Of Mechanism Design and Multiagent Planning

Roman van der Krogt[1] and **Mathijs de Weerdt**[2] and **Yingqian Zhang**[2]

Abstract. Multiagent planning methods are concerned with planning by and for a group of agents. If the agents are self-interested, they may be tempted to lie in order to obtain an outcome that is more rewarding for them. We therefore study the multiagent planning problem from a mechanism design perspective, showing how to incentivise agents to be truthful. We prove that the well-known truthful VCG mechanism is not always truthful in the context of optimal planning, and present a modification to fix this. Finally, we present some (domain-dependent) poly-time planning algorithms using this fix that maintain truthfulness in spite of their non-optimality.

1 Introduction

While planning has been, and is, extensively studied in single-agent environments, many interesting applications of planning feature an environment with more than one agent. This is where multiagent planning methods come into play. These enable the agents to reason about their interactions and ensure that their individual plans are efficient and effective. To solve various forms of the multiagent planning problem, several systems exist, such as (Generalised) Partial Global Planning (PGP) [3] and, more recently, MPOPR [15].

Most of these existing systems assume that the agents are cooperative. Consequently, complex interactions among multiple agents can be coordinated by one of them, or even by some central system, because it can be expected that all agents serve the same common cause, and that they all reveal any required information truthfully. However, often agents represent companies or other autonomous entities which may have (partially) conflicting preferences. Such self-interested agents do not have the option of simply trusting each other and solving the problem centrally, but will have to negotiate to ensure that their individual plans are valid in combination. This introduces the problem of mechanism design.

Mechanism design (MD) is a sub-field of economics and game theory. The goal of mechanism design is to design a set of rules for a "game", achieving certain criteria such as truthfulness. The designer may accomplish this by building in an incentive for the players to behave as intended. An example of such a mechanism ensuring truthfulness is the Vickrey-Clarke-Groves (VCG) mechanism. Mechanism Design has received a lot of attention over the past few years from the agent community, for example to help in the design of (auction) protocols for multiagent systems [13]. However, the consequences for multiagent planning have received only very little attention. It is from this perspective that the current paper arises.

First, we give a formal definition of a multiagent planning problem (MAP) and discuss mechanism design in this context. After that, we show how a well-known truthful mechanism (VCG) is not truthful for MAP, and we show how to modify this mechanism to make it truthful again. Our final contribution is to show which approximating planning algorithms can be used to obtain a truthful VCG-based mechanism.

2 Multiagent Planning Mechanisms

A *multiagent planning problem* $\theta \in \Theta$ for a set of agents $A = \{1, \ldots, n\}$ is a tuple $\theta = (\theta_1, \ldots, \theta_i, \ldots, \theta_n)$ of private planning problems $\theta_i \in \Theta_i$ for these agents. Following the set-theoretic notation for single-agent planning [5] where possible, agent i's planning problem θ_i consists of *(i)* a set F_i of ground atomic formulae, i.e. the propositions used to describe the domain of this agent; *(ii)* a set $O_i \subset \mathbb{O}$ of operators (actions) this agent may carry out that are defined by changes in the state, represented by sets of propositions; *(iii)* a cost function $c_i : O_i \to \mathbb{R}^-$ that assigns a cost to each operator; *(iv)* that part of the (common) initial state the agent is aware of, $I_i \subset F_i$; *(v)* a set of goals $G_i \subset F_i$; and *(vi)* a reward function $r_i : G_i \to \mathbb{R}^+$, assigning a reward to each of the goals.

We assume that all agents start from a global initial state I, although they may have a limited view of it. This global initial state is consistent (i.e. $I = \bigcup_{i \in A} I_i$ is conflict-free). The goals of different agents *can* be mutually exclusive, however.

