Cost-Recovering Bayesian Algorithmic Mechanism Design [*]

Hu Fu[†]   Brendan Lucier[‡]   Balasubramanian Sivan[§]   Vasilis Syrgkanis[¶]

Abstract

We study the design of Bayesian incentive compatible mechanisms in single parameter domains, for the objective of optimizing social efficiency as measured by social cost. In the problems we consider, a group of participants compete to receive service from a mechanism that can provide such services at a cost. The mechanism wishes to choose which agents to serve in order to maximize social efficiency, but is not willing to suffer an expected loss: the agents' payments should cover the cost of service in expectation.

We develop a general method for converting arbitrary approximation algorithms for the underlying optimization problem into Bayesian incentive compatible mechanisms that are cost-recovering in expectation. In particular, we give polynomial time black-box reductions from the mechanism design problem to the problem of designing a social cost minimization algorithm without incentive constraints. Our reduction increases the expected social cost of the given algorithm by a factor of $O(\log(\min\{n, h\}))$, where n is the number of agents and h is the ratio between the highest and lowest nonzero valuations in the support. We also provide a lower bound illustrating that this inflation of the social cost is essential: no BIC cost-recovering mechanism can achieve an approximation factor better than $\Omega(\log(n))$ or $\Omega(\log(h))$ in general.

Our techniques extend to show that a certain class of truthful algorithms can be made cost-recovering in the non-Bayesian setting, in such a way that the approximation factor degrades by at most $O(\log(\min\{n, h\}))$. This is an improvement over previously-known constructions with inflation factor $O(\log n)$.

1 Introduction

Consider the following scenario: n self-interested agents wish to receive service from a central service provider. The provider can give service to any set S of the agents, but at a cost $C(S)$, where costs are monotone: $C(S) \leq C(T)$ when $S \subseteq T$. Each agent has a private value for obtaining service, which they could misrepresent if they so choose. The provider must decide, given the reported values of the agents, which subset to serve and how much payment to collect from each one. The goal of the service provider is to maximize the social welfare: the value of the served agents minus the service costs. How should the server proceed, given that the agents are rational and may strategically manipulate their declarations?

If we ignore computational considerations, this mechanism design problem can be resolved via the well-known VCG mechanism, which optimizes social welfare and induces truth-telling as a dominant strategy (i.e., it is in each agent's best interest to report his value truthfully, regardless of the behavior of the other agents). If we ignore the incentive constraints, then for many problems of this form (e.g. steiner tree, vertex cover, etc.) there are known approximation algorithms that obtain nearly efficient outcomes; however, such algorithms in general do not admit payment schemes that would induce truth-telling behavior from the participants. Finding satisfactory solutions that overcome both the algorithmic and economic difficulties inherent in such problems is the primary research agenda in the field of algorithmic mechanism design.

A recent line of work has sought to address such problems by considering the *Bayesian* setting, where agent values are drawn independently from publicly-known distributions. In such settings, there exist black-box reductions that convert an arbitrary algorithm into an incentive compatible (i.e., truthful) mechanism with no loss in expected social welfare [9, 2, 10] (where truthfulness in the Bayesian setting means that truth-telling is a Bayes-Nash equilibrium of the mechanism). Such transformations reduce the mechanism design problem to a purely algorithmic one, decoupling the economic and computational constraints. A mechanism designer is therefore free to design approximation algorithms, tailored to the specifics of the problem at hand, without paying heed to issues of agent incentives.

Our study begins with the observation that these black-box reductions have an unfortunate property: the server may incur a net loss in expectation. That is, the payments collected by the mechanism may not cover service costs, in expectation over the agent types. Even a server who wishes to maximize the social welfare may balk at the prospect of following such a protocol. Our motivating question, then, is whether the theory of Bayesian black-box reductions can be modified to avoid such expected losses. This can be viewed as a Bayesian version of a *cost-sharing* mechanism design problem, in which the costs for service must be divided among the participants in the mechanism. Our contribution is to initiate the study of such cost-sharing problems in the Bayesian domain, and to exhibit general black-box reductions converting arbitrary algorithms into truthful cost-sharing mechanisms.

We note that, as in the theory of cost-sharing, one immediately encounters strong impossibility results in such problems: social welfare is an ill-behaved optimization metric for which no approximation guarantees are possible in polynomial time even in the full information setting [6]. Thus, following recent developments in the cost-sharing literature [18], we describe economic efficiency with respect to minimizing the *social cost*: the service costs plus the total value of the agents who are not served.

The problem of designing cost-sharing mechanisms that minimize social cost has been extensively studied in the non-Bayesian domain. Truthful cost-recovering mechanisms have been developed for many specific problem formulations, such as Steiner tree/forest [12, 18, 8, 16], facility location [16], multicast routing [6], and scheduling problems [3]. These mechanisms generally follow a high-level approach due to Moulin [13]. Roughly speaking, a Moulin mechanism proceeds by selecting an initial allocation and then iteratively offering cost-recovering prices to the current set of players. Any player who is not willing to pay his offered

price is then removed from the set and the process repeats. Such mechanisms have been used with great success for numerous problems, but in general the offered prices must be tailored to a particular problem and algorithm; the construction does not generally apply to arbitrary approximation methods.

A more general construction, also based upon Moulin mechanisms, was recently proposed by Georgiou and Swamy [7] in the non-Bayesian setting. They show that an arbitrary approximation mechanism that is dominant strategy truthful and satisfies a no-bossiness condition can be converted into a truthful cost-recovering mechanism, while increasing the social cost by a factor of $O(\log n)$. This dependency on n matches a lower bound due to Dobzinski et al. [5]. While their method applies to many types of algorithms, including a broad class of LP-based algorithms, the truthfulness and no-bossiness requirements limit its generality. We ask: is a fully general reduction possible in the Bayesian domain, where incentive and efficiency constraints are required to hold in expectation over the agent types?

Our Results Our main result is a general reduction that converts an arbitrary *algorithm* into a Bayesian incentive compatible mechanism with the property that the server does not incur an expected loss. Our reductions are *black-box*, meaning that they require only the ability to query the given algorithm on arbitrary input profiles. We actually provide two different reductions, with slightly different guarantees on social cost. The first increases the expected social cost of the original algorithm by a factor of $O(\log n)$, and the second increases the social cost of the original algorithm by a factor of $O(\log(v_{\max}/v_{\min}))$ where $v_{\max}$ and $v_{\min}$ are the largest and smallest non-zero values in the support of the value distributions. Combining these two constructions, we contain the increase in social cost to a factor of $O(\min\{\log n, \log(v_{\max}/v_{\min})\})$.

We also demonstrate that the increase in social cost exhibited by our constructions is essential. Specifically, based on the construction of Dobzinski et al. [5], we show that no BIC mechanism that recovers cost in expectation can achieve an approximation factor (to the optimal social cost) better than $O(\log(v_{\max}/v_{\min}) - \sqrt{v_{\max}/v_{\min}n})$. An implication of this bound is that our dependencies on n and $v_{\max}/v_{\min}$ are tight: no cost-recovering BIC mechanism can achieve approximation factor $o(\log n)$ or approximation factor $o\left(\log(v_{\max}/v_{\min})\right)$.

The ideas underpinning our reductions are motivated by the Moulin mechanism. We apply the paradigm of determining appropriate payments for each agent, and then repeatedly excluding agents who are unwilling to pay the required amount. However, rather than sequentially excluding agents from an outcome returned by the algorithm, we apply a pre-processing step to the given algorithm in which we sequentially exclude *potential agent declarations*. This analysis makes use of well-known characterizations of Bayesian incentive compatibility with respect to an algorithm's interim allocation curves (the expected allocation to a player, as a function of his declaration, over the space of declarations of the other agents). The result of this pre-processing step will be a pre-computed threshold, specific to the given algorithm; any agent who bids below the threshold will be denied service regardless of the original algorithm's outcome. High thresholds allow the mechanism to charge large payments, but may substantially increase social costs. We prove that this tension can be balanced so that costs are recovered but yet the social costs are not increased by too much.

A technical difficulty in the above approach is that knowledge about the algorithm, necessary to determine appropriate thresholds, must be obtained via sampling, which introduces errors. In order to guarantee that the mechanism recovers costs entirely, rather than only approximately, it is necessary to modify our mechanisms to recover more cost than strictly necessary. We prove that this has only a small impact on social cost, which can be made arbitrarily small via additional sampling.

We also note that our mechanism with approximation factor $O(\log(v_{\max}/v_{\min}))$ extends to the non-Bayesian setting as well. Indeed, we show that the cost-sharing construction due to Georgiou and Swamy [7] can be modified so that it increases the social cost of a given algorithm by a factor of $O(\min\{\log n, \log(v_{\max}/v_{\min})\})$,

rather than $O(\log n)$. This provides an improvement to the obtainable approximation factors when agent values lie in a small range or are drawn from a small set of possible values.

