex (3, 3) at t=2, agent A and agent C start negotiating the allocation of vertices on their path.

When an agent detects a conflict with more than one agent, which negotiation to be held first is determined on *first come, first serve* fashion. If a conflict with multiple agents is detected simultaneously, the agent to negotiate with is selected arbitrarily. For example, if d is 2, agent B and agent C will share their subpaths with a length of 4 with agent A. Agent A may first negotiate with agent B if agent B's message has been received before agent C. Afterward, it can encounter a bilateral negotiation with agent C. After carrying out any successful negotiation, agents will update their path accordingly. Several negotiation sessions might be held until current conflicts are resolved. If agents do not detect a conflict in their FoV, they move to the next location in their path. Once an agent reaches its target destination, it will no longer negotiate. The negotiation is carried out according to the proposed token-based negotiation protocol. The details of this protocol and the proposed bidding strategies mainly designed for this protocol to tackle the MAPF problem are explained in the following sections.

The proposed framework requires agents to negotiate to resolve conflicts in their paths. The conflict may occur between two or multiple agents at a given time. When it happens among more than two agents, we can formulate it as multiple bilateral negotiations. As it may be harder to find a joint agreement, especially when the number of participants is high [2], the proposed approach aims to resolve the conflicts in multiple consecutive bilateral negotiations. For simplicity, agents perform their bilateral negotiations consecutively. That is, a new negotiation can start after completing the previous one.

In the proposed approach, when there is a conflict in two agents' sub-paths, agents negotiate to allocate the conflicted states for certain time steps. According to the example illustrated in Fig. 1b, agent A may claim to use the vertices (3, 2) at $t = 1$ and (3, 3) at $t = 2$ while agent C may claim to use the vertices (2, 3) at $t = 1$ and (3, 3) at $t = 2$. Depending on how the negotiation proceeds, they may concede over time and change their request on vertex allocations to reach an agreement. If agents reach an agreement, they are supposed to obey the allocation for the other party. Agents are free to change their path as long as their current path allocation does not violate the agreed vertex allocation for the other party. For example, when agent C accepts agent A's vertex allocation for time steps $t = 1$ and $t = 2$, agent C confirms that it will not occupy those vertices allocated by agent A for the agreed time steps.

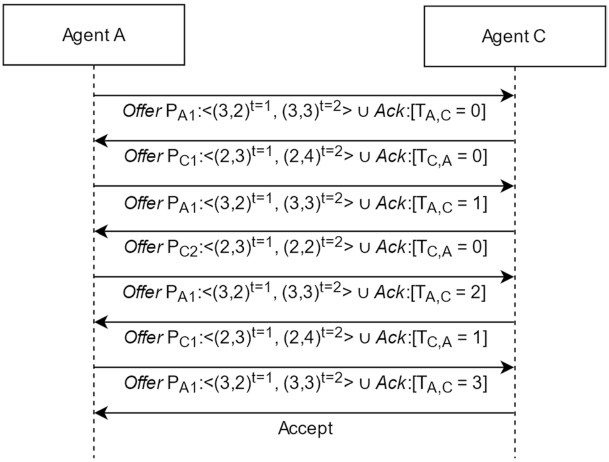
The interaction between agents needs to be governed by a negotiation protocol. In automated negotiation, agents mostly follow the Stacked Alternating Offers Protocol [3], exchanging offers in a turn-taking fashion until reaching a predefined deadline. This protocol does not force the agents to come up with an agreement. If both agents are selfish, they may fail the negotiation. However, having a consensus is necessary to find conflict-free paths. Therefore, agents preferably follow a protocol leading them to reach an agreement. Accordingly, we introduce a token-based negotiation protocol, namely *Token-based Alternating Offers Protocol* (TAOP) inspired by Monotonic Concession Protocol (MCP) [25] and Unmediated Single Text Protocol (USTP) [17].

According to the Monotonic Concession Protocol, agents make simultaneous offers to either stick to their previous offer or make a concession. The negotiation ends without consensus (i.e., failure) if both parties stick to their previous offers. Otherwise, agents continue negotiation until reaching an agreement or fail the negotiation. This protocol leads agents to complete the negotiation without setting a predefined deadline. However, there is a risk of ending up with failure. In the Unmediated Single Text Protocol, agents interchangeably become proposers or voters during the negotiation. Initially, some tokens are given to each agent, and agents can use those tokens to override others' rejected votes. One agent starts with a random offer, and the other agent votes to accept or reject it. It is considered the most recently accepted bid if the other agent accepts. This interaction is repeated multiple times, and the most recently accepted bid is updated over time. At the end of the negotiation, the most recently accepted bid is considered the agreement. Here, the tokens incentivize truthful voting of agents to not manipulate the system by rejecting all offers. Since this protocol is particularly designed for large-scale negotiation problems, the generated bids are variants of an initial random offer, not directly applicable to our problem. On the other hand, the token idea can force the agents to concede over time in a fairway.

The proposed token-based alternating offers protocol is a variant of alternating offers protocol enriched with token exchanges. One of the agents initiates the negotiation with an offer. The receiving party can accept this offer, make a counteroffer, or end the negotiation without agreement. The main difference is that agents can only repeat their previous offers if they pay for them. The protocol assumes that each agent owns a predefined number of tokens, $\mathcal{T}$. Those tokens enable an agent to make one of its previous offers during that negotiation. Unlike the Monotonic Concession Protocol, the negotiation does not fail if both agents stick to their current offer by exchanging tokens. The essential requirement for agents is to make offers that are not previously proposed during the negotiation or pay tokens to repeat an offer. In addition to the given offer, agents send an acknowledgment message specifying the number of tokens to be used to repeat an offer previously made by the same agent. The general flow of the proposed protocol is given below:

1. One of the agents makes an offer specifying its request to allocate some vertices for specific time steps and sends an acknowledgment message regarding using its token in the current negotiation. Initially, the usage of tokens is set to zero.
2. The receiving agent can take one of the following actions:
   - ends the negotiation without any consensus.
   - accepts the received offer and completes the negotiation successfully.
   - makes an offer specifying the vertices allocation for itself that has not been offered by that agent yet and sends the acknowledgment denoting the accumulated usage of its tokens.
   - can repeat one of its previous offers, increase the usage of its tokens by one, and send the token acknowledgment message.
3. If the agent accepts or ends the negotiation, the negotiation is finished. The accepting agent receives tokens amounting to the calculated token usage difference from its oppo-

Fig. 2 Example interaction between negotiating agents


Agent A
Agent C
Offer P_A1:<(3,2)^t=1, (3,3)^t=2> ∪ Ack:[T_A,C = 0]
Offer P_C1:<(2,3)^t=1, (2,4)^t=2> ∪ Ack:[T_C,A = 0]
Offer P_A1:<(3,2)^t=1, (3,3)^t=2> ∪ Ack:[T_A,C = 1]
Offer P_C2:<(2,3)^t=1, (2,2)^t=2> ∪ Ack:[T_C,A = 0]
Offer P_A1:<(3,2)^t=1, (3,3)^t=2> ∪ Ack:[T_A,C = 2]
Offer P_C1:<(2,3)^t=1, (2,4)^t=2> ∪ Ack:[T_C,A = 1]
Offer P_A1:<(3,2)^t=1, (3,3)^t=2> ∪ Ack:[T_A,C = 3]
Accept

nent, $min(\mathcal{T}_{opp} - \mathcal{T}_{self}, 0)$ where $\mathcal{T}_{opp}$ and $\mathcal{T}_{self}$ denote the total number of tokens used by the opponent and the accepting agent during the entire negotiation respectively. If the accepting agents spend more tokens than their opponent, it does not receive any tokens. Otherwise, the receiving agent can take any action mentioned in Step 2.

Considering the scenario in Fig. 1b, an example negotiation trace between agent A and C is illustrated in Fig. 2. Agent A initiates the negotiation with its offer P_{A1} requesting to claim the vertices (3, 2) for $t = 1$ and (3, 3) for $t = 2$. Agent C does not accept this offer and makes its offer specifying its allocation, such as (2, 3) and (2, 4). Since agent A insists on its previous offer, it increases its token usage by one. As the example shows, agents send an acknowledgment message and offer each turn. Agent C accepts agent A's offer in the fourth round. It confirms that agent C will not move to $(3, 2)^{t=1}$ and $(3, 3)^{t=2}$. In return, agent A will pay 2 tokens ($\mathcal{T}_A$-$\mathcal{T}_C$). It is worth noting that the token exchange is performed at the end of the negotiation, depending on who accepts the offer. The agreement is not committed if an agent needs to pay tokens but has insufficient tokens (i.e., negotiation fails).

4.1 Commitments of agreements

After reaching an agreement by following the proposed negotiation protocol above, agents may adhere to or retract from the agreed path allocation depending on their role. If the agent pays a token and convinces another party to accept its offer, it can change its remaining path later. Otherwise, the agent accepting the offer should determine another subpath that does not conflict with the agreed allocation (i.e., the tuple of opponent agent ID, timestep, and location) and broadcast its new subpath within FoV. However, that agent may end up with suboptimal or unsolvable positions at any time step after the agreement time t'. For such situations, we define different commitment types for agents as follows:

- *Standard commitment (SC)* The agent agreed on the offer and cannot enter the subpath allocated to the other agent according to the agreement [11].
- *Zero commitment (ZC)* At any time step after t', the agent can decommit the current agreement when it observes new conflicts.

