A Decentralized Token-based Negotiation Approach for Multi-Agent Path Finding

Cihan Eran[1] [0000-0002-1955-3575], **M. Onur Keskin**[1] [0000-0002-8025-4304], **Furkan Cantürk**[1] [0000-0003-4937-6538], and **Reyhan Aydoğan**[1,2] [0000-0002-5260-9999]

[1] Computer Science, Özyeğin University, Istanbul, Turkey
{cihan.eran,onur.keskin,furkan.canturk}@ozu.edu.tr
reyhan.aydogan@ozyegin.edu.tr
[2] Interactive Intelligence, TU Delft, Delft, Netherlands

Abstract. This paper introduces a negotiation approach to solve the Multi-Agent Path Finding problem. The approach 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 a compatible negotiation strategy are presented. The proposed approach is evaluated in a variety of scenarios by comparing it with state-of-the-art centralized approaches such as Conflict Based Search and its variant. The experimental results showed that the proposed approach can find conflict-free path solutions with a higher success rate, especially when the search space is large and high-density compared to centralized approaches while the gap between path cost differences is reasonably low. The proposed approach enables agents to have their autonomy; thus, it is convenient for MAPF problems involving self-interested agents.

Keywords: Multi-Agent Path Finding, Negotiation, Decentralized Coordination, Self-interested Agents

1 Introduction

Technological advancements in the last decades enable autonomous robots and vehicles to carry out a variety of tasks such as surveillance and transportation. To achieve their goal, they may need to navigate from one location to another. Imagine an environment in which hundreds of autonomous robots aiming to reach certain locations. Such an environment requires a coordination mechanism to avoid some potential collisions. This problem, allocating conflict-free paths to agents so as to navigate safely in an environment, is well-addressed in the field of Multi-Agent Systems and known as Multi-Agent Path Finding (MAPF) problem [26]. A vast number of studies tackle this problem; some propose a centralized solution while others focus on decentralized solutions [25].

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 [10]. Without any time constraints, centralized approaches can find optimal solutions if there exist. However, the performance of centralized solution approaches can suffer in high density and complex environments [24, 8]. Besides, full information may not always be available due to the limitation of communication, sensors, or privacy issues. On the one hand, 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.

This work pursues a decentralized approach to the MAPF problem targeting a good trade-off between privacy and effectiveness of the solutions. As agents can resolve their conflicts for varying problems from resource allocation to planning through negotiation [13, 19, 15, 3], we advocate to solve the aforementioned problem in terms of negotiation and accordingly propose a token-based alternating offers protocol. In the proposed approach, agents share their partial path information with only relevant agents that are close to them to some extent. If they detect any conflict on their partial path, they encounter a bilateral negotiation to allocate required locations for certain time steps. To govern this negotiation, this study introduces a variant of alternating offers protocol enriched with token exchanges. By enforcing the tokens' usage, the protocol leads agents to act collaboratively and search unexplored paths so that there is no conflict anymore. A path-aware negotiation strategy is also presented in line with the protocol. There are a few attempts to solve the MAPF problem in terms of negotiation [14, 21, 22]. Either they require sharing full path information with others, or they consider one-step decisions such as who will move to a certain direction at the time of conflict, or they aim to resolve the conflict in one shot (i.e., collaborate or reject). In contrast, our approach aims to reduce the complexity of the problem by resolving the conflicts in subpath plans iteratively instead of the entire path plans, thereby respecting the agents' privacy to some extent.

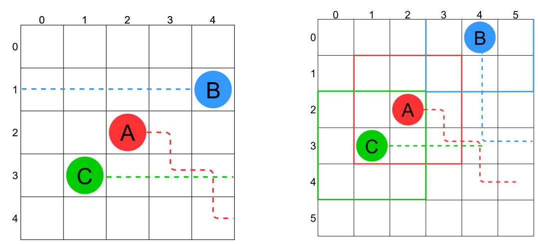
This paper is organized as Section 2 describes the problem addressed in this paper, while Section 3 lays out the proposed solution approach, introduces a new variant of Alternating Offers Protocol and a compatible negotiation strategy in line with that protocol. Experiment setup and results of the experiments are presented in section 4. This paper's main contributions and planned future work are discussed in Section 6.

2 Problem Statement

Multi-Agent Pathfinding (MAPF) as defined in [26], is the problem of assigning conflict free paths to agents from their respective starting locations to their destinations. Formally, we have k agents denoted by $A = \{\ A_1,\ A_2,\ ...,\ A_k\ \}$ navigating in an undirected graph $G = (V, E)$ where starting and destination location for each A_i are denoted by $s_i \in V$ and $g_i \in V$ respectively. The path of each agent A_i is denoted by π_i, a sequence of vertices indexed by each time step $0 \rightarrow n$, $(s_i^0, ..., g_i^n)$. $\pi_i^t \in V$ corresponds to current location of A_i at time step t. At any time step, the agents cannot be located in the same vertex − $\pi_i^t \neq \pi_j^t\ \forall$

$i \neq j$, and traverse the same edge. A swapping conflict occurs when $\pi_i^t = \pi_j^{t+1} \wedge \pi_i^{t+1} = \pi_j^t \; \forall \; i \neq j$.