Related work Moulin mechanisms were proposed by Moulin [13] and Moulin and Shenker [14], who show that the resulting mechanism will be cost-recovering as long as the prices offered satisfy a cross-monotonicity condition. Moulin mechanisms have been applied to various cost-sharing problems, such as Steiner tree/forest [12, 18, 8, 16], facility location [16], multicast routing [6], and scheduling problems [3]. For the most part such mechanisms are also required to be approximately budget balanced, meaning that the mechanism does generate (too large of) a profit. Immorlica et al. [11] showed that, for certain problems, such cross-monotonic pricing methods can imply that the budget-balance approximability factor can be very high.

Roughgarden and Sundararajan [18] suggested social cost as a metric for social efficiency, allowing the study of approximate efficiency in cost-recovering mechanisms. Subsequent work considered the approximation factors of cost-sharing methods according to this metric, for various problems [18, 3, 17, 4]. Dobzinski et al. [5] show that, for the public-excludable good problem ($C(S) = 1$ for all $S \neq \emptyset$, $C(\emptyset) = 0$), any (ex post) truthful cost-recovering mechanism will be an $\Omega(\log n)$ approximation to the optimal social cost.

Georgiou and Swamy provide a general method for converting truthful algorithms into truthful cost-recovering mechanisms. They say an algorithm $\mathcal{A}$ is *no-bossy* if, for each i, if $\mathcal{A}$ serves a set $S \ni i$ on input $\mathbf{v}$, then $\mathcal{A}$ will also serve this same set S on any input (v_i', v_{-i}) with $v_i' > v_i$. They show that any ρ-approximate algorithm that is dominant strategy truthful and no-bossy can be converted into a truthful cost-recovering $O(\rho \log n)$-approximate mechanism $\mathcal{M}$. They also provide a linear programming technique for constructing truthful no-bossy algorithms. Their reduction applies in the ex post (non-Bayesian) setting, rather than the Bayesian setting that we consider.

In the Bayesian domain, where truthfulness is relaxed to Bayesian incentive compatibility, there are black-box reductions that convert approximation algorithms into truthful mechanisms in single-parameter [9] and multi-parameter [2, 10] domains. These reductions incur an additive loss to the expected social welfare of the original algorithm, which can be made arbitrarily small. These constructions do not consider the cost recovery properties of the resulting mechanisms.

2 Preliminaries

Single Parameter Mechanism Design Mechanism design studies optimization problems with private information. Among a set of bidders $[n] = \{1, 2, \cdots, n\}$, a mechanism decides upon a subset S of receivers of a certain service. Each bidder i has a private valuation v_i for the service. To incentivize bidders to reveal their valuations truthfully, the mechanism also charges a payment. Formally, a mechanism consists of an *allocation* rule $x : \mathbb{R}_+^n \mapsto [0, 1]^n$ and a *payment* rule $p : \mathbb{R}_+^n \mapsto \mathbb{R}_+^n$. For a valuation profile $\mathbf{v} = (v_1, \ldots, v_n)$, $x_i(\mathbf{v})$ is the probability that bidder i receives the service, and $p_i(\mathbf{v})$ is the payment made by bidder i. Bidder i has a utility of $x_i(\mathbf{v})v_i - p_i(\mathbf{v})$. A mechanism is said to be *individually rational* (IR) if no bidder ever has a negative utility. We impose the IR condition throughout the paper. A mechanism is said to be *ex post incentive compatible* or *truthful* if,

$$x_i(v_i, v_{-i})v_i - p_i(v_i, v_{-i}) \geq x_i(v_i', v_{-i})v_i - p_i(v_i', v_{-i}), \qquad \forall i, \forall v_i, v_i', v_{-i}. \tag{IC}$$

Social Welfare and Social Cost A well studied objective in mechanism design is the *social welfare*, defined as $\sum_{i \in S} v_i$, where S is the set of bidders receiving a service. In this work, we focus on scenarios where a cost $C(S)$ is incurred when subset S is served. We assume that $C(\emptyset) = 0$, and that $\forall S \subset T$, $C(S) \leq C(T)$. The *social cost* of a subset S is $C(S) + \sum_{i \notin S} v_i$. Given an algorithm $\mathcal{A} : \mathbb{R}^n_+ \mapsto 2^{[n]}$, we write the social welfare of $\mathcal{A}$ on $\mathbf{v}$ as $SW(\mathcal{A}, \mathbf{v})$, and the social cost similarly as $SC(\mathcal{A}, \mathbf{v})$. We say a mechanism recovers its cost if for all $\mathbf{v}$ $\sum_i p_i(\mathbf{v}) \geq C(S)$.

Bayesian Mechanism Design This paper focuses on situations in which bidders have only incomplete information regarding the other bidders, captured by the study of Bayesian mechanism design. Each bidder's valuation v_i is independently drawn from a known distribution F_i, with probability density function f_i. By scaling all values and costs down, we may assume without loss of generality that all distributions are supported on $[0, 1]$. We denote by $v_{\max}$ the supremum of the support of all F_i's, and $v_{\min}$ the infimum of nonzero values in the support. We assume $v_{\min}$ is bounded away from 0 and denote by h the ratio $v_{\max}/v_{\min}$.

The allocation rule of a mechanism gives rise to an interim allocation for each bidder. The interim allocation $x_i(v_i)$ is bidder i's probability of getting served, taking an expectation over the other bidders' valuations, i.e., $\mathbf{E}_{v_{-i}}[x_i(v_i, v_{-i})]$. A mechanism is said to be *Bayesian incentive compatible* (BIC) if

$$x_i(v_i)v_i - \mathbf{E}_{v_{-i}}\left[p_i(v_i, v_{-i})\right] \geq x_i(v_i')v_i - \mathbf{E}_{v_{-i}}\left[p_i(v_i', v_{-i})\right], \qquad \forall i, v_i, v_i'. \qquad \text{(BIC)}$$

The term $\mathbf{E}_{v_{-i}}[p_i(v_i, v_{-i})]$ is also called the interim payment $p_i(v_i)$. Interim allocation rules and payments of BIC mechanisms are characterized by the following classic result.

Lemma 1 (Myerson 15). *A mechanism with interim allocation rules $\mathbf{x}$ is BIC iff each x_i is monotone non-decreasing, and the expected (or interim) payment $p_i(v_i)$ made by bidder i with valuation v_i is $v_i x_i(v_i) - \int_0^{v_i} x_i(y)\,\mathrm{d}y$.*

We will denote by $SC(\mathcal{A})$ the expected social cost of an algorithm $\mathcal{A}$, i.e., $\mathbf{E}_\mathbf{v}[C(\mathcal{A}(\mathbf{v})) + \sum_{i \notin \mathcal{A}(\mathbf{v})} v_i]$. We say a mechanism $\mathcal{M}$ recovers its cost in expectation if $\mathbf{E}_\mathbf{v}[\sum_i p_i(\mathbf{v})] \geq \mathbf{E}_\mathbf{v}[C(\mathcal{M}(\mathbf{v}))]$, where $\mathcal{M}(\mathbf{v})$ is the set of bidders $\mathcal{M}$ serves at valuation profile $\mathbf{v}$, which is allowed to be randomized. Under the requirement of cost recovery in expectation, it is without loss of generality to assume that a BIC mechanism charges a payment of $\frac{p_i(v_i)}{x_i(v_i)}$ to bidder i with valuation v_i when he is served, and 0 when he is not served.

Black-Box Reductions for BIC Mechanisms While ex post truthful mechanisms optimize objectives (such as social welfare) in the straitjacket of incentive constraints, Hartline and Lucier [9] showed that, if one were to relax the solution concept to that of BIC, essentially any (approximate) social welfare maximization algorithm can be transformed to a BIC mechanism with little loss of social welfare.

Theorem 2 (Hartline and Lucier 9). *In any single-dimensional setting where the agents' valuations are drawn independently from known distributions, given any $\epsilon > 0$, there is a polynomial time computable reduction $\mathcal{R}$ such that, given any algorithm $\mathcal{A}$, $\mathcal{R}(\mathcal{A})$ is a BIC mechanism with $\mathrm{E}_\mathbf{v}[SW(\mathcal{R}(\mathcal{A}))] \geq \mathrm{E}_\mathbf{v}[SW(\mathcal{A})] - \epsilon$.*

The resampling technique of Theorem 2 easily applies to the settings where expected social cost is the objective, and we have the following corollary.

