American Economic Review 2015, 105(10): 3102–3124
http://dx.doi.org/10.1257/aer.20130712

Non-Optimal Mechanism Design[†]

By Jason D. Hartline and Brendan Lucier*

The optimal allocation of resources in complex environments—like allocation of dynamic wireless spectrum, cloud computing services, and Internet advertising—is computationally challenging even given the true preferences of the participants. In the theory and practice of optimization in complex environments, a wide variety of special and general purpose algorithms have been developed; these algorithms produce outcomes that are satisfactory but not generally optimal or incentive compatible. This paper develops a very simple approach for converting any, potentially non-optimal, algorithm for optimization given the true participant preferences, into a Bayesian incentive compatible mechanism that weakly improves social welfare and revenue. (*JEL* D82, H82, L82)

Economic mechanisms mediate the strategic interactions between parties with diverse and selfish interests. The guiding principle of mechanism design is that a carefully chosen mechanism can align the participants' incentives with the designer's goals, whereas other mechanisms may have unintended strategic consequences. In practice, however, mechanisms are developed by governments, marketing firms, technology firms, and other organizations and are deployed in complex environments for which optimization is not straightforward. These mechanisms are developed by trial, error, and with the help of domain-specific understanding and without careful consideration of incentives. In this paper we develop a theory of mechanism design that is compatible with such real-world considerations.

While there are many context-specific reasons why a mechanism designer may not be able to optimally solve the mechanism design problem, our main motivation is one of computation. For many important settings of mechanism design, even ignoring the incentives of the participants, finding the exact optimal outcome is computationally intractable. This intractability means that any optimization algorithm must sometimes either take a prohibitively long time to find a solution, or find only a suboptimal solution. As an example, consider the dynamic allocation

* Hartline: Electrical Engineering and Computer Science Department, Northwestern University, 2133 Sheridan Road, Evanston, IL 60208 (e-mail: hartline@eecs.northwestern.edu); Lucier: Microsoft Research, One Memorial Drive, Cambridge, MA 02139 (e-mail: brlucier@microsoft.com). Supported in part by NSF grant CCF-0830773 and NSF Career Award CCF-0846113. We thank Randall Berry, Larry Blume, Peter Cramton, Ben Golub, Greg Lewis, Stephen Morris, Denis Nekipelov, and Glen Weyl for their perceptive comments and suggestions. We also thank anonymous referees for many helpful suggestions. The authors declare that they have no relevant or material financial interests that relate to the research described in this paper.

[†] Go to http://dx.doi.org/10.1257/aer.20130712 to visit the article page for additional materials and author disclosure statement(s).

of broadband wireless spectrum which is proposed to make the usage of spectrum more efficient and allow access to spectrum for small-scale technology applications. At the heart of this dynamic spectrum allocation is an *independent set* problem—access can only be granted to a set of broadcasts that do not interfere—that needs to be solved every time broadcast rights are to be reallocated. For problems at the scale of broadband wireless broadcast, finding optimal solutions to such independent set problems is infeasible. Ignoring incentive constraints, the literature on practical but non-optimal algorithm design provides numerous solutions to this intractability. These include: greedy algorithms, branch and bound, linear program relaxation, and local search. The theory we develop provides a method for constructing simple incentive-compatible mechanisms from any of these non-optimal algorithms for the (non-game-theoretic) optimization problem; the approach does not depend on the specifics of the algorithm.

The Vickrey-Clarke-Groves (Vickrey 1961, Clarke 1971, and Groves 1973) approach for theoretical mechanism design can be viewed as decomposing a mechanism into two components:

(i) An *algorithm* maps the reported values of the agents to an outcome.

(ii) *Payments* are determined that align the incentives of the agents with the outcome of the algorithm.

Incentive-aligning payments only exist for algorithms satisfying special properties (Myerson 1981). While optimal algorithms always satisfy these properties, non-optimal algorithms generally do not (e.g., Lehmann, O'Callaghan, and Shoham 2002). There is now a large literature on developing special-purpose mechanisms that simultaneously resolve incentive and engineering constraints for specific problems of interest.

We take an alternative approach and give a theory for non-optimal mechanism design that decomposes a mechanism into three components:

(i) An *ironing procedure*[1] maps reported values of the participants into ironed values.

(ii) The algorithm maps the ironed values to an outcome.

(iii) Payments are determined that align the incentives of the agents with the outcome of the ironing procedure and algorithm.

Our main result is that, for *any* algorithm (ii), there is a simple ironing procedure (i) that weakly improves the performance of the algorithm (e.g., welfare or revenue) and admits payments (iii) for aligning the agents' incentives with the resulting outcome. This construction also preserves many other properties of the algorithm,

[1] We refer to this procedure as ironing because of its similarities to the ironing procedure of Myerson (1981). This ironing procedure replaces any reported value that is within a set of prespecified intervals with a random value from the same interval.

as the final outcome is simply the algorithm's prescription when applied to ironed values instead of the original reports. The incentive problem of constructing a mechanism from any engineering solution to the underlying algorithmic optimization problem is resolved.

An Illustrative Example.—We describe below the literature on dynamic allocation of broadband spectrum and the relevance of our work thereto. We will begin by presenting, without motivation, one of the prevalent models from the engineering literature on this subject. We will describe both the failure of the straightforward engineering approach to give mechanisms with good incentive properties, the limited success of approaches that attempt to simultaneously address the incentive and engineering concerns, and how our approach can be used to resolve the issue of incentives for *any* engineering solution. We will conclude the discussion with motivation for the dynamic allocation of spectrum in comparison to the traditional static allocation of spectrum where, heretofore, auctions have had considerable success. It should be noted that the application of our methods to dynamic allocation of broadband spectrum is for illustrative purposes and it in itself is not a focus of our work.

The model for dynamic broadband spectrum allocation from Jia et al. (2009) is as follows. Each participating firm is associated with a collection of towers from which it could potentially broadcast, and this association is publicly known. Firms prefer to broadcast from as many towers as possible to increase customer coverage. However, certain pairs of towers interfere with each other, and only one of each such interfering pair can broadcast. A feasible outcome is a subset of the towers that contains no interfering pair, also known as an *independent set*. Each firm i has a private value $v_i \geq 0$ representing the value obtained for each of its towers selected in the outcome. The underlying optimization problem, faced each time spectrum is to be reallocated, is to find a maximum value independent set in the interference graph. The spectrum is to be reallocated periodically by the same mechanism, these reallocations are considered separately, and the model considers only a single period.[2]

It is not unreasonable for spectrum auctions to involve thousands of broadcast towers each with numerous interference constraints (Milgrom and Segal 2014; Fréchette, Newman, and Leyton-Brown 2015; see Figure 1). Given the scale and intractability of the allocation optimization problem, engineering solutions must use heuristic algorithms. The following *greedy heuristic* was empirically analyzed by Zhou et al. (2008). The heuristic assigns each tower a score equal to the owning firm's bid divided by the number of other towers with which it interferes. The towers are then considered in decreasing order of score. Broadcast access is granted to each tower that does not interfere with any towers previously granted access. Greedy heuristics are common in practice and for this dynamic broadband spectrum problem, Zhou et al. (2008) demonstrate empirically that this method achieves high welfare on simulated auction instances.

[2] This description omits many complicating factors that arise in practice, such as dynamic incentives, partial interference, and the simultaneous allocation of multiple bands. For example, with multiple bands the underlying optimization problem is that of *graph coloring* and is only more challenging. More generally, a firm may have different values for broadcasting from distinct towers. The follow-up work of Hartline, Kleinberg, and Malekian (2015) and Bei and Huang (2011) generalizes our approach to such multi-dimensional preferences; see Section V.

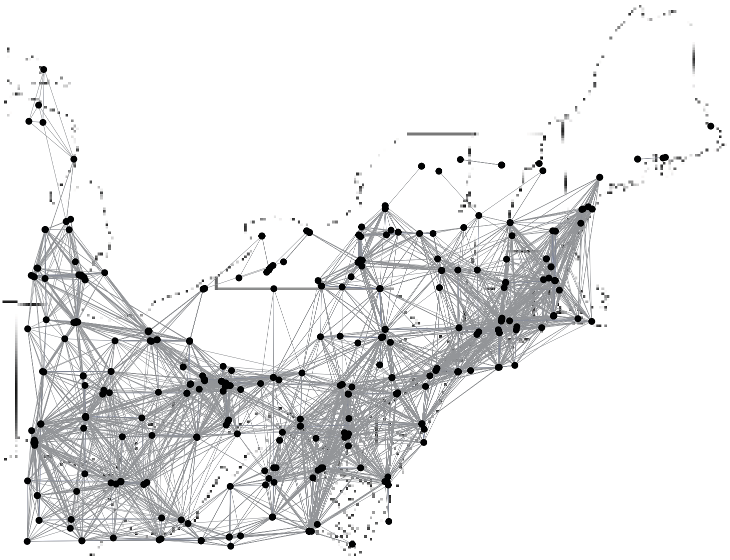


Figure 1. Interference Constraints between Broadcast Towers

Note: A sample of interference constraints between broadcast towers in the Northeastern United States, for an upcoming FCC spectrum auction (Fréchette, Newman, and Leyton-Brown 2015).

