In *Proceedings of The Fourth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-05)*

Computationally-Efficient Combinatorial Auctions for Resource Allocation in Weakly-Coupled MDPs

Dmitri Dolgov and Edmund Durfee
Department of Electrical Engineering and Computer Science
University of Michigan
Ann Arbor, MI 48109
{ddolgov, durfee}@umich.edu

ABSTRACT

The capabilities of an autonomous agent are often determined by the resources that are available to it. We examine the problem of allocating scarce resources among multiple self-interested agents, operating in complex, stochastic environments (modeled as MDPs), where the value of a particular resource bundle to an agent is the expected payoff of the best control policy the agent can implement using these resources. Combinatorial auctions are popular mechanisms for allocating resource bundles among agents with such complex preferences. In particular, generalized Vickrey auctions (GVAs) yield socially optimal outcomes and have excellent strategic complexity, but suffer from high computational complexity for the agents' preference-valuation and the auctioneer's winner-determination problems. We describe a GVA-based mechanism that exploits knowledge of the agents' MDP-based resource preferences, and we show analytically and demonstrate empirically that this leads to an exponential improvement in computational efficiency over a straightforward GVA implementation with flat preferences. We also present a distributed implementation of the winner-determination problem that leads to further speedup while maintaining strategy-proofness of the mechanism, and without revealing agents' private MDP information.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search; I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent systems*

General Terms

Algorithms, Performance, Design

Keywords

Markov Decision Processes, Combinatorial Auctions, Generalized Vickrey Auctions, Distributed Implementation

AAMAS'05, July 25-29, 2005, Utrecht, Netherlands.

1. INTRODUCTION

The task of formulating an optimal control policy for an autonomous agent operating in a stochastic environment is often modeled as a Markov decision process (MDP). In a standard MDP (e.g., [14]), an agent has a set of actions and must choose, for each state of the world, an action that will maximize some measure of performance. Implicit in the standard model is the assumption that a policy can be comprised of any combination of actions (an agent's choice of action in one state does not affect its choices in other states). However, this might not be the case for a constrained agent, where devoting limited resources to some actions might make it impossible to also carry out other actions. For example, a delivery agent might have limited capacity and choosing to deliver some packages might preclude it from delivering others on the same trip. The constrained policy optimization problem the agent faces thus requires finding a policy that maximizes payoffs without violating the agent's constraints.

The problem becomes even more challenging when it involves multiple self-interested agents that share the limited resources, such as multiple delivery agents that obtain their delivery contracts from the same place (rewards are received upon fulfillment of delivery contracts). Now, not only are the agents separately presented with difficult constrained policy optimization problems, but they also have conflicting objectives, since each agent wants to acquire the combination of resources that would enable it to execute its optimal control policy. Limitations in the amounts of shared resources generally mean that not every agent can acquire the combination of resources it would most desire.

Thus, the goal of the work described in this paper is to design a mechanism for allocating scarce shared resources among multiple constrained agents who are formulating MDP policies to selfishly maximize their individual expected rewards, and are *weakly-coupled* [11, 17], meaning that the agents' only interactions are due to contention over shared resources. We would like the mechanism to produce allocations that are efficient from the economic point of view (social welfare is maximized), and do so in a computationally-efficient manner.

Combinatorial auctions [5] are a natural starting place for designing such a mechanism. In particular, Generalized Vickrey Auctions (GVAs) [10] are widely-studied because they produce socially-optimal allocations and have excellent strategic complexity (agents' strategic reasoning is simple, because they have dominant bidding strategies). However, a weak point of GVAs is the high computational complex-

ity of the agents' preference-valuation problem (how much to bid for resource bundles) [12], and the winner-determination problem of the auctioneer (given the bids, how to allocate resources and how to set prices) [15]. The high computational complexity stems from the fact that, in general, the agents need to specify preferences over an exponential number of resource bundles, and the auctioneer has to solve a combinatorial winner-determination problem on that space of bundle allocations. However, in many domains, agents' preferences are not arbitrary and are well-structured, which suggests using concise preference-specification languages that exploit that structure. In particular, *logical bidding languages* have been proposed [3] that allow the agents to specify their valuations on logical formulas over allocations of goods, as well as efficient winner-determination algorithms that make use of such concise preference specifications [1]. In a similar fashion, we propose techniques for improving the computational efficiency for MDP-based agents.

Our main contribution in this paper is that we present a GVA-based mechanism that exploits the knowledge of agents' MDP-based preferences to achieve exponential gains in computational efficiency over a naïve GVA implementation that uses a flat representation for agents' preferences. Our mechanism inherits the nice properties of the standard GVAs, such as their socially-optimal outcomes and excellent strategic complexity. We also present a distributed implementation of the winner-determination problem that leads to additional speedup, and accomplishes this without sacrificing strategy-proofness of the mechanism and without requiring the agents to reveal their private MDP information. To avoid nonessential complications, in this paper, we focus on non-consumable resources that enable agents to execute actions but are themselves not consumed in the process, and then briefly mention the (simpler) case of consumable resources in Section 8.

In the rest of the paper we first briefly review some background and introduce our model in Section 2. Then, in Section 3, we discuss a straightforward implementation of a combinatorial auction for our MDP-based problem. Our new mechanism is developed in the sections that follow: in Section 4, we show how to improve computational efficiency of the bundle-valuation and winner-determination problems, Section 5 further speeds up the winner-determination process via a distributed algorithm that utilizes the computational power of the agents, and Section 6 discusses information privacy issues. We present an empirical evaluation of our mechanism in Section 7, and conclude in Section 8 with a discussion of generalizations of our approach and some interesting unanswered questions.

2. MODEL AND BACKGROUND

In the first half of this section we describe some well-known facts from the theory of classical MDPs and introduce our constrained MDP model. In the second half, we briefly review some properties of combinatorial auctions.