- *Dynamic commitment (DC)* At the end of the negotiation, the agent who accepted the opponent's offer should obey the agreed subpath until the conflicted state. Afterward, it can break the commitment if deemed necessary as a result of new observations at any moment after t'.

To illustrate how these commitment types work, consider the example in Fig. 1b, where the FoV size is three. Assume that agent C gives the path advantage – (3,3) location in the grid is allocated to agent A at the end of the negotiation at time step t.

- While ZC is utilized, agent C can again move ahead of the agent A path in $t + 1$ if necessary.
- While SC is utilized when agent C detects two different conflicts with agent A and B and agrees to give both complete path superiority, agent C cannot enter the paths where they are placed behind the conflicted positions. Since agent C already gives all usage rights of the paths in t, even if C needs to break its commitment to reach its destination.
- While DC is utilized, agent C cannot go from agent A's path until it passes a conflicted point (e.g., (3, 3)); it has to stick two steps to its commitment until $t + 2$. However, there is future negotiation with agent B at $t + s$ for a conflicted location at $t + s + 1$. In that case, agent C should stick only one step to its commitment to agent B. Agent C has a different commitment deadline for each negotiation according to its conflicted places in dynamic commitment.

For these effects, the commitment type can be a distinguishing feature for decentralized MAPF problems, especially in scenarios where the FoV is large in dense environments. Hence, this paper analyzed the effects of the commitment type as an independent variable. Due to the nature of different commitment types, agents can increase uncertain behaviors in the environment according to the changes in their observations. These behaviors cause to increase in the number of negotiations for each scenario. In other respects, these behaviors can increase agents' individual utility and the successful completion rate of each scenario since they will increase agents' resiliency and adaptation skills. Section 5.4 will further examine the impact of these commitment differences across all other configuration variables.

4.2 Proposed negotiation strategies

We introduce two bidding strategies, namely *Path-Aware* [11] and *Heatmap*. Note that the agents are called by the name of the bidding strategy they adopt. While the Path-Aware agent considers its available path and token information, the Heatmap agent aims to reduce potential conflicts and prefers low-density routes, as explained below.

4.2.1 Path-Aware negotiation strategy

Existing negotiation strategies focus on only which offer to make at a given time and when to accept a given offer [4]. Therefore, there is a need to design a new strategy considering token exchanges. Hereby, we propose a negotiation strategy called Path-Aware agent [11], determining when to repeat an offer or generate a new one. It aims to utilize the available information to decide when to insist on its current path. It is worth noting that each agent generates possible paths leading them to its destination by using the A^* Algorithm by considering the non-conflicting paths with those of other agents in its FoV. Those possible paths constitute the agent's bid space. Afterward, they sort those paths in descending order of their cost.

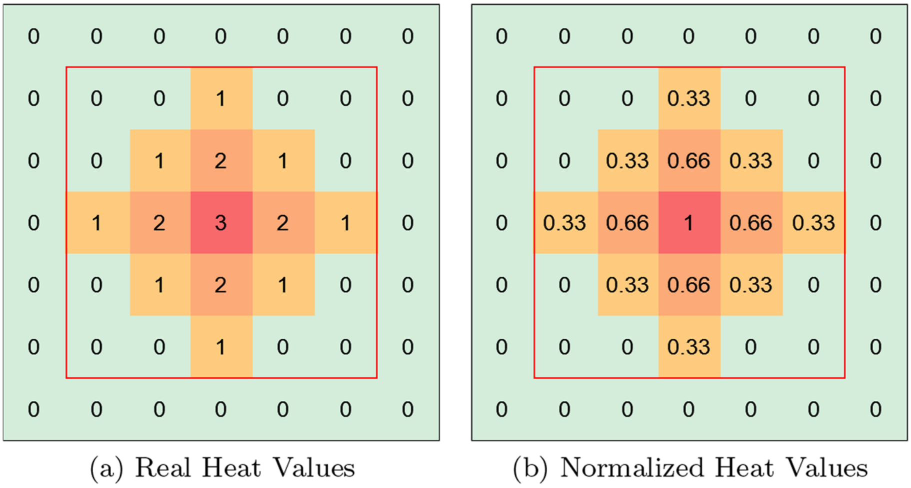
Algorithm 1 describes how an agent negotiates according to the Path-Aware agent. At the beginning of the negotiation, the current path in FoV (P_c) is the relevant part of the optimal path (i.e., the shortest path to its destination). It corresponds to the first offer in the negotiation. When the agent receives an offer from its opponents, it checks whether it is possible to generate a path of equal length or shorter than its current path to the destination (Line 1). If so, it accepts its opponent's offer (Line 2). Note that the path generation function takes the opponent's offer, O_{opp}, as a constraint while generating its best possible path to the destination. If the best possible route, in line with the opponent's bid, is better than or equal to the current path, then the agent accepts the opponent's offer. If the agent's remaining tokens are greater than the length of the remaining path to the destination (Line 4), it decides to repeat its previous offer. It updates its remaining tokens accordingly (Line 5). Recall that for each repetition, the agent needs to use one token. Otherwise, it concedes and sets the next possible best path from the sorted path space P_{Space} as its current path in its FoV (Line 7). Accordingly, the agent offers its current path in FoV P_c (Line 9). Note that agents must concede if they do not have any tokens left.

Algorithm 1 Negotiation strategy of Path-Aware agent

```
Data:
$Q_r, P_r$ : Agent's remaining tokens & path to destination
$P_c$ : Current path in FoV
$P_{Space}$: Sorted path space
$O_{opp}$ : Opponent offer
1  if $|P_c| \geq |generatePath(O_{opp})|$ then
2      accept()
3  else
4      if $Q_r > |P_r|$ then
5          $Q_r --$
6      else
7          $P_c \leftarrow P_{Space}$.next()
8      end
9      offer($P_c$)
10 end
```

4.2.2 Heatmap negotiation strategy

The Path-Aware agent focuses on the agent's remaining path length and the number of tokens. The density in the observable environment is neglected during the bidding process. Thus, the agreed paths may lead to further conflicts. Heatmap bidding strategy incorporates this environmental information into the evaluation of bids. This strategy uses the available data in FoV to generate a heatmap around the agent's terrain and detect high-density areas. Consequently, it avoids generating a path through dense regions as the cars avoid potential traffic jams by preferring the low-density routes in the traffic. In other words, by analyzing the received path information from other agents, the agent aims to foresee moving to which region of the environment causes path conflicts to be less likely. The generated heatmap is built based on the broadcasted information of other agents within FoV. It considers each heatmap while calculating the estimated cost of any path. Figure 3 shows an example of



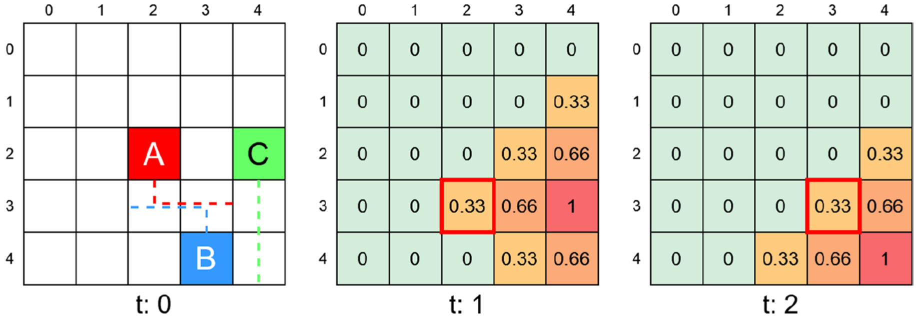
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	0	1	2	1	0	0
0	1	2	3	2	1	0
0	0	1	2	1	0	0
0	0	0	1	0	0	0
0	0	0	0	0	0	0

(a) Real Heat Values

0	0	0	0	0	0	0
0	0	0	0.33	0	0	0
0	0	0.33	0.66	0.33	0	0
0	0.33	0.66	1	0.66	0.33	0
0	0	0.33	0.66	0.33	0	0
0	0	0	0.33	0	0	0
0	0	0	0	0	0	0

(b) Normalized Heat Values

Fig. 3 Heatmap generated by an agent



	0	1	2	3	4
0					
1					
2			A		C
3					
4				B	

t: 0

	0	1	2	3	4
0	0	0	0	0	0
1	0	0	0	0	0.33
2	0	0	0	0.33	0.66
3	0	0	0.33	0.66	1
4	0	0	0	0.33	0.66

t: 1

	0	1	2	3	4
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0.33
3	0	0	0	0.33	0.66
4	0	0	0.33	0.66	1

t: 2

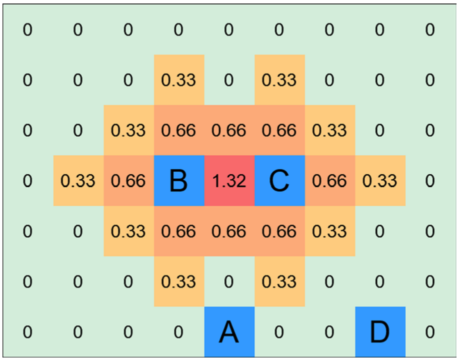
Fig. 4 Heatmap of agent C according to agent A

a heatmap created for an agent in the middle of the grid. When other agents detect the location information of an agent, they compute a heatmap centered at the location of the detected agent, as seen in Fig. 3a. Here, the values are normalized concerning FoV (see Fig. 3b).