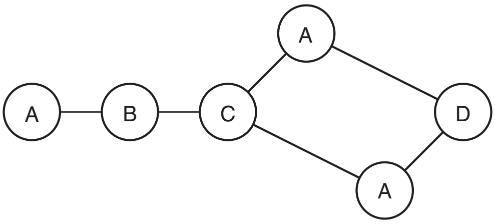
Source: Figure derived from the FCC's May 2014 constraint data (FCC, 2014). We thank Alexandre Fréchette and Kevin Leyton-Brown for their help with producing this image.

For the dynamic spectrum allocation problem above, Jia et al. (2009) observe that the greedy heuristic is not incentive compatible, i.e., there are no payments that incentivize the broadcasters to truthfully reveal their preferences. This failure of the greedy heuristic can be seen in the example illustrated in Figure 2. Recall that the interference graph is known and each firm makes a single bid representing a value per tower allocated. If each firm bids according to its true value, then the towers' scores are (from left to right) 10, 9, 6, 5, 5, and 1. The greedy heuristic would then allocate the first, third, and sixth towers, and in particular firm A receives the right to broadcast on a single tower. However, if firm A were to lower its bid to 8, then it would instead win broadcast rights for two towers (the fourth and fifth). This non-monotonicity in the allocation, with respect to the bid of firm A, implies that this auction is not incentive compatible.

Given this lack of incentive compatibility of what we might consider the first-best engineering solution (i.e., the greedy algorithm described above), the engineering literature has turned to the development of second-best solutions that simultaneously solve the engineering problem and the incentive problem. For the model above, one solution from Jia et al. (2009) is the following. Consider each firm in decreasing order by bid, and allocate to each firm either all of its requested towers (if this is feasible given prior allocations) or none. The resulting mechanism is incentive compatible (with the appropriate payments); however, measured empirically it underperforms relative to the first-best algorithm (Jia et al. 2009; Al-Ayyoub and Gupta 2011).

In contrast to this literature on special-purpose mechanisms that simultaneously solve incentive and engineering problems, we give a generic method that can be applied to any algorithm for spectrum allocation (or any other problem). In the general single-dimensional model for Bayes-Nash mechanism design, we derive from

Panel A. Interference graph



Panel B. Firm values

Firm	Value
A	10
B	18
C	18
D	2

Figure 2. A Toy Instance of a Spectrum Allocation Problem

Notes: Each of four firms (A, B, C, D) has value per tower as specified in panel B. The towers are laid out as in panel A, labeled by owning firm; two towers are connected by an edge if they mutually interfere. A real instance would have many thousands of towers.

this method a mechanism that preserves first-best performance (welfare or revenue) of the original algorithm. To illustrate the outcome of our construction when applied to the greedy heuristic of Zhou et al. (2008) described above, suppose that firm A's value is drawn uniformly from the interval $[0, 20]$ and the other firms' values are fixed as described in Figure 2. The constructed mechanism elicits a single reported value from each agent. If firm A's reported value lies in the interval $[9, 12]$, then the mechanism draws a new value uniformly at random from that interval and uses this random value in place of A's declared value. Finally, it invokes the original greedy algorithm on this modified valuation profile and implements the resulting outcome. This mechanism is Bayesian incentive compatible (when paired with the appropriate payment rule) and improves expected welfare.

We conclude our discussion of the dynamic broadband spectrum allocation with the motivation we previously deferred. Broadcast rights have traditionally been allocated via static, long-term contracts over large geographic regions (e.g., Cramton 2002). However, such long-term allocations are inefficient, in part because only the largest firms can afford such contracts, and because they are not easily adapted to dynamic variation of inter- and intra-firm demands for broadcasts. Alternatively, in a dynamic spectrum auction, smaller local contracts are allocated among many heterogeneous firms (Zhou et al. 2008). These contracts are localized both geographically and temporally. Auctions that have been traditionally used to allocate large contracts infrequently are inappropriate for dynamic spectrum allocation. The proposed dynamic spectrum auctions allocate access rights at a much more detailed geographic locality, exposing more dramatically the interference problem described above, and would be reallocated with far greater frequency. This shift toward frequent auctions of a large number of small parcels moves the auction design problem from one where information revelation is the driving concern and slow ascending auctions are an appropriate solution, to one where simplicity and speed are the driving concerns and sealed-bid auctions like the ones we construct in this paper are appropriate.

The engineering concerns of simplicity and speed are not unique to dynamic spectrum auctions. Other examples from the literature include advertising auctions (Varian 2007; Edelman, Ostrovsky, and Schwarz 2007) and auctions for cloud computing services (Lin, Lin, and Wei 2010). There may also be opportunities to employ these techniques in traditional markets that are becoming electronic.

Organization.—The remainder of the paper is organized as follows. In Section I we describe the basics of algorithm design and Bayesian mechanism design. In Section II we develop the theory of non-optimal mechanism design. In particular, we describe our basic reduction, an economic approach for constructing a mechanism from any algorithm. In Section III we describe computational methods for practical implementations of our basic reduction. We show that a mechanism can be constructed from an algorithm via a preprocessing step that samples the algorithm's outcome on valuation profiles from the distribution. When the mechanism is executed it simply runs the algorithm on a transformation of the reported valuation profile. A detailed survey of prior work on the interplay between computation and incentives in mechanism design is given in Section IV. Section V concludes with a discussion of extensions of our basic result and follow-up work.

I. Model and Definitions

A. *Algorithms*

We consider algorithms for providing an abstract service in single-dimensional linear agent environments. A single-dimensional linear agent i has private *valuation* v_i, associates with any outcome a level of *allocation* x_i, and has linear value $v_i x_i$ for this allocation. There are n agents, the *valuation profile* is denoted $\mathbf{v} = (v_1, \ldots, v_n)$, and the outcome decomposes across the agents and induces an *allocation profile* $\mathbf{x} = (x_1, \ldots, x_n)$. The service provider has a cost function $c(\cdot)$ over outcomes, which we denote by $c(\mathbf{x})$ for the outcome with allocation profile $\mathbf{x}$. Without loss of generality we normalize $v_i \in [0,1]$ and $x_i \in [0,1]$.

As an illustrative example, consider again the dynamic spectrum allocation problem described in the introduction. In this context we would interpret x_i as the fraction of firm i's towers that are served. For instance, in the example described in Figure 2, any outcome that includes exactly two of firm A's towers induces allocation $x_A = 2/3$. Interference constraints are encoded as a cost function where $c(\mathbf{x}) = 0$ whenever it is possible to select a set of non-interfering towers that induce allocation profile $\mathbf{x}$ and, otherwise, $c(\mathbf{x}) = \infty$. Again with respect to the example of Figure 2, the allocation $(x_A, x_B, x_C, x_D) = (1,1,0,0)$ is infeasible and has cost ∞, whereas the allocation $(2/3, 1, 0, 0)$ is feasible and has cost 0.

An algorithm $\mathcal{A}$ defines an *allocation rule* that maps valuation profiles to allocations, denoted $\mathbf{x}(\mathbf{v})$. Our objective is *social welfare* which we denote as $\mathcal{A}(\mathbf{v}) = \sum_i v_i x_i(\mathbf{v}) - c(\mathbf{x}(\mathbf{v}))$. We allow $\mathcal{A}$ to be randomized in which case $x_i(\mathbf{v})$ is a random variable and $\mathcal{A}(\mathbf{v})$ denotes the expected welfare of the algorithm for valuation profile $\mathbf{v}$. The function $\mathrm{OPT}(\mathbf{v}) = \max_{\mathbf{x}}\{\sum_i v_i x_i - c(\mathbf{x})\}$ will denote the maximum social welfare.

The valuations of the agents are assumed to be drawn from a product distribution $\mathbf{F} = F_1 \times \cdots \times F_n$. Agent i's *cumulative distribution* and *density* functions are denoted F_i and f_i, respectively. The distribution is assumed to be *common knowledge* to the agents and designer.

