Designing And Learning Optimal Finite Support Auctions [⋆]

Edith Elkind

Department of Computer Science, University of Liverpool

Abstract. A classical paper of Myerson [18] shows how to construct an optimal (revenue-maximizing) auction in a model where bidders' values are drawn from known continuous distributions. In this paper we show how to adapt this approach to finite support distributions that may be partially unknown. We demonstrate that a Myerson-style auction can be constructed in time polynomial in the number of bidders and the size of the support sets. Next, we consider the scenario where the mechanism designer knows the support sets, but not the probability of each value. In this situation, we show that the optimal auction may be learned in polynomial time using a weak oracle that, given two candidate auctions, returns one with a higher expected revenue. To study this problem, we introduce a new class of truthful mechanisms which we call order-based auctions. We show that the optimal mechanism is an order-based auction and use the internal structure of this class to prove the correctness of our learning algorithm as well as to bound its running time.

1 Introduction

The problem of designing an *optimal* single-item auction, i.e., finding a way to allocate an object to one of the n potential buyers so as to maximize the seller's revenue is fundamental to auction theory. Arguably, for many practical applications, revenue maximization is more important than *efficiency*, i.e., assigning the object to the bidder who values it most, and, perhaps surprisingly, efficient auctions are not always optimal and vice versa. As most of the standard auction formats are efficient, to extract the maximum profit for the seller one needs to construct a more sophisticated mechanism that is specifically tailored to this task.

The first optimal auction for the simplest possible case of independent values and continuous distributions was described by Myerson in 1981 [18]. In Myerson's paper, it is assumed that the mechanism designer knows the bidders' (continuous) value distributions (the particular realization of each bidder's value, however, is known to this bidder only, as is common in auction theory); the optimal allocation and payment rule depend on this information. More specifically, for each bidder i, Myerson's auction computes the *virtual bid* $c_i = c_i(w_i, \mathcal{D}_i)$ based on his actual bid w_i and value distribution $\mathcal{D}_i$. It then allocates the object to the bidder with the highest virtual bid; the price that the winner pays is the smallest amount he could bid and still win the auction, i.e., his *threshold bid*. This payment rule ensures that the auction is truthful; in particular, it guarantees that the highest bidder i cannot profit from bid shading, i.e., submitting a lower bid w' that is feasible, i.e., belongs

[⋆] Supported by the EPSRC research grants "Algorithmics of Network-sharing Games" and "Discontinuous Behaviour in the Complexity of randomized Algorithms". Part of this research was done while the author was at Princeton University.

to the support of the ith bidder's distribution, and still satisfies $c_i(w', \mathcal{D}_i) > c_j$, where c_j is the second-highest virtual bid.

In this paper, we consider the task of constructing an optimal mechanism under a set of assumptions that differ from those made by Myerson in two important aspects: first, we focus on the case when the bidders' value distributions are discrete and second, we assume that these distributions are only partially known to the mechanism designer.

We start by showing how to adapt Myerson's approach to the situation when bidders' valuations are drawn from finite support distributions, i.e., bidder i's value for the object is an element of a finite set $W_i = \{w_i^1, \dots, w_i^K\}$. While this model may be better suited for many real-life situations than the continuous one (we provide several examples where this is likely to be the case), the proofs in [18] make use of the continuity assumption and cannot be applied directly in our case. Also, our auctions can be designed and implemented in polynomial time, while the running time of Myerson's auctions depends on the choice of representation for the continuous distributions; in particular it is not clear if the "flattening" procedure discussed in subsequent sections (also known as "ironing") can be carried out in polynomial time.

We then address what many consider a significant shortcoming of Myerson's approach, namely, the assumption that the mechanism designer has full information about all bidders' value distributions. We consider a learning scenario, in which we are given the set of each bidder's possible values, but not the probability of each value, and are allowed to run a sequence of auctions and to observe their outcomes. (A more general scenario, in which the information about the distribution supports is absent, too, is the subject of future work.) We assume that the bidders are not strategic, i.e., in each round they behave as if this were a single-shot game. While this assumption may appear unrealistic, it is justified in the case when rather than having n bidders who repeatedly participate in our auctions, we have n bidder *types*, each corresponding to a finite support distribution, and in each round we observe a new participant of each type.

