Negotiated Path Planning for Non-Cooperative Multi-Robot Systems

Anna Gautier
University of Oxford
Oxford, United Kingdom
anna.gautier@eng.ox.ac.uk

Alex Stephens
University of Oxford
Oxford, United Kingdom
stephens@robots.ox.ac.uk

Bruno Lacerda
University of Oxford
Oxford, United Kingdom
bruno@robots.ox.ac.uk

Nick Hawes
University of Oxford
Oxford, United Kingdom
nickh@robots.ox.ac.uk

Michael Wooldridge
University of Oxford
Oxford, United Kingdom
mjw@cs.ox.ac.uk

ABSTRACT

As autonomous systems are deployed at a large scale in both public and private spaces, robots owned and operated by competing organisations will be required to interact. Interactions in such settings will be inherently *non-cooperative*. In this paper, we address the problem of non-cooperative multi-agent path finding. We design an auction mechanism that allows a group of agents to reach their goals whilst minimising the total cost of the system. In particular, we aim to design a mechanism such that rational agents are incentivised to participate. Our privileged knowledge auction consists of a modified combinatorial Vickrey-Clarke-Groves auction. Our approach limits the initial number of bids in the Vickrey-Clarke-Groves auction, then uses the privileged knowledge of the auctioneer to identify and solve path conflicts. In order to maintain agent autonomy in the non-cooperative system, individual agents are provided with final say over paths. The mechanism provides a heuristic method to maximise social welfare whilst remaining computationally efficient. We also consider single-agent bid generation and propose a similarity metric to use in dissimilar shortest path generation. We then show this bid generation method increases the success likelihood of both the limited-bid VCG auction and our novel approach on synthetic data. Our experiments with synthetic data outperform existing work on the non-cooperative problem.

CCS CONCEPTS

• **Computing methodologies** → **Multi-agent planning**; • **Theory of computation** → *Algorithmic mechanism design.*

KEYWORDS

Multi-agent systems; Multi-agent path finding; Auctions

ACM Reference Format:
Anna Gautier, Alex Stephens, Bruno Lacerda, Nick Hawes, and Michael Wooldridge. 2022. Negotiated Path Planning for Non-Cooperative Multi-Robot Systems. In *Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), Online, May 9–13, 2022*, IFAAMAS, 9 pages.

1 INTRODUCTION

When multiple autonomous systems interact in shared spaces, the systems can act individually and risk disrupting one another, or they can negotiate to avoid conflict. Any interface designed to negotiate between them must allow each system to achieve as many of their goals as possible, but must still respect their autonomy. Consider a superstore, with aisles of groceries, clothing, and electronics. The store does not have the technology to build a fleet of robots, so they contract out individual tasks. One company might be hired to deploy a robot to move around the store examining shelves to track inventory. Another company might be hired to deploy cleaning robots. These robots have misaligned incentives; while they both have to interact in the space, they only care about their own contract with the store, and so their only goal is to complete their own task. They may be penalised for failing to complete their tasks either with a built-in monetary penalty or the eventual loss of their contract. As a result, this system is *non-cooperative*. This situation is ill-suited to centralisation, because agents need autonomy over their own decision making and path planning.

But without a centralised system to direct agents to coordinate while they complete their goals, conflicts are likely. Conflicts occur when multiple robots try to traverse the same area at the same time. Multi-agent path finding (MAPF) focuses on preventing these types of conflicts. Though these robots have collision avoidance for non-stationary objects (like humans), relying on collision avoidance for robot-robot interaction places an unnecessary burden on low-level controllers and can cause significant delays or even collisions [32]. In this paper we consider MAPF where all agents have the ability to make their own choices and are not controlled by a central system. Agents could be indifferent to the success of other agents or may even want to stop other agents from achieving their own goals. Additionally, we assume all agents are *rational*: all agents are interested in maximising their own value given imposed constraints.

While multi-robot path finding problems [31] are well studied, very little research to date has considered the non-cooperative case. One notable exception is the work of Amir et al. [1], which introduced a mapping between MAPF and combinatorial auctions. Furthermore, they discuss how the use of combinatorial Vickrey-Clarke-Groves (VCG) auctions [5, 11, 34] can provide a mechanism robust to manipulation attempts by strategic, possibly self-interested agents. They then propose the use of *iBundle* [26] to find

solutions to the MAPF problem. iBundle is an iterative combinatorial auction algorithm which, under certain conditions, provides equivalent solutions to VCG. However, iBundle has a number of limitations. First, iBundle relies on decentralised conflict resolution such that agents must continue to submit and value more paths if the price of their high value paths become too large due to conflicts. As a result, iBundle requires agents to have access to an ordered list of all possible single solutions ranked by descending value. Depending on how agents assign value to different solutions, this could require agents to evaluate all possible solutions, or in the case where value functions are tied to shortest paths, this requires the agents to have the ability to access the next shortest path at any given time. Second, the bidding strategy that agents are expected to take will slow down iBundle in the MAPF setting. An agent's myopic best response strategy to the iBundle auction, i.e., the bidding strategy which makes the most sense for them to execute, requires them to submit all paths of equal length at the same time. Because there can be many paths with the same value, agents are incentivised to submit a large set of bids at once, and this detracts from the iterative advantage of iBundle and can lead to a high computation time. Finally, because iBundle finds the optimal allocation, and because the winner determination subproblem of VCG is NP-complete [8], iBundle is also NP-complete. All three of these issues taken together result in a prohibitively large worst-case computation time.

In this paper, we build on [1] and tackle the limitations listed above. We do this by designing a mechanism for non-cooperative path planning that exploits centralised conflict resolution via the *privileged knowledge* it obtains from the initial agent bids. This heuristic mechanism is designed to optimise for social welfare while incentivising agents to participate. Our mechanism is tailored for the MAPF problem, allowing agents to submit a smaller number of paths to the auction, which helps us overcome the limitations of iBundle. The auctioneer takes on the burden of removing conflicts, and uses a method specific to MAPF to do so. We also consider the problem of single-agent bid generation, and propose an adaptation of the dissimilar path search algorithm from Jeong and Kim [15]. Our adaptation modifies the similarity metric to better suit a MAPF context by considering both spatial and temporal similarity. Because agents submit a smaller number of bids, using a dissimilar path algorithm allows agents to provide the auctioneer with a wider range of possibilities, which makes a conflict-free solution in the initial stage of the auction more likely. We analyse our mechanism and single-agent bid generation on two synthetic domains: a model of a warehouse and maps from *Dragon Age: Origins* [31]. Because our mechanism relaxes guarantees and optimality to improve computation time, it outperforms iBundle in almost all cases.

2 RELATED WORK

Cooperative MAPF is widely studied, and several algorithms have been proposed for it, e.g. [29, 30]. The problem has been extended in a variety of ways, for example to consider uncertainty [28, 35] or kinematic constraints [13, 36]. For more details on cooperative MAPF, see [31].