In this paper, agents are located in a $M \times M$ grid-like environment where each cell corresponds to a vertex as illustrated in Figure 1a. Each agent has an initial path to follow in the addressed problem to reach their destination, shown by dashed lines in the grid. For the current example, we have three agents A, B, and C whose planned paths are colored in their respective colors (i.e., red, blue, and green). Here, an agent located in a cell can only move to their vertical and horizontal adjacent neighboring cells (i.e., cardinal directions). For instance, Agent A located in $(2,2)$ can move to one of the following cells: $(1,2)$, $(3,2)$, $(2,1)$, and $(2,3)$. In the proposed framework, agents cannot wait at a certain cell unless agents reach their destinations. That is, $\pi_i^t \neq \pi_i^{t+1}$ whereas $\pi_i^t \neq g_i$. As seen in the example, there is a conflict between Agent A and C at time step t=2 (cell $(3,3)$). They need to resolve this conflict to achieve their goals. Final solutions will be evaluated under the objective of some of the individual costs, $\sum_{i=1}^{k} |\pi_i|$ where individual cost of each agent i corresponds to their path length denoted by $|\pi_i|$. Furthermore, when an agent reaches its destination, it stops there to act as an obstacle for other agents. This behavior makes this problem a *stay at target* MAPF problem.



(a) (b)

Fig. 1: Example Environment & Field of View Representation

3 Proposed Approach

This work presents a decentralized solution in which agents autonomously negotiate with each other in order to refine their path to avoid possible collisions. The main challenge is to deal 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 that are 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 may not exchange information with each other if their distance is above a threshold value. Besides, full 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 our framework, agents are located in a grid as shown in Figure 1b. Initially, each agent knows only their starting location, destination, and a path plan to reach their destination. Those planned paths are shown in colored dot lines in the grid for each agent. For simulating the aforementioned partially observable environment, we adopt the concept of *field of view*. 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 field of view determines the scope of its communication and perception capacity. An agent can only observe and communicate with other agents if they are within its field of view. That is described as a certain number of cells d from its location. For instance, when d is equal to 1, the boundary of the field of view is shown by a red rectangle for Agent A. In such a case, Agent A can receive/send information from/to only Agent C, which is located in the scope of Agent A's field of view and vice versa. However, in the given snapshot, Agent B cannot communicate or see other agents at that time.

In this framework, agents broadcast their sub-planned path. Agents are free to determine to what extent the path to be shared with other agents in the field of view. 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 is equal to 1, agents will share their current subpath with a length of 2 (i.e., its next two moves) with the agents located the scope of their field of view. 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 C start negotiating on the allocation of vertices on their path since they detect a conflict in $(3,3)$.

When an agent detects a conflict with more than one agent, which negotiation to be held first is determined in *first come first serve* basis. For example, if d is 2, then 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's one. Afterward, it can encounter a bilateral negotiation with Agent C. After carrying out any successful negotiation, agents will update their path accordingly. A number of negotiation sessions might be held until resolving current conflicts. If there are no conflicts left in the current field of view, agents move to their next location in their path. Once an agent reaches its desired destination, it will not encounter a negotiation anymore. The negotiation between agents is carried out according to the proposed token-based negotiation protocol. The details of this protocol and a specific bidding strategy particularly designed for this protocol to tackle the MAPF problem are explained in the following sections.

3.1 Token-based Alternating Offers Protocol (TAOP)

The proposed framework requires agents to engage in negotiation to resolve conflicts in their paths. At a given time, the conflict may occur either between two agents or among multiple agents. When it happens among more than two agents, we can formulate it as multiple bilateral negotiations and consider it a multilateral negotiation. As it may be harder to find a joint agreement, especially when the number of participants is high [4], the proposed approach aims to solve 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-path, agents negotiate on allocating the relevant vertices for certain time steps. Following the previous example illustrated in Figure 1b, Agent A may claim to allocate the vertices at $(3, 2)$ at time $t = 1$, $(3, 3)$ at time $t = 2$ while Agent C may aim to allocate the vertices at $(2, 3)$ at time $t = 1$, $(3, 3)$ at time $t = 2$. Depending on how the negotiation proceeds, they may concede over time and change their request on vertex allocations to come up with an agreement. If agents find an agreement, they are supposed to obey the allocation for the other party. That is, agents are free to change their own 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 to be 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 [5] in which they exchange 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, finding a consensus plays a key role in the context of MAPF. Therefore, agents preferably follow a protocol leading them to reach an agreement. Accordingly, we introduce a novel token-based negotiation protocol namely *Token-based Alternating Offers Protocol* (TAOP) inspired from Monotonic Concession Protocol (MCP) [23] and Unmediated Single Text Protocol (USTP) [16]. According to MCP, agents make simultaneous offers in a way that either they can stick to their previous offer or make a concession. If both parties stick to their previous offers, the negotiation ends without any consensus. Otherwise, agents continue negotiation until reaching an agreement or failing the negotiation. This protocol leads agents to complete the negotiation without setting a predefined deadline. However, there is a high risk of ending up with a failure. In USTP, agents interchangeably are becoming a proposer or voter during the negotiation. Initially, a number of tokens are given to each agent where agents can use those tokens to override other's reject votes. One agent starts with a random offer, and the other agent votes to accept or reject it. If the other agent accepts, it is considered as the most recently accepted bid. 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 as the agreement. Here, the tokens are used to incentivize truthful voting of agents to not manipulate the system by rejecting all offers. Since this protocol is particularly designed for large-scaled 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 enforce the agents to concede over time in a fairway.