To better understand how the agent uses a heatmap in its bidding strategy, we illustrate a toy example in Fig. 4. In this example, agent *A* and agent *B* start negotiation because they detect a swapping conflict on their paths. Therefore, when agent A estimates the cost of each possible path, it considers only the heatmap of agent *C*. As seen from Fig. 4, agent *A* generates the heatmap of agent *C* for *t*=1 and *t*=2 based on the broadcasted path information by agent *C*. Let us assume it will calculate the estimated cost of its initial path, (2, 2), (2, 3), (3, 3). At *t*=1, it is supposed to be at (2, 3), and the value of this cell is 0.33, while at *t*=2 it will be (3, 3), and its heatmap value is also 0.33 according to the heatmap of the agent *C*. The path cost would be estimated as (0.66 =0.33+0.33).

When multiple agents are present in the heatmap, the values of the cells are added up, which is illustrated in Fig. 5 depicting a portion of the heatmap generated by agent *A* at time step *t* for its negotiation session with agent *D*. It is assumed that agent *A* can listen to

Fig. 5 Heat generated by agents B and C for agent A



0	0	0	0	0	0	0	0	0
0	0	0	0.33	0	0.33	0	0	0
0	0	0.33	0.66	0.66	0.66	0.33	0	0
0	0.33	0.66	B	1.32	C	0.66	0.33	0
0	0	0.33	0.66	0.66	0.66	0.33	0	0
0	0	0	0.33	0	0.33	0	0	0
0	0	0	0	A	0	0	D	0

agent B, C, and D broadcasts, and agent A is engaged in a negotiation session with agent D. Note that the heat values of the opponent agent are not considered during the negotiation. Consequently, heat values generated by agent D are ignored. However, the paths of agent B and C are assumed to be steady for the duration of the negotiation. Therefore, their broadcasted information is used to generate a heatmap, which has a portion of it seen in Fig. 5.

Heatmap agent follows almost the same strategy as the Path-Aware strategy but takes the potential future conflicts into account; therefore, its cost estimation differs from that of the Path-Aware agent. Unlike the Path-Aware agent, which uses the actual cost of the paths, the Heatmap agent estimates the cost of the bids (i.e., paths) according to Algorithm 2. The overall cost is the sum of the estimated cost of each cell in the given path. The cumulative cost is estimated by considering the relevant cell value on the Heatmap of each neighbor agent other than the current opponent agent. Line 2 iterates through the points in the given path whose cost is estimated, while Line 3 iterates through the heatmaps of the neighbor agents except for the opponent. The total cost is increased by each relevant cell value of the heatmaps (Line 4). After calculating the estimated cost of each possible bid, the Heatmap agent sorts them in descending order and employs the approach explained in Algorithm 1.

Algorithm 2 Path cost estimation from heatmap

```
  Data:
  $H_T^j$ : Heatmap for Agent $j$ at a given time $T$
  $N_c$: Neighbor agents excluding the opponent in FoV
  $P_c$ : A given path whose cost will be estimated
  Output:
  $EC$ : Estimated cost of a given path, $P_c$
1 $EC \leftarrow 0$
2 for $i \leftarrow 1$ to $n \in |P_c|$ do
3     foreach $j \in N_c$ do
4         $EC \leftarrow EC + H_{t+i}^j.\mathrm{get}(P_c^i)$
5     end
6 end
7 return $EC$
```

Throughout its life cycle in the environment, an agent can continuously observe its environment. By observing the changes in their environments, agents can deduce congestion in an area and behave to avoid dense regions. One way to introduce congestion avoidance while designing the agent is to introduce a system to help the agent avoid bids that would direct it toward congested areas. In this strategy, the agent utilizes the information available in FoV to generate a simple heatmap to perceive the crowd and avoid making bids leading to those regions. The generated heatmap will allow the agent to differentiate between offers by weighting bids that lead into crowded regions heavier than those that do not.

5 Evaluation

We simulate scenarios for the four introduced MAPF settings to evaluate the proposed negotiation approaches. We report the empirical results regarding several metrics: solution rate, solution quality, number of negotiations, and information sharing rate. Considering these metrics, we analyze the effect of negotiation strategy, the field of view (FoV) size, and the commitment types on the system behavior. Besides, we compare the performance of our decentralized MAPF framework with the centralized MAPF approach. In the following sections, we first introduce experimental setup and evaluation metrics. Then, we present and analyze the results.

5.1 Experimental setting

We deploy the state-of-the-art optimal search algorithm for centralized MAPF, Conflict-based Search (CBS) [28], to examine the performance of our decentralized MAPF approach. CBS is a two-level search algorithm in which high-level search resolves path conflicts by generating a set of constraints for conflicted agents. In contrast, an optimal shortest-path algorithm, like A*, generates a single path under these constraints in the low-level search. Here, a constraint is a tuple (i, v, t) where agent i is prohibited from occupying vertex v at time step t. At the high-level search, a binary tree of constraint nodes is operated to resolve conflicts between paths where each node consists of paths and a set of constraints for each agent. When a conflict between two paths is detected, two child nodes are generated by constraining respective agents. In each child node, one agent in the conflict is prohibited from using a conflicted vertex or edge by adding a constraint, and a new path is searched for that agent at the low level under the new constraint set. CBS was used for benchmarking in our previous work [11], demonstrating the limited scalability of CBS.

In this work, our experiments utilize the most enhanced versions of CBS, which are CBSH2-RTC [18] and Explicit Estimation Conflict-Based Search (EECBS) with all improvements, named 'EECBS+' in [19]. CBSH2-RTC is the latest advanced optimal MAPF algorithm, including several reasoning techniques and improved heuristics. EECBS is a bounded suboptimal MAPF algorithm that guarantees the solution cost by an suboptimality factor of w. It applies different node selection rules than CBS at low and high-level searches. Instead of a best-first search in A* at the low level, it follows a focal search in which solutions in the focal list have a cost value at most $w \cdot f$ where f is the lower bound to the optimal solution cost found during the search. EECBS manages its high-level search to reach a suboptimal solution faster by utilizing inadmissible information about potential cost increments [32].

We run optimal EECBS (i.e., with the suboptimality factor of 1.0) and EECBS with the suboptimality factor of 1.1 (EECBS-1.1), using its official implementation at https://github.com/Jiaoyang-Li/EECBS.. The EECBS repository includes a separate branch for the 'disappear-at-target' MAPF problem; therefore, we can include it in benchmarking for all four MAPF settings introduced in Sect. 3 (by turning off wait-action for Setting 1 and

3). However, the official implementation of CSBH2-RTC at https://github.com/Jiaoyang-Li/CBSH2-RTC does not include a separate branch for the 'disappear-at-target' MAPF problem; thus, it is benchmarked for only Setting 1 and 2. We set the time limit to one minute for CBSH2-RTC, EECBS, and EECBS-1.1 per scenario. For optimality evaluation, we only consider the scenarios in which the optimal solution is found with EECBS.

In this evaluation study, we focus on how decentralized approaches become useful when the density of the environment increases. Accordingly, we provide extensive analysis over 20, 40, 60, and 80-agent scenarios on 16×16 empty map (i.e., no obstacles), generated using the benchmark dataset provided by [30]. Additionally, we conducted experiments on 32×32 maps (from the benchmark dataset by [30]) with no obstacle, 10%, and 20% obstacle density to reveal the effect of obstacle density on the system performance. The results for 32×32 maps are provided in the "Appendix 1", and the code repository and data are publicly available for reproducibility at https://github.com/erancihan/mapp.

Each setting of the MAPF problem (i.e., settings 1–4) experimented with 100 scenarios in a 16×16 grid without obstacles for 20, 40, 60, and 80-agent scenarios in standard commitment protocol. In all scenarios, each agent initially has five tokens ($Q = 5$). All scenarios were generated with randomly distributed initial path lengths between 4 and 24 using the MAPF benchmark dataset provided by [30]. Each combination of the configurations mentioned above was repeated for FoV sizes 5, 7, and 9 (i.e., FoV5, FoV7, and FoV9). These runs were repeated five times, considering randomness in the agent's suboptimal bid selection and bilateral partner selection. All experiments were conducted on 12 computers with an 8-core 3.2 GHz processor and 32 GB RAM.