Corollary 3. *In any single-dimensional setting where the agents' valuations are drawn independently from known distributions, given any $\epsilon > 0$, there is a polynomial time computable reduction $\mathcal{R}$ such that, given any algorithm $\mathcal{A}$, $\mathcal{R}(\mathcal{A})$ is a BIC mechanism with $\mathrm{E}_\mathbf{v}[SC(\mathcal{R}(\mathcal{A}))] \leq \mathrm{E}_\mathbf{v}[SC(\mathcal{A})] + \epsilon$.*

ALGORITHM 1: A reduction from BIC cost-recovering-in-expectation social cost minimization to BIC social cost minimization

---

    Input : A BIC algorithm $\mathcal{A}$ and a valuation profile $\mathbf{v}$
    Output: A set of agents to be served, and a price for each agent

1 Let $S(\mathbf{v})$ = set of winners returned by $\mathcal{A}$ on input $\mathbf{v}$; compute the interim allocation rule $\mathbf{x}$ in $\mathcal{A}$;

2 **for** $j = 0$ *to* $1 + \lfloor \log h \rfloor$ **do**

$$\text{Let } S_j(\mathbf{v}) = \{i \in S(\mathbf{v}) | v_i \geq 2^j\}$$

$$\text{Let } p_{i,j}(v_i) = \begin{cases} 0 & \text{if } v_i < 2^j \\ v_i x_i(v_i) - \int_{2^j}^{v_i} x_i(y) \, \mathrm{d}y & \text{if } v_i \geq 2^j \end{cases}$$

      if $\mathbf{E_v}[C(S_j(\mathbf{v})] \leq \mathbf{E_v}[\sum_i p_{i,j}(v_i)]$ **then**
         | Set $k = j$;
         | Go to step 3.

3 Serve agents in $S_k(\mathbf{v})$, and charge agent i a price of $\frac{p_{i,k}(v_i)}{x_i(v_i)}$.

    We note that Hartline and Lucier prove Theorem 2 by first showing how to construct an ϵ-BIC mechanism, then showing how to convert this into a BIC mechanism at a small loss to social welfare. Due to the difference in the objective, the technique does not directly apply to our case; a minor modification to the construction of Hartline and Lucier is required. In Appendix A we briefly describe the construction involved.

3   Bayesian Incentive Compatible Cost Recovery

In this section we present our main result of converting an arbitrary algorithm to a Bayesian Incentive Compatible mechanism that approximately minimizes social-cost in expectation, and recovers cost in expectation. We give two separate reductions, one that gives a $O(\log h)$ approximation, and the other gives an $O(\log n)$ approximation, and thus we have a $O\big(\min \{\log h, \log n\} \big)$ approximation.

Remark 4. Our reductions require computing certain expectations, which can be obtained only via sampling, and hence with small error. In this section, all expectations are assumed to be accurately available, i.e., we assume functional access to the interim allocation rules and valuation distributions. We refer to this as functional access to an algorithm. We address the error due to sampling in the more realistic black-box model in Section 4. Also, the truthful payment corresponding to the algorithm $\mathcal{A}$ requires knowing the output of $\mathcal{A}$ on infinitely many values. In this section, we also assume that interim payments are accurately available, and we describe the procedure to circumvent this issue in Section 4.

In the presentation of Theorem 5, it is convenient to scale valuations to $[1, h]$ from $[0, 1]$, mapping $v_{\min}$ to 1.

Theorem 5. *Given functional access to an algorithm $\mathcal{A}$ which incurs an expected social cost of $C(\mathcal{A})$, and when the values of all agents are distributed in $[1, h]$, the reduction in Algorithm 1 outputs a BIC mechanism, which incurs an expected social cost of $O(\log h)C(\mathcal{A})$ and recovers the cost in expectation.*

Proof. By Theorem 3.1 of [9] (arxiv version), it is without loss of generality to assume that the input algorithm $\mathcal{A}$ is BIC, i.e., has a monotone increasing interim allocation rule. (Theorem 3.1 of [9] is a version

of Corollary 3 where full functional access to allocation rule is available, and shows that in such settings there is not even an additive ϵ loss.) Algorithm 1 proceeds in two phases. In the preprocessing phase, it computes a number k to be used in the next phase. This phase does not depend on the agents' actual valuations, and only uses information of the valuation distributions and the algorithm $\mathcal{A}$. In the second phase, the mechanism uses bidders' bids (declarations of their valuations) and k to modify the set $S(\mathbf{v})$ returned by $\mathcal{A}$. The actual set of winners and payments are determined in the second phase.

In the preprocessing phase, the mechanism experiments with *truncating* the interim allocation of $\mathcal{A}$ at different thresholds. To truncate an interim allocation rule x_i at a threshold is to refuse service to all agents that report values below the threshold, while keeping intact services to others. The resulting interim allocation rule is still monotone after truncation. The payment $p_{i,j}(v_i)$ computed by the algorithm is simply the expected payment made by the agent in such a truncated allocation rule (recall Lemma 1).

By the procedure outlined for computing k in Algorithm 1, it follows that the resulting mechanism recovers the cost in expectation, i.e., when the "if" condition in the algorithm becomes true, cost is recovered in expectation. (Note that there is always a k for which the "if" condition becomes true: at $k = 1 + \lfloor \log h \rfloor$, we have $S_k(\mathbf{v}) = \emptyset$, and the "if" condition becomes true.) In addition, for all $j \in 0 \ldots k - 1$, we have

$$\mathbf{E_v} \left[\sum_i p_{i,j}(v_i) \right] < \mathbf{E_v} \left[C(S_j(\mathbf{v})) \right]. \tag{1}$$

We claim that, in expectation, the additional social cost incurred by the mechanism when dropping the agents in $S(\mathbf{v}) \setminus S_k(\mathbf{v})$ is bounded by an $O(\log h)$ factor times $C(\mathcal{A})$. To begin with, the expected social cost of the mechanism is

$$\mathbf{E_v} \left[C(S_k(\mathbf{v})) + \sum_{i \notin S(\mathbf{v})} v_i x_i(\mathbf{v}) + \sum_{i \in S(\mathbf{v}) \setminus S_k(\mathbf{v})} v_i \right].$$

Since $S_k(\mathbf{v}) \subseteq S(\mathbf{v})$, the first two terms are upper bounded by $C(\mathcal{A})$. We therefore need only to bound the last term. Looking at this term from each agent's perspective, we have

$$\mathbf{E_v} \left[\sum_{i \in S(\mathbf{v}) \setminus S_k(\mathbf{v})} v_i \right] = \sum_i \sum_{j=0}^{k-1} \int_{2^j}^{2^{j+1}} v_i x_i(v_i) f_i(v_i) \, dv_i \leq 2 \sum_i \sum_{j=0}^{k-1} \int_{2^j}^{2^{j+1}} 2^j x_i(v_i) f_i(v_i) \, dv_i.$$

Since $p_{i,j}(v_i) = v_i x_i(v_i) - \int_{2^j}^{v_i} x_i(y) \, dy \geq 2^j x_i(v_i)$ for $v_i \geq 2^j$, we have

$$\sum_i \sum_{j=0}^{k-1} \int_{2^j}^{2^{j+1}} 2^j x_i(v_i) f_i(v_i) \, dv_i \leq \sum_{j=0}^{k-1} \mathbf{E_v} \left[\sum_i p_{i,j}(v_i) \right].$$

But by (1), for each $j < k$, $\mathbf{E_v}[\sum_i p_{i,j}(v_i)]$ is in turn bounded by $\mathbf{E_v}[C(S_j(\mathbf{v}))] \leq \mathbf{E_v}[C(S(\mathbf{v}))] \leq C(\mathcal{A})$. As there are only $k \leq \log h$ such j's, the additional social cost is at most $O(\log h)$ times $C(\mathcal{A})$. $\qquad \blacksquare$

We now present the other reduction that gives better approximations when n is smaller than h.

Theorem 6. *Given functional access to an arbitrary algorithm $\mathcal{A}$ which incurs an expected social cost of $C(\mathcal{A})$, the reduction in Algorithm 2 outputs a BIC mechanism, which incurs an expected social cost of $O(\log n)C(\mathcal{A})$ and recovers the cost in expectation.*

7

ALGORITHM 2: A reduction from BIC cost-recovering-in-expectation social cost minimization to BIC social cost minimization

Input : A BIC algorithm $\mathcal{A}$; $C(S)$ for every set S of agents; a value $\delta > 0$
Output: A set of agents to be served, and a price for each agent

1 Initialize $S_0(\mathbf{v}) = S(\mathbf{v}) = $ set of winners returned by $\mathcal{A}$ on input $\mathbf{v}$, calculate interim allocation rule $\mathbf{x}$ in $\mathcal{A}$;
2 **for** $j = 0, 1, 2, \ldots$ **do**

$$\text{Let } t_j = \left\lceil \frac{\mathbf{E}_{\mathbf{v}}[C(S_{j-1}(\mathbf{v}))]}{\mathbf{E}_{\mathbf{v}}[|S_{j-1}(\mathbf{v})|]} \right\rceil_\delta, \text{ where } t_0 = 0$$

$$\text{Let } S_j(\mathbf{v}) = \{i \in S(\mathbf{v}) | v_i \geq t_j\}$$

$$\text{Let } p_{i,j}(v_i) = \begin{cases} 0 & \text{if } v_i < t_j \\ v_i x_i(v_i) - \int_{t_j}^{v_i} x_i(y) \, \mathrm{d}y & \text{if } v_i \geq t_j \end{cases}$$

  if $\mathbf{E}_{\mathbf{v}}[C(S_j(\mathbf{v})) \leq \mathbf{E}_{\mathbf{v}}[\sum_i p_{i,j}(v_i)]$ **then**
  | Set $k = j$;
  | Go to step 3.

