

# From Convex Optimization to Randomized Mechanisms: Toward Optimal Combinatorial Auctions

[Extended Abstract] \*

Shaddin Dughmi<sup>†</sup>  
shaddin@cs.stanford.edu

Tim Roughgarden<sup>‡</sup>  
tim@cs.stanford.edu

Qiqi Yan<sup>§</sup>  
qiqiyan@cs.stanford.edu

Department of Computer Science  
Stanford University  
Stanford, CA 94305

## ABSTRACT

We design an expected polynomial time, truthful in expectation,  $(1 - 1/e)$ -approximation mechanism for welfare maximization in a fundamental class of combinatorial auctions. Our results apply to bidders with valuations that are *matroid rank sums (MRS)*, which encompass most concrete examples of submodular functions studied in this context, including coverage functions and matroid weighted-rank functions. Our approximation factor is the best possible, even for known and explicitly given coverage valuations, assuming  $P \neq NP$ . Ours is the first truthful-in-expectation and polynomial-time mechanism to achieve a constant-factor approximation for an  $NP$ -hard welfare maximization problem in combinatorial auctions with heterogeneous goods and restricted valuations.

Our mechanism is an instantiation of a new framework for designing approximation mechanisms based on randomized rounding algorithms. A typical such algorithm first optimizes over a fractional relaxation of the original problem, and then randomly rounds the fractional solution to an integral one. With rare exceptions, such algorithms cannot be converted into truthful mechanisms. The high-level idea of our mechanism design framework is to optimize *directly over the (random) output of the rounding algorithm*, rather than over the *input* to the rounding algorithm. This

\*A full version of this paper is available at <http://arxiv.org/abs/1103.0040>

<sup>†</sup>Supported by NSF Grant CCF-0448664 and a Siebel Foundation Scholarship.

<sup>‡</sup>Supported in part by NSF CAREER Award CCF-0448664, an ONR Young Investigator Award, an ONR PECASE Award, an AFOSR MURI grant, and an Alfred P. Sloan Fellowship.

<sup>§</sup>Supported by a Stanford Graduate Fellowship.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

STOC'11, June 6–8, 2011, San Jose, California, USA.

Copyright 2011 ACM 978-1-4503-0691-1/11/06 ...\$10.00.

approach leads to truthful-in-expectation mechanisms, and these mechanisms can be implemented efficiently when the corresponding objective function is concave. For bidders with MRS valuations, we give a novel randomized rounding algorithm that leads to both a concave objective function and a  $(1 - 1/e)$ -approximation of the optimal welfare.

## Categories and Subject Descriptors

F.0 [Theory of Computation]: General

## General Terms

Algorithms, Economics, Theory

## Keywords

Algorithmic Mechanism Design, Combinatorial Auctions

## 1. INTRODUCTION

The overarching goal of *algorithmic mechanism design* is to design computationally efficient algorithms that solve or approximate fundamental optimization problems in which the underlying data is a priori unknown to the algorithm. A central example in both theory and practice is welfare-maximization in combinatorial auctions. Here, there are  $m$  goods for sale and  $n$  bidders vying for them. Each bidder  $i$  has a private *valuation*  $v_i(S)$  for each subset  $S$  of the goods.<sup>1</sup> The *welfare* of an allocation  $S_1, \dots, S_n$  of the goods to the bidders is  $\sum_{i=1}^n v_i(S_i)$ . Since valuations are initially unknown to the seller, computing a near-optimal allocation requires eliciting information from the (self-interested) bidders, for example via a bid. A *mechanism* is a protocol that extracts such information and computes an allocation of the goods and payments.

The “holy grail” for a mechanism designer is to devise a computationally efficient and incentive-compatible mechanism with an approximation factor that matches the best one known for the (easier) problem in which the underlying data is provided up front.<sup>2</sup> Such results are usually difficult to obtain, and in some cases are provably impossible using

<sup>1</sup>Each bidder has an exponential number of private values; we ignore the attendant representation issues for the moment.

<sup>2</sup>In this paper, by “incentive compatible” we generally mean a (possibly randomized) mechanism such that every partic-

deterministic mechanisms [15, 22]. The space of randomized mechanisms, however, is much more promising as shown recently in [6, 9].<sup>3</sup> This paper provides such a positive result for a fundamental class of combinatorial auctions, via a novel randomized mechanism design framework based on convex optimization.

Algorithmic mechanism design is difficult because incentive compatibility severely limits how the algorithm can compute an outcome, which prohibits use of most of the ingenious approximation algorithms that have been developed for different optimization problems. More concretely, the only general approach known for designing (randomized) truthful mechanisms is via *maximal-in-distributional range (MIDR) algorithms* [6, 9]. An MIDR algorithm fixes a set of distributions over feasible solutions — the *distributional range* — independently of the valuations reported by the self-interested participants, and outputs a random sample from the distribution that maximizes expected (reported) welfare. The *Vickrey-Clarke-Groves (VCG)* payment scheme renders an MIDR algorithm truthful in expectation.

Most approximation algorithms are not MIDR algorithms. Consider, as an example, a *randomized rounding* algorithm for welfare maximization in combinatorial auctions (e.g. [11, 8]). We can view such an algorithm as the composition of two algorithms, a *relaxation algorithm* and a *rounding algorithm*. The relaxation algorithm is deterministic and takes as input the problem data (players’ valuations  $v$ ), and outputs the (fractional) solution to a linear programming relaxation of the welfare-maximization problem that is optimal for the objective function defined by  $v$ . The rounding algorithm is randomized and takes as input this fractional solution and outputs a feasible allocation of the goods to the players. Taken together, these algorithms assign to each input  $v$  a probability distribution  $D_v$  over integral allocations. For almost all known randomized rounding algorithms, there is an input  $v$  such that the expected objective function value  $\mathbb{E}_{y \sim D_v}[v^T y]$  with the distribution  $D_v$  is inferior to that  $\mathbb{E}_{y \sim D_w}[v^T y]$  with a distribution  $D_w$  that the algorithm would produce for a different input  $w$  — and this is a violation of the MIDR property. Informally, such violations are inevitable unless a rounding algorithm is designed explicitly to avoid them, on top of the usual approximation requirements.

The exception that proves the rule is the important and well-known mechanism design framework of Lavi and Swamy [16]. Lavi and Swamy [16] begin with the foothold that the *fractional* welfare maximization problem — the relaxation algorithm above — can be made truthful by charging appropriate VCG payments. Further, they identify a very special type of rounding algorithm that preserves truthfulness: if the expected allocation produced by the rounding algorithm is *always identical to the input to the rounding algorithm*, component-wise, up to some universal scaling factor  $\alpha$ , then composing the two algorithms easily yields an  $\alpha$ -approximate truthful-in-expectation mechanism (after

ipant maximizes its expected payoff by truthfully revealing its information to the mechanism, no matter how the other participants behave. Such mechanisms are called truthful in expectation, and are defined formally in Section 2.2.

<sup>3</sup>We note that the impressively general positive results for implementations in Bayes-Nash equilibria that were recently obtained in [13, 12, 1] do not apply to the stronger incentive-compatibility notions used in this paper and in most of the algorithmic mechanism design literature.

scaling the fractional VCG payments by  $\alpha$ ). Perhaps surprisingly, there are some interesting problems, such as welfare maximization in combinatorial auctions with general valuations, that admit such a rounding algorithm with a best-possible approximation guarantee (assuming  $P \neq NP$ ). However, most  $NP$ -hard welfare maximization problems do not seem to admit good randomized rounding algorithms of the rigid type required by this design framework.

## 1.1 Our Contributions

We introduce a new approach to designing truthful-in-expectation approximation mechanisms based on randomized rounding algorithms; we outline it here for the special case of welfare maximization in combinatorial auctions. The high-level idea is to optimize *directly on the outcome of the rounding algorithm*, rather than merely on the outcome of the relaxation algorithm (the *input* to the rounding algorithm). In other words, let  $r(x)$  denote a randomized rounding algorithm, from fractional allocations to integer allocations. Given players’ valuations  $v$ , we compute a fractional allocation  $x$  that maximizes the expected welfare  $\mathbb{E}_{y \sim r(x)}[v^T y]$  over all fractional allocations  $x$ . This methodology evidently gives MIDR algorithms. This optimization problem is often intractable, but when the rounding algorithm  $r$  and the space of valuations  $v$  are such that the function  $\mathbb{E}_{y \sim r(x)}[v^T y]$  is always concave in  $x$  — in which case we call  $r$  a *convex rounding algorithm* — it can be solved in polynomial time using convex programming (modulo numerical issues that we address later).

We use this design framework to give an expected polynomial time, truthful in expectation,  $(1 - 1/e)$ -approximation mechanism for welfare maximization in combinatorial auctions in which bidders’ valuations are *matroid rank sums (MRS)* — non-negative linear combinations of matroid rank functions on the goods. MRS valuations are submodular and encompass most concrete examples of submodular functions that have been studied in the combinatorial auctions literature, including all coverage functions and matroid weighted-rank functions (see Section 2.4 for formal definitions). Our approximation guarantee is optimal, assuming  $P \neq NP$ , even for the special case of the welfare maximization problem with known and explicitly presented coverage valuations. Our mechanism is the first truthful-in-expectation and polynomial-time mechanism to achieve a constant-factor approximation for any  $NP$ -hard special case of combinatorial auctions that doesn’t assume that there are multiple copies of every type of good. It works with “black-box” valuations, provided that they support a randomized analog of a “value oracle”. We also give a (non-oracle-based) version of the mechanism for explicitly represented coverage valuations (see Appendix A).