The solution to a multiagent planning problem is a plan. We consider a plan to be a partially ordered sequence of actions $\pi = \langle o_1^{i_1}, \ldots, o_m^{i_m} \rangle$, where each $o_j^i \in O_i$. The subplan of π for agent i is $\pi_i = \langle o_j \mid o_j^i \in \pi \rangle$. The space of all plans is denoted by Π. The result of executing the plan π in the context of a planning problem θ is the state that is achieved after executing all actions sequentially. We denote this by $\mathit{Result}\,(I, \pi)$, where $I = \bigcup_{i \in A} I_i$ is the initial state as specified by θ.

Given a plan $\pi = \langle o_1^{i_1}, \ldots, o_m^{i_m} \rangle$ for a planning problem θ, we define the *cost* of that plan to be $c\,(\pi, \theta) = \sum_{o_j^i \in \pi_i} c_i\left(o_j^i\right)$. The costs that a particular agent i incurs equals the sum over the actions it is to execute: $c_i\,(\pi, \theta) = \sum_{o_j^i \in \pi} c_i\left(o_j^i\right)$. The *revenue* of π for a given problem θ is given by the reward functions for the goals that have been attained:

$$r\,(\pi, \theta) = \sum_{i \in A} \sum_{g \in G_i} \begin{cases} r_i(g) & \text{if } g \in \mathit{Result}\,(I, \pi) \\ 0 & \text{otherwise.} \end{cases}$$

Similar to $c_i\,(\pi, \theta)$, we identify $r_i(\pi, \theta)$ with the revenue of a particular agent i. The *utility* of plan π is defined as: $U(\pi, \theta) = c(\pi, \theta) + r(\pi, \theta)$. An optimal planner returns the plan which has the highest utility.

[1] Cork Constraint Computation Centre, University College Cork, Ireland; email: roman@4c.ucc.ie
[2] Delft University of Technology, Delft, The Netherlands; email: {M.M.deWeerdt,Yingqian.Zhang}@tudelft.nl

2.1 Mechanism Design

We are interested in a mechanism to construct the best multiagent plan. This formation of a multiagent plan π can be seen as a *social choice* over all possible plans Π, where each agent i has preferences over the possible plans defined by its *valuation* $v_i(\pi, \theta) = c_i(\pi, \theta) + r_i(\pi, \theta)$. The set of all possible plans depends on the local planning problems of the agents. These local planning problems comprise the input of our mechanism.

In this paper we consider these local planning problems to be private information of the concerned agent. This private information is usually called the *type* of an agent i. For MAP problems, this is $\theta_i = \{F_i, O_i, c_i, I_i, G_i, r_i\}$. The space of all possible types for an agent i is denoted by Θ_i. When all agents declare their type to the mechanism, we can use a planning algorithm to try to find the best multiagent plan.

As the agents can directly influence the generated plan by their declaration, sometimes they can profit from *lying* about their type. We distinguish three types of lying: *(i)* lying about the *value* of a plan, i.e. the functions c_i and r_i, and the goals G_i; *(ii) under-reporting* the available objects, i.e. reporting $O_i' \subset O_i$ and/or $I_i' \subset I_i$; and *(iii) over-reporting* non-existing objects, i.e. $O_i' \supset O_i$ and/or $I_i' \supset I_i$.

The output of the mechanism given the types as declared by the agents is a plan $\pi \in \Pi$. A mechanism using an optimal planning algorithm will choose the best plan in Π. Determining the best plan is not trivial, as self-interested agents may have conflicting preferences. One way is to look at the total *valuation* v of the agents for solving θ with a plan π:

$$v(\pi, \theta) = \sum_{i \in A} v_i(\pi, \theta) = \sum_{i \in A} c_i(\pi, \theta) + r_i(\pi, \theta)$$

Here $v_i(\pi, \theta)$ refers to the valuation of a particular agent i. We call $v(\pi, \theta)$ the *social welfare*. However, when agents are over-reporting their initial state or operations, the plan π may not be executed completely. In that case, we use $\hat{\pi}$ to denote that part of π that can still be successfully executed, and the valuation of an agent i becomes $v_i(\hat{\pi}, \theta)$.