Our main result is that it is possible to learn the optimal auction even if we only get to observe the expected profit of each auction. Indeed, the problem would have been much easier if we had full access to bid statistics, as this would allow us to estimate the probability of each value. However, in practice the bidders are often reluctant to reveal their true values and insist on some form of data protection. For example, they may require that the auction is run by a trusted third party or uses cryptography to preserve data integrity; in fact, cryptographic auction mechanisms have recently received a lot of attention [19, 17]. If this is indeed the case, the only course of action available to the mechanism designer is to try out several candidate solutions, observe their expected profits, and pick the best one. We show that in this setting, there is a way to choose the candidate mechanisms adaptively so as to arrive at the optimal solution after a polynomial number of trials.

We model the scenario when the auction is run by a third party by considering an oracle that given a description of an auction mechanism, outputs the expected profit of this mechanism. We emphasise that the term 'oracle' is only used to highlight the fact that the we get no extra infortmation about the outcome of the auction; our oracle does not have superpolynomial computational power and can be implemented by running the auction sufficiently many times. First, we consider the simplified model where the oracle returns the exact value of the expected profit. We then generalize our results to a more realistic setting, where the oracle returns an ϵ-approximation to the expected profit.

To use this model (and, more generally, to be able to subcontract running the auction), we must be able to encode an auction succinctly. This is not a trivial task: the total number of possible bid vectors is K^n, and therefore the number of admissible allocation rules is exponential in K^n. To resolve this problem, we propose a new class of truthful mechanisms for finite support value distributions, which we call *order-based auctions*, and show that the optimal mechanism belongs to this class. Order-based auctions have many other attractive properties: besides having a compact representation (namely, an order-based auction can be viewed as a permutation of size $nK + 1$), they allow efficient winner determination and payment computation, and are easy to implement in a cryptographically secure manner. Therefore, we believe that this concept may be of independent interest, especially in relation to designing revenue-maximizing mechanisms for the case when bidders' values are not necessarily independent. By choosing our candidate solutions among order-based auctions, we reduce the size of the search space to just $(nK+1)!$; also, it allows us to provide the oracle with a compact description of an auction. We then show that to find an optimal auction, it suffices to make $O(n^2K^2)$ calls to an even weaker oracle that can compare the expected profits of any two auctions (clearly, this oracle can be simulated by the expected profit oracle). This means that the optimal auction can be learned under truly minimal informational assumptions.

Related work To the best of our knowledge, the problem of constructing an optimal auction for finite support distributions was first addressed by Bergemann and Pesendorfer [5], who describe a solution for distributions which satisfy an additional *regularity* constraint [18]; our arguments for the regular case closely follow those of [5], and are provided here for completeness only. A solution for the general case was obtained simultaneously and independently by Cheng [8]; however, he does not consider the incomplete information scenario studied in Section 5. A number of papers have investigated learning and profit maximization in online auctions (see, e.g., [12, 13, 4, 6, 15, 14]). However, in most of these papers it is assumed that the bidders are symmetric, while we are interested in the case when all n bidders have different value distributions. A more general approach is taken by Aggarwal and Hartline [1], Blum and Hartline [3] and Balcan et al. [2], who consider attribute auctions, in which bidders may have publicly known attributes; our model can be viewed within this framework, where attributes can be identified with bidders' value distributions (or, in the case of learning scenario, the support sets). Ronen [20] and Ronen and Saberi [21] study optimal auctions in the case where bidders' value distributions are not necessarily independent; however, their lower bounds do not apply in our scenario. The results presented in this paper have appeared in [10].

