SIAM J. COMPUT.
Vol. 43, No. 1, pp. 312–336

BLACK-BOX RANDOMIZED REDUCTIONS IN ALGORITHMIC MECHANISM DESIGN*

SHADDIN DUGHMI† AND TIM ROUGHGARDEN‡

Abstract. We give the first black-box reduction from approximation algorithms to truthful approximation mechanisms for a non-trivial class of multi-parameter problems. Specifically, we prove that every welfare-maximization problem that admits a fully polynomial-time approximation scheme (FPTAS) and can be encoded as a packing problem also admits a truthful-in-expectation randomized mechanism that is an FPTAS. Our reduction makes novel use of smoothed analysis by employing small perturbations as a tool in algorithmic mechanism design. We develop a "duality" between linear perturbations of the objective function of an optimization problem and of its feasible set, and we use the "primal" and "dual" viewpoints to prove the running time bound and the truthfulness guarantee, respectively, for our mechanism.

Key words. algorithmic mechanism design, smoothed analysis, black-box reductions

AMS subject classifications. 68Q25, 91A99

DOI. 10.1137/110843654

1. Introduction.

Algorithmic mechanism design studies optimization problems where the underlying data—such as a value of a good or a cost of performing a task—is a priori unknown to the algorithm designer and must be elicited from self-interested participants (e.g., via a bid). The high-level goal of mechanism design is to design a protocol, or "mechanism," that interacts with participants so that self-interested behavior yields a desirable outcome. *Algorithmic* mechanism design adopts computational tractability as an equally important requirement.

An important research agenda, suggested roughly twelve years ago [23], is to understand rigorously what can and cannot be efficiently computed when the problem data is held by selfish agents, thereby reconciling strategic concerns with the computational requirements customary in computer science. The central question in the field is

To what extent is "incentive-compatible" efficient computation fundamentally less powerful than "classical" efficient computation?

This question remains poorly understood, despite some recent positive results for single-parameter problems[1] and negative results for deterministic mechanisms (discussed further below). A starry-eyed mechanism designer might hope for the best-possible answer:

*Received by the editors August 8, 2011; accepted for publication (in revised form) August 5, 2013; published electronically February 20, 2014. A preliminary version of this paper appeared in *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, October 2010.

http://www.siam.org/journals/sicomp/43-1/84365.html

†Department of Computer Science, University of Southern California, Los Angeles, CA 90089 (shaddin@usc.edu). This work was done while the author was a student at Stanford University and was supported by NSF grants CCF-0448664 and CCF-1016885 and a Siebel Foundation Scholarship.

‡Department of Computer Science, Stanford University, Stanford, CA 94305 (tim@cs.stanford.edu). The work of this author was supported in part by NSF grants CCF-0448664 and CCF-1016885, an ONR Young Investigator Award, an ONR PECASE Award, an AFOSR MURI grant, and an Alfred P. Sloan Fellowship.

[1] Informally, a mechanism design problem is *single-parameter* if every participant's utility function can be described naturally using a single real number, and is *multi-parameter* otherwise.

Not at all: If an optimization problem Π *admits a polynomial-time* α*-approximation algorithm* $\mathcal{A}$*, then it admits a polynomial-time* α*-approximate incentive-compatible mechanism.*

Since such a result makes no hypotheses about the algorithm $\mathcal{A}$ beyond those on its running time and approximation factor, it would presumably be proved via a "black-box reduction"—a generic method that invokes $\mathcal{A}$ at most polynomially many times and restores incentive compatibility without degrading $\mathcal{A}$'s approximation factor.

The primary contribution of this paper is the first such black-box reduction for a non-trivial class of multi-parameter problems.

In this paper, by "incentive compatible" we mean a (possibly randomized) mechanism such that every participant 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. Our main result can be summarized as follows.

MAIN RESULT (INFORMAL). *If a welfare-maximization packing problem* Π *admits a fully polynomial-time approximation scheme (FPTAS),*[2] *then it admits a truthful-in-expectation randomized mechanism that is an FPTAS.*

Thus the requirement of (randomized) incentive compatibility *imposes no loss in performance* in packing problems that admit an FPTAS. This result suggests the intriguing possibility of general black-box (randomized) reductions in algorithmic mechanism design.

1.1. Executive summary of results and techniques. We follow the most general approach known for designing (randomized) truthful multi-parameter mechanisms, via *maximal-in-distributional range (MIDR)* algorithms [7]. An MIDR algorithm fixes a set of distributions over feasible solutions—the *distributional range*—independently of the reported player utilities, and outputs a random sample from the distribution that maximizes expected welfare. These algorithms are randomized analogues of *maximal-in-range* algorithms (see, e.g., [23, 9]). Since the Vickrey–Clarke–Groves (VCG) payment scheme renders an MIDR algorithm truthful in expectation, we can focus on the purely algorithmic problem of designing an MIDR FPTAS.[3]

Our primary and most sweeping result concerns *binary packing problems of polynomial dimension*, instances of which are described by a feasible set $\mathcal{S} \subseteq \{0,1\}^d$ and an objective function $v \in \mathbb{R}^d_+$, where d is polynomial in the description of $\mathcal{S}$ and $\mathcal{S}$ is downward-closed (i.e., if $x \in \mathcal{S}$ and $y \leq x$ componentwise, then $y \in \mathcal{S}$). The goal is to maximize $v^T x$ over $x \in \mathcal{S}$. We consider problems where the objective function v is the sum $\sum_i u_i$ of several players' utility functions—$v^T x$ is then the *social welfare* of outcome x. (See sections 4 and 5 for several concrete examples.) Consider such a problem Π that admits an FPTAS, and hence—via a recent result of Röglin and Teng [25]—admits an exact algorithm $\mathcal{A}$ with polynomial smoothed complexity. (See

[2]Recall that an FPTAS for a maximization problem takes as input an instance and an approximation parameter ϵ and returns a feasible solution with objective function value at least $1-\epsilon$ times that of an optimal solution, in time polynomial in the size of the instance and in $1/\epsilon$. For randomized algorithms, the running time bound and approximation guarantee should hold, for every input, in expectation over the random coin flips of the algorithm.

[3]The reader may be tempted to assume that a (non-MIDR) FPTAS can be turned into an approximately truthful mechanism using the VCG payment scheme, implying that the results of this paper are of purely theoretical interest. This is not the case: a simple exercise shows that, in general, an FPTAS can fail to be approximately truthful (in the multiplicative sense) when combined with any payment scheme.

section 2 for precise definitions.)

As a naive starting point, suppose we apply a perturbation to a given instance of Π and then invoke the smoothed polynomial-time algorithm $\mathcal{A}$ to compute an optimal solution to the perturbed instance. The good news is that this solution will be near-optimal for the unperturbed instance provided the perturbation is not too large. The twofold bad news is that an algorithm with smoothed polynomial running time has polynomial expected running time only when the magnitude of perturbations is commensurate with that of the input numbers (to within a polynomial factor, say); and, moreover, exact optimization using perturbed valuations does not generally yield a truthful mechanism. On the first point, simultaneously learning the scale of players' utility functions and using this knowledge to compute an outcome seems incompatible with the design of truthful mechanisms, particularly for multi-parameter problems where essentially only minor variations on the VCG mechanism are known to be truthful. Is there a way to truthfully apply perturbations of the necessary magnitude? Since we use perturbations only as an algorithmic tool internal to our algorithm, we bear no burden of ensuring that the perturbations are "natural" in any sense (unlike in traditional smoothed analysis).

We provide an affirmative answer to the above question by developing a simple "duality theory" for perturbations of the following form: for a random $d \times d$ matrix P and a given objective function v, the perturbed objective function is defined as Pv. We observe that exact maximization of the perturbed objective function Pv over the feasible solutions of an instance is equivalent to exact maximization of the true objective function v over a set of "perturbed solutions" with the "adjoint" perturbation matrix P^T. When P satisfies certain conditions, each such perturbed solution can be expressed as a probability distribution over solutions. In this case, the "adjoint problem" can be solved truthfully via an MIDR algorithm. Moreover, a valuation-independent perturbation of the feasible solutions is necessarily "scale free," and we show that if it is designed appropriately, it dualizes to a perturbation of the valuations at the correct scale. Thus the "dual perspective" and the use of perturbed solutions allow us to argue truthfulness for perturbation schemes that seem, at first blush, fundamentally incompatible with truthful mechanisms. Blending these ideas together, we design a perturbation scheme that, in effect, learns the scale of the objective function v and applies perturbations of the appropriate magnitude, thereby obtaining simultaneously expected polynomial running time, an approximation factor of $(1 - \epsilon)$ for arbitrary $\epsilon > 0$, and an MIDR (and hence truthful-in-expectation) implementation.

We also extend our main result in various ways, including to binary covering problems in section 5.1; to non-packing binary maximization problems in section 5.2; and to the multi-unit auctions problem with exponential dimension in section 5.4, thereby recovering and simplifying the main result in [7].

1.2. Comparison to previous work. Our approach, based on perturbing instances of a mechanism design problem, is inspired by the work of Dobzinski and Dughmi [7], who design a truthful-in-expectation FPTAS for the problem of *multi-unit auctions* (see section 5.4 for a definition). Their approach places different and carefully chosen weights on the feasible allocations of multi-unit auctions, so that optimization over the range of weighted allocations is possible in polynomial time. All of the weights lie between $1 - \epsilon$ and 1 and can be interpreted as a probability of not canceling the corresponding allocation. Weighted allocations thus correspond to a distributional range, and optimizing over them can be implemented as part of a truthful-in-expectation mechanism. Moreover, the choice of weights guarantees that the resulting mechanism is a $(1 - \epsilon)$-approximation. The perturbations we employ generalize these weights. Our main result does not directly imply that in [7], but an extension does (section 5.4).[4]

There are three known black-box reductions from approximation algorithms to truthful approximation mechanisms for *single-parameter* mechanism design problems, where outcomes can be encoded as vectors in $\mathbb{R}^n$ (where n is the number of players) and the utility of a player i for an outcome x is $u_i x_i$, where u_i is a parameter privately known to i (the value per allocation unit). The space of truthful mechanisms for single-parameter problems is well understood and reasonably forgiving: an approximation algorithm can be used in a truthful mechanism if and only if it is monotone, meaning that the computed allocation x_i for player i is non-decreasing in the reported utility u_i (holding other players' reported utilities fixed). See [18] for precise definitions and many examples of monotone approximation algorithms. The first black-box reduction is due to Briest, Krysta, and Vöcking [4], who proved that every single-parameter binary optimization problem with polynomial dimension that admits an FPTAS also admits a truthful mechanism that is an FPTAS. Their black-box reduction is also deterministic. Second, Babaioff, Lavi, and Pavlov [1] exhibit a black-box reduction that converts an approximation algorithm for a single-parameter problem to a truthful mechanism. However, their reduction degrades the approximation ratio by a super-constant factor. Finally, Hartline and Lucier [14] consider the weaker goal of implementation in Bayes–Nash equilibria—as opposed to in dominant strategies, the notion considered here and in most of the algorithmic mechanism design literature—and show that for *every* single-parameter welfare maximization problem, every non-monotone approximation algorithm can be made monotone without degrading the expected approximation factor. All three of these black-box reductions rely heavily on the richness of the monotone algorithm design space and do not admit obvious extensions to multi-parameter problems.[5]

For multi-parameter problems, the result of Lavi and Swamy [20] is a type of black-box reduction. They show how to convert certain approximation algorithms to truthful-in-expectation mechanisms without degrading the approximation ratio. However, their result imposes non-trivial requirements on the approximation algorithm, and for many problems it is not clear if there are near-optimal approximation algorithms that meet these extra requirements.