2.2 Introducing Payments

Finding the best plan would be much easier if agents were not lying. In this paper we therefore study so-called *truthful* mechanisms that guarantee that agents are not better off by lying. In other words, truth-telling is a dominant strategy for agents in such mechanisms. Unfortunately, a direct corollary of the Gibbard-Satterthwaite Theorem [6] says that the only truthful mechanism for MAP is a *dictatorship*, i.e., there exists an agent $i \in A$ such that if $f(\theta)$ denotes the outcome of the mechanisms, then $f(\theta) \in \text{argmax}_{\pi \in \Pi}\ v_i(\pi, \theta)$ for all θ.

Corollary. *The only mechanisms for multiagent planning with general utility functions that are truthful are dictatorial.*

Clearly, such dictatorial mechanisms are not desirable. We can circumvent this issue by introducing payments to penalize some agents and possibly reimburse some others based on their contribution to the social welfare. For this, we introduce a payment function $p_i : \Theta_1 \times \cdots \times \Theta_n \to \mathbb{R}$ that specifies for each agent i the amount that i pays. From here on, we consider a mechanism to be a tuple $(f, p_1, \ldots p_n)$ where $f : \Theta_1 \times \cdots \times \Theta_n \to \Pi$ is planning function, and $p_1, \ldots, p_n$ are payment functions. The goal of mechanism design for MAP is thus to find a mechanism $(f, p_1, \ldots p_n)$ such that $f(\theta)$ returns the plan which maximizes the social welfare. With payments, the *utility* of the agent i on the outcome $\pi = f(\theta)$ is defined by: $u_i(\pi, \theta) = v_i(\pi, \theta) - p_i(\theta)$. The utility is what rational agents aim to maximize.

3 VCG Mechanisms for MAP

In this section we investigate the applicability of the most common class of payment functions to a mechanism for MAP. For this we first introduce some definitions and notations (based on [11]). For example, when we reason about replacing the input of one of the agents i in a MAP problem θ, we use the notation (θ_i', θ_{-i}) to indicate the agents' declared planning problems where agent i declared θ_i', and all other declared planning problems are left unchanged.

Definition 1. A mechanism $(f, p_1, \ldots p_n)$ is called *incentive compatible* (or *truthful*) iff for every agent i, every set of true types $\theta_1, \ldots, \theta_i, \ldots, \theta_n \in \Theta_1 \times \ldots \times \Theta_i \times \ldots \times \Theta_n$ and every alternative $\theta_i' \in \Theta_i$: $v_i(f(\theta_i, \theta_{-i}), \theta) - p_i(\theta_i, \theta_{-i}) \geq v_i(f(\theta_i', \theta_{-i}), \theta) - p(\theta_i', \theta_{-i})$.

In words, no agent can achieve a higher utility by lying about its type. Truthfulness is one of the most desirable properties of a mechanism. So-called Vickrey-Clarke-Groves mechanisms are very successful in satisfying this property [11].

Definition 2. A mechanism $(f, p_1, \ldots, p_n)$ is called a *Vickrey-Clarke-Groves (VCG) mechanism* if:

- $f(\theta) \in \arg\max_{\pi \in \Pi} v(\pi, \theta)$, i.e. f maximises social welfare;
- for some functions $h_1, \ldots, h_n : \Theta^{n-1} \to \mathbb{R}$, we have that for all valuations $\theta = (\theta_1, \ldots, \theta_n)$: $p_i(\theta) = h_i(\theta_1, \ldots, \theta_{i-1}, \theta_{i+1}, \ldots, \theta_n) - \sum_{j \neq i} v_j(f(\theta), \theta)$.

In the remainder of the paper we choose $h_i(\theta_{-i}) = 0$ for ease of representation. However, all truthfulness arguments hold for any choice for $h_i(\theta_{-i})$, because its value is independent of agent i's declared type.[3] Given a set of declared types of the agents, the mechanism generates a plan using f and computes the payment p_i for each agent i. The agents then deliver the payments $p_1, \ldots, p_n$ to the mechanism. Previous work has shown that every VCG mechanism is incentive compatible [11]. Using an optimal planning algorithm f, VCG mechanisms work that well, because *(i)* the agents' utility and thus their incentives are aligned with the social welfare, and moreover *(ii)* the goal of the algorithm is also to maximise the social welfare. In the following we show that VCG works for two out of the three types of lying introduced in the previous section.