2.1 Weakly-Coupled Constrained MDPs

A classical single-agent, fully-observable MDP can be defined as a 4-tuple $\langle \mathcal{S}, \mathcal{A}, p, r \rangle$, where: $\mathcal{S} = \{i\}$ is a finite set of states the agent can be in; $\mathcal{A} = \{a\}$ is a finite set of actions it can execute; $p : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto [0, 1]$ is a stochastic ($\sum_j p_{iaj} = 1$) transition function (p_{iaj} is the probability the agent will transition to state j upon executing action a in state i); $r : \mathcal{S} \times \mathcal{A} \mapsto [-r_{max}, r_{max}]$ is a bounded reward function (the agent gets a reward of r_{ia} for executing action a in state i).

A solution to an MDP is a policy (a procedure for selecting an action in every state) that maximizes some measure of aggregate reward. In this paper we will focus on MDPs with the total expected discounted reward optimization criterion, but our results can easily be extended to other optimization criteria as well. It is known (e.g., [14]) that, for such MDPs, there always exist policies that are *uniformly-optimal* (optimal for all initial conditions), history-independent (action depends only on the current state), stationary (time independent), and deterministic (always select the same action in a given state). Such policies can be described as mappings of states to probability distributions over actions $\pi : \mathcal{S} \times \mathcal{A} \mapsto [0, 1]$, where π_{ia} defines the probability that the agent will execute action a in state i.[1]

A policy π and the initial conditions $\alpha : \mathcal{S} \mapsto [0, 1]$ that specify the probability distribution over the state space at time 0 (the agent starts in state i with probability α_i) together determine the evolution of the system and the total expected discounted reward the agent will receive:

$$U_\gamma(\pi, \alpha) = \sum_{t=0}^{\infty} \gamma^t \sum_{i,a} \varphi_i(t) \pi_{ia} r_{ia}, \tag{1}$$

where $\varphi_i(t)$ refers to the probability of being in state i at time t, and $\gamma \in [0, 1)$ is the discount factor.

A standard [14, 9] way of solving MDPs that is well-suited for modeling resource constraints is to formulate the problem as the following linear program (this maximization LP is the dual to the more-commonly used minimization LP in the value function coordinates):

$$\max \sum_{i,a} r_{ia} x_{ia} \left| \begin{array}{l} \sum_a x_{ja} - \gamma \sum_{i,a} x_{ia} p_{iaj} = \alpha_j, \\ x_{ia} \geq 0 \end{array} \right. \tag{2}$$

The set of optimization variables x_{ia} is often called the *occupation measure* of a policy, where x_{ia} can be interpreted as the total expected discounted number of times action a is executed in state i. Then, $\sum_a x_{ia}$ gives the total expected discounted *flow* through state i, and the constraints in the above LP can be interpreted as the conservation of flow through each of the states.

An optimal policy can be computed from a solution to the above LP as: $\pi_{ia} = x_{ia} / \sum_a x_{ia}$. Although this appears to lead to randomized policies, in the absence of external constraints, and if we use strictly positive initial conditions ($\alpha_i > 0$), a basic feasible solution to this LP always maps to a deterministic policy [14, 9]. The above LP also produces policies that are *uniformly-optimal*, i.e., optimal for all initial distributions. However, the solution (x_{ia}) to the LP (eq. 2) retains its interpretation as the expected discounted number of times action a is executed in state i only for the initial probability distribution α that was used in the LP.

We now introduce our model of weakly-coupled constrained MDPs with limited shared resources, which is a slightly rephrased version of the constrained agent model used in

[1]We use degenerate distributions for representing deterministic policies (instead of the more standard mappings of states to actions), because that notation maps more naturally to other concepts used in this paper.

[6]. As mentioned earlier, weakly-coupled MDPs are widely used [11, 17] to model agents that only affect each other through the shared resources, and once a resource allocation is fixed, they act completely independently.

There are two types of constraints our model needs to represent: the global resource scarcity constraints, and the agents' local limitations on the combinations of resources they can use. To represent the latter, we use the concept of a *capacity*: we say that agents' actions require resources (the total amounts of which are limited), and that each resource (if used by an agent) consumes some of the agent's limited capacities. For example, delivery agents might contend for packages to be delivered (limited resources), and each agent might have bounds on the total size and weight of packages that it can deliver in one trip (limited capacities).

To simplify the discussion, throughout the main part of the paper we will assume that agents' resource requirements are binary, i.e., an action either uses a unit of a resource or does not use it at all. This does not greatly limit the generality of our model, because arbitrary integer resource requirements can be reduced to binary ones by expanding the resource set. It is also possible to avoid this expansion of the input by directly generalizing our model and algorithms to integer-valued bundles, as described in Section 8.

More formally, given a set of agents $\mathcal{M}$ who share a set of indivisible resources $\mathcal{O}$, a weakly-coupled multiagent constrained MDP has the following components, where $\mathcal{C}$ refers to the set of agents' capacities:

- $\Lambda^m = \langle \mathcal{S}, \mathcal{A}, p^m, r^m, \alpha^m \rangle$ is the standard unconstrained MDP of agent m. To simplify notation, we assume that agents have the same state and action spaces, but different transition and reward functions, and initial conditions.
- $\rho^m : \mathcal{A} \times \mathcal{O} \mapsto \{0, 1\}$ defines action resource requirements; ρ^m_{ao} specifies whether action a of agent m needs resource o in order to be executable.
- $\widehat{\rho} : \mathcal{O} \mapsto \mathbb{R}$ defines global resource bounds; $\widehat{\rho}_o$ gives the total amount of resource o available to the agents.
- $\kappa : \mathcal{O} \times \mathcal{C} \mapsto \mathbb{R}$ defines capacity costs of resources; κ_{oc} is the amount of capacity c consumed by a unit of resource o.
- $\widehat{\kappa}^m : \mathcal{C} \mapsto \mathbb{R}$ defines capacity bounds for each agent; $\widehat{\kappa}^m_c$ is the upper bound on capacity c for agent m.