Auctions are a class of mechanisms by which resources are distributed among agents. Auctioning approaches have been widely used in robotics, particularly for cooperative coordination of teams of robots. Agents request resources from the auctioneer and an auctioneer distributes resources based on those requests, often for a price. Early work on this topic used combinatorial auctions [14] and first-price one-round auctions [10] to distribute tasks across a set of agents. Later, in [19], a sequential single item (SSI) auctioning mechanism was proposed for multi-robot routing and task allocation. This approach combines the advantages of parallel single-item auctions and combinatorial auctions, achieving good quality task allocation with low computational effort [17]. SSI auctioning has also been extended in [25] to handle temporal constraints and in [22] to handle precedence constraints. From the perspective of the agents participating in SSI auctions, [33] investigated the problem of generating different bids taking into account different global objectives to be achieved. There has also been work on using auctions to distribute parts of a global task across a team of robots, in the context of planning under uncertainty. [4] does this considering role policies for partially observable Markov decision processes, whilst [27] considers auctioning subtasks such that a global linear temporal logic task is achieved by the team. In contrast to these works, we use auctioning to allocate a set of shared resources (more specifically, points in time and space) across a set robots. Furthermore, in our work the robots are *competing* for the use of these resources rather than *cooperating* to achieve a set of tasks.

There are examples of game theoretic non-cooperative planning algorithms, but they are ill-suited to our scenario. [2] presents a market-based approach for the management of shared resources for a multi-robot team. However, they assume loosely-coupled problems, i.e., problems that can be modelled using few interactions between agents. This is not the case in our work, where all locations yield a potential conflict between the robots. Stochastic games are a model for non-cooperative multi-agent interaction, and have been applied to reinforcement learning [20] and planning [16, 24]. Stochastic games have been adapted to deal with a wide array of scenarios, but rely on a large amount of shared knowledge. This can not be assumed in our scenario. [3] introduce a game theoretic model that can be adopted to path planning via STRIPS, but it assumes that agents are willing to form coalitions.

The closest related work to ours is Amir et al. [1]. This was the first application of mechanism design to non-cooperative path planning. In particular, they reduce MAPF to a combinatorial auction. They then demonstrate how the VCG auction mechanism (see Section 3.2) can be used in MAPF with strategic agents. They provide experimental results using iBundle, an iterative combinatorial auction with equivalent solutions to VCG [26]. We build on this work, introducing a mechanism that allows us to find solutions with less computational effort, as shown in Section 6.

In order to carry out an auction with space as a resource, agents need to be able to submit bids on paths to the auctioneer, and doing so requires that they evaluate how much they value certain paths. One way agents can find and evaluate multiple path options is through a *k-shortest path algorithm*. k-shortest path algorithms can search for *simple paths*, i.e those without loops, as in [37]. Alternatively, they can search for all paths, including ones with loops and self-loops as in [9]. However, doing so is often very time consuming as it greatly increases the number of possible paths. Agents can also generate paths with some specific attribute, e.g. [15] aims to generate a class of spatially dissimilar paths.

3 PRELIMINARIES

MAPF. In MAPF, agents act on a graph $\mathcal{G} = (V, E)$, where V is a set of vertices and $E \subseteq V \times V$ is a set of edges. We assume each vertex $x \in V$ represents coordinates in an environment, i.e., $x \in \mathbb{R}^2$. Each of n agents starts at their own unique start vertex, and at each discrete timestep agents can wait in their current vertex or travel to another vertex that is connected to their current position by some edge in E. Each agent has a unique goal location, and MAPF aims to find *paths* for each agent to get to their required goal, such that no agents are in *conflict*. We denote the set of all finite sequences of elements of V as V^*, and define a path for an agent as a sequence of vertices $x_1 x_2 \dots x_\ell \in V^*$ where $(x_d, x_{d+1}) \in E$ for all $d \in \{1, \dots, \ell - 1\}$. While many different types of conflict have been explored in the MAPF literature, in this paper we consider vertex conflicts. A *vertex conflict* occurs when two agents are at the same vertex at the same time. We consider the MAPF problem of optimising for *flowtime* or *sum-of-costs*, i.e., the sum of the arrival times of all agents at their goal locations.

Combinatorial Auctions. A combinatorial auction contains a set of agents $\mathcal{I} = \{1, 2, \dots, n\}$ and a set of items $\mathcal{Z} = \{z_1, z_2, \dots, z_m\}$. Each subset of items $A \in 2^{\mathcal{Z}}$ is known as a *bundle*. Each agent i is rational, so they are assumed to have a well-defined value for every bundle $A \in 2^{\mathcal{Z}}$, determined by function $v_i : 2^{\mathcal{Z}} \to \mathbb{R}$. The agents are asked to report a *bid* $b_i(A) \in \mathbb{R}$ to the auctioneer for every subset $A \in 2^{\mathcal{Z}}$. Ideally, this would be their value for that bundle, but rational agents are not necessarily truthful; if it is in their best interest to lie, they will. More specifically, each agent has a strategy $\sigma_i : \mathbb{R} \to \mathbb{R}$ that transforms their values to bids. Each agent's bid is uniquely defined by $b_i(A) := \sigma_i(v_i(A))$ for all $A \in 2^{\mathcal{Z}}$. The auctioneer must use these bids to decide which items are distributed to which agent. We call a set $\{S_j\}_{j \in \mathcal{I}}$ an *allocation*, where each $S_j \in 2^{\mathcal{Z}}$ represents the bundles allocated to agent j. In the remainder of the paper, we will use $\{S_j\}$ as shorthand for $\{S_j\}_{j \in \mathcal{I}}$. A *feasible allocation* $\{S_j\}$ is one where each agent i is allocated a bundle of items $S_i \in 2^{\mathcal{Z}}$ such that $S_i \cap S_k = \emptyset$ for all agents $k \neq i$. We denote the set of all feasible allocations by Γ. The job of the auctioneer is to define a function that maps bids to outcomes. An outcome consists of a feasible allocation $\{S_j\}$ and a set of prices $\{p_j\}$, which each agent must pay to the auctioneer. The auctioneer chooses some optimisation function to consider while determining the allocation, such as maximising *utility* or revenue.

3.1 MAPF as a Combinatorial Auction

We now describe how a MAPF problem can be solved using a combinatorial auction. The overall idea is considering *paths as bundles*. We start by noting that a path $w = x_1 x_2 \dots x_\ell$ can be interpreted as a bundle of vertex-timestep pairs, i.e., $\{(x_1, 1), (x_2, 2), \dots, (x_\ell, l)\} \subseteq V \times \mathbb{N}$. Thus, for notational convenience, for the remainder of the paper we will denote paths interchangeably as either bundles or sequences, i.e., we will denote a l-length path for agent i as $S_i = \{(x_1, 1), (x_2, 2), \dots, (x_\ell, l)\} \subseteq V \times \mathbb{N}$ or as $w_i = x_1 x_2 \dots x_\ell \in V^*$. Agents value these bundles based on how effective the corresponding path is in achieving their goal and what cost it takes them to traverse that path. When optimising for flowtime, path cost is equivalent to path length if agents are moving at a constant speed.

Table 1: A high level overview of the parallel between MAPF and combinatorial auctions (CA) adapted from [1].