## 1.2 Related Work

We discuss only the results most pertinent to this work; see [4] for an introduction to combinatorial auctions, and [3] for a survey of truthful approximation mechanisms for combinatorial auctions.

For the welfare maximization problem in combinatorial auctions with general valuations (assuming only that  $v_i(\emptyset) = 0$  and that  $v_i(S) \leq v_i(T)$  whenever  $S \subseteq T$ ), the best approximation factor possible by a polynomial-time approximation algorithm is roughly  $\min\{\sqrt{m}, n\}$ , where  $n$  and  $m$  are the number of bidders and goods. There are compara-

ble unconditional lower bounds in various oracle models, assuming polynomial communication and unbounded computation [19]; and, assuming that  $P \neq NP$ , for various classes of succinctly represented valuations [18].

These strong negative results for welfare maximization with general valuations motivate the study of important special cases. Numerous special cases have been considered (see [3, Fig 1.2]), and the most well-studied one is for bidders with valuations that are *submodular*, meaning that  $v_i(S \cap T) + v_i(S \cup T) \leq v_i(S) + v_i(T)$  for every bidder  $i$  and bundles  $S, T$  of goods. Submodular functions play a fundamental role in combinatorial optimization, and have a natural economic interpretation in terms of diminishing marginal returns.

Without incentive-compatibility constraints, the welfare maximization problem with submodular bidder valuations is completely solved. Vondrák [24] gave a  $(1 - \frac{1}{e})$ -approximation algorithm for the problem, improving over the  $\frac{1}{2}$ -approximation given in [17]. The algorithm in [24] works in the *value oracle* model, where each valuation  $v$  is modeled as a “black box” that returns the value  $v(S)$  of a queried set  $S$  in a single operation. The approximation factor of  $1 - \frac{1}{e}$  is unconditionally optimal in the value-oracle model (for polynomial communication) [19], and is also optimal (for polynomial time) for certain succinctly represented submodular valuations, assuming  $P \neq NP$  [14].

Despite intense study, prior to this work, there were no truthful-in-expectation and polynomial-time constant-factor approximation mechanisms for welfare maximization with any non-trivial subclass of submodular bidder valuations. The best previous results, which apply to all submodular valuations, are an  $O\left(\frac{\log m}{\log \log m}\right)$  approximation mechanism in the communication complexity model due to Dobzinski, Fu and Kleinberg [7], and an  $O(\log m \log \log m)$  approximation mechanism in the *demand oracle* model due to Dobzinski [5].

The aforementioned works [7, 16] are precursors to our general design framework that optimizes directly over the output of a randomized rounding algorithm. In the framework of Lavi and Swamy [16], the input to and output of the rounding algorithm are assumed to coincide up to a scaling factor, so optimizing over its input (as they do) is equivalent to optimizing over its output (as we do). In the result of Dobzinski et al. [7], optimizing with respect to their “proxy bidders” is equivalent to optimizing over the output of a particular randomized rounding algorithm.

## 2. PRELIMINARIES

### 2.1 Optimization Problems

We consider optimization problems  $\Pi$  of the following general form. Each instance of  $\Pi$  consists of a *feasible set*  $\mathcal{S}$ , and an *objective function*  $w : \mathcal{S} \rightarrow \mathbb{R}$ . The solution to an instance of  $\Pi$  is given by the following optimization problem.

$$\begin{aligned} & \text{maximize} && w(x) \\ & \text{subject to} && x \in \mathcal{S}. \end{aligned} \tag{1}$$

### 2.2 Mechanism Design Basics

We consider mechanism design optimization problems of the form in (1). In such problems, there are  $n$  players, where each player  $i$  has a *valuation function*  $v_i : \mathcal{S} \rightarrow \mathbb{R}$ . We are

concerned with *welfare maximization* problems, where the objective is  $w(x) = \sum_{i=1}^n v_i(x)$ .

We consider direct-revelation mechanisms for optimization mechanism design problems. Such a mechanism comprises an *allocation rule*, which is a function from (hopefully truthfully) reported valuation functions  $v_1, \dots, v_n$  to an outcome  $x \in \mathcal{S}$ , and a *payment rule*, which is a function from reported valuation functions to a required payment from each player. We allow the allocation and payment rules to be randomized.

A mechanism with allocation and payment rules  $\mathcal{A}$  and  $p$  is *truthful in expectation* if every player always maximizes its expected payoff by truthfully reporting its valuation function, meaning that

$$\mathbf{E}[v_i(\mathcal{A}(v)) - p_i(v)] \geq \mathbf{E}[v_i(\mathcal{A}(v'_i, v_{-i})) - p_i(v'_i, v_{-i})] \tag{2}$$

for every player  $i$ , (true) valuation function  $v_i$ , (reported) valuation function  $v'_i$ , and (reported) valuation functions  $v_{-i}$  of the other players. The expectation in (2) is over the coin flips of the mechanism.

The mechanisms that we design can be thought of as randomized variations on the classical VCG mechanism, as we explain next. Recall that the *VCG mechanism* is defined by the (generally intractable) allocation rule that selects the welfare-maximizing outcome with respect to the reported valuation functions, and the payment rule that charges each player  $i$  a bid-independent “pivot term” minus the reported welfare earned by other players in the selected outcome. This (deterministic) mechanism is truthful; see e.g. [20].

Now let  $\text{dist}(\mathcal{S})$  denote the probability distributions over a feasible set  $\mathcal{S}$ , and let  $\mathcal{D} \subseteq \text{dist}(\mathcal{S})$  be a compact subset of them. The corresponding *Maximal in Distributional Range (MIDR)* allocation rule is defined as follows: given reported valuation functions  $v_1, \dots, v_n$ , return an outcome that is sampled randomly from a distribution  $D^* \in \mathcal{D}$  that maximizes the expected welfare  $\mathbf{E}_{x \sim D}[\sum_i v_i(x)]$  over all distributions  $D \in \mathcal{D}$ . Analogous to the VCG mechanism, there is a (randomized) payment rule that can be coupled with this allocation rule to yield a truthful-in-expectation mechanism (see [6]).

### 2.3 Combinatorial Auctions

In *Combinatorial Auctions* there is a set  $[m] = \{1, 2, \dots, m\}$  of items, and a set  $[n] = \{1, 2, \dots, n\}$  of players. Each player  $i$  has a valuation function  $v_i : 2^{[m]} \rightarrow \mathbb{R}^+$  that is normalized ( $v_i(\emptyset) = 0$ ) and monotone ( $v_i(A) \leq v_i(B)$  whenever  $A \subseteq B$ ). A feasible solution is an *allocation*  $(S_1, \dots, S_n)$ , where  $S_i$  denotes the items assigned to player  $i$ , and  $\{S_i\}_i$  are mutually disjoint subsets of  $[m]$ . Player  $i$ ’s value for outcome  $(S_1, \dots, S_n)$  is equal to  $v_i(S_i)$ . The goal is to choose the allocation maximizing *social welfare*:  $\sum_i v_i(S_i)$ .

### 2.4 Matroid Rank Sum Valuations

We now define matroid rank sum valuations. Relevant concepts from matroid theory are reviewed in Appendix C.1.

**DEFINITION 2.1.** *A set function  $v : 2^{[m]} \rightarrow \mathbb{R}^+$  is a matroid rank sum (MRS) function if there exists a family of matroid rank functions  $u_1, \dots, u_\kappa : 2^{[m]} \rightarrow \mathbb{N}$ , and associated non-negative weights  $w_1, \dots, w_\kappa \in \mathbb{R}^+$ , such that  $v(S) = \sum_{\ell=1}^\kappa w_\ell u_\ell(S)$  for all  $S \subseteq [m]$ .*

We do not assume any particular representation of MRS

valuations, and require only oracle access to their (expected) values on certain distributions (see Section 2.5).

MRS functions include most concrete examples of monotone submodular functions that appear in the literature – this includes coverage functions<sup>4</sup>, matroid weighted-rank functions<sup>5</sup>, and all convex combinations thereof. Moreover, as shown in [14],  $1 - 1/e$  is the best approximation possible for MRS valuations in polynomial time, even ignoring strategic considerations.

## 2.5 Lotteries and Value Oracles

A *value oracle* for a valuation  $v : 2^{[m]} \rightarrow \mathbb{R}$  takes as input a set  $S \subseteq [m]$ , and returns  $v(S)$ . We define an analogous oracle that takes in a description of a simple lottery (where each good is included independently with some probability) over sets  $S \subseteq [m]$ , and outputs the expectation of  $v$  over this lottery. For a vector  $x \in [0, 1]^m$  of probabilities on the goods, we use  $F_v(x)$  to denote the expected value of  $v(S)$  over draws  $S \sim D_x$  from this lottery.