We include the initial conditions α in the MDP of each agent, because unlike unconstrained MDPs, where uniformly-optimal policies always exist, this is not true for constrained problems [6]. From the resource-allocation perspective, this means that the value of a resource bundle depends not only on the rewards and dynamics of an agent's MDP, but also on the initial conditions.

To summarize, given the above model, our task is to design a computationally and economically efficient mechanism for allocating resources $\mathcal{O}$ (subject to scarcity constraints $\widehat{\rho}_o$) among self-interested agents $\mathcal{M}$, given that each agent $m \in \mathcal{M}$ will be using the resources to selfishly maximize the expected total discounted reward of its MDP Λ^m, subject to individual capacity constraints $\widehat{\kappa}^m_c$.

2.2 Combinatorial Auctions

The problem of finding an optimal resource allocation among self-interested agents that have complex valuations over combinations of resources arises in many different domains (e.g., [7, 19]) and is often called a *combinatorial allocation problem*. A natural and widely-used mechanism for solving such problems is a *combinatorial auction* (CA) (e.g., [5]). In a CA, each agent submits bids for a set of resource bundles to the auctioneer, who then decides what resources each agent will get and at what price.

Let us consider the problem of allocating among a set of agents $\mathcal{M}$ a set of indivisible resources $\mathcal{O}$, where the total quantity of resource $o \in \mathcal{O}$ is bounded by $\widehat{\rho}_o$. Our earlier assumption that actions' resource requirements are binary implies that agents will only be interested in bundles that contain no more than one unit of a particular resource.

In a combinatorial auction, each agent $m \in \mathcal{M}$ submits a bid b^m_w (specifying how much the agent is willing to pay) for every bundle $w \in \mathcal{W}^m$ that has some value $u^m_w > 0$ to the agent. After collecting all the bids, the auctioneer solves the winner-determination problem (WDP), a solution to which prescribes how the resource bundles should be distributed among the agents and at what prices. If agent m wins bundle w at price q^m_w, its utility is $u^m_w - q^m_w$ (we are assuming risk-neutral agents with quasi-linear utility functions). Thus, the optimal bidding strategy of an agent depends on how the auctioneer allocates the resources and sets prices.

A Generalized Vickrey Auction (GVA) [10], which is an extension of Vickrey-Clarke-Groves mechanisms [18, 4, 8] to combinatorial auctions, allocates resources and sets prices as follows. Given the bids of all agents, the auctioneer chooses an allocation that maximizes the sum of agents' bids. This problem is NP-complete [15] and can be expressed as the following integer program, where the optimization variables $z^m_w = \{0, 1\}$ are indicator variables that show whether bundle w is assigned to agent m, and $n_{wo} = \{0, 1\}$ specifies whether bundle w contains o:

$$\max \sum_{m \in \mathcal{M}} \sum_{w \in \mathcal{W}^m} z^m_w b^m_w \left| \begin{array}{l} \sum_{w \in \mathcal{W}^m} z^m_w \leq 1 \\ \sum_{m \in \mathcal{M}} \sum_{w \in \mathcal{W}^m} z^m_w n_{wo} \leq \widehat{\rho}_o, \end{array} \right. \quad (3)$$

The first constraint in (eq. 3) says that no agent can receive more than one bundle, and the second constraint ensures that the total amount of resource o assigned to the agents does not exceed the total amount available.

A GVA assigns resources according to the optimal solution $\widetilde{z}$ to (eq. 3) and sets the payment for agent m to:

$$q^m_w = V^*_{-m} - \sum_{m' \neq m} \widetilde{z}^{m'}_w b^{m'}_w, \quad (4)$$

where V^*_{-m} is the value of (eq. 3) if m were to not participate in the auction (the optimal value if m does not submit any bids), and the second term is the sum of other agents' bids in the solution $\widetilde{z}$ to the WDP with m participating.

A GVA has a number of nice properties. It is *strategy-proof*, meaning that the dominant strategy of every agent is to bid its true value for every bundle: $b^m_w = u^m_w$. The auction is economically efficient, meaning that it allocates the resources to maximize the social welfare of the agents (because, when agents bid their true values, the objective function of (eq. 3) becomes the social welfare). Finally, a GVA satisfies the *participation constraint*, meaning that the agents never decrease their utility by participating in the auction. A weak point of GVAs is their computational complexity, as discussed in more detail in the next section.

3. NAIVE GVA DESIGN

In this section, we describe a straightforward implementation of a GVA for our MDP-based problem.

Let each agent $m \in \mathcal{M}$ enumerate all possible resource bundles $\mathcal{W}^m$ that satisfy its local capacity constraints ($\widehat{\kappa}_c^m$). For each bundle $w \in \mathcal{W}^m$, agent m would determine the feasible action set $\mathcal{A}(w)$ and formulate an MDP $\Lambda^m(w) = \langle \mathcal{S}, \mathcal{A}(w), p^m(w), r^m(w), \alpha^m \rangle$, where $p^m(w)$ and $r^m(w)$ are projections of the agent's transition and reward functions onto $\mathcal{A}(w)$. Every agent would then solve each $\Lambda^m(w)$ corresponding to a feasible bundle to find the optimal policy $\widetilde{\pi}^m(w)$, whose expected discounted reward would define the value of bundle w: $u_w^m = U_\gamma^m(\widetilde{\pi}^m(w), \alpha^m)$.

Then, the auction proceeds in the standard GVA manner. The agents submit their bids b_w^m to the auctioneer, who solves the standard winner-determination problem (eq. 3) and sells the resources to the agents at prices (eq. 4). Since this is just a special case of a GVA, the strategy-proof property implies that agents would not deviate from bidding their true values $u_w^m = U_\gamma^m(\widetilde{\pi}^m(w), \alpha^m)$, and the mechanism will yield socially-optimal resource allocations.

