Linear Mechanisms for Single-Parameter Domains: Characterization, Existence, and Construction

Victor Naroditskiy Maria Polukarov Nicholas R. Jennings

School of Electronics and Computer Science,
University of Southampton, United Kingdom
{vn,mp3,nrj}@ecs.soton.ac.uk

Abstract

We give sufficient conditions for the existence of piecewise linear optimal mechanisms in single-parameter domains, and identify a rich class of mechanism design problems that satisfy these conditions. Specifically, we consider anonymous settings where the allocation is *constant-dependent*: i.e., determined by a set of hyperplanes $v_i = c_j$, where v_i is agent i's type and c_j is some constant. Our proof is constructive and yields a general procedure for finding an optimal mechanism for any given problem in this class.

1 Introduction

Mechanism design, traditionally studied in economics, is now a rapidly growing field in computer science (e.g., see [12, 16]). Generally, it deals with problems where multiple self-interested participants take actions to optimize individual utilities based on the *incentives* provided to them. The overlap of computer science and mechanism design is natural: reasoning about incentives is unavoidable in many fundamental computer science problems—e.g., peer-to-peer networks, packet routing, and resource allocation. To this end, the term *algorithmic mechanism design* was introduced by Nisan and Ronen [11] to refer to the study of incentives in computer science scenarios. At the same time, some classic mechanism design solutions rely on computationally efficient approximations and implementations to be of practical use (e.g., computationally efficient VCG-like combinatorial auctions [5]).

This paper interfaces computer science and mechanism design in yet another way: instead of considering a particular domain and designing a mechanism with certain properties, we are looking for a general, unified technique that would take a mechanism design problem as an input, and output a procedure for finding an optimal mechanism to this problem. While this goal, in general, may sound unrealistic, this work shows it can be effectively achieved for a wide class of problems in "single-parameter domains" where agents have private types expressed by single numbers. Examples include well-known fundamental settings, such as surplus-maximizing resource allocation [10, 7] or fair task imposition [13], as well as novel applications, such as allocation with costs, that have not been previously studied. Furthermore, our—at first glance, purely algorithmic—approach enables derivation of theoretical results that provide a base for the following contributions:

- **Characterization.** We formulate sufficient conditions for the existence of (piecewise) linear optimal mechanisms in single-parameter domains (Theorem 2).

- **Existence.** We identify a rich class of mechanism design problems characterized by a "constant-dependent" allocation function (to be defined), and prove the above conditions hold for each problem in this class (Theorem 3).

- **Construction.** We develop an algorithm that finds an optimal mechanism for any given problem in this class (Theorem 2 and Figure 4).

We start with preliminaries and related work in Section 2, and present our main theorems in Section 3. These results provide a general and powerful tool for analysis of mechanism design problems in single-parameter domains: as the existence and construction results apply to *any* problem with constant-dependent allocation, the algorithm for an optimal mechanism is unified within this class of problems. In Section 4, we demonstrate the strength of this approach on two central mechanism design problems of (1) surplus-maximizing resource allocation and (2) fair task imposition. First, we easily re-derive the mechanisms for surplus-maximizing allocation of free items by Moulin [10] and Guo and Conitzer [7] and fair imposition of a single task by Porter *et al.* [13]. Second, we use our method to obtain an optimal mechanism for fair imposition of multiple tasks, for which no closed form has been previously found. Furthermore, in Section 5 we apply our technique to open problems. Specifically, we extend the consideration to scenarios where objects have costs and provide first algorithms for computing optimal mechanisms for surplus maximizing allocation and fair imposition in these generalized settings. Finally, our work suggests several directions for future research outlined in Section 6.

2 Preliminaries and related work

Informally, a *mechanism* refers to a procedure for making decisions (or, choices) involving multiple agents. Suppose one item needs to be allocated among a group of agents. A mechanism might collect bids from each agent, give the item to the highest bidder, and charge him his bid: this mechanism is a first-price auction, and the choice made defines an allocation.

Mechanism design is concerned with finding the best way of making decisions in a given scenario (e.g., allocation). The "best" way is specified by the properties the decision must satisfy: e.g., the allocation should be fair, the agent with the highest value should be allocated, the revenue of the seller must be maximized. Crucially, quality of a decision depends on private information called *types* of the agents (e.g., their values for the item). Therefore, a mechanism must ensure the agents have the incentive to reveal their types truthfully: without knowing the true types, there is no way to know how good the decision is. We study the so-called "strategy-proof" mechanisms that make it in each agent's best interest to truthfully reveal his type—regardless of whether the other agents do so or not. This—the strongest—concept of truthfulness is called *dominant-strategy*.

Implementation in dominant strategies is virtually impossible without restrictions on agents' types. In fact, for unrestricted types (i.e., different value for each possible choice) only dictatorial choice functions are implementable [6]. One way to get out of the impossibility is by introducing monetary payments which are added to agent's valuation of the chosen alternative: in this case, agents are said to have *quasi-linear utilities*. A mechanism is therefore defined by a choice rule f and a payment scheme t—both are functions of the agents' types.

However, even with money (a.k.a. transferable utilities), the set of implementable mechanisms is rather limited—the only such mechanisms are weighted VCG [14], which motivates further restrictions on agents' types. In this work, we focus on single-parameter domains: we restrict our attention to allocation domains where agent's type represents the value for being allocated—in this context, it is intuitive to refer to the choice rule as the *allocation function*. It is known that in these settings, any allocation function that is

monotone[1] in the agent's report, is implementable:

Theorem 1 (e.g., see [12] p. 229) *A mechanism (f, t) is implementable if and only if for each agent i:* (i) *f_i is monotone in v_i;* (ii) *$t_i = h(v_{-i}) - \tau(v_{-i})$ if $f_i = 1$ (i.e., i is allocated) and $t_i = h(v_{-i})$ otherwise ($f_i = 0$), where $\tau(v_{-i}) = \sup_{v_i | f_i(v_i, v_{-i})=0} v_i$ defines the* critical value[2].

Thus, any pair of functions (f, t) that satisfy the conditions in Theorem 1, defines a strategy-proof mechanism, and so truthfulness is guaranteed. In this work, we take a monotone allocation function as an input and look for an optimal (according to provided properties) payment function of the form above. We develop a general algorithmic method for finding optimal payments for an important class of allocation functions we term "constant-dependent": in particular, *efficient*[3] allocation functions in settings of consideration fall in this class.

A dominant-strategy mechanism does not make any assumptions about the values of the agents: the desirable properties (e.g., efficiency, individual rationality or no subsidy) of the mechanism must hold for all possible values the agents may have. These properties can be expressed as a system of constraints to be met for each possible profile v of agents' valuations and (optionally) an objective function (e.g., revenue maximization). Thus, a mechanism design problem is defined by an optimization program over payment functions t:

$$\begin{aligned} &\text{optimize}_{t:\mathbb{R}^n \to \mathbb{R}^n} \text{ objective value} \quad \text{s.t.} \quad \forall\, v \in \mathbb{R}^n \\ &\text{objective value is achieved} \\ &\text{properties hold} \end{aligned}$$

At the first glance, this problem is hard: optimization is over functions and there is an infinite number of constraints. However, in this paper we propose an algorithmic approach that allows to effectively tackle such problems in single-parameter domains. Our technique exploits and makes explicit the linearity structure present in many standard mechanism design problems. As a result, the existence of (piecewise) linear optimal mechanisms follows immediately. For some important application problems, this method offers an easy way to find the optimal payment function analytically, by solving a simple system of linear equations (see, for example, the results in 4.2). For a general problem in this class, it provides an algorithm for finding an optimal mechanism computationally.