Theorem 3. *The VCG mechanism for MAP with an optimal algorithm prevents lying about values and under-reporting, or a combination of both.*

This theorem is not very surprising, considering the previous work on VCG mechanisms [11]. The reason that agents

[3] The Clarke pivot rule is a popular alternative, defining $h_i(\theta_{-i})$ as the social welfare without i's participation [11].

have no incentive to lie about their values is that the payment function changes an agent's utility as follows.

$$u_i(\pi, \theta) = v_i(\pi, \theta) - p_i(\theta_i, \theta_{-i}) = \\ v_i(\pi, \theta) + \sum_{j \neq i} v_j(\pi, \theta) + h_i(\theta_{-i}) = v(\pi, \theta),$$

where $\pi = f(\theta_i, \theta_{-i})$. Therefore, agent i's utility can be seen as being equal to the social welfare. This social welfare is maximised by an optimal algorithm only if it is given the correct (i.e., true) valuations to optimise.

With respect to the second type of lying, agents run the risk of generating less revenue by not reporting all their objects, because the plan generated by the optimal planner when i under-reports cannot have a higher utility than the one generated when i fully reports its objects. Hence, since $U(\pi, \theta) = \sum_{i \in A} r_i(\pi, \theta) + c_i(\pi, \theta) = v(\pi, \theta)$, agent i's utility can also not be higher.

Surprisingly, for the third type of lying, i.e. over-reporting, agent i *can* gain from reporting *more* than it has at its disposal. Intuitively, the reason for this is in the case of VCG for MAP the outcome is a global, distributed plan that only achieves its value upon *successful* execution. This gives the agents additional ways to cheat which aren't prevented by the VCG mechanism: their penalties are based on what they *promise* to do; not on what they *actually* achieve. An agent may for example be rewarded for actions that it claims it has and that help other agents to achieve their goals, but which it cannot actually execute. If those actions are included in the generated plan π, the utility of π is not representing the social welfare. So even an optimal planner cannot guarantee to output a "best" plan which maximizes the social welfare.

Theorem 4. *The VCG mechanism with an optimal algorithm for MAP cannot prevent over-reporting.*

Proof. As we have shown above, agent i's utility for π when reporting truthfully is $u_i(\pi, \theta) = v(\pi, \theta)$. Now let agent i over-declare in θ_i' the actions it can execute, i.e. $O_i' \supset O_i$, and let the costs of such declared actions be 0. Consider a resulting plan π' such that the over-declared actions do not let i achieve more goals compared to those in π, but helps other agents $j \neq i$ to reach some additional goals G', i.e. $G' = \{g \in G_j \mid g \in \mathit{Result}\,(I, \pi') \setminus \mathit{Result}\,(I, \pi)\}$. Agent i's utility of π' is $u_i(\pi', \theta) = v_i(\hat{\pi}', \theta) - p_i(\theta_i', \theta_{-i}) = v(\pi', \theta) + v_i(\hat{\pi}', \theta) - v_i(\pi', \theta_i', \theta_{-i})$, where $v_i(\hat{\pi}', \theta)$ denotes the valuation of i based on the feasible plan $\hat{\pi}'$ and its true type θ_i, and $v_i(\pi', \theta_i', \theta_{-i})$ is the valuation based on declaration θ_i'. Since the costs of over-declared actions are 0 and no additional goal of agent i is attained in π', we have $v_i(\hat{\pi}', \theta) = r_i(\hat{\pi}', \theta) + \sum_{o_j^i \in \hat{\pi}_i} c_i(o_j) = v_i(\pi', \theta_i', \theta_{-i})$. Hence, $u_i(\pi', \theta) = v(\pi', \theta)$. Furthermore, because the goals in G' are attained in π' but not in π, we have $v(\pi', \theta) > v(\pi, \theta)$. Thus the utility of π' for i is greater than that of π, i.e. $u_i(\pi', \theta) > u(\pi, \theta)$. Therefore, agent i increases its utility by over-reporting. □