This mechanism suffers from two major complexity problems. First, the agents have to enumerate an exponential number of resource bundles and compute the value of each by solving the corresponding MDP. Second, the auctioneer has to solve an NP-complete winner-determination problem on exponential input. The next two sections are devoted to tackling these complexity problems.

4. AVOIDING BUNDLE ENUMERATION

Let us begin by considering the agents' valuation problem. We can trivially simplify the agents' lives by creating an auction where they submit the specifications of their constrained MDPs to the auctioneer as bids. This is an extreme application of the revelation principle that moves the burden of solving the valuation problem from the agents to the auctioneer. Clearly, by itself this not only leads to absolutely no efficiency gains, but it also has implications on information privacy issues, because the agents have to reveal their local MDPs to the auctioneer (which they might not want to do). Nevertheless, we can build on this idea to increase the efficiency of solving both the valuation and winner-determination problems and do so without sacrificing agents' privacy. We address ways of maintaining information privacy in Section 6 and focus on the valuation and the winner-determination problems in this section.

The question we pose in this section is as follows. Given that the bid of each agent consists of its MDP and resource information, can the auctioneer formulate and solve the winner-determination problem more efficiently than by simply enumerating each agent's resource bundles and solving the standard exponentially-sized integer program (eq. 3)?

Our approach is based on the mixed integer linear programming (MILP) method described in [6] for optimal centralized resource allocation and policy formulation for fully cooperative agents. Here, we briefly summarize that method and discuss its use in the context of our auction mechanism.

Given that each agent submits to the auctioneer its MDP $\Lambda^m = \langle \mathcal{S}, \mathcal{A}, p^m, r^m, \alpha^m \rangle$, its resource requirements ρ^m, and its capacity bounds $\widehat{\kappa}^m$, the valuation and the winner-determination problems can be combined into one constrained optimization problem, which will solve the policy optimization and resource allocation problems simultaneously. This problem can be expressed in terms of the occupation measures x_{ia}^m of the agents as follows:

$$\max \sum_{m,i,a} x_{ia}^m r_{ia}^m \left| \begin{array}{l} \sum_a x_{ja}^m - \gamma \sum_{i,a} x_{ia}^m p_{iaj}^m = \alpha_j^m \\ \sum_o \kappa_{oc} H\big(\rho_{ao}^m \sum_i x_{ia}^m\big) \le \widehat{\kappa}_c^m \\ \sum_m H\big(\rho_{ao}^m \sum_i x_{ia}^m\big) \le \widehat{\rho}_o \\ x_{ia}^m \ge 0 \end{array} \right. \quad (5)$$

where H is the Heaviside "step" function of a nonnegative argument, defined as $H(0) = 0$, and $H(z) = 1$ for $z > 0$. The first and last sets of constraints in (eq. 5) are the standard conservation of flow and non-negativity constraints on the occupation measure of agent m (as in (eq. 2)). The second set constrains the combinations of resources that can be assigned to agent m by ensuring that its capacity limits $\widehat{\kappa}^m$ are not exceeded. Notice that the resource costs of a policy are expressed via the step function, which models the fact that resources are non-consumable, i.e., regardless of how many times action a is executed, the policy will require ρ_{ao}^m units of resource o as long as $\rho_{ao}^m \sum_i x_{ia}^m > 0$. The third set of constraints models resource scarcity: the total amount of resource o assigned to the agents cannot exceed $\widehat{\rho}_o$.

As shown in [6], problem (eq. 5) is NP-complete, but can be reduced to a mixed integer program, which can be solved using a variety of efficient tools. The nonlinearity of (eq. 5) can be alleviated by appropriately normalizing the occupation measure x and augmenting the program with a set of binary variables $\Delta_o^m \in \{0, 1\}$, where

$$\Delta_o^m = H\Big(\sum_a \rho_{ao}^m \sum_i x_{ia}^m\Big)$$

is an indicator variable that shows whether agent m plans to use resource o. Using these variables, resource and capacity constraints can be expressed as linear functions of Δ_o^m. Furthermore, the relationship between Δ_o^m and x_{ia}^m can be maintained via a set of linear inequality constraints, yielding the following MILP:

$$\max \sum_{m,i,a} x_{ia}^m r_{ia}^m \left| \begin{array}{l} \sum_a x_{ja}^m - \gamma \sum_{i,a} x_{ia}^m p_{iaj}^m = \dfrac{\alpha_j^m}{X} \\ \sum_o \kappa_{oc} \Delta_o^m \le \widehat{\kappa}_c^m \\ \sum_m \Delta_o^m \le \widehat{\rho}_o \\ \sum_a \rho_{ao}^m \sum_i x_{ia}^m \le \Delta_o^m \\ x_{ia}^m \ge 0, \quad \Delta_o^m \in \{0, 1\} \end{array} \right. \quad (6)$$

where $X \ge \max \sum_a \rho_{ao} \sum_i x_{ia}$ is an upper bound on the argument of $H(\cdot)$, used for normalization. The bound is guaranteed to exist for discounted MDPs and can be obtained in polynomial time [6]. The first three sets of constraints have the same meaning as in (eq. 5), and the fourth set of constraints ties binary resource-usage indicator variables Δ_o^m and the occupation measures x_{ia}^m.

The MILP (eq. 6) allows the auctioneer to efficiently solve the WDP without having to enumerate the possible resource bundles. As compared to the standard WDP formulation

(eq. 3), which has on the order of $|\mathcal{M}|2^{|\mathcal{O}|}$ binary variables, (eq. 6) has only $|\mathcal{M}||\mathcal{O}|$ binary variables. This exponential reduction is attained by exploiting the knowledge of the agents' MDP-based valuations and combining the policy optimization and resource allocation problems.