MAPF	CA
Vertex-time pair	Item
Path (set of items)	Bundle
Path cost	Valuation function
Minimal sum of path costs	Maximal social welfare

We define the best MAPF solution as one in which the sum of agents' costs is as low as possible. The minimum cost allocation of a MAPF problem is equivalent to the maximum value allocation of a combinatorial auction. Thus, maximising the sum of all values over all agents, known as the *social welfare*, results in a solution to the MAPF problem in which as many agents reach their goals with as little cost as possible. A high-level overview of the parallel between MAPF and combinatorial auctions can be seen in Table 1. Determining feasible allocations via a combinatorial auction ensures that agents do not enter into conflicts. If two agents' bundles S_i and S_k have no intersection, then they will never be at the same vertex at the same time.

3.2 Combinatorial VCG Auction

Framing non-cooperative path planning as a combinatorial auction leads us to the problem of designing an auction mechanism that maximises social welfare, i.e., the sum of all valuations. However the auctioneer only has access to the bids the agents report, not their true valuation for the items. Therefore the auctioneer cannot naively sum up the bids of agents to determine the social welfare of an allocation. Doing so is highly susceptible to manipulation by agents. *Incentive compatible mechanisms* are a class of auctions that solve this problem. A mechanism is incentive compatible if every agent cannot benefit from misreporting their valuation. More formally, this means that all agents' strategies are defined by $\sigma_i(v_i) = v_i$. Because, by definition, $b_i := \sigma_i(v_i)$ in incentive compatible auctions, the auctioneer can assume that each agent's valuation for a bundle is equivalent to their bid.Then, maximising the total utility becomes equivalent to maximising the total of the bids. In combinatorial auctions, only optimal solvers can be incentive compatible [23]. Heuristic methods like the one we present in Section 4 cannot be incentive compatible, though can be difficult to manipulate. One combinatorial auction that satisfies the property of incentive compatibility is the combinatorial VCG auction, named after ideas combined from [5, 11, 34]. The descriptions in this section are based on [6]. In VCG the auctioneer allocates bundles based on the *welfare maximising* outcome. An allocation $\{S_j\} \in \Gamma$ is welfare maximising if, for every feasible allocation $\{T_j\} \in \Gamma$,

$$\sum_{i \in \mathcal{I}} b_i(S_i) \geq \sum_{i \in \mathcal{I}} b_i(T_i). \tag{1}$$

In VCG, prices p_i are set as follows. Let $\{S_j\} \in \Gamma$ be the welfare maximising outcome and $\{S_j^{-i}\} \in \Gamma$ be the welfare maximising outcome if agent i was not involved in the auction, i.e., agent i is allocated the empty set. Then,

$$p_i = \sum_{k \in \mathcal{I} \setminus \{i\}} b_k(S_k^{-i}) \; - \sum_{k \in \mathcal{I} \setminus \{i\}} b_k(S_k). \tag{2}$$

This payment represents the amount that agent i has perturbed the system, as agent i pays the difference between the total welfare of all other agents if they had not been in the auction minus the welfare of all other agents when they are included in the auction. Note that if an agent's presence in the system does not change anyone else's path, or only changes another agent's path to one of equal value, they pay nothing. This can be thought of as a way to ensure agents don't take up desirable space in the map (like vertices with many edges, or vertices that connect otherwise unconnected portions of the graph) unless it has high value to them. Finally, combinatorial VCG auctions are *individually rational*, which ensures that agents are willing to participate. A mechanism is individually rational if each agent is better off participating in the auction than they would be otherwise [18].

Unfortunately, determining the welfare maximising outcome for VCG is NP-complete [8], so solving this problem optimally is time consuming. Additionally, using combinatorial VCG auctions for MAPF require agents to give a value for every possible path ($O(2^{|V|\times h}$ for $|V|$ vertices and h timesteps). This puts an unrealistic computational burden on the agents. The iterative algorithm iBundle allows agents to value less paths, but requires the ability to list paths in value order. Another limitation comes from the strategy that agents are expected to use in the iBundle auction. In an iterative auction like iBundle, agents' strategies determine what bids to make at each iteration of the auction. Parkes [26] show that the *myopic best response strategy* for iBundle (i.e., the strategy agents use when considering only the outcome of the current iteration) requires agents to submit all bids with equivalent value at the same time. But the nature of the MAPF problem means that many paths will have the same value, and thus in practice agents bidding in iBundle will still be required to submit a large number of bids at each iteration of the iBundle algorithm.

4 PRIVILEGED KNOWLEDGE AUCTION

We now introduce a new mechanism called the privileged knowledge auction (PKA) which removes the problem of enumerating so many bids, by allowing agents to place a fixed number of bids. Our aim is to create a mechanism to solve the non-cooperative MAPF problem. To be a MAPF solution, a set of paths must avoid conflicts and allow agents to reach their goals. To be a non-cooperative solution, agents must have autonomy over their own paths and the system should be difficult to manipulate by strategic agents. Figure 1 provides an overview of our proposed mechanism. Agents submit bids and the auctioneer carries out a modified VCG auction (Section 4.1). Then the auctioneer uses knowledge from the VCG auction, which we refer to as *privileged knowledge*, to find a solution outside of the submitted proposals if necessary (Section 4.2). Finally, the auctioneer gives agents autonomy over what bids are chosen through a descending auction (Section 4.3).

4.1 Initial Bidding and VCG Auctioning

Agents propose timed bundles in the form $\{(x_1, 1), (x_2, 2), ...(x_\ell, \ell)\}$. In the first step of the mechanism, each agent proposes a fixed number of paths to the auctioneer. In practice, this number will typically be small to cope with the scalability issues of VCG and is chosen as a design parameter. The auctioneer then searches for a potential

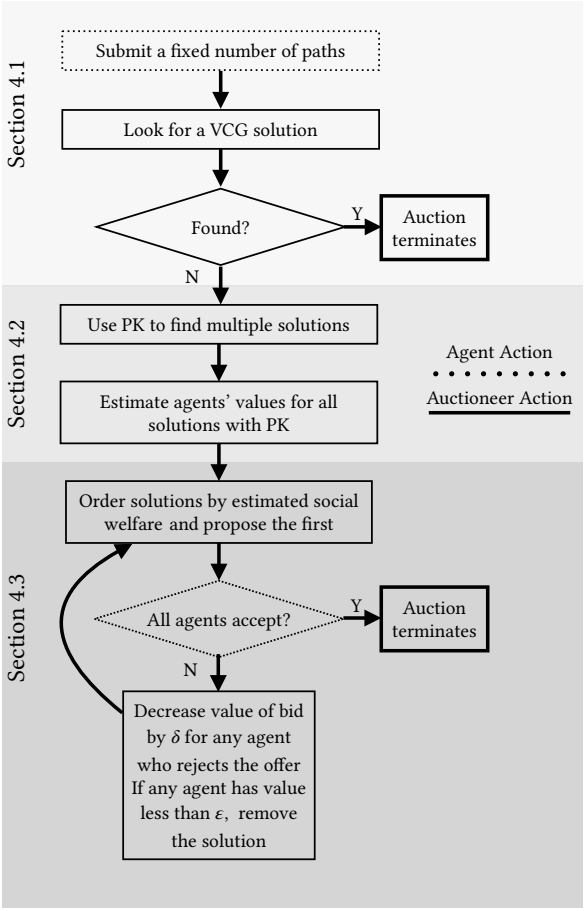

Figure 1: A high level overview of our PKA protocol.

feasible allocation using the same winner determination and pricing as combinatorial VCG. At this stage of the auction, agents can only be allocated bundles which they themselves proposed. As a result, all allocations will correspond to actual paths. If no solution is found in which every agent is allocated a path, we move into the second part of our protocol.

