Understanding Incentives: Mechanism Design becomes Algorithm Design

Yang Cai[1]
EECS, MIT
ycai@csail.mit.edu

Constantinos Daskalakis[2]
EECS, MIT
costis@mit.edu

S. Matthew Weinberg[3]
EECS, MIT
smw79@mit.edu

***Abstract*—We provide a computationally efficient black-box reduction from mechanism design to algorithm design in very general settings. Specifically, we give an approximation-preserving reduction from truthfully maximizing *any* objective under *arbitrary* feasibility constraints with *arbitrary* bidder types to (not necessarily truthfully) maximizing the same objective plus virtual welfare (under the same feasibility constraints). Our reduction is based on a fundamentally new approach: we describe a mechanism's behavior indirectly only in terms of the expected value it awards bidders for certain behavior, and never directly access the allocation rule at all.**

Applying our new approach to revenue, we exhibit settings where our reduction holds *both ways*. That is, we also provide an approximation-sensitive reduction from (non-truthfully) maximizing virtual welfare to (truthfully) maximizing revenue, and therefore the two problems are computationally equivalent. With this equivalence in hand, we show that both problems are NP-hard to approximate within any polynomial factor, even for a single monotone submodular bidder.

We further demonstrate the applicability of our reduction by providing a truthful mechanism maximizing fractional max-min fairness.

I. Introduction

Mechanism design is the problem of optimizing an objective subject to "rational inputs." The difference to *algorithm design* is that the inputs to the objective are not known, but are owned by rational agents who need to be provided incentives in order to share enough information about their inputs such that the desired objective can be optimized. The question that arises is how much this added complexity degrades our ability to optimize objectives, namely

How much more computationally difficult is mechanism design for a certain objective compared to algorithm design for the same objective?

This question has been at the forefront of algorithmic mechanism design, starting already with the seminal work of Nisan and Ronen [27]. In a non-Bayesian setting, i.e. when no prior distributional information is known about the inputs, we now have strong separation results between algorithm and mechanism design. Indeed, a sequence of recent breakthroughs [28], [8], [21], [23] has culminated in combinatorial auction settings where welfare can be optimized computationally efficiently to within a constant factor for "honest inputs," but it cannot be computationally efficiently optimized to within a polynomial factor for "rational inputs," subject to well-believed complexity theoretic assumptions. Besides, the work of Nisan and Ronen studied the problem of minimizing makespan on unrelated machines, which can be well-approximated for honest machines, but whose approximability for rational machines still remains unknown.

In a Bayesian world, where every input is drawn from some known distribution, algorithm and mechanism design appear more tightly connected. Indeed, a sequence of surprising works [26], [25], [4] have established that mechanism design for welfare optimization in an arbitrary environment[1] can be computationally efficiently reduced to algorithm design in the same environment, in an approximation-preserving way. A similar reduction has been recently discovered for the revenue objective [11], [12] in the case of *additive bidders*.[2] Here, mechanism design for revenue optimization in an arbitrary additive environment (computationally efficiently) reduces to algorithm design for *virtual welfare optimization* in the same environment, in an approximation-preserving manner. The natural question is whether such mechanism-to algorithm-design reduction is achievable for general bidder types (i.e. beyond additive) and general objectives (i.e. beyond revenue and welfare). This is what we achieve in this paper.

[1]Supported by NSF Award CCF-0953960 (CAREER) and CCF-1101491.

[2]Supported by a Sloan Foundation Fellowship, a Microsoft Research Faculty Fellowship and NSF Award CCF-0953960 (CAREER) and CCF-1101491.

[3]Supported by a NSF Graduate Research Fellowship and NSF award CCF-1101491.

[1]An *environment* constrains the feasible outcomes of the mechanism as well as the allowable bidder *types*, or *valuations*. The latter map outcomes to value units.

[2]A bidder is *additive* if her value for a bundle of items is just the sum of her values for each item in the bundle.

0272-5428/13 $26.00 © 2013 IEEE
DOI 10.1109/FOCS.2013.72

Informal Theorem 1. *There is a generic, computationally efficient, approximation-preserving reduction from mechanism design for an arbitrary* concave *objective* $\mathcal{O}$, *under arbitrary feasibility constraints and arbitrary allowable bidder types, to algorithm design, under the same feasibility constraints and allowable bidder types, and objective:*

- $\mathcal{O}$ *plus virtual welfare, if* $\mathcal{O}$ *is an allocation-only objective (i.e.* $\mathcal{O}$ *depends only on the allocation chosen and not on payments made).;*
- $\mathcal{O}$ *plus virtual welfare plus virtual revenue, if* $\mathcal{O}$ *is a general objective (i.e.* $\mathcal{O}$ *may depend on the allocation chosen as well as payments made);*
- *virtual welfare, if* $\mathcal{O}$ *is the revenue objective.*

A formal statement of our result is provided as Theorem 4 in Section C.2 of the full version of the paper [13]. Specifically, we provide a Turing reduction from the *Multi-Dimensional Mechanism Design Problem* (MDMDP) to the *Solve Any-Differences Problem* (SADP). MDMDP and SADP are formally defined in Section 5.1 of the full version, and informally below. They are both parameterized by a set $\mathcal{F}$, specifying feasibility constraints on outcomes,[3] a set $\mathcal{V}$ of functions, specifying allowable types of bidders,[4] and an objective function $\mathcal{O}$, mapping a profile $\vec{t} \in \mathcal{V}^m$ of bidder types (m is the number of bidders), a distribution $X \in \Delta(\mathcal{F})$ over feasible outcomes, and a randomized price vector P, to the reals.[5] In terms of these parameters:

- MDMDP is the problem of designing a mechanism M that maximizes $\mathcal{O}$ in expectation over the types $t_1, \ldots, t_m$ of the bidders, given a product distribution over $\mathcal{V}^m$ for $t_1, \ldots, t_m$, and assuming that the bidders play M truthfully. M is restricted to choose outcomes in $\mathcal{F}$ with probability 1, it must be Bayesian Incentive Compatible, and Individually Rational.
- SADP is input a type vector $t_1, \ldots, t_m \in \mathcal{V}$, a list of hyper-types $\mathfrak{t}_1, \ldots, \mathfrak{t}_k \in \mathcal{V}^*$, where $\mathcal{V}^*$ is the closure of $\mathcal{V}$ under addition and positive scalar multiplication, and weights $c_0 \in \mathbb{R}_{\geq 0}$ and $c_1, \ldots, c_m \in \mathbb{R}$. The goal is to choose a distribution X over outcomes and a randomized price vector P so that
$$c_0 \mathcal{O}(\vec{t}, X, P) + \sum_i c_i \mathbb{E}[P_i] + (\mathfrak{t}_j(X) - \mathfrak{t}_{j+1}(X)).$$
is maximized for at least one $j \in \{1, \ldots, k-1\}$. The first term in the above expression is a scaled version of $\mathcal{O}$, the second is a "virtual revenue" term (where c_i is the virtual currency that bidder i uses), and the last term is a "virtual welfare" term (of a pair of adjacent hyper-types the first of which is scaled by 1 and the other by -1). The name "Solve Any-Differences Problem" alludes to the freedom of choosing any value of j and then optimizing.