Of course, over-reporting by agents does not necessarily induce infeasible (local) plans. However, when over-reporting results in a feasible plan, i.e., $\hat{\pi} = \pi$, then the optimal planning algorithm ensures that this plan maximises the social welfare. Since each agent's utility is aligned with the social welfare, agents will not be better off by over-reporting.

Proposition 5. *Under a VCG mechanism, the agents have an incentive to over-report or to mix over-reporting with other lying types only if such declarations incur an infeasible plan.*

4 VCG with Deposits

In order to avoid over-reporting agents, we introduce the deposit-VCG mechanism. The idea of this mechanism is that every agent is required to place a deposit in order to participate. Each agent gets its deposits back only after the successful execution of its local plan.

Definition 6. Given a planning problem, the *deposit-VCG* mechanism works as follows:

1. The mechanism asks the agents to declare their types θ_i.
2. The mechanism then asks each agent to pay the amount $r(G) = \sum_{i \in A} \sum_{g_i \in G_i} r_i(g_i)$ as a deposit.
3. The mechanism finds a plan π using an optimal planning algorithm f, taking into account only the agents who paid the deposit.
4. Each agent i pays p_i according to the VCG formula in Definition 2 with for example $h_i(\theta) = 0$.
5. The mechanism informs the agents of the plan π, and each agent i executes its part π_i.
6. If any local plan fails due to the agent i's declaration, agent i will not get its deposit back. All other agents are returned their deposits.

Since the separate deposit stage does not enlarge the strategy space of the agents, it is straightforward to see that if the agents are truthful under the VCG mechanism, they will not be better off by lying under the deposit-VCG mechanism.

Proposition 7. *The deposit-VCG mechanism is truthful when the VCG mechanism is truthful.*

Consequently, deposit-VCG is truthful with respect to lying about values and under-reporting. Moreover, we show below that it can also prevent over-reporting.

Theorem 8. *The deposit-VCG mechanism with an optimal algorithm is truthful for MAP.*[4]

Proof. From Proposition 5, we know that agent i has an incentive to over-report only if the resulting plan is infeasible. Suppose such an infeasible plan π' is generated due to i's declaration θ_i'. Agent i will then be caught by the mechanism during the execution of π', because some goal in π' cannot be reached. Using a VCG mechanism its utility would have been $u_i(\pi', \theta) = v(\pi', \theta)$. In the deposit-VCG mechanism, the agent will not be returned its deposit of $r(G)$ and we know that $r(G) \geq v(\pi', \theta)$ by definition. Therefore, we have $u_i(\hat{\pi}', \theta) \leq 0$ if i over-reports. This is usually a much smaller amount than the case where agent i is truthful, because then it will get its deposit returned. So, its utility then is: $u_i(\pi, \theta) = r(G) - r(G) + v(\pi, \theta) \geq 0$. Therefore, agent i is never worse off by truth-telling. □

The following setting illustrates that the proposed deposit is the smallest deposit possible in the worst case. Suppose no goal can be attained, but one agent claims it can help

[4] If all lying agents can be detected.

to achieve the goals of all (other) agents. This agent then collects $r(G)$ in payments (both when $h_i(\theta_{-i}) = 0$ and using the Clarke pivot rule), but fails to execute his actions.