4.2 Deconflicting

In the second part of PKA, the auctioneer uses the privileged knowledge gained in the bidding stage to find a better solution. If the auctioneer cannot find a solution given the proposed paths supplied by the agents, they must instead generate their own, conflict-free solution. In order to preserve the agents' autonomy, the auctioneer generates multiple alternate conflict-free solutions from which the agents can choose. Before we discuss the incentive-aware process that the auctioneer uses to choose between possible solutions, we address how those solutions can be generated. Our mechanism is agnostic to the method used to find such solutions, the suitability of each possible method being tied with the particular environment. In situations where the auctioneer believes value is closely tied to the shortest path to the goal, classical cooperative MAPF algorithms

are a good choice. Any MAPF algorithm (or set of MAPF algorithms) which can be designed to return multiple possible solutions is suitable. In our experiments, we used hierarchical cooperative A* (HCA*) [30]. HCA* is a quick, suboptimal, solution algorithm where agents plan their paths sequentially in a graph extended to include both space and time. After each agent plans their path, the locations they pass through on the graph are removed from the graph at that particular time, and the next agent is allowed to plan their path. Agents are allowed to plan their paths in different orders to produce different solutions. We leave comparing different solution methods to future work. Ideally, we desire at least two solutions from the MAPF solver but if there is only one possible solution, the auctioneer will proceed with only that one solution.

Assuming that the auctioneer has a set of alternate solutions $\mathcal{AS} = \{\{S_j^1\}, ..., \{S_j^m\}\}$, it is necessary to order them from most desirable to least desirable. The goal of the mechanism is to elicit truthful values, so while we are unable to extrapolate the exact social welfare of these solutions with the partial value information obtained in the first stage of the mechanism, we can develop a heuristic, and then use that to elicit true preferences by asking the agents for more information (Section 4.3). To approximate the social welfare of a specific alternate solution $\{S_j\}$, we need to approximate the value of each path $S_i \in \{S_j\}$ allocated to each agent i. We define the approximate value of a path by how different it is from a path that we know agent i's true value of, i.e., a path that agent i bid on in the first stage of the mechanism. In particular we choose the path $\tilde{S}_i$ to be the 'closest' path to our solution path S_i out of all the paths that agent i bid on in the first stage of the mechanism. Our heuristic for 'closeness' is defined by the following distance function between two bundles $S, T \in 2^{\mathcal{Z}}$:

$$d(S,T) = \lambda||S - T|| + (1-\lambda)|t_S - t_T|. \tag{3}$$

t_S and t_T are the times at which paths S and T reach the goal of agent i respectively. For paths S, T, which can each be represented by a set of points $x_1, x_2, ..., x_\ell \in \mathcal{R}^2$, we define norm $||S|| = ||x_1, x_2, \cdots, x_\ell|| = \sum_{e=0}^{m} ||x_\ell||_2$. In short, $||S - T||$ is the sum of the Euclidean distances between the points on each path at each timestep. This distance function measures two important factors that would change an agent's value as a path changes. The first term accounts for physical locations (like rough terrain that is difficult to traverse or an area with a high number of humans that may get in the way). Paths that cross through these locations would have a higher cost. The second term accounts for a difference in flowtime for the individual agent. $0 \leq \lambda \leq 1$ is thus a parameter that allows us to trade off between these two factors. With this distance function in mind, we formally define the path $\tilde{S}_i$ which is the 'closest' path to our solution path S_i (among the paths that agent i submitted as bids) as $\tilde{S}_i = \arg\min_{S \in \text{Bids}_i} d(S_i, S)$. Finally, we define the approximate value of solution $\{S_j\}$ to agent i as $\tilde{v}_i(\{S_j\}) = b_i(\tilde{S}_i)$.

4.3 Descending Auction

Once we have an approximation for the social welfare of each proposed solution in $\mathcal{AS}$, we sort $\mathcal{AS}$ in decreasing order of approximate social welfare. Then, we carry out a simultaneous descending auction to elicit the true values of each solution to each agent. Specifically, the auctioneer first offers the solution which best approximates social welfare, $\{S_j^1\}$, to each agent at $\tilde{v}_i(\{S_j^1\})$. $\tilde{v}_i(\{S_j^1\})$ is the maximum possible price agent i could be charged for being allocated $\{S_j^1\}$ at the end of the auction. If all agents accept the offer, this solution is chosen. Otherwise, the approximate value for any agent i that did not accept the offer is reset to $\tilde{v}_i(\{S_j^1\}) = \tilde{v}_i(\{S_j^1\}) - \epsilon$, for some value $\epsilon > 0$, and $\mathcal{AS}$ is reordered. For large ϵ, the auction will execute quicker, but agents are likely to be proposed offers below their value for the item. For small ϵ, the descending auction will occur in shorter intervals, thus taking longer but gaining a more accurate representation of agent valuations. If at any point an agent accepted a solution, they are assumed to have accepted it in the future and are not offered a new value. If at any point an agent rejects an offer with value $\tilde{v}_i(\{S_j^1\}) < \epsilon$, the offer is reset to $\tilde{v}_i(\{S_j^1\}) = 0$. If an agent rejects an offer with value $\tilde{v}_i(\{S_j^1\}) = 0$, this solution is assumed to be infeasible and removed from the set.

This process is repeated until all agents accept the same solution or all solutions have been made infeasible. If all agents accept the same solution $\{\tilde{S}_j\} \in \mathcal{AS}$, then this becomes the implemented solution and the payment for agent i is

$$p_i = \max(0, \sum_{k \in \mathcal{I} \setminus \{i\}} b_k(S_k^{-i}) - \sum_{k \in \mathcal{I} \setminus \{i\}} \tilde{v}_j(\tilde{S}_k)), \tag{4}$$

where $\{S_j^{-i}\} \in \Gamma$ is the hypothetical solution described in Section 4.1 that results from a traditional VCG auction with the originally proposed bids, but without agent i. If all solutions have been deemed infeasible, the mechanism terminates with no solution.

4.4 Analysis

We now discuss how our modifications affect the important properties of individual rationality and incentive compatibility, and present arguments on how these are preserved by our approach.

Individual Rationality. We argue that each agent will take their assigned path. Suppose that an agent is considering deviating from their assigned path w, because there is some other path w' that has higher value for them. If w' was a path submitted to the auctioneer, they know that if it did not conflict with the other agents, they would have been allocated it. Thus, if they choose to follow w' instead of w, they are guaranteed to be in conflict with at least one other agent, which will cause w' to be infeasible. Otherwise, agents would not have determined the value of w', since there are prohibitively too many paths to value all of them. As a result, the agent would not be incentivised to deviate to it.