Notice that MDMDP is a mechanism design problem, where our task is to optimize objective $\mathcal{O}$ given distributional information about the bidder types. On the other hand, SADP is an algorithm design problem, where types and hyper-types are perfectly known and the task is to optimize the sum of the same objective $\mathcal{O}$ plus a virtual revenue and welfare term. In this terminology, Theorem 4 of the full version (stated above as Informal Theorem 1) establishes that there is a computationally efficient reduction from α-approximating MDMDP to α-approximating SADP, for any value α of the approximation, as long as $\mathcal{O}$ is concave in X and P.[6]

It is worth stating a few caveats of our reduction:

1) First, it is known from [18] that it is not possible to have a general reduction from mechanism design to algorithm design with the exact same objective. This motivates the need to include the extra terms of virtual revenue and virtual welfare in the objective of SADP.
2) If $\mathcal{O}$ is allocation-only, i.e. it does not depend on the price vector P, then all coefficients $c_1, \ldots, c_m$ can be taken 0 in the reduction from MDMDP to SADP. Hence, to α-approximate MDMDP it suffices to be able to α-optimize $\mathcal{O}$ plus virtual welfare. In Sections I-B, we discuss *fractional max-min fairness* as an example of such an objective, providing optimal mechanisms for it through our reduction.
3) If $\mathcal{O}$ is price-only, i.e. it does not depend on the outcome X, then the objective in SADP is separable into a price-dependent component ($\mathcal{O}$ plus virtual revenue) and an outcome-dependent component (virtual welfare). Hence, our reduction implies that to α-approximate MDMPD it suffices to be able to α-optimize each of these components separately.

[3]These could encode, e.g., matching constraints of a collection of items to the bidders, or locations to build a public project, etc.

[4]A *type t* of a bidder is a function mapping $\mathcal{F}$ to the reals, specifying how much the bidder values every outcome in $\mathcal{F}$. If a set $\mathcal{V}$ of functions parameterizes one of our problems, then all bidders are restricted to have types in $\mathcal{V}$. E.g., $\mathcal{F}$ may be $\mathbb{R}^\ell$ and $\mathcal{V}$ may contain all additive functions over $\mathcal{F}$.

[5]E.g. $\mathcal{O}$ could be revenue (in this case, $\mathcal{O}(\vec{t}, X, P) = \mathbb{E}[\sum_i P_i]$), or it could be welfare (in this case, $\mathcal{O}(\vec{t}, X, P) = \sum_i t_i(X)$, where $t_i(X) = \mathbb{E}_{x \leftarrow X}[t(x)]$ is the expected value of type t_i for the distribution over outcomes X), or it could be some fairness objective such as $\mathcal{O}(\vec{t}, X, P) = \min_i t_i(X)$.

[6]Formally $\mathcal{O}$ is concave in X and P if for any (X_1, P_1) and (X_2, P_2) and any $c \in [0,1]$ and $\vec{t}$, $\mathcal{O}(\vec{t}, cX_1 + (1-c)X_2, cP_1 + (1-c)P_2) \geq c\mathcal{O}(\vec{t}, X_1, P_1) + (1-c)\mathcal{O}(\vec{t}, X_2, P_2)$, where $cX_1 + (1-c)X_2$ is the mixture of distributions X_1 and X_2 over outcomes, with mixing weights c and $1-c$ respectively. All objectives mentioned or studied in this paper are concave.

4) If $\mathcal{O}$ is the revenue objective (this is a special case of 3), the price-dependent component in SADP is trivial to optimize. In this case, to α-approximate MDMPD it suffices to be able to α-optimize virtual welfare (i.e. we can take $c_0 = c_1 = \cdots = c_m = 0$ in the SADP instance output by the reduction). See Theorem 3. Additionally we note the following.
   a) This special case of our reduction already generalizes the results of [11] to arbitrary types. Recall that the reduction of [11] from MDMDP to virtual-welfare optimization could only accommodate additive types.
   b) For a special family $\mathcal{V}$ of functions, we provide a reduction in the other direction, i.e. from SADP to MDMDP. As a corollary of this reduction we obtain strong inapproximability results for optimal multi-dimensional mechanism design with submodular bidders. We discuss this in more detail in Section I-A.
5) Our generic reduction from MDMDP to SADP can take the number k of hyper-types input to SADP to be 2. We define SADP for general k for flexibility. In particular, general k enables our inapproximability result for optimal mechanism design via a reduction from SADP (general k) to MDMDP.
6) Finally, it is worth noting that our reduction follows a similar outline to that of previous work on revenue maximization [11]: we first come up with a way to describe a mechanism's behavior succinctly, via a description that we call the *implicit form*. Next, we write a LP to find the implicit form of the $\mathcal{O}$-optimal truthful mechanism. Solving this LP will require a separation oracle for the space of implicit forms that correspond to feasible (not necessarily truthful) mechanisms. Such a separation oracle can be obtained with black-box access to an (approximate) algorithm for SADP with $k = 2$.

A. Revenue

Our framework described above provides reductions from mechanism design for some arbitrary objective $\mathcal{O}$ to algorithm design for the same objective $\mathcal{O}$ plus a virtual revenue and a virtual welfare term. As pointed out earlier in this section, we can't avoid some modification of $\mathcal{O}$ in the algorithm design problem sitting at the output of a general reduction such as ours, due to the impossibility result of [18]. Nevertheless, there could very well be other modified objectives that a general reduction could be reducing to, with better or worse algorithmic properties. The question that arises is this: *Could we be hurting ourselves focusing on SADP as an algorithmic vehicle to solve MDMDP?* Our previous work on revenue maximization for additive bidders [11] exhibits very general $\mathcal{F}$'s where the answer is "no," motivating our generalization here to non-additive bidders and general objectives. Indeed, we illustrate the reach of our new framework in Section I-B by providing optimal mechanisms for non-linear objectives, an admittedly difficult and under-developed topic in Bayesian mechanism design [18], [15].

Here we provide a different type of evidence for the tightness of our approach via reductions going the other way, i.e. from SADP to MDMDP. Recall that MDMDP($\mathcal{F}, \mathcal{V}, \mathrm{Revenue}$) reduces to solving SADP instances, which satisfy $c_0 = c_1 = \cdots = c_m = 0$ and therefore only have a virtual welfare component depending on some $\mathfrak{t}_1, \ldots, \mathfrak{t}_k \in \mathcal{V}^*$. In Section IV, we identify conditions for a collection of functions $\mathfrak{t}_1, \ldots, \mathfrak{t}_k \in \mathcal{V}^*$ under which SADP reduces to MDMDP, showing that for such instances solving SADP is unavoidable for solving MDMDP. Indeed, our reduction is strong enough that we obtain very strong inapproximability results for revenue optimization, even when there is a single monotone submodular bidder. To the best of our knowledge our result is the first inapproximability result for optimal mechanism design.