Basically, 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 are not allowed to repeat their previous offers unless they pay for them. The protocol assumes that each agent owns a predefined number of tokens, $\mathcal{T}$. Those tokens are used to enable an agent to make one of its previous offers during that negotiation. Different from MCP, agents are not required to make conceding moves. The essential requirement for agents is to make unproposed offers during the negotiation or pay tokens to repeat an offer. In addition to the given offer, agents send an acknowledgement 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 certain time steps and sends an acknowledgement message regarding the usage of 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.
   - accept the received offer and complete the negotiation successfully.
   - makes an offer specifying the vertices allocation for itself that has not been offered by that agent yet and sends the acknowledgement denoting the accumulated usage of its tokens.
   - can repeat one of its previous offers, increase the usage of its tokens by one, and sends the token acknowledgement message.
3. If the agent accepts or ends the negotiation, negotiation is finished. The accepting agent receives tokens amounting to the calculated token usage difference from its opponent, $min(\mathcal{T}_{opp,self} - \mathcal{T}_{self,opp}, 0)$ where $\mathcal{T}_{opp,self}$ and $\mathcal{T}_{self,opp}$ 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 its opponent, it does not receive any tokens. Otherwise, the receiving agent can take any action mentioned in Step 2.

Considering the scenario given in Figure 1b, an example negotiation trace between Agent A and C is illustrated in Figure 2. Agent A initiates the negotiation with its offer P_{A1} requesting to claim the vertices $(3,2)$ for $t=1$ ad $(3,3)$ for $t=2$. Agent C does not accept this offer and makes its own offer specifying the allocation for itself, such as $(2,3)$ and $(2,4)$. Since Agent A insists on its previous offer, it increases its token usage by one. As seen from the example, agents send an acknowledgement message and their offer in each turn. In the fourth round, Agent C accepts Agent A's offer. 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 ($T_{A,C}$-$T_{C,A}$). It is worth to note that the token exchange is performed at the end of the negotiation depending on who accepts the offer. If an agent needs to pay tokens, but it has an insufficient number of tokens, the agreement is not committed (i.e., negotiation fails).

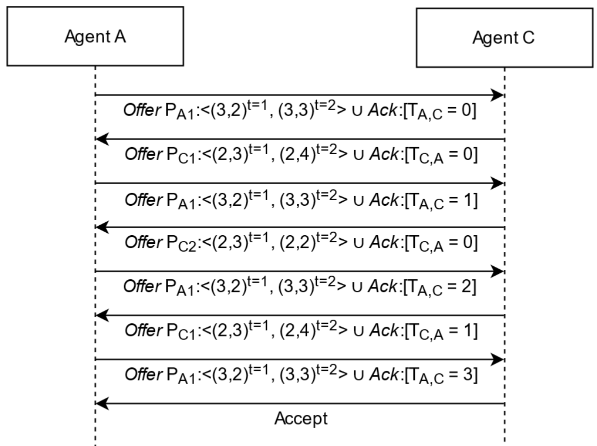


Fig. 2: Example Interaction between Negotiating Agents

3.2 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 [6]. Therefore, there is a need to design a new strategy taking token exchanges into account. Hereby, we propose a negotiation strategy determining when to repeat an offer or to generate a new offer. The proposed strategy, namely *Path-Aware* negotiation strategy, aims to utilize the information available to determine when to insist on its current path. It is worth noting that each agent generates its possible paths leading them to their destination by using A-Star Algorithm in a way that the generated paths would not conflict with the neighbor agents' current path. Afterward, they sort those paths in descending order with respect to their path cost.

Algorithm 1 describes how an agent negotiates according to Path-Aware Negotiation Strategy. At the beginning of the negotiation, the current path in the field of view ($P_{current}$)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 that is 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, $O_{opponent}$ as a constraint while generating the best possible path to the destination. 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 and 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 field of view (Line 7). Accordingly, the agent offers its previous path in the field of view $P_{current}$ (Line 9). Note that agents have to concede if they don't have any tokens left.

Algorithm 1: Negotiation Strategy of Path-Aware Agent

Data:
$T_{remaining}$: Agent's remaining tokens count
$P_{remaining}$: Agent's remaining path to destination
$P_{current}$: Current path in FoV
P_{Space}: Sorted path space
$O_{opponent}$: Opponent offer

1 **if** $|P_{current}| \geq |generatePath(O_{opponent})|$ **then**
2   accept()
3 **else**
4   **if** $T_{remaining} > |P_{remaining}|$ **then**
5         $T_{remaining} --$
6   **else**
7         $P_{current} \leftarrow P_{Space}$.next()
8   **end**
9   offer($P_{current}$)
10 **end**

4 Evaluation

We evaluated the proposed approach empirically from three different perspectives: by comparing its performance with centralized solutions, by comparing the performance of the Path-aware negotiation strategy with a baseline strategy, and by studying the effect of field of view (FoV) in the proposed approach. The following sections will explain our experimental setup and result elaborately.