Incentive Compatibility. While our mechanism is not incentive compatible due to its sub-optimality [23], our design decisions prevent easy manipulation. If the algorithm does not enter the privileged knowledge sub-protocol then it remains in a VCG auction which is incentive compatible. If it does enter the sub-protocol, truthtelling means accepting the first value that is lower than your actual value for a path. Clearly, agents are not incentivised to accept a path that is higher than their value because they may be charged for it, netting a negative utility from the process. It is also important to show they will never turn down a price that is equal to or overestimates their value. Suppose they do not, then there is a chance

they will instead get offered a deal next which underestimates their value less, or a deal in which they have lower value. In the both cases, they would end up with either a worse utility solution or no solution.

5 SINGLE-AGENT DECISION MAKING

Our mechanism assumes agents have the ability to submit a fixed (but small) number of bids to the auctioneer. Each bid is on a bundle of items, and it is important that an agent understands what value to assign each bundle. To better describe the MAPF problem, for the remainder of the paper we refer to minimising the total cost, instead of maximising value, as the two problems are equivalent. If a bundle of items is not a path then it will cost an agent ∞ as it is impossible to execute. The cost of any path that eventually reaches the goal should be defined for each agent by the distance travelled and the time it takes. This will depend on the specifications of the agent, but an increase in both distance and time will be assumed to increase costs. While these costs and paths can be derived in any number of ways, and do not affect the properties of the mechanism, we outline a bidding strategy specific to the case where cost is directly linked to path length, as in a traditional MAPF setting. As we will show in Section 6, this method allows for a better overall result for the auction.

5.1 Agent Planning

We propose a method for each agent to quickly generate m sub-optimal paths $\{P^1_{rg}, \ldots, P^m_{rg}\}$ from a start r to a goal g, adapting the dissimilar path search algorithm from Jeong and Kim [15] for the MAPF context. The dissimilar path search algorithm first computes the all-to-one shortest paths to g from each other vertex using Dijkstra's algorithm [7]. It then takes the shortest path from r to g as its first selected path P^1_{rg}. We denote this path as $w = rx_1x_2 \ldots x_\ell g$. The new candidate paths are generated at each $x_d \in \{x_e\}_{e\in\{0,\ldots,\ell\}}$ (where $x_0 = r$), by branching off onto a neighbouring vertex $\tilde{x} \notin W \setminus \{x_d\}$, where W is the set of vertices in path w. Let $\tilde{x}\tilde{x_1}\tilde{x_2}\ldots\tilde{x}_{\ell'}g$ be the shortest path between $\tilde{x}$ and g. Then the new candidate path is $\tilde{w} = rx_1x_2\ldots x_d\tilde{x}\tilde{x_1}\tilde{x_2}\ldots\tilde{x}_{\ell'}g$. The special case $\tilde{x} = x_d$, which was disallowed in the original algorithm but is included here, effectively incorporates wait actions into the agents' path bids, thereby providing additional flexibility to the auctioneer.

After adding the newly generated paths to the candidate set C_{rg}, a new path is selected from the set of candidates based on minimisation of a similarity metric with the currently selected paths. We introduce a novel similarity metric that considers both spatial and temporal similarity of two paths. This metric measures the overlap of a candidate path with existing paths in both space and time:

$$P^{b+1}_{rg} = \underset{P\in C_{rg}}{\arg\min} \sum_{a=1}^{b} \frac{C\left[P^a_{rg} \bigcap^t P\right]}{C\left[P^a_{rg} \bigcup^t P\right]} \tag{5}$$

Here $C[\cdot]$ denotes the number of vertex-timestep pairs, and $\bigcap^t$ is a *temporal intersection* of two paths – for two paths p and q, $p \bigcap^t q$ contains all vertex-time pairs which appear in both paths, and the *temporal union* $\bigcup^t$ contains all vertex-time pairs which appear in either path. The similarity metric is designed to generate paths which do not have the agent occupying the same vertex *at the same time*, thereby providing the auctioneer with greater flexibility and improving its chances of finding a feasible allocation. The path found by Equation 5 is then removed from C_{rg} and used to generate new candidates, a process that is repeated until the m required paths have been selected.

6 EVALUATION

To analyse the performance of PKA, we simulated its performance on two synthetic domains from *Dragon Age: Origins* [31] and a warehouse domain, and compared its performance to baseline methods.

6.1 Methods

PKA, as described in Section 4, is implemented to request 10 bids from each agent. During the second phase of the auction, 3 possible solutions are generated with HCA* [30]. Because we consider MAPF scenarios where cost is equivalent to the path length, the auctioneer sets $\lambda = 0$ in Equation 3, i.e., it only considers time in the distance metric d. In the third stage of the auction, the price update amount ϵ is set to 1, as all edges in the simulated domain take time 1.

The primary method we compare to PKA is **iBundle** [26]. iBundle is an optimal iterative combinatorial auction. iBundle is implemented with anonymous prices, but because the start location of each robot is unique, discriminatory prices would reach the same result. As in PKA, the price update amount $\epsilon = 1$. Agents' bidding strategies are implemented as myopic best-responses to the auction. At the first stage of the auction, agents bid all simple paths with length equal to their shortest path. As prices increase when agents are unhappy after any iteration, agents then bid the paths from the previous rounds (updated to reflect the new prices) along with new paths, which comprise of all simple paths equal to the next best path length. This bidding strategy is implemented using Yen's algorithm [37] through *NetworkX* [12]. We also compare PKA to the concatenated bid version of **VCG**, where agents submit only 10 paths instead of bidding their entire list of possible paths, and then a traditional VCG auction is carried out. This method is equivalent to the first stage of PKA.

We simulate two different methods that allow each agent to generate and submit 10 bids to the auctioneer for both VCG and PKA. In the first method, **simple path** generation, agents submit 10 bids corresponding to their 10 shortest paths. As in iBundle bidding, these are generated with Yen's algorithm through *NetworkX*. In the second method, **dissimilar path** generation, agents submit 10 bids through the algorithm described in Section 5.1, which is based on [15] with a novel similarity metric that diversifies the spread of bids from each agent over the map.

We directly analyse the average path costs (as opposed to social welfare) because we were evaluating on MAPF benchmarks. Note that, in this context, maximising social welfare is equivalent to minimising the sum of path costs, as shown in [1] and mentioned in Table 1.