Informal Theorem 2. MDMDP($2^{[n]}$, monotone submodular functions, revenue) *cannot be approximated to within any polynomial factor in polynomial time, if we are given value or demand oracle access to the submodular functions in the support of the bidders distributions,*[7] *even if there is only one bidder. The same is true if we are given explicit access to these functions (as Turing machines) unless $NP \subseteq RP$.*

B. Fractional Max-Min Fairness

Certainly revenue and welfare are the most widely studied objectives in mechanism design. Nevertheless, resource allocation often requires optimizing non-linear objectives such as the fairness of an allocation, or the makespan of some scheduling of jobs to machines. Already the seminal paper of Nisan and Ronen studies minimizing makespan when scheduling jobs to selfish machines, in a non-Bayesian setting. Following this work, a lot of algorithmic mechanism design research has focused on non-linear objectives in non-Bayesian settings (see, e.g., [19], [3] and their references), but positive results have been scarce. More recently, research has studied non-linear objectives in Bayesian settings [18], [15]. While [18] provide impossibility results, the results of [15] give hope that non-linear objectives might be better behaved in Bayesian settings. In part, this is our motivation for providing an algorithmic framework for general objectives in this work.

[7]We explain the difference between value and demand oracle access in Section IV.

As a concrete example of the reach of our techniques, we provide optimal mechanisms for a (non-linear) max-min fairness objective in Section 6 of the full version. The setting we solve is this: There are n items that can be allocated to m additive bidders, subject to some constraints $\mathcal{F}$. $\mathcal{F}$ could be matching constraints, matroid constraints, downwards-closed constraints, or more general constraints. Now, given a distribution X over allocations in $\mathcal{F}$, how fair is it? E.g., if there is one item and two bidders with value 1 for the item, what is the fairness of a randomized allocation that gives the item to each bidder with probability $\frac{1}{2}$? Should it be 0, because with probability 1, exactly one bidder gets value 0 from the allocation? Or, should it be $1/2$ because each bidder gets an expected value of $1/2$? Clearly, both are reasonable objectives, but we study the latter. Namely, we define the *fractional max-min fairness* objective as:

$$\mathcal{O}(\vec{t}, X) = \min_i t_i(X).$$

We obtain the following result, which is stated formally as Corollary 6 in Section 6 of the full version.

Informal Theorem 3. *Let G be a polynomial-time α-approximation algorithm for*

Max-Weight($\mathcal{F}$): *Given weights $(w_{ij})_{ij}$, find $S \in \mathcal{F}$ maximizing $\sum_{(i,j)\in S} w_{ij}$.*

With black-box access to G, we can α-approximate MDMDP($\mathcal{F}$, additive functions, $\mathcal{O}$) in polynomial time. For instance, if $\mathcal{F}$ are matching constraints, matroid constraints, or the intersection of two matroids, we can optimally solve MDMDP($\mathcal{F}$, additive functions, $\mathcal{O}$) in polynomial-time.

C. Related Work

a) Revenue Maximization.: There has been much work in recent years on revenue maximization in multi-dimensional settings [1], [2], [6], [9], [10], [14], [16], [17], [22], [24]. Our approach is most similar to that of [11], [12], which was recently extended in [5]. These works solved the revenue maximization problem for *additive* bidders via a black-box reduction to welfare maximization. In [5], numerous extensions are shown that accommodate risk-averse buyers, ex-post budget constraints, and more. But both approaches are inherently limited to revenue maximization and additive bidders. Even just within the framework of revenue maximization, our work breaks through a major barrier, as every single previous result studies only additive bidders.

b) Hardness of Revenue Maximization.: Three different types of results regarding the computational hardness of revenue maximization are known. It is shown in [7] that (under standard complexity theoretic assumptions) no efficient algorithm can find a deterministic mechanism whose revenue is within any polynomial factor of the optimal (for deterministic mechanisms), even for very simple single bidder settings. However, the optimal randomized mechanism in those same settings can be found in polynomial time [6]. Hardness results for randomized mechanisms are comparatively scarce.

Very recently, a new type of hardness was shown in [20]. There, they show that it is #P-hard to find (any description of) an optimal randomized mechanism even in very simple single bidder settings. Specifically, the problem they study is of a single additive bidder whose value for each of n items is drawn independently from a distribution of support 2. The natural description complexity of this problem is $O(n)$ (just list the values for each item and their probabilities), but they show that the optimal randomized mechanism cannot be found or even executed in time poly(n) (unless ZPP = #P). This is a completely different type of hardness than what is shown in this paper. Specifically, we show that certain instances are hard to solve even when the support of the input distribution is small (whereas it is 2^n in the hard examples of [20]), but the instances are necessarily more involved (we use submodular bidders), as the optimal randomized mechanism can be found in time polynomial in the support of the input distribution for additive bidders [6].

The existing result that is most similar to ours appears in [22]. There, they show that it is NP-hard to maximize revenue exactly when there is a single bidder whose value for subsets of n items is an OXS function.[8] Our approaches are even somewhat similar: we both aim to understand the necessary structure on a type space in order for the optimal revenue to satisfy a simple formula. The big difference between their result and ours is that their results are inherently limited to settings with a single bidder who has *two* possible types. While this suffices to show hardness of exact maximization, there is no hope of extending this to get hardness of approximation.[9] Our stronger results are enabled by a deeper understanding of the optimal revenue for single bidder settings with many possible types, which is significantly more involved than the special case of two types.

c) General Objectives.: Following the seminal paper of Nisan and Ronen, much work in algorithmic mechanism design has been devoted to maximizing non-linear objectives in a truthful manner. Recently, more attention has been given to Bayesian settings, as there are numerous strong hardness results in non-Bayesian

[8] OXS functions are a subclass of submodular functions

[9] The seller always has the option of completely ignoring one type and charging the other their maximum value for their favorite set. This mechanism achieves a $\frac{1}{2}$-approximation in every setting.

settings. Still, it is shown in [18] that no polynomial-time black-box reduction from truthfully maximizing a non-linear objective to non-truthfully maximizing the same non-linear objective exists without losing a polynomial factor in the approximation ratio, even in Bayesian settings. Even more recently, a non-black box approach was developed in [15] to minimize makespan in certain Bayesian settings. Our black-box approach sidesteps the hardness result of [18] by reducing the problem of truthfully maximizing an objective to non-truthfully maximizing a modified objective.

D. *Paper Structure*

To make our framework easier to understand, we separate the paper as follows. In Sections II through IV, we provide the necessary details of our framework to show how it applies to revenue maximization. We defer a formal discussion of the full generality of our approach to Section 5 of the full version, and show how it applies to the fractional max-min fairness objective in Section 6 of the full version. To ease notation we initially define restricted versions of the MDMDP and SADP problems in Section II as they apply to revenue, using these restricted definitions through Section IV. These definitions are expanded in Section 5 of the full version to accommodate general objectives.