2 Preliminaries and notation

We consider the setting in which n bidders $1, \dots, n$ compete for a single object. All bidders' values are independent random variables; the value v_i of the bidder i is drawn from a finite set $W_i =$

$\{w_i^k \mid k = 1, \dots, K\}$, according to a distribution $\mathcal{D}_i$; we assume $w_i^1 < \cdots < w_i^K$. The distribution $\mathcal{D}_i$ is completely described by its set of mass points W_i and their probabilities $g_i^1, \dots, g_i^K$, that is, $\mathbf{Pr}[v_i = w_i^k] = g_i^k$. We assume $g_i^k > 0$ for all $i = 1, \dots, n$, $k = 1, \dots, K$, and set $G_i^k = g_i^1 + \cdots + g_i^k$. Our results can be easily generalized to the case when sets $W_1, \dots, W_n$ have different sizes. Whenever we discuss computational efficiency, we assume that all g_i^k, w_i^k are rational numbers whose representation size is polynomial in n and K. In what follows, 'polynomial' always means 'polynomial in the size of the problem description', i.e., 'polynomial in n and K'.

Set $W = W_1 \times \cdots \times W_n$, $\mathcal{D} = (\mathcal{D}_1, \dots, \mathcal{D}_n)$; an element of W is denoted by $\mathbf{w} = (w_1, \dots, w_n)$. Set $\mathbf{w}_{-i} = (w_1, \dots, w_{i-1}, w_{i+1}, \dots, w_n)$ and $(\mathbf{w}_{-i}, w) = (w_1, \dots, w_{i-1}, w, w_{i+1}, \dots, w_n)$.

We assume that the number of bidders and their value distributions are common knowledge (we relax this assumption in Section 5); however, the actual valuation of each bidder is known to this bidder only. Our goal is to design an optimal, i.e., revenue-maximizing mechanism for the pair $\mathcal{G} = (W, \mathcal{D})$ that is *individually rational*, i.e., no bidder can lose money by participating in the auction. By the revelation principle [18], we can restrict ourselves to incentive-compatible mechanisms, in which the equilibrium strategy for each bidder is to reveal his true value. An incentive-compatible mechanism is completely defined by its allocation rule $Q : W \mapsto [0,1]^n$ and payment rule $M : W \mapsto \mathbb{R}^n$: given a bid vector $\mathbf{w}$, $Q_i(\mathbf{w})$ is the probability that bidder i wins, and $M_i(\mathbf{w})$ is the payment to bidder i. For any $\mathbf{w} \in W$ we have $\sum_{i=1}^n Q_i(\mathbf{w}) \leq 1$, $Q_i(\mathbf{w}) \geq 0$. Note that we do not require that one of the bidders wins, i.e., we allow $\sum_{i=1}^n Q_i(\mathbf{w}) < 1$: the seller can keep the object if it is profitable for him to do so.

3 Auction design for known distributions

In this section, we show how to design an optimal finite support auction given a full description of each bidder's value distribution.

First, we show that in the optimal auction, the payment rule (and hence the auctioneer's revenue) can be expressed as a function of the allocation rule and derive the expression for the seller's expected revenue as a function of the allocation rule. Later, we will use this expression to pick the optimal allocation rule.

Define the *virtual utility* c_i^k of the bidder i whose value is w_i^k to be

$$c_i^k = w_i^k - (w_i^{k+1} - w_i^k)\frac{1 - G_i^k}{g_i^k}.$$

Theorem 1. *The seller's revenue from an optimal truthful auction with allocation rule Q can be expressed as*

$$R_Q = \sum_{k_1=1}^{K} \cdots \sum_{k_n=1}^{K} \left[\sum_{i=1}^{n} Q_i(w_1^{k_1}, \dots, w_n^{k_n}) c_i^{k_i} \right] g_1^{k_1} \dots g_n^{k_n}. \quad (1)$$