Most related to our work is the literature on optimizing rebates in VCG mechanisms. In [10] and [7], the authors independently discover the optimal VCG redistribution mechanism for allocating free homogenous items. VCG redistribution schemes have also been designed for a public good domain [1]. A similar result has been derived in [2] in the context of allocating a single item. Alternatively, the objective of fairness was considered in [13] for task imposition scenarios. In this paper, we provide a general approach for addressing all of these problems.

The model of allocating homogeneous items that have costs was considered in [3, 9], although for a different purpose—to compare "random priority" and "average cost" mechanisms. We are the first to obtain (algorithmic) solutions for surplus-maximization and fairness in this setting.

We are aware of only one another attempt to approach mechanism design problems algorithmically—that of Automated Mechanism Design (AMD) [15]. However, AMD applies when the space of agents' types is finite and a prior over the types is available. In contrast, we deal with infinite types spaces and no priors. Our method is based on partitioning the space of value profiles into a finite number of convex regions, on each of which, as we prove, a linear optimal payment function can be defined. A similar idea was exploited in [4]; however, there the partitioning is heuristic and does not result in an optimal mechanism.

[1] In words, monotonicity of f_i in v_i means that if an agent is allocated when he reports v_i he is also allocated when he reports $v_i' \geq v_i$.

[2] An agent i is allocated if and only if his report is above the critical value $\tau(v_{-i})$.

[3] An allocation is efficient if the items are assigned to the agents who value them the most.

3 Optimization, Linearity and Partition

In this section, we present our main results: Theorem 2 provides sufficient conditions for the existence of a piecewise linear optimal mechanism, and Theorem 3 constructively proves these conditions holds for the class of constant-dependent allocations (to be defined). We start by formally stating the problem in 3.1 and explaining the idea of our solution in 3.2. The theorems appear in 3.3 and 3.4.

3.1 Setting

We consider *single parameter* domains where each of n agents desires one unit of a (homogenous) good, and $v \in \mathbb{R}^n_+$ represents the agents' valuations for consuming the good (or, *item*). Monetary transfers are possible, and agents' utilities are quasi-linear. The value profiles are such that $v_1 \geq v_2 \geq \ldots \geq v_n$ (this is without loss of generality for anonymous mechanisms), and since the values are non-negative, one can scale all vectors to be in the interval $[0, 1]$. We denote the space of value profiles by $V = \{v \in \mathbb{R}^n \mid 1 \geq v_1 \geq v_2 \geq \ldots \geq v_n \geq 0\}$ and define vector $v_{-i} \in \mathbb{R}^{n-1}$ to indicate values of the agents other than i. The space of all such $(n-1)$-dimensional vectors is the same for each i, and is denoted by W.

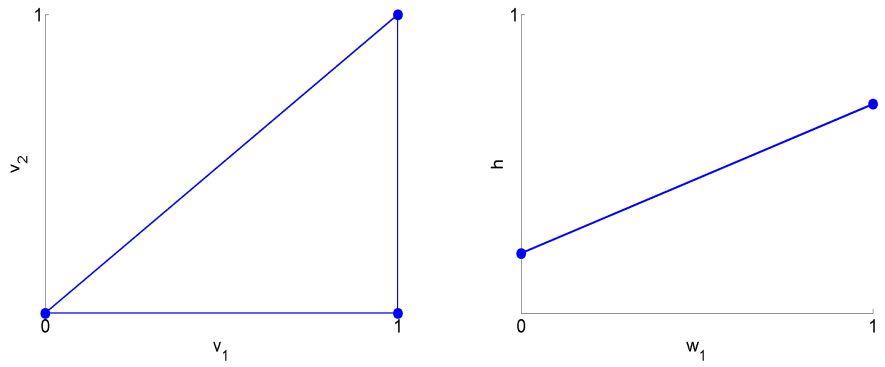
An *outcome* is a pair $(\mathrm{f}, \mathrm{t}) \in \{0,1\}^n \times \mathbb{R}^n$, where f_i indicates whether agent i is *allocated* (gets the item), and t_i represents the *payment* he receives (t_i can be negative, in which case agent i pays that amount); the total utility of agent i from the outcome (f, t) is given by $u_i = \mathrm{f}_i v_i + \mathrm{t}_i$. A *mechanism* is defined by a pair of functions $f : \mathbb{R}^n_+ \to \{0,1\}^n$ and $t : \mathbb{R}^n_+ \to \mathbb{R}^n$ that determine the allocation and payments for each possible report from the agents regarding their value for the item. We take as an input an allocation function f satisfying the monotonicity condition in Theorem 1. This determines the critical value, $\tau(v_{-i})$, for each agent i, and the only remaining degree of freedom is the function $h(v_{-i})$ that adjusts payments to the agents. In some applications, it is intuitive to view τ as the price for being allocated and h as the rebate distributed back to all agents; henceforth, we refer to h as the *rebate function*.

Our goal is to find *optimal rebates* that guarantee the best possible value of a given objective function and satisfy given constraints for each possible vector of agents' valuations—that is, provide a dominant-strategy implementation. The objective of optimization may be, for example, maximization of social surplus (i.e., redistributing back as much of the budget surplus as possible when there is no auctioneer), some measure of fairness (e.g., maximizing the lowest utility), or minimization of budget deficit. Desirable properties of mechanisms (e.g., no subsidy, individual rationality or k-fairness) are specified as constraints in the optimization problem. Some combinations of properties (e.g., no subsidy and 2-fairness) may be impossible to implement: this is identified by the lack of a feasible solution to the optimization problem.

3.2 Linear properties

Our approach exploits the linear structure which characterizes standard mechanism design problems. Typical constraints (e.g., individual rationality, no subsidy, k-fairness) and objectives (e.g., utility maximization, deficit minimization) are linear in values and payments of the agents. For example, the no subsidy (or weak budget balance) constraint requires the *sum* of payments to the agents to be non-positive; utilitarian objective function maximizes the *sum* of agents' values and payments. This linearity structure lies at the heart of the idea presented next.

Consider the following illustrating example. Recall that we are after an optimal rebate function $h : \mathbb{R}^{n-1} \to \mathbb{R}$: in the simplest case with 2 agents, the domain of the rebate function is the real interval between 0 and 1. The space of values is a triangle given by the extreme points $(0,0)$, $(1,0)$, and $(1,1)$ shown in Figure 1(a). Suppose that allocation is fixed for all profiles of values (e.g., agent 1 is always allocated and agent 2 is never allocated) and that constraints are linear in v and h. Indeed, h is a function of v itself, but it is a variable in our problem. For a finite set of points $v \in V$ we can talk of a finite set of values of the

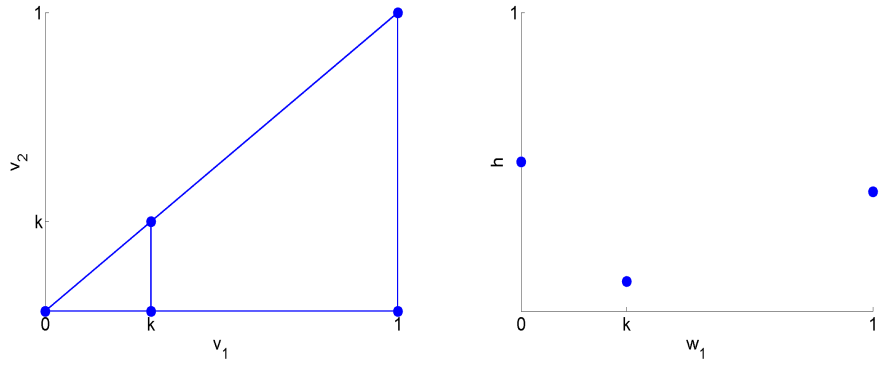


(a) Allocation region. (b) Rebate function.