II. PRELIMINARIES

Mechanism Design Setting. The mechanism designer has a set of feasible outcomes $\mathcal{F}$ to choose from, which depending on the application could be feasible allocations of items to bidders, locations to build a public project, etc. Each bidder participating in the mechanism may have several possible *types*. A bidder's type consists of a value for each possible outcome in $\mathcal{F}$. Specifically, a bidder's type is a function t mapping $\mathcal{F}$ to $\mathbb{R}_+$. T_i denotes the set of all possible types of bidder i, which we assume to be finite. The designer has a prior distribution $\mathcal{D}_i$ over T_i for bidder i's type. Bidders are *quasi-linear* and *risk-neutral*. That is, the utility of a bidder of type t for a randomized outcome (distribution over outcomes) $X \in \Delta(\mathcal{F})$, when he is charged (a possibly random price with expectation) p, is $\mathbb{E}_{x\leftarrow X}[t(x)]-p$. Therefore, we may extend t to take as input distributions over outcomes as well, with $t(X) = \mathbb{E}_{x\leftarrow X}[t(x)]$. A *type profile* $\vec{t} = (t_1, \ldots, t_m)$ is a collection of types for each bidder. We assume that the types of the bidders are independent so that $\mathcal{D} = \times_i \mathcal{D}_i$ is the designer's prior distribution over the complete type profile.

Mechanisms. A (direct) mechanism consists of two functions, a (possibly randomized) allocation rule and a (possibly randomized) price rule, and we allow these rules to be correlated. The allocation rule takes as input a type profile $\vec{t}$ and (possibly randomly) outputs an allocation $A(\vec{t}) \in \mathcal{F}$. The price rule takes as input a profile $\vec{t}$ and (possibly randomly) outputs a price vector $P(\vec{t})$. When the bid profile $\vec{t}$ is reported to the mechanism $M = (A, P)$, the (possibly random) allocation $A(\vec{t})$ is selected and bidder i is charged the (possibly random) price $P_i(\vec{t})$. We will sometimes discuss the *interim allocation rule* of a mechanism, which is a function that takes as input a bidder i and a type $t_i \in T_i$ and outputs the distribution of allocations that bidder i sees when reporting type t_i over the randomness of the mechanism and the other bidders' types. Specifically, if the interim allocation rule of $M = (A, P)$ is X, then $X_i(t_i)$ is a distribution satisfying

$$\Pr[x \leftarrow X_i(t_i)] = \mathbb{E}_{\vec{t}_{-i}\leftarrow\mathcal{D}_{-i}}\left[\Pr[A(t_i;\vec{t}_{-i}) = x \mid \vec{t}_{-i}]\right],$$

where t_{-i} is the vector of types of all bidders but bidder i in $\vec{t}$, and $\mathcal{D}_{-i}$ is the distribution of t_{-i}. Sometimes we write $\vec{t}_{-i}$ instead of t_{-i} to emphasize that it's a vector of types.

A mechanism is said to be *Bayesian Incentive Compatible (BIC)* if it is in every bidder's best interest to report truthfully their type, conditioned on the fact that the other bidders report truthfully their type. A mechanism is said to be *Individually Rational (IR)* if it is in every bidder's best interest to participate in the mechanism, no matter their type. Formal definitions can be found in Appendix A of the full version.

Goal of the designer. In Section III we present our mechanism- to algorithm-design reduction for the revenue objective. The problem we reduce from is designing a BIC, IR mechanism that maximizes expected revenue, when encountering a bidder profile sampled from some given distribution $\mathcal{D}$. Our reduction is described in terms of the problems MDMDP and SADP defined next. In Section 5 of the full version we generalize our reduction to general objectives and accordingly generalize both problems to accommodate general objectives. But our approach is easier to understand for the revenue objective, so we give that here separately.

Formal Problem Statements. We present black-box reductions between two problems: the Multi-Dimensional Mechanism Design Problem (MDMDP) and the Solve-Any Differences Problem (SADP). MDMDP is a well-studied mechanism design problem [10], [11], [12]. SADP is a new algorithmic problem that we show has strong connections to MDMDP. In order to discuss our reductions appropriately, we will parameterize the problems by two parameters $\mathcal{F}$ and $\mathcal{V}$. Parameter $\mathcal{F}$ denotes the feasibility constraints of the setting; e.g., $\mathcal{F}$ might be "each item is awarded to at most one bidder" or "a bridge may be built in location A or B", etc. Parameter $\mathcal{V}$ denotes the allowable valuation functions, mapping $\mathcal{F}$ to the reals; e.g., if $\mathcal{F} = \mathbb{R}^\ell$, then $\mathcal{V}$ may be "all additive functions over $\mathcal{F}$" or "all submodular functions", etc. Informally, MDMDP asks for a BIC, IR mechanism that maximizes expected revenue for certain feasibility constraints $\mathcal{F}$ and a restricted class of valuation functions $\mathcal{V}$. SADP asks for an element in $\mathcal{F}$ maximizing the difference of two functions in $\mathcal{V}$, but the algorithm is allowed to choose any two adjacent functions in an ordered list of size k. Throughout the paper will use $\mathcal{V}^*$ to denote the closure of $\mathcal{V}$ under addition and positive scalar multiplication, and the term "α-approximation" ($\alpha \leq 1$) to denote a (possibly randomized) algorithm whose expected value for the desired objective is an α-fraction of the optimal.

MDMDP($\mathcal{F}$, $\mathcal{V}$): INPUT: For each bidder $i \in [m]$, a finite set of types $T_i \subseteq \mathcal{V}$ and a distribution $\mathcal{D}_i$ over T_i. GOAL: Find a feasible (outputs an outcome in $\mathcal{F}$ with probability 1) BIC, IR mechanism M, that maximizes expected revenue, when n bidders with types sampled from $\mathcal{D} = \times_i \mathcal{D}_i$ play M truthfully (with respect to all feasible, BIC, IR mechanisms). M is said to be an α-approximation to MDMDP if its expected revenue is at least a α-fraction of the optimal obtainable expected revenue.

SADP($\mathcal{F}$, $\mathcal{V}$): Given as input functions $f_j \in \mathcal{V}^*$ ($1 \leq j \leq k$), find a feasible outcome $X \in \mathcal{F}$ such that there exists an index $j^* \in [k-1]$ such that:

$$f_{j^*}(X) - f_{j^*+1}(X) = \max_{X' \in \mathcal{F}} \{f_{j^*}(X') - f_{j^*+1}(X')\}.$$

X is said to be an α-approximation to SADP if there exists an index $j^* \in [k-1]$ such that:

$$f_{j^*}(X) - f_{j^*+1}(X) \geq \alpha \max_{X' \in \mathcal{F}} \{f_{j^*}(X') - f_{j^*+1}(X')\}.$$

Representation Questions. Notice that both MDMDP and SADP are parameterized by $\mathcal{F}$ and $\mathcal{V}$. As we aim to leave these sets unrestricted, we assume that their elements are represented in a computationally meaningful way. That is, we assume that elements of $\mathcal{F}$ can be indexed using $O(\log |\mathcal{F}|)$ bits and are input to functions that evaluate them via this representation. We assume elements $f \in \mathcal{V}$ are input either via a turing machine that evaluates f (and the size of this turing machine counts towards the size of the input), or as a black box. Moreover, all of our reductions apply whether or not the input functions are given explicitly or as a black box [10]. Finally, whenever we evaluate the running time of an algorithm for either MDMDP or SADP, or of a reduction from one problem to the other, we count the time spent in an oracle call to functions input to these problems as one. Similarly, whenever we show a computational hardness result for either MDMDP or SADP, the time spent in one oracle call is considered as one.