3 Serve agents in $S_k(\mathbf{v})$, and charge agent $i \in S_k(\mathbf{v})$ a price of $\frac{p_{i,k}(v_i)}{x_i(v_i)}$.

Proof. The idea behind the reduction is similar to that of Theorem 5. The main difference is in the definition of the sets $S_j(\mathbf{v})$. They are defined inductively, as sets of agents whose value is above the average cost threshold.

By the same argument as before, the mechanism recovers the cost in expectation by its definition of k. In addition, we show that the algorithm will terminate after $O(1/\delta)$ steps. If for some j we had $t_j \leq t_{j-1}$ then by definition of t_j we have:

$$\frac{\mathbf{E}_{\mathbf{v}}[C(S_{j-1}(\mathbf{v}))]}{\mathbf{E}_{\mathbf{v}}[|S_{j-1}(\mathbf{v})|]} \leq \left\lceil \frac{\mathbf{E}_{\mathbf{v}}[C(S_{j-1}(\mathbf{v}))]}{\mathbf{E}_{\mathbf{v}}[|S_{j-1}(\mathbf{v})|]} \right\rceil_\delta = t_j \leq t_{j-1},$$

which in turn implies:

$$\mathbf{E}_{\mathbf{v}}[C(S_{j-1}(\mathbf{v}))] \leq t_{j-1} \, \mathbf{E}_{\mathbf{v}}[|S_{j-1}(\mathbf{v})|] \leq \mathbf{E}_{\mathbf{v}}\left[\sum_i p_{i,j-1}(v_i) \right],$$

where the last inequality follows from noting that at iteration $j-1$ the payment of any player that is served is at least t_{j-1}. Thus at that point it must be that the algorithm terminated at iteration $j-1$, since the "if" condition gets satisfied. Therefore, it follows that as long as the algorithm has not terminated, $t_j > t_{j-1}$, and since by definition the thresholds are multiples of δ [1] it must be that $t_j \geq t_{j-1} + \delta$. Thus the algorithm will terminate after $O(1/\delta)$ steps (since values lie in $[0,1]$).

To complete the proof, similar to Theorem 5, we need to bound the term $\mathbf{E}_{\mathbf{v}}[\sum_{i \in S(\mathbf{v}) \setminus S_k(\mathbf{v})} v_i]$. Define $r(l)$ inductively as follows: $r(0) = 0$, and

$$r(j) = \begin{cases} \min \left\{ \ell : \mathbf{E}_{\mathbf{v}}[|S_{r(j-1)}(\mathbf{v})| - |S_\ell(\mathbf{v})|] \geq 1 \right\} & \text{if such an } \ell \ (\leq k) \text{ exists} \\ k & \text{otherwise} \end{cases}$$

---

[1] $\lceil x \rceil_\delta$ is the smallest multiple of δ that is larger than x

Note that $r(j) > r(j-1)$, and $t_{r(j)} > t_{r(j-1)}$. Let $j_{\max}$ be the smallest j for which $r(j) = k$. Note that $j_{\max} \leq n$ since there are at most n agents, i.e., $\mathbf{E_v}[|S(\mathbf{v})|] \leq n$ and therefore the number of times that the expected service set size can decrease is at most n. Let $\alpha_j = \mathbf{E_v}[|S_{r(j-1)}(\mathbf{v})| - |S_{r(j)}(\mathbf{v})|]$. By definition $\alpha_j \geq 1$ for all $j < j_{\max}$. We have,

$$\mathbf{E_v}\left[\sum_{i \in S(\mathbf{v}) \setminus S_k(\mathbf{v})} v_i\right] \leq \sum_i \sum_{j=1}^{j_{\max}} \int_{t_{r(j-1)}}^{t_{r(j)}} v_i x_i(v_i) f_i(v_i)\, \mathrm{d}v_i$$

$$\leq \sum_i \sum_{j=1}^{j_{\max}} \int_{t_{r(j-1)}}^{t_{r(j)}} t_{r(j)} x_i(v_i) f_i(v_i)\, \mathrm{d}v_i$$

$$\leq \sum_{j=1}^{j_{\max}} t_{r(j)} \mathbf{E_v}\left[|S_{r(j-1)}(\mathbf{v})| - |S_{r(j)}(\mathbf{v})|\right]$$

$$= \sum_{j=1}^{j_{\max}} t_{r(j)} \alpha_j \leq \sum_{j=1}^{j_{\max}} \left\lceil \frac{\mathbf{E_v}[C(S_{r(j)-1}(\mathbf{v}))]}{\mathbf{E_v}[|S_{r(j)-1}(\mathbf{v})|]} \right\rceil_\delta \alpha_j$$

$$\leq \sum_{j=1}^{j_{\max}} \left[\frac{\mathbf{E_v}[C(S_{r(j)-1}(\mathbf{v}))]}{\mathbf{E_v}[|S_{r(j)-1}(\mathbf{v})|]} + \delta \right] \alpha_j$$

$$\leq \sum_{j=1}^{j_{\max}} \frac{\mathbf{E_v}[C(S(\mathbf{v}))]}{\mathbf{E_v}[|S(\mathbf{v})|] - \sum_{\ell<j} \alpha_\ell} \alpha_j + n\delta \leq O(\log n)\, \mathbf{E_v}\left[C(S(\mathbf{v}))\right] + n\delta,$$

where the last-but-one inequality follows from noting that $\sum_{j=1}^{j_{\max}} \alpha_j \leq \mathbf{E_v}[|S(\mathbf{v})|] \leq n$.

For the last inequality, begin by noting that $j_{\max} \leq n$ and for all $j < j_{\max}$, $\alpha_j \geq 1$. We need to show that $\sum_{j=1}^{j_{\max}} \frac{\alpha_j}{\mathbf{E_v}[|S(\mathbf{v})|] - \sum_{\ell<j} \alpha_\ell} \leq O(\log n)$. Note that $\sum_{j=1}^{j_{\max}} \frac{\alpha_j}{\mathbf{E_v}[|S(\mathbf{v})|] - \sum_{\ell<j} \alpha_\ell} \leq \sum_{j=1}^{j_{\max}} \frac{\alpha_j}{\sum_{j \leq \ell \leq j_{\max}} \alpha_\ell} \leq 1 + \sum_{j=1}^{j_{\max}-1} \frac{\alpha_j}{\sum_{j \leq \ell \leq j_{\max}-1} \alpha_\ell}$. In the final summation, all the α_j's involved are at least 1. The following claim completes the proof of the last inequality since we have $\sum_{j=1}^{j_{\max}} \alpha_j \leq n$.

Claim 7. *Given k real numbers $a_1, \ldots, a_k$, such that $a_i \geq 1$ for all i,*

$$\sum_{j=1}^{k} \frac{a_j}{\sum_{t \geq j} a_t} \leq 2 \cdot H_{\sum_{j=1}^{k} \lfloor a_j \rfloor},$$

where $H_r = \sum_{i=1}^{r} \frac{1}{i} \leq 1 + \log r$.