Proof. Define

$$q_i(w_i) = \sum_{k_1=1}^{K} \cdots \sum_{k_{i-1}=1}^{K} \sum_{k_{i+1}=1}^{K} \cdots \sum_{k_n=1}^{K} Q_i(w_1^{k_1}, \ldots, w_i, \ldots, w_n^{k_n}) g_1^{k_1} \ldots g_{i-1}^{k_{i-1}} g_{i+1}^{k_{i+1}} g_n^{k_n}$$

and

$$m_i(w_i) = \sum_{k_1=1}^{K} \cdots \sum_{k_{i-1}=1}^{K} \sum_{k_{i+1}=1}^{K} \cdots \sum_{k_n=1}^{K} M_i(w_1^{k_1}, \ldots, w_i, \ldots, w_n^{k_n}) g_1^{k_1} \ldots g_{i-1}^{k_{i-1}} g_{i+1}^{k_{i+1}} g_n^{k_n}.$$

That is, $q_i(w_i)$ is the expected probability that bidder i wins if he reports his value as w_i and everyone else draws their values at random and reports them truthfully, and $m_i(w_i)$ is his expected payment under this scenario. To simplify the notation, we set $q_i^k = q_i(w_i^k)$ and $m_i^k = m_i(w_i^k)$.

Given a bid vector $\mathbf{w}$, the *utility* of a truthful bidder i is $U_i(\mathbf{w}) = w_i Q_i(\mathbf{w}) - M_i(\mathbf{w})$. Note that this definition of utility assumes that the payment made by an agent is not conditional on his winning the auction; rather, the probability of this event is incorporated into the payment rule. The individual rationality constraint says that $U_i(\mathbf{w}) \geq 0$ for all $i = 1, \ldots, n$, $\mathbf{w} \in W$. The *expected utility* of a truthful bidder i can be expressed as a function of his expected probability of winning and his expected payment: $u_i(w_i) = w_i q_i(w_i) - m_i(w_i)$. Set $u_i^k = u_i(w_i^k)$.

Since the mechanism is truthful, when bidder i's value is w_i^k, he cannot increase his utility by reporting w_i^{k-1}, and vice versa, that is

$$U_i(\mathbf{w}_{-i}, w_i^k) \geq U_i(\mathbf{w}_{-i}, w_i^{k-1}) + (w_i^k - w_i^{k-1}) Q_i(\mathbf{w}_{-i}, w_i^{k-1})$$

and

$$U_i(\mathbf{w}_{-i}, w_i^{k-1}) \geq U_i(\mathbf{w}_{-i}, w_i^k) - (w_i^k - w_i^{k-1}) Q_i(\mathbf{w}_{-i}, w_i^k).$$

Rewriting these inequalities, we get

$$(w_i^k - w_i^{k-1}) Q_i(\mathbf{w}_{-i}, w_i^{k-1}) \leq U_i(\mathbf{w}_{-i}, w_i^k) - U_i(\mathbf{w}_{-i}, w_i^{k-1}) \leq (w_i^k - w_i^{k-1}) Q_i(\mathbf{w}_{-i}, w_i^k) \quad (2)$$

or, averaging over other bidders' values,

$$(w_i^k - w_i^{k-1}) q_i^{k-1} \leq u_i^k - u_i^{k-1} \leq (w_i^k - w_i^{k-1}) q_i^k.$$

Hence, for any incentive compatible mechanism there exist $\tilde{q}_i^j \in [q_i^{j-1}, q_i^j]$ such that

$$u_i^k = u_i^1 + \sum_{j=2}^{k} (w_i^j - w_i^{j-1}) \tilde{q}_i^j.$$

As $u_i^k = w_i^k q_i^k - m_i^k$, the expected payment of the ith bidder when his value is w_i^k equals

$$m_i^k = w_i^k q_i^k - u_i^1 - \sum_{j=2}^{k} (w_i^j - w_i^{j-1}) \tilde{q}_i^j. \quad (3)$$