The pair $(c(\cdot), \mathbf{F})$ defines an environment for single-parameter algorithm design which we will take as implicit. For this environment, the optimal expected welfare is $\mathrm{OPT} = E_{\mathbf{v}\sim\mathbf{F}}[\mathrm{OPT}(\mathbf{v})]$ and the algorithm's expected welfare is $\mathcal{A} = E_{\mathbf{v}\sim\mathbf{F}}[\mathcal{A}(\mathbf{v})]$.

B. *Mechanisms*

A mechanism $\mathcal{M}$ consists of an *allocation rule* and a *payment rule*. We denote by $\mathbf{x}(\mathbf{v})$ and $\mathbf{p}(\mathbf{v})$ the allocation and payment rule of an implicit mechanism $\mathcal{M}$. We assume agents are *risk neutral* and individually desire to maximize their expected *utilities*. Agent i's utility for allocation $\mathbf{x}$ and payments $\mathbf{p}$ is $v_i x_i - p_i$. We consider single-round, sealed-bid mechanisms where agents simultaneously bid and the mechanism then computes the allocation and payments.

Our goal is to construct a mechanism that generates outcomes with high social welfare in equilibrium. We will use the standard notion of *Bayes-Nash equilibrium* (BNE) as the equilibrium concept for our game of incomplete information. The revelation principle says that any equilibrium that is implementable in BNE is implementable with truth-telling as the BNE strategies of the agents. That is, an agent that believes the other agents are reporting their values truthfully as given by the distribution has a best response of also reporting truthfully. A mechanism with truth-telling as a BNE is said to be *Bayesian incentive compatible* (BIC).

It will be useful to consider agent i's expected payment and probability of allocation conditioned on her value. To this end, denote the *interim payment rule* by $p_i(v_i) = E_{\mathbf{v},\mathcal{M}}[p_i(\mathbf{v}) \mid v_i]$ and the *interim allocation rule* by $x_i(v_i) = E_{\mathbf{v},\mathcal{M}}[x_i(\mathbf{v}) \mid v_i]$. The following theorem characterizes BIC mechanisms and motivates the following definition.

THEOREM 1 (Myerson 1981): *A mechanism is BIC if and only if for all agents i:*

- $x_i(v_i)$ *is monotone non-decreasing, and*
- $p_i(v_i) = v_i x_i(v_i) - \int_0^{v_i} x_i(z)\,dz + p_i(0)$.

Usually, $p_i(0)$ is *normalized to be zero.*

DEFINITION 1: *An allocation rule* $\mathbf{x}(\cdot)$ *is* monotone *for distribution* $\mathbf{F}$ *if* $x_i(v_i)$ *is monotone non-decreasing for all i. An algorithm is* monotone *if its allocation rule is monotone.*

From Theorem 1, BIC and monotonicity are equivalent and we will use them interchangeably for both algorithms and mechanisms, though we will prefer "BIC" when the focus is incentive properties and "monotone" when the focus is algorithmic properties.

II. The Reduction

Our goal is to *reduce* the problem of designing a good mechanism to the problem of designing a good algorithm. In other words, given an algorithm, we wish to construct from it a mechanism with comparable performance. A key idea is to consider only constructed mechanisms of a specific form: reported values are to be independently transformed, the transformed values are input to the algorithm, and the algorithm's outcome is implemented. If each agent's transformation is stationary with respect to the distribution (i.e., the distribution of transformed values is equal to the original value distribution) then the mechanism design problem is greatly simplified. First, the expected cost of the outcome produced by the mechanism is identical to the expected cost of the algorithm. Second, the details of the transformation for one agent do not affect the expected outcome of the algorithm for the other agents. The problem is therefore reduced to independently finding good transformations for each agent. Notice that the outcome of the mechanism is always simply the outcome from a single call to the algorithm.

In the sections to follow we first describe a class of interval resampling transformations: whenever a value lies within one of a set of given intervals, the transformed value is a new random draw from the same interval. Such a transformation is stationary. We then solve the problem of finding the best intervals for resampling (in terms of welfare maximization) and show that resampling on these intervals indeed results in a monotone allocation rule and preserves expected welfare. Finally, we construct a mechanism by transforming each agent's value by the optimal interval resampling transformation and running the algorithm on the transformed valuation profile.

A. *Interval Resampling Transformations*

Consider an agent, say Alice, with value v drawn from distribution F. (We will drop the typical subscript i from our notation in this section, since we will be focusing on this single agent.) Suppose Alice faces interim allocation rule $x(\cdot)$ (when other agents' values are drawn from the distribution and directly input into the algorithm). We would like to identify a possibly randomized transformation σ that satisfies the following:

D1. *Monotonizing*: $E_\sigma[x(\sigma(v))]$ is monotone in v,

D2. *Welfare preservation*: social welfare is (weakly) improved, i.e., $E_{v,\sigma}[vx(\sigma(v))] \geq E_v[vx(v)]$, and

D3. *Stationarity*: if $v \sim F$ then $\sigma(v) \sim F$.

The following discussion shows that it is possible to locally fix non-monotonicities (D1) while improving welfare (D2) and maintaining stationarity (D3). Suppose allocation rule $x(\cdot)$ of $\mathcal{A}$ is non-monotone for Alice on some interval $[a,b]$. A simple approach to fixing the non-monotonicity on this interval is to treat her identically whenever her value is on this interval. To ensure stationarity (D3) this must be done by resampling v' from $F[a,b]$, i.e., the distribution F conditioned on $v' \in [a,b]$. The new probability of allocation will be precisely the distribution weighted average of $x(\cdot)$ over the interval $[a,b]$. Moreover, given that the allocation rule was previously non-monotone in this interval, this averaging will shift probability of allocation from lower values to higher values and, thus, improve welfare (D2).

DEFINITION 2: *An* interval resampling transformation σ *corresponding to a set of disjoint intervals* $\mathcal{I} = \{[a_1, b_1], \ldots, [a_k, b_k]\}$ *is defined by*

$$\sigma(v) = \begin{cases} v' \sim F[a_j, b_j] & \textit{if } v \in [a_j, b_j] \textit{ for any } 1 \leq j \leq k, \textit{ and} \\ v & \textit{otherwise.} \end{cases}$$

The interval resampling of allocation rule x on $\mathcal{I}$ *gives allocation rule*

$$x^{\sigma}(v) = \begin{cases} E_{v' \sim F[a_j, b_j]}[x(v')] & \textit{if } v \in [a_j, b_j] \textit{ for any } 1 \leq j \leq k, \textit{ and} \\ x(v) & \textit{otherwise.} \end{cases}$$

By the above discussion and definition we have the following proposition.

PROPOSITION 1: *Interval resampling transformations are stationary* (D3).

An interval resampling applied to an allocation rule can fix local non-monotonicities and locally improve welfare. There are obviously transformations that globally fix non-monotonicities, e.g., by resampling the whole interval $[0, 1]$; and others that globally preserve welfare, e.g., by doing nothing. In what remains of this section we will show that the welfare-optimal resampling transformation is simultaneously monotonizing and welfare preserving.

B. *Cumulative Allocation Rules and Expected Welfare*

Optimization over interval resampling transformations can be easily understood if we consider allocation rules in *quantile space* instead of value space. An agent, e.g., Alice, with value v drawn from distribution F has quantile q corresponding to the probability that she is weaker (i.e., has a lower value) than a random draw from the distribution. Formulaically, $q = 1 - F(v)$. Note that Alice's quantile q is distributed uniformly on $[0, 1]$. The value function that maps quantiles back to values is $\nu(q) = F^{-1}(1 - q)$.

We will denote the *allocation rule in quantile space* and the *cumulative allocation rule* as follows:

- let $y(q) = x(\nu(q))$ be the allocation rule in quantile space.
- let $Y(q) = \int_0^q y(z)\, dz$ be the cumulative allocation rule.

Notice that monotonicity (non-decreasing) of $x(\cdot)$ is equivalent to monotonicity (non-increasing) of $y(\cdot)$ which is equivalent to concavity of $Y(\cdot)$.

The remainder of the analysis will be with respect to an abstract value function $\nu(\cdot)$ which maps Alice's quantile to the objective to be optimized. The welfare objective corresponds to $\nu(q) = F^{-1}(1 - q)$, and in Section IIE we show how to capture the profit objective by using a different value function. For the remainder of this section, statements about welfare will be with respect to this abstract value function. We require only that the value function be monotonically non-increasing in quantile, i.e., stronger quantiles correspond to higher objective value. The optimal interval resampling transformation will not depend on the value function, so it will therefore simultaneously improve welfare with respect to all such value functions.