4.1 Experimental Setup

To inspect the performance of our decentralized approach against centralized approaches to resolve conflicts in MAPF, we make a comparison with two well-known centralized methods, namely Conflict-Based Search (CBS) [24] and CBS with the Weighted Pairwise Dependency Graph Heuristic and Rectangle Reasoning by Multi-Valued Decision Diagrams, named WDG+R in [17]. These centralized solutions are detailed below. We used the code of WDG+R provided by its authors. We removed wait-action (no movement in a time step) from the action space of agents in all solvers to be suitable for the problem definition in Section 2. All experiments were carried on machines with the computing power of 16-Core 3.2 GHz Intel Xeon and 32 GB RAM. Each scenario configuration experimented with 100 different scenarios in no obstacle, 8x8, and 16x16 grid environments.

- **CBS:** Conflict-Based Search (CBS) [24] is a two-level algorithm for centralized and optimal MAPF. At the low-level search, a single path is planned by an optimal shortest-path algorithm, like A*, under given constraints. A constraint is a tuple (i, v, t) where agent a_i is prohibited from occupying vertex v at time step t. At the high-level search, a constraint tree (CT) is operated to resolve conflicts between paths. CT is a binary tree of constraint nodes. Each CT node consists of a set of constraints for each agent. When a conflict is found between two agents, two child nodes are generated. In each child node, one agent in the conflict is prohibited from using 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.
- **WDG+R:** It is one of the recently enhanced variants of CBS and a state-of-art optimal MAPF solver. WDG+R operates smaller CTs by using an admissible heuristic in the high-level search named the Weighted Pairwise Dependency Graph (WDG) Heuristic. WDG represents the pairwise dependencies requiring some cost increase to resolve conflicts. Value of the minimum vertex cover of WDG serves as an admissible heuristic of a lower bound to cost increase to resolve conflicts. Besides, it efficiently resolves the rectangle conflicts by a reasoning technique introduced to CBS in [18]. A rectangle conflict occurs when two locations are required to be taken by both agents simultaneously, which means a certain cost increase to resolve the conflict. These enhancements provide a large factor of speedup compared to CBS.

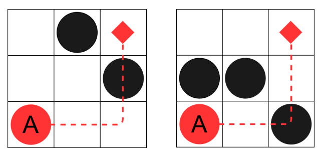
We generated MAPF scenarios from the MAPF benchmark datasets provided by [25]. Table 1 provides the information of experimented MAPF scenarios. We set eight different problem configurations, which are 10, 15, 20, and 25-agent scenarios in an empty 8×8 grid, and 20, 40, 60, and 80-agent scenarios in an empty 16×16 grid. For each configuration, 100 different scenarios have experimented with randomly distributed path lengths between 2 and 14 for 8×8 grid scenarios, and 4 and 24 for 16×16 grid scenarios. We determined the number of agents in the environment to such levels to observe remarkable breakdowns in success rates of CBS and WDG+R, which helps to see which MAPF problem complexity levels these centralized MAPF solution approaches become to fail at. 8×8 grid scenarios are experimented to benchmark CBS specifically and our proposed solution considering the performance evaluation of CBS done in [24]. In 16×16 grid scenarios, we aim to see when the scaling capability of centralized and decentralized solutions are discriminated by changing the number of agents with large increments.

We set the runtime of CBS and WDG+R to 30 minutes 8×8 for grid scenarios and 1 hour for 16×16 grid scenarios in favor of obtaining optimal solutions for a rigorous evaluation of experiments, although CBS benchmarked in 5 minutes by [24] and WDG+R benchmarked in 1 minute by [17]. However, the runtime metric does not represent the success capability of our decentralized solution since it would proceed in real-time. Nevertheless, we limited the simulation runtime of the decentralized MAPF framework. In addition, decentralized solution can fail

Table 1: Scenario Types

Configuration Name	Grid Size	Number of Agents	Initial Path Range
Config-1	8x8	10	2-14
Config-2	8x8	15	2-14
Config-3	8x8	20	2-14
Config-4	8x8	25	2-14
Config-5	16x16	20	4-24
Config-6	16x16	40	4-24
Config-7	16x16	60	4-24
Config-8	16x16	80	4-24

to find a solution, when inactive agents close off movement to destination (Figure 3a), or surround others (Figure 3).



(a) (b)

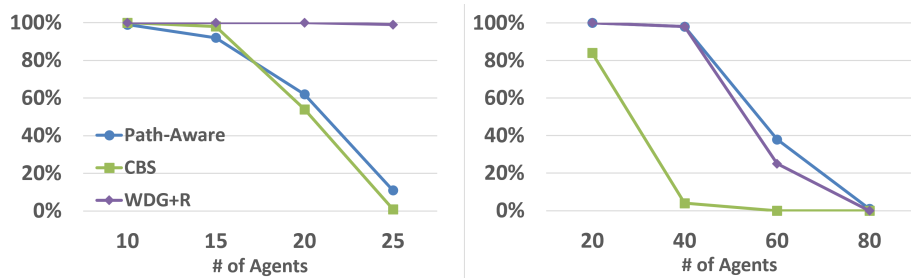
Fig. 3: Agent and Destination Blocked by Agents Reached Destinations

The field of view (FoV) is set to 2 for all agents to experiment with all 8×8 and 16×16 scenarios. Setting FoV to 1 corresponds that agents can be aware of conflicts just a one-time step before, limiting the practicality of negotiation to resolve conflicts. To observe the effect of FoV in our framework's solution performance, we repeat the experiments of Config-6 and Config-7 scenarios, setting FoV to 2, 3, and 4 for all agents. We do not test the effect of field of view in 8×8 grid environment since it is not much practicable to change the range.