The resulting mechanism is an instantiation of the GVA, so by the well-known properties of VCG mechanisms, this auction is strategy-proof (the agents have no incentive to lie to the auctioneer about their MDPs), it attains the socially optimal resource allocation, and the agents never decrease their utility by participating in the auction.

To sum up the results of this section: by having the agents submit their MDP information to the auctioneer instead of their valuations over resource bundles, we have essentially removed all computational burden from the agents and at the same time significantly simplified the auctioneer's winner-determination problem (the number of integer optimization variables is reduced exponentially). As mentioned earlier, the approach raises some information privacy concerns, and we will discuss them in Section 6.

5. DISTRIBUTING THE WDP

Unlike the straightforward GVA implementation discussed earlier, where the agents shared some computational burden with the auctioneer, in the mechanism from the previous section, the agents submit their information to the auctioneer and then just idle while waiting for a solution. This suggests further potential efficiency improvements. Indeed, given the complexity of MILPs, it would be beneficial to exploit the computational power of the agents to offload some of the computation from the auctioneer back to the agents (we assume that agents have no cost for "helping out" and would prefer for the outcome to be computed faster).[2] Thus, the goal of this section is to distribute the computation of the winner-determination problem (eq. 6).

For concreteness, we will base the algorithm of this section on the branch and bound method for solving MILPs [20], but exactly the same techniques will also work for other MILP algorithms (e.g., cutting planes) that perform a search in the space of *LP relaxations* of the MILP. In branch and bound for MILPs with binary variables, LP relaxations are created by choosing a binary variable and setting it to either 0 or 1, and relaxing the integrality constraints of other binary variables. If a solution to an LP relaxation happens to be integer-valued, it provides a lower bound on the value of the global solution. A non-integer solution provides an upper bound for the current subproblem, which (combined with other lower bounds) is used to prune the search space.

Thus, a simple way for the auctioneer to distribute the branch and bound algorithm is to simply farm out LP relaxations to other agents and ask them to solve the LPs. However, it is easy to see that this mechanism is not strategy-proof. Indeed, an agent that is tasked with solving parts of the global winner-determination problem could benefit from lying to the auctioneer, i.e., the agent might be better off optimizing a function that differs from the social welfare. This is a common phenomenon in distributed mechanism implementations (e.g., [13]): whenever some WDP calculations are offloaded to an agent participating in the auction, the agent might be able to benefit from sabotaging the computation. As suggested by Parkes [13], there are several approaches to ensuring the strategy-proofness of a distributed implementation. The approach best suited for our problem is based on the idea of *redundant computation*, where multiple agents are asked to do the same task and any disagreement is carefully punished to discourage lying. In the rest of this section, we demonstrate that this is very easy to implement in our case.

The basic idea is very simple: let the auctioneer distribute LP relaxations to the agents, but check their solutions and re-solve the problems if the agents return incorrect solutions (this would make truthful computation a weakly-dominant strategy for the agents, and a nonzero punishment can be used to achieve strong dominance). This strategy of the auctioneer removes the incentive for the agents to lie and yields exactly the same solution as the centralized algorithm. However, in order for this to be beneficial, the complexity of checking a solution must be significantly lower than the complexity of solving the problem. Fortunately, this is true for LPs due to a well-known property.

Suppose the auctioneer has to solve the following LP, which can be written in two equivalent ways (let us refer to the one on the left as the primal, and to the one on the right as the dual):

$$\min \boldsymbol{\alpha}^T \mathbf{v} \;\Big|\; A^T \mathbf{v} \geq \mathbf{r}\,, \quad \max \mathbf{r}^T \mathbf{x} \;\Big|\; \begin{array}{l} A\mathbf{x} = \boldsymbol{\alpha} \\ \mathbf{x} \geq 0 \end{array} \tag{7}$$

By the strong duality property, if the primal LP has a solution $\mathbf{v}^*$, then the dual also has a solution $\mathbf{x}^*$, and $\boldsymbol{\alpha}^T \mathbf{v}^* = \mathbf{r}^T \mathbf{x}^*$. Furthermore, given a solution to the primal LP, it is easy to compute a solution to the dual: by complimentary slackness, $\mathbf{v}^{*T} = \mathbf{r}^T B^{-1}$ and $\mathbf{x}^* = B^{-1}\boldsymbol{\alpha}$, where B is a square invertible matrix, composed of columns of A that correspond to basic variables of the solution.

These well-known properties can be used by the auctioneer to quickly check optimality of solutions returned by the agents. Suppose that an agent returns $\mathbf{v}$ as a solution to the primal LP. The auctioneer can calculate the dual solution $\mathbf{v}^T = \mathbf{r}^T B^{-1}$ and check whether $\mathbf{r}^T \mathbf{x} = \boldsymbol{\alpha}^T \mathbf{v}$. Thus, the most expensive operation that the auctioneer has to perform is the inversion of B, which can be done in sub-cubic time. As a matter of fact, from the implementation perspective, it would be more efficient to ask the agents to return both the primal and the dual solutions, since many popular algorithms compute both in the process of solving LPs.

Thus, we have provided a simple method that allows us to effectively distribute the winner-determination problem, while maintaining strategy-proofness of the mechanism and with a negligible computation overhead for the auctioneer.

6. PRESERVING INFORMATION PRIVACY

The mechanism discussed earlier has the drawback that it requires agents to reveal complete information about their MDPs to the auctioneer. The problem is also exacerbated by the distributed WDP algorithm from the previous section, since not only does each agent reveal its MDP information to the auctioneer, but that information is then also spread to other agents via the LP relaxations of the global MILP. In this section we try to alleviate this problem.