We can write the welfare of an allocation rule in terms of the cumulative allocation rule $Y(\,\cdot\,)$ as follows.

LEMMA 1: *The expected welfare of any allocation rule $y(\,\cdot\,)$ for distribution F in terms of its cumulative allocation rule $Y(\,\cdot\,)$ and value function $\nu(\,\cdot\,)$ is*

$$E_q[\nu(q)\,y(q)] \;=\; \nu(1)Y(1) + E_q[-\nu'(q)Y(q)]$$

for $q \sim U[0,1]$ and $\nu'(q) = \frac{d}{dq}\nu(q)$.

PROOF:

Consider the following sequence of equalities:

$$\begin{aligned} E_q[\nu(q)\,y(q)] &= \int_0^1 \nu(q)\,y(q)\,dq \\ &= [\nu(q)Y(q)]_0^1 - \int_0^1 \nu'(q)Y(q)\,dq \\ &= \nu(1)Y(1) \;+\; \int_0^1 \nu'(q)Y(q)\,dq \\ &= \nu(1)Y(1) \;+\; E_q[-\nu'(q)Y(q)]. \end{aligned}$$

The first and fourth equalities are from the definition of expectation and the fact that the quantile of a random value is always distributed uniformly on $[0,1]$. The second equality is integration by parts. The third equality is because $Y(0) = 0$. ∎

Notice that in the above lemma the term $-\nu'(q)$ is always non-negative as Alice's value is non-increasing in her quantile. Therefore, if we are comparing two allocation rules and one "is above" the other, then its expected welfare is larger than the other's as well. This observation is formalized in the definition and lemma below.

DEFINITION 3: *Allocation rule x^a* majorizes *x^b if their cumulative allocation rules Y^a and Y^b satisfy $Y^a(q) \geq Y^b(q)$ for all quantiles $q \in [0,1]$ and $Y^a(1) = Y^b(1)$.*

LEMMA 2: *For any monotone value function $\nu(\,\cdot\,)$, if allocation rule x^a majorizes x^b then the expected welfare of x^a is at least that of x^b.*

PROOF:

The valuation function $\nu(\,\cdot\,)$ is non-increasing; therefore, its derivative $\nu'(\,\cdot\,)$ is non-positive. By the assumption of the lemma and definition of majorization, the cumulative allocation rules satisfy $Y^a(q) \geq Y^b(q)$ for all $q \in [0,1]$, and $Y^a(1) = Y^b(1)$. Therefore, $E_q[-\nu'(q)Y^a(q)] \geq E_q\big[-\nu'(q)Y^b(q)\big]$ and $\nu(1)Y^a(1) = \nu(1)Y^b(1)$. The lemma is proved by summing these equations and invoking Lemma 1. ∎

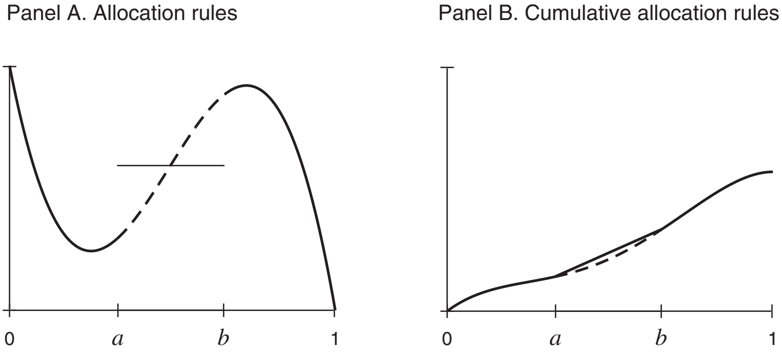


Figure 3

Notes: The allocation rule $y(q)$ (dashed) and $y'(q)$ (solid) constructed by drawing $q' \sim U[a,b]$ and running $y'(q)$ when $q \in [a,b]$. Corresponding cumulative allocations rules $Y(q)$ (dashed) and $Y'(q)$ (solid).

C. *Optimal Interval Selection*

We now consider the effect of resampling some interval $[a,b]$ on the allocation rule (in quantile space, which corresponds to interval $[\nu(b), \nu(a)]$ in value space). Let $x'(\,\cdot\,)$ be $x(\,\cdot\,)$ resampled on interval $[\nu(b), \nu(a)]$, and consider the corresponding curves $y'(\,\cdot\,)$ and $Y'(\,\cdot\,)$. This resampling transformation corresponds to replacing $y(\,\cdot\,)$ with its average on $[a,b]$, or equivalently $Y(\,\cdot\,)$ with the line segment connecting $(a, Y(a))$ to $(b, Y(b))$ (see Figure 3).

With this line-segment interpretation of resampling, it is possible to give resampling intervals to obtain a cumulative allocation rule equal to the *concave hull* of the given cumulative allocation rule, i.e., the pointwise smallest concave function lying above the allocation rule (see Figure 4).

DEFINITION 4: *The* ironed allocation rule $\bar{x}$ *for x corresponds to the concave hull $\bar{Y}$ of Y; the* ironed intervals *for x are the intervals where $\bar{Y}$ and Y are distinct; the* ironing transformation *is the interval resampling transformation corresponding to the ironed intervals.*

LEMMA 3: *The ironing transformation for x with distribution F is its welfare-optimal resampling transformation for any monotone value function.*

PROOF:

By definition, this resampling transformation has ironed allocation rule $\bar{x}$ and cumulative allocation rule $\bar{Y}$. Consider any allocation rule x' that can be obtained from x by a resampling transformation, say with cumulative allocation rule Y'. Then each point on the curve Y' is a convex combination of points on the curve Y. Since $\bar{Y}$ is the convex hull of the region below the curve Y, it follows that each point $(a, Y'(a))$ lies below the curve $\bar{Y}$. The allocation rule $\bar{x}$ therefore majorizes x', and hence majorizes any allocation rule that can be obtained from x by resampling transformations. The proof then follows from Lemma 2. ■

We conclude with the main theorem of the section.

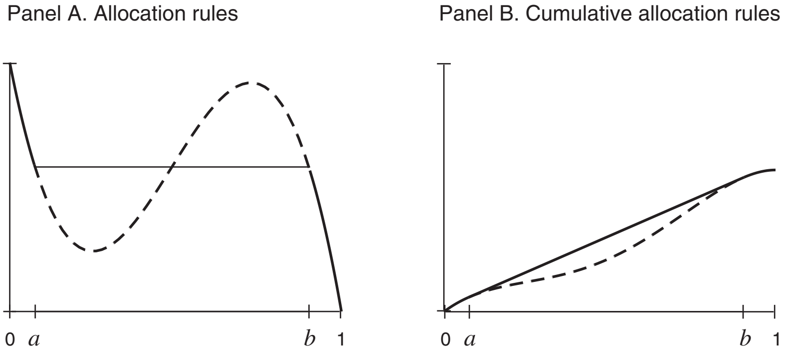


FIGURE 4

Notes: The allocation rule $\bar{y}(q)$ (solid) and cumulative allocation rule $\overline{Y}(q)$ (solid) constructed by taking the concave hull of $Y(q)$ (dashed). Interval $I = [a,b]$ is defined as $\{q\colon Y(q) \neq Y(q)\}$.

THEOREM 2: *The ironing transformation is monotonizing* (D1) *and welfare preserving* (D2).

PROOF:

By definition, $\bar{x}(\cdot)$ majorizes $x(\cdot)$, so by Lemma 2 this transformation weakly improves welfare (D2). Moreover, $\bar{x}(\cdot)$ is monotone (D1) because its cumulative allocation rule $\overline{Y}(\cdot)$ is concave. ∎

D. *The Ironed Algorithm*

The techniques discussed above can be used to construct a transformation σ_i for each agent i that is monotonizing, welfare-preserving, and stationary.

DEFINITION 5: *The* ironed algorithm $\overline{\mathcal{A}}$ *for distribution F is the composition of the algorithm $\mathcal{A}$ with its ironing transformations* $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_n)$.

The ironed algorithm can be implemented as a preprocessing stage where the ironing transformation of each agent is calculated; and an execution stage where the agents report their values, the values are transformed, the algorithm is run on the transformed values, and the outcome of the algorithm is implemented with payments from the payment identity of Theorem 1.