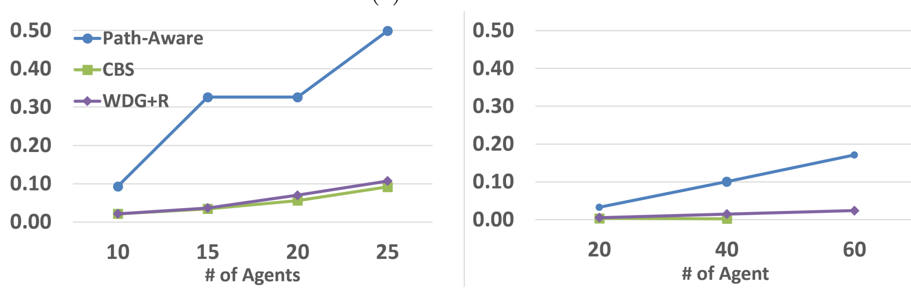
4.2 Experimental Results

Each metric for the results of each solution method is averaged over scenarios solved by itself throughout the evaluations in the following subsections.

Decentralized versus Centralized Approach: Solution rate (R) of the decentralized MAPF framework with Path-aware agents (PA) and the centralized solutions for 8×8 and 16×16 grids are represented in Figure 4a where the left chart corresponds for 8×8 grid results, and the right chart corresponds for 16×16 grid results. Although the decision complexity of agents is essential to measure the framework performance, this basic agent strategy outperforms CBS in 8×8 grid scenarios and also WDG+R in 16×16 grid scenarios in terms of R. However, PA results are not desirable 8×8 grid scenarios when the solution quality is



(a) Solution Rate



(b) Average Normalized Path Difference

Fig. 4: Decentralized versus Centralized Approach Results

considered according to the left chart in Figure 4b. To measure how much extra cost is produced to resolve conflicts in optimal paths, we use a metric named Average Normalized Path Difference (D_{avg}), which is equal to (C_1-C_0)$/k$ where C_0 is the cost of initial paths of all agents and C_1 is the sum of individual path costs (SIC) value attained in a solution. This metric means how much-added cost is yielded to resolve conflicts compared to C_0. D_{avg} gives the information of how much cost increase occurs in which environments for the side of the self-interested agent only consider its own cost valuation based on its initial path cost. 4b shows that PA performs well in the scenarios of the high number of agents in larger maps, which indicates the scalability of the decentralized solution compared to centralized solutions. However, only one scenario of Config-8 was solved by PA because agents cannot negotiate to resolve a conflict caused due to the demonstrated situations in Figure 3. We note that Config-8 scenarios are not evaluated in the following metrics since optimal solutions could not be obtained for them.

As all scenarios are not solved by WDG+R, we do not have the complete information to measure the solution quality of the decentralized solutions since optimal solutions are taken as the basis for it. It is not a health assessment to compare the cases that CBS and WDG+R can solve with the scenarios they cannot solve, but PA solves. For this reason, there is a need for a variable that shows the relationship between solution rates and normalized path differences more dynamically, which is D_{avg}/R. Figure 5 has enabled dynamic changes to be observed in a wide range with D_{avg}/R. Although CBS seems to be more successful than PA in 8×8 grid scenarios in terms of D_{avg}, Figure 5 shows that

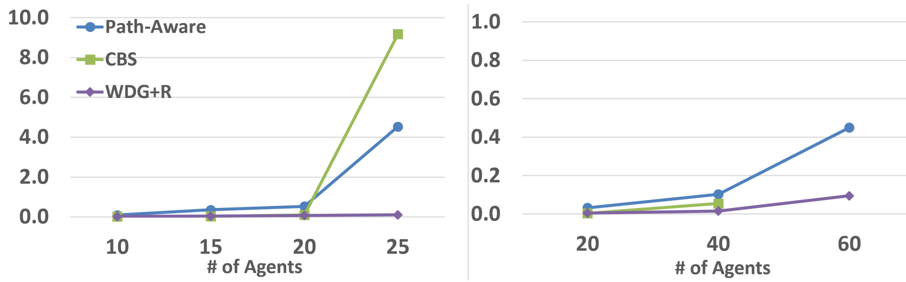


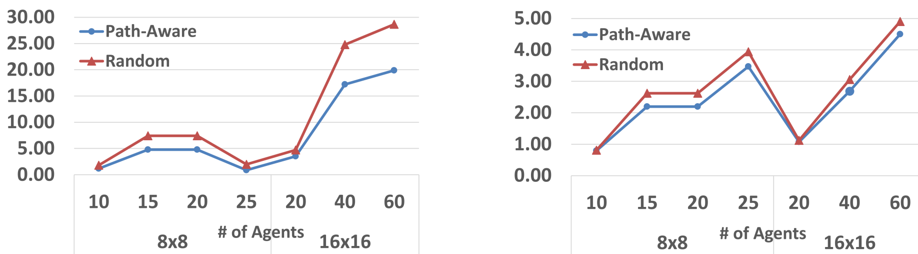
Fig. 5: Average Normalized Path Difference / Solution Rate

Path-Aware agents are more successful when considering the performances in new metric solution rates. It is observed that CBS fails dramatically in scenarios involving 25 Agents. When the 8×8 and 16×16 grid scenarios are compared, it is observed that the performance of the WDG+R remained the same, while the performance of the Path Aware Agents was 10 times better. This result shows that in larger domains, decentralized and intelligent agents such as Path-Aware have less performance difference from optimal solvers and high privacy support.