We evaluate the performance of the proposed decentralized approaches in each problem setting by comparing their performance with that of the centralized MAPF algorithms in terms of several metrics. The first metric we use is the *solution rate*, which denotes the ratio of the number of conflict-free path solutions for all agents over a given MAPF configuration. In addition to solution rate, solution quality is another criterion for benchmarking suboptimal solvers, typically defined as an *optimality gap*. For the MAPF problem, the optimality gap describes the rate of the path cost difference between the solutions obtained by the given approach and an optimal solver. It is formulated in Eq. 1 where $|\pi_{solution}|$ is the average of the final path lengths of agents provided by the proposed approach and $|\pi^*_{solution}|$ is the average of optimal path lengths provided by EECBS. Since proposed decentralized approaches could yield solutions for some scenarios that EECBS cannot solve, we use *normalized path difference* metric to compare the performance of those decentralized approaches. It is defined in Eq. 2 where N denotes the number of scenarios, $|\pi^i_{Dsolution}|$ is the average of the final path lengths of agents provided by the proposed approach on the i^{th} scenario and $|\pi^{*i}_{Dsolution}|$ is the average length of the shortest paths provided by available decentralized approaches.

$$\text{Optimality Gap} = \frac{|\pi_{solution}| - |\pi^*_{solution}|}{|\pi^*_{solution}|} \tag{1}$$

$$\text{Normalized Path Difference} = \frac{1}{N} * \sum_{i=1}^{N} \frac{|\pi^i_{Dsolution}| - |\pi^{*i}_{Dsolution}|}{|\pi^{*i}_{Dsolution}|} \tag{2}$$

Moreover, we evaluate the proposed decentralized approaches considering the average number of negotiations and privacy of agents' paths. We define information sharing rate (IS) to consider how much an agent's final path is shared with other k-1 agents. Equation 3 defines the information sharing rate of agent j by averaging the rate of agent j's shared path with other agents where $\pi^{j}_{solution}$ and $\pi^{j\to i}_{broadcasted}$ denote the final path of agent j and the part of that path shared with other agents during broadcasting, respectively. The information-sharing rate is averaged over all agents for evaluating a given scenario.

$$IS_j = \frac{1}{k-1} * \sum_{i=1}^{k-1} \frac{|\pi^{j\to i}_{broadcasted} \cap \pi^{j}_{solution}|}{|\pi^{j}_{solution}|}, i \neq j \tag{3}$$

For all metrics mentioned here, we only consider the results of the runs where each strategy could find a solution for the given scenario to evaluate the strategies fairly. The results used for each metric calculation are averaged over methods that all compared approaches could solve.

In the following sections, we first compare proposed decentralized approaches with centralized ones (Sect. 5.2) and analyze the decentralized MAPF approaches elaborately (Sect. 5.3). Afterward, we investigate the effect of the commitment types (Sect. 5.4).

5.2 Decentralized versus centralized MAPF approaches

To compare the performance of our decentralized approaches with the centralized MAPF approach, we have run CBSH2-RTC, EECBS, and its variant EECBS-1.1 to find optimal and suboptimal solutions with an optimality gap of 0.1, respectively, for all scenarios per each set consisting of 100 scenarios. We run our proposed decentralized approaches with the standard commitment type, namely Heatmap and Path-Aware. Note that the FoV size for those approaches is set to five.[1] In decentralized scenarios, each negotiation has a 1-minute deadline to find an agreement. Therefore, they have to complete the negotiation soon. For example, the total simulation time of an 80-agent 16x16 map scenario took 168 s on average (see Table 4 in the "Appendix 1"). Note that this time limit is more than needed as a negotiation typically ends within a second. In fact, without relying on the time limit, agents might not resolve a conflict during a negotiation due to the previous commitments with other agents blocking their alternative paths. In those settings, agents try to resolve the conflicts when they occur on their way, while the suboptimal solvers find the solution before taking action. Without a doubt, both systems have significant differences, such as the availability of information and communication, pre-action versus during action, etc., so they may not require precisely the same timeline. This study aims to examine whether the distributed negotiation-based system with limited information exchanges may find a solution and centralized solvers find a solution before actualizing a scenario.

Figure 6 shows the solution rate for CBSH2-RTC, EECBS, EECBS-1.1, Path-Aware, and Heatmap agents with FoV5. It is observed that the decentralized agents can provide higher success rates in high-density scenarios (e.g., when the number of agents is 60 or 80) compared to CBSH2-RTC and EECBS since they cannot find optimal solutions for some scenarios in the given time limit. In comparison for the dense scenarios, decentralized approaches, particularly Heatmap, reach higher success rates for Setting-4 (e.g., the average 0.16 versus 0.67 on 80-agent scenario) while they reach close success rates with CBSH2-RTC and EECBS on 60 and 80-agent scenarios of Setting 1 and 2. On the other hand, when EECBS is allowed to perform with the suboptimality factor of 0.1, it can

[1] We have tested with varying FoV values and got a higher solution rate on average when their FoV size is five (FoV5).

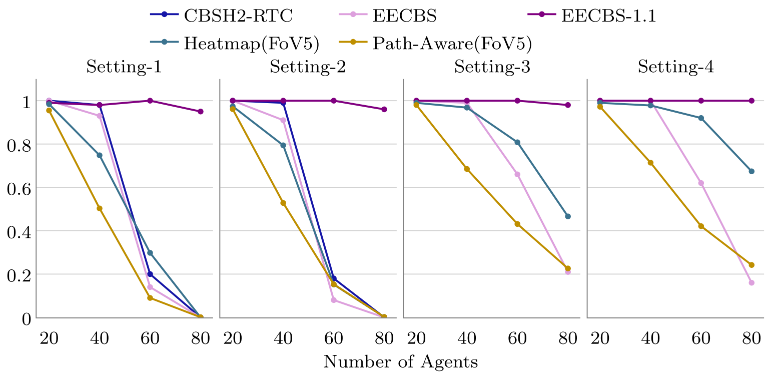


Fig. 6 The average success rates by CBSH2-RTC, EECBS, EECBS-1.1, Path-Aware, and Heatmap agents with FoV(5)

Table 1 The success rates by CBSH2-RTC, EECBS, EECBS-1.1, Heatmap, and Path-Aware, respectively

	k = 20	k = 40	k = 60	k = 80
Set. 1	1/1/0.99/0.98/0.96	0.99/0.93/0.98/0.75/0.5	0.2/0.14/1/0.30/0.09	0/0/0.95/0/0
Set. 2	1/1/1/0.97/0.96	0.99/0.91/1/0.79/0.53	0.18/0.08/1/0.15/0.15	0/0/1/0/0
Set. 3	NA/1/1/0.99/0.98	NA/0.99/1/0.97/0.68	NA/0.66/1/0.81/0.43	NA/0.21/0.98/0.47/0.23
Set. 4	NA/1/1/0.99/0.97	NA/1/1/0.98/0.71	NA/0.62/1/0.92/0.42	NA/0.16/1/0.67/0.24

NA refers that CBSH2-RTC runs are not available for Setting 3 and 4

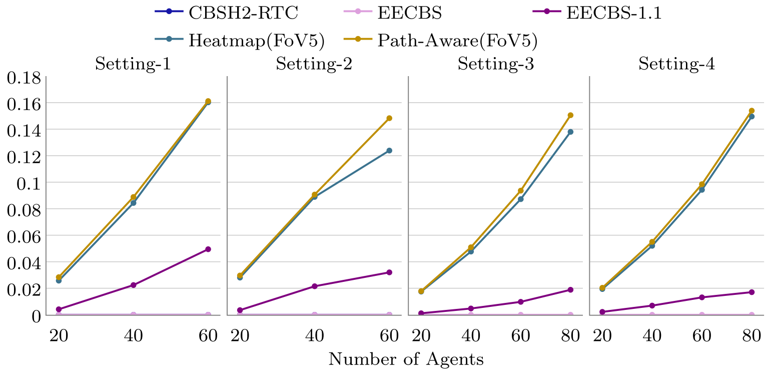


Fig. 7 Optimality gaps by EECBS, EECBS-1.1, Path-Aware, and Heatmap agents with FoV(5)

provide solutions for most scenarios in all problem settings. The success rates of CBSH2-RTC, EECBS, EECBS-1.1, Heatmap, and Path-Aware approaches are given in each cell in Table 1 where *k* denotes the number of agents.

The decentralized approaches perform with a 16% optimality gap on average, whereas EECBC-1.1 produces solutions with a 6% average optimality gap compared to EECBS (Fig. 7). Moreover, we can see that extending the Path-Aware agent and developing the

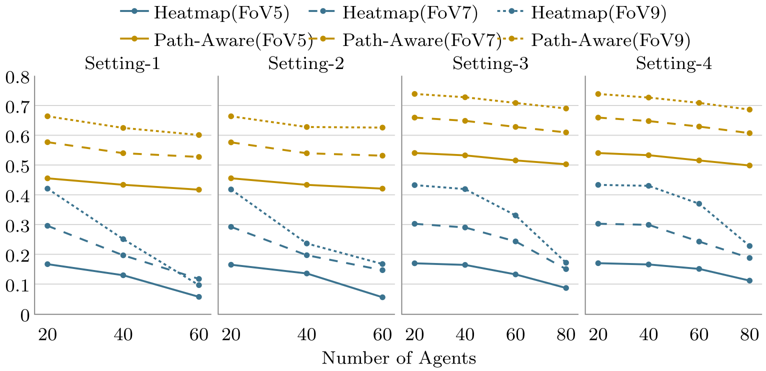


Fig. 8 Average information sharing rate for each agent

Heatmap agent has improved the overall solution rate and slightly decreased the optimality gap in decentralized setup. Since the Heatmap agent is designed to be aware of the environment, these results show that a more sophisticated agent design can yield better performance in the proposed negotiation-based MAPF framework.