Figure 1: Single allocation region and optimal linear rebate function.

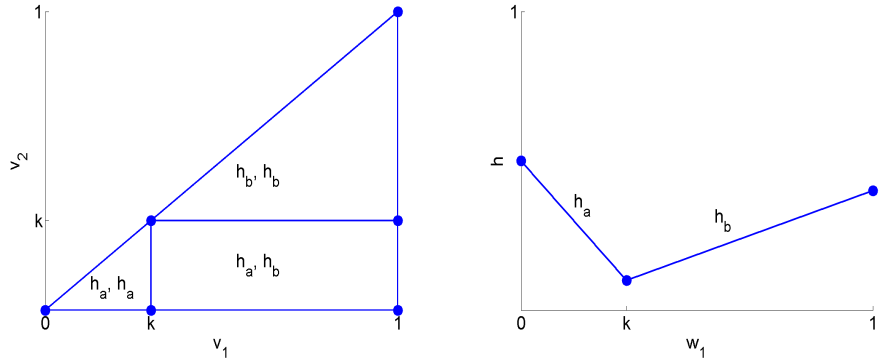
rebate function $h(w)$, where w corresponds to v_{-i} for agent i. It is easy to show that a linear constraint is satisfied everywhere on a convex region if and only if it is satisfied on its extreme points: in our example, enforcing linear constraints on the profiles $(0,0)$, $(1,0)$ and $(1,1)$ guarantees that they are satisfied for all profiles $v \in \mathbb{R}^2 \mid 1 \geq v_1 \geq v_2 \geq 0$. Note that constraints for these profiles involve exactly two rebates $h(0)$ and $h(1)$: thereby, restricting the optimization problem to constraints for these extreme profiles gives a linear program with two variables, which we call $\hat{h}(0)$ and $\hat{h}(1)$. Also, since this restricted linear program includes only a subset of constraints from the original mechanism design problem, its optimal objective value provides an upper (in the case of maximization) bound on the objective value of the original problem (in problems with no objectives, if the original problem has a feasible solution, so does the restricted problem.) Now, having solved the *restricted problem* by computing the rebates $\hat{h}(0)$ and $\hat{h}(1)$, the equation of the line on which these two points lie provides us with a—linear—rebate function h: that is, for an arbitrary point $w \in W$, we can define $h(w) = a_1 w_1 + b$, where the coefficients a_1, b are obtained by solving the system of two equations: for $(0, \hat{h}(0))$ and $(1, \hat{h}(1))$. The rebate function is the line segment connecting points $(0, \hat{h}(0))$ and $(1, \hat{h}(1))$ (see Figure 1(b)). *This function is linear in v, so all constraints remain linear.* These constraints are satisfied on the extreme points of a convex region, and therefore hold everywhere on this region. Thus, we can "expand" an optimal solution to the restricted problem to a feasible solution to the original problem, and achieve the same objective value. Since the objective value of the restricted problem was an upper bound on the objective of the original problem, the constructed solution is optimal and the upper bound it tight. Finally, note that we were able to linearly combine rebate values $h(0)$ and $h(1)$ in the rebate space because there were exactly two (i.e., n) of them.

In more general cases, allocation may not be linear on the whole value space. For instance, consider the allocation rule that allocates to agent 1 if his value is above $k \in (0,1)$ and never allocates to agent 2. The value space is partitioned into 2 allocation regions: agent 1 is not allocated in the region to the left of $v_1 = k$ and is allocated in the region to the right (see Figure 2(a)). Constraints for the extreme points $(0,0)$, $(k,0)$, $(1,0)$, (k,k), and $(1,1)$ of the allocation regions include three rebates $\hat{h}(0)$, $\hat{h}(k)$, and $\hat{h}(1)$. Proceeding as we did in the previous example we would have to linearly connect the values of these rebates. However, in general, three points do not lie on the same line, as is illustrated in Figure 2(b). A natural idea is to define two linear rebate functions: one connecting $\hat{h}(0)$ to $\hat{h}(k)$ and the other connecting $\hat{h}(k)$ to $\hat{h}(1)$ (see Figure 3(b)). We refer to these functions as h_a and h_b: thus, $h(w) = h_a(w)$ if $0 \leq w_1 \leq k$ and $h(w) = h_b(w)$ if $k \leq w_1 \leq 1$. In order for constraints to be linear on a region, for each agent i the choice of the rebate function (h_a or h_b) must be constant throughout the region. The allocation region to the right of $v_1 = k$ does not satisfy this condition: the rebate for agent 1 is given by h_a for $v_2 \leq k$ and by h_b for



(a) Two allocation regions. (b) Relevant rebate values.

Figure 2: Two allocation regions and rebates in corresponding extreme points.



(a) Value regions. (b) Rebate function.

Figure 3: Refinement of allocation regions and optimal piecewise linear rebate function.

$v_2 \geq k$. However, we can *refine* the allocation regions along $v_2 = k$ to fix this problem. In Figure 3(a), the regions are labeled with the rebate function used by each agent. Partitioning along $v_2 = k$ introduced a new extreme point: $(1, k)$. However, $\hat{h}(0)$, $\hat{h}(k)$, and $\hat{h}(1)$ are still the only rebates used by constraints on the extreme points. Two line segments passing through the points $(0, \hat{h}(0))$, $(k, \hat{h}(k))$ and $(k, \hat{h}(k))$, $(1, \hat{h}(1))$, respectively, define the—piecewise linear—rebate function (see Figure 3(b)). As before, this implies that the constraints are linear in v on each region of this—refined—partition, and since they are satisfied on the extreme points of each region, they hold for all points of each region.

Next, we generalize this idea and formalize conditions on partitions into regions of the value space V and the rebate space W, which we prove to be sufficient for the existence of an optimal mechanism with piecewise linear rebates.

3.3 Linearly consistent partitions

We need the following definitions.

Definition 1 *A set P_X of polytopes is called a* partition *of the polytope X if the polytopes do not overlap: $p \cap q = \varnothing, \ \forall\, p, q \in P_X$, and cover exactly the polytope X: $\bigcup_{p \in P_X} p = X$.*

Definition 2 *The partition P_X* refines *the set of polytopes Q if for all $p \in P_X,\ q \in Q$, their intersection is either empty or p: $p \cap q = \varnothing \vee p \cap q = p$.*

We are interested in partitions that consist of convex polytopes. A convex d-dimensional polytope p can be defined as a finite intersection of halfspaces: $p = \{x \in \mathbb{R}^d \mid Ax \geq b\}$, where $A \in \mathbb{R}^{k \times d}$, $b \in \mathbb{R}^k$, k is the number of halfspaces.[4] In the examples we provided, the rebate space is 1-dimensional and intersections of halfspaces specify line segments. The partition P_W in Figure 3(b) is given by 2 polytopes $k \geq w_1 \geq 0$ and $1 \geq w_1 \geq k$. Recall the corresponding partition P_V of the value space V in Figure 3(a). Crucially, for each agent i, the choice of the rebate function is fixed on each *value region* $q \in P_V$ (see Figure 3(a)). Stating the property mathematically, we obtain $\forall q \in P_V,\ \forall i \in \{1, \ldots, n\}$ there exists $p \in P_W \mid v_{-i} \in p,\ \forall v \in q$. Observation 1 below characterizes the partitions of the value space that satisfy this property: it notes, that P_V must refine a set of polytopes *lift*(P_W) which is obtained by "lifting" the partition P_W of the $n-1$ dimensional space W to the n-dimensional space V.