Effect of Intelligence of Agents: To figure out the effect of intelligence of agents in our framework, a baseline representing a random decision behavior for the negotiation protocol is needed. Therefore, we present a basic decision mechanism adapted by an experimental agent named *Random Agent*. Random Agent accepts its opponent's offer with %50 probability. Otherwise, it repeats its previous offer with %50 probability if it has enough tokens. It generates its offer space exactly in the same way as the Path-aware strategy. The main difference is about accepting and deciding the usage of tokens. We highlight that Random Agent provides a lower bound performance for any prospective self-interested rational agent designed for the proposed decentralized MAPF framework.

Figure 6a shows the total number of token exchanges in a session for both agent types. Negotiation between Path-Aware agents results in less number of token exchanges compared to negotiations of Random agents, which shows that Path-Aware agents insist on their offers only to maintain their own cost balance, whereas Random agents insist or concede randomly. Negotiations between Path-Aware agents reach an agreement faster than random agents in all environments, as seen in Figure 6b. The number of negotiations by Random agents represents a baseline for the negotiation protocol if the agent decides indifferent to counter's bids. So, it can be concluded that when agents behave more analytical, they can reach an agreement faster with our proposed negotiation protocol. Since the solution rates of the decentralized solution with Random agents (RA) and the decentralized solution with Path-Aware agents (PA) is low, we do not seek a trend for the curves in Figure 6.

Effect of Field of View: We experiment with 40 and 60-agent scenarios in 16×16 grid with different FoV values to figure out how the perception and in-



(a) Total Number of Token Exchange

(b) Number of Negotiations/Agent

Fig. 6: Path-Aware Agent versus Random Agent

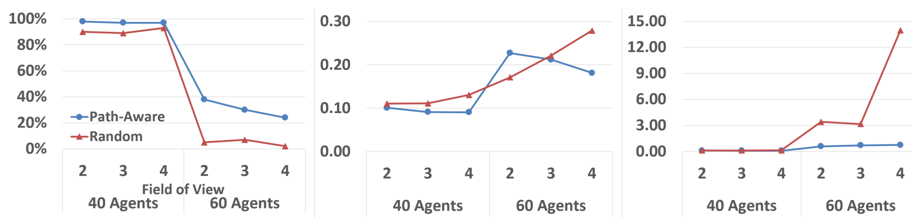


Fig. 7: (a) Solution Rate with FoV, (b) Average Normalized Path Difference with FoV, (c) Average Normalized Path Difference / Solution Rate with FoV

formation broadcast range of agents relate to the solution performance of the decentralized MAPF approach. This relation branches in three aspects, R, D_{avg}, and D_{avg}/R. Figure 7 presents the related curves for PA and RA. Change in FoV of Random agent has no general trend in R in all scenarios, while the increase in FoV decreases R in all scenarios when all agents are Path-Aware. This decrease is 1% for 40-agent scenarios and 14% for 60-agent scenarios, which shows that Path-Aware agents struggle to agree in dense environments when they have a wider FoV. On the other hand, Path-Aware agents can have better paths with wider FoV according to D_{avg} trend in Chart B. Besides, the third chart in Figure 7 shows that the solutions achieved with PA are much better in terms of D_{avg}/R compared to RA solutions. So, it can be concluded that intelligent agents can preserve their own path cost interest by negotiating with others under TAOP. This output is expected because Path-Aware agents change their bidding behavior based on current cost analysis (of remaining path length and the remaining token amount at the negotiation time). This evaluation becomes more useful if more information about the environment is used.

In large FoV cases, Random agents tend to perform worse as their decision-making process is stochastic. As the accepted offers are registered, having a large FoV thus results in more constraints for agents, which reduces path search space. Path-Aware Agents provide solutions to these problems both in acceptance and by spending their tokens correctly. On the other hand, Random Agents do not have a specific strategy other than entering into a negotiation and accepting long paths randomly in the face of these problems. This situation is reflected in the average number of negotiations per agent. When looking at the difference of the average negotiation per agent between FoV 2 and FoV 3, Random agents (2.17) are 2.59 times more than Path-Aware agents (0.83).When the same variable is examined between FoV 3 and Fov 4, Random agents (2.11) increased 5.90 times more than Path-Aware agents (0.35).