Lastly, we investigate to what extent the decentralized approaches consider privacy. It is worth noting that the centralized approaches may find better solutions; however, they use global information on all agents' paths to resolve conflicts, whereas agents utilize local information to find conflict-free paths in the proposed decentralized MAPF framework. Figure 8 shows Path-Aware and Heatmap agents' average information share rate for each setting with varying FoV sizes. It is seen that when agents employ the Heatmap strategy during their negotiation, the information shared with other agents is significantly lower than that of Path-Aware agents. Since the Heatmap agent employs a density-aware strategy, which avoids crowded areas, it tends to change its path rapidly.

Moreover, it is observed that the average information-sharing rate is higher when the FoV is larger, irrespective of the condition, as is expected, since a larger FoV makes agents perceive more neighbor agents. Contrary to expectations, when the environment becomes denser (i.e., involving more agents), the information-sharing rate and the number of broadcasted agents decrease. Our observation is that the percentage of agents encountered in FoV is increased relatively less than the growth rate of the population (see Fig. 9). Therefore, even though the number of message exchanges is similar in both settings, as seen in Fig. 9, the proportion of its final path is shared with others relatively less than the case of the Path-Aware agents.

Besides, we might expect to see a decrease in the information sharing rate when dealing with a MAPF problem in the case of the agents disappearing at their targets (Settings 3 & 4) because they are not obstacles for others anymore when they reach their destination. However, when we analyzed the scenarios solved, we noticed that agents mostly solved less conflicting scenarios in all settings, and agents in settings 3–4 could also solve more complicated scenarios. Therefore, agents in those settings encountered more agents; thus, the average information-sharing rate was higher (Fig. 8).

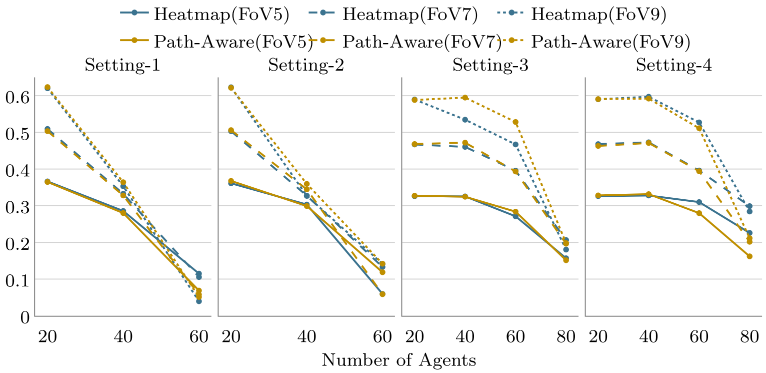


Fig. 9 The ratio of the average number of broadcasted agents to total number of agents

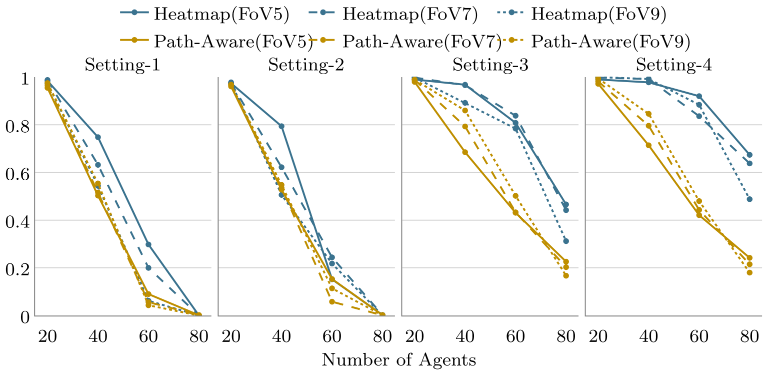


Fig. 10 Success rates by Heatmap and Path-Aware agents for each FoV size

5.3 Analysis of decentralized MAPF approaches

As discussed before, Heatmap agents can provide solutions in challenging settings where agents have to take action in each time step and act as an obstacle for other agents in the system when they reach their destinations (i.e., standard setting – Setting 1). This section examines how the proposed negotiation approaches handle the same scenarios if agents can wait at any time (i.e., Settings 2 & 4) and disappear from the environment when they reach their goals (i.e., Settings 3 & 4). Consequently, we evaluate the performance of the proposed approaches on the previously generated 16×16 scenarios for four different settings of MAPF problem as defined in Sect. 3.

Figure 10 presents the success rates of Heatmap and Path-Aware agents in each environment density with each FoV size (i.e., FoV = 5, 7, and 9) for all problem settings. Enabling wait action and the behavior of disappearing at the destination (Setting 4) positively affects the solution rates. Both agent types perform much better when disappearing at target, especially for scenarios consisting of 40, 60, and 80 agents. That may stem from the fact that the complexity of the problem is reduced since the agents at targets are no longer obstacles for others. Another observation is that the performance of the Heatmap agent is affected more positively when the agents are allowed to wait anytime, especially in the case of the disappearing setting (Setting-3 versus Setting-4).

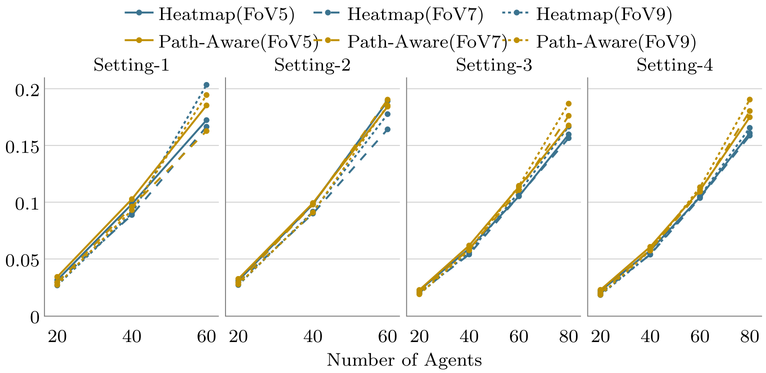


Fig. 11 Average Normalized path differences for Heatmap and Path-Aware agents for each FoV size

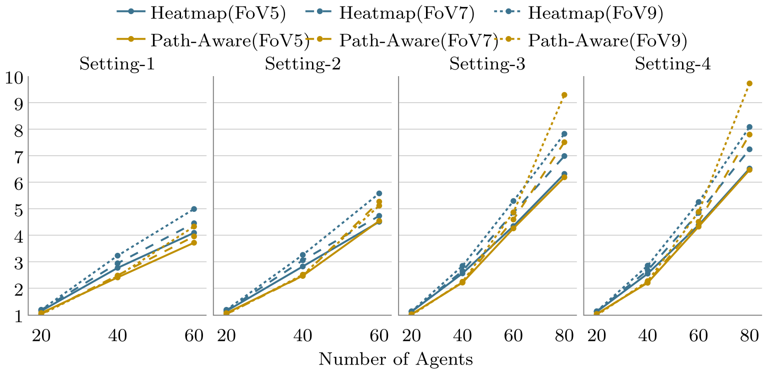


Fig. 12 Average number of negotiations for each agent type

The solution rate drastically drops when we increase the number of agents to 80, especially when the agents are allowed to take wait-action to resolve path conflicts with others. The empty cells in the 16x16 grids significantly decrease when there are 80 agents-in addition to allowing agents to wait when they can act as an obstacle for other agents. To sum up, this will lead to more conflicts among agents, which cannot be straightforwardly resolved. In most cases, agents are stuck and cannot move in any direction – ending up with a low solution rate.

Regarding the quality of the solutions, Fig. 11 shows the average normalized path differences for both agent types in four settings with varying FoV sizes. For 80-agent scenarios of Settings 1 and 2, average normalized path differences are missing since the solution rate equals zero in these problem configurations. It is observed that solution quality differences between Path-Aware and Heatmap agents. The heatmap agent is slightly better than Path-Aware, while the success rates of the Heatmap agent are higher. That shows the superiority of the Heatmap agent over the Path-Aware agent. Heatmap agents can resolve path conflicts without sacrificing path length. Heatmap agents can achieve solutions at slightly lower average normalized path differences than Path-Aware agents. This result promotes the design of environment-aware agents for MAPF problems rather than agent designs only focusing on token consumption. Figure 12 shows the average number of negotiations per agent. In general, agents have an almost equal

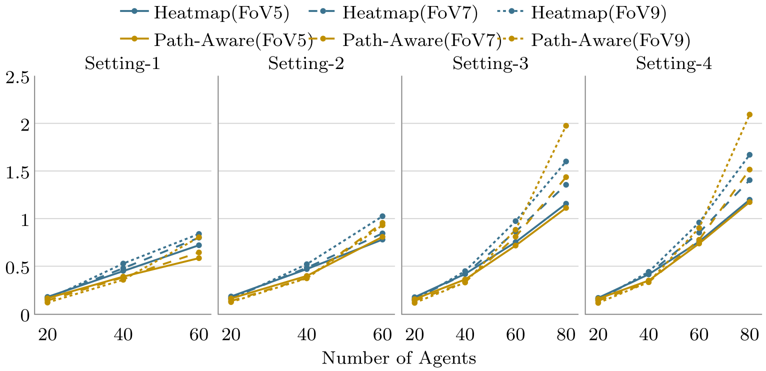


Fig. 13 The normalized average of total token exchange

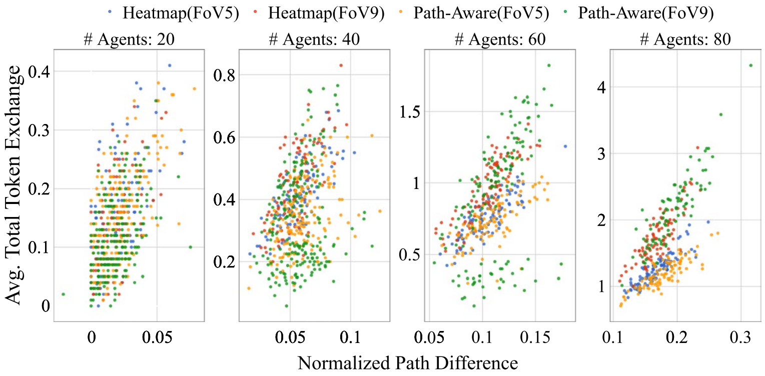


Fig. 14 The normalized average of total token exchange and normalized path difference distributions for setting-4

number of negotiations in each setting except when the number of agents is 80. In that case, the higher the FoV is, the higher the number of negotiations that occur (see Setting 4) since agents can detect more conflicts with larger FoV.