Let us note that, in saying that agents prefer not to re-

[2] As observed by Parkes [13], this assumption is a bit controversial, since a desire for efficient computation implies nonzero cost for computation, while the agents' cost for "helping out" is not modeled. It is, nonetheless, a common assumption in distributed mechanism implementations.

veal their local information, we are implicitly assuming that there is an external factor that affects agents' utilities, and it depends on how much information is revealed. The value of information is typically measured by how it changes one's decision-making process and its outcomes. Since this effect is not part of our model (in fact, it contradicts our weak-coupling assumption), we cannot in a domain-independent manner define what constitutes "useful" information, and how bad it is for an agent to reveal too much about its MDP. Modeling such effects and carefully analyzing them is an interesting research task, but it is outside the scope of this paper. Thus, for the purposes of this section, we will be content with a mechanism that hides enough information to make it impossible for the auctioneer or an agent to uniquely determine the transition or reward function of any other agent (in fact, information revealed to any agent will map to infinitely many MDPs of other agents).[3]

The main idea of our approach is to modify the previous mechanism so that the agents submit their private information to the auctioneer in an "encrypted" form that will allow the auctioneer to solve the winner-determination problem, but will not allow it to infer agents' original MDPs.

Let us observe that, instead of passing an MDP to the auctioneer, each agent can submit an equivalent LP (eq. 2). So, the question becomes: can the agent transform its LP in such a way that the auctioneer will be able to solve it, but will not be able to infer the transition and reward functions of the originating MDP? In other words, the problem reduces to the following. Given an LP L_1 (created from an MDP $\Lambda = \langle \mathcal{S}, \mathcal{A}, p, r \rangle$ via (eq. 2)), we need to find a transformation $L_1 \rightarrow L_2$ such that a solution to the transformed LP L_2 will uniquely map to a solution to the original LP L_1, but L_2 will not reveal the transition or the reward functions of the original MDP (p or r). We show that a simple change of variables suffices.

Suppose agent m_1 has an MDP-originated LP and is going to ask agent m_2 to solve it. In order to maintain the linearity of the problem (to keep it simple for m_2 to solve), m_1 should limit itself to linear transformations. Consider a linear, invertible, and positive (to maintain the sign of inequality constraints) transformation of the primal coordinates $\mathbf{u} = F\mathbf{v}$, and a linear, invertible transformation of the dual coordinates $\mathbf{y} = D\mathbf{x}$. Then, the LP from (eq. 7) will be transformed (by applying F, switching to the dual, and then applying D) to an equivalent LP in the new coordinates $\mathbf{y}$:

$$\max \mathbf{r}^T D^{-1}\mathbf{y} \left| \begin{array}{l} (F^{-1})^T A D^{-1}\mathbf{y} = (F^{-1})^T \boldsymbol{\alpha} \\ D^{-1}\mathbf{y} \geq 0 \end{array} \right. \quad (8)$$

The value of the optimal solution to (eq. 8) will be the same as the value of the optimal solution to (eq. 7), and given an optimal solution $\mathbf{y}^*$ to (eq. 8), it is easy to compute the solution to the original: $\mathbf{x}^* = D^{-1}\mathbf{y}^*$. Indeed, from the perspective of the dual, the primal transformation F is equivalent to multiplying the dual equality constraints $A\mathbf{x} = \alpha$ by constants and adding some equations together, which (given that F is positive and invertible) has no effect on the solution or the objective function. Furthermore, the dual transformation D is equivalent to a change of variables that modifies the solution but not the value of the objective function.

However, a problem with the above transformations is that it gives away D^{-1}. Indeed, agent m_2 will be able to simply read (up to a set of multiplicative constants) the transformation off the constraints $D^{-1}\mathbf{y} \geq 0$. Therefore, only diagonal matrices with positive coefficients (which are equivalent to stretching the coordinate system) are not trivially deduced by m_2, since they also map to $\mathbf{y} \geq 0$. Choosing a negative multiplier for some x_i (inverting the axis) is pointless, because that flips the non-negativity constraints to $y_i \leq 0$, immediately revealing the sign to m_2.

Let us demonstrate that, given any MDP Λ and the corresponding LP L_1, we can choose D and F such that it will be impossible for m_2 to determine the coefficients of L_1 (or equivalently the original transition and reward functions p and r). When agent m_2 receives L_2 (as in (eq. 8)), all it knows is that L_2 was created from an MDP, so the columns of the constraint matrix of the original LP L_1 must sum to a constant:

$$\sum_j A_{ji} = 1 - \gamma \sum_j p_{iaj} = 1 - \gamma. \quad (9)$$

This gives m_2 a system of $|\mathcal{S}|$ nonlinear equations for the diagonal D and arbitrary F, which have a total of $|\mathcal{S}||\mathcal{A}|+|\mathcal{S}|^2$ free parameters. For everything but the most degenerate cases (which can be easily handled by an appropriate choice of D and F), these equations are hugely under-constrained and will have infinitely many solutions. As a matter of fact, by sacrificing $|\mathcal{S}|$ of the free parameters, m_1 can choose D and F in such a way that the columns of constraints in L_2 will also sum to a constant $c \in (0, 1)$, which would have the effect of transforming L_1 to an L_2 that corresponds to another valid MDP Λ_2. Therefore, given an L_2 there are infinitely many original MDPs Λ and transformations D, F that map to the same LP L_2.

One aspect of the auctioneer's problem that we have not considered so far is the connection of resource and capacity costs to agents' occupation measures in the global WDP (eq. 6). There are essentially two things that the auctioneer has to be able to do: 1) determine the value of each agent's policy (to be able to maximize the social welfare), and 2) determine the resource requirements of the policies (to check the resource constraints). So, the question is, how does our transformation affect these? As noted earlier, the transformation does not change the objective function, so the first requirement holds. On the other hand, D does change the occupation measure x^m_{ia} by arbitrary multipliers. However, a multiplicative factor of x^m_{ia} has no effect on the usage of non-consumable resources, as it only matters whether the corresponding x^m_{ia} is zero or not (step function H nullifies the scaling effect). Thus, the second condition also holds.