Linear Programming. Our results require the ability to solve linear programs with separation oracles as well as "weird" separation oracles, a concept recently introduced in [12]. Throughout the paper we will use the notation αP to denote the polytope P shrunk by a factor of $\alpha \leq 1$. That is, $\alpha P = \{\alpha \vec{x} | \vec{x} \in P\}$. We make use of the following Theorem from [12], as well as other theorems regarding solving linear programs which are all stated below. Theorem 1 comes from recent work [12]. Theorem 2 states well-known properties of the ellipsoid algorithm. Corollary 1 is an obvious corollary of part 1 of Theorem 1. In addition, a complete discussion of this can be found in [12].

Theorem 1. *([12]) Let P be a d-dimensional bounded convex polytope containing the origin, and let $\mathcal{A}$ be an algorithm that takes any direction $\vec{w} \in [-1,1]^d$ as input and outputs a point $\mathcal{A}(\vec{w}) \in P$ such that $\mathcal{A}(\vec{w}) \cdot \vec{w} \geq \alpha \cdot \max_{\vec{x} \in P}\{\vec{x} \cdot \vec{w}\}$ for some absolute constant $\alpha \leq 1$. Then there is a weird separation oracle WSO for αP such that,*

1) *Every halfspace output by the WSO will contain αP.*
2) *Whenever $WSO(\vec{x})$ = "yes," the execution of WSO explicitly finds directions $\vec{w}_1, \ldots, \vec{w}_l$ such that $\vec{x} \in Conv\{\mathcal{A}(\vec{w}_1), \ldots, \mathcal{A}(\vec{w}_l)\}$.*
3) *Let b be the bit complexity of the input vector $\vec{x}$, and ℓ be an upper bound of the bit complexity of $\mathcal{A}(\vec{w})$ for all $\vec{w} \in [-1,1]^d$, $rt_{\mathcal{A}}(y)$ be the running time of algorithm $\mathcal{A}$ on some input with bit complexity y. Then on input $\vec{x}$, WSO terminates in time $poly(d, b, \ell, rt_{\mathcal{A}}(poly(d, b, \ell)))$ and makes at most $poly(d, b, \ell)$ many queries to $\mathcal{A}$.*

Corollary 1. *([12]) Let Q be an arbitrary intersection of halfspaces. Let SO be a separation oracle for αP, where P is a bounded convex polytope containing the origin and $\alpha \leq 1$ some constant. Let c_1 be the solution output by the Ellipsoid algorithm that maximizes some linear objective $\vec{c} \cdot \vec{x}$ subject to $\vec{x} \in Q$ and $SO(\vec{x})$ = "yes". Let also c_2 be the solution output by the exact same algorithm, but replacing SO with WSO, a "weird" separation oracle for αP as in Theorem 1—i.e. run the Ellipsoid algorithm with the exact same parameters as if WSO was a valid separation oracle for αP. Then $c_2 \geq c_1$.*

Theorem 2. *[Ellipsoid Algorithm for Linear Programming] Let P be a bounded convex polytope in $\mathbb{R}^d$ specified via a separation oracle SO, and let $\vec{c} \cdot \vec{x}$ be a linear function. Suppose that ℓ is an upper bound on the bit complexity of the coordinates of $\vec{c}$ as well*

[10] When we claim that we can solve problem $\mathcal{P}_1$ given black-box access to a solution to problem $\mathcal{P}_2$, we mean that the functions input to problem $\mathcal{P}_1$ may be given either explicitly or as a black box, and that they are input in the same form to $\mathcal{P}_2$.

as the extreme points of P,[11] *and also that we are given a ball $B(x_0, R)$ containing P such that x_0 and R have bit complexity poly(d, ℓ). Then we can run the ellipsoid algorithm to optimize $\vec{c}\cdot\vec{x}$ over P, maintaining the following properties:*

1) *The algorithm will only query SO on rational points with bit complexity poly(d, ℓ).*
2) *The ellipsoid algorithm will solve the Linear Program in time polynomial in d, ℓ and the runtime of SO when the input query is a rational point of bit complexity poly(d, ℓ).*
3) *The output optimal solution is a corner of P.*

A. Implicit Forms

Here, we give the necessary preliminaries to understand a mechanism's *implicit form*. The implicit form is oblivious to what allocation rule the mechanism actually uses; it just stores directly the necessary information to decide if a mechanism is BIC and IR. For a mechanism $M = (A, P)$ and bidder distribution $\mathcal{D}$, the implicit form of M with respect to $\mathcal{D}$ consists of two parts. The first is a function that takes as input a bidder i and a pair of types t_i, t_i', and outputs the expected value of a bidder with type t_i for reporting t_i' instead. Formally, we may store this function as an mk^2-dimensional vector $\vec{\pi}(M)$ with:

$$\pi_i(t_i, t_i') = \mathbb{E}_{\vec{t}_{-i}\leftarrow\mathcal{D}_{-i}}[t_i(A(t_i'; \vec{t}_{-i}))].$$

The second is just a function that takes as input a bidder i and a type t_i and outputs the expected price paid by bidder i when reporting type t_i. Formally, we may store this function as a mk-dimensional vector $\vec{P}(M)$ with:

$$P_i(t_i) = \mathbb{E}_{\vec{t}_{-i}\leftarrow\mathcal{D}_{-i}}[P_i(t_i; \vec{t}_{-i})].$$

We will denote the implicit form of M as $\vec{\pi}^I(M) = (\vec{\pi}(M), \vec{P}(M))$, and may drop the parameter M where appropriate. We call $\vec{\pi}$ the allocation component of the implicit form and $\vec{P}$ the price component. Sometimes, we will just refer to $\vec{\pi}$ as the implicit form if the context is appropriate.

We say that (the allocation component of) an implicit form, $\vec{\pi}$, is *feasible* with respect to $\mathcal{F}, \mathcal{D}$ if there exists a (possibly randomized) mechanism M that chooses an allocation in $\mathcal{F}$ with probability 1 such that $\vec{\pi}(M) = \vec{\pi}$. We denote by $F(\mathcal{F}, \mathcal{D})$ the set of all feasible (allocation components of) implicit forms. We say that an implicit form $\vec{\pi}^I$ is feasible if its allocation component $\vec{\pi}$ is feasible. We say that $\vec{\pi}^I$ is BIC if every mechanism with implicit form $\vec{\pi}^I$ is BIC. It is easy to see that $\vec{\pi}^I$ is BIC if and only if for all i, and $t_i, t_i' \in T_i$, we have:

$$\pi_i(t_i, t_i) - P_i(t_i) \geq \pi_i(t_i, t_i') - P_i(t_i').$$

Similarly, we say that $\vec{\pi}^I$ is IR if every mechanism with implicit form $\vec{\pi}^I$ is IR. It is also easy to see that $\vec{\pi}^I$ is IR if and only if for all i and $t_i \in T_i$ we have:

$$\pi_i(t_i, t_i) - P_i(t_i) \geq 0.$$

[11] If a d-dimensional convex region with extreme points of bit complexity ℓ is non-empty, then it certainly has volume at least $2^{-\text{poly}(d,\ell)}$.