On the negative side, there is no general and lossless black-box reduction from approximation algorithms to *deterministic* truthful approximation mechanisms for multi-parameter problems. This fact was first established by Lavi, Mu'alem, and Nisan [19], and Papadimitriou, Schapira, and Singer [24] gave a quantitatively much stronger version of this lower bound. Additional evidence of the difficulty of multi-parameter mechanism design was provided in [9] and [5] in the context of combinatorial auctions. These negative results do not apply to randomized mechanisms, however, and Dobzinski and Dughmi [7] showed that, for a variant of multi-unit auctions, truthful-in-expectation mechanisms are strictly more powerful than deter-

[4]Very recently, Vöcking [28] showed that a perturbation approach, implemented differently, yields a universally truthful FPTAS for multi-unit auctions.

[5]For example, the black-box reduction in [4] uses a simple truncation trick that preserves monotonicity but violates the weak monotonicity condition needed for truthfulness in multi-parameter problems; it also uses a monotonicity-preserving MAX operator to effectively learn the scale of the valuations, which again appears possible only in a single-parameter context. Very recently, the third reduction [14] has been extended to multi-parameter problems [2, 13].

ministic ones.[6]

Finally, we know of only one previous application of smoothed analysis techniques to the design of new algorithms: Kelner and Spielman [17] used an iterative perturbation approach to design a randomized simplex-type algorithm that has (weakly) polynomial expected running time.

2. Preliminaries.

2.1. Binary packing problems.

An instance of a *binary maximization problem* Π is given by a *feasible set* $\mathcal{S}$ encoded—perhaps implicitly—as vectors in $\{0,1\}^d$, as well as a non-negative vector $v \in \mathbb{R}^d_+$ of coefficients. The goal is to compute a feasible solution $x \in \mathcal{S}$ that maximizes the linear objective $v^T x$. Many natural maximization problems are *packing problems*, meaning that if x belongs to the feasible set $\mathcal{S}$ and $y_i \le x_i$ for all i, then $y \in \mathcal{S}$ as well. (Binary covering problems can be defined analogously; see section 5.1.)

We are mainly interested in *welfare-maximization binary packing problems*, where the objective function $v^T x$ is the welfare of self-interested players with private utility functions. Consider a feasible set $\mathcal{S} \subseteq \{0,1\}^d$ and n players, where player i has utility $\sum_{j=1}^d u_{ij} x_j$ for each $x \in \mathcal{S}$. The corresponding welfare-maximization problem—computing the outcome x that maximizes the sum of players' utilities—is then the binary maximization problem with $v_j = \sum_{i=1}^n u_{ij}$ for each $j = 1, 2, \ldots, d$. We next give a simple example to make these definitions concrete for the reader; see sections 4 and 5 for several more examples.

Example 2.1 (knapsack public projects). In the *knapsack public projects* problem, there are m projects and n players. Each project j has a publicly known cost s_j, and the feasible sets correspond to subsets of projects that have total cost at most a publicly known budget C. Each player i has a private utility u_{ij} for each project j. Welfare maximization knapsack public project instances are a binary packing problem: the feasible set is naturally encoded as the vectors x in $\{0,1\}^m$ with $\sum_j s_j x_j \le C$, and the coefficient v_j is defined as the total utility $\sum_i u_{ij}$ to all players of selecting the project j.

The binary packing problem in Example 2.1 has *polynomial dimension*, meaning that the number d of decision variables is polynomial in the size of the description of the feasible set. Our most sweeping results (section 4) are for problems with polynomial dimension, but our techniques also extend to some interesting problems with exponential dimension—see section 5.4.

2.2. Mechanism design basics.

An instance of a mechanism design problem is given by a feasible set $\mathcal{S}$ and utility functions $u_1, \ldots, u_n$, where $u_i : \mathcal{S} \to \mathbb{R}$ is the utility function of player i. We consider direct-revelation mechanisms for mechanism design problems. Such a mechanism comprises an *allocation rule* $\mathcal{A}$, which is a function from (hopefully truthfully) reported player utility functions $u_1, \ldots, u_n$ to an outcome $x \in \mathcal{S}$, and a *payment rule* p, which is a function from reported utility 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 utility

[6]Very recently, Dobzinski [6] and Dughmi and Vondrák [10] proved strong lower bounds for randomized mechanisms for certain combinatorial auction problems, in particular showing that there is no general and lossless black-box reduction from approximation algorithms to randomized truthful approximation mechanisms.

function, meaning that

$$\mathbf{E}[u_i(\mathcal{A}(u)) - p_i(u)] \geq \mathbf{E}[u_i(\mathcal{A}(u'_i, u_{-i})) - p_i(u'_i, u_{-i})] \tag{1}$$

for every player i, (true) utility function u_i, (reported) utility function u'_i, and (reported) utility functions u_{-i} of the other players. The expectation in (1) is over the coin flips of the mechanism.

We require that our mechanisms be *individually rational in expectation*—that each player i's expected payoff $\mathbf{E}[u_i(\mathcal{A}(u)) - p_i(u)]$ when reporting his true utility is non-negative. For mechanism design problems where player utilities are always non-negative, as in the welfare-maximization problems we consider for our main result, we also require that each player i's expected payment $\mathbf{E}[p_i(u)]$ is non-negative. The mechanisms that we design in this paper can be modified to obtain individual rationality and non-negative payments ex post—that is, for every flip of the mechanism's coins—as described in section 5.3.

We design mechanisms that can be thought of as randomized variations on the classical VCG mechanism. Recall that the *VCG mechanism* is defined by the (generally intractable) allocation rule that selects the welfare-maximizing outcome with respect to the reported utility 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., [22].

Now let $dist(\mathcal{S})$ denote the probability distributions over a feasible set $\mathcal{S}$, and let $\mathcal{R} \subseteq dist(\mathcal{S})$ be a compact subset of them. The corresponding *maximal in distribution range* (MIDR) mechanism has the following (randomized) allocation rule: given reported utility functions $u_1, \ldots, u_n$, return an outcome that is sampled randomly from a distribution $D^* \in \mathcal{R}$ that maximizes the expected welfare $\mathbf{E}_{x\sim D}[\sum_{i,j} u_{ij}x_j]$ over all distributions $D \in \mathcal{R}$. Analogous to the VCG mechanism, there is a deterministic payment rule p^{vcg} that can be coupled with this allocation rule to yield a truthful-in-expectation mechanism:

$$p_i^{vcg}(u) = \mathbf{E}\left[\sum_{i'\neq i} u_{i'} \cdot \mathcal{A}(u_{-i}, 0) - \sum_{i'\neq i} u_{i'} \cdot \mathcal{A}(u)\right]. \tag{2}$$

It is not always possible to compute these VCG payments efficiently using only black-box access to $\mathcal{A}$. Fortunately, any (randomized) payment scheme p satisfying $\mathbf{E}[p_i(u)] = p_i^{vcg}(u)$ also guarantees truthfulness in expectation. For every allocation rule, there is an efficient such payment scheme.

PROPOSITION 2.2. *Let $\mathcal{A}$ be an MIDR allocation rule for a welfare-maximization mechanism design problem, and let p^{vcg} be the VCG payments for $\mathcal{A}$ as in* (2)*. There is a randomized payment scheme p with $\mathbf{E}[p_i(u)] = p_i^{vcg}(u)$ that runs in polynomial time given black-box access to $\mathcal{A}$. The resulting mechanism $(\mathcal{A}, p)$ is truthful in expectation and individually rational in expectation, and the payments are non-negative in expectation.*

Proof. Given $\mathcal{A}$ as a black box and u, we sample the random variable $p_i(u)$ as follows: Sample $x \sim \mathcal{A}(u)$ and $x_{-i} \sim \mathcal{A}(u_{-i}, 0)$, and let $p_i(u) = \sum_{i'\neq i} u_{i'} \cdot x_{-i} - \sum_{i'\neq i} u_{i'} \cdot x$. To complete the proof, we observe that $\mathbf{E}[p_i(u)] = p_i^{vcg}(u)$ for each i and u. ∎

We also need the fact that probability distributions over MIDR allocation rules are again MIDR allocation rules.

LEMMA 2.3. *An allocation rule that chooses an MIDR allocation rule randomly from an arbitrary distribution over such rules is also an MIDR allocation rule.*

Proof. We fix a feasible set $\mathcal{S}$ and consider an allocation rule $\mathcal{A}$ that runs the MIDR allocation rule $\mathcal{A}_k$ with probability p_k. We use $\mathcal{R}_k$ to denote the range of $\mathcal{A}_k$. We let D_k^v be the distribution over outcomes sampled from $\mathcal{A}_k$ given the valuations v. (As usual, v_j denotes $\sum_i u_{ij}$, where u_i is the private utility function of player i.) By definition, $\mathcal{D}_k^v \in \operatorname{argmax}_{D\in\mathcal{R}_k}\{\mathbf{E}_{x\sim D}[v^T x]\}$. Now, the induced distribution over outcomes in the allocation rule $\mathcal{A}$ for v can be written as $D^v = \sum_k p_k D_k^v$. Similarly, the range of $\mathcal{A}$ is a subset of

$$\mathcal{R} = \left\{ \sum_k p_k D_k : D_k \in \mathcal{R}_k \right\}.$$

Since D_k^v maximizes welfare over all elements of $\mathcal{R}_k$ for every v and k, D^v maximizes expected welfare over $\mathcal{R}$—and hence also over the range of $\mathcal{A}$—for every v. ☐

2.3. Smoothed complexity basics. Smoothed complexity was defined by Spielman and Teng [27]; our formalism is similar to that in Beier and Vöcking [3] and Röglin and Teng [25]. A *perturbed instance* of a binary packing problem Π consists of a fixed feasible set $\mathcal{S} \subseteq \{0,1\}^d$ and d random variables $v_1, \dots, v_d$, where each v_i is drawn independently from a distribution with support in $[0, v_{\max}]$ and a density function that is bounded above everywhere by $\phi/v_{\max}$. The parameter ϕ measures the maximum concentration of the distributions of the v_i's. We say that an algorithm $\mathcal{A}$ for a binary packing problem Π runs in *smoothed polynomial time* if its expected running time is polynomial in the description length of $\mathcal{S}$ and ϕ for every perturbed instance.

Our work relies on the fact that every FPTAS for a binary optimization problem with polynomial dimension can be converted into an algorithm that runs in smoothed polynomial time. This is a special case of a result of Röglin and Teng [25], who strengthen a result of Beier and Vöcking [3].

PROPOSITION 2.4 (see [3, 25]). *For every FPTAS $\mathcal{F}$ for a binary maximization problem Π of polynomial dimension, there is an exact algorithm $\mathcal{A}^{\mathcal{F}}$ for Π that runs in smoothed polynomial time.*

Moreover, the quite natural algorithm $\mathcal{A}^{\mathcal{F}}$ in Proposition 2.4 treats $\mathcal{F}$ as an "oracle" or "black box," meaning that its behavior depends only on the outputs of $\mathcal{F}$ and not on the actual description of $\mathcal{F}$.[7]

3. Perturbation schemes that yield truthful FPTASs.

3.1. Perturbation schemes. A *perturbation scheme* for a binary packing problem Π is a randomized algorithm Ψ that takes as input an instance $(\mathcal{S}, v)$ of Π and an approximation parameter ϵ and outputs another objective function $\widehat{v} = \Psi(v, \mathcal{S}, \epsilon)$ of the same dimension as v. Such a scheme is *approximation preserving* if for every approximation parameter $\epsilon > 0$, instance $(\mathcal{S}, v)$ of Π, and outcome $x \in \mathcal{S}$, the absolute difference between the (expected) perturbed objective value of x and its unperturbed objective value is at most an ϵ fraction of the value of the optimum solution; formally, $|\mathbf{E}[\widehat{v}^T x] - v^T x| \le \epsilon \max_{y\in\mathcal{S}} v^T y$, where the expectation is over the random coin flips of the perturbation scheme.