**DEFINITION 2.2.** *A lottery-value oracle for set function  $v : 2^{[m]} \rightarrow \mathbb{R}$  takes as input a vector  $x \in [0, 1]^m$ , and outputs*

$$F_v(x) = \mathbb{E}_{S \sim D_x} [v(S)] = \sum_{S \subseteq [m]} v(S) \prod_{j \in S} x_j \prod_{j \notin S} (1 - x_j). \quad (3)$$

In addition to being the natural randomized analog of a value oracle, a lottery-value oracle is easily implemented for various succinctly represented examples of MRS valuations, like explicit coverage functions (see Appendix A).

## 3. CONVEX ROUNDING FRAMEWORK

### 3.1 Relaxations and Rounding Schemes

Let  $\Pi$  be an optimization problem. A *relaxation*  $\Pi'$  of  $\Pi$  defines for every  $(\mathcal{S}, w) \in \Pi$  a convex and compact *relaxed feasible set*  $\mathcal{R} \subseteq \mathbb{R}^m$  that is independent of  $w$  (we suppress the dependence on  $\mathcal{S}$ ); and an *extension*  $w_{\mathcal{R}} : \mathcal{R} \rightarrow \mathbb{R}$  of the objective  $w$  to the relaxed feasible set  $\mathcal{R}$ . This gives the following *relaxed optimization problem*.

$$\begin{aligned} & \text{maximize} && w_{\mathcal{R}}(x) \\ & \text{subject to} && x \in \mathcal{R}. \end{aligned} \quad (4)$$

Generally, the extension is defined so that it is computationally tractable to find a point  $x \in \mathcal{R}$  that maximizes  $w_{\mathcal{R}}(x)$  (possibly approximately).

For example,  $\mathcal{S}$  could be the allocations of  $m$  goods to  $n$  bidders in a combinatorial auction,  $w(x)$  the welfare of an allocation,  $\mathcal{R}$  the feasible region of a linear programming relaxation, and  $w_{\mathcal{R}}$  the natural linear extension of  $w$  to fractional allocations.

<sup>4</sup>A coverage function  $f$  on ground set  $[m]$  designates some set  $\mathcal{L}$  of elements, and  $m$  subsets  $A_1, \dots, A_m \subseteq \mathcal{L}$ , such that  $f(S) = |\cup_{j \in S} A_j|$ . We note that  $\mathcal{L}$  may be an infinite, yet measurable, space. Coverage functions are arguably *the* canonical example of a submodular function, particularly for combinatorial auctions.

<sup>5</sup>This is a generalization of matroid rank functions, where weights are placed on elements of the matroid. It is true, though not immediately obvious, that a matroid weighted-rank function can be expressed as a weighted combination of matroid (unweighted) rank functions – see e.g. [10].

The solution  $x \in \mathcal{R}$  to the relaxed problem need not be in  $\mathcal{S}$ . A *rounding scheme* for relaxation  $\Pi'$  of  $\Pi$  defines for each feasible set  $\mathcal{S}$  of  $\Pi$ , and its corresponding relaxed set  $\mathcal{R}$ , a (possibly randomized) function  $r : \mathcal{R} \rightarrow \mathcal{S}$ . Since our rounding scheme will be randomized, we will frequently use  $r(x)$  to denote the distribution over  $\mathcal{S}$  resulting from rounding the point  $x \in \mathcal{R}$ . Commonly, the rounding scheme satisfies the following approximation guarantee:  $\mathbb{E}_{y \sim r(x)} [w(y)] \geq \alpha \cdot w_{\mathcal{R}}(x)$  for every  $x \in \mathcal{R}$ . In this case, if  $x^*$  maximizes  $w_{\mathcal{R}}$  over  $\mathcal{R}$  and  $w_{\mathcal{R}}$  agrees with  $w$  on  $\mathcal{S}$ , then  $\mathbb{E}_{y \sim r(x^*)} [w(y)] \geq \alpha \cdot \max_{y \in \mathcal{S}} w(y)$ .

### 3.2 Convex Rounding Schemes and MIDR

Our technique is motivated by the following observation: instead of solving the relaxed problem and subsequently rounding the solution, why not *optimize directly on the outcome of the rounding scheme*? In particular, consider the following relaxation of  $\Pi$  that “absorbs” rounding scheme  $r$  into the objective.

$$\begin{aligned} & \text{maximize} && \mathbb{E}_{y \sim r(x)} [w(y)] \\ & \text{subject to} && x \in \mathcal{R}. \end{aligned} \quad (5)$$

The solution to this problem rounds to the best possible distribution in the range of the rounding scheme, over all possible fractional solutions in  $\mathcal{R}$ . While this problem is often intractable, it always leads to an MIDR allocation rule.

---

#### Algorithm 1 MIDR Allocation Rule via Optimizing over Output of Rounding Scheme

---

**Parameter:** Feasible set  $\mathcal{S}$  of  $\Pi$ .

**Parameter:** Relaxed feasible set  $\mathcal{R} \subseteq \mathbb{R}^m$ .

**Parameter:** (Randomized) rounding scheme  $r : \mathcal{R} \rightarrow \mathcal{S}$ .

**Input:** Objective  $w : \mathcal{S} \rightarrow \mathbb{R}$  satisfying  $(\mathcal{S}, w) \in \Pi$ .

**Output:** Feasible solution  $z \in \mathcal{S}$ .

- 1: Let  $x^*$  maximize  $\mathbb{E}_{y \sim r(x)} [w(y)]$  over  $x \in \mathcal{R}$ .
  - 2: Let  $z \sim r(x^*)$ .
- 

LEMMA 3.1. *Algorithm 1 is an MIDR allocation rule.*

We say a rounding scheme  $r : \mathcal{R} \rightarrow \mathcal{S}$  is  $\alpha$ -*approximate* for  $\alpha \leq 1$  if  $w(x) \geq \mathbb{E}_{y \sim r(x)} [w(y)] \geq \alpha \cdot w(x)$  for every  $x \in \mathcal{S}$ . When  $r$  is  $\alpha$ -approximate, so is the allocation rule of Algorithm 1.

LEMMA 3.2. *If  $r$  is an  $\alpha$ -approximate rounding scheme, then Algorithm 1 returns an  $\alpha$ -approximate solution (in expectation) to the original optimization problem (1).*

For most rounding schemes in the approximation algorithms literature, the optimization problem (5) cannot be solved in polynomial time (assuming  $P \neq NP$ ). The reason is that for any rounding scheme that always rounds a feasible solution to itself – i.e.,  $r(x) = x$  for all  $x \in \mathcal{S}$  – an optimal solution to (5) is also optimal for (1). Thus, in this case, hardness of the original problem (1) implies hardness of (5). We conclude that we need to design rounding schemes with the unusual property that  $r(x) \neq x$  for some  $x \in \mathcal{S}$ .

We call a (randomized) rounding scheme  $r : \mathcal{R} \rightarrow \mathcal{S}$  *convex* if  $\mathbb{E}_{y \sim r(x)} [w(y)]$  is concave function of  $x \in \mathcal{R}$ .

LEMMA 3.3. *When  $r$  is a convex rounding scheme for  $\Pi'$ , (5) is a convex optimization problem.*

Under additional technical conditions, discussed in the context of combinatorial auctions in Appendix B, the convex program (5) can be solved efficiently (e.g., using the ellipsoid method). This reduces the design of a polynomial-time  $\alpha$ -approximate MIDR algorithm to designing a polynomial-time  $\alpha$ -approximate convex rounding scheme.

Summarizing, Lemmas 3.1, 3.2, and 3.3 give the following informal theorem.

**THEOREM 3.4.** *(Informal) Let  $\Pi$  be a welfare-maximization optimization problem, and let  $\Pi'$  be a relaxation of  $\Pi$ . If there exists a polynomial-time,  $\alpha$ -approximate, convex rounding scheme for  $\Pi'$ , then there exists a truthful-in-expectation, polynomial-time,  $\alpha$ -approximate mechanism for  $\Pi$ .*

Of course, there is no reason a priori to believe that useful convex rounding schemes – let alone ones computable in polynomial time – exist for any important problems. We show in Section 4 that they do in fact exist and yield new results for an interesting class of combinatorial auctions.

## 4. COMBINATORIAL AUCTIONS

In this section, we use the framework of Section 3 to prove our main result.

**THEOREM 4.1.** *There is a  $(1-1/e)$ -approximate, truthful-in-expectation mechanism for combinatorial auctions with matroid rank sum valuations in the lottery-value oracle model, running in expected poly( $n, m$ ) time.*

We formulate welfare maximization in combinatorial auctions as an optimization problem  $\Pi$ . An instance  $(\mathcal{S}, w) \in \Pi$  is given by the following integer program with feasible set  $\mathcal{S}$  contained in  $\{0, 1\}^{n \times m}$ . Variable  $x_{ij}$  indicates whether item  $j$  is allocated to player  $i$ , and  $w(x)$  denotes the social welfare of allocation  $x$ .

$$\begin{aligned} & \text{maximize} && w(x) = \sum_i v_i(\{j : x_{ij} = 1\}) \\ & \text{subject to} && \sum_i x_{ij} \leq 1, && \text{for } j \in [m]. \\ & && x_{ij} \in \{0, 1\}, && \text{for } i \in [n], j \in [m]. \end{aligned} \tag{6}$$

We let the relaxed feasible set  $\mathcal{R} = \mathcal{R}(\mathcal{S})$  be the result of relaxing the constraints  $x_{ij} \in \{0, 1\}$  of (6) to  $0 \leq x_{ij} \leq 1$ .