Definition 3 *A set of polytopes* lift(P_W) *in the value space V is said to be obtained by* lifting *the partition P_W of the rebate space W if*

$$\text{lift}(P_W) = \bigcup_{(A,b) \in P_W} \bigcup_{i=1}^{n} V \cap (Av_{-i} \geq b)$$

Note that each polytope $(A, b) \in P_W$ adds n (possibly overlapping) polytopes to *lift*(P_W). In our example in Figures 3(a) and 3(b), lifting the polytope $k \geq w_1 \geq 0$ yields overlapping polytopes $k \geq v_1 \geq v_2 \geq 0$ and $1 \geq v_1 \geq v_2 \geq 0;\ k \geq v_2$.

Observation 1 *Let P_V and P_W be partitions of the value and the rebate space, respectively. The condition $\forall q \in P_V,\ \forall i \in [1, n],\ \exists p \in P_W \mid v_{-i} \in p,\ \forall v \in q$ is satisfied when P_V refines* lift(P_W).

Next we derive additional conditions that would let us define rebate functions h_p for *rebate regions $p \in P_W$* so that each such function is linear and $\{h_p \,|\, p \in P_W\}$ is optimal. For linearity, we need each polytope p in the rebate partition to have exactly n extreme points. For optimality, we need to guarantee the linearity of constraints on each of the value regions. This is formally stated in Definition 4 and below. Finally, Theorem 2 shows the sufficiency of these conditions.

We refer to the union of extreme points of the partition P_X of a polytope X as $\hat{P}_X$. Given the partition P_V of the value space, the *projection* of its extreme points into the rebate space W is defined as follows:

$$\Pi_W(\hat{P}_V) = \bigcup_{v \in \hat{P}_V} \bigcup_{i=1}^{n} v_{-i}$$

Definition 4 *Partitions P_V and P_W are called* linearly consistent *if:* (i) *all polytopes $q \in P_V$ and $p \in P_W$ are convex;* (ii) $\Pi_W(\hat{P}_V) = \hat{P}_W$*;* (iii) *$P_V$ refines both the set of polytopes* lift(P_W) *and the allocation partition P_V^a as defined by the allocation function f*[5]*; and* (iv) *each polytope in P_W has n extreme points.*

Consider the graph of a rebate function $(w, h(w))$. Note that any n rebate values can be described by a linear rebate function: indeed, there exists an $(n-1)$ dimensional hyperplane passing through n points in $\mathbb{R}^n$. Therefore, partitioning the rebate space into polytopes each having n extreme points lets us define a linear rebate function on each polytope. Finally, setting the rebates on extreme points $\hat{P}_W$ accordingly to the optimal solution to the optimization problem, restricted to $\hat{P}_V$, implies the optimality of rebates for linearly consistent partitions.

[4] Thus, a pair (A, b) defines a polytope in $\mathbb{R}^d$.

[5] That is, for any v^1, v^2 in the same allocation region $q^a \in P_V^a$, $f(v^1) = f(v^2)$.

Theorem 2 *Given linearly consistent partitions P_V and P_W, let $\{\hat{h}(w) \mid w \in \hat{P}_W\}$ denote the set of rebates from an optimal solution to the restricted problem and let $\hat{p}$ denote the set of n extreme points of a polytope $p \in P_W$. For each polytope, define a linear rebate function $h_p(w) = \sum_{i=1}^{n-1} a_i^p w_i + b^p$ with coefficients $a^p \in \mathbb{R}^{n-1}, b^p \in \mathbb{R}$ given by a solution to the system of linear equations $\{\hat{h}(w) = \sum_{i=1}^{n-1} a_i^p w_i + b^p \mid w \in \hat{p}\}$. Then, the following rebate function is* optimal*: for $w \in p$, $h(w) = h_p(w)$.*

By Theorem 2, if one can partition spaces V and W in a linearly consistent way, an optimal, piecewise linear, mechanism follows immediately. Next, we present an algorithm for finding such partitions for the important class of, what we call, "constant-dependent" allocation functions: these, in particular, include commonly desirable efficient allocations.

3.4 Constant-dependent allocations

We start with a definition.

Definition 5 *The allocation function is called constant-dependent if there exists a finite set of constants $C = \{c_1, \ldots, c_q\}$, so that the allocation is fixed on each of the regions defined by a set of hyperplanes $v_i = c_j$, $i \in \{1, \ldots, n\}$, $j \in \{1, \ldots, q\}$. For $C = \varnothing$, it is fixed on the whole space of agents' valuations.*

Obviously, a constant-dependent allocation function is monotone, and Theorem 1 holds. In Figure 4, we present the *partition* function, and prove it defines linearly-consistent partitions of the value and rebate spaces for a given constant-dependent allocation function (Theorem 3).

Algorithm *partition*
Input: polytope X
1. partition X along $x_i = c \quad \forall c \in C,\ i \in \{1, \ldots, dim(X)\}$ /* denote the partition by P_X^g */
2. **for** each hyperrectangle $p \in P_X^g$
   for each pair (i, j) of dimensions $i, j \in \{1, \ldots, dim(X)\},\ i \neq j$,
   partition p along $x_i = ax_j + b$ where $a, b \in \mathbb{R}$ define the diagonal
   from the lower left to the upper right corner of projection onto the i-j plane

Figure 4: Linearly-consistent partitions.

Theorem 3 *For a constant-dependent allocation, the partitions $P_V = \text{partition}(V)$ and $P_W = \text{partition}(W)$ are linearly-consistent.*

In the following sections, we demonstrate that Theorems 2 and 3 provide an algorithmic technique for finding optimal mechanisms for a wide class of mechanism design problems. In particular, we consider the surplus-maximizing allocation and fair imposition problems and show that our method provides an easy way of obtaining mechanisms for the (previously studied) case with free objects. Moreover, uniqueness of the mechanisms follows immediately. These results are presented in Section 4. Finally, in Section 5, we extend the consideration to the open problem where items have costs and show that the efficient allocation in this generalized setting is constant-dependent, and so Theorems 2 and 3 apply.

4 "Free" Homogeneous Objects

In this section, we apply our technique to two central mechanism design problems in single parameter domains. We start by re-deriving the results on surplus-maximizing allocation of free items by Moulin [10]

and Guo and Conitzer [7] and fair imposition of a single task by Porter *et al.* [13]. We then proceed to show that an optimal mechanism for fair imposition of multiple tasks, for which no closed form has been previously derived, can be easily obtained using our method.

4.1 Surplus-maximizing allocation

There are m identical items and n agents, who each needs (at most) one item. The items are (weakly) desirable, so we talk of distribution of "goods", and there is a rationing problem: each agent claims a unit but all claims cannot be met ($m < n$). Efficiency requires assigning the items to m agents who value them the most. Agents' valuations for consuming the item, $1 \geq v_1 \geq \ldots \geq v_n \geq 0$, are private information. To ensure truthful reporting (see Theorem 1), a payment to each agent i must be of the form $t_i(v) = h_i(v_{-i}) - \tau_i(v_{-i})$, where the critical value $\tau_i(v_{-i})$ equals v_{m+1} if agent i receives the item and 0 otherwise. Monetary transfers are subsidy-free—that is, the sum of payments is non-positive, $\sum_{i=1}^{n} t_i(v) \leq 0$, to guarantee no budget deficit. The agents' utilities are quasi-linear and need to satisfy the individual rationality, $u_i(v) \geq 0$, to provide incentives for agents to participate in the mechanism.