THEOREM 3: *The ironed algorithm $\overline{\mathcal{A}}$ for $\mathcal{A}$ is BIC and has welfare satisfying $\overline{\mathcal{A}} \geq \mathcal{A}$.*

PROOF:

Fix the single dimensional environment, i.e., cost function $c(\cdot)$ and distribution $\mathbf{F}$. For agent i, let x_i be the allocation rule of the algorithm and $\bar{x}_i$ be the ironed allocation that results from composition of the allocation rule with the ironing transformation σ_i. By Proposition 1 both v_i and $\sigma_i(v_i)$ are distributed as F_i. Therefore, transforming i's value does not affect the algorithm's allocation rules for the other agents, nor does it affect the distribution over outcomes $\mathbf{x}(\boldsymbol{\sigma}(\mathbf{v}))$ produced by the mechanism. The same holds for all agents, so the expected cost of $\overline{\mathcal{A}}$ is the same as $\mathcal{A}$, i.e., $E_{\mathcal{A},\mathbf{v},\boldsymbol{\sigma}}[c(\mathbf{x}(\boldsymbol{\sigma}(\mathbf{v})))] = E_{\mathcal{A},\mathbf{v}}[c(\mathbf{x}(\mathbf{v}))]$. By Theorem 2, each agent has weakly improved welfare. For example, for the social welfare objective (corresponding to $\nu(q) = F^{-1}(1-q)$), we have $E_{v_i}[v_i \bar{x}_i(v_i)] = E_{v_i}[v_i x_i(\sigma_i(v_i))] \geq E_{v_i}[v_i x_i(v_i)]$ for all i. Combining the expected welfare of the agents with the expected costs we have $E_{\overline{\mathcal{A}},\mathbf{v}}[\overline{\mathcal{A}}(\mathbf{v})] \geq E_{\mathcal{A},\mathbf{v}}[\mathcal{A}(\mathbf{v})]$. ■

E. *Profit*

The methodology we have developed for constructing good mechanisms from good algorithms can also be applied to the profit objective. The profit objective is the total payments minus the cost of the allocation produced, i.e., for the outcome $(\mathbf{x}, \mathbf{p})$ the profit is $\sum_i p_i - c(\mathbf{x})$. Myerson (1981) derived a mapping from values to virtual values and proved that in expectation the profit of any incentive compatible mechanism is equal to its virtual welfare, i.e., the sum of the virtual values of the agents served less the service cost. Virtual welfare does not correspond to profit out of equilibrium, e.g., for a non-monotone algorithm; however, it does give a yardstick by which to tune non-optimal algorithms. We will show that under some mild distributional assumptions, described below, the ironed algorithm of Definition 5, which is incentive compatible, also weakly improves virtual welfare. Thus, it obtains at least the profit suggested by the yardstick of virtual welfare.

DEFINITION 6 (Myerson 1981): *The* virtual value *of an agent with value v drawn from distribution F is $\varphi(v) = v - \frac{1-F(v)}{f(v)}$. The* virtual welfare *of an allocation rule $x(\cdot)$ is $E_v[\varphi(v)x(v)]$. The distribution F is regular if $\varphi(\cdot)$ is monotone non-decreasing.*

LEMMA 4 (Myerson 1981): *The expected payment of an agent with value v drawn from distribution F in a mechanism with monotone allocation rule $x(\cdot)$ is equal to the expected virtual welfare, i.e., $E[p(v)] = E[\varphi(v)x(v)]$.*

The corollary of Theorem 3 below shows that the ironed algorithm (Definition 5) preserves profit in environments where agents' values are drawn from regular distributions. Many standard distributions are regular, e.g., uniform, normal, and exponential distributions; bimodal distributions are not generally regular. In Section A of the online Appendix we give a reduction that preserves profit for general, possibly irregular, distributions.

COROLLARY 1: *For agents with values drawn from regular distributions, the expected profit of the ironed algorithm $\overline{\mathcal{A}}$ is at least the expected virtual welfare of the algorithm $\mathcal{A}$.*

PROOF:

Theorem 3 shows that the ironed algorithm preserves welfare with respect to the monotone value function $\nu(\cdot)$. This corollary follows by considering virtual value function $\nu(q) = \varphi(F^{-1}(1-q))$ rather than value function $\nu(q) = F^{-1}(1-q)$;

this invocation of the theorem is valid because regularity of the distribution F implies that the virtual value function $\nu(\cdot)$ is monotone. ■

It should be noted that the ironed algorithms of Theorem 3 and Corollary 1 are identical. In other words, there is a single transformation that takes any non-monotone algorithm, makes it monotone, and simultaneously preserves both its welfare and its virtual welfare (assuming the value distributions are regular) which corresponds to profit.

COROLLARY 2: *The ironed algorithm $\overline{\mathcal{A}}$ for $\mathcal{A}$ is BIC and weakly improves welfare simultaneously for* every *monotone value function $\nu(\cdot)$. In particular, $\overline{\mathcal{A}}$ weakly improves both the welfare and (for regular distributions) the virtual welfare of $\mathcal{A}$.*

III. Computation

Our primary motivation for designing optimal resampling transformations is to develop a systematic method for converting algorithms into incentive compatible mechanisms without inflating the computational time needed to run the mechanism. We now return to this motivation explicitly and establish both that the reduction can be performed quickly and that the resulting mechanism imposes only modest computational overhead upon the original algorithmic solution. As a warm-up we will show that optimal resampling transformations can be found and executed quickly under an idealized (but unrealistic) computational model. We then present a more realistic "black-box" model, and show how our idealized implementation can be modified to work in this model as well.

A. *Background: Quantifying Computation*

To analyze computational requirements we will adopt the standard convention from computer science of bounding the number of *basic steps* required to perform tasks. Such basic steps include arithmetic on scalar values, referencing a value stored from a previous calculation, etc. Our goal will be to demonstrate that the number of steps needed to implement the mechanism scales well as the problem instance grows large. The standard distinction between scaling well and scaling badly is whether the runtime is a polynomial or super-polynomial function of the input size, respectively.[3] Our construction will take a linear number of steps, which scales well.[4]

For computational mechanism design problems, the computational burden can be partitioned into a *preprocessing* component and an *execution* component. The preprocessing component is essentially the design of the mechanism. We assume that we have access to the prior distribution of preferences and the algorithm, and, e.g., the algorithm can be run on samples from the distribution. The execution component

[3] Notice that if the runtime $T(n)$ of an algorithm in terms of the size n of the input is a polynomial like n^k (where k is a low constant) then if the input gets a constant c factor bigger, the runtime gets only a constant $d = c^k$ factor longer. For a linear time algorithm, the constants are the same.

[4] Since it takes linear time to inspect the input, this is essentially the best possible.

then implements the designed mechanism: agents arrive, submit bids, and the mechanism determines an outcome. In such a decomposition, the preprocessing phase is used to construct data structures (e.g., lists of intervals upon which to resample) that can then be accessed during the execution component. The running time of both phases should scale well, but it is especially important that the execution phase take as few steps as possible, since the work done in the preprocessing phase could potentially be reused across many executions of the mechanism.

B. *Warm-Up: Computation in an Idealized Model*

To provide some intuition for the computational requirements of the reduction from Section II, we first consider an idealized setting. In this ideal scenario, the interim allocation rules of the original algorithm $\mathcal{A}$ are piecewise constant, with $k \geq 1$ pieces of equal width. Moreover, these allocation rules are provided explicitly in an $n \times k$ matrix of allocation quantities.

In this ideal scenario, simple numerical calculus operations can be performed on the interim allocation rules, such as integration, inversion, and construction of a convex hull. The number of basic steps needed for any such operation scales linearly with k. Constructing the welfare-optimal resampling transformations $\boldsymbol{\sigma}$, as described in Section II, therefore requires an execution time that grows linearly with $n \cdot k$, the size of the input matrix. Appropriate payments can be precomputed in a similar number of steps via integration operations on the transformed allocation rules, as specified by Theorem 1. This completes the preprocessing phase. Given this construction, executing the mechanism is straightforward. The overhead necessary to execute the ironed algorithm $\overline{\mathcal{A}}$, given declared values $\mathbf{v}$, is that of determining which interval of σ_i contains value v_i, for each i. This can easily be done in a constant number of basic steps per agent by using a table that maps each of the k pieces of value space to an interval of σ_i. Optimal resampling transformations can thus be found and implemented efficiently.

C. *The Black-Box Model*

It is unrealistic, but assumed by the idealized model, that the interim allocation rule is known exactly.[5] In the more realistic, computationally rigorous, and standard (in computer science) *black-box model* of computation, the algorithm and value distributions are not known in advance to the mechanism designer. Instead, the algorithm can be run on any input, conditional samples can be drawn from the distribution of agent values, and the cumulative distribution function can be evaluated at any value. We will show that even in this weak model, it is still possible to approximately implement the reduction from Section II with a small number of basic computational steps and a small number of queries of the algorithm and distributions.