We structure the proof of Theorem 4.1 as follows. We define the *Poisson rounding scheme*, which we denote by  $r_{\text{poiss}}$ , in Section 4.1. We prove that  $r_{\text{poiss}}$  is  $(1-1/e)$ -approximate (Lemma 4.2), and convex (Lemma 4.3). Lemmas 3.1, 3.2 and 4.2, taken together, imply that Algorithm 1 when instantiated for combinatorial auctions with  $r = r_{\text{poiss}}$ , is a  $(1-1/e)$ -approximate MIDR allocation rule. Lemma 4.3 reduces implementing this allocation rule to solving a convex program.

In Appendix B, we handle the technical and numerical issues related to solving convex programs. First, we prove that our instantiation of Algorithm 1 for combinatorial auctions can be implemented in expected polynomial time using the ellipsoid method under a simplifying assumption on the numerical conditioning of our convex program (Lemma B.2). Then we show in Section B.3 that the previous assumption can be removed by slightly modifying our algorithm.

Finally, we prove that truth-telling VCG payments can be computed efficiently in Lemma D.1. Taken together, these lemmas complete the proof of Theorem 4.1.

### 4.1 The Poisson Rounding Scheme

In this section we define the *Poisson rounding scheme*, which we denote by  $r_{\text{poiss}}$ . The random map  $r_{\text{poiss}} : \mathcal{R} \rightarrow \mathcal{S}$  renders the the following optimization problem over  $\mathcal{R}$  a convex optimization problem.

$$\begin{aligned} & \text{maximize} && f(x) = \mathbb{E}_{y \sim r_{\text{poiss}}(x)}[w(y)] \\ & \text{subject to} && \sum_i x_{ij} \leq 1, && \text{for } j \in [m]. \\ & && 0 \leq x_{ij} \leq 1, && \text{for } i \in [n], j \in [m]. \end{aligned} \tag{7}$$

We define the Poisson rounding scheme as follows. Given a fractional solution  $x$  to (7), do the following independently for each item  $j$ : assign  $j$  to player  $i$  with probability  $1 - e^{-x_{ij}}$ . (This is well defined since  $1 - e^{-x_{ij}} \leq x_{ij}$  for all players  $i$  and items  $j$ , and  $\sum_i x_{ij} \leq 1$  for all items  $j$ .) We make this more precise in Algorithm 2. For clarity, we represent an allocation as a function from items to players, with an additional null player  $*$  reserved for items that are left unassigned.

---

#### Algorithm 2 The Poisson Rounding Scheme $r_{\text{poiss}}$

---

**Input:** Fractional allocation  $x$  with  $\sum_i x_{ij} \leq 1$  for all  $j$ , and  $0 \leq x_{ij} \leq 1$  for all  $i, j$ .  
**Output:** Feasible allocation  $a : [m] \rightarrow [n] \cup \{*\}$ .  
1: **for**  $j = 1, \dots, m$  **do**  
2:   Draw  $p_j$  uniformly at random from  $[0, 1]$ .  
3:   **if**  $\sum_i (1 - e^{-x_{ij}}) \geq p_j$  **then**  
4:     Let  $a(j)$  be the minimum index such that  $\sum_{i \leq a(j)} (1 - e^{-x_{ij}}) \geq p_j$ .  
5:   **else**  
6:      $a(j) = *$   
7:   **end if**  
8: **end for**

---

The Poisson rounding scheme is  $(1-1/e)$ -approximate and convex. The proof of Lemma 4.2 is not difficult, and we sketch it in the appendix. We prove Lemma 4.3 in Section 4.3. As a warm-up, we first present a simplified proof of Lemma 4.3 for the special case of coverage valuations in Section 4.2.

**LEMMA 4.2.** *The Poisson rounding scheme is  $(1-1/e)$ -approximate when valuations are submodular.*

**LEMMA 4.3.** *The Poisson rounding scheme is convex when player valuations are matroid rank sum functions.*

### 4.2 Warm-up: Convexity for Coverage Valuations

In this section, we prove the special case of Lemma 4.3 for coverage valuations, as defined in Section 2.4. For each player  $i \in [n]$ , let  $\mathcal{L}^i$  be a ground set and  $A_1^i, \dots, A_m^i \subseteq \mathcal{L}^i$  be such that  $v_i(S) = |\cup_{j \in S} A_j^i|$  for each  $S \subseteq [m]$ . Let  $\mathcal{R}$  denote the feasible set of (7). Fix  $x \in \mathcal{R}$ , and let  $(S_1, \dots, S_n) \sim r_{\text{poiss}}(x)$ . The expected social welfare of the Poisson rounding scheme at  $x$  can be written as follows.

$$\begin{aligned} \mathbb{E} \left[ \sum_i v_i(S_i) \right] &= \sum_i \mathbb{E}[v_i(S_i)] \\ &= \sum_i \mathbb{E}[|\cup_{j \in S_i} A_j^i|] \\ &= \sum_i \sum_{\ell \in \mathcal{L}^i} \Pr[\ell \in \cup_{j \in S_i} A_j^i] \end{aligned}$$

The term  $\Pr[\ell \in \cup_{j \in S_i} A_j^i]$  can be interpreted as the probability that element  $\ell$  is covered by an item in  $S_i$ , where an item  $j \in [m]$  covers  $\ell \in \mathcal{L}^i$  if  $\ell \in A_j^i$ . Let  $C_\ell^i$  be the set of items that cover  $\ell \in \mathcal{L}^i$ . Element  $\ell \in \mathcal{L}^i$  is covered by  $S_i$  precisely when  $C_\ell^i \cap S_i \neq \emptyset$ . Each item  $j \in C_\ell^i$  is included in  $S_i$  independently with probability  $1 - e^{-x_{ij}}$ . Therefore, the probability  $\ell \in \mathcal{L}^i$  is covered by  $S_i$  equals  $1 - \prod_{j \in C_\ell^i} e^{-x_{ij}} = 1 - e^{-\sum_{j \in C_\ell^i} x_{ij}}$ . Therefore,

$$\mathbb{E} \left[ \sum_i v_i(S_i) \right] = \sum_i \sum_{\ell \in \mathcal{L}^i} \left( 1 - e^{-\sum_{j \in C_\ell^i} x_{ij}} \right)$$

Each term in this summation is concave in  $x$ , completing the proof.

### 4.3 Convexity for MRS Valuations

In this section, we will prove Lemma 4.3 in its full generality. First, we define a discrete analogue of a hessian matrix for set functions, and show that these discrete Hessians are negative semi-definite for matroid rank sum functions.

**DEFINITION 4.4.** *Let  $v : 2^{[m]} \rightarrow \mathbb{R}$  be a set function. For  $S \subseteq [m]$ , we define the discrete Hessian matrix  $\mathcal{H}_S^v \in \mathbb{R}^{m \times m}$  of  $v$  at  $S$  as follows:*

$$\mathcal{H}_S^v(j, k) = v(S \cup \{j, k\}) - v(S \cup \{j\}) - v(S \cup \{k\}) + v(S) \quad (8)$$

for  $j, k \in [m]$ .

**CLAIM 4.5.** *If  $v : 2^{[m]} \rightarrow \mathbb{R}^+$  is a matroid rank sum function, then  $\mathcal{H}_S^v$  is negative semi-definite for each  $S \subseteq [m]$ .*

**PROOF.** We observe that  $\mathcal{H}_S^v$  is linear in  $v$ , and recall that a non-negative weighted-sum of negative semi-definite matrices is negative semi-definite. Therefore, it is sufficient to prove this claim when  $v$  is a matroid rank function.

Let  $v$  be a matroid rank function, and fix  $S$ . Applying the matroid axioms reveals that there are two cases:

$$\mathcal{H}_S^v(j, k) = \begin{cases} -1 & \text{if } v(S \cup \{j\}) = v(S \cup \{k\}) = \\ & v(S \cup \{j, k\}) = v(S) + 1, \\ 0 & \text{otherwise} \end{cases}$$

Applying the matroid exchange property reveals that the binary matrix  $-\mathcal{H}_S^v$  encodes a *transitive* relation on  $[m]$  – namely, if  $-\mathcal{H}_S^v(j, k) = 1$  and  $-\mathcal{H}_S^v(k, \ell) = 1$  then  $-\mathcal{H}_S^v(j, \ell) = 1$ . Moreover, it is apparent by definition that  $-\mathcal{H}_S^v$  is symmetric. A binary matrix encoding a symmetric and transitive relation is a block diagonal matrix where each diagonal block is an all-ones or all-zeros sub-matrix. It is known, and easy to check, that such a matrix is positive semi-definite. Therefore  $\mathcal{H}_S^v$  is negative semi-definite.  $\square$

We now return to Lemma 4.3. Fix  $n, m$ , and MRS valuations  $\{v_i\}_{i=1}^n$ , and let  $\mathcal{R}$  denote the feasible set of mathematical program (7). Let  $(S_1, \dots, S_n) \sim r_{\text{poiss}}(x)$  be the (random) allocation computed by the Poisson rounding scheme for point  $x \in \mathcal{R}$ . The expected welfare  $\mathbb{E}[w(r_{\text{poiss}}(x))]$  can be written as  $\mathbb{E}[\sum_{i=1}^n v_i(S_i)]$ , where the expectation is taken over the internal random coins of the rounding scheme. By linearity of expectation, as well as the fact that the sum of concave functions is concave, it suffices to show that  $\mathbf{E}[v_i(S_i)]$  is a concave function of  $x$  for an arbitrary player  $i$  with MRS valuation  $v_i$ .