The detection of more conflicts leads to exchanging more tokens. Figure 13 shows the normalized average value of the total token exchanges in a scenario (i.e., the total amount of exchanged tokens in a scenario, normalized by the number of agents) for each problem set. Token exchanges are highly correlated with environment density and FoV size. As agents with larger FoVs engage in negotiations more, they exchange more tokens. To reveal the correlation between the normalized path difference and the amount of token exchanges, we present Fig. 14 for Setting-4. As agents face conflicts with each other more, they diverge from their optimal length paths more, resulting more token exchanges. As clearly seen in the dense scenarios (60 & 80-agent), agents with narrower FoVs use tokens more effectively while preserving solution quality. As a result, larger FoVs inflate token exchanges.

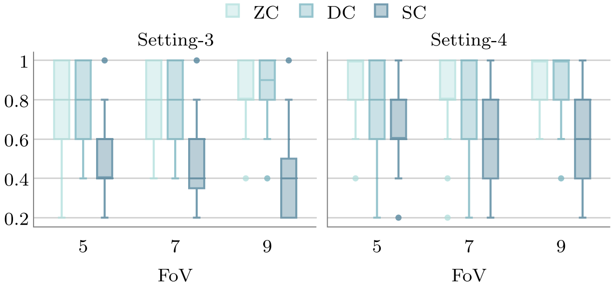
5.4 The effect of commitment types

In this section, we provide a focal analysis of the system performance using the proposed negotiation approach to examine how it evolves in dense environments; therefore, we consider only 80-agent scenarios for Setting-3 and Setting-4 MAPF configurations. Recall that agents disappear when they reach their target in both settings. Agents can take a wait action at any time step until reaching their destination in Setting 4, whereas they cannot wait in Setting 3. Generally, the Heatmap agent has provided better results than the Path-Aware agent in many metrics when using the standard commitment protocol. Hence, we only focus on the Heatmap agent to analyze the effect of the commitment types. All results throughout the section are provided in the taxonomy of agent configurations, which are the commitment types (SC, ZC, and DC) and the FoV sizes (5, 7, and 9). We consider the aforementioned performance metrics other than the optimality gap since EECBS could not provide optimal solutions for the scenarios involving 80 agents.

Depending on their commitment types, agents may adhere to or retract from the agreed path allocation from their previous negotiations. While agents utilizing SC entirely stick to their prior agreements and generate offers that do not conflict with the previous agreements, agents utilizing ZC can change their path and conflict with their previous commitments anytime after the agreement. In contrast, agents utilizing DC must stick to the agreed allocation until the time step of resolved conflict (i.e., previously conflicted state). However, after that state, the agent is free to change its path. Figure 15 shows the success rates of the Heatmap agent for Setting 3 and 4. When we analyze those results, it is seen that the Heatmap agent reaches more solutions in zero commitment than SC in both settings. In particular, the average success rate differences between zero and standard commitments are %31-%35-%52 in Setting-3 and %18-%18-%40 in Setting-4 with the respective FoV sizes. In SC, the agents narrow their search space by sticking to their previous agreements; thus, this situation may prevent agents from finding an agreement in further negotiations. As a result, allowing agents to decommit their previous negotiation agreements may increase the chance of reaching their final destination. In addition, the success rate differences between agents utilizing ZC and DC are minimal (%1-%1-%0 in Setting-3 and %6-%1-%3 in Setting-4 with the respective FoV sizes). It is worth noting that the success rates of agents utilizing ZC and DC slightly increase, whereas the success rates of agents utilizing SC drastically decrease when FOV size increases.

As seen from Fig. 16, there is no evidence of remarkable differentiation between medians (also deviations) of the normalized path difference in Setting-3 and Setting-4. When we apply the Mann–Whitney U statistical tests, there is no statistically significant difference among SC, ZC, and DC ($p > 0.05$) except in the case when the FOV is equal to 5 in Setting 3 (SC vs. ZC: $p = 0.046 < 0.05$) and SC vs. DC: $p = 0.024 < 0.05$). This supports

Fig. 15 Success rates for each commitment type



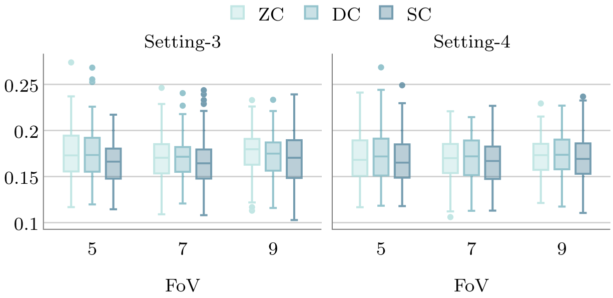


Fig. 16 Normalized path difference for each commitment type

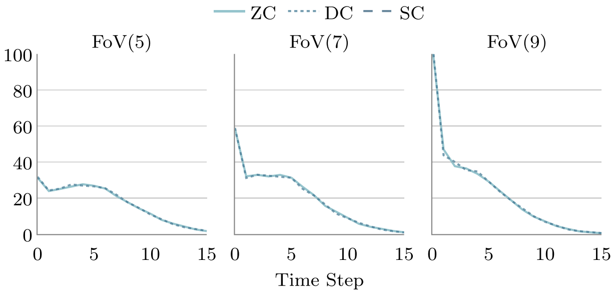


Fig. 17 Average number of negotiations per time step in setting-4

the observation that agents utilizing ZC and DC can find more non-conflicting paths for the given scenarios while their solutions are as cost-efficient as in the case of the SC.

Moreover, we present a new metric to analyze how often agents interact with each other to resolve observed conflicts throughout a scenario in our negotiation-based decentralized MAPF system for each commitment type. The average number of negotiations per time step is presented for each commitment type and each FoV size in Fig. 17. This metric provides granular information about the interaction level of agents in a decentralized system. Agents engaged in a comparably higher number of negotiations in the system's initial step (i.e., t = 0), and the number of negotiations significantly decreased at t = 1 in all FoV configurations. Note that the average number of negotiations naturally decreases as the number of agents arriving at their destination increases.

Secondly, another distinctive effect of the commitment types comes from the system behavior differences concerning FoV size. Agents have a chance to resolve a more significant number of conflicts beforehand within a time step with larger FoV sizes. Hence, agents engage less frequently with FoV9, continuously decreasing the number of negotiations. On the other hand, the average number of negotiations between t = 1–4 increases in the FoV5 setting and remains stable in the FoV7 setting. This characteristic curve change from FoV size 5 to 9 reveals how information sharing among self-interested pathfinder agents affects the conflict resolution behavior of a decentralized system. Agents with narrow FoV observe fewer conflicts beforehand but repeatedly interact with near agents to resolve conflicts at earlier stages. On the other hand, agents with larger FoV can resolve more conflicts at once and monotonically fewer interactions with each other later. It indicates that the system converges to a solution faster with larger FoVs, and the solution rate slightly rises without compensation of solution quality as seen in Figs. 15 and 16.