Therefore, if his value is drawn from W_i according to $\mathcal{D}_i$, his expected payment is computed as

$$P_i = \sum_{k=1}^{K} m_i^k g_i^k = -u_i^1 + \sum_{k=1}^{K} w_i^k q_i^k g_i^k - \sum_{k=1}^{K} g_i^k \sum_{j=2}^{k} (w_i^j - w_i^{j-1}) \tilde{q}_i^j,$$

or, changing the order of summation,

$$P_i = -u_i^1 + \sum_{k=1}^{K} \left[w_i^k - (w_i^{k+1} - w_i^k) \frac{1 - G_i^k}{g_i^k} \frac{\tilde{q}_i^{k+1}}{q_i^k} \right] q_i^k g_i^k,$$

where we define $w_i^{K+1} = w_i^K$. Any mechanism that chooses $\tilde{q}_i^{k+1}$ between q_i^k and q_i^{k+1} is truthful, and all such mechanisms allocate the object in the same way, so any choice of $\tilde{q}_i^{k+1}$ within these bounds only affects the total payoff, but not the bidders' behavior. As the optimal mechanism maximizes the seller's revenue, it has $\tilde{q}_i^{k+1} = q_i^k$, and, by the same argument, $u_i^1 = 0$ for all $i = 1, \dots, n$, $k = 1, \dots, K$.

Using the definition of virtual utility, we can express the expected payment of the ith bidder in an optimal auction with allocation rule Q as $\sum_{k=1}^{K} c_i^k q_i^k g_i^k$. Using the expression for q_i^k, we derive that the overall seller's revenue from an optimal auction with allocation rule Q is given by (1). □

Theorem 1 implies that we should select Q so as to maximize R_Q. However, our choice is restricted by incentive compatibility and individual rationality constraints. Next, we give a simple description of all allocation rules that can be completed to an incentive compatible and individually rational mechanism.

Definition 1. *An allocation rule* $Q(\mathbf{w}) = (Q_1(\mathbf{w}), \dots, Q_n(\mathbf{w}))$ *is* valid *if it satisfies the following conditions:*

(i) $\sum_{i=1}^{n} Q_i(\mathbf{w}) \le 1$ *for any* $\mathbf{w} \in W$

(ii) $Q_i(\mathbf{w}) \ge 0$ *for any* $\mathbf{w} \in W$, $i = 1, \dots, n$

(iii) $Q_i(\mathbf{w}_{-i}, w_i^k) \le Q_i(\mathbf{w}_{-i}, w_i^{k+1})$.

Proposition 1. *Any incentive-compatible mechanism has a valid allocation rule. Conversely, given a valid allocation rule Q, the mechanism (Q, M) whose payment rule M is given by*

$$M_i(\mathbf{w}_{-i}, w_i^k) = w_i^k Q_i(\mathbf{w}_{-i}, w_i^k) - \sum_{j=2}^{k} (w_i^j - w_i^{j-1}) Q_i(\mathbf{w}_{-i}, w_i^{j-1}) \qquad (4)$$

satisfies incentive compatibility and individual rationality, and its expected revenue is given by (1).

Proof. If (Q, M) is an incentive-compatible mechanism, (i) and (ii) are satisfied by definition, and (iii) follows from (2), which has been shown to hold for all incentive-compatible mechanisms.

For the opposite direction, note that for the payment rule M defined by (4), we have

$$U_i(\mathbf{w}_{-i}, w_i^k) = w_i^k Q_i(\mathbf{w}_{-i}, w_i^k) - M_i(\mathbf{w}_{-i}, w_i^k) = \sum_{j=2}^{k} (w_i^j - w_i^{j-1}) Q_i(\mathbf{w}_{-i}, w_i^{j-1}) \geq 0,$$

which means that (Q, M) satisfies individual rationality. For incentive compatibility, it suffices to show that for any $l < k$

$$U_i(\mathbf{w}_{-i}, w_i^k) \geq U_i(\mathbf{w}_{-i}, w_i^l) + (w_i^k - w_i^l) Q_i(\mathbf{w}_{-i}, w_i^l)$$

and

$$U_i(\mathbf{w}_{-i}, w_i^l) \geq U_i(\mathbf{w}_{-i}, w_i^k) - (w_i^k - w_i^l) Q_i(\mathbf{w}_{-i}, w_i^k).$$