Under no subsidy, the goal is to minimize the budget imbalance (the efficiency loss) of the mechanism or, equivalently, to maximize the (utilitarian) surplus it achieves. Note, however, that the absolute surplus realized by the mechanism is not an appropriate measure of its performance since it does not show how far this value is from the first-best solution, i.e. the maximal surplus one could achieve if the agents' values were known. In order to have an index that is unit-free (i.e. homogenous of degree zero), it is natural to use a ratio. Finally, since the agents' values are not known and no prior is available, we consider a worst-case index. We therefore, use the following *surplus ratio* to evaluate the performance of a truthfully implemented mechanism:

$$S = \min_{v \in V} \frac{\sum_{i \leq m} v_i - m v_{m+1} + \sum_{i=1}^{n} h(v_{-i})}{\sum_{i \leq m} v_i}$$

Finding a mechanism whose surplus ratio (henceforth, the *ratio*) is S means that a proportion S of the efficient allocation (i.e., maximum social surplus) is achieved, *independently* of what the agents' values are.

Below we formally state the problem of finding the surplus-maximizing allocation mechanism. In order to remove minimization over all value profiles from the objective function, we introduce additional variable S and require it not to exceed the surplus ratio for each profile:

$$\max_{S \in \mathbb{R},\ r:\mathbb{R}^{n-1} \to \mathbb{R}} S \quad \text{s.t.} \quad \forall\, v \in V \tag{1}$$

$$\sum_{i=1}^{n} h(v_{-i}) - m v_{m+1} \leq 0 \tag{2}$$

$$v_i - v_{m+1} + h(v_{-i}) \geq 0 \qquad \forall i \leq m \tag{3}$$

$$h(v_{-i}) \geq 0 \qquad \forall i > m \tag{4}$$

$$\sum_{i \leq m} v_i - m v_{m+1} + \sum_{i=1}^{n} h(v_{-i}) \geq S \sum_{i \leq m} v_i \tag{5}$$

The no-subsidy property is enforced in (2): the payments collected from the agents create no budget deficit. Inequalities (3), (4) guarantee that the utility of each agent is non-negative.[6] Finally, (5) ensures that the ratio is satisfied under all value profiles.

Note that the allocation is fixed on the whole space of value profiles V, and all the constraints are linear in v and h. Moreover, we notice that restricting the optimization program (1)-(5) to the natural set of extreme

[6]Notice that since for $i \leq m$ we have $v_i \geq v_{m+1}$, it follows that constraint (4) is, in general, stronger than constraint (3). However, for some value profiles we will get $v_1 = \ldots = v_m = v_{m+1}$, canceling out the term $v_i - v_{m+1}$ in (3) for all $i \leq m$. Since the constraints must be satisfied for all possible value vectors, it is therefore equivalent to simply require (4) for all i.

points of the value space, $\hat{V} = V \cap \{0,1\}^n$, we get the system of inequalities with exactly n variables h_x, $x = 0, 1, \ldots, n-1$, that correspond to rebates $h(w^x)$ where w^x is an $(n-1)$-dimensional non-decreasing binary vector with x ones: $w^x \in \hat{W} = W \cap \{0,1\}^{n-1}$, $\sum_{i=1}^{n-1} w_i^x = x$. By Theorem 3, this immediately implies the existence of an optimal linear solution.

To find it, we first solve the restricted optimization problem defined by (1)-(5) on $\hat{V}$ (note that $\hat{V}$ contains only $n+1$ vectors, and so the solution is found efficiently). This provides us with rebate values for n points in $\hat{W}$. Now, for any $w \in W$, the rebate is defined by a linear equation $h(w) = \sum_{i=1}^{n-1} a_i w_i + b$, where the coefficients are determined by points in $\hat{W}$ (i.e., by a solution to the system $\{h(w) = \sum_{i=1}^{n-1} a_i w_i + b \,|\, w \in \hat{W}\}$ of n linear equations with n variables).

Our approach is conceptually different from the one proposed by Guo and Conitzer [7], though they also use linear programming. Briefly, they start with a feasible solution to the optimization problem (1)-(5)—this solution provides a lower bound on the objective value, and then show its optimality; in contrast, we obtain an upper bound by solving a restricted problem and use it to find a feasible (and thus, optimal) solution to the original problem.

The optimal linear solution to this problem was derived analytically by Moulin [10] and Guo and Conitzer [7]. While our characterization of this solution is algorithmic, it was very easy to obtain. The uniqueness results that had to be proven in [10, 7], follow immediately in our case. Next, we describe a setting where applying our technique also yields a closed-form solution.

4.2 Fair imposition

In this setting, one wishes to fairly assign tasks to agents each having private information about the level of effort required from the agent to perform the task. The net payment to the agents is non-positive (that is, there is no budget deficit), and the assignment is efficient (agents with lowest levels of required effort get the tasks). Since the agents are *obligated* to provide the service and make payments to the center, the standard constraint of individual rationality is replaced with k-fairness, which requires the agents' disutilities to be "as low as possible"—namely, no greater than $\frac{m}{n}$ of the k^{th} lowest level of effort. Finally, strategy-proofness has to be satisfied as both efficiency and k-fairness depend on the true agents' types.

For convenience, we consider this problem in the context of "distribution of goods" rather than "imposition of bads"—this can be done equivalently by viewing tasks as items with negative values.[7] We choose this interpretation as 1) it is consistent with the previous allocation example, and 2) it provides a more natural motivation to our further extension of both models to the case where items come at a cost (see Section 5).[8]

Formally, there is a fixed number, m, of items to be distributed among n agents, with private values $1 \geq v_1 \geq \ldots \geq v_n \geq 0$ and quasi-linear utilities. We are looking for a strategy-proof efficient mechanism (f, t), that provides no budget deficit and k-fairness: the items are allocated to the first m agents, and a payment to each agent i is of the form $t_i(v) = h_i(v_{-i}) - \tau_i(v_{-i})$, where the critical value $\tau_i(v_{-i})$ equals v_{m+1} if agent i gets the item and 0 otherwise; the sum of payments is non-positive, $\sum_{i=1}^{n} t_i(v) \leq 0$, and each agent gets a "fairly high" utility, $u_i(v) \geq \frac{m v_k}{n}$.[9] Thus, there is a no-deficit k-fair mechanism if and only if there exists a rebate function $h : \mathbb{R}^{n-1} \to \mathbb{R}$ satisfying the constraints (6)-(8) below for each possible

[7] See discussion in [13].

[8] In this—extended—setting, we could alternatively (and equivalently) represent items with costs by tasks with benefits they provide to the society when completed; however, we find this interpretation somewhat less convenient.

[9] By Porter *et al.* [13], if there exists a strategy-proof, k-fair mechanism with no budget deficit, then there exists such a mechanism which is efficient.

value profile v. These constraints are linear in v and h:

$$v_i - v_{m+1} + h(v_{-i}) \geq \frac{mv_k}{n} \qquad \forall i \leq m \tag{6}$$

$$h(v_{-i}) \geq \frac{mv_k}{n} \qquad \forall i > m \tag{7}$$

$$\sum_{i=1}^{n} h(v_{-i}) \leq mv_{m+1} \tag{8}$$

We now show that the fair imposition problem can be easily solved using our method. We first re-derive the results by Porter *et al.* [13]—the impossibility theorem for $1, 2$-fairness and a 3-fair mechanism for a single task ($m = 1$), and then generalize these results to the case with multiple tasks: we show impossibility for $k \leq m + 1$ and find an $(m + 2)$-fair mechanism. Although, as [13] propose, this mechanism could be achieved by applying sequentially the Porter's protocol for a single task, our approach for the first time provides a "one-shot" solution in closed form. Moreover, our technique implies the uniqueness of a linear $(m + 2)$-fair mechanism.