The approach will be to estimate the interim allocation rule by sampling. For any value of an agent, values of other agents can be sampled, and the algorithm can be simulated on the sampled valuation profile. Repeating this process gives an estimate

[5] Exactly computing the probability with which an algorithm serves a given agent when the valuation profile is drawn from a product distribution is not generally tractable, see Hartline, Kleinberg, and Malekian (2015).

with accuracy that scales inversely proportional to the square root of the number of simulations. The questions considered in the remainder of this section are (i) the analysis question of how the inevitable errors of this method manifest in loss of performance and loss of incentive compatibility, and (ii) the design question of whether a more sophisticated approach can remove the loss of incentive compatibility completely (at potentially a greater loss in performance). We ultimately obtain the following result:

THEOREM 4: *For an arbitrary algorithm $\mathcal{A}$, there is an algorithm $\tilde{\mathcal{A}}$ that is monotone, satisfies $\tilde{\mathcal{A}} \geq \mathcal{A} - \epsilon$, and can be computed using a polynomial* (*in n and* $1/\epsilon$) *number of basic operations and calls to algorithm $\mathcal{A}$.*

The full proof of Theorem 4 appears in Section C of the online Appendix accompanying this paper. The remainder of this section describes a weaker version of this theorem: a (black-box) construction that converts an arbitrary approximation algorithm into a mechanism that is *approximately* Bayesian incentive compatible, in the following sense.

DEFINITION 7: *A mechanism $\mathcal{M}$ is ϵ-BIC if truth-telling is an ϵ-equilibrium, in expectation over agent values $\mathbf{v} \sim \mathbf{F}$ and any randomization of $\mathcal{M}$. That is, for each agent i with value v_i and every possible report v_i',*

$$E_{\mathbf{v},\mathcal{M}}[v_i \cdot x_i(\mathbf{v}) - p_i(\mathbf{v}) \mid v_i] \;\geq\; E_{\mathbf{v},\mathcal{M}}[v_i \cdot x_i(v_i', \mathbf{v}_{-i}) - p_i(v_i', \mathbf{v}_{-i}) \mid v_i].$$

THEOREM 5: *For an arbitrary algorithm $\mathcal{A}$, there is an ϵ-BIC algorithm $\overline{\mathcal{A}}$ with $\overline{\mathcal{A}} \geq \mathcal{A} - \epsilon$ that can be computed using a polynomial* (*in n and* $1/\epsilon$) *number of basic operations and calls to algorithm $\mathcal{A}$.*

We present the proof of Theorem 5 in the next section. The proof of Theorem 4, which appears in Section C of the online Appendix, builds directly upon Theorem 5 by showing how to convert the ϵ-BIC mechanism from the statement of Theorem 5 into a fully BIC mechanism.

D. *Approximate Incentive Compatibility in a Black-Box Model*

In this section we describe the proof of Theorem 5. Recall that our strategy for implementing the optimal resampling transformation will be to estimate the interim allocation rules of $\mathcal{A}$, via sampling, then apply the implementation from the idealized model to these estimated allocation rules. For any agent i and interval $I = [a,b]$, $E_{w \sim F[a,b]}[x(w)]$ can be estimated by running the algorithm on many samples of valuation profiles $\mathbf{v} \sim \mathbf{F}$ conditioned on $v_i \in I$ and averaging. In particular, for any $\epsilon > 0$, standard concentration bounds imply that approximately $1/\epsilon^2$ samples are sufficient to reduce the expected error in the estimate to within an additive ϵ.[6]

[6] An extra logarithmic factor is required to guarantee that these error bounds hold with high probability, but for clarity these details are deferred to the formal proof for correctness in Section B of the online Appendix.

Our approximate reduction is as follows. Given error parameters $\epsilon > 0$ and $\delta > 0$, we first partition the space of values $[0, 1]$ into $k = 1/\delta$ intervals of width δ. For each agent and each interval we estimate the allocation probability to within an additive error of at most ϵ. From the resulting $n \times k$ table we apply the reduction from the idealized model of computation to obtain the resampling transformation $\boldsymbol{\sigma}$ and algorithm $\overline{\mathcal{A}}$. Of course, this reduction is being applied to an approximation of the interim allocation rules, rather than the true allocation rules. The rest of this section bounds the impact of the approximation on incentives and welfare.

Our first source of error is due to equating (by ironing together) values within intervals of width δ. This discretization of the valuation space reduces overall social welfare by at most $n \cdot \delta$, since there are n agents and the loss from an agent i is bounded by the effect of replacing each agent i's value v_i with $v_i - \delta$.

We next bound the effect of misestimation of each allocation rule on each interval. It will be useful to think of the estimates for an allocation rule x as defining a new allocation rule that approximates x, in the sense that it differs from x by at most ϵ, pointwise.

DEFINITION 8 (ϵ-close): *Two allocation rules x and $\tilde{x}$ are ϵ-close if $|x(v) - \tilde{x}(v)| < \epsilon$ for all v.*

Importantly, composition with an interval resampling transformation does not increase additive errors between allocation rules.

LEMMA 5: *If allocation rules x and $\tilde{x}$ are ϵ-close then, for any interval resampling transformation σ, $x \circ \sigma$ and $\tilde{x} \circ \sigma$ are ϵ-close.*

PROOF:

For any v not in an interval where σ resamples, x and $\tilde{x}$ are ϵ-close. For any v within an interval where σ resamples, the values of both x and $\tilde{x}$ are subject to a weighted average; however, x and $\tilde{x}$ are ϵ-close pointwise so their weighted average is also ϵ-close. ∎

If we write $\tilde{x}$ for the estimated allocation rule obtained when sampling x, then the resampling transformation σ chosen by our reduction will be welfare-optimal and monotonizing for $\tilde{x}$. We can therefore conclude that $x \circ \sigma$ (the transformation composed with the true allocation rule) is ϵ-close to a monotone allocation rule, and is thus approximately monotone. We call an allocation rule x *ϵ-monotone* if $x(v) \geq x(w) - \epsilon$ for all $w < v$.

LEMMA 6: *If an allocation rule x is ϵ-close to a monotone allocation rule $\tilde{x}$ then it is 2ϵ-monotone.*

PROOF:

The proof follows as, for all $w < v$,

$$x(v) \geq \tilde{x}(v) - \epsilon \geq \tilde{x}(w) - \epsilon \geq x(w) - 2\epsilon,$$

where the first and last inequalities are from ϵ-closeness and the second is from monotonicity of $\tilde{x}$. ∎

As observed above, $x \circ \sigma$ is ϵ-close to $\tilde{x} \circ \sigma$ which dominates $\tilde{x}$ which is itself ϵ-close to x. Since curves that are ϵ-close cannot differ in expected welfare by more than ϵ, this implies that the interval resampling transformation σ is approximately welfare-preserving for x.

LEMMA 7: *If x and $\tilde{x}$ are ϵ-close, and σ is welfare-preserving for $\tilde{x}$, then $E_{v,\sigma}[v \cdot x(\sigma(v))] \geq E_v[v \cdot x(v)] - 2\epsilon$.*

PROOF:

Since x and $\tilde{x}$ are ϵ-close, as are $x \circ \sigma$ and $\tilde{x} \circ \sigma$, we have

$$E_{v,\sigma}[v \cdot x(\sigma(v))] \geq E_{v,\sigma}[v \cdot \tilde{x}(\sigma(v))] - \epsilon \geq E_{v,\sigma}[v \cdot \tilde{x}(v)] - \epsilon \geq E_{v,\sigma}[v \cdot x(v)] - 2\epsilon;$$

the second inequality is due to σ being welfare-preserving for $\tilde{x}$. ∎

If we apply this approximate reduction to all agents, we find a set of resampling transformations that approximately satisfies the conditions of the welfare-optimal resampling transformations. Furthermore, as in the ideal setting, we can precompute payments using our estimates for the allocation rule. Such payments might not satisfy the requirements of Theorem 1 for the true allocation rule, but since they are calculated with respect to an allocation rule that approximates it, the resulting mechanism will be BIC in the approximate sense (Definition 7).

To summarize, given an algorithm $\mathcal{A}$, we can construct approximately welfare-optimal transformations $\boldsymbol{\sigma}$, where the approximation is with respect to error parameters ϵ and δ. These transformations can then be composed with the algorithm $\mathcal{A}$ to obtain an approximately monotone algorithm. Write $\overline{\mathcal{A}}$ for this approximately monotone algorithm. From our discussion above, $\overline{\mathcal{A}}$ is 2ϵ-BIC and has welfare satisfying $\overline{\mathcal{A}} \geq \mathcal{A} - (2\epsilon + \delta)n$, in expectation over randomization in the construction of $\overline{\mathcal{A}}$. Moreover, as we have argued, this construction process can be performed in a polynomial (in n, $1/\epsilon$, and $1/\delta$) number of basic operations and calls to $\mathcal{A}$. An appropriate choice of parameters thus concludes the proof of Theorem 5. A detailed derivation of the appropriate parameters can be found in Section B of the online Appendix.