To sum up, we can, to a large extent, maintain information privacy in our mechanism by allowing agents to apply linear transformations to their original LPs. The information that is revealed by our mechanism consists of agents' resource costs ρ^m_{ao}, capacity bounds $\widehat{\kappa}^m_c$, and the sizes of their state and action spaces.[4] The revealed information can be used to infer agents' preferences and resource requirements. Further, numeric policies are revealed, but the lack of information about transition and reward functions makes it difficult to map that to meaningful courses of action.

[3]A more stringent condition would require that agents' preferences over resource bundles are not revealed [12], but we set a lower bar here.

[4]Resource information is revealed for non-consumable resources, but can be hidden for consumable ones (Section 8).

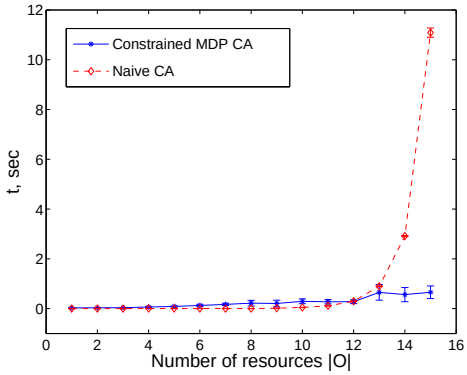

Figure 1: Gains in Computation Efficiency: MDP-based GVA versus a naïve GVA implementation.

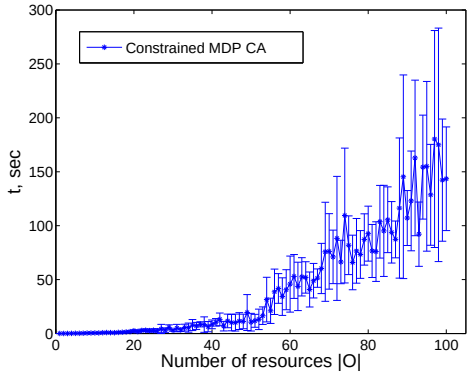

Figure 2: Scaling the MDP-based WDP to larger problems.

7. EMPIRICAL EVALUATION

A thorough empirical evaluation of our mechanism would analyze it from the perspectives of both information revelation and computational efficiency. However, as discussed earlier, the model of agents' interactions we used in this work is not rich enough to allow us to properly quantify the amount of revealed information. Therefore, we will focus solely on evaluating the mechanism from the computational efficiency perspective.

For our experiments, we implemented a simple multiagent delivery problem (based on the multiagent rover problem in [6]), where several agents operate in a grid world with delivery locations randomly placed throughout the grid. There is a limited number of delivery contracts to be fulfilled by the agents, where each contract consists of delivering a set of packages to a location on the grid. Each package has some weight, and each agent has a bound on how much weight it can carry (limited capacity). Agents (delivery vehicles) also differ in size: the bigger the agent, the more packages it can carry (thus capable of making more deliveries in one trip), but the more costly it is to operate (bigger trucks need more gas). The agents' movements through the grid have a stochastic component to it. The agents participate in an auction where they bid on delivery contracts. In this setting, the value of a contract depends on what other contracts the agent acquires (e.g., it is beneficial to obtain contracts for locations that are geographically close, because that reduces the expenses). Given a set of contracts, an agent's policy optimization problem is to find the optimal delivery route.[5]

We compared the performance of our MDP-based auction (Section 4) to the performance of the standard GVA with flat preferences (Section 3). The results are summarized in Figure 1, which compares the time it takes (using CPLEX 8.1 on a P-4) to solve the standard winner-determination problem on the space of all resource bundles (eq. 3) to the time needed to solve the combined MDP-WDP problem (eq. 6) used in our mechanism (without the additional speedup of parallelization of Section 5), as the number of resources is increased (with 5 agents, on a 5-by-5 grid). Despite the fact that both algorithms have exponential worst-case running time, the number of integer variables in (eq. 3) is exponentially larger than in our MILP (eq. 6), the effect of which is clearly demonstrated in Figure 1. This comparison gives an optimistic view of the performance of the standard CA, as it does not take into account the additional complexity of solving the valuation problem, which is embedded into the WDP of our mechanism (eq. 6). Including the time for solving the valuation problem in the comparison only magnifies the effect. No parallelization of the WDP was performed for these experiments for either algorithm.

We also evaluated our method on larger problems infeasible for the standard CA. Figure 2 shows that in under five minutes we could solve problems that, in the standard CA, would require the agents to enumerate up to 2^{100} bundles (yielding as many MDPs to solve), and the auctioneer to solve an NP-complete WDP with an input of that size.

8. GENERALIZATIONS AND DISCUSSION

We made a number of assumptions in this paper. Some of them are fundamental to the work, such as weak-coupling of the agents, without which the value of a resource bundle to an agent might depend on other agents' bundles, which violates an important GVA assumption. Other assumptions can be relaxed without too much effort, as discussed below.

Our work can be easily extended to handle consumable resources that are used up whenever agents execute actions. In fact, the problem can be considerably simplified for domains with only these kinds of resources. The most important change is that we have to redefine the value of a particular resource bundle to an agent. The difficulty is that, given a policy, the total use of consumable resources is uncertain. One possibility is to define the value of a bundle as the payoff of the best policy whose *expected* resource usage does not exceed the amounts of resources in the bundle. The interpretation of ρ^m_{ao} would also change to mean the amount of resource o consumed by action a every time it is executed. This would make the constraints in (eq. 6) *linear* in the occupation measure, which would tremendously simplify the WDP (making it polynomial). Information privacy can be handled similarly to the case of non-consumable resources. However, given the transformation $\mathbf{y} = D\mathbf{x}$, the resource cost function ρ^m will also have to be scaled by D^{-1} (since the total consumption of consumable resources is proportional to the occupation measure). This has the additional benefit of hiding the resource cost functions (unlike the case of non-consumable resources where they were revealed).

[5]We also investigated other, randomly generated domains, and the results were qualitatively the same.