All methods are implemented in Python, and Mixed-Integer Linear Programs (e.g., solving the winner determination problem and calculating prices for VCG and for each iBundle iteration) are solved using Gurobi. All methods generate paths as a list of vertices at a given timestep. All experiments were conducted on an AWS C5a.large EC2 instance, with 2 CPUs and 4GB of memory. We

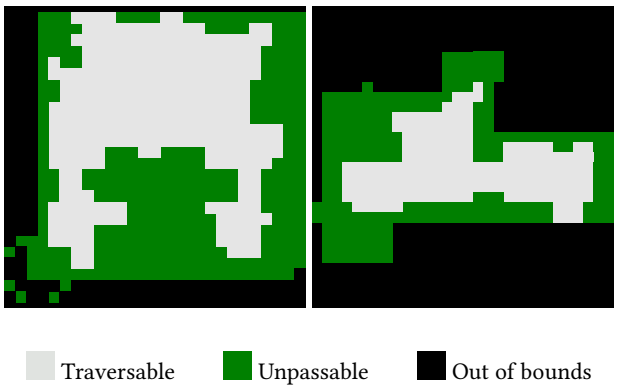

Figure 2: *Dragon Age: Origins* maps lak108d (left) and lak110d (right). Each pixel represents a vertex.

set a timeout of 300 seconds for all methods. Data points were calculated over 100 trials. Full distributional data can be found in the supplementary materials.

6.2 Dragon Age

Domain Description. Agents interact on two maps from *Dragon Age: Origins*, lak108d and lak110d, as shown in Figure 2. Both maps are drawn from a set of path finding baselines as described in [31]. In these baseline domains, agents value paths exclusively based on the time it takes to reach their goal location. Agents can move left, right, up, and down, which all take one timestep. Distinct start and goal vertices for each agent are generated randomly. Map lak108 has 286 vertices and map lak110 has 168 vertices.

Results. For maps lak108d and lak110d, we show the success rates of all tested methods over 100 trials in Figure 3. The iBundle algorithm consistently performs the worst. This is because in large, open domains like the two Dragon Age maps, agents have a large number of bids in the initial stage of the auction, as in open space there can be many possible best paths. It takes time for each agent to calculate these bids, but more importantly the large number of bids in the initial provisional allocation of iBundle can exceed the timeout of 300 seconds, as seen in Figure 4, and this becomes more likely as the number of agents increases. Figure 4 shows the time for each trial for every method, and trials that timed out are represented in the plot as taking time equal to the timeout. By comparing VCG (simple paths) and VCG (dissimilar paths), we see the dissimilar paths are much more likely to result in a initial solution than the simple paths, and this trend persists for all numbers of agents. Because the dissimilar paths intentionally discourages space/time overlap in the submitted paths, the VCG auction has a more diverse set of options to choose from, which results in a higher chance of finding a non-conflicting solution. PKA (dissimilar paths) is more frequently successful than VCG (dissimilar paths) and PKA (simple paths) is more frequently successful than VCG (simple paths) by consistently finding acceptable solutions in the second and third stage of PKA. The cost per agent of successful trials can be seen in Figure 5. A successful trial is defined per method, i.e., the plot representing iBundle represents all trials where iBundle was successful.

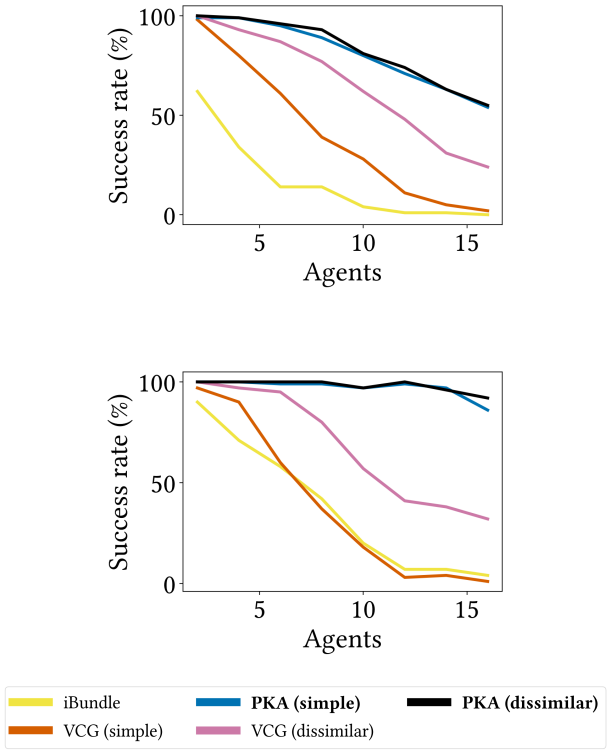

Figure 3: Success probabilities for *Dragon Age: Origins* maps lak108d (top) lak110d (bottom).

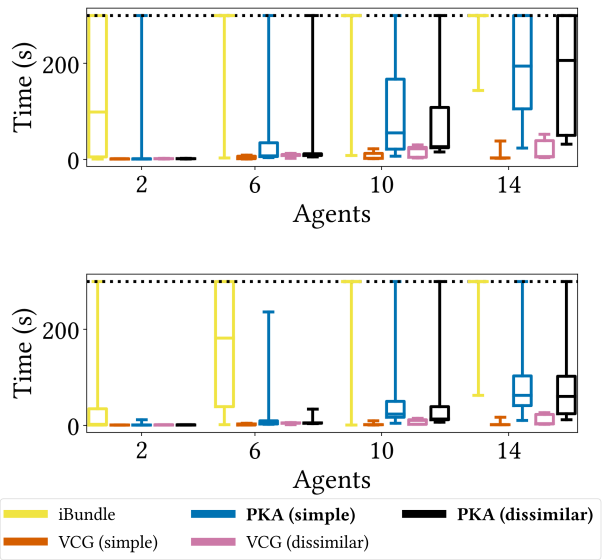

Figure 4: Computation time for *Dragon Age: Origins* maps lak108d (top) lak110d (bottom).

As a result, iBundle cost is lowest, as it is solving less problems, and in fact easier problems than the other methods. Similarly, across all

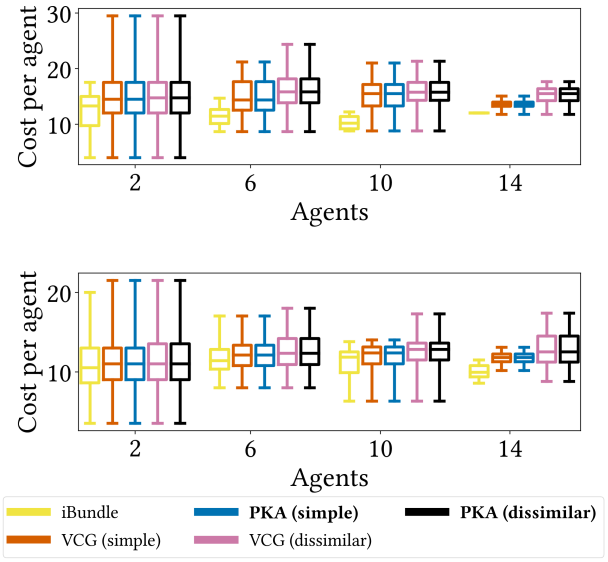

Figure 5: Average cost for *Dragon Age: Origins* maps lak108d (top) lak110d (bottom).

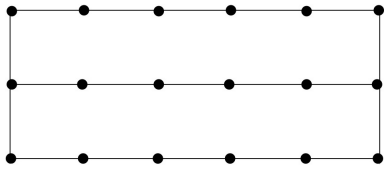

Figure 6: An example of a warehouse with $k = 3$ and $l = 6$.

methods the average cost path decreases as the number of agent increases as only the easier instances are solved.