We also consider the privacy of self-interested agents as a third dimension for evaluating a decentralized MAPF system performance alongside success rate and solution quality metrics. Information sharing among the agents is significantly higher when they employ

Fig. 18 Average information sharing rates for each commitment type

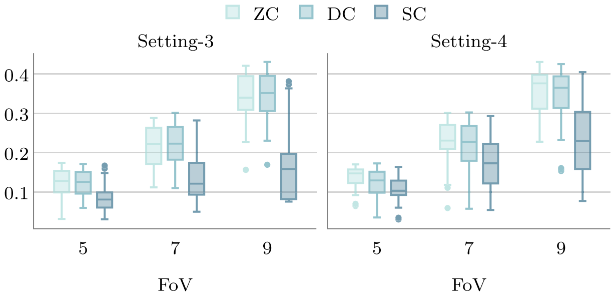


Fig. 19 The ratio of success rate to average information sharing rate for each commitment type (higher is better)

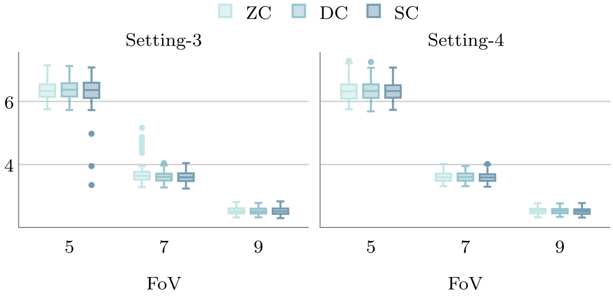
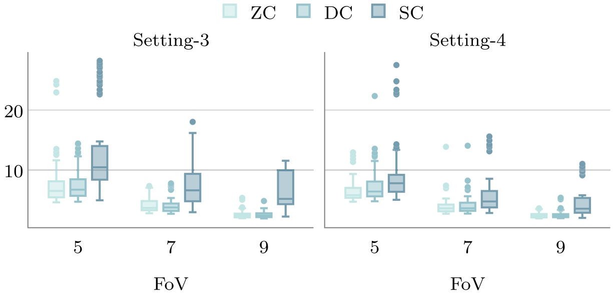


Fig. 20 The ratio of '1—normalized path difference' to average information sharing rate for each commitment type (higher is better)



ZC and DC commitment types than SC, as seen in Fig. 18. Another observation is that the increase in the information sharing rate is higher for ZC and DC when the FoV size of agents increases since the agents with SC engage in a fewer number of negotiations (see Fig. 17), so they share less unique information about their planned paths with others.

On the other hand, Heatmap agents with ZC and DC share more information to resolve conflicts, thus achieving much higher success rates (see Fig. 15). The success rate and the solution quality are evaluated proportionally with the information sharing rate, which is presented in Figs. 19 and 20, respectively. The system performance trades in favor of privacy by a clear margin when agents have a narrow FoV. Note that trade-offs between privacy and solution performance in terms of success rate and quality are linearly taken. Compared to the FoV5 configuration, the success rate over the average information sharing rate substantially decreases in Settings 3 and 4 with FoV7 and FoV9 configurations. Similarly, the decrease in the solution quality over the average information-sharing rate is significant when agents have FoV7 and FoV9 in both problem settings.

6 Conclusion

This paper addresses how self-interested agents can coordinate in a grid environment to reach their destination without collision and proposes a negotiation approach to resolve conflicts between their paths. Accordingly, we propose a negotiation protocol and two compatible negotiation strategies: Path-Aware and Heatmap. While the Path-Aware agent is more concerned about the length of its absolute path, the Heatmap agent considers future conflict among agents. The heatmap agent exploits the broadcasted path information by other agents accordingly. The proposed approach is evaluated elaborately in different settings where agents are allowed/disallowed to take a wait action and/or disappear when they reach their final destinations. In addition, we introduce three types of commitments and study their effect on the success rate and solution quality. Furthermore, we present a new metric called *information sharing rates* to discuss the trade-off between privacy and solution performances for MAPF problems.

According to the experimental result, the Heatmap agent outperforms the Path-Aware agent concerning the success rate and information sharing rate in all settings. Moreover, the Heatmap agent achieves higher success rates than the optimal algorithms for MAPF, CBSH2-RTC, and EECBS when agents can take wait action and disappear at their targets (Setting 3 and 4). On the other hand, EECBS, with a suboptimality factor of 1.1, outperformed the proposed approaches regarding both solution and optimality rates. Nevertheless, we emphasize that our approach concerns the agents' privacy; therefore, there is a trade-off between privacy and performance. When agents utilize zero or dynamic commitment types, they perform better regarding success rates than when adopting a standard commitment. Furthermore, we observed that the average number of negotiations in each time step and information sharing rates are higher when FoV increases.

In future work, we plan to extend our approach by adopting multilateral negotiation instead of multiple consecutive bilateral negotiations and compare its performance with the proposed method. Moreover, designing more sophisticated agents capable of thinking ahead in their negotiation and path-planning processes is necessary to improve the solution quality and rate. Additionally, when a negotiation fails, our framework stops the run for the entire scenario. However, some agents can continue negotiating with each other to reach their destination, while others fail. One of the open questions in negotiation-based solutions for regulating self-interested agents is whether they need an incentive to move after reaching their destination due to negotiations with other agents who need to pass there. Currently, we are working on another incentive mechanism for the agent who reaches its destination. As a future work, our approach could be extended to provide such incentives using more advanced token mechanisms. Examining how many agents can reach their destinations in such settings would be interesting. Without a doubt, the trade-off between privacy and performance should be further investigated elaborately. Lastly, new decentralized agent strategies could be designed to handle special conflict types in different topologies by adopting the reasoning techniques introduced in the recent CBS studies [18].

Appendix 1: The results for the 32 × 32 maps

To inspect the effect of obstacle density and map size, we experimented with 25 scenarios for 32 × 32 maps with 0, 10, and 20% obstacle density, using the benchmark dataset by [30]. Table 2 includes the average success rates of CBSH2-RTC, EECBS,

Table 2 The average success rates by CBSH2-RTC, EECBS, EECBS-1.1, and heatmap agents for 25 80-agent scenarios in 32 × 32 Maps

Map	MAPF setting	CBSH2-RTC	EECBS	EECBS-1.1	Heatmap FoV(5)	Heatmap FoV(7)	Heatmap FoV(9)
Empty 32-32	Setting-1	0.92	0.88	1	1	0.96	1
	Setting-2	NA	1	1	1	1	1
	Setting-3	0.92	0.92	1	1	1	0.84
	Setting-4	NA	1	1	1	1	1
Random 32-32-10	Setting-1	0.84	0.64	1	0.96	0.88	0.64
	Setting-2	NA	0.8	1	0.92	0.96	0.88
	Setting-3	0.8	0.68	1	0.84	0.76	0.44
	Setting-4	NA	0.96	1	1	1	0.8
Random 32-32-20	Setting-1	0.04	0	1	0.16	0	0
	Setting-2	NA	0.4	1	0.08	0.12	0.04
	Setting-3	0.04	0	1	0.24	0.04	0.04
	Setting-4	NA	0.44	1	0.52	0.12	0

Table 3 The average normalized path differences by heatmap agents with FoV sizes of 5, 7, and 9 for commonly solved scenarios

Map	MAPF setting	Heatmap FoV(5)	Heatmap FoV(7)	Heatmap FoV(9)
Empty 32-32	Setting-1	0.0307	0.0282	0.0254
	Setting-2	0.0174	0.0170	0.0152
	Setting-3	0.0320	0.0271	0.0232
	Setting-4	0.0175	0.0162	0.0142
Random 32-32-10	Setting-1	0.1004	0.0885	0.0776
	Setting-2	0.0728	0.0604	0.0511
	Setting-3	0.1043	0.0824	0.0772
	Setting-4	0.0728	0.0602	0.0529

EECBS-1.1, and Heatmap agents with zero commitment and FoV sizes of 5, 7, and 9. The advantage of Heatmap with FoV(5) over CBSH2-RTC and EECBS in terms of the success rate becomes more prominent as the obstacle density of the map increases. While agents with narrow FoV is leads to resolve conflicts more successfully than those with larger FoV, their average solution quality is lower (i.e., higher normalized path difference) than those having larger FoV as seen in Table 3. Note that only commonly solved instances are used for comparing the average normalized path differences by each agent type for a fair evaluation. Since three agent types cannot commonly solve any instance in a 32×32 map with 20% density, the results for that map are not available in Table 3.

Table 4 presents the average simulation run times of solved scenarios for each decentralized agent type with standard commitment. Note that our implementation handles negotiations sequentially, as we cannot allocate one CPU thread per agent. In a real application, each agent would have its own computation power, allowing a decentralized system with multiple negotiations among different agent pairs simultaneously. Therefore, these run times do not present the application-side performance of our decentralized MAPF solution.

Table 4 The average (± standard deviation) simulation run times (in seconds)

Number of agents	MAPF setting	Heatmap FoV(5)	Heatmap FoV(7)	Heatmap FoV(9)	Path-Aware FoV(5)	Path-Aware FoV(7)	Path-Aware FoV(9)
20	Setting-1	52.5 ± 10	54.2 ± 11	54.1 ± 10	52.3 ± 10	52.6 ± 11	52.7 ± 9
	Setting-2	52.1 ± 9	53.1 ± 10	55.2 ± 10	51.9 ± 10	52.2 ± 10	53.3 ± 10
	Setting-3	41.3 ± 6	40.6 ± 5	39.7 ± 5	40.2 ± 5	39.3 ± 5	38.3 ± 5
	Setting-4	41.4 ± 5	40.5 ± 5	39.6 ± 4	40.4 ± 5	39.2 ± 5	38.3 ± 4
40	Setting-1	116.8 ± 19	120.0 ± 18	121.4 ± 18	110.7 ± 21	115.5 ± 19	117.1 ± 22
	Setting-2	120.6 ± 17	122.4 ± 18	121.8 ± 22	110.2 ± 21	116.1 ± 19	113.5 ± 21
	Setting-3	70.2 ± 8	70.0 ± 9	74.4 ± 16	63.8 ± 9	60.7 ± 10	59.1 ± 10
	Setting-4	70.0 ± 9	68.5 ± 8	67.3 ± 8	63.4 ± 9	61.2 ± 10	60.6 ± 12
60	Setting-1	226.0 ± 50	228.6 ± 41	205.6 ± 28	223.6 ± 35	220.6 ± 44	216.7 ± 44
	Setting-2	215.5 ± 39	228.0 ± 39	238.7 ± 42	220.5 ± 33	226.9 ± 40	239.1 ± 48
	Setting-3	111.3 ± 15	111.0 ± 13	111.9 ± 14	105.6 ± 17	106.5 ± 21	103.9 ± 27
	Setting-4	112.7 ± 17	111.4 ± 14	110.3 ± 14	108.8 ± 16	105.4 ± 21	105.3 ± 28
80	Setting-1	NA	NA	NA	309.2 ± 0	437.6 ± 50	NA
	Setting-2	351.9 ± 0	437.8 ± 0	NA	365.9 ± 11	275.0 ± 0	365.2 ± 0
	Setting-3	176.1 ± 33	167.4 ± 22	169.0 ± 21	162.9 ± 21	182.5 ± 29	199.7 ± 35
	Setting-4	173.8 ± 22	170.2 ± 22	180.8 ± 25	164.1 ± 17	178.9 ± 23	207.9 ± 33

Author contributions All authors have equal contributions to this study.