The total computation time (preprocessing plus execution) is polynomial, and the execution time of algorithm $\overline{\mathcal{A}}$ from Theorem 5 scales especially well. Execution involves determining which resampling interval of σ_i contains value v_i, for each i. Due to the manner in which we discretized the valuation space, this operation can be performed trivially: one can simply round the value of v_i to a multiple of δ and consult a precomputed table indicating the appropriate distribution from which the resampled input should be drawn. The execution of the mechanism thus imposes effectively no overhead beyond the requirements of the original algorithm $\mathcal{A}$. The transformation of input values can be done in a constant number of basic operations per agent, which is the degree of overhead necessary to simply read the declared valuation profile.

We emphasize that since the allocation rules of $\mathcal{A}$ are learned only via sampling, our construction does not require any prior information about how $\mathcal{A}$ behaves, or about the cost function $c(\,\cdot\,)$ that defines the optimization problem being solved by $\mathcal{A}$. The reduction is general in the sense that a single implementation of this transformation applies to arbitrary algorithms and single-dimensional agent environments.

IV. Related Work

As described in the introduction, the Vickrey-Clarke-Groves (VCG) mechanism for maximizing social welfare can be viewed as a reduction from the mechanism design problem of optimizing social welfare to the analogous optimization problem (Vickrey 1961; Clarke 1971; Groves 1973). Similarly, in single-dimensional settings Myerson's (1981) virtual-value-based approach reduces the mechanism design problem of maximizing profit to, indirectly, the analogous social welfare optimization problem. These reductions imply that an optimal mechanism can be constructed from an optimal algorithm. Our reduction, which constructs a mechanism from any algorithm (even a non-optimal one) relies heavily on Myerson's construction and analysis.

There is a significant literature, primarily in computer science, that considers engineering issues, such as computational tractability, in mechanism design. This literature can be roughly seen in two classes. The first class considers paradigmatic problems such as combinatorial auctions (see e.g., Cramton, Shoham, and Steinberg 2006) and gives novel mechanisms that simultaneously address computational tractability and incentive compatibility requirements. The second class considers algorithms with special properties and gives methods for constructing mechanisms from these special algorithms. Our work relates more to the second line of work and we survey it and the connection below.

Greedy algorithms, which in the context of mechanism design consider the agents sorted by some criteria and make irrevocable decisions on whether to serve each agent at the time she is considered, are known to obtain a large fraction of the optimal welfare in many settings; see, e.g., Wan (2013) for a recent survey. Lehmann, O'Callaghan, and Shoham (2002) study the construction of mechanisms from a family of greedy algorithms. For all algorithms in this family there exist payments that align incentives. With respect to our discussion of dynamic spectrum allocation in the introduction, the greedy algorithm of Jia et al. (2009) is in the Lehmann, O'Callaghan, and Shoham (2002) family while that of Zhou et al. (2008) is not.

A second common paradigm from approximation algorithms is that of *linear program rounding*; see, e.g., Motwani and Raghavan (1995). The intractable non-game-theoretic optimization problem corresponding to many mechanism design problems can be expressed as a (linear) *integer program*, i.e., one where variables are required to take on integral values. The relaxed program, where the integrality requirement is dropped, is a linear program and is tractable to solve. Relaxing integrality allows for a better solution to be found, but this solution generally has fractional variables where integral ones are required. The *integrality gap* quantifies how much better the optimal solution to the relaxation is to the optimal integral solution (in worst case). Lavi and Swamy (2011) identify a property of *rounding methods* that allow an integral solution to be found that is worse than the optimal fractional solution by exactly the integrality gap (in all cases). They show that an incentive compatible mechanism can easily be constructed from such an *linear program rounding* algorithm.

An *approximation scheme* is an algorithm that can be parameterized by any *loss tolerance* ϵ and always finds an outcome with welfare that at least a $1 - \epsilon$ fraction of the optimal welfare; see, e.g., Vazirani (2001). Importantly, smaller loss tolerance ϵ generally requires more computation. Dughmi and Roughgarden (2014) show that an approximation scheme for the algorithmic problem can always be converted into an approximation scheme for the mechanism design problem.

The results above all require algorithms with special properties. The following two results make no assumptions on the algorithm except that it has a worst-case performance guarantee, but they can be only applied in restricted settings. Babaioff, Lavi, and Pavlov (2009) give a general construction for converting any worst-case approximation algorithm for combinatorial auctions into a mechanism with welfare that is worse than the algorithm's guaranteed welfare by at most a logarithmic factor in the ratio between the highest and lowest agent valuation. Their approach is to build an ascending auction out of the algorithm. The loss per round of the auction is bounded by the algorithm's approximation factor and the number of rounds is bounded by the aforementioned logarithm. For the symmetric special case where all agents' preferences for a bundle of items are given by the product of a common value for the bundle and the agent's private value, (Huang, Wang, and Zhou 2011) give a method for converting any algorithm into a mechanism with welfare that is at least a $1 - \epsilon$ fraction of the algorithm's guaranteed welfare for any ϵ.

The mechanisms of the preceding literature review are all dominant strategy incentive compatible. Our work differs in that (i) our mechanisms are Bayesian incentive compatible (i.e., truth-telling is a BNE not a DSE); (ii) we do not restrict to any specific environment for mechanism design; and (iii) we do not require the optimization algorithm to satisfy special properties.

There is also a recent line of work that looks at designing mechanisms with only good Nash or Bayes-Nash equilibria. Most of these works focus on specific settings or mechanisms, we briefly survey the few that give general methods for constructing mechanisms with good equilibria from good algorithms. Lucier and Borodin (2010) give a complementary theory to that of Lehmann, O'Callaghan, and Shoham (2002) by showing that greedy algorithms with natural payment rules, e.g., pay your bid, generally give mechanisms with Nash and Bayes-Nash equilibria that satisfy almost the same worst-case guarantees that the algorithm satisfies. Part of their analysis has recently been improved by Syrgkanis and Tardos (2013). A different approach for constructing Bayes-Nash mechanisms for combinatorial auctions is to restrict the message space of the mechanism to a class for which the optimization problem is computationally easy, then employ an optimal algorithm for the reduced space. This method was employed by Dütting, Henzinger, and Starnberger (2013) and Babaioff et al. (2014) in the context of complement-free combinatorial auctions. Our results differ from these works in that our mechanisms make no assumptions on the algorithm or restrictions to the setting (and that the Bayes-Nash equilibrium we analyze for our mechanisms is the truth-telling one).

V. Conclusions and Extensions

We conclude by reviewing a number of extensions of our reduction. In particular, we discuss non-linear objectives, multi-dimensional agent values, dominant strategy incentive compatibility, and more sophisticated techniques for reducing the welfare loss in our black-box reduction.

Our reduction (in the black-box model of computation) incurs an additive loss to the expected social welfare, scaled with respect to the support of the agent value distributions. In circumstances where the expectations of the agent values are much smaller than their maximum possible values, it may be preferable to suffer a multiplicative loss to expected social welfare. That is, an approximation mechanism with expected social welfare within a $(1 + \epsilon)$ fraction of the welfare of the original algorithm. Our bounds can be extended to obtain this sort of multiplicative approximation. This straightforward but technically cumbersome extension follows primarily by normalizing the (additive) errors in the reduction in Section III by the maximum of any agent's expected valuation. Details of this construction can be found in the appendix of Hartline and Lucier (2009).

In the statement of Theorem 5, the approximate incentive compatibility of the ironed mechanism holds in expectation over randomness in the samples used in the mechanism's construction. One might therefore be concerned that its incentive guarantees do not hold if these random samples are (partially) observed by the participants, e.g., via repeated executions of the mechanism. A more careful analysis, described in Section B of the online Appendix, shows that the ironed mechanism is ϵ-BIC with probability $1 - \epsilon$, even if one were to reveal the samples used in the mechanism's construction.