[7]The results in [3, 25] are stated as conversions from randomized pseudopolynomial-time algorithms to smoothed polynomial-time algorithms. Proposition 2.4 follows since every FPTAS for a binary optimization problem of polynomial dimension can be converted easily to a pseudopolynomial-time exact algorithm in a black-box manner.

3.2. An overly simplistic approach. Suppose we design an exact algorithm $\mathcal{A}$ and an approximation-preserving perturbation scheme Ψ for a binary packing problem Π such that, for every instance $(\mathcal{S}, v)$ and $\epsilon > 0$, algorithm $\mathcal{A}$ has expected running time polynomial in the instance size and $1/\epsilon$ when the instance is perturbed by Ψ. Then, we immediately get an FPTAS for Π: given an instance of Π and ϵ, use the scheme Ψ to perturb the instance and the algorithm $\mathcal{A}$ to efficiently solve the perturbed instance. Since Ψ is approximation preserving, this algorithm gives a $(1-2\epsilon)$-approximation (in expectation).

Can we design such a perturbation scheme so that the resulting FPTAS can be used in a truthful-in-expectation mechanism? We face two quandaries. First, the perturbations have to be at the same "scale" as the largest coefficient $v_{\max}$ of the objective function (recall section 2); but truthfulness seems to preclude explicitly learning and subsequently using this scale in a mechanism. Second, exactly optimizing a randomly perturbed objective function does not generally yield a truthful mechanism. To address both of these issues, we require another idea.

3.3. Adjoint perturbations. We now narrow the discussion to *linear* perturbation schemes, where $\Psi(\mathcal{S}, v, \epsilon) = Pv$ for a (random) matrix P whose distribution is independent of v. We next develop a "duality" for such schemes. We will need both the "primal" and "dual" viewpoints to prove the running time bound and the truthfulness guarantee, respectively, of our final mechanism.

Here is a trivial observation: for every fixed perturbation matrix P, objective function v, and feasible solution $x \in \mathcal{S}$, the value $(Pv)^T x$ of the solution x with respect to the perturbed objective Pv equals the value $v^T(P^T x)$ of the "perturbed solution" $P^T x$ with respect to the true objective v. We say that the perturbation P^T is *adjoint* to P. Taking this alternative adjoint viewpoint, solving a linearly perturbed instance $(\mathcal{S}, Pv)$ of a binary packing problem is equivalent to solving the optimization problem

$$\begin{array}{ll} \text{maximize} & v^T \widetilde{x} \\ \text{subject to} & \widetilde{x} \in P^T \mathcal{S}, \end{array} \tag{3}$$

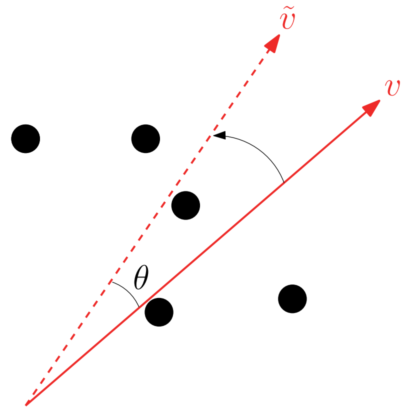
where $\widetilde{x} = P^T x$ and $P^T \mathcal{S} = \{\widetilde{x} : x \in \mathcal{S}\}$. See Figure 1 for an illustration of this relationship.

The adjoint problem (3) is meaningful when we can associate every $\widetilde{x} \in P^T \mathcal{S}$ with a probability distribution over the feasible solutions $\mathcal{S}$ that has expectation $\widetilde{x}$. This is possible if and only if $P^T \mathcal{S} \subseteq \text{convexhull}(\mathcal{S})$. Assume that we have designed P to possess this property, and for every $x \in \mathcal{S}$ let D_x be an arbitrary distribution over $\mathcal{S}$ with expectation $\widetilde{x} = P^T x$. Let $\mathcal{R} = \{D_x\}_{x \in \mathcal{S}}$ denote the corresponding distributional range. By linearity, the adjoint problem (3) is then equivalent to the problem of maximizing the expected objective function value over $\mathcal{R}$:

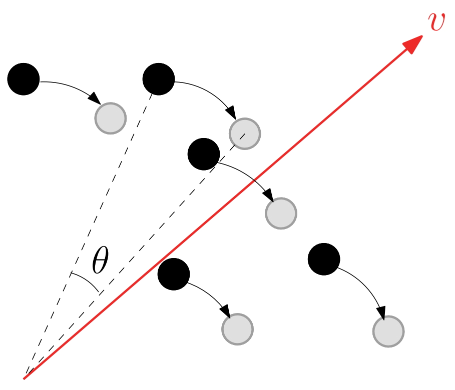
$$\begin{array}{ll} \text{maximize} & \mathbf{E}_{y \sim D_x}[v^T y] \\ \text{subject to} & D_x \in \mathcal{R}. \end{array} \tag{4}$$

The key point is that *this is precisely the type of optimization problem that can be solved—truthfully—using an MIDR allocation rule and the corresponding payment rule* (recall section 2).

3.4. Structure of the black-box reduction. The next theorem formalizes our progress so far: designing truthful-in-expectation mechanisms reduces to designing perturbation schemes that meet a number of requirements.



(a) Perturbation P rotates v by an angle θ to $\tilde{v}$.



(b) This is equivalent to rotating each feasible solution by an angle of $-\theta$.

FIG. 1. *Two views of a perturbation.*

For a linear perturbation scheme Ψ for a binary packing problem Π, we say that Ψ is *feasible* if, for every feasible set $\mathcal{S}$ of Π and $\epsilon > 0$ the perturbation matrix $P \sim \Psi(\mathcal{S}, \epsilon)$ satisfies $P^T\mathcal{S} \subseteq \text{convexhull}(\mathcal{S})$ surely. Such a scheme is *tractable* if it runs (i.e., outputs the matrix P) in time polynomial in the length of the description of $\mathcal{S}$ and if for each $x \in \mathcal{S}$ a distribution D_x with support in $\mathcal{S}$ and expectation $P^T x$ can be sampled in time polynomial in the length of the description of $\mathcal{S}$. A *FLAT* perturbation scheme is one that is feasible, linear, approximation-preserving, and tractable. The outline of our black-box reduction is displayed below as Algorithm 1.

Algorithm 1 Perturbation-based (PB) allocation rule for a binary packing problem Π.

```
Parameter: Approximation parameter $\epsilon > 0$.
Parameter: Exact algorithm $\mathcal{A}$ for $\Pi$.
Parameter: FLAT perturbation scheme $\Psi$ for $\Pi$.
Input: Instance $(\mathcal{S}, v)$.
Output: Solution $y \in \mathcal{S}$.
1: Draw $P \sim \Psi(\mathcal{S}, \epsilon)$.
2: Let $x = \mathcal{A}(\mathcal{S}, Pv)$.
3: Let $D_x$ be a distribution over $\mathcal{S}$ with expectation $P^T x$, chosen independently of $v$.
4: Return a sample $y \sim D_x$.
```

THEOREM 3.1. *For every binary packing problem Π and FLAT perturbation scheme Ψ, the corresponding perturbation-based (PB) allocation rule (Algorithm 1) satisfies the following properties:*

(a) *it is MIDR and hence defines a truthful-in-expectation mechanism when combined with suitable payments;*
(b) *for every instance of Π and $\epsilon > 0$, it outputs a feasible solution with expected objective function value at least $(1 - \epsilon)$ times the maximum possible;*
(c) *its worst-case expected running time is bounded by a polynomial plus that of the exact algorithm $\mathcal{A}$ on a perturbed instance $(\mathcal{S}, Pv)$.*

The key point of Theorem 3.1 is part (a), which guarantees truthfulness while permitting remarkable freedom in designing perturbation schemes.

Proof Theorem 3.1. First, we note that the choice of P in step 1 is independent of v by the definition of a linear scheme and that step 3 is well defined because Ψ is feasible. Part (c) follows immediately from the assumption that Ψ is tractable. Part (b) follows from the definition of an approximation-preserving scheme, the fact that $\mathcal{A}$ is an exact algorithm, and the fact that the expected value of the solution y returned by the PB allocation rule equals $\mathbf{E}_{y\sim D_x}[v^Ty] = v^T(P^Tx) = (Pv)^Tx$, which is the objective function value (with respect to the perturbed objective Pv) of the solution returned by $\mathcal{A}$.

To prove part (a), consider an instance $(\mathcal{S}, v)$ and approximation parameter ϵ. To begin, condition on the choice of P by $\Psi(S, \epsilon)$ in Step 1 of the PB allocation rule. Let D_x be the distribution over $\mathcal{S}$ with expectation P^Tx that the allocation rule chooses in Step 3 in the event that $x = \mathcal{A}(\mathcal{S}, Pv)$, and set $\mathcal{R} = \{D_x : x \in \mathcal{S}\}$. By the definition of this step, the range $\mathcal{R}$ depends only on $\mathcal{S}$ and ϵ and is independent of the valuations v. Since the allocation rule explicitly computes the solution x^* that maximizes $(Pv)^Tx$ over $x \in \mathcal{S}$ and then samples an outcome from the corresponding distribution D_{x^*}, and this x^* is the same solution that maximizes $\mathbf{E}_{y\sim D_x}[v^Ty]$ over $x \in \mathcal{S}$ (i.e., over D_x in $\mathcal{R}$), the output of the allocation rule is the same (for each v) as that of the MIDR allocation rule with distributional range $\mathcal{R}$.

We have established that for each fixed choice of P, the PB allocation rule is an MIDR rule. Since the random choice of P is independent of the valuations v, the PB allocation rule is a probability distribution over MIDR rules. By Lemma 2.3, it is an MIDR allocation rule. ∎

4. The main result.

4.1. The random singleton scheme.

We now describe a FLAT perturbation scheme that leads to our main result: every binary packing problem with polynomial dimension that admits an FPTAS also admits a truthful-in-expectation mechanism that is an FPTAS.

We call our FLAT scheme the *random singleton* (RS) perturbation scheme, and we first describe it via its adjoint. Let $(\mathcal{S}, v)$ be an instance of a binary packing problem Π with polynomial dimension, with $\mathcal{S} \subseteq \{0,1\}^d$. Since Π is a packing problem, the all-zero vector lies in $\mathcal{S}$, and we can assume without loss of generality that each basis vector $e_1, \ldots, e_d$ lies in $\mathcal{S}$ (if $e_i \notin \mathcal{S}$, then we can ignore coordinate i). Given $x \in \mathcal{S}$ and a parameter $\epsilon > 0$, we consider the following randomized algorithm:

(1) for each $i = 1, 2, \ldots, d$, draw δ_i uniformly from the interval $[0, \epsilon/d]$;
(2) output a random solution $y \in \mathcal{S}$ according to the following distribution: output the given solution x with probability $1 - \epsilon$, the "singleton" e_j with probability $(\sum_{i=1}^d \delta_i x_i)/d$ (for each $j = 1, \ldots, d$); and the all-zero solution with the remaining probability.