Fix player  $i$ , and use  $x_j, v, S$  as short-hand for  $x_{ij}, v_i, S_i$  respectively. The Poisson rounding scheme includes each item  $j$  in  $S$  independently with probability  $1 - e^{-x_j}$ . We can now write the expected value of player  $i$  as the following function  $G_v : \mathbb{R}^m \rightarrow \mathbb{R}$ :

$$G_v(x_1, \dots, x_m) = \sum_{S \subseteq [m]} v(S) \prod_{j \in S} (1 - e^{-x_j}) \prod_{j \notin S} e^{-x_j} \quad (9)$$

The following claim, combined with Claim 4.5, completes the proof of Lemma 4.3.

**CLAIM 4.6.** *If all discrete Hessians of  $v$  are negative semi-definite, then  $G_v$  is concave.*

**PROOF.** Assume  $\mathcal{H}_S^v$  is negative semi-definite for each  $S \subseteq [m]$ . We work with  $G_v$  as expressed in equation (9). We will show that the hessian matrix of  $G_v$  at an arbitrary  $x \in \mathbb{R}^m$  is negative semi-definite, which is a sufficient condition for concavity. We take the mixed-derivative of  $G_v$  with respect to  $x_j$  and  $x_k$  (possibly  $j = k$ ).

$$\begin{aligned} \frac{\partial^2 G_v(x)}{\partial x_j \partial x_k} &= \sum_{S \subseteq [m] \setminus \{j, k\}} \left( \prod_{\ell \in S} (1 - e^{-x_\ell}) \prod_{\ell \in [m] \setminus S} e^{-x_\ell} \right. \\ &\quad \left. (v(S) - v(S \cup \{j\}) - v(S \cup \{k\}) + v(S \cup \{j, k\})) \right) \\ &= \sum_{S \subseteq [m]} \left( \prod_{\ell \in S} (1 - e^{-x_\ell}) \prod_{\ell \in [m] \setminus S} e^{-x_\ell} \right. \\ &\quad \left. (v(S) - v(S \cup \{j\}) - v(S \cup \{k\}) + v(S \cup \{j, k\})) \right) \\ &= \sum_{S \subseteq [m]} \prod_{\ell \in S} (1 - e^{-x_\ell}) \prod_{\ell \in [m] \setminus S} e^{-x_\ell} \mathcal{H}_S^v(j, k) \end{aligned}$$

The first equality follows by appropriately grouping the terms of equation (9) before differentiating. The second equality follows from the fact that  $v(S) - v(S \cup \{j\}) - v(S \cup \{k\}) + v(S \cup \{j, k\}) = 0$  when  $S$  includes either of  $j$  and  $k$ . The last equality follows by definition of  $\mathcal{H}_S^v$ .

The above derivation immediately implies that we can write the hessian matrix of  $G_v(x)$  as a non-negative weighted sum of discrete hessian matrices.

$$\nabla^2 G_v(x) = \sum_{S \subseteq [m]} \prod_{\ell \in S} (1 - e^{-x_\ell}) \prod_{\ell \in [m] \setminus S} e^{-x_\ell} \mathcal{H}_S^v \quad (10)$$

A non-negative weighted-sum of negative semi-definite matrices is negative semi-definite. This completes the proof of the claim.  $\square$

## Acknowledgments

We thank Ittai Abraham, Moshe Babaioff, Bobby Kleinberg, and Jan Vondrák for helpful discussions and comments.

## 5. REFERENCES

- [1] X. Bei and Z. Huang. Towards optimal bayesian algorithmic mechanism design. In *Proc. 22nd ACM Symp. on Discrete Algorithms (SODA)*, 2011.
- [2] A. Ben-Tal and A. Nemirovski. *Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications*. SIAM, 2001.

- [3] L. Blumrosen and N. Nisan. Combinatorial auctions (a survey). In N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani, editors, *Algorithmic Game Theory*. Cambridge University Press, 2007.
- [4] P. Cramton, Y. Shoham, and R. S. (Editors). *Combinatorial Auctions*. MIT Press., 2006.
- [5] S. Dobzinski. Two randomized mechanisms for combinatorial auctions. In *Proc. 10th Intl. Workshop on Approximation Algorithms for Combinatorial Optimization Problems*, 2007.
- [6] S. Dobzinski and S. Dughmi. On the power of randomization in algorithmic mechanism design. In *Proc. 50th IEEE Symp. on Foundations of Computer Science (FOCS)*, 2009.
- [7] S. Dobzinski, H. Fu, and R. Kleinberg. Truthfulness via proxies. *CoRR*, abs/1011.3232, 2010.
- [8] S. Dobzinski and M. Schapira. An improved approximation algorithm for combinatorial auctions with submodular bidders. In *Proc. 17th ACM Symp. on Discrete Algorithms (SODA)*, 2006.
- [9] S. Dughmi and T. Roughgarden. Black-box randomized reductions in algorithmic mechanism design. In *Proc. 51st IEEE Symp. on Foundations of Computer Science (FOCS)*, 2010.
- [10] S. Dughmi, T. Roughgarden, and M. Sundararajan. Revenue submodularity. In *Proc. 11th ACM Conf. on Electronic Commerce (EC)*, 2009.
- [11] U. Feige. On maximizing welfare where the utility functions are subadditive. In *Proc. 37th ACM Symp. on Theory of Computing (STOC)*, 2006.
- [12] J. Hartline, R. Kleinberg, and A. Malekian. Multi-parameter bayesian algorithmic mechanism design. In *Proc. 22nd ACM Symp. on Discrete Algorithms (SODA)*, 2011.
- [13] J. D. Hartline and B. Lucier. Bayesian algorithmic mechanism design. In *Proc. 41st ACM Symp. on Theory of Computing (STOC)*, 2010.
- [14] S. Khot, R. J. Lipton, E. Markakis, and A. Mehta. Inapproximability results for combinatorial auctions with submodular utility functions. *Algorithmica*, 52(1):3–18, 2008.
- [15] R. Lavi, A. Mu’alem, and N. Nisan. Towards a characterization of truthful combinatorial auctions. In *Proc. 44th IEEE Symp. on Foundations of Computer Science (FOCS)*, 2003.
- [16] R. Lavi and C. Swamy. Truthful and near-optimal mechanism design via linear programming. In *Proc. 46th IEEE Symp. on Foundations of Computer Science (FOCS)*, 2005.
- [17] B. Lehmann, D. Lehmann, and N. Nisan. Combinatorial auctions with decreasing marginal utilities. In *Proc. 3rd ACM Conf. on Electronic Commerce (EC)*, 2001.
- [18] D. Lehmann, L. I. O’Callaghan, and Y. Shoham. Truth revelation in approximately efficient combinatorial auctions. In *JACM* 49(5), pages 577–602, Sept. 2002.
- [19] V. Mirrokni, M. Schapira, and J. Vondrák. Tight information-theoretic lower bounds for welfare maximization in combinatorial auctions. In *Proc. 10th ACM Conf. on Electronic Commerce (EC)*, 2008.
- [20] N. Nisan. Introduction to mechanism design (for computer scientists). In N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani, editors, *Algorithmic Game Theory*. Cambridge University Press, 2007.
- [21] J. G. Oxley. *Matroid Theory*. Oxford University Press, 1992.
- [22] C. Papadimitriou, M. Schapira, and Y. Singer. On the hardness of being truthful. In *Proc. 49th IEEE Symp. on Foundations of Computer Science (FOCS)*, 2008.
- [23] A. Schrijver. *Combinatorial Optimization*. Springer, 2003.
- [24] J. Vondrák. Optimal approximation for the submodular welfare problem in the value oracle model. In *Proc. 39th ACM Symp. on Theory of Computing (STOC)*, 2008.

## APPENDIX

### A. EXPLICIT COVERAGE VALUATIONS

In this section, we apply our mechanism to explicitly represented coverage valuations. This demonstrates the utility of our mechanism in a concrete, non-oracle-based setting. In the full version of this paper, we also show that deterministic VCG-based mechanisms cannot approximate this problem by a factor better than  $n$  in polynomial-time unless  $NP \subseteq P/poly$ , establishing the first separation between randomized and deterministic VCG-based mechanisms in the *computational complexity* model.

An  $n$  player,  $m$  item instance combinatorial auctions with *explicit coverage valuations* is described as follows. For each player  $i$ , there is a finite set  $\mathcal{L}^i$ , and a family  $A_1^i, \dots, A_m^i$  of subsets of  $\mathcal{L}^i$ . The valuation function of player  $i$  is then defined as  $v_i(S) = |\cup_{j \in S} A_j^i|$ . The set system  $(\mathcal{L}^i, \{A_j^i\}_{j=1}^m)$  is encoded explicitly as a bi-partite graph.