As in the previous example, the allocation is fixed on the whole value space, and its natural set of extreme points $\hat{V} = V \cap \{0, 1\}^n$ defines a system of inequalities with exactly n variables h_x, $x = 0, 1, \ldots, n - 1$, that correspond to rebates $h(w^x)$ where w^x is an $(n - 1)$-dimensional non-decreasing binary vector with x ones: $w^x \in \hat{W} = W \cap \{0, 1\}^{n-1}$, $\sum_{i=1}^{n-1} w_i^x = x$. We therefore only need to find a feasible solution to this system of inequalities and then linearly extend the rebate values obtained from this solution: this way we, in particular, find an $(m + 2)$-fair mechanism. The impossibility result for $k \leq m + 1$ follows from the fact that the above system has no feasible solution.

4.2.1 Single task

In this case, $m = 1$. Let $k \leq 2$ and show there is no feasible solution to the system (6)-(8), even when restricted to vectors in $\hat{V}$. From (7) for $v^0 = (0, 0, \ldots, 0)$ we have $h_0 \geq \frac{v_k^0}{n} = 0$; coupled with (8) for $v^1 = (1, 0, \ldots, 0)$: $h_0 + (n - 1)h_1 \leq mv_2^1 = 0$, this implies that $h_1 \leq 0$. This contradicts the k-fairness constraint (7) for $i = 2$ and $v^2 = (1, 1, 0, \ldots, 0)$: $h_1 \geq \frac{v_k^1}{n} = \frac{1}{n} > 0$.

However, for $k = 3$ the system does have a feasible solution, and we can find a linear 3-fair mechanism. First, we solve the system for vectors in $\hat{V}$. From (7) and (8) for $v^n = (1, 1, \ldots, 1)$ we get $h_n = \frac{1}{n}$. Then, (7) and (8) for $v^{n-1} = (1, 1, \ldots, 1, 0)$ we have $h_{n-1} \geq \frac{1}{n}$ and $h_n + (n - 1)h_{n-1} = \frac{1}{n} + (n - 1)h_{n-1} \leq 1$, implying $h_{n-1} = \frac{1}{n}$. Proceeding this way, for any $x \geq 2$ we obtain $h_x = \frac{1}{n}$. Finally, from (7) and (8) for $v^0 = (0, 0, \ldots, 0)$ and $v^1 = (1, 0, \ldots, 0)$ we have $h_0 = h_1 = 0$. Note that this solution is unique.

Now, for any $w \in W$ its corresponding rebate is defined by the equation $h(w) = \sum_{i=1}^{n-1} a_i w_i + b$, where the coefficients are determined by the extreme points in $\hat{W}$:

$$\begin{aligned}
&h_0 = a_1 0 + a_2 0 + \ldots + a_{n-1} 0 + b && \Rightarrow b = 0\\
&h_1 = a_1 1 + a_2 0 + \ldots + a_{n-1} 0 + b && \Rightarrow a_1 = 0\\
&h_2 = a_1 1 + a_2 1 + a_3 0 + \ldots + a_{n-1} 0 + b && \Rightarrow a_2 = h_2 = \frac{1}{n}\\
&h_x = a_1 1 + a_2 1 + \ldots + a_x 1 + a_{x+1} 0 + \ldots + a_{n-1} 0 + b && \Rightarrow a_x = 0, \forall\, 3 \geq x \geq n - 1
\end{aligned}$$

Thus, for any $w \in W$, $h(w) = \frac{1}{n} w_2$, which coincides with the mechanism by Porter *et al.* [13].

We have demonstrated that a 3-fair mechanism can be obtained by solving simple systems of linear equations; moreover, our technique implies that such linear mechanism is unique. Furthermore, we show that a closed form solution can be similarly found for settings with multiple tasks.

4.2.2 Multiple tasks

The following proposition generalizes the results to the case with $m \geq 1$.

Proposition 1 *There is no strategy-proof mechanism that satisfies no-deficit and k-fairness for $k \leq m+1$. The unique efficient linear $(m+2)$-fair such mechanism is: for $i \leq m$, $f_i(v) = 1$, $t_i(v) = h(v_{-i}) - v_{m+1}$, and for $i > m$, $f_i(v) = 0$, $t_i(v) = h(v_{-i})$, where $h(w) = \frac{m}{n} w_{m+1}$.*

Proof Let $v^x \in \hat{V} = V \cap \{0,1\}^n$ such that $\sum_{i=1} v_i^x = x$, and assume $k \leq m+1$. From (7) for v^{m-1} and (8) for v^m it follows that $h_m \leq 0$. However, from (7) for v^{m+1}, $i = m+1$, we have $h_m \geq \frac{m}{n} > 0$, a contradiction.

Now let $k = m+2$. Solving the system (6)-(8) for vectors in $\hat{V}$, we obtain the following unique solution: $h_x = 0$ for $x = 0, \ldots, m$, and $h_x = \frac{m}{n}$ for $x = m+1, \ldots, n$. Now, solve $\{h(w) = \sum_{i=1}^{n-1} a_i w_i + b \,|\, w \in \hat{W}\}$ and get $a_{m+1} = \frac{m}{n}$, $a_x = b = 0$, where $x = 1, \ldots, m, m+2, \ldots, n$. Thus, we have $h(w) = \frac{m}{n} w_{m+1}$. □

5 Allocation with Costs

In this section, we apply our technique to solve open mechanism design problems. Specifically, we consider more realistic scenarios where items may come at a cost. This generalization significantly complicates the setting for both surplus-maximizing allocation and fair imposition problems, which have not been previously tackled for items with costs. We observe that the generalized model still falls in the framework of single-parameter domains with constant-dependent allocation, and Theorem 3 holds. Given this, we provide the first algorithm for computing optimal mechanisms for these scenarios.

5.1 Motivation

We consider a setting where (identical) items must be assigned to the agents, assuming each agent wants exactly one item, and the items have (increasing marginal) costs. The goal, as before, may be either to maximize the social surplus or to achieve k-fairness.

The allocation problem with increasing marginal costs is a simple and fundamental example of the problem of the commons [8], in which multiple participants, acting independently to optimize their own objectives, will ultimately deplete a shared limited resource even when it is clear that it is not in anyone's long-term interest for this to happen. Increasing marginal costs model decreasing returns to every agent as the number of allocated items increases. For instance, consider membership in a free gym. As the gym becomes more crowded, the utility each member derives from exercising there decreases. Membership in the gym corresponds to an item in our model. Cost of item i represents the marginal disutility of the members, which increases as the gym becomes more crowded.

Allocating items with increasing unit costs also arises in other familiar contexts, such as scheduling and disaster management. For example, consider multiple teams willing to be deployed in a disaster response. Each team has information (i.e., private value) about different regions of the affected area and can judge how much their region needs help. For teams to operate, they need communication frequencies for intra-team communication. The number of frequencies is limited and the more frequencies are allocated, the higher is the noise. The goal of a disaster response manager is to solicit truthful evaluations of situations in each team's region and to allocate frequencies to teams in the regions that need help the most. Additional frequencies should be allocated as long as the benefit derived from deploying an extra team outweighs the cost corresponding to the increase in noise on the communication channel. More generally, agents could be either emergency responders or sensor networks. The important part is that each agent is self-interested and maximizes its own utility, which is the case, for example, when agents are owned by different companies.