The motivation of the random choices in the first step is to ensure that the distribution defined by the perturbation is diffuse enough to permit algorithms with polynomial smoothed complexity (cf. the parameter ϕ in section 2). The motivation of the random choices in the second step is to reward a solution $x \in \mathcal{S}$ with a "bonus" of a random singleton with probability δ_i for each coordinate i with $x_i = 1$. Since there exists a singleton e_j with value v_j that is at least a $1/d$ fraction of the optimal value $\max_{y\in\mathcal{S}} v^Ty$, these bonuses effectively ensure that the perturbations occur at the correct "scale."

We now make this vague intuition precise. After conditioning on the random choices in step (1), the expectation $\widetilde{x}$ of the distribution D_x over solutions $\mathcal{S}$ defined by step (2) can be expressed via the adjoint perturbation P^T given by

$$\widetilde{x} = P^T x = (1-\epsilon)x + \left(\sum_{i=1}^{d} \delta_i x_i\right)\left(\sum_{j=1}^{d} \frac{e_j}{d}\right).$$

Let δ denote the d-vector of the δ_i's. Since P^T can be written as $(1-\epsilon)I + \frac{1}{d}\vec{\mathbf{1}}\delta^T$, dualizing gives the following formal definition of the RS perturbation scheme for Π, given $(\mathcal{S}, v)$ and ϵ and conditioned on the random choices of the δ_i's:

$$P = (1-\epsilon)I + \frac{\delta\vec{\mathbf{1}}^T}{d}.$$

This corresponds to the perturbation

$$v_i \mapsto (1-\epsilon)v_i + \frac{\delta_i}{d}\sum_{j=1}^{d} v_j \tag{5}$$

for each coefficient i. This perturbation depends on the v_i's and might appear unsuitable for deployment in a truthful mechanism. Its use is justified by our development of adjoint perturbations.

LEMMA 4.1. *For every binary packing problem Π of polynomial dimension, the RS perturbation scheme is FLAT.*

Proof. Since the choice of the perturbation matrix P depends only on the feasible set $\mathcal{S}$, the approximation parameter ϵ, and the (valuation-independent) choices of the δ_i's, the RS scheme is linear. It is feasible because it is defined explicitly via the adjoint P^T and the distributions D_x over solutions whose expectations agree with $P^T x$ (for each $x \in \mathcal{S}$). It is clearly tractable. Finally, we observe from (5) that for every perturbation matrix P of the scheme and feasible solution $x \in \mathcal{S}$, $(1-\epsilon)v^T x \leq (Pv)^T x \leq v^T x + \epsilon \max_i v_i$. Since both $v^T x$ and $\max_i v_i$ are no greater than the value of the optimum solution, the RS scheme is approximation preserving. ∎

4.2. Putting it all together. We are now prepared to prove our main result.

THEOREM 4.2 (main result). *Every welfare-maximization binary packing problem of polynomial dimension that admits an FPTAS also admits a truthful-in-expectation mechanism that is an FPTAS.*

Proof. Let Π be a binary packing problem of polynomial dimension and $\mathcal{F}$ be an arbitrary FPTAS for it. By Proposition 2.4, there is an exact algorithm $\mathcal{A}^{\mathcal{F}}$ for Π that runs in smoothed polynomial time in the sense of section 2. Let Ψ denote the RS perturbation scheme for Π, and instantiate the PB allocation rule with the scheme Ψ and algorithm $\mathcal{A}^{\mathcal{F}}$. Since Ψ is FLAT (Lemma 4.1), Theorem 3.1 implies that this allocation rule is MIDR, has an approximation guarantee of $1-\epsilon$ in expectation (for an arbitrary supplied parameter ϵ), and has expected running time bounded by a polynomial plus that of $\mathcal{A}^{\mathcal{F}}$ on the perturbed instance $(\mathcal{S}, Pv)$.

To analyze the expected running time of $\mathcal{A}^{\mathcal{F}}$ on $(\mathcal{S}, Pv)$, recall the perturbation formula (5). Let $v_{\max}$ denote $\max_{i=1}^{d} v_i$. Every coordinate of Pv is bounded above by $v_{\max}$ with probability 1, and these coordinates are independent random variables (since the δ_i's are independent). Since $\sum_{j=1}^{d} v_j \geq v_{\max}$ and δ_i is drawn uniformly from $[0, \epsilon/d]$, the density of the random variable $(Pv)_i$ is bounded above everywhere by $\frac{d^2}{\epsilon v_{\max}}$. Thus the concentration parameter ϕ from section 2 is bounded by d^2/ϵ. Since Π has polynomial dimension and $\mathcal{A}^{\mathcal{F}}$ has polynomial smoothed complexity, the expected running time of $\mathcal{A}^{\mathcal{F}}$ on $(\mathcal{S}, Pv)$ is polynomial in the input size and $1/\epsilon$.

To complete the proof, we recall from Proposition 2.2 that truth-telling payments for this allocation rule can be computed with only polynomial overhead in the running time. $\square$

4.3. Examples. We feel that the primary point of Theorem 4.2 is conceptual: it shows that requiring (randomized) incentive compatibility requires no sacrifice in performance for a non-trivial class of multi-parameter problems and suggests that even more general "black-box randomized reductions" might be possible. Of course, a general result like Theorem 4.2 can be instantiated for various concrete problems, and we conclude the section by listing a few examples. Numerous single-parameter examples are given in Briest, Krysta, and Vöcking [4]. Below we present some multi-parameter examples, which are beyond the reach of the results in [4].

Knapsack public projects. From a purely algorithmic perspective, the problem in Example 2.1 is equivalent to the knapsack problem and hence admits a (non-truthful) FPTAS.

Arborescent knapsack public projects. This is a generalization of the knapsack public projects problem, where additional constraints are placed on the feasible solutions $\mathcal{S} \subseteq \{0,1\}^m$. Namely, the projects $[m]$ are the ground set of a laminar[8] set system $\mathcal{L} \subseteq 2^{[m]}$, and there is a budget C_T for each $T \in \mathcal{L}$. The feasible set $\mathcal{S}$ is constrained so that $\sum_{j \in T} s_j \leq C_T$ for each $T \in \mathcal{L}$. A (non-truthful) FPTAS for this problem was given in [12].

Tree-ordered knapsack public projects. This is another generalization of the knapsack public projects problem, where precedence constraints are placed on the projects $[m]$. Namely, a directed acyclic graph G with vertices $[m]$ encodes precedence constraints, and the feasible set $\mathcal{S} \subseteq \{0,1\}^m$ is constrained so that $x_j \geq x_k$ whenever $(j,k) \in E(G)$ for every $x \in \mathcal{S}$. For the case in which G is a directed-out tree, a (non-truthful) FPTAS for this problem was given in [16]. Observe, however, that this is no longer a binary packing problem. Fortunately, our proof of Theorem 4.2 relied very little on the packing assumption, and we argue in section 5.2 that we require only $\vec{\mathbf{0}} \in \mathcal{S}$, which is certainly the case here.

Maximum job sequencing with deadlines. In this problem, m jobs are to be scheduled on a single machine. Job $j \in [m]$ has processing time p_j and deadline d_j. There are n players, and player i has private utility u_{ij} for each job j that completes before its deadline d_j. The goal is to find the welfare-maximizing subset of the jobs that can be scheduled so that each finishes before its deadline. Converting such a set of jobs to a schedule can be done via the obvious greedy algorithm. This yields a binary packing problem with a welfare objective. A (non-truthful) FPTAS for this problem was given in [26].

5. Extensions.

5.1. Binary covering problems. In this section, we consider *binary covering problems* of polynomial dimension. Such problems are defined analogously to binary packing problems, except that the feasible $\mathcal{S}$ is upward-closed, and the goal is to minimize $v^T x$ over $x \in \mathcal{S}$. We consider social-cost-minimization binary covering

[8] A set system $\mathcal{L} \subseteq 2^{[m]}$ is *laminar* if for every $T, T' \in \mathcal{L}$, either $T \cap T' = \emptyset$, $T \subseteq T'$, or $T' \subseteq T$.

problems, where $v^T x$ is the social cost of outcome x. In these problems, $v_j = \sum_{i=1}^n c_{ij}$ for each $j = 1, 2, \ldots, d$, and c_i denotes the private cost function of player i.

By reduction to Theorem 4.2, we show in Theorem 5.1 that binary covering problems admit a truthful-in-expectation "additive FPTAS." We then show in Theorem 5.3 that this is the best we can hope for by an MIDR mechanism, as no polynomial-time MIDR mechanism obtains a finite approximation for an NP-hard binary covering problem (assuming $P \neq NP$).[9]

THEOREM 5.1. *Every social-cost-minimization binary covering problem of polynomial dimension that admits an FPTAS also admits a truthful-in-expectation mechanism, parametrized by ϵ, that runs in time polynomial in the description of the instance and $\frac{1}{\epsilon}$, and outputs a solution with expected social cost at most an additive $\epsilon c_{\max}$ more than the optimum value, where $c_{\max}$ denotes the maximum cost of a feasible solution.*

Proof. Let Π be a binary covering problem of polynomial dimension d that admits an FPTAS. We now define the *complementary problem* $\overline{\Pi}$ as follows. For $x \in \{0,1\}^d$, we define its complement $\overline{x} = \vec{\mathbf{1}} - x$. For $\mathcal{S} \subseteq \{0,1\}^d$ let $\overline{\mathcal{S}} = \{\overline{x} : x \in \mathcal{S}\}$. Now let $\overline{\Pi} = \{(\overline{\mathcal{S}}, v) : (\mathcal{S}, v) \in \Pi\}$ be the problem of maximizing $v^T \overline{x}$ for $\overline{x} \in \overline{\mathcal{S}}$. It is clear that $\overline{\Pi}$ is a binary packing problem of polynomial dimension. It is easy to see that x is an optimal solution for Π if and only if $\overline{x}$ is an optimal solution to $\overline{\Pi}$. We utilize this complementarity between covering and packing problems twice—once in each direction—in the following proof: first we argue that an FPTAS for Π yields an FPTAS for $\overline{\Pi}$, and then we invoke the result in Theorem 4.2 to obtain an MIDR FPTAS for $\overline{\Pi}$, which we then show can be converted to an MIDR "additive FPTAS" for Π. Invoking Proposition 2.2 then completes the proof.

Now we argue that an FPTAS for Π can be converted to an FPTAS for $\overline{\Pi}$ as follows: for an instance $(\overline{\mathcal{S}}, v)$ of $\overline{\Pi}$ and approximation parameter ϵ, we simply invoke the FPTAS for Π on $(\mathcal{S}, v)$ with approximation parameter ϵ/d and output the complement of the returned solution. Let OPT denote the optimal objective function value of the covering problem instance $(\mathcal{S}, v)$. The cost of the solution x returned by the FPTAS for Π is within an additive error of at most $(\epsilon/d)OPT \leq (\epsilon/d) \cdot dv_{\max} = \epsilon v_{\max}$ from optimal, where $v_{\max} = \max_{j=1}^n v_j$. Since the optimal value of the complementary packing problem $\overline{\Pi}$ is—without loss of generality—at least $v_{\max}$, the value of $\overline{x}$ is at least a $(1-\epsilon)$ factor of the optimum value for instance $(\overline{\mathcal{S}}, v)$ of the packing problem $\overline{\Pi}$.