As discussed previously, MRS valuations include all coverage valuations. Therefore, in order to implement the MIDR allocation rule of Section 4 for this problem, it suffices to answer lottery-value queries in time polynomial in the number of bits encoding the instance.

CLAIM A.1. *In combinatorial auctions with explicit coverage valuations, lottery-value queries can be answered in time polynomial in the length of the encoding of the instance.*

PROOF. Let  $v : 2^{[m]} \rightarrow \mathbb{R}^+$  be a coverage valuation presented explicitly as a set system  $(\mathcal{L}, \{A_j\}_{j=1}^m)$ , and let  $x \in [0, 1]^m$ . Let  $S$  be a random set that includes each  $j \in [m]$  independently with probability  $x_j$ . The outcome of the lottery value oracle of  $v$  evaluated at  $x$  is equal to the sum, over all  $\ell \in \mathcal{L}$ , of the probability that  $\ell$  is “covered” by  $S$  – specifically,  $\sum_{\ell \in \mathcal{L}} \Pr[\ell \in \cup_{j \in S} A_j]$ . It is easy to verify that a term of this sum can be expressed as the following closed form expression.

$$\Pr[\ell \in \cup_{j \in S} A_j] = 1 - \prod_{j: A_j \ni \ell} (1 - x_j)$$

This expression can be evaluated in time polynomial in the representation of the set system. This completes the proof.  $\square$

Claim A.1 implies the following Theorem.

THEOREM A.2. *There is an expected polynomial-time,  $(1 - 1/e)$ -approximate, truthful-in-expectation mechanism for combinatorial auctions with explicit coverage valuations.*

## B. SOLVING THE CONVEX PROGRAM

In this section, we overcome some technical difficulties related to the solvability of convex programs. We show in Section B.1 that, in the lottery-value oracle model, the four conditions for “solvability” of convex programs, as stated in Fact C.3, are easily satisfied for convex program (7). However, an additional challenge remains: “solving” a convex program – as in Definition C.2 – returns an approximately optimal solution. Indeed the optimal solution of a convex program may be irrational in general, so this is un-avoidable.

We show how to overcome this difficulty if we settle for polynomial runtime in expectation. While the optimal solution  $x^*$  of (7) cannot be computed explicitly, the random variable  $r_{\text{poiss}}(x^*)$  can be sampled in expected polynomial time. The key idea is the following: *sampling the random variable  $r_{\text{poiss}}(x^*)$  rarely requires precise knowledge of  $x^*$* . Depending on the coin flips of  $r_{\text{poiss}}$ , we decide how accurately we need to solve convex program (7) in order to compute  $r_{\text{poiss}}(x^*)$ . Roughly speaking, we show that the probability of requiring a  $(1 - \epsilon)$ -approximation falls exponentially in  $\frac{1}{\epsilon}$ . As a result, we can sample  $r_{\text{poiss}}(x^*)$  in expected polynomial time. We implement this plan in Section B.2 under the simplifying assumption that convex program (7) is *well-conditioned* – i.e. is “sufficiently concave” everywhere. In Section B.3, we show how to remove that assumption by slightly modifying our algorithm.

### B.1 Approximating the Convex Program

CLAIM B.1. *There is an algorithm for Combinatorial Auctions with MRS valuations in the lottery-value oracle model that takes as input an instance of the problem and an approximation parameter  $\epsilon > 0$ , runs in  $\text{poly}(n, m, \log(1/\epsilon))$  time, and returns a  $(1 - \epsilon)$ -approximate solution to convex program (7).*

It suffices to show that the four conditions of Fact C.3 are satisfied in our setting. The first three are immediate from elementary combinatorial optimization (see for example [23]). It remains to show that the first-order oracle, as defined in Fact C.3, can be implemented in polynomial-time in the lottery-value oracle model. The objective  $f(x)$  of convex program (7) can, by definition, be written as

$$f(x) = \sum_i G_{v_i}(x_i),$$

where  $v_i$  is the valuation function of player  $i$ ,  $x_i$  is the vector  $(x_{i1}, \dots, x_{im})$ , and  $G_{v_i}$  is as defined in (9). By definition,  $G_{v_i}(x_i)$  is the outcome of querying the lottery-value oracle of player  $i$  with  $(1 - e^{-x_{i1}}, \dots, 1 - e^{-x_{im}})$ . Therefore, we can evaluate  $f(x)$  using  $n$  lottery-value query, one for each player. It remains to show that we can also evaluate the (multi-variate) derivative  $\nabla f(x)$  of  $f(x)$ . Using definition (9), we take the partial derivative corresponding to  $x_{ij}$ . By rearranging the sum appropriately, we get that

$$\frac{\partial f}{\partial x_{ij}}(x) = e^{-x_{ij}} \left( F_{v_i}((1 - e^{-x_{i1}}, \dots, 1 - e^{-x_{im}}) \vee 1_j) - F_{v_i}((1 - e^{-x_{i1}}, \dots, 1 - e^{-x_{im}}) \wedge 0_j) \right),$$

where  $F_{v_i}$  is as defined in Equation (3). Here,  $\vee$  and  $\wedge$  denote entry-wise minimum and maximum respectively,  $1_j$  denotes the vector with all entries equal to 1 except for a

1 at position  $j$ , and  $0_j$  denotes the vector with all entries equal to 0 except for a 1 at position  $j$ . It is clear that this entry of the gradient of  $f$  can be evaluated using two lottery-value queries. Therefore,  $\nabla f(x)$  can be evaluated using  $2n$  lottery-value queries, 2 for each player. This completes the proof of Claim B.1.

### B.2 The Well-Conditioned Case

In this section, we make the following simplifying assumption: The objective function  $f(x)$  of convex program (7), when restricted to any line in the feasible set  $\mathcal{R}$ , has a second derivative of magnitude at least  $\lambda = \frac{\sum_{i=1}^n v_i([m])}{2^{\text{poly}(n, m)}}$  everywhere, where the polynomial in the denominator may be arbitrary. This is equivalent to requiring that every eigenvalue of the Hessian matrix of  $f(x)$  has magnitude at least  $\lambda$  when evaluated at any point in  $\mathcal{R}$ . Under this assumption, we prove Lemma B.2.

LEMMA B.2. *Assume the magnitude of the second derivative of  $f(x)$  is at least  $\lambda = \frac{\sum_{i=1}^n v_i([m])}{2^{\text{poly}(n, m)}}$  everywhere. Algorithm 1, instantiated for combinatorial auctions with  $r = r_{\text{poiss}}$ , can be simulated in time polynomial in  $n$  and  $m$  in expectation.*

Let  $x^*$  be the optimal solution to convex program (7). Algorithm 1 allocates items according to the distribution  $r_{\text{poiss}}(x^*)$ . The Poisson rounding scheme requires making  $m$  independent decisions, one for each item  $j$ . Therefore, we fix item  $j$  and show how to simulate this decision. It suffices to do the following in expected polynomial time: flip uniform coin  $p_j \in [0, 1]$ , and find the minimum index  $a(j)$  (if any) such that  $\sum_{i \leq a(j)} (1 - e^{-x^{*ij}}) \geq p_j$ . For most realizations of  $p_j$ , this can be decided using only coarse estimates  $\tilde{x}_{ij}$  to  $x_{ij}^*$ . Assume we have an *estimation oracle* for  $x^*$  that, on input  $\delta$ , returns a  $\delta$ -estimate  $\tilde{x}$  of  $x^*$ : Specifically,  $\tilde{x}_{ij} - x_{ij}^* \leq \delta$  for each  $i$ . When  $p_j$  falls outside the “uncertainty zones” of  $\tilde{x}$ , such as when  $|p_j - \sum_{i' \leq i} (1 - e^{-\tilde{x}_{i'j}})| > \delta n$  for each  $i \in [n]$ , it is easy to see that we can correctly determine  $a(j)$  by using  $\tilde{x}$  in lieu of  $x$ . The total measure of the uncertainty zones of  $\tilde{x}$  is at most  $2n^2\delta$ , therefore  $p_j$  lands outside the uncertainty zones with probability at least  $1 - 2n^2\delta$ . The following claim shows that if the estimation oracle for  $x^*$  can be implemented in time polynomial in  $\log(1/\delta)$ , then we can simulate the Poisson rounding procedure in expected polynomial time.

CLAIM B.3. *Let  $x^*$  be the optimal solution of convex program (7). Assume access to a subroutine  $B(\delta)$  that returns a  $\delta$ -estimate of  $x^*$  in time  $\text{poly}(n, m, \log(1/\delta))$ . Algorithm (1) with  $r = r_{\text{poiss}}$  can be simulated in expected  $\text{poly}(n, m)$  time.*

PROOF. It suffices to show that we can simulate the allocation of an item  $j$  by Algorithm (2) on input  $x^*$ . Draw  $p_j \in [0, 1]$  uniformly at random. Start with  $\delta = \delta_0 = \frac{1}{2n^2}$ . Let  $\tilde{x} = B(\delta)$ . While  $|p_j - \sum_{i' \leq i} (1 - e^{-\tilde{x}_{i'j}})| \leq \delta n$  for some  $i \in [n]$  (i.e.  $p_j$  may fall inside an “uncertainty zone”) do the following: let  $\delta = \delta/2$ ,  $\tilde{x} = B(\delta)$  and repeat. After the loop terminates, we have a sufficiently accurate estimate of  $x^*$  to calculate  $a(j)$  as in Algorithm (2).