6.3 Warehouses

Domain Description. To further analyse PKA, we simulated its performance on maps designed to mirror warehouses and supermarkets. Warehouse maps consist of aisles of shelves connected to each other only at the right and left corridors. These corridors are the most likely points of conflict in the maps. All maps of size (k, l) are constructed from k aisles of length l units. An example graph $\mathcal{G}$ is shown in Figure 6. Start and goal vertices for each agent are generated randomly, with no two agents sharing a start or goal vertex. This domain is similar to warehouse environments used in MAPF experiments, e.g., [21].

Results. In the warehouse environment, iBundle performs better because of the topology in the graph. With less connectivity, there are less valid simple paths that share the same number of edges, and thus iBundle is able to achieve the optimal solution more often. But even in these cases, PKA, a heuristic method, is able to achieve almost as good of a success probability (Figure 7). The time of iBundle and PKA are comparable as the warehouse size increases (Figure 8). The cost per agent of successful trials is similar across all methods, and can be found in the supplementary material. While simple

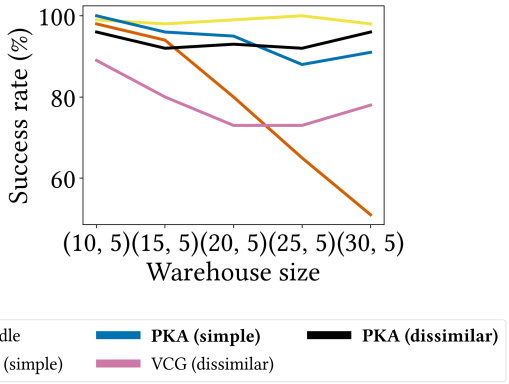

Figure 7: Success probabilities for a warehouse domain.

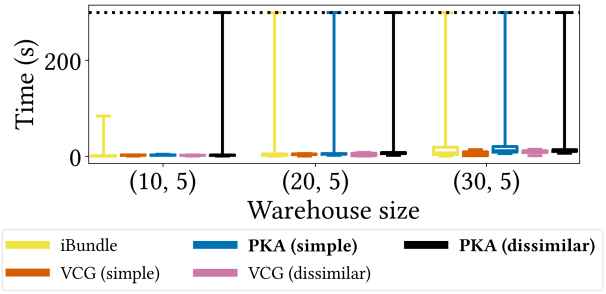

Figure 8: Computation time for a warehouse domain.

path generation is initially very successful for the same reason as iBundle, as the ratio of possible paths to submitted paths increases, the success of simple path generation plummets. Dissimilar path generation, on the other hand, provides a good balance throughout the warehouse sizes.

7 CONCLUSIONS

We have presented a novel method for non-cooperative multi-agent path finding. We extend prior work solving MAPF problems as combinatorial auctions by proposing a heuristic auction protocol that allows agents to value fewer paths by exploiting the privileged knowledge of the central planner and the physical layout of the planning space. Empirical results show that our mechanism outperforms the state-of-the-art approach for MAPF as a combinatorial auction on more general graphs. Future work includes exploring different methods of deconflicting paths and considering uncertainty over usage of shared resources.

ACKNOWLEDGMENTS

This work was supported by a gift from Amazon Web Services. Gautier was supported by the AIMS Centre for Doctoral Training. Lacerda and Hawes were supported by the EPSRC Programme Grant 'From Sensing to Collaboration' [EP/V000748/1], and a gift from Amazon Web Services. Wooldridge was supported by a UKRI AI World Leading Researcher Fellowship [EP/W002949/1].

REFERENCES

[1] Ofra Amir, Guni Sharon, and Roni Stern. 2015. Multi-Agent Pathfinding as a Combinatorial Auction. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence* (Austin, Texas) *(AAAI'15)*. AAAI Press, Palo Alto, California, 2003–2009. https://doi.org/10.5555/2886521.2886599

[2] Curt Bererton, Geoff Gordon, Sebastian Thrun, and Pradeep Khosla. 2003. Auction Mechanism Design for Multi-Robot Coordination. In *Proceedings of the 16th International Conference on Neural Information Processing Systems* (Whistler, British Columbia, Canada) *(NIPS'03)*. MIT Press, Cambridge, MA, USA, 879–886. https://doi.org/10.5555/2981345.2981455

[3] Ronen I. Brafman, Carmel Domshlak, Yagil Engel, and Moshe Tennenholtz. 2009. Planning Games. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence* (Pasadena, California, USA) *(IJCAI'09)*. AAAI Press, Palo Alto, California, 73–78. https://doi.org/10.5555/1661445.1661458

[4] Jesus Capitan, Matthijs T.J. Spaan, Luis Merino, and Anibal Ollero. 2013. Decentralized Multi-Robot Cooperation with Auctioned POMDPs. *The International Journal of Robotics Research* 32, 6 (2013), 650–671. https://doi.org/10.1177/0278364913483345

[5] Edward H. Clarke. 1971. Multipart Pricing of Public Goods. *Public Choice* 11, 1 (Sept. 1971), 17–33. https://doi.org/10.1007/BF01726210

[6] Peter Cramton, Yoav Shoham, and Richard Steinberg. 2006. *Combinatorial Auctions.* The MIT Press, Cambridge, MA, USA.

[7] Edsger W. Dijkstra. 1959. A Note on Two Problems in Connexion with Graphs. *Numerische mathematik* 1, 1 (1959), 269–271. https://doi.org/10.1007/BF01386390

[8] Shahar Dobzinski and Noam Nisan. 2007. Limitations of VCG-Based Mechanisms. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing* (San Diego, California, USA) *(STOC '07)*. Association for Computing Machinery, New York, NY, USA, 338–344. https://doi.org/10.1145/1250790.1250842

[9] David Eppstein. 1998. Finding the k Shortest Paths. *SIAM J. Comput.* 28, 2 (1998), 652–673. https://doi.org/10.1137/S0097539795290477

[10] Brian P. Gerkey and Maja J. Mataric. 2002. Sold!: Auction Methods for Multirobot Coordination. *IEEE Transactions on Robotics and Automation* 18, 5 (2002), 758–768. https://doi.org/10.1109/TRA.2002.803462

[11] Theodore Groves. 1973. Incentives in Teams. *Econometrica* 41, 4 (1973), 617–631. https://doi.org/10.2307/1914085

[12] Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. 2008. Exploring Network Structure, Dynamics, and Function using NetworkX. In *Proceedings of the 7th Python in Science Conference*, Gaël Varoquaux, Travis Vaught, and Jarrod Millman (Eds.). Pasadena, CA USA, 11 – 15.