Proof.

$$\sum_{j=1}^{k} \frac{a_j}{\sum_{t \geq j} a_t} = \sum_{j=1}^{k} \frac{\lfloor a_j \rfloor}{\sum_{t \geq j} a_t} + \sum_{j=1}^{k} \frac{a_j - \lfloor a_j \rfloor}{\sum_{t \geq j} a_t}$$

$$\leq \sum_{j=1}^{k} \frac{\lfloor a_j \rfloor}{\sum_{t \geq j} \lfloor a_t \rfloor} + \sum_{j=1}^{k} \frac{1}{\sum_{t \geq j} a_t}$$

$$\leq \sum_{j=1}^{k} \frac{\lfloor a_j \rfloor}{\sum_{t \geq j} \lfloor a_t \rfloor} + H_k$$

9

$$\leq \sum_{j=1}^{k} \frac{\lfloor a_j \rfloor}{\sum_{t \geq j} \lfloor a_t \rfloor} + H_{\sum_{j=1}^{k} \lfloor a_j \rfloor}$$

We now show that $\sum_{j=1}^{k} \frac{\lfloor a_j \rfloor}{\sum_{t \geq j} \lfloor a_t \rfloor} \leq H_{\sum_{j=1}^{k} \lfloor a_j \rfloor}$. We drop the floors, and assume that the a_j's are integers in the part below. Consider the term $\frac{a_j}{\sum_{t \geq j} a_t}$.

$$\frac{a_j}{\sum_{t \geq j} a_t} = \underbrace{\frac{1}{a_j + a_{j+1} + \ldots + a_k} + \ldots + \frac{1}{a_j + a_{j+1} + \ldots + a_k}}_{a_j \text{ times}}$$

$$\leq \frac{1}{1 + a_{j+1} + \ldots + a_k} + \frac{1}{2 + a_{j+1} + \ldots + a_k} + \ldots + \frac{1}{a_j + a_{j+1} + \ldots + a_k}$$

$$= \sum_{t=1+\sum_{k>j} a_k}^{a_j + \sum_{k>j} a_k} \frac{1}{t}$$

So we have,

$$\sum_{j=1}^{k} \frac{a_j}{\sum_{t \geq j} a_t} \leq \sum_{j=1}^{k} \sum_{t=1+\sum_{k>j} a_k}^{\sum_{k \geq j} a_k} \frac{1}{t} = \sum_{t=1}^{\sum_j a_j} \frac{1}{t} = H_{\sum_j a_j}$$

$\square$

On choosing $\delta = \epsilon/n$, we approximate social cost to a factor of $O(\log n)$ with an additive loss of ϵ. The number of iterations is at most $\frac{1}{\delta} = \frac{n}{\epsilon}$ because after $1/\delta$ iterations the threshold would have reached 1. $\square$

4 Sampling and the Black-Box Model

The mechanisms of Section 3 work under the assumption that the mechanism designer has complete knowledge of the interim allocation rules and valuation distributions in functional form, and can perform arbitrary calculus on those functions. This is a strong assumption; in general it may be highly non-trivial to precisely determine the interim allocation rules of an arbitrary algorithm. In this section we describe ways to implement the reductions in Section 3 in a more realistic model: the algorithm $\mathcal{A}$ is provided as a black box that can be queried on arbitrary input vectors. We refer to this as the *black-box* model of computation.

Our approach will be to estimate the allocation rules of $\mathcal{A}$ via sampling, then apply the reductions from the ideal model. This introduces sampling error that must be bounded; the result will be a mechanism that is *approximately* cost-recovering. We will then show how to modify our constructions to be cost-recovering in the non-approximate sense. The following theorem summarizes the result.

Theorem 8. *Given $\epsilon > 0$, black-box access to algorithm $\mathcal{A}$ and distribution $\mathbf{F}$, one can construct a BIC mechanism $\mathcal{M}$ that is cost-recovering in expectation, with $\mathrm{E}_{\mathbf{v}}[SC(\mathcal{M})] \leq O(\min\{\log(h), \log(n)\}) \mathrm{E}_{\mathbf{v}}[SC(\mathcal{A})] + \epsilon$. The mechanism runs in time polynomial in $1/\epsilon$, n, and the runtime of $\mathcal{A}$.*

4.1 Computing BIC payments

Suppose that we are given an algorithm $\mathcal{A}$ with monotone interim allocation rules $\mathbf{x}$, and moreover we are told that charging the expected payments of Lemma 1, $p_i(v_i) = v_i x_i(v_i) - \int_0^{v_i} x_i(y) \, dy$, would recover costs in expectation. In this case, all that would be required to obtain a BIC mechanism is to execute algorithm $\mathcal{A}$ and compute payments so that the expected payment of agent i is $p_i(v_i)$. However, in the black-box model the mechanism can determine the value of the allocation rules (and hence the required payments) only approximately; charging approximate payments is insufficient for Bayesian incentive compatibility.

There is a well-known procedure to estimate integrals via random sampling, used by Archer et al. [1] to compute payments. For the purpose of having a self-contained exposition, we explain the procedure below.

Theorem 9. *Let $f(\cdot)$ be the probability density function of a random variable $Y \in [0, v]$. Then $\mathbf{E}_Y[\frac{h(Y)}{f(Y)}] = \int_0^v h(z) \, dz$.*

The proof of the theorem follows from the definition of a probability density function. Thus one way to estimate the integral $\int_0^{v_i} x_i(y) \, dy$ is to draw a random variable Y from the uniform distribution $U[0, v_i]$, and return $v_i x_i(Y)$. In expectation, this quantity precisely equals $\int_0^{v_i} x_i(y) \, dy$. Furthermore, the payment of $v_i x_i(v_i) - v_i x_i(Y)$ is always non-negative since $x_i(\cdot)$ is monotone, and thus the mechanism is ex-post IR.

4.2 Estimating Interim Allocation via Sampling

We now describe a method for implementing Algorithms 1 and 2 when the interim allocation rules are not given explicitly. Recall first that, by Corollary 3, given $\epsilon_1 > 0$ and an arbitrary algorithm $\mathcal{A}$, we can construct an algorithm $\overline{\mathcal{A}}$ with monotone interim allocation rules such that $\mathbf{E}_\mathbf{v}[SC(\overline{\mathcal{A}})] \leq \mathbf{E}_\mathbf{v}[SC(\mathcal{A})] + \epsilon_1$. We will therefore assume for the remainder of this section that the algorithm $\mathcal{A}$ has monotone interim allocation rules.

Given black-box access to algorithm $\mathcal{A}$, we will construct approximations to its allocation rules as follows. We first choose some $\delta > 0$, and partition value space $[0, 1]$ into intervals $I_k = ((k-1)\delta, k\delta]$ for $k \in [1/\delta]$. Let $\overline{\mathbf{x}}$ denote the interim allocation rule $\mathbf{x}$, *discretized* over the intervals I_k: that is, for each $v_i \in I_k$, we define $\overline{x}_i(v_i) = \mathbf{E}_{v_i}[x_i(v_i) | v_i \in I_k]$.

For each i and each $k \in [1/\delta]$, we will sample $N = \frac{1}{\epsilon^2} \log(nk/\epsilon)$ valuation profiles $\mathbf{v} \sim \mathbf{F}$, conditional on $v_i \in I_k$. We will then run $\mathcal{A}$ on each of these N inputs, and count the number of times that the resulting allocation includes agent i. Denote by M_{ik} this number. Let $\tilde{\mathbf{x}}$ be the allocation rule defined by $\tilde{x}_i(v_i) = \max_{\ell \leq k}\{M_{i\ell}\}/N$ for all $v_i \in I_k$. We think of $\tilde{\mathbf{x}}$ as an estimated version of $\mathbf{x}$. Note that the reason for the max in the definition of $\tilde{x}_i$ is to guarantee that $\tilde{\mathbf{x}}$ is monotone.

We claim that the result of this sampling generates an estimate to allocation rule $\mathbf{x}$, in the following sense.

Definition 1. Allocation rules $\mathbf{x}$ and $\mathbf{x}'$ are *ϵ-close* if $|x_i(v_i) - x_i'(v_i)| < \epsilon$ for all i and v_i.

Lemma 10. *Let $\tilde{\mathbf{x}}$ be the allocation rule defined by $\tilde{x}_i(v_i) = M_{ik}/N$. Then with probability $1 - \epsilon$ over the randomness in the sampling procedure, $\tilde{x}_i$ is ϵ-close to $\overline{x}_i$ for all i.*

Once our sampling is complete, we have full functional access to curves $\tilde{\mathbf{x}}$. We can therefore apply Algorithms 1 and 2 to the curves $\tilde{\mathbf{x}}$. We claim that, for either algorithm, the analysis of Section 3 will go through unchanged, except that each mechanism will be only approximately cost recovering. We obtain the following result, the proof of which appears in Appendix B.