By the result in Theorem 4.2, the packing problem $\overline{\Pi}$ admits an MIDR FPTAS $\overline{\mathcal{A}}$. We will now convert $\overline{\mathcal{A}}$ to an MIDR "additive FPTAS" $\mathcal{A}$ for the covering problem Π. We fix the approximation parameter ϵ and define $\mathcal{A}$ as follows: On input $(\mathcal{S}, v)$, let $\overline{x}$ be the output of $\overline{\mathcal{A}}$ with approximation parameter ϵ/d and input $(\overline{\mathcal{S}}, v)$, and output $x = 1 - \overline{x}$.

To show that $\mathcal{A}$ is MIDR, let $\overline{R}$ be the range of $\overline{\mathcal{A}}$ when the approximation parameter is ϵ/d. Define the complementary range $\mathcal{R}$ in the obvious way: for every $\overline{D} \in \overline{\mathcal{R}}$, we let D be the distribution that simply draws $\overline{y} \sim \overline{D}$ and outputs $y = 1 - \overline{y}$. Then, we let $\mathcal{R} = \{D : \overline{D} \in \overline{\mathcal{R}}\}$. First, it is easy to see, by construction, that the range of $\mathcal{A}$ is a subset of $\mathcal{R}$. Now, fix an instance $(\mathcal{S}, v) \in \Pi$, and let $\overline{D}$ be the distribution of $\overline{\mathcal{A}}(\overline{\mathcal{S}}, v)$. By definition, $\mathcal{A}$ outputs the complementary distribution D for the same input. Since $\overline{D}$ maximizes $\mathbf{E}_{\overline{y} \sim \overline{D}}[v^T \overline{y}]$ over $\overline{\mathcal{R}}$, it must minimize $\mathbf{E}_{\overline{y} \sim \overline{D}}[v^T(\vec{\mathbf{1}} - \overline{y})]$ over $\overline{\mathcal{R}}$. This, by definition, implies that D minimizes $\mathbf{E}_{y \sim D}[v^T y]$ over $\mathcal{R}$, and $\mathcal{A}$ is MIDR

[9]This is noteworthy because MIDR mechanisms are essentially the only known general technique for designing truthful approximation mechanisms for multi-parameter problems.

with range $\mathcal{R}$.

It remains to prove the approximation guarantee of $\mathcal{A}$. Let $x \sim D$ and $\overline{x} \sim \overline{D}$ be the (random) outputs of $\mathcal{A}$ and $\overline{\mathcal{A}}$ on inputs $(\mathcal{S}, v)$ and $(\overline{\mathcal{S}}, v)$, respectively. We then derive

$$\begin{aligned}
\mathbf{E}[v^T x] &= \mathbf{E}[v^T(\vec{\mathbf{1}} - \overline{x})] \\
&= ||v||_1 - \mathbf{E}[v^T \overline{x}] \\
&\leq ||v||_1 - (1 - \epsilon/d) \max_{\overline{y} \in \overline{\mathcal{S}}} v^T \overline{y} \\
&= ||v||_1 - (1 - \epsilon/d)(||v||_1 - \min_{y \in \mathcal{S}} v^T y) \\
&\leq \min_{y \in \mathcal{S}} v^T y + \frac{\epsilon}{d} ||v||_1 \\
&\leq \min_{y \in \mathcal{S}} v^T y + \epsilon \max_j v_j.
\end{aligned}$$

Since $\max_j v_j$ is a lower bound on the maximum cost of solutions in $\mathcal{S}$, this completes the proof. $\square$

The bound in Theorem 5.1 becomes an FPTAS in the usual multi-plicative sense when we restrict our attention to instances of the problem in which the value of the optimal solution can be bounded below by an inverse polynomial fraction of $c_{\max}$. In general, however, additive loss is inevitable if we restrict ourselves to MIDR algorithms.

LEMMA 5.2. *Let* Π *be a binary minimization problem. If an MIDR algorithm* $\mathcal{A}$ *provides a finite approximation ratio for* Π*, then* $\mathcal{A}$ *is optimal.*

Proof. Assume $\mathcal{A}$ is MIDR and provides a finite approximation ratio for Π. Fix a feasible set $\mathcal{S}$ of Π, and let $\mathcal{R}$ be the corresponding distributional range of $\mathcal{A}$. We say a feasible solution $x \in \mathcal{S}$ is *minimal* if there does not exist $y \neq x$ in $\mathcal{S}$ with $y_i \leq x_i$ for all i. It is clear that for every objective $v \in \mathbb{R}_+^d$, there exists an optimal solution that is minimal. Since $\mathcal{A}$ is MIDR, it then suffices to show that $\mathcal{R}$ contains all point distributions corresponding to minimal feasible solutions.

Consider a minimal $x \in \mathcal{S}$, and let the objective function v be such that $v_i = 0$ when $x_i = 1$, and $v_i = 1$ when $x_i = 0$. By definition we have $v^T x = 0$. Moreover, since x is minimal, $v^T y > 0$ for every $y \in \mathcal{S}$ with $y \neq x$. Therefore, the only distribution over $\mathcal{S}$ providing a finite approximation ratio for v is the point distribution corresponding to x. Thus, $\mathcal{R}$ contains all point distributions of minimal feasible solutions, as needed. $\square$

Our negative result for binary covering problems follows immediately from Lemma 5.2.

THEOREM 5.3. *Let* Π *be an NP-hard binary minimization problem. No polynomial-time MIDR allocation rule provides a finite approximation ratio for* Π *unless* $P = NP$.

We note that Theorem 5.3 and its proof easily extend to the slightly more general class of *distributional affine maximizers* (see [7]) and hence to all known types of VCG-based mechanisms.

Examples. We conclude the section with a few multi-parameter problems to which Theorem 5.1, and the complementary negative result in Theorem 5.3, apply. Again, for numerous single-parameter examples see Briest, Krysta, and Vöcking [4].

Minimum job sequencing with deadlines. In this problem, m jobs are to be scheduled on a single machine. Job $j \in [m]$ has processing time p_j and deadline d_j. There are n players, and player i incurs private cost c_{ij} for each job j that completes past its deadline d_j. The goal is to find a schedule minimizing social cost. This is a binary covering problem. A (non-truthful) FPTAS for this problem was given in [11].

Constrained shortest path. We are given a graph $G = (V, E)$ and two terminals $s, t \in V$. Additionally, there is latency l_j for each $j \in E$. The mechanism is interested in selecting a path from s to t of total latency at most L. There are n players, and player i incurs private cost c_{ij} if $j \in E$ is selected. We consider a covering variant of this problem, where the mechanism may select any subgraph of G connecting s to t via a path of latency at most L, and the goal is to minimize social cost. A (non-truthful) FPTAS for this problem was given in [15].

Constrained minimum spanning tree on treewidth bounded graphs. We are given a graph $G = (V, E)$ with bounded treewidth. Additionally, there is a weight w_j for each $j \in E$. The mechanism is interested in selecting a spanning tree of G with total weight at most W. There are n players, and player i incurs private cost c_{ij} if $j \in E$ is selected. We consider the covering variant of this problem, where the mechanism may select any spanning subgraph of G containing a spanning tree of total weight at most W, and the goal is to minimize social cost. A (non-truthful) FPTAS for this problem was given in [21].

5.2. Non-packing binary maximization problems. We observe that the packing assumption of Theorem 4.2 can be relaxed. In particular, if Π is a binary maximization problem, it suffices that $\vec{\mathbf{0}} \in \mathcal{S}$ for every feasible set $\mathcal{S}$ of Π. To see this, note that the only other property of packing problems that is needed in the proof of Theorem 4.2 is that $e_j \in \mathcal{S}$ for each $j = 1, \ldots, d$. It is straightforward to modify the proof to use the following weaker assumption: For each $j = 1, \ldots, d$, there exists $y^j \in \mathcal{S}$ such that $y^j_j = 1$ (and y^j can be identified in polynomial time). Letting $y = \sum_j y^j$, we then modify Ψ as follows: the δ_i's are drawn as before, and we define $P = (1 - \epsilon)I + \frac{\delta y^T}{d}$. The proof proceeds in a similar fashion. Similarly, Theorem 5.1 extends to binary minimization problems where $\vec{\mathbf{1}} \in \mathcal{S}$ for every feasible set $\mathcal{S}$.

5.3. Stronger guarantees on payments. The payments in Proposition 2.2 are non-negative and individually rational only *in expectation.* We now explain how to achieve the stronger properties of ex post non-negativity and individual rationality—i.e., to guarantee that player i's payment and payoff are non-negative for every flip of the mechanism's coins—while maintaining that the payments can be computed in polynomial time.

Lavi and Swamy [20] observed that a payment rule guaranteeing ex post individual rationality and non-negativity exists for every MIDR mechanism for a problem with non-negative utilities. We summarize this observation for welfare-maximization binary packing problems in Proposition 5.4. This payment rule carefully couples its random choices with those of the accompanying allocation rule.

Proposition 5.4 (see [20]). *Fix a welfare-maximization binary packing problem, and let $\mathcal{A}$ be an MIDR allocation rule for this problem with distributional range $\mathcal{R}$. Let y be the random output of $\mathcal{A}$ on an arbitrary input u. The payment rule*

$$\overline{p}_i(u, y) = \frac{p_i^{vcg}(u)}{\mathbf{E}\left[u_i \cdot \mathcal{A}(u)\right]}\ u_i \cdot y, \tag{6}$$

where p_i^{vcg} is as defined in (2), *is always non-negative and results in a truthful-in-expectation and ex post individually rational mechanism when coupled with allocation rule $\mathcal{A}$.*

Proof. That $\overline{p}_i(u, y)$ is non-negative follows from the fact that the VCG payment $p_i^{vcg}(u)$ of player i is non-negative, as are his expected utility $\mathbf{E}[u_i \cdot \mathcal{A}(u)]$ and his realized utility $u_i \cdot y$. For truthfulness in expectation, we note that taking expectation over choices of y in (6) recovers the VCG payment $p_i^{vcg}(u)$. Since using VCG payments yields a truthful-in-expectation mechanism, the same is true for the payments in (6).

We still need to show individual rationality. Recall that the VCG payments in (2) are individually rational in expectation, with $p_i^{vcg}(u) \leq \mathbf{E}[u_i \cdot \mathcal{A}(u)]$. This implies that the payment $\overline{p}_i(u, y)$ in (6) is no greater than the realized utility $u_i \cdot y$, as needed for individual rationality. ∎

Efficient computation of the payments of Proposition 5.4 is not possible in general. Fortunately, the PB allocation rule has additional structure that we can exploit to efficiently compute ex post individually rational and non-negative payments. The payments will be based on those of Proposition 5.4, and we describe them next.

We recall from Theorem 3.1 that the PB allocation rule (Algorithm 1) is MIDR for every choice of the perturbation matrix P drawn from Ψ. Therefore, we need only compute the payments of (6) for a fixed choice of P; this induces a probability distribution over truthful-in-expectation mechanisms, one for each choice of P, each of which is ex post individually rational and charges non-negative payments.