5 Related Work

We classify approaches to resolve conflicts in MAPF based on two factors: the centralization of solution mechanism and cooperation of agents. Centralized solution approaches to pathfinding of cooperative agents provide optimal plans [11]. If a trusted center with the information of all agents moving in a certain area and the ability to command all of them is not available, negotiation can be used for a conflict resolution mechanism [1, 12, 22, 21, 27]. One negotiation approach to allocating resources to multiple parties is Combinatorial Auction (CA). To resolve conflicts between self-interested agents in an environment, Amir *et al.* reduce MAPF problem to CA and implements iBundle, an iterative CA algorithm [20], for MAPF [1]. Self-interested agents might not provide their own utilization truthfully to the auctioneer. Considering this aspect of the auction, Amir *et al.* propose Vickrey-Clarke-Groves (VCG) auction for MAPF, a strategy-proof auction mechanism for manipulation attempts by the agents. In this iBundle auction, the auctioneer is exposed to a computational burden as agents submit their all-desirable bundles, which requires even impractical auction time. Addressing this limitation of iBundle, Gautier *et al.* introduce an auction design that allows agents to submit a limited number of bundles so that a feasible allocation is more likely to be found, and the auction terminates in fewer time [12]. They also provide a further auction solution procedure applied if a feasible allocation to submitted bundles is not found. Auctioneer finds some feasible allocations using a MAPF solver, and it evaluates them to maximize social welfare using its privileged knowledge gained in the bidding. Then it proposes the most valuable allocation to the agents. Auction ends when all agents accept one allocation; otherwise, the auctioneer updates allocation values based on rejecting agents' bids and proposes the best new allocation.

Key challenges of addressing MAPF problem within the decentralized method can be summarized as establishing a framework for agents to use while interacting with the environment, defining an interaction protocol between agents, and designing agents that are able to reach a solution [7, 9, 2]. In their paper Purwin *et al.*, proposes a decentralized framework where agents allocate portions of the environment in which they move. Similarly, the framework proposed in this paper also allows agents to exchange vertex information while trying to allocate a conflict-free path. However, their negotiation protocol resolves the conflicts in one shot, whereas the protocol proposed in this paper allows agents to engage in negotiation sessions in length. Sujit *et al.* focuses on resolving a task allocation problem in their work, using a multilateral negotiation structure [27]. Agents only utilize the presented token structure 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 this paper, tokens are treated as a limitation in making repeated offers. The work of Pritchett *et al.* defines a simultaneous bilateral negotiation structure to resolve conflicts in air traffic control [21]. Their work defines a structure where agents negotiate over the trajectories that they will take. In each round of the negotiation session, the cost of all offers increases until an agreement is reached. While due to the nature of the environment, this forces agents to concede over time, the protocol proposed in this paper defines a hard constraint on how many times an agent can refuse an opponent's offer. Inotsume *et al.* demonstrates a negotiation-based approach to MAPF from the perspective of an operator [14]. In their setup, each agent tries to maximize their utility by completing tasks, reaching a certain destination in a shared space. An area manager interface manages this shared space, and each agent is expected to submit their desired paths to the area manager before they begin their movement. Here, the area manager is the entity that checks whether each path conflicts with already reserved paths or prohibited locations. As they utilize a path reservation system managed by a non-agent entity, this setup deviates from the proposed decentralized approach. Additionally, they propose a trading structure for their paths, which can correspond to token exchange. They value these tokens equivalent to each edge traversal, whereas our study values tokens in a completely different economy. Nevertheless, both systems focus on resolving path conflicts using negotiation mechanisms.

6 Conclusion

This paper addresses how self-interested agents can coordinate in a grid environment to reach their destination without any collision and proposes solving the conflicts on the paths by means of bilateral negotiations. Accordingly, we propose a novel negotiation protocol and a compatible path-aware negotiation strategy. The proposed approach enables agents to optimize their paths in real time without sharing their complete path information with everyone. This problem is harder to solve especially when the grid size gets larger and higher-density (i.e., large number of agents and long paths per agent). In such cases, the proposed approach has an edge over centralized approaches. The analysis of experimental evaluation showed that Path-aware negotiation approach finds reasonably good solutions in most of the cases and it performed better on aforementioned challenging scenarios than the state-of-the-art centralized solution such as CBS and WDG+R. As future work, we are planning to extend our approach by adopting multilateral negotiation instead of multiple consecutive bilateral negotiations and to compare its performance with the current approach. In the current work, agents should move constantly in line with their path. However, enabling agents to wait for any time step (i.e, no move action) may lead agents to discover new solutions while it increases the search space dramatically. We think of incorporating wait action to our framework as well as other variants of actions. Furthermore, it would be interesting to design more sophisticated negotiation agents thinking ahead when they use their tokens.

Bibliography

[1] Ofra Amir, Guni Sharon, and Roni Stern. Multi-agent pathfinding as a combinatorial auction. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, page 2003–2009. AAAI Press, 2015.

[2] Dor Atzmon, Yonathan Zax, Einat Kivity, Lidor Avitan, J. Morag, and Ariel Felner. Generalizing multi-agent path finding for heterogeneous agents. In *SOCS*, 2020.

[3] Reyhan Aydoğan, Tim Baarslag, Katsuhide Fujita, Johnathan Mell, Jonathan Gratch, Dave de Jonge, Yasser Mohammad, Shinji Nakadai, Satoshi Morinaga, Hirotaka Osawa, Claus Aranha, and Catholijn M. Jonker. Challenges and main results of the automated negotiating agents competition (ANAC) 2019. In Nick Bassiliades, Georgios Chalkiadakis, and Dave de Jonge, editors, *Multi-Agent Systems and Agreement Technologies*, pages 366–381, Cham, 2020. Springer International Publishing.

[4] Reyhan Aydoğan, Koen V. Hindriks, and Catholijn M. Jonker. *Multilateral Mediated Negotiation Protocols with Feedback*, pages 43–59. Springer Japan, Tokyo, 2014.