Lemma 11. *Given $\epsilon > 0$ and black-box access to algorithm $\mathcal{A}$, one can construct a BIC mechanism $\mathcal{M}$ with $\mathbf{E_v}[SC(\mathcal{M})] \leq O(\min\{\log(h), \log(n)\})\,\mathbf{E_v}[SC(\mathcal{A})] + \epsilon$. Moreover, the expected payments in $\mathcal{M}$ are at least $\mathbf{E_v}[C(S(\mathbf{v}))] - \epsilon$.*

It remains to show that we can modify our mechanism to recover expected costs entirely, rather than approximately. This requires a modification to Algorithms 1 and 2. Each algorithm is currently designed to iterate until $\mathbf{E_v}[\sum_i p_i(\mathbf{v})] \geq \mathbf{E_v}[C(S(\mathbf{v}))]$. We will modify each algorithm to instead iterate until $\mathbf{E_v}[\sum_i p_i(\mathbf{v})] \geq \mathbf{E_v}[C(S(\mathbf{v}))] + \epsilon_0$, for some appropriate $\epsilon_0 > 0$. This additional payment of ϵ_0 will be chosen to cover the expected losses due to sampling error. What we must show is that this modification does not inflate the expected social cost by too much. However, this follows immediately from the form of our analysis: in either case, our analysis proceeds by bounding the loss with respect to the chosen threshold, then bounding this threshold with respect to $\mathbf{E_v}[C(S(\mathbf{v}))]$. If we replace this latter bound with $(\mathbf{E_v}[C(S(\mathbf{v}))] + \epsilon)$, the result is an extra term that is at most ϵn. An appropriate choice of ϵ therefore leads to an arbitrarily small increase to the social cost, and the expected sum of payments is at least $\mathbf{E}[C(S_{\tilde{\mathbf{x}}}(\mathbf{v}))] + \epsilon \geq (\mathbf{E}[C(S_{\mathbf{x}}(\mathbf{v}))] - \epsilon) + \epsilon = \mathbf{E}[C(S_{\mathbf{x}}(\mathbf{v}))]$, as required. The resulting mechanisms therefore recover costs in expectation, completing the proof of Theorem 8.

5 Lower Bound for BIC Expected Cost-Recovering Mechanisms

We now show that a lower bound on the approximation to social cost given by Dobzinski et al. [5] extends to BIC mechanisms, and we tighten the analysis there so that the lower bound is in terms of both n and h. In particular, if $h < n$, then the lower bound is $\Omega(\log h)$. In general, we show a lower bound of $\Omega(\log h - \sqrt{h/n})$.

Example 1 (Lower Bound on BIC Social Cost Minimization with Cost Recovery)**.** Consider the following public excludable good problem: agent i's valuation is $v_i = \frac{a_i}{4n}$ where each a_i is drawn independently according to the so-called equal revenue distribution with density function $f(z) = \frac{1}{z^2}$ for $z \in [1, h]$ and is 0 with probability $\frac{1}{h}$. The cost function is given by $C(\emptyset) = 0$ and $C(S) = 1$ for all $S \neq \emptyset$.

It is easy to see that, without requiring cost recovery, we may simply serve every agent and incur a cost of 1. Next we give a lower bound for the expected social cost of any cost recovering BIC mechanism. The proof of Theorem 12 is deferred to Appendix C.

Theorem 12. *Any BIC mechanism for the public excludable good problem described above that recovers cost in expectation has expected social cost at least $\Omega(\log(h) - \sqrt{\frac{h}{n}})$.*

6 Ex-post Truthful Cost Recovery

Georgiou and Swamy [7] proposed the following notion of *no bossiness* for an algorithm and gave a procedure to convert any truthful, no-bossy algorithm to an ex post truthful, cost recovering mechanism with an inflation of social cost up to a factor of $O(\log n)$.

Definition 2 (No Bossy, [7])**.** An algorithm $\mathcal{A}$ is said to be *no bossy* if, for every i, v_i, $v_i{}'$ and v_{-i}, if $i \in S(v_i, v_{-i})$ and $i \in S(v_i{}', v_{-i})$, then $S(v_i, v_{-i}) = S(v_i{}', v_{-i})$.

In this section, we show that such a conversion is also possible with an inflation of $O(\log h)$ in social cost. For the special case in which all agents have either value 0 or 1, our conversion does not require the input algorithm to be either truthful or no bossy. Proofs from this section appear in Appendix D

ALGORITHM 3: A black-box reduction from ex-post truthful cost-recovering social cost minimization to social cost minimization for $0/1$ valuations

---

Input : An algorithm $\mathcal{A}$ and a valuation profile $\mathbf{v}$
Output: A set of agents to be served, and a price for each agent

1 Initialize $S(\mathbf{v}) =$ set of winners returned by $\mathcal{A}$ on input $\mathbf{v}$;

2 Let $\widehat{S}(\mathbf{v}) \leftarrow S(\mathbf{v}) \setminus Z(\mathbf{v})$, where $Z(\mathbf{v}) = \{i \in S(\mathbf{v}) | v_i = 0\}$;

3 **if** $C(\widehat{S}(\mathbf{v})) \leq |\widehat{S}(\mathbf{v})|$ **then**

Serve agents in $\widehat{S}(\mathbf{v})$; charge a price of 1 for each agent in $\widehat{S}(\mathbf{v})$ and zero for the rest.

else

Don't serve any agent and charge zero

ALGORITHM 4: A black-box reduction from truthful cost-recovering social cost minimization to truthful, no-bossy social cost minimization

---

Input : A truthful, no-bossy mechanism $\mathcal{M}$, and a valuation profile $\mathbf{v}$
Output: A set of agents, and a payment for each of them

1 Initialize $S(\mathbf{v}) =$ set returned by $\mathcal{M}$ on input $\mathbf{v}$, and $S_j(\mathbf{v}) = \{i \in S(\mathbf{v}) | v_i \geq 2^j\}$;

2 **for** $j = 0$ *to* $\lfloor \log h \rfloor$ **do**

If $2^j \cdot |S_j(\mathbf{v})| \geq C(S_j(\mathbf{v}))$:

    1. set $k = j$

    2. Serve agents in $S_k(\mathbf{v})$; charge each of them a price of 2^k

3 Set $k = 1 + \lfloor \log h \rfloor$; no agent is served or charged.

6.1 Black-box reduction for $0/1$ valuations

When all bidders' valuations are either 0 or 1, there is a simple procedure to convert any social cost minimization algorithm to an ex post truthful, cost recovering mechanism without increasing the social cost, as we show in Algorithm 3 and Theorem 13.

Theorem 13. *When bidders have only valuations 0 or 1, given black-box access to an arbitrary algorithm $\mathcal{A}$ which incurs a social cost of $C(\mathcal{A})$, the black-box reduction in Algorithm 3 outputs an ex post truthful mechanism whose social cost is no more than $C(\mathcal{A})$.*

6.2 Black-box reduction for general valuations

In this section, for convenience of presentation we again scale up the valuations so that they lie in the range $[1, h]$. We give a black-box conversion from a truthful, no bossy mechanism to a truthful, cost recovering mechanism with an inflation of social cost by a factor of $O(\min\{\log h, \log n\})$. This is achieved by choosing the better one between Algorithm 4 and the reduction by Georgiou and Swamy [7], whose inflation factor is bounded by $O(\log n)$ alone. We now show that the inflation factor of Algorithm 4 is bounded by $O(\log h)$.

Theorem 14. *When values of all agents lie in $[1, h]$, given black-box access to a truthful, no-bossy mechanism $\mathcal{M}$ with social cost $C(\mathcal{M})$, the black-box reduction in Algorithm 4 outputs a mechanism which recovers cost and incurs a social cost of $O(\log h)C(\mathcal{M})$.*

Instead of experimenting with thresholding at powers of 2, Algorithm 4 has the option of proceeding at more flexible paces. In particular, we easily obtain the following corollary.

Corollary 15. *Given a truthful, no-bossy mechanism $\mathcal{M}$ that incurs a social cost of $C(\mathcal{M})$, and when valuations of all agents reside in $\{v_1, \ldots, v_k\}$, there exists an efficiently computable black-box reduction that outputs a mechanism which recovers cost and incurs a social cost of at most $O(kC(\mathcal{M}))$.*

References

[1] Aaron Archer, Christos H. Papadimitriou, Kunal Talwar, and Éva Tardos. An approximate truthful mechanism for combinatorial auctions with single parameter agents. In *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 205–214, 2003.

[2] Xiaohui Bei and Zhiyi Huang. Bayesian incentive compatibility via fractional assignments. In *Proceedings of 22nd annual ACM-SIAM Symposium on Discrete Algorithms*, pages 720–733, 2011.

[3] Janina Brenner and Guido Schäfer. Cost sharing methods for makespan and completion time scheduling. In *Proceedings of the 24th annual conference on Theoretical aspects of computer science*, STACS'07, pages 670–681, 2007. ISBN 978-3-540-70917-6.

[4] Shuchi Chawla, Tim Roughgarden, and Mukund Sundararajan. Optimal cost-sharing mechanisms for steiner forest problems. In *Proceedings of the Second international conference on Internet and Network Economics*, WINE'06, pages 112–123, 2006. ISBN 3-540-68138-8, 978-3-540-68138-0.

[5] Shahar Dobzinski, Aranyak Mehta, Tim Roughgarden, and Mukund Sundararajan. Is shapley cost sharing optimal? In *Proceedings of the 1st International Symposium on Algorithmic Game Theory*, SAGT '08, pages 327–336, 2008.