5.2 Setting

The setting of *allocation with costs* is defined by a triple $\langle n, c, v\rangle$, where n is the number of agents each desiring one unit of a homogenous good, c is the vector of marginal costs for producing each additional unit (item), and $v \in \mathbb{R}^n_+$ represents the agents' valuations for consuming the item. The marginal cost is increasing in the number of items, i.e. $c_1 \leq c_2 \leq \ldots \leq c_n$, and value profiles are such that $1 \geq v_1 \geq v_2 \geq \ldots \geq v_n \geq 0$. Monetary transfers are possible, and agents' utilities are quasi-linear.

In contrast to the case with free items, the number of allocated agents is *not* fixed but depends on c and v: we do not assign the item to an agent whose value for the item is lower than its cost. An efficient mechanism in this setting will maximize the total value of agents minus the total cost; the number of items allocated this way is $m(v, c) = \max_i(i \mid v_i \geq c_i)$ and the value of the efficient allocation is $\sum_{i \leq m(v,c)}(v_i - c_i)$.

Finally, we assume that at least one, but no more than $n-1$ items, are allocated: $c_1 < v_1$ and $c_n = 1$. It is easy to see that the efficient allocation in this setting is constant-dependent and defined by set $C = \{c_1, \ldots, c_{n-1}\}$. Hence, Theorem 3 implies.

5.3 Mechanisms

We now formulate the surplus-maximizing allocation and fair imposition problems in this domain. First, we modify the surplus ratio as follows:

$$S(c) = \min_{v \in V} \frac{\sum_{i=1}^{m(v,c)} v_i - m(v,c)\tau^a + \sum_{i=1}^{n} h(v_{-i})}{\sum_{i=1}^{m(v,c)}(v_i - c_i)}$$

where τ^a is the critical value of an allocated agent. Note that we fix the cost vector c and consider the worst ratio over all possible value profiles: we do not take the minimum over costs as that would obviously result in zero ratio—when the first $n-1$ costs are the same, the ratio is zero. The surplus-maximizing allocation problem is then defined by the following optimization program:

$$\max_{S \in \mathbb{R},\ r:\mathbb{R}^{n-1} \to \mathbb{R}} S \quad \text{s.t.} \quad \forall\, v \in V \tag{9}$$

$$m = \text{argmax}_i(v_i \geq c_i) \tag{10}$$

$$\tau^a = \max\{v_{m+1}, c_m\} \tag{11}$$

$$\sum_{i=1}^{n} h(v_{-i}) - m\tau^a \leq -\sum_{i \leq m} c_i \tag{12}$$

$$h(v_{-i}) \geq 0 \quad \forall i \tag{13}$$

$$\sum_{i \leq m}(v_i) - m\tau^a + \sum_{i=1}^{n} h(v_{-i}) \geq S \sum_{i \leq m}(v_i - c_i) \tag{14}$$

Here, (10) determines the number of items in an efficient allocation for the profile v, and corresponding critical values are defined by (11). The no-deficit property is enforced in (12): the payments collected from the agents must cover the costs of the allocated items. Constraint (13) guarantees that the utility of each agent is non-negative (recall from the previous section that enforcing the non-negativity of rebates is equivalent). Finally, as before, (14) ensures that the ratio is satisfied under all value profiles.

Similarly, we modify the fair imposition problem as follows: $\forall\ v \in V$,

$$m = \text{argmax}_i(v_i \geq c_i) \tag{15}$$

$$\tau^a = \max\{v_{m+1}, c_m\} \tag{16}$$

$$h(v_{-i}) \geq \frac{mv_k}{n} \quad \forall i \tag{17}$$

$$\sum_{i=1}^{n} h(v_{-i}) - m\tau^a \leq -\sum_{i \leq m} c_i \tag{18}$$

We have observed that the efficient allocation is constant-dependent in this model (as defined by the set of costs). Therefore, a *piecewise linear* surplus-maximizing and a k-fair mechanisms are obtained by solving (9)-(14) and (15)-(18), respectively, for the subset of profile values $\hat{V}$ as defined in 3, and linearly combining the rebate values in these—extreme—points on each of the regions of partition they define on space W.

6 Open Questions

Our work suggests several directions for future research. First, the characterization result in Theorem 2 can potentially be used to conclude the existence of linear optimal mechanisms in classes of problems, other than those with constant-dependent allocations: here, combinatorial auctions with single-minded bidders may be of particular interest; another extension is to public good settings. Second, a more general question in this context is about the necessity of conditions in Theorem 2. These conditions imply the existence of a partition of the space of agents' types, with certain properties: is it the case that if no such partition exists, an optimal linear mechanism does not exist either? Finally, an optimal partition may be complicated: in the setting of allocation with costs, the space is partitioned into $\binom{2n-2}{n-1}n!$ regions, where n is the number of agents. For small n, we empirically observed that most of the regions are required for an optimal mechanism, but it is likely that merging some of the regions does not decrease the solution quality too much. The tradeoff between efficiency and optimality remains open for further study.

References

[1] Martin J Bailey. The demand revealing process: To distribute the surplus. *Public Choice*, 91(2):107–26, April 1997.

[2] Ruggiero Cavallo. Optimal decision-making with minimal waste: Strategyproof redistribution of vcg payments. In *AAMAS'06*, Hakodate, Japan, 2006.

[3] H. Crès, H. Moulin, and HEC Groupe. Commons with increasing marginal costs: random priority versus average cost. *Int. econ. review*, 44:1097–1115, 2003.

[4] Geoffroy de Clippel, Victor Naroditskiy, and Amy Greenwald. Destroy to save. In *EC'09*, 2009.

[5] S. De Vries, R.V. Vohra, Center for Mathematical Studies in Economics, and Management Science. Combinatorial auctions: A survey. *INFORMS Journal on Computing*, 15(3):284–309, 2003.

[6] A. Gibbard. Manipulation of voting schemes: A general result. *Econometrica*, 41:587–601, 1973.

[7] Mingyu Guo and Vincent Conitzer. Worst-case optimal redistribution of vcg payments. In *EC'07*, pages 30–39, New York, NY, USA, 2007. ACM.

[8] G. Hardin. The tragedy of the commons. *Science*, 162(3859):1243–1248, December 1968.

[9] R. Juarez. The worst absolute surplus loss in the problem of commons: random priority versus average cost. *Economic Theory*, 34(1):69–84, 2008.

[10] Herve Moulin. Almost budget-balanced vcg mechanisms to assign multiple objects. *Journal of Economic Theory*, 144(1):96–119, 2009.

[11] Noam Nisan and Amir Ronen. Algorithmic mechanism design (extended abstract). In *STOC '99: Proceedings of the thirty-first annual ACM symposium on Theory of computing*, pages 129–140, New York, NY, USA, 1999. ACM.

[12] Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, New York, NY, USA, 2007.

[13] Ryan Porter, Yoav Shoham, and Moshe Tennenholtz. Fair imposition. *Journal of Economic Theory*, 118(2):209 – 228, 2004.

[14] Kevin Roberts. The characterization of implementable social choice rules. *In Jean-Jacques Laffont, editor, Aggregation and Revelation of Preferences*, 1979.

[15] Tuomas Sandholm. Automated mechanism design: A new application area for search algorithms. In *In Proceedings of the International Conference on Principles and Practice of Constraint Programming (CP 03), Kinsale, County*. Springer, 2003.

[16] Yoav Shoham and Kevin Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, December 2008.