[5] Reyhan Aydoğan, David Festen, Koen V. Hindriks, and Catholijn M. Jonker. Alternating offers protocols for multilateral negotiation. *Studies in Computational Intelligence*, 674:153–167, January 2017.

[6] Tim Baarslag, Enrico H. Gerding, Reyhan Aydoğan, and M.C. Schraefel. 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)*, volume 2, pages 190–197, Dec 2015.

[7] Subhrajit Bhattacharya, Maxim Likhachev, and Vijay Kumar. Multi-agent path planning with multiple tasks and distance constraints. In *2010 IEEE International Conference on Robotics and Automation*, pages 953–959. IEEE, 2010.

[8] Vishnu R. Desaraju and Jonathan P. How. Decentralized path planning for multi-agent teams with complex constraints. *Auton. Robots*, 32(4):385–403, May 2012.

[9] Esra Erdem, Doga G. Kisa, Umut Oztok, and Peter Schüller. A general formal framework for pathfinding problems with multiple agents. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*, AAAI'13, page 290–296. AAAI Press, 2013.

[10] M. Erdmann and T. Lozano-Perez. On multiple moving objects. In *Proceedings. 1986 IEEE International Conference on Robotics and Automation*, volume 3, pages 1419–1424, 1986.

[11] Ariel Felner, R. Stern, S. E. Shimony, Eli Boyarski, Meir Goldenberg, G. Sharon, Nathan R Sturtevant, G. Wagner, and Pavel Surynek. Search-based optimal solvers for the multi-agent pathfinding problem: Summary and challenges. In *SOCS*, 2017.

[12] Anna Gautier, Bruno Lacerda, Nick Hawes, and Michael Wooldridge. Negotiated path planning for non-cooperative multi-robot systems. In *IJCAI*, 2020.

[13] E. Hoz, J. M. Giménez-Guzmán, Ivan Marsá-Maestre, and D. Orden. Automated negotiation for resource assignment in wireless surveillance sensor networks. *Sensors (Basel, Switzerland)*, 15:29547 – 29568, 2015.

[14] Hiroki Inotsume, Aayush Aggarwal, Ryota Higa, and Shinji Nakadai. Path negotiation for self-interested multirobot vehicles in shared space. In *Proceedings of the International Conference on Intelligent Robots and Systems*, pages 11587–11594, Las Vegas, NV, USA, 2020. IEEE.

[15] Dave Jonge, Filippo Bistaffa, and Jordi Levy. A heuristic algorithm for multi-agent vehicle routing with automated negotiation. In *20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021)*, 05 2021.

[16] Mark Klein, Peyman Faratin, Hiroki Sayama, and Yaneer Bar-Yam. Protocols for negotiating complex contracts. *IEEE Intelligent Systems*, 18:32–38, 12 2003.

[17] Jiaoyang Li, Ariel Felner, Eli Boyarski, Hang Ma, and Sven Koenig. Improved heuristics for multi-agent path finding with conflict-based search. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 442–449. International Joint Conferences on Artificial Intelligence Organization, 7 2019.

[18] Jiaoyang Li, Daniel Harabor, Peter J. Stuckey, Hang Ma, and Sven Koenig. Symmetry-breaking constraints for grid-based multi-agent path finding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6087–6095, Jul. 2019.

[19] Ivan Marsá-Maestre, M. Klein, C. Jonker, and Reyhan Aydogan. From problems to protocols: Towards a negotiation handbook. *Decis. Support Syst.*, 60:39–54, 2014.

[20] David C. Parkes. Ibundle: An efficient ascending price bundle auction. In *Proceedings of the 1st ACM Conference on Electronic Commerce*, EC '99, page 148–157, New York, NY, USA, 1999. Association for Computing Machinery.

[21] A. Pritchett and Antoine Genton. Negotiated decentralized aircraft conflict resolution. *IEEE Transactions on Intelligent Transportation Systems*, 19:81–91, 2018.

[22] O. Purwin, R. D'Andrea, and J. Lee. Theory and implementation of path planning by negotiation for decentralized agents. *Robotics Auton. Syst.*, 56:422–436, 2008.

[23] Jeffrey S. Rosenschein and Gilad Zlotkin. *Rules of Encounter: Designing Conventions for Automated Negotiation among Computers.* MIT Press, Cambridge, MA, USA, 1994.

[24] G. Sharon, R. Stern, Ariel Felner, and Nathan R Sturtevant. Conflict-based search for optimal multi-agent pathfinding. *Artif. Intell.*, 219:40–66, 2012.

[25] Roni Stern. *Multi-Agent Path Finding – An Overview*, pages 96–115. Springer International Publishing, Cham, 2019.

[26] Roni Stern, Nathan Sturtevant, Ariel Felner, Sven Koenig, Hang Ma, Thayne T. Walker, Jiaoyang Li, Dor Atzmon, Liron Cohen, T. K. Satish Kumar, Eli Boyarski, and Roman Barták. Multi-agent pathfinding: Definitions, variants, and benchmarks. *CoRR*, abs/1906.08291, 2019.

[27] P. B. Sujit, A. Sinha, and D. Ghose. Multiple uav task allocation using negotiation. In *AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, AAMAS '06, page 471–478, New York, NY, USA, 2006. Association for Computing Machinery.