Fix ϵ, and let $\mathcal{B}$ denote an instantiation of the PB allocation rule with a smoothed polynomial-time algorithm $\mathcal{A}^{\mathcal{F}}$ and a FLAT perturbation scheme Ψ. Let $\mathcal{B}^P$ denote $\mathcal{B}$ when the perturbation matrix in step 1 of Algorithm 1 is fixed to P. Let u_i be the utility vector of player i, so that $v = \sum_i u_i$. By examining (6) and (2), we observe that computing the payments of (6) reduces in polynomial time to computing $\mathbf{E}[\mathcal{B}^P(v)]$ and $\mathbf{E}[\mathcal{B}^P(v - u_i)]$ for each player i. The expected outcome $\mathbf{E}[\mathcal{B}^P(v)]$ is $P^T x$, where $x = \mathcal{A}^{\mathcal{F}}(\mathcal{S}, Pv)$ is as computed in step 2. Similarly, the expected outcome $\mathbf{E}[\mathcal{B}^P(v - u_i)]$ is $P^T \mathcal{A}^{\mathcal{F}}(\mathcal{S}, P(v - u_i))$. Since P is drawn from the tractable perturbation scheme Ψ and $\mathcal{A}^{\mathcal{F}}$ runs in smoothed polynomial time, computing $\mathcal{A}^{\mathcal{F}}(\mathcal{S}, Pv)$ and $\mathcal{A}^{\mathcal{F}}(S, P(v - u_i))$ for all players i takes expected time polynomial in the description of $\mathcal{S}$, $\frac{1}{\epsilon}$ and the number of players.

5.4. Beyond polynomial dimension: Multi-unit auctions. In this section, we extend our main result to a problem with exponential dimension: *multi-unit auctions.* This problem is one of a handful that have guided much of the research in algorithmic mechanism design over the past decade (e.g., [20, 8, 7]). The state of the art in randomized mechanisms for this problem is a truthful-in-expectation FPTAS due to Dobzinski and Dughmi [7]. This section provides a different proof of their result by extending our perturbation-based framework. We view the contribution of this section as a technical and conceptual simplification of the main result of [7]. We note, however, that the result we prove here is weaker than theirs in one sense: our mechanism runs in polynomial time in expectation, whereas theirs runs in polynomial time surely.

5.4.1. Multi-unit auctions. In a multi-unit auction, a set of m identical items must be allocated to n bidders. Each bidder i is equipped with a valuation function $v_i : [m] \to \mathbb{R}^+$, where v_i is non-decreasing with $v_i(0) = 0$. The goal is to find an allocation $(s_1, \ldots, s_n)$ of the items, where each s_i is a non-negative integer and $\sum_{i=1}^n s_i \leq m$, which maximizes the social welfare: $\sum_i v_i(s_i)$. Multi-unit auctions can be written as a binary packing problem as follows, where decision variable x_{ij}

indicates whether player i is assigned j items:

$$\begin{array}{llll} & \text{maximize} & \sum_{ij} v_i(j) \cdot x_{ij} & \\ & \text{subject to} & \sum_{ij} j \cdot x_{ij} \leq m, & \\ (7) & & \sum_j x_{ij} \leq 1 & \text{for all } i \in [n], \\ & & x_{ij} \in \{0,1\} & \text{for all } i \in [n], j \in [m]. \end{array}$$

We assume that an instance of multi-unit auctions is given by the number of items m, and the n valuation functions are each presented via a *value oracle*. Specifically, for each player i and number of items j, our algorithms may query $v_i(j)$ in constant time. Since all items are identical, the natural "length" of the description of the items is $\log m$, rather than m. Therefore, we require a computationally efficient algorithm for multi-unit auctions to run in time polynomial in n and $\log m$.

Using n and $\log m$ as the natural parameters, we observe that the binary packing problem (7) has exponential dimension, with mn variables. Therefore, the results of sections 3 and 4 do not apply directly to multi-unit auctions.

5.4.2. Outline. Application of the techniques of sections 3 and 4 to multi-unit auctions faces two main difficulties, both due to the exponential dimensionality of the problem. The first relates to our use of smoothed complexity: an FPTAS for a problem with exponential dimension does not appear to, in general, imply polynomial-smoothed complexity of the problem in the same general sense discussed in section 2.3. Our solution is to design a specific perturbation scheme and accompanying exact algorithm that are tailored to multi-unit auctions. Second, even if instances perturbed by a specific scheme can be solved in expected polynomial time, this is only useful if said perturbation can be applied to an instance efficiently. This was called *tractability* in section 3, and required both the sampling of a perturbation matrix as well as the decomposition of a perturbed solution into a convex combination of other solutions; both are evidently impossible for problems of exponential dimension, as the perturbation matrix has an exponential number of entries. We relax tractability, using the principle of deferred decisions, and show that the relaxed definition suffices for multi-unit auctions. We now outline these challenges and our solution approach in more detail.

Perturbations for smoothed polynomial running time with exponential dimension. The PB allocation rule requires the existence of an exact algorithm for multi-unit auctions that runs in expected polynomial time over perturbed instances. Unfortunately, the results of smoothed analysis (section 2.3) do not hold in general for problems with exponential dimension. Nevertheless, we show that there is a specific perturbation scheme and accompanying exact algorithm, both designed carefully to exploit the particular structure of multi-unit auctions, such that the expected running time of the algorithm over perturbed instances is polynomial in n and $\log m$. We now describe the intuition behind our perturbation scheme.

We start with the following observation. In searching for an optimal solution to an instance of multi-unit auctions, it suffices to evaluate a valuation function v of a player only at *dominant points*, defined as follows: A point $k \in [m]$ is *dominant* in valuation function $v : [m] \to \mathbb{R}_+$ if $v(k) > v(j)$ for all $j < k$. If we restrict our attention to instances of multi-unit auctions with a polynomial number of dominant points in each player's valuation function and, moreover, assume that a list of these points is provided as part of the input to the problem, we could rewrite an instance as a binary packing problem with *polynomial dimension* and apply the techniques of sections 3 and 4 to obtain a truthful-in-expectation FPTAS for these instances.

No such guarantee on the number of dominant points is possible in general. However, we show that there is a specific linear perturbation of valuation functions, which we refer to as the *2-adic* perturbation, such that the perturbed function is guaranteed to have only a polynomial number of dominant points, and moreover a list of these points can be computed in polynomial time. Composing the 2-adic perturbation with a variant of the RS perturbation scheme (section 4.1) then yields a truthful-in-expectation FPTAS as in section 4.

Tractable perturbation with exponential dimension. Polynomial-time solvability of the perturbed optimization problem is not enough to apply our framework in sections 3 and 4. We must additionally apply our perturbation scheme efficiently. In section 3, this requirement is called *tractability*, and it requires efficient sampling of a perturbation matrix P, as well as the construction of a perturbed solution with expectation $P^T x$ given a feasible solution x. Unfortunately, this requirement is a non-starter for problems of exponential dimension; this is because no scheme can produce in polynomial time an explicit description of a perturbation matrix with an exponential number of entries. Nevertheless, we observe that a relaxed definition of tractability suffices. We say a perturbation scheme Ψ for multi-unit auctions is *tractable with deferred decisions* if the following hold.

(I) A matrix $P \in \mathbb{R}^{n\times m}$ drawn from Ψ is uniquely determined by a vector δ of independent random variables. We require that the length of δ is at most exponential in n and $\log m$ so that individual entries of δ may be indexed using polynomial space. Moreover, we require that each entry of δ can be sampled efficiently.

(II) For each objective vector $v \in \mathbb{R}^{n\times m}$ for multi-unit auctions and each $i \in [n]$ and $j \in [m]$, an entry $(Pv)_{ij}$ of the perturbed valuation vector can be evaluated in polynomial time by reading only a polynomial (in n and $\log m$) number of entries of δ.

(III) For every binary vector x in the feasible set $\mathcal{S}$ for multi-unit auctions, represented succinctly as a mapping from each player to a number of items, a distribution D_x with support in $\mathcal{S}$ and expectation $P^T x$ can be sampled from in time polynomial in n and $\log m$ by reading only a polynomial number of entries of δ.

We design our perturbation scheme to satisfy tractability with deferred decisions. To account for this modified notion, we slightly modify the PB allocation rule by combining steps 1 and 2 of Algorithm 1. In the *modified PB allocation rule*, we give algorithm $\mathcal{A}$ oracle access to both v and the vector δ of random variables describing the perturbation matrix P, and we require that $\mathcal{A}$ output an exact solution to the perturbed instance $(\mathcal{S}, Pv)$. The following modification of Theorem 3.1 now sets the stage.

THEOREM 5.5. *If Ψ is a feasible, linear, approximation-preserving perturbation scheme for multi-unit auctions, and, moreover, Ψ is tractable with deferred decisions, then the modified PB allocation rule for multi-unit auctions satisfies the following properties:*

(a) *it is MIDR and hence defines a truthful-in-expectation mechanism when combined with suitable payments;*

(b) *for every instance of Π and $\epsilon > 0$, it outputs a feasible solution with expected objective function value at least $(1-\epsilon)$ times the maximum possible;*

(c) *its worst-case expected running time is bounded by a polynomial in n and $\log m$ plus the running time of the exact algorithm $\mathcal{A}$ for applying the perturbation*

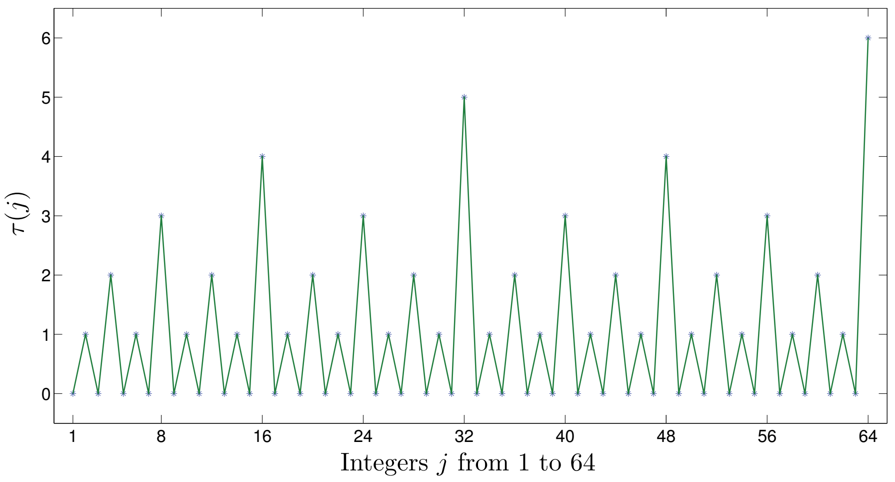


FIG. 2. *The 2-adic valuation.*

and producing the exact solution to the perturbed instance $(\mathcal{S}, Pv)$.

Proof. The proofs of properties (a) and (b) are identical to those in Theorem 3.1. For (c), note that properties (I) and (III) imply that steps 3 and 4 of the PB allocation rule (Algorithm 1) can be implemented in polynomial time. ∎

5.4.3. The 2-adic perturbation. In this section, we define the *2-adic perturbation* $v \to v^\sigma$, parametrized by $\sigma > 0$, and explore some of its formal properties. The 2-adic perturbation will serve as a building block for our perturbation scheme for multi-unit auctions, to be defined in section 5.4.4. We note that the 2-adic perturbation is similar to the "weights" employed in [7].