In this work, we assumed that $\rho_{ao}^m = \{0, 1\}$, implying that agents are only interested in binary resource bundles. The model and algorithms can be easily generalized to integer resource costs ρ_{ao}^m. To accommodate this, we would not have to change anything, except the reduction of the WDP (eq. 5) to the MILP (eq. 6). To handle integer ρ^m, we will need to introduce binary indicator variables Δ_a^m (instead of Δ_o^m) that show whether agent m uses action a in its policy (instead of resource o). The linearization of (eq. 5) would also change, and would require $|\mathcal{M}| \prod_o n_o$ capacity constraints, where n_o is the number of actions that use resource o (instead of $|\mathcal{M}||\mathcal{O}|$ for the binary costs).

To summarize the results of this paper, we presented a computationally-efficient variant of a GVA for resource allocation among self-interested agents whose valuations of resource bundles are defined by their weakly-coupled constrained MDPs. For such problems, we showed analytically and empirically that our mechanism, which exploits knowledge of the structure of agents' MDP-based preferences, achieves exponential speedup over a straightforward GVA implementation, and does so without sacrificing any of the nice properties of GVAs (optimal outcomes, strategy-proofness, and voluntary participation). We also discussed a distributed implementation of the mechanism that retains strategy-proofness (using the fact that an LP solution can be easily verified), and does not reveal agents' private MDP information (using a transformation of agents' MDPs).

However, as mentioned earlier, in order to carefully study information revelation in this setting, it is necessary to expand our model to define how revealed information affects agents' behavior, which is an interesting direction of future work. In this work, we considered MDPs with "flat" state and action spaces, which do not scale well due to the curse of dimensionality. In order for our mechanism to be applicable to real-world MDP instances, we need to modify it to work with *factored* MDPs [2], and the approximate linear programming methodology [16] seems particularly well-suited for this purpose. Finally, we have not given special treatment to the problem of price determination. It can be easily solved by repeatedly applying exactly the same sorts of techniques as for the WDP, but an interesting question is whether this can be done more effectively than just running our algorithm on $|\mathcal{M}|$ separate sub-problems.

9. ACKNOWLEDGMENTS

This material is based upon work supported by Honeywell International, and by the DARPA/IPTO COORDINATORs program and the Air Force Research Laboratory under Contract No. FA8750–05–C–0030. The views and conclusions contained in this document are those of the authors, and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.Thanks to Jianhui Wu and the anonymous reviewers for helpful comments and suggestions.

10. REFERENCES

[1] C. Boutilier. Solving concisely expressed combinatorial auction problems. In *Eighteenth national conference on Artificial intelligence*, pages 359–366, 2002.

[2] C. Boutilier, R. Dearden, and M. Goldszmidt. Exploiting structure in policy construction. In *Proc. of the 14th Int. Joint Conf. on Artificial Intelligence*, pages 1104–1111. Morgan Kaufmann, 1995.

[3] C. Boutilier and H. H. Hoos. Bidding languages for combinatorial auctions. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, pages 1211–1217, 2001.

[4] E. H. Clarke. Multipart pricing of public goods. *Public Choice*, 18:19–33, 1971.

[5] S. de Vries and R. V. Vohra. Combinatorial auctions: A survey. *INFORMS Journal on Computing*, 15(3):284–309, 2003.

[6] D. A. Dolgov and E. H. Durfee. Optimal resource allocation and policy formulation in loosely-coupled Markov decision processes. In *Proc. of the 14th Int. Conf. on Automated Planning and Scheduling*, 2004.

[7] D. Ferguson, C. Nikolaou, J. Sairamesh, and Y. Yemini. Economic models for allocating resources in computer systems. In S. Clearwater, editor, *Market-Based Control: A Paradigm for Distributed Resource Allocation*, 1996.

[8] T. Groves. Incentives in teams. *Econometrica*, 41(4):617 – 631, 1973.

[9] L. Kallenberg. *Linear Programming and Finite Markovian Control Problems*. Math. Centrum, Amsterdam, 1983.

[10] J. K. MacKie-Mason and H. Varian. Generalized Vickrey auctions. Technical report, University of Michigan, 1994.

[11] N. Meuleau, M. Hauskrecht, K.-E. Kim, L. Peshkin, L. Kaelbling, T. Dean, and C. Boutilier. Solving very large weakly coupled Markov decision processes. In *AAAI/IAAI*, pages 165–172, 1998.

[12] D. Parkes. *Iterative Combinatorial Auctions: Achieving Economic and Computational Efficiency*. PhD thesis, Department of Computer and Information Science, University of Pennsylvania, May 2001.

[13] D. C. Parkes and J. Shneidman. Distributed implementations of Vickrey-Clarke-Groves mechanisms. In *Proc. of 3rd Int. Joint Conf. on Autonomous Agents and Multi Agent Systems*, 2004.

[14] M. L. Puterman. *Markov Decision Processes*. John Wiley & Sons, New York, 1994.

[15] M. H. Rothkopf, A. Pekec, and R. M. Harstad. Computationally manageable combinational auctions. *Management Science*, 44(8):1131–1147, 1998.

[16] P. Schweitzer and A. Seidmann. Generalized polynomial approximations in Markovian decision processes. *Journal of Math. Analysis and Applications*, 110:568 – 582, 1985.

[17] S. Singh and D. Cohn. How to dynamically merge Markov decision processes. In M. I. Jordan, M. J. Kearns, and S. A. Solla, editors, *Advances in Neural Information Processing Systems*, volume 10, 1998.

[18] W. Vickrey. Counterspeculation, auctions and competitive sealed tenders. *J. of Finance*, 1961.

[19] M. Wellman, W. Walsh, P. Wurman, and J. MacKie-Mason. Auction protocols for decentralized scheduling. *Games and Economic Behavior*, 35:271–303, 2001.

[20] L. Wolsey. *Integer Programming*. John Wiley & Sons, 1998.