5 VCG-based Approximations

The (deposit-)VCG mechanism requires that f makes optimal decisions. Except for some specific domains (such as reported in [7, 8]), or for domains with restrictions as identified in [2], this is intractable, as planning in general is PSPACE-complete. However, there are many non-optimal planning algorithms that often produce very reasonable results on general domains. This begs the question whether we can design a truthful, polynomial-time mechanism around such non-optimal planners. We will call a mechanism $(f, p_1, \ldots, p_n)$ *deposit-VCG-based*, if f is a sub-optimal planning algorithm and $p(\cdot)$ is calculated according to the deposit-VCG mechanism (Definition 6). Unfortunately, where deposit-VCG mechanisms are incentive compatible, deposit-VCG-based mechanisms are generally not. The reason is that VCG payments align the agent's utility with value of the system's solution. Therefore by lying, an agent may "help" a non-optimal mechanism to achieve a better solution, and thus make more profit for itself. Hence, only under special conditions can we show (deposit-)VCG-based mechanisms to be truthful [12].

Definition 9. Given a planning algorithm f, let Π' denote the range of f at Θ, i.e. $\Pi' = \{f(\theta) \mid \theta \in \Theta\}$. We say f is *maximal in its range* (MIR) if for every type $\theta \in \Theta$, $f(\theta)$ maximises $\sum_{i \in A} v_i(\pi, \theta)$ over $\pi \in \Pi'$.

Proposition 10. *A (deposit-)VCG-based mechanism $(f, p_1, \ldots, p_n)$ with f maximal in its range is truthful.*

Informally speaking, a planning algorithm f is MIR if it optimises the social welfare by selecting the best plan out of an on *forehand determined* set of allowable plans. Obviously, optimal planning algorithms are MIR. In general, non-optimal planning algorithms are not. However, for a number of planning domains, approximations are known that can be used to create MIR mechanisms, because they select the best plan among a *restricted* set of plans.

In the remainder of this section we give one such example. A well-known result for Blocks World (BW) states that although *optimal* planning for BW is NP-hard, non-optimal planning is tractable [14]: one can first *unstack* all blocks onto the table, and then use *move* actions to assemble the towers of these blocks that match the goal(s). In fact, there exists a slightly more efficient version of this method. Instead of unstacking *all* blocks to the table first, we only unstack those blocks that are either *(i)* not in their final position, or *(ii)* prevent a block that satisfies the first condition from being moved (i.e. it is above such a block or taking the goal location of such a block). Denote this algorithm by f_{bw}. Note that this algorithm is not optimal, because sometimes a block can be moved immediately from its current position to its goal position without being unstacked to the table in between. Now imagine a multiagent BW-variant, where the agents' goals specify which blocks should be on top of which others. Morever, these goals are the only private information in the system. We discuss why this algorithm is MIR.

Such a BW instance $\theta = \{F, O, c, I, G, r\}$ consists of *(i)* the domain F of all block's positions; *(ii)* two operations in O: unstack a block to the table, or move a block upon another block and their costs c; *(iii)* the initial state I that specifies the initial position of each block; and *(iv)* for each agent i a set of goals G_i to specify the final position of some blocks, and the reward function r_i. Given θ, let first m be the number of blocks that are not in their final position yet, and then let n be the number of remaining blocks that are to be moved out of the way of the first m blocks (without counting a block by both m and n).

Assume the set of goals does not contain any conflicts, and that the rewards for the goals are relatively high.[5] Then the range of f_{bw} is the set of all plans that consist of first unstacking the n blocks that are in the way and the m blocks that are to be moved, and then assembling the stacks of blocks to attain the goals by m stacking actions. We now verify that f_{bw} is MIR. In this situation, the plan that attains the most goals is the plan with the highest social welfare $\sum_{i \in A} v_i(\pi, \theta)$. If the goals are not conflicting, f_{bw} always attains all goals with at most $n + 2m$ actions. By Proposition 10, a (deposit-)VCG-based mechanism using f_{bw} is truthful.

Proposition 11. *The (deposit-)VCG-based mechanism $(f_{bw}, p_1, \ldots, p_n)$ is truthful and runs in polynomial time.*

If, however, the goals have conflicts, the social welfare depends on which goals are satisfied. Therefore f_{bw} is only MIR if it selects the set of non-conflicting goals with the highest reward. To realise this it should either *(i)* consider all possible combinations of goals that can be satisfied, an intractable task, or *(ii)* put a limit on the number of goals it will satisfy, and consider all combinations of this number of goals. By limiting the number of goals to be attained by K, we can impose a polynomial bound on the mechanism's time complexity.