[6] Joan Feigenbaum, Christos H. Papadimitriou, and Scott Shenker. Sharing the cost of multicast transmissions. *J. Comput. Syst. Sci.*, 63(1):21–41, August 2001. ISSN 0022-0000.

[7] Konstantinos Georgiou and Chaitanya Swamy. Black-box reductions for cost-sharing mechanism design. In *Proceedings of 23rd annual ACM-SIAM Symposium on Discrete Algorithms*, pages 896–913, 2012.

[8] A. Gupta, J. Könemann, S. Leonardi, R. Ravi, and G. Schäfer. An efficient cost-sharing mechanism for the prize-collecting steiner forest problem. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA '07, pages 1153–1162, 2007. ISBN 978-0-898716-24-5.

[9] Jason D. Hartline and Brendan Lucier. Bayesian algorithmic mechanism design. In *44th ACM Symposium on Theory of Computing*, pages 301–310, 2010.

[10] Jason D. Hartline, Robert Kleinberg, and Azarakhsh Malekian. Bayesian incentive compatibility via matchings. In *Proceedings of the 22nd annual ACM-SIAM Symposium on Discrete Algorithms*, pages 734–747, 2011.

[11] Nicole Immorlica, Mohammad Mahdian, and Vahab S. Mirrokni. Limitations of cross-monotonic cost-sharing schemes. *ACM Trans. Algorithms*, 4(2):24:1–24:25, May 2008. ISSN 1549-6325.

[12] Kamal Jain and Vijay Vazirani. Applications of approximation algorithms to cooperative games. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, STOC '01, pages 364–372, 2001. ISBN 1-58113-349-9.

[13] Hervé Moulin. Incremental cost sharing: Characterization by coalition strategy-proofness. *Social Choice and Welfare*, 16(2):279–320, 1999.

[14] Herv Moulin and Scott Shenker. Strategyproof sharing of submodular costs:budget balance versus efficiency. *Economic Theory*, 18(3):511–533, 2001.

[15] Roger Myerson. Optimal auction design. *Mathematics of Operations Research*, 6(1):pp. 58–73, 1981. ISSN 0364765X.

[16] Martin Pál and Éva Tardos. Group strategyproof mechanisms via primal-dual algorithms. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '03, pages 584–, 2003. ISBN 0-7695-2040-5.

[17] Tim Roughgarden and Mukund Sundararajan. Optimal efficiency guarantees for network design mechanisms. In *Proceedings of the 12th international conference on Integer Programming and Combinatorial Optimization*, IPCO '07, pages 469–483, 2007. ISBN 978-3-540-72791-0.

[18] Tim Roughgarden and Mukund Sundararajan. Quantifying inefficiency in cost-sharing mechanisms. *J. ACM*, 56(4), 2009.

A Improving ϵ-BIC to BIC

In this section we discuss the construction from the statement of Corollary 3. The purpose of the discussion is to illustrate a minor modification to the method of Hartline and Lucier 9 for converting ϵ-BIC mechanisms to BIC mechanisms. We will briefly recall their ϵ-BIC construction for the sake of completeness, then describe how to modify it to obtain a BIC mechanism while incurring only a small increase to the social cost.

A.1 The ϵ-BIC Reduction

The ϵ-BIC reduction due to Hartline and Lucier 9 is as follows. Suppose $\mathcal{A}$ is an arbitrary social cost algorithm with (unknown) interim allocation rules $\mathbf{x}$. For each i, we will first partition the value space $[0, 1]$ into $1/\delta$ intervals of width δ; write $I_k = (k\delta, (k+1)\delta]$ for the kth such interval.

Suppose first that we knew the value of $\mathrm{E}_{v_i}[x_i(v_i) \mid v_i \in I_k]$ for each i and k. Given this information, we could perform the following monotonizing operation. We construct a certain partition $\mathcal{P}$ of $[0, 1]$ (into intervals), where $\mathcal{P}$ is a coarsening of the intervals I_k; that is, each interval endpoint in $\mathcal{P}$ will be a multiple of δ. Suppose the intervals in $\mathcal{P}$ are $I'_1, \ldots, I'_\ell$ for some $\ell \leq 1/\delta$. Given $\mathcal{P}$ (whose construction we have not described), we will define algorithm $\overline{\mathcal{A}}$ as follows:

1. For each agent i, if $v_i \in I'_j \in \mathcal{P}$, then draw $v_i{}' \sim F_i$.

2. Return $\mathcal{A}(\mathbf{v}')$

Hartline and Lucier 9 show that there is a way to construct partition $\mathcal{P}$ so that $\overline{\mathcal{A}}$ is BIC, and $\overline{\mathcal{A}}$ has the same social cost as $\mathcal{A}$.

Next suppose that the value of $E_{v_i}[x_i(v_i) \mid v_i \in I_k]$ is not known explicitly for all i and k. In this case, our reduction will attempt to estimate these values. We do so by taking many samples $\mathbf{v} \sim \mathbf{F}$ subject to $v_i \in I_k$ and executing $\mathcal{A}$ on each sampled value profile; our estimate for $E_{v_i}[x_i(v_i) \mid v_i \in I_k]$ will be the fraction of these samples for which $\mathcal{A}$ serves agent i. Chernoff-Hoeffding bounds imply that if we take $O(\frac{1}{\lambda^2} \log(n/\lambda\delta))$ samples per agent and interval, then every estimate will be within λ of the true value with probability at least $1 - \lambda$. Write $\tilde{\mathbf{x}}$ for the estimated allocation curves.

Estimates in hand, we can perform the monotonizing operation described above on the estimated allocation curves, as if they were the actual curves. Hartline and Lucier 9 show that, if this is done, the resulting algorithm is approximately monotone: with probability $(1 - \lambda)$, it is true that for each i, if $\overline{x}_i$ is the interim allocation rule for agent i, then $\overline{x}_i(v_i) \leq \overline{x}_i(v_i) + 2\lambda$ for all $v_i \leq v_i'$. Also, the expected social welfare decreases by at most $(\delta + 2\epsilon)n$ as a result of this reduction. Since social cost is simply $\sum_{i \in [n]} v_i$ minus the social welfare, this implies the social cost increases by at most ϵ for an appropriate choice of λ and δ.

A.2 Obtaining a BIC Reduction

Suppose that $\overline{\mathcal{A}}$ is an ϵ-BIC algorithm, constructed via the reduction described above. An important fact about $\overline{\mathcal{A}}$ is that it has allocation rules that are piecewise-constant on the intervals I_k. Thus, any non-monotonicities in an interim allocation curve of $\overline{\mathcal{A}}$ can occur only at finitely many possible input values; specifically, at multiples of δ. Our approach for modifying $\overline{\mathcal{A}}$ to be BIC will therefore be to modify its allocation curve in a blatantly monotone way over the intervals between these multiples of δ. Specifically, we will reduce the probability of allocating to an agent i by ϵk whenever he declares a value on interval k. Since there are only $1/\delta$ such intervals, the overall increase in social cost due to this change will be small.

More formally, we perform the following modification to algorithm $\overline{\mathcal{A}}$. Our new algorithm, $\tilde{\mathcal{A}}$, proceeds as follows, where we set $\gamma = 2\epsilon/\delta$.

1. With probability $1 - \gamma$, return $\overline{\mathcal{A}}(\mathbf{v})$.

2. Otherwise, choose an agent i uniformly at random. If $v_i \in I_k$, then return $\{i\}$ with probability $k\delta$, otherwise return $\emptyset$.

Claim 16. *If $\overline{\mathcal{A}}$ is ϵ-BIC and piecewise constant on intervals I_k, then $\tilde{\mathcal{A}}$ is BIC.*

Proof. Write $\tilde{x}_i$ for the interim allocation rule of $\tilde{\mathcal{A}}$. Choose any $1 \leq k < 1/\delta$ and suppose $v \in I_k$ and $v' \in I_{k+1}$. Then

$$
\begin{aligned}
\tilde{x}_i(v) &= (1 - \gamma)\overline{x}_i(v) + \gamma k\delta \\
&\leq (1 - \gamma)(\overline{x}_i(v') + 2\epsilon) + \gamma k\delta \\
&\leq (1 - \gamma)\overline{x}_i(v') + 2\epsilon + \gamma(k + 1)\delta - \gamma\delta \\
&= (1 - \gamma)\overline{x}_i(v') + \gamma(k + 1)\delta \\
&= \tilde{x}_i(v')
\end{aligned}
$$

and hence $\tilde{x}_i$ is monotone, as required. $\square$

Claim 17. *The expected social cost of $\tilde{\mathcal{A}}$ is at most the expected social cost of $\overline{\mathcal{A}}$ plus γn.*

Proof. Note that we can assume that $C(\{i\}) \leq n$ for all i; otherwise we would never serve i and could remove agent i from the mechanism. We therefore have

$$\mathrm{E}_{\mathbf{v}}[SC(\tilde{\mathcal{A}}, \mathbf{v})] \leq (1 - \gamma)\mathrm{E}_{\mathbf{v}}[SC(\overline{\mathcal{A}}, \mathbf{v})] + \gamma \cdot \max\left\{\sum_i v_i, C(\{1\}), \ldots, C(\{n\})\right\}$$

$$\leq (1 - \gamma)\mathrm{E}_{\mathbf{v}}[SC(\overline{\mathcal{A}}, \mathbf{v})] + \gamma n$$

as required. □

Thus, for $\epsilon' = 2n\epsilon/\delta$, we conclude that $\tilde{\mathcal{A}}$ is BIC and has expected social cost at most that of $\overline{\mathcal{A}}$ plus ϵ'. In other words, given an arbitrary ϵ', we can choose ϵ and δ sufficiently small (but polynomial in ϵ' and $1/n$) such that $\tilde{\mathcal{A}}$ is BIC and increases expected social cost of the original algorithm $\mathcal{A}$ by at most ϵ', as required by Corollary 3.