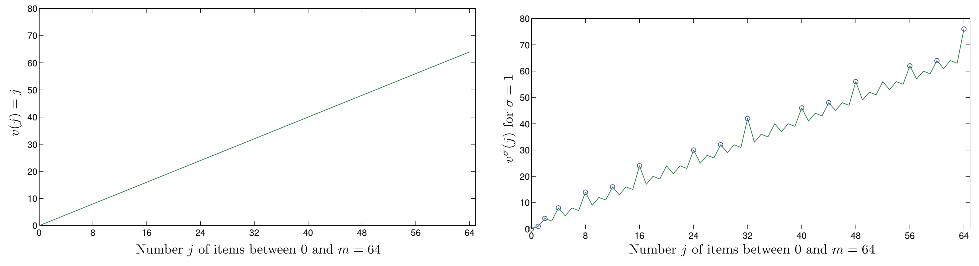
For a positive integer j, we let $\tau(j)$ denote the exponent of the largest power of 2 that divides j. The function τ appears in many contexts and is known as the 2-*adic valuation function*, among other names. We illustrate the structure of τ by the plot in Figure 2. There are multiple equivalent definitions of τ: $\tau(j)$ is the number of trailing zeros in the binary representation of j, and, equivalently, $\tau(j) = t$ if and only if $j/2^t$ is an odd integer. For convenience, we set $\tau(0) = 0$. It is immediate that $\tau(j)$ can be evaluated in time polynomial in the length of the binary representation of j. Since we will need to evaluate τ only for integers between 1 and m, this is possible in time polynomial in $\log m$.

Given a non-decreasing valuation function $v : [m] \to \mathbb{R}_+$ and $\sigma > 0$, we define the perturbed valuation v^σ as follows:

$$v^\sigma(j) = v(j) + 2\sigma \cdot \tau(j). \tag{8}$$

Thus the 2-adic perturbation gives a "bonus" to the valuation at each point j, proportional to the 2-adic valuation $\tau(j)$ of j. This bonus varies in a periodic and hierarchical pattern with the number j (Figure 2). The 2-adic perturbation has the effect of reducing the number of dominant points of a valuation, as illustrated in Figure 3.

Looking ahead to section 5.4.4, our perturbation scheme for multi-unit auctions will in effect compose the 2-adic perturbation with the RS perturbation scheme of section 4. Because the RS perturbation scheme may change the number of dominant points of a valuation in general, we bound both the number of dominant points of a valuation v^σ perturbed by the 2-adic perturbation and the number of points that are



(a) Unperturbed valuation $v(j) = j$. Every point is dominant.

(b) 2-adic perturbation of v with $\sigma = 1$. Dominant points are circled.

Fig. 3. *Applying the 2-adic perturbation.*

even *near-dominant.* This "leaves room" for a variant of the RS perturbation scheme to act on v^σ, without substantially increasing the number of its dominant points.

Given a function $u : [m] \to \mathbb{R}_+$, we say $k \in [m]$ is a *σ-dominant point of u* if $u(k) > u(\ell) - \sigma$ for all $\ell < k$. Observe that a point is dominant (section 5.4.2) if and only if it is 0-dominant. If u is non-decreasing and $\sigma > 0$, then every point of u is σ-dominant and the definition of approximate dominance is uninteresting. The 2-adic perturbation is designed to reduce the "dimensionality" of the valuation function from $[m]$ to a polynomial in $\log m$ and $\frac{v(m)}{\sigma}$, in the following strong sense.

Lemma 5.6. *Fix $\sigma > 0$ and a non-decreasing function $v : [m] \to [0, v_{\max}]$. The number of σ-dominant points of v^σ is at most $O(\log m \cdot \frac{v_{\max}}{\sigma})$. Moreover, given only value-oracle access to v, there is an algorithm that runs in time polynomial in $\log m$ and $\frac{v_{\max}}{\sigma}$ and outputs a subset of $[m]$ that is guaranteed to include all σ-dominant points of v^σ.*

The proof of Lemma 5.6 builds on the following claim.

Claim 5.7. *Fix $\sigma > 0$ and a non-decreasing function $v : [m] \to [0, v_{\max}]$. If k is a σ-dominant point of v^σ and $j < k$, then either $v(k) > v(j) + \sigma$ or $\tau(k) > \tau(j)$.*

Proof. We assume that $j < k$, $v(k) \le v(j) + \sigma$, and $\tau(k) \le \tau(j)$ and show that k is not a σ-dominant point of v^σ. Recall that $\tau(j)$ and $\tau(k)$ are the number of trailing zeros in the binary representations of j and k, respectively. Let $\ell = k - 2^{\tau(k)}$. Using the binary-representation interpretation of τ shows that ℓ is simply the result of zeroing out the rightmost 1 from the binary representation of k, and therefore $\tau(\ell) \ge \tau(k) + 1$. Moreover, since the binary representation of j has at least $\tau(k)$ trailing zeros and $j < k$, it follows that $j \le l$. The following derivation shows that k does not σ-dominate ℓ in v^σ, completing the proof:

$$\begin{aligned} v^\sigma(\ell) &= v(\ell) + 2\sigma\tau(\ell) \\ &\ge v(\ell) + 2\sigma\tau(k) + 2\sigma \\ &\ge v(j) + 2\sigma\tau(k) + 2\sigma \\ &\ge v(k) - \sigma + 2\sigma\tau(k) + 2\sigma \\ &= v^\sigma(k) + \sigma. \qquad \square \end{aligned}$$

Proof of Lemma 5.6. We bound the number of σ-dominant points of v^σ by enumerating a list of at most $O(\log m \cdot \frac{v_{\max}}{\sigma})$ points that must include all σ-dominant points. First, we divide $\{0, 1, \ldots, m\}$ into $O(\frac{v_{\max}}{\sigma})$ disjoint intervals, the sth of which is

$$I_s = \{j \in [m] : \sigma s < v(j) \leq \sigma(s+1)\}\,.$$

Claim 5.7 implies that, for each interval I_s and integer $t \in \{0, 1, \ldots, \log m\}$, there is at most one σ-dominant point k of v^σ with both $k \in I_s$ and $\tau(k) = t$. This bounds the number of σ-dominant points of v^σ by $O(\log m \cdot \frac{v_{\max}}{\sigma})$, completing the proof of the first part of the lemma.

Claim 5.7 implies an even stronger statement: if $k \in I_s$ is a σ-dominant point of v^σ and $\tau(k) = t$, then k must be the smallest integer j in I_s with $\tau(j) = t$. It is easy to verify that, given the endpoints of a segment I_s and an integer t, we can compute in $O(\log m)$ time the smallest integer $k \in I_s$ that is divisible by 2^t, if any. Moreover, the delimiting points of the intervals $\{I_s\}_s$ can each be computed in $O(\log m)$ time using binary search. This completes the proof of the second part of the lemma. ∎

5.4.4. Combining the 2-adic and RS perturbation schemes. We now describe a perturbation scheme combining the RS perturbation scheme with the 2-adic perturbation, which we call RS2. As for the RS scheme, we describe RS2 via its adjoint. Consider an instance of multi-unit auctions described by the number m of items and n valuation functions $v_1, \ldots, v_n : [m] \to \mathbb{R}_+$. Let $\mathcal{S}$ be the set of feasible mn-dimensional binary vectors x, where x_{ij} indicates whether exactly j items are assigned to player i, as described by the binary packing problem in (7). Given $x \in \mathcal{S}$, described succinctly as a mapping from players to integers in $[m]$, and a parameter $\epsilon > 0$, we consider the following randomized algorithm:

(1) For each $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, m$, draw δ_{ij} uniformly from the interval $[0, \frac{\epsilon}{4n \log m}]$, and let $\delta'_{ij} = \frac{\epsilon}{2n \log m}\tau(j)$, where τ is as defined in section 5.4.3.
(2) Output a random allocation $y \in \mathcal{S}$ according to the following distribution: With probability $1-\epsilon$, output $y = x$. Otherwise, choose a player $i \in \{1, \ldots, n\}$ uniformly at random, let e_{im} denote the allocation that assigns all items to player i, and output $y = e_{im}$ with probability

$$\sum_{i,j} (\delta_{ij} + \delta'_{ij}) x_{ij}. \tag{9}$$

With the remaining probability, output $y = 0$, the empty allocation.

The probabilities in step (2) are well defined in that they sum to 1, because a binary vector x that encodes a feasible solution for multi-unit auctions includes at most n non-zero entries, and $\delta_{ij} + \delta'_{ij} \leq \epsilon/n$ for each i and j.

There are two main differences between the RS2 and RS perturbation schemes. First, the RS scheme gives an allocation x a "bonus," in the form of a random-singleton allocation, with probability proportional to the number of non-zero entries of x. Polynomial dimension guarantees that the value of the random-singleton bonus is at the right scale, up to a polynomial factor, so that the perturbation is diffuse enough to permit polynomial smoothed complexity. In exponential dimension, however, only an exponentially small fraction of the singletons may have sufficient value. Therefore, RS2 employs a random choice of n specific singletons—those that allocate all items to a single player. The structure of multi-unit auctions implies that the expected value of this random bonus is at the right scale, up to a factor of n. The second and perhaps most important difference between RS and RS2 concerns the probability with which the bonus is awarded. Whereas in RS this probability depends only on the number of non-zero entries of x, in RS2 this probability depends additionally on which entries of x are non-zero, using the 2-adic valuation to effectively place "weights" on various entries of x.

After conditioning on the random choices in step (1), the expectation $\widetilde{x}$ of the distribution D_x over solutions $\mathcal{S}$ defined by step (2) can be expressed via the adjoint perturbation P^T given by

$$\widetilde{x} = P^T x = (1-\epsilon)x + \left(\sum_{i=1}^{n}\sum_{j=1}^{m}(\delta_{ij}+\delta'_{ij})x_{ij}\right)\left(\sum_{i=1}^{n}\frac{e_{im}}{n}\right).$$

Let δ and δ' denote the nm-vectors of the δ_{ij}'s and δ'_{ij}'s, respectively. Since P^T can be written as $(1-\epsilon)I + \frac{1}{n}(\sum_{i=1}^{n} e_{im})(\delta+\delta')^T$, dualizing gives the following formal definition of the RS2 perturbation scheme for multi-unit auctions, given an instance of the problem with n players, the approximation parameter ϵ, and conditioned on the random choices of the δ_{ij}'s:

$$P = P(\delta) = (1-\epsilon)I + \frac{(\delta+\delta')(\sum_{i=1}^{n} e_{im})^T}{n}. \tag{10}$$

This corresponds to the linear perturbation

$$v_i(j) \mapsto (1-\epsilon)v_i(j) + \frac{\delta_{ij}+\delta'_{ij}}{n}\sum_{i'=1}^{n} v_{i'}([m]) \tag{11}$$

for each i and j.

5.4.5. A truthful FPTAS for multi-unit auctions. In this section, we show that the RS2 perturbation scheme can be used in conjunction with the modified PB-allocation rule to yield a truthful-in-expectation FPTAS for multi-unit auctions. We begin by proving an analogue of Lemma 4.1.

LEMMA 5.8. *The RS2 perturbation scheme for multi-unit auctions is feasible, linear, approximation preserving, and tractable with deferred decisions.*

Proof. The proof of feasibility, linearity, and approximation preservation is identical to that in Lemma 4.1. We now prove that the scheme is tractable with deferred decisions. Property (I) is satisfied because P is uniquely defined by the mn independent random variables δ_{ij}, each of which can be sampled in polynomial time. Property (II) is satisfied because the perturbed entry $(Pv)_{ij}$, as seen in (11), can be evaluated in $O(n)$ time given the value of δ_{ij} and oracle access to entries of the original valuations v. For property (III), we recall that we described our perturbation explicitly via its adjoint: For each solution x we described a distribution D_x over feasible solutions that has expectation $P^T x$. Examining our definition, it is clear that sampling from D_x reduces to evaluating the probability $\sum_{ij}(\delta_{ij}+\delta'_{ij})x_{ij}$ (equation (9)). This is possible in time polynomial in n and $\log m$: a feasible solution x for multi-unit auctions has at most n non-zero entries, δ_{ij} can be sampled in polynomial time, and $\tau(j)$ and hence δ'_{ij} can be evaluated in polynomial time. ∎

We next show that instances of multi-unit auctions can be perturbed by the RS2 perturbation scheme and then solved exactly, all in expected polynomial time.