7 Proof Sketches

7.1 Sketch of Proof of Theorem 2

Recall that a mechanism design problem can be specified by constraints. The objective function is represented as constraints enforcing that the objective value is achieved for each value profile. Consider constraints on a value region $q \in P_V$. As P_V and P_W are linearly consistent, the rebate of agent i is given by the same function $h_p(v_{-i})$ and allocation is constant for all $v \in q$. These together with linearity of h_p for every i guarantee that the constraints are linear on q.

By construction of $h_p(w)$, the constraints hold on the extreme points of q. But if linear constraints hold on the extreme points of a convex region, they hold throughout the region. As q is convex, the constraints are satisfied (including the constraint that checks if the objective value of the restricted problem is achieved) for all $v \in q$. The argument holds for all $q \in P_V$, and the rebate function *for* $w \in p$, $h(w) = h_p(w)$ satisfies all the constraints.

We found the solution $h(w)$ that satisfies all the constraints if the solution $\{\hat{h}(w) \mid w \in \hat{P}_W\}$ satisfies only a subset of constraints. Optimality of the solution $h(w)$ follows immediately from the optimality $\{\hat{h}(w) \mid w \in \hat{P}_W\}$ in the restricted problem.

7.2 Sketch of Proof of Theorem 3

Consider the grid in $\mathbb{R}^d$

$$x_i = c_j \quad \forall\, i \in \{1, \ldots, d\},\ j \in \{1, \ldots, |C|\}$$

The grid partitions the polytope $\{x \in R^d \mid 1 \geq v_1 \geq \ldots \geq v_d \geq 0\}$ into hyperrectangles[10]. We call this partition P_X^g. Each hyperrectangle is further partitioned with $\binom{d}{2}$ "diagonal" hyperplanes: for each pair (i, j) of dimensions chosen from $1, \ldots, d$, draw a diagonal from the lower left to the upper right corner of projection onto the i-j plane (the projection is a rectangle). The equation of the diagonal in i-j (i.e., in $\mathbb{R}^2$) space defines an $(d-1)$-dimensional hyperplane in R^d partitioning the hyperrectangle in halves. These steps are summarized in the *partition*(X) algorithm in Figure 4. We denote partitions of n and $n-1$ dimensional unit hypercubes by $P_n =$ *partition*$([0,1]^n)$ and $P_{n-1} =$ *partition*$([0,1]^{n-1})$. We show that the partitions $P_V =$ *partition*(V) and $P_W =$ *partition*(W) are linearly consistent: i.e., satisfy all properties of Definition 4.

Property (i): all polytopes $q \in P_V$ and $p \in P_W$ are convex Each polytope in P_V and P_W is defined by a finite intersection of halfspaces and is therefore convex.

Property (ii): $\Pi_W(\hat{P}_V) = \hat{P}_W$ We refer to the set of constants extended to include 0 and 1 as $\bar{C} = C \cup 0 \cup 1$. The extreme points of P_n are $\hat{P}_n = \{v \in \mathbb{R}^n \mid v_i \in \bar{C}, \forall i\}$. The extreme points of P_V are $\hat{P}_V = \hat{P}_n \cap V = \{v \in V \mid v_i \in \bar{C}, \forall i\}$. In words, these are all non-increasing vectors of length n with each element taking a value from $\bar{C}$. Applying the $\Pi_W(\hat{P}_V)$ operation

$$\Pi_W(\hat{P}_V) = \bigcup_{v \in \hat{P}_V} \bigcup_{i=1}^{n} v_{-i} = \{w \in W \mid w_i \in \bar{C}, \forall i\}$$

we get the set of all non-increasing vectors of length $n-1$. But this is exactly the set of extreme points of polytopes in the P_W partition: $\hat{P}_W = \hat{P}_{n-1} \cap W$.

Property (iii): P_V refines both the set of polytopes *lift*(P_W) and the allocation partition P_V^a First we focus on grid partitions and show that P_V^g refines *lift*(P_W^g). The grid partition P_W^g partitions W into $(n-1)$-dimensional hyperrectangles. The number of hyperrectangles is $\binom{|\bar{C}|-n-2}{n-1}$, the same as the number of non-increasing vectors[11] t of length $n-1$ where each element can take one of $|\bar{C}|-1$ values. Indeed, for each of the $n-1$ vector coordinates, a region is characterized by the intervals defined by two sequential values in $\bar{C}$, and there are $|\bar{C}|-1$ such intervals. We can enumerate all hyperrectangles using the set of nonincreasing vectors $T = \{t \in \mathbb{Z}^{n-1} \mid t(i) \in \{0, 1, \ldots, |\bar{C}|-1\}\ 1 \leq i \leq n-1\}$. The hyperrectangle corresponding to $t \in T$ is

$$c_{t(i)} \leq w_i \leq c_{t(i)+1} \quad \forall\, i \in \{1, \ldots, n-1\}$$

"Lifting" this hyperrectangle we get n hyperrectangles in V; one for each $1 \leq k \leq n$

$$c_{t(i)} \leq (v_{-k})_i \leq c_{t(i)+1} \quad \forall\, i \in \{1, \ldots, n-1\}$$

These hyperplanes are present in P_V^g: indeed, P_V^g includes all hyperplanes of the form $v_i = c_j\ \forall\ i \in \{1, \ldots, n\},\ j \in \{1, \ldots, |C|\}$. Therefore, P_V^g refines *lift*(P_W^g), and we can proceed to partitions within each hyperrectangle.

A diagonal hyperplane inside a hyperrectangle is given by the equation of the line $w_i = aw_j + b$, connecting the lower left corner of the projection on $i-j$ to the upper right corner. When the *lift* operation is applied, this diagonal hyperplane yields n hyperplanes in V: $(v_{-k})_i = a(v_{-k})_j + b$ each falling within a

[10] On a boundary $x_i = x_j$ only part of the hyperrectangle is in the polytope.

[11] Calculated as the number of ways to distribute k unlabeled balls into x labeled bins: $\binom{x-1+k}{k}$.

corresponding hyperrectangle. By definition of P_V, diagonal hyperplanes of the form $v_i = av_j + b$ for all pairs i and j and all hyperrectangles in P_V^g are included in the partition. The projection $(v_{-k})_i = a(v_{-k})_j + b$ is guaranteed to coincide with one of the diagonal hyperplanes already in P_V^g. Since all lifted hyperplanes coincide with hyperplanes existing in P_V, this partition refines *lift*(P_W). Notice that P_V is the finest partition possible with the given constants C, and therefore refines any constant-dependent allocation partition P_V^a.

Property (iv): each polytope in P_W has n extreme points The diagonal hyperplanes partition each hyperrectangle into *rebate regions*. A rebate region is given by the intersection of $\binom{n-1}{2}$ halfspaces corresponding to either side of the diagonal hyperplanes. In other words, a rebate region is given by consistent (otherwise the region would be empty) pairwise comparison along each pair of coordinates. These comparisons specify a linear order on the coordinates and can be represented by a permutation π, where $\pi(j)$ designates the order of the jth coordinate. Each coordinate has two possible values: the boundaries of the cost region in the coordinate's dimension. Using change of variables, we can assume that these two values are the same for each coordinate, without loss of generality. Now, the linear order restricts possible combinations to nonincreasing vectors of size $n-1$ where each element can take two values. The total number of such vectors is n. These n vectors denoted by $e^1, \ldots, e^n$ are the extreme points of the rebate region $t\pi$. The kth of these extreme points is

$$\begin{aligned} e^k_{\pi(i)} &= c_{t(i)+1} \quad 1 \le i \le k-1 \\ e^k_{\pi(i)} &= c_{t(i)} \qquad k \le i \le n-1 \end{aligned}$$