Funding No funding for this study.

Data availability The authors can share data and materials upon request.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval Not applicable to this study.

References

1. Amir, O., Sharon, G., & Stern, R. (2015). Multi-agent pathfinding as a combinatorial auction. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, AAAI'15* (pp. 2003–2009). AAAI Press.
2. Aydoğan, R., Hindriks, K. V., & Jonker, C. M. (2014). Multilateral mediated negotiation protocols with feedback. In I. Marsa-Maestre, M. Lopez-Carmona, T. Ito, M. Zhang, Q. Bai, & K. Fujita (Eds.), *Novel Insights in Agent-based Complex Automated Negotiation* (pp. 43–59). Studies in Computational Intelligence, Vol. 535. Springer, Tokyo. https://doi.org/10.1007/978-4-431-54758-7_3
3. Aydoğan, R., Festen, D., Hindriks, K., & Jonker, C. (2017). Alternating offers protocols for multilateral negotiation. *Studies in Computational Intelligence, 674*, 153–167.
4. Baarslag, T., Gerding, E. H., Aydoğan, R., & Schraefel, M. (2015). Optimal negotiation decision functions in time-sensitive domains. In *2015 IEEE/WIC/ACM international joint conferences on web intelligence (WI) and intelligent agent technologies (IAT)* (Vol. 2, pp. 190–197).
5. Bnaya, Z., Stern, R., Felner, A., Zivan, R., & Okamoto, S. (2013). Multi-agent path finding for self interested agents. In *Sixth annual symposium on combinatorial search* (pp. 39–46).
6. Bonisoli, A., Gerevini, A., Saetti, A., & Serina, I. (2014). A privacy-preserving model for the multi-agent propositional planning problem. In *Proceedings of the Twenty-first European Conference on Artificial Intelligence (ECAI'14)* (pp. 973–974). IOS Press, NLD.
7. Brafman, R. I., & Domshlak, C. (2008). From one to many: Planning for loosely coupled multi-agent systems. In *ICAPS 2008 - Proceedings of the 18th International Conference on Automated Planning and Scheduling* (pp. 28–35).
8. Cole, R. J., Dodis, Y., & Roughgarden, T. (2003). Pricing network edges for heterogeneous selfish users. In *STOC '03* (pp. 521–530).
9. de Oliveira Ramos, G., Rădulescu, R., Nowé, A., & Tavares, A. R. (2020). Toll-based learning for minimising congestion under heterogeneous preferences. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS'20)* (pp. 1098–1106). International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC.
10. Desaraju, V. R., & How, J. P. (2012). Decentralized path planning for multi-agent teams with complex constraints. *Autonomous Robots, 32*(4), 385–403. https://doi.org/10.1007/s10514-012-9275-2
11. Eran, C., Keskin, M. O., Cantürk, F., & Aydoğan, R. (2021). A decentralized token-based negotiation approach for multi-agent path finding. In *The 18th European Conference on Multi-Agent Systems (EUMAS)* (pp. 264–280).
12. Erdmann, M., & Lozano-Perez, T. (1986). On multiple moving objects. In *Proceedings. 1986 IEEE international conference on robotics and automation* (Vol. 3, pp. 1419–1424). https://doi.org/10.1109/ROBOT.1986.1087401

13. Felner, A., Stern, R., Shimony, S. E., Boyarski, E., Goldenberg, M., Sharon, G., Sturtevant, N. R., Wagner, G., & Surynek, P. (2017). Search-based optimal solvers for the multi-agent pathfinding problem: Summary and challenges. In *SOCS* (pp. 29–37).
14. Gautier, A., Stephens, A., Lacerda, B., Hawes, N., & Wooldridge, M. (2022). Negotiated path planning for non-cooperative multi-robot systems. In *Proceedings of the 21st international conference on autonomous agents and multiagent systems, AAMAS '22* (pp. 472–480). International Foundation for Autonomous Agents and Multiagent Systems.
15. Ho, F., Geraldes, R., Gonçalves, A., Rigault, B., Sportich, B., Kubo, D., Cavazza, M., & Prendinger, H. (2022). Decentralized multi-agent path finding for UAV traffic management. *IEEE Transactions on Intelligent Transportation Systems, 23*(2), 997–1008. https://doi.org/10.1109/TITS.2020.3019397
16. Inotsume, H., Aggarwal, A., Higa, R., & Nakadai, S. (2020). Path negotiation for self-interested multi-robot vehicles in shared space. In *Proceedings of the international conference on intelligent robots and systems* (pp. 11587–11594). IEEE.
17. Klein, M., Faratin, P., Sayama, H., & Bar-Yam, Y. (2003). Protocols for negotiating complex contracts. *IEEE Intelligent Systems, 18*, 32–38. https://doi.org/10.1109/MIS.2003.1249167
18. Li, J., Harabor, D., Stuckey, P. J., Ma, H., Gange, G., & Koenig, S. (2021). Pairwise symmetry reasoning for multi-agent path finding search. *Artificial Intelligence, 301*, 103574. https://doi.org/10.1016/j.artint.2021.103574
19. Li, J., Ruml, W., & Koenig, S. (2021). Eecbs: A bounded-suboptimal search for multi-agent path finding. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 35(14), pp. 12353–12362).
20. Li, Q., Gama, F., Ribeiro, A., & Prorok, A. (2020). Graph neural networks for decentralized multi-robot path planning. In *2020 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 11785–11792). IEEE. https://doi.org/10.1109/iros45743.2020.9341668
21. Parkes, D. C. (1999). Ibundle: An efficient ascending price bundle auction. In: *Proceedings of the 1st ACM conference on electronic commerce, EC '99* (pp. 148–157). Association for Computing Machinery. https://doi.org/10.1145/336992.337032
22. Patwardhan, A., Murai, R., & Davison, A. J. (2023). Distributing collaborative multi-robot planning with gaussian belief propagation. *IEEE Robotics and Automation Letters, 8*(2), 552–559. https://doi.org/10.1109/LRA.2022.3227858
23. Pritchett, A., & Genton, A. (2018). Negotiated decentralized aircraft conflict resolution. *IEEE Transactions on Intelligent Transportation Systems, 19*, 81–91.
24. Purwin, O., D'Andrea, R., & Lee, J. (2008). Theory and implementation of path planning by negotiation for decentralized agents. *Robotics and Autonomous Systems, 56*, 422–436.
25. Rosenschein, J. S., & Zlotkin, G. (1994). *Rules of encounter: Designing conventions for automated negotiation among computers*. MIT Press.
26. Roughgarden, T., & Tardos, É. (2002). How bad is selfish routing? *Journal of ACM, 49*, 236–259.
27. Salzman, O., & Stern, R. (2020). Research challenges and opportunities in multi-agent path finding and multi-agent pickup and delivery problems. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '20* (pp. 1711–1715). International Foundation for Autonomous Agents and Multiagent Systems.
28. Sharon, G., Stern, R., Felner, A., & Sturtevant, N. R. (2012). Conflict-based search for optimal multi-agent pathfinding. *Artificial Intelligence, 219*, 40–66.
29. Stern, R. (2019). *Multi-agent path finding—An overview Artificial Intelligence* (pp. 96–115). Springer-Verlag, Berlin, Heidelberg. https://doi.org/10.1007/978-3-030-33274-7_6
30. Stern, R., Sturtevant, N. R., Felner, A., Koenig, S., Ma, H., Walker, T. T., Li, J., Atzmon, D., Cohen, L., Kumar, T. K. S., Boyarski, E., & Barták, R. (2019). Multi-agent pathfinding: Definitions, variants, and benchmarks. In *Symposium on combinatorial search*. https://api.semanticscholar.org/CorpusID:195218865
31. Sujit, P.B., Sinha, A., & Ghose, D. (2006). Multiple UAV task allocation using negotiation. In *AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems, AAMAS '06* (pp. 471–478). Association for Computing Machinery. https://doi.org/10.1145/1160633.1160719
32. Thayer, J., & Ruml, W. (2011). Bounded suboptimal search: A direct approach using inadmissible estimates. In *The Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence* (pp. 674-679).