Our method of ironing allocation rules is identical to Myerson's (1981) method for ironing virtual value functions, a step in his construction of revenue optimal auctions. The revenue optimal auction is the one that maximizes virtual welfare (the cumulative virtual values of the winners of the auction less the service cost) subject to monotonicity of the allocation rule. If virtual value functions are non-monotone, then optimizing virtual welfare does not give a monotone allocation rule. Myerson addresses this issue by first ironing the virtual value functions so that they are monotone, then optimizing ironed virtual welfare. This procedure is distinct from optimizing virtual welfare (which results in a non-monotone allocation rule) and then ironing this allocation rule (using our technique). The former is revenue optimal and the latter is not. An example illustrating this distinction is given in Section E of the online Appendix.

Handling non-additively-separable objectives is a fundamental challenge in mechanism design. One such objective, where there has been work on computationally tractable and approximately optimal mechanisms, is *makespan scheduling*. There are m jobs (items) to be scheduled on n machines (agents). Each machine has a private speed (value) and each job has a public length. The amount of time it takes to run a job on a machine is the product of the job length and the machine speed. The mechanism's task is to assign the jobs to machines so as to minimize the maximum working time of any machine. One might ask for a black-box reduction, like the one described in this paper, that converts any scheduling algorithm to a BIC mechanism that preserves expected makespan. In follow-up work, Chawla, Immorlica, and Lucier (2012) show that this is impossible: there is no black-box reduction from mechanism design to algorithm design that preserves expected makespan. On the other hand, Cai, Daskalakis, and Weinberg (2013) show that one can reduce the problem of designing BIC mechanisms for makespan to the problem of designing algorithms for a modified objective: minimizing makespan plus a virtual welfare term.

One distinction between our reduction from (non-optimal) algorithm design to mechanism design and the VCG reduction from optimal algorithm design to optimal mechanism design is that our reduction gives a BIC mechanism whereas the VCG mechanism is DSIC. One might hope that a stronger VCG-like reduction is possible for non-optimal algorithms. The follow-up work of Chawla, Immorlica, and Lucier (2012) gives some evidence to the contrary. They show that there is no reduction from algorithm design to DSIC mechanism design that does not lose a large fraction of the algorithm's welfare in worst-case over valuation profiles and cost functions. The relaxation to expected welfare preservation is an open question.

Our approach for reducing mechanism design to algorithm design is inherently constrained to single-dimensional agent preferences where there is a natural ordering on agent values. For multi-dimensional agent preferences, there is no such ordering. The follow-up work of Hartline, Kleinberg, and Malekian (2015) and Bei and Huang (2011) extends our basic approach to multi-dimensional agent preferences. The relationship between their methods and the construction described in this paper is discussed further in Section G of the online Appendix.

REFERENCES

Al-Ayyoub, Mahmoud, and Himanshu Gupta. 2011. "Truthful Spectrum Auctions with Approximate Revenue." In *INFOCOM, 2011 Proceedings IEEE*, 2813–21. IEEE.

Babaioff, Moshe, Ron Lavi, and Elan Pavlov. 2009. "Single-Value Combinatorial Auctions and Algorithmic Implementation in Undominated Strategies." *Journal of the ACM* 56 (1): 1–32.

Babaioff, Moshe, Brendan Lucier, Noam Nisan, and Renato Paes Leme. 2014. "On the Efficiency of the Walrasian Mechanism." In *Proceedings of the Fifteenth ACM Conference on Economics and Computation*, 783–800. New York: ACM.

Bei, Xiaohui, and Zhiyi Huang. 2011. "Bayesian Incentive Compatibility via Fractional Assignments." In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, 720–33. New York: ACM.

Cai, Yang, Constantinos Daskalakis, and S. Matthew Weinberg. 2013. "Understanding Incentives: Mechanism Design Becomes Algorithm Design." In *Proceedings of the 2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, 618–27. IEEE.

Chawla, Shuchi, Nicole Immorlica, and Brendan Lucier. 2012. "On the Limits of Black-Box Reductions in Mechanism Design." In *Proceedings of the Forty-Fourth Annual ACM Symposium on Theory of Computing*, 435–48. New York: ACM.

Clarke, Edward H. 1971. "Multipart Pricing of Public Goods." *Public Choice* 11 (1): 17–33.

Cramton, Peter. 2002. "Spectrum Auctions." In *Handbook of Telecommunications Economics*, edited by Martin E. Cave, Sumit K. Majumdar, and Ingo Vogelsang, 605–39. Amsterdam: Elsevier Science.

Cramton, Peter, Yoav Shoham, and Richard Steinberg. 2006. *Combinatorial Auctions.* Cambridge, MA: MIT Press.

Dughmi, Shaddin, and Tim Roughgarden. 2014. "Black-Box Randomized Reductions in Algorithmic Mechanism Design." *SIAM Journal on Computing* 43 (1): 312–36.

Dütting, Paul, Monika Henzinger, and Martin Starnberger. 2013. "Valuation Compressions in VCG-Based Combinatorial Auctions." In *Proceedings of the 9th International Conference*, *WINE 2013*, edited by Yiling Chen and Nicole Immorlica, 146–59. New York: Springer.

Edelman, Benjamin, Michael Ostrovsky, and Michael Schwarz. 2007. "Internet Advertising and the Generalized Second-Price Auction: Selling Billions of Dollars Worth of Keywords." *American Economic Review* 97 (1): 242–59.

Federal Communications Commission (FCC). 2014. "Repacking Constraint Files." http://data.fcc.gov/download/incentiveauctions/Constraint_Files/.

Fréchette, Alexandre, Neil Newman, and Kevin Leyton-Brown. 2015. "Solving the Station Repacking Problem." Unpublished.

Groves, Theodore. 1973. "Incentives in Teams." *Econometrica* 41 (4): 617–31.

Hartline, Jason D., Robert Kleinberg, and Azarakhsh Malekian. 2015. "Bayesian Incentive Compatibility via Matchings." *Games and Economic Behavior* 92: 401–29.

Hartline, Jason D., and Brendan Lucier. 2009. "Bayesian Algorithmic Mechanism Design." http://arxiv.org/abs/0909.4756.

Huang, Zhiyi, Lei Wang, and Yuan Zhou. 2011. "Black-Box Reductions in Mechanism Design." In *Proceedings of the 14th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems*, 254–65.

Jia, Juncheng, Qian Zhang, Qin Zhang, and Mingyan Liu. 2009. "Revenue Generation for Truthful Spectrum Auction in Dynamic Spectrum Access." In *Proceedings of the Tenth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 3–12. New York: ACM.

Lavi, Ron, and Chaitanya Swamy. 2011. "Truthful and Near-Optimal Mechanism Design via Linear Programming." *Journal of the ACM* 58 (6): Article 25.

Lehmann, Daniel, Liaden Ita O'Callaghan, and Yoav Shoham. 2002. "Truth Revelation in Approximately Efficient Combinatorial Auctions." *Journal of the ACM* 49 (5): 577–602.

Lin, Wei-Yu, Guan-Yu Lin, and Hung-Yu Wei. 2010. "Dynamic Auction Mechanism for Cloud Resource Allocation." In *Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, edited by Manish Parashar and Rajkumar Buyya, 591–92. IEEE.

Lucier, Brendan, and Allan Borodin. 2010. "Price of Anarchy for Greedy Auctions." In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, edited by Moses Charikar, 537–53. SIAM.

Milgrom, Paul, and Ilya Segal. 2014. "Deferred-Acceptance Auctions and Radio Spectrum Reallocation." In *Proceedings of the Fifteenth ACM Conference on Economics and Computation*, 185–86. New York: ACM.

Motwani, Rajeev, and Prabhakar Raghavan. 1995. *Randomized Algorithms.* Cambridge, UK: Cambridge University Press.

Myerson, Roger B. 1981. "Optimal Auction Design." *Mathematics of Operations Research* 6 (1): 58–73.

Syrgkanis, Vasilis, and Eva Tardos. 2013. "Composable and Efficient Mechanisms." In *Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing*, 211–20. New York: ACM.

Varian, Hal R. 2007. "Position Auctions." *International Journal of Industrial Organization* 25 (6): 1163–78.

Vazirani, Vijay V. 2001. *Approximation Algorithms.* Berlin: Springer.

Vickrey, William. 1961. "Counterspeculation, Auctions, and Competitive Sealed Tenders." *Journal of Finance* 16 (1): 8–37.

Wan, Peng-Jun. 2013. "Greedy Approximation Algorithms." In *Handbook of Combinatorial Optimization*, ed. Panos M. Pardalos, Ding-Zhu Du, and Ronald L. Graham, 1599–1629. New York: Springer.

Zhou, Xia, Sorabh Gandhi, Subhash Suri, and Haitao Zheng. 2008. "eBay in the Sky: Strategy-Proof Wireless Spectrum Auctions." In *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*, 2–13. New York: ACM.