III. Revenue Maximization

In this section, we describe and prove correctness of our reduction when the objective is revenue. Every result in this section is a special case of our general reduction (that applies to any concave objective) from Section 5 of the full version, and could be obtained as an immediate corollary. We present revenue separately as a special case with the hope that this will help the reader understand the general reduction. Here is an outline of our approach: In Section III-A, we show that $F(\mathcal{F}, \mathcal{D})$ is a convex polytope and write a poly-size linear program that finds the revenue-optimal implicit form provided that we have a separation oracle for $F(\mathcal{F}, \mathcal{D})$. In Section III-B we show that any poly-time α-approximation algorithm for SADP($\mathcal{F}$,$\mathcal{V}$) implies a poly-time weird separation oracle for $\alpha F(\mathcal{F}, \mathcal{D})$, and therefore a poly-time α-approximation algorithm for MDMDP($\mathcal{F}$, $\mathcal{V}$).

A. Linear Programming Formulation

We now show how to write a poly-size linear program to find the implicit form of a mechanism that solves the MDMDP. The idea is that we will search over all feasible, BIC, IR implicit forms for the one that maximizes expected revenue. We first show that $F(\mathcal{F}, \mathcal{D})$ is always a convex polytope, then state the linear program and prove that it solves MDMDP. For ease of exposition, most proofs can be found in Appendix D of the full version.

Lemma 1. *$F(\mathcal{F}, \mathcal{D})$ is a convex polytope.*

Observation 1. *Any α-approximate solution to the linear program of Figure 1 corresponds to a feasible, BIC, IR implicit form whose revenue is at least a α-fraction of the optimal obtainable expected revenue by a feasible, BIC, IR mechanism.*

Corollary 2. *The program in Figure 1 is a linear program with $\sum_{i\in[m]}(|T_i|^2 + |T_i|)$ variables. If b is an upper bound on the bit complexity of $Pr[t_i]$ and $t_i(X)$ for all i, t_i and $X \in \mathcal{F}$, then with black-box access to a weird separation oracle, WSO, for $\alpha F(\mathcal{F}, \mathcal{D})$, the implicit form of an α-approximate solution to MDMDP*

Variables:

- $\pi_i(t_i, t_i')$, for all bidders i and types $t_i, t_i' \in T_i$, denoting the expected value obtained by bidder i when their true type is t_i but they report t_i' instead.
- $P_i(t_i)$, for all bidders i and types $t_i \in T_i$, denoting the expected price paid by bidder i when they report type t_i.

Constraints:

- $\pi_i(t_i, t_i) - P_i(t_i) \geq \pi_i(t_i, t_i') - P_i(t_i')$, for all bidders i, and types $t_i, t_i' \in T_i$, guaranteeing that the implicit form $(\vec{\pi}, \vec{P})$ is BIC.
- $\pi_i(t_i, t_i) - P_i(t_i) \geq 0$, for all bidders i, and types $t_i \in T_i$, guaranteeing that the implicit form $(\vec{\pi}, \vec{P})$ is individually rational.
- $\vec{\pi} \in F(\mathcal{F}, \mathcal{D})$, guaranteeing that the implicit form $(\vec{\pi}, \vec{P})$ is feasible.

Maximizing:

- $\sum_i \sum_{t_i} \Pr[t_i \leftarrow \mathcal{D}_i] \cdot P_i(t_i)$, the expected revenue when played truthfully by bidders sampled from $\mathcal{D}$.

Fig. 1. A linear programming formulation for MDMDP.

can be found in time polynomial in $\sum_{i\in[m]} |T_i|$, b, and the runtime of WSO on inputs with bit complexity polynomial in $\sum_{i\in[m]} |T_i|, b$.

B. A Reduction from MDMDP to SADP

Based on Corollary 2, the only obstacle to solving the MDMDP is obtaining a separation oracle for $F(\mathcal{F}, \mathcal{D})$ (or "weird" separation oracle for $\alpha F(\mathcal{F}, \mathcal{D})$). In this section, we use Theorem 1 to obtain a weird separation oracle for $\alpha F(\mathcal{F}, \mathcal{D})$ using only black box access to an α-approximation algorithm for SADP. For ease of exposition, most proofs can be found in Appendix E of the full version.

In order to apply Theorem 1, we must first understand what it means to compute $\vec{x} \cdot \vec{w}$ in our setting. Proposition 1 below accomplishes this. In reading the proposition, recall that $\vec{x}$ is some implicit form $\vec{\pi}$, so the direction $\vec{w}$ has components $w_i(t_i, t_i')$ for all i, t_i, t_i'. Also note that a type t_i is a function that maps allocations to values. So $\sum_{t_i \in T_i} C_i t_i$ is also a function that maps allocations to values (and therefore could be interpreted as a type or virtual type). Namely, it maps X to $\sum_{t_i \in T_i} C_i t_i(X)$

Proposition 1. *Let $\vec{\pi} \in F(\mathcal{F}, \mathcal{D})$ and let $\vec{w}$ be a direction in $[-1,1]^{\sum_i |T_i|^2}$. Then $\vec{\pi} \cdot \vec{w}$ is exactly the expected virtual welfare of a mechanism with implicit form $\vec{\pi}$ when the virtual type of bidder i with real type t_i' is $\sum_{t_i \in T_i} \frac{w_i(t_i, t_i')}{\Pr[t_i']} \cdot t_i$.*

Now that we know how to interpret $\vec{w} \cdot \vec{\pi}$, recall that Theorem 1 requires an algorithm $\mathcal{A}$ that takes as input a direction $\vec{w}$ and outputs a $\vec{\pi}$ with $\vec{w} \cdot \vec{\pi} \geq \alpha \cdot \max_{\vec{x} \in F(\mathcal{F}, \mathcal{D})}\{\vec{w} \cdot \vec{x}\}$. With Proposition 1, we know that this is exactly asking for a feasible implicit form whose virtual welfare (computed with respect to $\vec{w}$) is at least an α-fraction of the virtual welfare obtained by the optimal feasible implicit form. The optimal feasible implicit form corresponds to a mechanism that, on every profile, chooses the allocation in $\mathcal{F}$ that maximizes virtual welfare. One way to obtain an α-approximate implicit form is to use a mechanism that, on every profile, chooses an α-approximate outcome in $\mathcal{F}$. Corollary 3 below states this formally.