LEMMA 5.9. *Let $\epsilon > 0$ be a parameter. Let $v_1, \ldots, v_n : [m] \to \mathbb{R}_+$ denote an instance of multi-unit auctions, presented as n value oracles. Let $\delta_{ij} \in [0, \frac{\epsilon}{4n\log m}]$ for $(i,j) \in [n]\times[m]$, and assume that δ is presented as an oracle indexed by i and j. There* *is an algorithm $\mathcal{A}$ that outputs an exact solution for perturbed instance Pv of multi-unit auctions, where $P = P(\delta)$ is as in* (10)*, and runs in expected time polynomial in n, $\log m$, and $1/\epsilon$ when the δ_{ij}'s are independently and uniformly distributed in $[0, \frac{\epsilon}{4n\log m}]$.*

Proof. Define $\sigma = \frac{\epsilon}{4n\log m} \cdot \frac{\sum_{i=1}^{n} v_i([m])}{n}$ and $\eta_{ij} = \delta_{ij} \cdot \frac{\sum_{i=1}^{n} v_i([m])}{n}$ for each $(i,j) \in [n] \times [m]$. In this notation, the RS2 perturbation scheme mapping v to $\widehat{v} = Pv$ (equation (11)) is

$$\widehat{v}_i(j) = (1-\epsilon)v_i(j) + 2\sigma\tau(j) + \eta_{ij}, \tag{12}$$

where the η_{ij}'s are independently and uniformly distributed in $[0,\sigma]$. Equation (12) allows us to describe the RS2 perturbation scheme as the composition of two steps. For each player i, the RS2 scheme (a) scales player i's valuation v_i by $1-\epsilon$ and applies the 2-adic perturbation (section 5.4.3) with parameter σ to the scaled valuation, and then (b) adds "noise" η_{ij}, drawn independently and uniformly from $[0,\sigma]$, to each entry j of the valuation.

We first show that, for each player i, there is a polynomial number of points $B_i \subseteq [m]$, independent of the noise vector η, such that all dominant points of $\widehat{v}_i$ lie in B_i, and, moreover, the list B_i can be computed in polynomial time. Denote by $\widetilde{v}_i(j) = (1-\epsilon)v_i(j) + 2\sigma\tau(j)$ the result of applying only part (a) of the RS2 perturbation, so that $\widehat{v}_i(j) = \widetilde{v}_i(j) + \eta_{ij}$. Observe that a dominant point of $\widehat{v}_i$ is a σ-dominant point of $\widetilde{v}_i$, because $0 \le \eta_{ij} \le \sigma$ for all i and j. Moreover, Lemma 5.6 guarantees that a list of σ-dominant points of $\widetilde{v}_i$ can be computed in time polynomial in $\log m$ and $\frac{\widetilde{v}_i([m])}{\sigma}$, where the latter quantity is at most $\frac{4n^2\log m}{\epsilon}$ by definition of σ.

For each player i, an optimal solution must assign i a number of items that is dominant for $\widehat{v}_i$. Therefore, after computing the sets B_i for each player i, we can rewrite the problem of maximizing the perturbed welfare $\sum_i \widehat{v}_i(j)x_{ij}$ as a binary packing problem of polynomial dimension, with a variable x_{ij} for each player i and $j \in B_i$. Moreover, this reformulation is independent of the "noise" vector η. As a result, ours is a perturbed instance where each entry of the objective function is independent and has density $\frac{1}{\sigma} \le \frac{4n^2\log m}{\epsilon v_{max}}$, where $v_{max} = \max_{ij} v_i(j)$. Since the (non-truthful) FPTAS for the knapsack problem extends easily to multi-unit auctions, Proposition 2.4 implies that this perturbed instance can be solved in expected time polynomial in n, $\log m$, and $1/\epsilon$. This completes the proof. ∎

Combining Lemmas 5.8 and 5.9 with Theorem 5.5 provides a truthful-in-expectation FPTAS for multi-unit auctions. This rederives the main result of [7] in our perturbation-based framework.

THEOREM 5.10 (see [7]). *There is a truthful-in-expectation FPTAS for multi-unit auctions.*

Proof. Let $\mathcal{A}$ denote an algorithm for perturbing and solving an instance of multi-unit auctions, as described in Lemma 5.9. Let Ψ denote the RS2 perturbation scheme. Instantiate the modified PB allocation rule for multi-unit auctions with the scheme Ψ and algorithm $\mathcal{A}$. Since Ψ is feasible, linear, approximation preserving, and tractable with deferred decisions (Lemma 5.8), Theorem 5.5 implies that this allocation rule is MIDR, has an approximation guarantee of $1-\epsilon$ in expectation (for an arbitrary supplied parameter ϵ), and has expected running time bounded by a polynomial in n and $\log m$ plus the running time of Algorithm $\mathcal{A}$. Lemma 5.9 then bounds the expected running time by a polynomial in n, $\log m$, and $\frac{1}{\epsilon}$.

To complete the proof, Proposition 2.2 implies that computing truth-telling payments for this allocation rule increases the overall running time by only a polynomial factor. ☐

Acknowledgments. We thank Amos Fiat and Brendan Lucier for helpful discussions and comments.

REFERENCES

[1] M. Babaioff, R. Lavi, and E. Pavlov, *Single-value combinatorial auctions and algorithmic implementation in undominated strategies*, J. ACM, 56 (2009), 4.

[2] X. Bei and Z. Huang, *Bayesian incentive compatibility via fractional assignments*, in Proceedings of the 22nd ACM Symposium on Discrete Algorithms (SODA), ACM, New York, 2011, pp. 720–733.

[3] R. Beier and B. Vöcking, *Typical properties of winners and losers in discrete optimization*, SIAM J. Comput., 35 (2006), pp. 855–881.

[4] P. Briest, P. Krysta, and B. Vöcking, *Approximation techniques for utilitarian mechanism design*, SIAM J. Comput., 40 (2011), pp. 1587–1622.

[5] D. Buchfuhrer, S Dughmi, H. Fu, R. Kleinberg, E. Mossel, C. Papadimitriou, M. Schapira, Y. Singer, and C. Umans, *Inapproximability for VCG-based combinatorial auctions*, in Proceedings of the 21st ACM Symposium on Discrete Algorithms (SODA), ACM, New York, 2010, pp. 518–536.

[6] S. Dobzinski, *An impossibility result for truthful combinatorial auctions with submodular valuations*, in Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC), ACM, New York, 2011, pp. 139–148.

[7] S. Dobzinski and S. Dughmi, *On the power of randomization in algorithmic mechanism design*, in Proceedings of the 50th IEEE Symposium on Foundations of Computer Science (FOCS), IEEE, Washington, DC, 2009, pp. 505–514.

[8] S. Dobzinski and N. Nisan, *Mechanisms for multi-unit auctions*, in Proceedings of the 8th ACM Conference on Electronic Commerce (EC), ACM, New York, 2007, pp. 346–351.

[9] S. Dobzinski and N. Nisan, *Limitations of VCG-based mechanisms*, Combinatorica, 31 (2011), pp. 379–396.

[10] S. Dughmi and J. Vondrák, *Limitations of randomized mechanisms for combinatorial auctions*, in Proceedings of the 52nd IEEE Symposium on Foundations of Computer Science (FOCS), IEEE, Washington, DC, 2011, pp. 502–511.

[11] G. V. Gens and E. V. Levner, *Approximate algorithms for certain universal problems in scheduling theory*, Soviet J. Comput. System Sci., 6 (1978), pp. 38–43.

[12] G. V. Gens and E. V. Levner, *Fast approximation algorithms for knapsack type problems*, in Optimization Techniques, Part 2, K. Iracki, K. Malinowski, and S. Walukiewicz, eds., Springer, Berlin, 1980, pp. 185–194.

[13] J. D. Hartline, R. D. Kleinberg, and A. Malekian, *Bayesian incentive compatibility via matchings*, in Proceedings of the 22nd ACM Symposium on Discrete Algorithms (SODA), ACM, New York, 2011, pp. 734–747.

[14] J. D. Hartline and B. Lucier, *Bayesian algorithmic mechanism design*, in Proceedings of the 41st ACM Symposium on Theory of Computing (STOC), ACM, New York, 2010, pp. 301–310.

[15] R. Hassin, *Approximation schemes for the restricted shortest path problem*, Math. Oper. Res., 17 (1992), pp. 36–42.

[16] D. S. Johnson and K. A. Niemi, *On knapsacks, partitions, and a new dynamic programming technique for trees*, Math. Oper. Res., 8 (1983), pp. 1–14.

[17] J. A. Kelner and D. A. Spielman, *A randomized polynomial-time simplex algorithm for linear programming*, in Proceedings of the 37th ACM Symposium on Theory of Computing (STOC), ACM, New York, 2006, pp. 51–60.

[18] R. Lavi, *Computationally efficient approximation mechanisms*, in Algorithmic Game Theory, N. Nisan, T. Roughgarden, É. Tardos, and V. Vazirani, eds., Cambridge University Press, Cambridge, UK, 2007, pp. 301–329.

[19] R. Lavi, A. Mu'alem, and N. Nisan, *Towards a characterization of truthful combinatorial auctions*, in Proceedings of the 44th IEEE Symposium on Foundations of Computer Science (FOCS), IEEE, Washington, DC, 2003, pp. 574–583.

[20] R. Lavi and C. Swamy, *Truthful and near-optimal mechanism design via linear programming*, J. ACM, 58 (2011), 25.

[21] M. V. Marathe, R. Ravi, R. Sundaram, S. S. Ravi, D. J. Rosenkrantz, and H. B. Hunt, *Bicriteria network design problems*, J. Algorithms, 28 (1998), pp. 142–171.

[22] N. Nisan, *Introduction to mechanism design (for computer scientists)*, in Algorithmic Game Theory, N. Nisan, T. Roughgarden, É. Tardos, and V. Vazirani, eds., Cambridge University Press, Cambridge, UK, 2007, pp. 209–241.

[23] N. Nisan and A. Ronen, *Algorithmic mechanism design*, Games Econom. Behav., 35 (2001), pp. 166–196.

[24] C. Papadimitriou, M. Schapira, and Y. Singer, *On the hardness of being truthful*, in Proceedings of the 49th IEEE Symposium on Foundations of Computer Science (FOCS), IEEE, Washington, DC, 2008, pp. 250–259.

[25] H. Röglin and S.-H. Teng, *Smoothed analysis of multiobjective optimization*, in Proceedings of the 50th IEEE Symposium on Foundations of Computer Science (FOCS), IEEE, Washington, DC, 2009, pp. 681–690.

[26] S. Sahni, *General techniques for combinatorial approximation*, Oper. Res., 25 (1977), pp. 920–936.

[27] D. A. Spielman and S.-H. Teng, *Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time*, J. ACM, 51 (2004), pp. 385–463.

[28] B. Vöcking, *A universally-truthful approximation scheme for multi-unit auctions*, in Proceedings of the 23rd ACM Symposium on Discrete Algorithms (SODA), ACM, New York, 2012, pp. 846–855.