Proposition 12. *Given a polynomial-time algorithm $f_d : \Theta \to \Pi$ for a planning domain d that is MIR on problems without conflicting goals, and an upper-bound K on the number of goals that is considered, an algorithm f_d^K exists that is MIR and polynomial in the input size.*

Proof. The algorithm f_d^K should impose a limit K on the number of goals that is satisfied at most. In the worst case, f_d^K needs to consider at most $\sum_{i=0}^{K} \binom{|G|}{i} \leq K \cdot |G|^K$ possible (i.e. non-conflicting) combinations of goals, and select the best one to ensure it is maximal in range. As this is a polynomial amount (except in K), and f_d^K runs polynomially, the result can still be computed in polynomial time. □

Let f_{bw}^K denote the algorithm that limits the number of goals considered by K and is based on f_{bw}. It follows immediately from the above proposition that f_{bw}^K is a polynomial-time algorithm for BW problems with or without conflicting goals.

Proposition 13. *The approximation algorithm f_{bw}^K for BW problems with an upper-bound K on the number of goals to be satisfied is MIR and runs in polynomial time.*

Fortunately, many realistic domains are naturally conflict-free. So it is not necessary to limit the number of goals to

[5] We assume that the rewards of the goals are higher than the costs of the required actions. If not, the agents have no incentive to achieve the goals.

be attained. For example, if the agents involved have control over distinct sets of resources, and their individual goals are formulated over their own resources, and locally conflict-free, the domain as a whole is conflict-free as well. An example of such a domain is the Logistics domain (for which non-optimal strategies exist similar to the BW-strategy [9]). In a MAP variant of this domain, each of the trucks and airplanes is owned by a single agent. Moreover, it is not unreasonable to assume that each task of delivering a package belongs to precisely one agent, and that this agent takes on only one delivery order for each package. The deposit-VCG mechanism can be used to ensure that the local deliveries are coordinated with the scheduled flights such that all goals are met efficiently.

6 Discussion

When planning is performed by and for a group of self-interested agents, mechanisms must be in place to ensure that the agents behave honestly. Heretofore, this notion has been largely overlooked. One early attempt is described in [4]. It uses an iterative voting mechanism that lets the agents vote on subsequent steps in the plan. Truthfulness is guaranteed for each individual voting round using a Clarke taxation, but not for the complete mechanism. At each step, the agents reveal additional information to the other agents regarding their goals, and based on this, a set of candidate successor states is generated. It is this set that a vote is held over. The main difference between that work and ours is that we also consider truthfulness with respect to the declaration of goals, operators, and the initial state and thus indirectly over the set of candidate plans, instead of only making the declaration of the value of candidates plans strategyproof. Also our theoretical results apply to any planning algorithm; not just to an iterative forward state-space planning algorithm. Moreover, in our work the private information of agents is only shared with the central mechanism, not with the other agents.

More recent work [10] starts from a setting where the individual plans of the agents are already known, but where the agents collectively need to decide on the order in which the actions are executed. The strategy space for each agent is defined there as the set of possible orders for its individual plan. A solution is then indicated by a Nash equilibrium in the resulting matrix game. In our work, we generalise this setting, starting not from individual plans, but from the individual sets of possible operations and goals. We have shown how to guarantee desirable solutions by truthful mechanisms. In such mechanisms these solutions are indicated by a dominant strategy equilibrium, which is a much stronger solution concept than a Nash equilibrium, making such guarantees possible.

We believe that a better understanding of such game theoretical notions is essential to the development of future multi-agent planning systems for self-interested agents. In this paper we therefore placed several results from the field of mechanism design in the context of planning. We showed how the multiagent planning problem can be conceived as a social choice over all possible plans, where each agent has preferences over the plans as given by its part of the planning problem, i.e. it will favour plans that achieve its goals in a cost-effective way.