Corollary 3. *Let M be a mechanism that on profile $(t_1', \ldots, t_m')$ chooses a (possibly randomized) allocation $X \in \mathcal{F}$ such that*

$$\sum_{i\in[m]} \sum_{t_i \in T_i} \frac{w_i(t_i, t_i')}{\Pr[t_i']} \cdot t_i(X) \geq \alpha \cdot \max_{X' \in \mathcal{F}} \left\{ \sum_{i\in[m]} \sum_{t_i \in T_i} \frac{w_i(t_i, t_i')}{\Pr[t_i']} \cdot t_i(X') \right\}.$$

Then the implicit form, $\vec{\pi}(M)$ satisfies:

$$\vec{\pi}(M) \cdot \vec{w} \geq \alpha \cdot \max_{\vec{x} \in F(\mathcal{F}, \mathcal{D})} \{\vec{x} \cdot \vec{w}\}.$$

With Corollary 3, we now want to study the problem of maximizing virtual welfare on a given profile. This turns out to be exactly an instance of SADP.

Proposition 2. *Let $t_i \in \mathcal{V}$ for all i, t_i. Let also $C_i(t_i)$ be any real numbers, and $\sum_{t_i \in T_i} C_i(t_i) t_i(\cdot)$ be the virtual type of bidder i. Then any $X \in \mathcal{F}$ that is an α-approximation to SADP($\mathcal{F}$,$\mathcal{V}$) on input $(f_1 = \sum_i \sum_{t_i | C_i(t_i) > 0} C_i(t_i) t_i(\cdot), f_2 = \sum_i \sum_{t_i | C_i(t_i) < 0} -C_i(t_i) t_i(\cdot))$ is also an α-approximation for maximizing virtual welfare. That is:*

$$\sum_i \sum_{t_i} C_i(t_i) t_i(X) \geq \alpha \cdot \max_{X' \in \mathcal{F}} \left\{ \sum_i \sum_{t_i} C_i(t_i) t_i(X') \right\}$$

Combining Corollary 3 and Proposition 2 yields Corollary 4 below.

Corollary 4. *Let G be any α-approximation algorithm for SADP($\mathcal{F}$,$\mathcal{V}$). Let also M be the mechanism that, on* *profile* $(t'_1, \ldots, t'_m)$ *chooses the allocation*

$$G(\{\sum_i \sum_{t_i | w_i(t_i, t'_i) > 0} (w_i(t_i, t'_i)/\Pr[t'_i])t_i(\cdot) \ ,$$
$$\sum_i \sum_{t_i | w_i(t_i, t'_i) < 0} -(w_i(t_i, t'_i)/\Pr[t'_i])t_i(\cdot)\}).$$

Then the interim form $\vec{\pi}(M)$ *satisfies:*

$$\vec{\pi}(M) \cdot \vec{w} \geq \alpha \cdot \max_{\vec{x} \in F(\mathcal{F}, \mathcal{D})} \{\vec{x} \cdot \vec{w}\}.$$

At this point, we would like to just let $\mathcal{A}$ be the algorithm that takes as input a direction $\vec{w}$ and computes the implicit form prescribed by Corollary 4. Corollary 4 shows that this algorithm satisfies the hypotheses of Theorem 1, so we would get a weird separation oracle for $\alpha F(\mathcal{F}, \mathcal{D})$. Unfortunately, this requires some care, as computing the implicit form of a mechanism exactly would require enumerating every profile in the support of $\mathcal{D}$, and also enumerating the randomness used on each profile. Luckily, however, both of these issues arose in previous work and were solved [11], [12]. We overview the necessary approach in Appendix E of the full version, and refer the reader to [11], [12] for complete details.

After these modifications, the only remaining step is to turn the implicit form output by the LP of Figure 1 into an actual mechanism. This process is simple and made possible by guarantee 2) of Theorem 1. We overview the process in Section E of the full version, as well as give a formal description of our algorithm to solve MDMDP as Algorithm 3 in the full version. We conclude this section with a theorem describing the performance of this algorithm. In the following theorem, G denotes a (possibly randomized) α-approximation algorithm for SADP($\mathcal{F}$,$\mathcal{V}$).

Theorem 3. *Let b be an upper bound on the bit complexity of $t_i(X)$ and $\Pr[t_i]$ for any $i \in [m]$, $t_i \in T_i$, and $X \in \mathcal{F}$. Then our algorithm (formally defined as Algorithm 3 in the full version) makes poly$(\sum_i |T_i|, 1/\epsilon, b)$ calls to G, and terminates in time poly$(\sum_i |T_i|, 1/\epsilon, b, rt_G($poly$(\sum_i |T_i|, 1/\epsilon, b)))$, where $rt_G(x)$ is the running time of G on input with bit complexity x. If the types are normalized so that $t_i(X) \in [0, 1]$ for all i, $t_i \in T_i$, and $X \in \mathcal{F}$, and OPT is the optimal obtainable expected revenue for the given MDMDP instance, then the mechanism output by our algorithm obtains expected revenue at least $\alpha OPT - \epsilon$, and is ϵ-BIC with probability at least $1 - \exp($poly$(\sum_i |T_i|, 1/\epsilon, b))$.*

IV. Reduction from SADP to MDMDP

In this section we overview our reduction from SADP to MDMDP that holds for a certain subclass of SADP instances (a much longer exposition of the complete approach can be found in Appendix B of the full version). The subclass is general enough for us to conclude that revenue maximization, even for a single submodular bidder, is impossible to approximate within any polynomial factor unless $NP = RP$. For this section, we will restrict ourselves to single-bidder settings, as our reduction will always output a single-bidder instance of MDMDP.

Here is an outline of our approach: In Appendix B.1 of the full version, we start by defining two properties of allocation rules. The first of these properties is the well-known *cyclic monotonicity*. The second is a new property we define called *compatibility*. Compatibility is a slightly (and strictly) stronger condition than cyclic monotonicity. The main result of this section is a simple formula (of the form of the objective function that appears in SADP) that upper bounds the maximum obtainable revenue using a given allocation rule, as well as a proof that this bound is attainable when the allocation rule is compatible. Both definitions and results can be found in Appendix B.1 of the full version.

Next, we relate SADP to MDMDP using the results of Appendix B.1 of the full version, showing how to view any (possibly suboptimal) solution to a SADP instance as one for a corresponding MDMDP instance and vice versa. We show that for compatible SADP instances, any optimal solution is also optimal in the corresponding MDMDP instance. Furthermore, we show (using the work of Appendix B.1 of the full version) that, for any α-approximate MDMDP solution X, the corresponding SADP solution Y is necessarily an approximate solution to SADP as well, and a lower bound on its approximation ratio as a function of α. Therefore, this constitutes a black-box reduction from approximating compatible instances of SADP to approximating MDMDP. This is presented in Appendix B.2 of the full version.

Finally, in Appendix B.3 of the full version, we give a class of compatible SADP instances, where $\mathcal{V}$ is the class of submodular functions and $\mathcal{F}$ is trivial, for which SADP is impossible to approximate within any polynomial factor unless $NP = RP$. Using the reduction of Appendix B.2 we may immediately conclude that unless $NP = RP$, revenue maximization for a single monotone submodular bidder under trivial feasibility constraints (the seller has one copy of each of n goods and can award any subset to the bidder) is impossible to approximate within any polynomial factor. Note that, on the other hand, welfare is trivial to maximize in this setting: simply give the bidder every item. This section is concluded with the proof of the following theorem. Formal definitions of submodularity, value oracle, demand oracle, and explicit access can be found at the start of Appendix B.3 of the full version.