[13] Wolfgang Hönig, T.K. Satish Kumar, Liron Cohen, Hang Ma, Hong Xu, Nora Ayanian, and Sven Koenig. 2016. Multi-Agent Path Finding with Kinematic Constraints. In *Twenty-Sixth International Conference on Automated Planning and Scheduling (ICAPS'16)*. https://doi.org/10.5555/3038594.3038654

[14] Luke Hunsberger and Barbara J. Grosz. 2000. A combinatorial auction for collaborative planning. In *Proceedings Fourth International Conference on MultiAgent Systems (ICMAS'00)*. IEEE, 151–158. https://doi.org/10.1109/ICMAS.2000.858447

[15] Yeonjeong Jeong and Dong-Kyu Kim. 2011. Dissimilar Alternative Path Search Algorithm Using a Candidate Path Set. In *Search Algorithms and Applications*. IntechOpen, Rijeka, Croatia, 409. https://doi.org/10.5772/14850

[16] Michael Kearns, Yishay Mansour, and Satinder Singh. 2000. Fast Planning in Stochastic Games. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence* (Stanford, California) *(UAI'00)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 309–316. https://doi.org/10.5555/2073946.2073983

[17] Sven Koenig, Craig A. Tovey, Michail G. Lagoudakis, Vangelis Markakis, David M. Kempe, Pinar Keskinocak, Anton J. Kleywegt, Adam Meyerson, and Sonal Jain. 2006. The Power of Sequential Single-Item Auctions for Agent Coordination. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 2* (Boston, Massachusetts) *(AAAI'06)*. AAAI Press, Palo Alto, California, 1625–1629. https://doi.org/10.5555/1597348.1597457

[18] Vijay Krishna. 2009. *Auction Theory* (2nd ed.). Elsevier Academic Press, Boston.

[19] Michail G. Lagoudakis, Evangelos Markakis, David Kempe, Pinar Keskinocak, Anton J. Kleywegt, Sven Koenig, Craig A. Tovey, Adam Meyerson, and Sonal Jain. 2005. Auction-Based Multi-Robot Routing. In *Robotics: Science and Systems (RSS'05)*. Rome, Italy, MIT Press, Cambridge, MA, USA, 343–350.

[20] Michael L. Littman. 1994. Markov Games as a Framework for Multi-Agent Reinforcement Learning. In *Proceedings of the Eleventh International Conference on International Conference on Machine Learning* (New Brunswick, NJ, USA) *(ICML'94)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 157–163. https://doi.org/10.5555/3091574.3091594

[21] Hang Ma, T. K. Satish Kumar, and Sven Koenig. 2017. Multi-Agent Path Finding with Delay Probabilities. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence* (San Francisco, California, USA) *(AAAI'17)*. AAAI Press, Palo Alto, California, 3605–3612. https://doi.org/10.5555/3298023.3298092

[22] Mitchell McIntire, Ernesto Nunes, and Maria Gini. 2016. Iterated Multi-Robot Auctions for Precedence-Constrained Task Scheduling. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems* (Singapore, Singapore) *(AAMAS '16)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 1078–1086. https://doi.org/10.5555/2936924.2937082

[23] Noam Nisan and Amir Ronen. 2007. Computationally Feasible VCG Mechanisms. *Journal of Artificial Intelligence Research* 29 (2007), 19–47. https://doi.org/10.5555/1622606.1622608

[24] Luyao Niu and Andrew Clark. 2019. Optimal Secure Control with Linear Temporal Logic Constraints. *IEEE Trans. Automat. Control* 65, 6 (2019), 2434–2449. https://doi.org/10.1109/TAC.2019.2930039

[25] Ernesto Nunes and Maria Gini. 2015. Multi-Robot Auctions for Allocation of Tasks with Temporal Constraints. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence* (Austin, Texas) *(AAAI'15)*. AAAI Press, Palo Alto, California, 2110–2216. https://doi.org/10.5555/2886521.2886614

[26] David C. Parkes. 1999. iBundle: An Efficient Ascending Price Bundle Auction. In *Proceedings of the 1st ACM Conference on Electronic Commerce* (Denver, Colorado, USA) *(EC '99)*. Association for Computing Machinery, New York, NY, USA, 148–157. https://doi.org/10.1145/336992.337032

[27] Philipp Schillinger, Mathias Bürger, and Dimos V. Dimarogonas. 2018. Auctioning over Probabilistic Options for Temporal Logic-Based Multi-Robot Cooperation under Uncertainty. In *IEEE International Conference on Robotics and Automation (ICRA'18)*. IEEE, 7330–7337. https://doi.org/10.1109/ICRA.2018.8462967

[28] Tomer Shahar, Shashank Shekhar, Dor Atzmon, Abdallah Saffidine, Brendan Juba, and Roni Stern. 2021. Safe Multi-Agent Pathfinding with Time Uncertainty. *Journal of Artificial Intelligence Research* 70 (2021), 923–954. https://doi.org/10.1613/jair.1.12397

[29] Guni Sharon, Roni Stern, Ariel Felner, and Nathan R Sturtevant. 2015. Conflict-Based Search for Optimal Multi-Agent Pathfinding. *Artificial Intelligence* 219 (2015), 40–66. https://doi.org/10.1016/j.artint.2014.11.006

[30] David Silver. 2005. Cooperative Pathfinding. In *Proceedings of the First AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* (Marina del Rey, California) *(AIIDE'05)*. AAAI Press, 117–122. https://doi.org/10.5555/3022473.3022494

[31] Roni Stern, Nathan R. Sturtevant, Ariel Felner, Sven Koenig, Hang Ma, Thayne T. Walker, Jiaoyang Li, Dor Atzmon, Liron Cohen, T. K. Satish Kumar, Eli Boyarski, and Roman Bartak. 2019. Multi-Agent Pathfinding: Definitions, Variants, and Benchmarks. In *Symposium on Combinatorial Search (SoCS'19)*. 151–158.

[32] Charlie Street, Sebastian Pütz, Manuel Mühlig, Nick Hawes, and Bruno Lacerda. 2021. Congestion-Aware Policy Synthesis for Multirobot Systems. *IEEE Transactions on Robotics* (2021), 1–19. https://doi.org/0.1109/TRO.2021.3071618

[33] Craig Tovey, Michail G. Lagoudakis, Sonal Jain, and Sven Koenig. 2005. The Generation of Bidding Rules for Auction-Based Robot Coordination. In *Multi-Robot Systems. From Swarms to Intelligent Automata Volume III*, Lynne E. Parker, Frank E. Schneider, and Alan C. Schultz (Eds.). Springer Netherlands, Dordrecht, 3–14. https://doi.org/10.1007/1-4020-3389-3_1

[34] William Vickrey. 1961. Counterspeculation, Auctions, and Competitive Sealed Tenders. *The Journal of Finance* 16, 1 (1961), 8–37. https://doi.org/10.2307/2977633

[35] Glenn Wagner and Howie Choset. 2017. Path Planning for Multiple Agents Under Uncertainty. In *Twenty-Seventh International Conference on Automated Planning and Scheduling (ICAPS'20)*. 577–585.

[36] Thayne T. Walker, David M. Chan, and Nathan R. Sturtevant. 2017. Using Hierarchical Constraints to Avoid Conflicts in Multi-Agent Pathfinding. In *Twenty-Seventh International Conference on Automated Planning and Scheduling (ICAPS'17)*. 316–324.

[37] Jin Y Yen. 1971. Finding the K Shortest Loopless Paths in a Network. *Management Science* 17, 11 (1971), 712–716. https://doi.org/10.1287/mnsc.17.11.712