In problem domains where the only private information is a value, as for instance in combinatorial auctions [1], the standard VCG mechanism with an optimal algorithm is truthful. However, as we have seen in this paper, the VCG mechanism is not generally applicable in the setting of MAP. Informally, the reason for this is that VCG cannot prevent over-reporting of capabilities resulting in infeasible plans. This issue can be resolved by extending the VCG mechanism. In this mechanism agents pay a deposit, to be returned when the plan is verified to be feasible (i.e. upon successful execution). Our current work is to show that this result generalises to other mechanism design settings where the private information of agents is not only used to calculate the social welfare of an alternative, but in fact to determine this set of alternatives.

Finally, we studied deposit-VCG-based approximation mechanisms for MAP. Although it is not possible to construct a general, domain-independent VCG-based approximation [12], we were able to show how some domain-dependent approximations can be used to solve MAP in polynomial time.

As future work, we are interested in studying how other (approximation) algorithms for planning can be used to construct efficient and truthful mechanisms, focusing especially on variants of existing distributed MAP algorithms.

Acknowledgements
Roman van der Krogt is supported by an Irish Research Council for Science, Engineering and Technology (IRCSET) Postdoctoral Fellowship. Mathijs de Weerdt and Yingqian Zhang are supported by the Technology Foundation STW, applied science division of NWO, and the Ministry of Economic Affairs of the Netherlands.

References

[1] L. Blumrosen and N. Nisan, 'Combinatorial auctions', in *Algorithmic Game Theory*, 267–300, Cambridge University Press, (2007).
[2] Tom Bylander, 'The computational complexity of propositional STRIPS planning', *Artificial Intelligence*, **69**(1–2), 165–204, (1994).
[3] K.S. Decker and J. Li, 'Coordinating mutually exclusive resources using GPGP', *Autonomous Agents and Multi-Agent Systems*, **3**(2), 113–157, (2000).
[4] E. Ephrati and J.S. Rosenschein, 'Multi-agent planning as search for a consensus that maximises social welfare', in *Artificial Social Systems*, pp. 207–226, (1994).
[5] M. Ghallab, D. Nau, and P. Traverso, *Automated Planning, theory and practice*, Morgan Kaufmann Publishers, 2004.
[6] A. Gibbard, 'Manipulation of voting schemes: A general result', *Econometrica*, **41**(4), 587–601, (July 1973).
[7] M. Helmert, 'Complexity results for standard benchmark domains in planning', *Art. Int.*, **143**(2), 219–262, (2003).
[8] M. Helmert, 'New complexity results for classical planning benchmarks', in *Proc. of the 16th Int. Conf. on Automated Planning and Scheduling (ICAPS)*, pp. 52–62, (2006).
[9] M. Helmert, R. Mattmüller, and G. Röger, 'Approximation properties of planning benchmarks', in *Proc. of the 17th European Conf. on AI (ECAI)*, pp. 585–589, (2006).
[10] R. Ben Larbi, S. Konieczny, and P. Marquis, 'Extending classical planning to the multi-agent case: A game-theoretic approach', in *Eur. Conf. on Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, pp. 731–742, (2007).
[11] N. Nisan, 'Introduction to mechanism design (for computer scientists)', in *Algorithmic Game Theory*, 209–242, Cambridge University Press, (2007).
[12] N. Nisan and A. Ronen, 'Computationally feasible VCG mechanisms', *Journal of AI Research*, **29**, 19–47, (2007).
[13] Y. Shoham and K. Leyton-Brown, *Multiagent Systems: Algorithmic, Game-Theoretic and Logical Foundations*, Cambridge University Press, 2008.
[14] J. Slaney and S. Thiébaux, 'Blocks world revisited', *Artificial Intelligence*, **125**(1-2), 119–153, (2001).
[15] R. van der Krogt and M. de Weerdt, 'Coordination through plan repair', in *MICAI 2005: Advances in Artificial Intelligence*, pp. 264–274. Springer, (2005).