Theorem 4. *The problems SADP($2^{[n]}$,monotone submodular functions) (for k = poly(n)) and*

MDMDP($2^{[n]}$,monotone submodular functions) (for $k = |T_1| =$ poly(n)) are:

1) *Impossible to approximate within any $1/$poly(n)-factor with only poly(k,n) value oracle queries.*
2) *Impossible to approximate within any $1/$poly(n)-factor with only poly(k,n) demand oracle queries.*
3) *Impossible to approximate within any $1/$poly(n)-factor given explicit access to the input functions in time poly(k,n), unless $NP = RP$.*

V. Conclusions and Future Work

This work provides a poly-time approximation-preserving reduction from truthfully optimizing any objective to algorithmically optimizing that same objective plus virtual welfare. In addition, we show that this reduction is essentially tight when applied to revenue: for appropriately structured instances, any truthful mechanism that approximately optimizes revenue can be converted (in poly-time and a black box way) to an algorithm that approximately optimizes virtual welfare. Using this, we show that no computationally efficient algorithm can maximize revenue for a single monotone submodular bidder within any polynomial factor. Finally, we apply our new reduction to obtain optimal truthful mechanisms for the specific non-linear objective of fractional max-min fairness.

Two important questions motivated by our work are:

1) Is our reduction essentially tight for any objective besides revenue? In other words, are there reductions from SADP to MDMDP for other objectives?

2) For what other objectives can we (approximately) solve SADP? Optimization is a rich area of study with numerous interesting objectives. Designing algorithms to (approximately) optimize any of these objectives plus virtual welfare would imply new truthful mechanisms via our reduction.

References

[1] S. Alaei. Bayesian Combinatorial Auctions: Expanding Single Buyer Mechanisms to Many Buyers. In *the 52nd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2011.

[2] S. Alaei, H. Fu, N. Haghpanah, J. Hartline, and A. Malekian. Bayesian Optimal Auctions via Multi- to Single-agent Reduction. In *the 13th ACM Conference on Electronic Commerce (EC)*, 2012.

[3] I. Ashlagi, S. Dobzinski, and R. Lavi. Optimal lower bounds for anonymous scheduling mechanisms. *Mathematics of Operations Research*, 37(2):244–258, 2012.

[4] X. Bei and Z. Huang. Bayesian Incentive Compatibility via Fractional Assignments. In *the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2011.

[5] A. Bhalgat, S. Gollapudi, and K. Munagala. Optimal Auctions via the Multiplicative Weight Method. *Manuscript, http://www.seas.upenn.edu/ bhalgat/optimal_bic.pdf*, 2013.

[6] P. Briest, S. Chawla, R. Kleinberg, and S. M. Weinberg. Pricing Randomized Allocations. In *the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2010.

[7] P. Briest and P. Krysta. Buying Cheap is Expensive: Hardness of Non-Parametric Multi-Product Pricing. In *the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2007.

[8] D. Buchfuhrer, S. Dughmi, H. Fu, R. Kleinberg, E. Mossel, C. H. Papadimitriou, M. Schapira, Y. Singer, and C. Umans. Inapproximability for VCG-Based Combinatorial Auctions. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2010.

[9] Y. Cai and C. Daskalakis. Extreme-Value Theorems for Optimal Multidimensional Pricing. In *the 52nd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2011.

[10] Y. Cai, C. Daskalakis, and S. M. Weinberg. An Algorithmic Characterization of Multi-Dimensional Mechanisms. In *the 44th Annual ACM Symposium on Theory of Computing (STOC)*, 2012.

[11] Y. Cai, C. Daskalakis, and S. M. Weinberg. Optimal Multi-Dimensional Mechanism Design: Reducing Revenue to Welfare Maximization. In *the 53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2012.

[12] Y. Cai, C. Daskalakis, and S. M. Weinberg. Reducing Revenue to Welfare Maximization : Approximation Algorithms and other Generalizations. In *the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2013.

[13] Y. Cai, C. Daskalakis, and S. M. Weinberg. Understanding Incentives: Mechanism Design Becomes Algorithm Design. *arxiv report*, 2013. http://arxiv.org/abs/1305.4002.

[14] Y. Cai and Z. Huang. Simple and Nearly Optimal Multi-Item Auctions. In *the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2013.

[15] S. Chawla, J. Hartline, D. Malec, and B. Sivan. Prior-Independent Mechanisms for Scheduling. In *Proceedings of 45th ACM Symposium on Theory of Computing (STOC)*, 2013.

[16] S. Chawla, J. D. Hartline, and R. D. Kleinberg. Algorithmic Pricing via Virtual Valuations. In *the 8th ACM Conference on Electronic Commerce (EC)*, 2007.

[17] S. Chawla, J. D. Hartline, D. L. Malec, and B. Sivan. Multi-Parameter Mechanism Design and Sequential Posted Pricing. In *the 42nd ACM Symposium on Theory of Computing (STOC)*, 2010.

[18] S. Chawla, N. Immorlica, and B. Lucier. On the limits of black-box reductions in mechanism design. In *Proceedings of the 44th Symposium on Theory of Computing (STOC)*, 2012.

[19] G. Christodoulou, E. Koutsoupias, and A. Vidali. A lower bound for scheduling mechanisms. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1163–1170. Society for Industrial and Applied Mathematics, 2007.

[20] C. Daskalakis, A. Deckelbaum, and C. Tzamos. The Complexity of Optimal Mechanism Design. *Manuscript: http://arxiv.org/pdf/1211.1703v2.pdf*, 2013.

[21] S. Dobzinski. An Impossibility Result for Truthful Combinatorial Auctions with Submodular Valuations. In *Proceedings of the 43rd ACM Symposium on Theory of Computing (STOC)*, 2011.

[22] S. Dobzinski, H. Fu, and R. D. Kleinberg. Optimal Auctions with Correlated Bidders are Easy. In *the 43rd ACM Symposium on Theory of Computing (STOC)*, 2011.

[23] S. Dobzinski and J. Vondrak. The Computational Complexity of Truthfulness in Combinatorial Auctions. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*, 2012.

[24] S. Hart and N. Nisan. Approximate Revenue Maximization with Multiple Items. In *the 13th ACM Conference on Electronic Commerce (EC)*, 2012.

[25] J. D. Hartline, R. Kleinberg, and A. Malekian. Bayesian Incentive Compatibility via Matchings. In *the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2011.

[26] J. D. Hartline and B. Lucier. Bayesian Algorithmic Mechanism Design. In *the 42nd ACM Symposium on Theory of Computing (STOC)*, 2010.

[27] N. Nisan and A. Ronen. Algorithmic Mechanism Design (Extended Abstract). In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing (STOC)*, 1999.

[28] C. H. Papadimitriou, M. Schapira, and Y. Singer. On the hardness of being truthful. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2008.