B Omitted proofs from Section 4

B.1 Proof of Lemma 10

The expected value of M_{ik}/N is precisely $\mathrm{E}_{\mathbf{v}}[x_i(\mathbf{v}) \mid v_i \in I_k]$. By Chernoff-Hoeffding bounds, after N samples the probability that $|M_{ik}/N - \mathbf{E}[M_{ik}/N]| \geq \epsilon$ is at most $\epsilon\delta/n$. Taking the union bound over all i and k, we have that $|M_{ik}/N - \mathbf{E}[M_{ik}/N]| < \epsilon$ for all i and k with probability at most $1 - \epsilon$. This also implies that $|\tilde{x}_i(v_i) - \overline{x}_i(v_i)| < \epsilon$ for all i and v_i, as required. □

B.2 Proof of Lemma 11

Let us first recall the statement of the lemma. Given $\epsilon > 0$ and black-box access to algorithm $\mathcal{A}$, we claim that one can construct a BIC mechanism $\mathcal{M}$ with $\mathbf{E}_{\mathbf{v}}[SC(\mathcal{M})] \leq O(\min\{\log(h), \log(n)\})\, \mathbf{E}_{\mathbf{v}}[SC(\mathcal{A})] + \epsilon$. Moreover, the expected payments in $\mathcal{M}$ are at least $\mathbf{E}_{\mathbf{v}}[C(S(\mathbf{v}))] - \epsilon$.

We first note that discretizing allocation curves along intervals of length δ can increase social cost by at most δn, as each agent's value changes by at most δ as a result of this approximation. We therefore note this increase in social cost and assume for notational convenience that each x_i is constant on each interval I_k.

Our mechanism will proceed by constructing $\tilde{\mathbf{x}}$ as described above, and then apply either Algorithm 1 or Algorithm 2, depending on which of n or h is smaller. In either case, the algorithm will compute some threshold T. The mechanism will proceed by eliciting valuation profile $\mathbf{v}$, querying $\mathcal{A}(\mathbf{v})$, and then serving those agents in the resulting set S with value at least T. Regardless of the threshold returned, this mechanism will be BIC.

In the event that $\tilde{\mathbf{x}}$ is not ϵ-close to $\mathbf{x}$, the social cost generated by the resulting mechanism is trivially bounded by n. Since this event occurs with probability at most ϵ, its contribution to the expected social cost is at most ϵn. We therefore assume that $\tilde{\mathbf{x}}$ and $\mathbf{x}$ are ϵ-close.

For either algorithm, the threshold T is chosen so that expected payments, as computed from $\tilde{\mathbf{x}}$, recover expected costs in expectation. Since each $\tilde{x}_i$ is ϵ-close to the true curve x_i, the estimated payments differ from the true payments by at most ϵ, for each agent. This has two effects: first, in our analysis of each algorithm, bounds on the increase to social cost include an error of up to ϵ per agent, as $\tilde{x}_i(v_i) \leq x_i(v_i) + \epsilon$ for all i. Thus, there can be up to an additional ϵn increase in social cost due to the threshold T applied.

Second, the true BIC payment (from Lemma 1) may differ from the approximate BIC payment by up to ϵ, for each agent. Thus, the expected payments of mechanism $\mathcal{M}$ may be up to ϵn less than was computed by either algorithm. In particular, it may be that the expected payments are as low as $\mathbf{E_v}[C(S(\mathbf{v}))] - \epsilon n$.

To summarize, our resulting mechanism will have $\mathbf{E_v}[SC(\mathcal{M})] \leq O(\min\{\log(h), \log(n)\}) \mathbf{E_v}[SC(\mathcal{A})] + (\delta + 2\epsilon)n$, and will have expected payments at least $\mathbf{E_v}[C(S(\mathbf{v}))] - \epsilon n$. Taking an appropriate choice of ϵ and δ then completes the proof. $\square$

C   Proof of Theorem 12

It is well known in auction theory (e.g. Myerson 15) that, from an agent whose valuation is drawn from the equal revenue distribution, a BIC mechanism can extract a payment of at most 1 in expectation. Therefore, any BIC mechanism for Example 1 can collect payments at most $\frac{1}{4}$ in expectation. Since the mechanism recovers cost in expectation, the expected cost must be at most $\frac{1}{4}$ as well. But unless $S = \emptyset$, $C(S) = 1$. Therefore,

$$\Pr[S \neq \emptyset] \leq \frac{1}{4}.$$

Let V be $\sum_i v_i$. Observe that

$$\mathbf{E}[V] = n \mathbf{E}[v_i] = \frac{\log h}{4}, \quad \mathrm{Var}[V] = n \mathrm{Var}[v_i] \leq \frac{h}{16n}, \quad \sigma(V) \leq \frac{1}{4}\sqrt{h/n}.$$

By Chebyshev's inequality, we have

$$\mathbf{Pr}\left[V < \frac{\log(h) - 2\sqrt{h/n}}{4}\right] \leq \frac{1}{4}.$$

The expected social cost of a cost recovering BIC mechanism $\mathcal{M}$ is at most:

$$\mathbf{E}[SC(\mathcal{M})] \geq \frac{\log(h) - 2\sqrt{h/n}}{4} \mathbf{Pr}\left[S = \emptyset \wedge V \geq \frac{\log(h) - 2\sqrt{h/n}}{4}\right].$$

By the union bound, the latter probability is at least $\frac{1}{2}$. Therefore,

$$\mathbf{E}[SC(\mathcal{M})] \geq \frac{\log(h) - 2\sqrt{h/n}}{8}.$$

$\square$

D   Omitted Proofs from Section 6

D.1   Proof of Theorem 13

The mechanism output by Algorithm 3 can be clearly seen to be truthful: an agent with value 0 never wins, and an agent with value 1 gets a zero utility, and so no agent has motivation to misreport his value. It recovers cost because it serves agents only if the cost can be recovered. Also, as $\widehat{S}(\mathbf{v}) \subseteq S(\mathbf{v})$, if $\widehat{S}$ is served, then the social cost is less than that $C(\mathcal{A})$ since the agents in $S(\mathbf{v}) - \widehat{S}(\mathbf{v})$ does not add to the social cost; on the other hand, if no agents are served, the change in social cost is $C(\widehat{S}(\mathbf{v})) - |\widehat{S}(\mathbf{v})| < 0$. $\square$

D.2 Proof of Theorem 14

It is easy to see that the procedure in Algorithm 4 guarantees cost recovery. To see that it is truthful, note that if an agent is served, then misreporting his valuation leads either to non-service (to his disadvantage) or to service with the same cost and payment (by the no bossiness of $\mathcal{M}$); if an agent is not served, he will not have an incentive to overreport his valuation to be served because he would not do that in $\mathcal{M}$ (because $\mathcal{M}$ is truthful) and now the payment is even higher than in $\mathcal{M}$. Now, similarly to the proof of Theorem 5, we need only to bound the additional social cost inflicted by refusing service to bidders with valuations no more than 2^k.

By the way k is determined, we have

$$2^j|S_j(\mathbf{v})| < C(S_j(\mathbf{v})), \quad \forall j < k. \tag{2}$$

Using this, we have

$$\sum_{i \in S(\mathbf{v}) \backslash S_k(\mathbf{v})} v_i = \sum_{j=0}^{k-1} \sum_{i \in S_j(\mathbf{v}) \backslash S_{j+1}(\mathbf{v})} v_i \leq \sum_{j=0}^{k-1} 2^{j+1}|S_j(\mathbf{v})| \leq \sum_{j=0}^{k-1} 2C(S_j(\mathbf{v})) \leq O(\log h)C(\mathcal{M}).$$

$\square$