It is easy to see that the above procedure is a faithful simulation of Algorithm (2) on  $x^*$ . It remains to bound its expected running time. Let  $\delta_k = \frac{1}{2^{k+1}n^2}$  denote the value of  $\delta$  at the  $k$ 'th iteration. By assumption, the  $k$ 'th iteration takes



$\text{poly}(n, m, \log(1/\delta_k)) = \text{poly}(n, m, \log(2^{k+1}n^2)) = \text{poly}(n, m, k)$  time. The probability this procedure does not terminate after  $k$  iterations is at most  $2n^2\delta_k = 1/2^k$ . Taken together, these two facts and a simple geometric summation imply that the expected runtime is polynomial in  $n$  and  $m$ .  $\square$

It remains to show that the estimation oracle  $B(\delta)$  can be implemented in  $\text{poly}(n, m, \log(1/\delta))$  time. At first blush, one may expect that the ellipsoid method can be used in the usual manner here. However, there is one complication: we require an estimate  $\tilde{x}$  that is close to  $x^*$  in *solution space* rather than in terms of objective value. Using our assumption on the curvature of  $f(x)$ , we will reduce finding a  $\delta$ -estimate of  $x^*$  to finding an  $1 - \epsilon(\delta)$  approximate solution to convex program (7). The dependence of  $\epsilon$  on  $\delta$  will be such that  $\epsilon \geq \text{poly}(\delta)/2^{\text{poly}(n, m)}$ , thereby we can invoke Claim B.1 to deduce that  $B(\delta)$  can be implemented in  $\text{poly}(n, m, \log(1/\delta))$  time.

Let  $\epsilon = \epsilon(\delta) = \frac{\delta^2\lambda}{2\sum_i v_i([m])}$ . Plugging in the definition of  $\lambda$ , we deduce that  $\epsilon \geq \delta^2/2^{\text{poly}(n, m)}$ , which is the desired dependence. It remains to show that if  $\tilde{x}$  is  $(1 - \epsilon)$ -approximate solution to (7), then  $\tilde{x}$  is also a  $\delta$ -estimate of  $x^*$ .

Using the fact that  $f(x)$  is concave, and moreover its second derivative has magnitude at least  $\lambda$ , it a simple exercise to bound distance of any point  $x$  from the optimal point  $x^*$  in terms of its sub-optimality  $f(x^*) - f(x)$ , as follows:

$$f(x^*) - f(x) \geq \frac{\lambda}{2} \|x - x^*\|^2. \quad (11)$$

Assume  $\tilde{x}$  is a  $(1 - \epsilon)$ -approximate solution to (7). Equation (11) implies that

$$\|\tilde{x} - x^*\|^2 \leq \frac{2}{\lambda} \epsilon f(x^*) = \frac{\delta^2}{\sum_i v_i([m])} f(x^*) \leq \delta^2,$$

where the last inequality follows from the fact that  $\sum_i v_i([m])$  is an upper-bound on the optimal value  $f(x^*)$ . Therefore,  $\|x - x^*\| \leq \delta$ , as needed. This completes the proof of Lemma B.2.

### B.3 Guaranteeing Good Conditioning

In this section, we propose a modification  $r_{\text{poiss}}^+$  of the Poisson rounding scheme  $r_{\text{poiss}}$ . We will argue that  $r_{\text{poiss}}^+$  satisfies all the properties of  $r_{\text{poiss}}$  established so far, with one exception: the approximation guarantee of Lemma 4.2 is reduced to  $1 - 1/e - 2^{-2mn}$ . Then we will show that  $r_{\text{poiss}}^+$  satisfies the curvature assumption of Lemma B.2, demonstrating that said assumption may be removed. Therefore Algorithm 1, instantiated with  $r = r_{\text{poiss}}^+$  for combinatorial auctions with MRS valuations in the lottery-value oracle model, is  $(1 - 1/e - 2^{-2mn})$  approximate and can be implemented in expected  $\text{poly}(n, m)$  time. Finally, we show in Remark B.4 how to recover the  $2^{-2mn}$  term to get a clean  $1 - 1/e$  approximation ratio, as claimed in Theorem 4.1.

Let  $\mu = 2^{-2nm}$ . We define  $r_{\text{poiss}}^+$  in Algorithm 3. Intuitively,  $r_{\text{poiss}}^+$  at first makes a tentative allocation using  $r_{\text{poiss}}$ . Then, it cancels said allocation with small probability  $\mu$ . Finally, with probability  $\beta$  it chooses a random “lucky winner”  $i^*$  and gives him all the items.  $\beta$  is defined as the fraction of items allocated in the original tentative allocation. The motivation behind this seemingly bizarre definition of  $r_{\text{poiss}}^+$  is purely technical: as we will see, it can be thought of as adding “concave noise” to  $r_{\text{poiss}}$ .

---

#### Algorithm 3 Modified Poisson Rounding Scheme $r_{\text{poiss}}^+$

---

**Input:** Fractional allocation  $x$  with  $\sum_i x_{ij} \leq 1$  for all  $j$ , and  $0 \leq x_{ij} \leq 1$  for all  $i, j$ .

**Output:** Feasible allocation  $(S_1, \dots, S_n)$ .

- 1: Let  $(S_1, \dots, S_n) \sim r_{\text{poiss}}(x)$ .
  - 2: Let  $\beta = \frac{\sum_i |S_i|}{m}$ .
  - 3: Draw  $q_1 \in [0, 1]$  uniformly at random.
  - 4: **if**  $q_1 \in [0, \mu]$  **then**
  - 5:   Let  $(S_1, \dots, S_n) = (\emptyset, \emptyset, \dots, \emptyset)$ .
  - 6:   Draw  $q_2 \in [0, 1]$  uniformly at random.
  - 7:   **if**  $q_2 \in [0, \beta]$  **then**
  - 8:     Choose a player  $i^*$  uniformly at random.
  - 9:     Let  $S_{i^*} = [m]$ , and  $S_i = \emptyset$  for all  $i \neq i^*$ .
  - 10:   **end if**
  - 11: **end if**
- 

We can write the expected welfare  $\mathbb{E}[w(r_{\text{poiss}}^+(x))]$  as follows. We use linearity of expectations and the fact that  $\beta$  is independent of the choice of  $i^*$  to simplify the expression.

$$\begin{aligned} \mathbb{E}[w(r_{\text{poiss}}^+(x))] &= \mathbb{E}[(1 - \mu)w(r_{\text{poiss}}(x)) + \mu\beta v_{i^*}([m])] \\ &= (1 - \mu) \mathbb{E}[w(r_{\text{poiss}}(x))] + \mu \mathbb{E}[\beta] \mathbb{E}[v_{i^*}([m])] \\ &= (1 - \mu) \mathbb{E}[w(r_{\text{poiss}}(x))] + \mu \mathbb{E}[\beta] \frac{\sum_i v_i([m])}{n} \end{aligned}$$

Observe that  $r_{\text{poiss}}$  allocates an item  $j$  with probability  $\sum_i (1 - e^{-x_{ij}})$ . Therefore, the expectation of  $\beta$  is  $\frac{\sum_{i,j} (1 - e^{-x_{ij}})}{m}$ . This gives:

$$\begin{aligned} \mathbb{E}[w(r_{\text{poiss}}^+(x))] &= (1 - \mu) \mathbb{E}[w(r_{\text{poiss}}(x))] \\ &\quad + \frac{\mu}{mn} \sum_i v_i([m]) \sum_{i,j} (1 - e^{-x_{ij}}). \quad (12) \end{aligned}$$

It is clear that the expected welfare when using  $r = r_{\text{poiss}}^+$  is within  $1 - \mu = 1 - 2^{-2nm}$  of the expected welfare when using  $r = r_{\text{poiss}}$  in the instantiation of Algorithm 1. Using Lemma 4.2, we conclude that  $r_{\text{poiss}}^+$  is a  $(1 - 1/e - 2^{-2nm})$ -approximate rounding scheme. Moreover, using Lemma 4.3, as well as the fact that  $(1 - e^{-x_{ij}})$  is a concave function, we conclude that  $r_{\text{poiss}}^+$  is a convex rounding scheme. Therefore, this establishes the analogues of Lemmas 4.2 and 4.3 for  $r_{\text{poiss}}^+$ . It is elementary to verify that our proof of Lemma B.2 can be adapted to  $r_{\text{poiss}}^+$  as well.

It remains to show that  $r_{\text{poiss}}^+$  is “sufficiently concave”. This would establish that the conditioning assumption of Section B.2 is unnecessary for  $r_{\text{poiss}}^+$ . We will show that expression (12) is a concave function with curvature of magnitude at least  $\lambda = \frac{\sum_{i=1}^n v_i([m])}{enm2^{2nm}}$  everywhere. Since the curvature of concave functions is always non-positive, and moreover the curvature of the sum of two functions is the sum of their curvatures, it suffices to show that the second term of the sum (12) has curvature of magnitude at least  $\lambda$ . We note that the curvature of  $\sum_{i,j} (1 - e^{-x_{ij}})$  is at least  $e^{-1}$  over  $x \in [0, 1]^{n \times m}$ . Therefore, the curvature of the second term of (12) is at least

$$\frac{\mu}{nm} \left( \sum_i v_i([m]) \right) e^{-1} = \lambda$$

as needed.

REMARK B.4. In this section, we sacrificed  $2^{-2nm}$  in the approximation ratio in order to guarantee expected polynomial runtime of our algorithm even when convex program (7) is not well-conditioned. This loss can be recovered to get a clean  $1 - 1/e$  approximation as follows. Given our  $(1 - 1/e - 2^{-2nm})$ -approximate MIDR algorithm  $\mathcal{A}$ , construct the following algorithm  $\mathcal{A}'$ : Given an instance of combinatorial auctions,  $\mathcal{A}'$  runs  $\mathcal{A}$  on the instance with probability  $1 - e^{-2^{-2nm}}$ , and with the remaining probability solves the instance optimally in exponential time  $O(2^{2nm})$ . It was shown in [9] that a random composition of MIDR mechanisms is MIDR, therefore  $\mathcal{A}'$  is MIDR. The expected runtime of  $\mathcal{A}'$  is bounded by the expected runtime of  $\mathcal{A}$  plus  $e^{-2^{-2nm}} \cdot O(2^{2nm}) = O(1)$ . Finally, the expected approximation of  $\mathcal{A}'$  is the weighted average of the approximation ratio of  $\mathcal{A}$  and the optimal approximation ratio 1, and is at least  $(1 - e^{-2^{-2nm}})(1 - 1/e - 2^{-2nm}) + e^{-2^{-2nm}} \geq 1 - 1/e$ .

## C. ADDITIONAL PRELIMINARIES

### C.1 Matroid Theory

In this section, we review some basics of matroid theory. For a more comprehensive reference, see [21].

A *matroid*  $M$  is a pair  $(\mathcal{X}, \mathcal{I})$ , where  $\mathcal{X}$  is a finite *ground set*, and  $\mathcal{I}$  is a non-empty family of subsets of  $\mathcal{X}$  – often referred to as the *independent sets* of the matroid – satisfying the following two properties. (1) *Downward closure*: If  $S$  belongs to  $\mathcal{I}$ , then so do all subsets of  $S$ . (2) *The Exchange Property*: Whenever  $T, S \in \mathcal{I}$  with  $|T| < |S|$ , there is some  $x \in S \setminus T$  such that  $T \cup \{x\} \in \mathcal{I}$ .

We associate with matroid  $M$  a set function  $rank_M : 2^{\mathcal{X}} \rightarrow \mathbb{N}$ , known as the *rank function* of  $M$ , defined as follows:  $rank_M(A) = \max_{S \in \mathcal{I}} |S \cap A|$ . Equivalently, the rank of set  $A$  in matroid  $M$  is the maximum size of an independent set contained in  $A$ . A set function  $f$  on a ground set  $\mathcal{X}$  is a *matroid rank function* if there exists a matroid  $M$  on the same ground set such that  $f = rank_M$ . It is well-known that matroid rank functions are monotone, normalized, and submodular.

### C.2 Convex Optimization

In this section, we distill some basics of convex optimization. For more details, see [2].

DEFINITION C.1. A *maximization problem* is given by a set  $\Pi$  of instances  $(\mathcal{P}, c)$ , where  $\mathcal{P}$  is a subset of some euclidean space,  $c : \mathcal{P} \rightarrow \mathbb{R}$ , and the goal is to maximize  $c(x)$  over  $x \in \mathcal{P}$ . We say  $\Pi$  is a *convex maximization problem* if for every  $(\mathcal{P}, c) \in \Pi$ ,  $\mathcal{P}$  is a compact convex set, and  $c : \mathcal{P} \rightarrow \mathbb{R}$  is concave. If  $c : \mathcal{P} \rightarrow \mathbb{R}^+$  for every instance of  $\Pi$ , we say  $\Pi$  is *non-negative*.

DEFINITION C.2. We say a *non-negative maximization problem*  $\Pi$  is  $R$ -solvable in polynomial time if there is an algorithm that takes as input the representation of an instance  $\mathcal{I} = (\mathcal{P}, c) \in \Pi$  — where we use  $|\mathcal{I}|$  to denote the number of bits in the representation — and an approximation parameter  $\epsilon$ , and in time  $\text{poly}(|\mathcal{I}|, \log(1/\epsilon))$  outputs  $x \in \mathcal{P}$  such that  $c(x) \geq (1 - \epsilon) \max_{y \in \mathcal{P}} c(y)$ .

FACT C.3. Consider a non-negative convex maximization problem  $\Pi$ . If the following are satisfied, then  $\Pi$  is  $R$ -solvable in polynomial time using the ellipsoid method. We let  $\mathcal{I} =$

$(\mathcal{P}, c)$  denote an instance of  $\Pi$ , and let  $m$  denote the dimension of the ambient euclidean space.

1. *Polynomial Dimension*:  $m$  is polynomial in  $|\mathcal{I}|$ .
2. *Starting ellipsoid*: There is an algorithm that computes, in time  $\text{poly}(|\mathcal{I}|)$ , a point  $c \in \mathbb{R}^m$ , a matrix  $A \in \mathbb{R}^{m \times m}$ , and a number  $\mathcal{V} \in \mathbb{R}$  such that the following hold. We use  $E(c, A)$  to denote the ellipsoid given by center  $c$  and linear transformation  $A$ .
  - (a)  $E(c, A) \supseteq \mathcal{P}$
  - (b)  $\mathcal{V} \leq \text{volume}(\mathcal{P})$
  - (c)  $\frac{\text{volume}(E(c, A))}{\mathcal{V}} \leq 2^{\text{poly}(|\mathcal{I}|)}$
3. *Separation oracle for  $\mathcal{P}$* : There is an algorithm that takes takes input  $\mathcal{I}$  and  $x \in \mathbb{R}^m$ , and in time  $\text{poly}(|\mathcal{I}|, |x|)$  where  $|x|$  denotes the size of the representation of  $x$ , outputs “yes” if  $x \in \mathcal{P}$ , otherwise outputs  $h \in \mathbb{R}^m$  such that  $h^T x < h^T y$  for every  $y \in \mathcal{P}$ .
4. *First order oracle for  $c$* : There is an algorithm that takes input  $\mathcal{I}$  and  $x \in \mathbb{R}^m$ , and in time  $\text{poly}(|\mathcal{I}|, |x|)$  outputs  $c(x) \in \mathbb{R}$  and  $\nabla c(x) \in \mathbb{R}^m$ .

## D. OMITTED PROOFS

PROOF SKETCH OF LEMMA 4.2. Let  $S_1, \dots, S_n$  be an allocation, and let  $x$  be an integer point of (7) corresponding to  $S_1, \dots, S_n$ . Let  $(S'_1, \dots, S'_n) \sim r_{\text{poiss}}(x)$ . It suffices to show that  $\mathbb{E}[\sum_i v_i(S'_i)] \geq (1 - 1/e) \cdot \sum_i v_i(S_i)$ .

By definition of the poisson rounding scheme,  $S'_i$  includes each  $j \in S_i$  independently with probability  $1 - 1/e$ . Submodularity implies that  $\mathbb{E}[v_i(S'_i)] \geq (1 - 1/e) \cdot v_i(S_i)$  – this is well-known, and we omit the details from this extended abstract. This completes the proof.  $\square$

LEMMA D.1. Let  $\mathcal{A}$  be an MIDR allocation rule for combinatorial auctions, and let  $v_1, \dots, v_n$  be input valuations. Assume black-box access to  $\mathcal{A}$ , and value oracle access to  $\{v_i\}_{i=1}^n$ . We can compute, with  $\text{poly}(n)$  over-head in runtime, payments  $p_1, \dots, p_n$  such that  $\mathbb{E}[p_i]$  equals the VCG payment of player  $i$  for MIDR allocation rule  $\mathcal{A}$  on input  $v_1, \dots, v_n$ .

PROOF. Without loss of generality, it suffices to show how to compute  $p_1$ . Let  $\mathbf{0} : 2^{[m]} \rightarrow \mathbb{R}$  be the valuation evaluating to 0 at each bundle. Recall (see e.g. [20]) that the VCG payment of player 1 is equal to

$$\mathbb{E}_{T \sim \mathcal{A}(\mathbf{0}, v_2, \dots, v_n)} \left[ \sum_{i=2}^n v_i(T_i) \right] - \mathbb{E}_{S \sim \mathcal{A}(v_1, \dots, v_n)} \left[ \sum_{i=2}^n v_i(S_i) \right]. \quad (13)$$

Let  $(S_1, \dots, S_n)$  be a sample from  $\mathcal{A}(v_1, \dots, v_n)$ , and let  $(T_1, \dots, T_n)$  be a sample from  $\mathcal{A}(\mathbf{0}, v_2, \dots, v_n)$ . Let  $p_1 = \sum_{i=2}^n v_i(T_i) - \sum_{i=2}^n v_i(S_i)$ . Using linearity of expectations, it is easy to see that the expectation of  $p_1$  is equal to the expression in (13). This completes the proof.  $\square$

REMARK D.2. We note that the mechanism resulting from Lemma D.1 is individually rational in expectation, and each payment is non-negative in expectation. We leave open the question of whether it is possible to enforce individual rationality and non-negative payments *ex-post*.