To prove the first inequality, set $Q_i^j = Q_i(\mathbf{w}_{-i}, w_i^j)$ and observe that

$$U_i(\mathbf{w}_{-i}, w_i^k) - U_i(\mathbf{w}_{-i}, w_i^l) = \sum_{j=l+1}^{k} (w_i^j - w_i^{j-1}) Q_i^{j-1} \geq \sum_{j=l+1}^{k} (w_i^j - w_i^{j-1}) Q_i^l = (w_i^k - w_i^l) Q_i^l. \tag{5}$$

The second inequality can be verified in a similar manner.

To analyze the expected revenue, observe that the expected payment of the ith bidder is given by expression (3), in which u_i^1 is set to 0 and $\tilde{q}_i^j = q_i^{j-1}$. Repeating the argument in the proof of Theorem 1, we see that the expected revenue of (Q, M) is given by (1). □

Remark 1. If Q only takes values 0 and 1, the expression for M in Proposition 1 can be simplified to

$$M_i(w_1, \ldots, w_i^k, \ldots, w_n) = w_i^l,$$

where $l = \min\left\{j \mid Q_i(w_1, \ldots, w_i^j, \ldots, w_n) = 1\right\}$. This means that the winner pays the minimal amount that he can bid and still win the auction, i.e., his *threshold bid*.

Regular distributions If for all $i = 1, \ldots, n$, $k = 1, \ldots, K-1$ we have $c_i^k \leq c_i^{k+1}$, i.e., all c_i^k are monotone in k (the value distributions that have this property are called *regular*), the revenue can be maximized pointwise. That is, for any $\mathbf{k} = (k_1, \ldots, k_n)$, we set $I_0 = \operatorname{argmax}\{c_1^{k_1}, \ldots, c_n^{k_n}\}$ and $i_0 = \min\{i \mid i \in I_0\}$ and define $Q_i(w_1^{k_1}, \ldots, w_n^{k_n}) = 1$ if $i = i_0$ and $c_i^{k_i} \geq 0$ and $Q_i(w_1^{k_1}, \ldots, w_n^{k_n}) = 0$ otherwise. It is not hard to see that this Q is valid. Moreover, for any other valid Q' and any $\mathbf{w}$, $\mathbf{k}$ we have

$$\sum_{i=1}^{n} Q_i'(w_1^{k_1}, \ldots, w_n^{k_n}) c_i^{k_i} \leq \sum_{i=1}^{n} Q_i(w_1^{k_1}, \ldots, w_n^{k_n}) c_i^{k_i}$$

and hence $R_{Q'} \leq R_Q$.

Remark 2. This allocation rule resolves draws by giving the object to the lexicographically first bidder among the ones with the highest virtual utility. Any other draw resolution rule, either deterministic or probabilistic, would also produce a valid Q which optimizes the seller's revenue. However, under a probabilistic rule, it may happen that a bidder who does not get the object still pays a non-zero amount. That is, even though the mechanism will be individually rational in expectation (over the coin tosses of Q), particular runs of the mechanism may leave some bidders with negative surplus. To avoid this, we opt for an (admittedly unfair) deterministic rule.

Remark 3. Observe that when all virtual utilities are negative, we do not allocate the object at all. While this may seem counterintuitive, it allows us to charge higher prices when the bidders have high values.

General distributions If the virtual utilities c_i^k are not monotone in k, defining Q in this manner may not produce a valid allocation rule as it may happen that $Q_i(\mathbf{w}_{-i}, w_i^{k-1}) > Q_i(\mathbf{w}_{-i}, w_i^k)$. On the other hand, for any $\bar{c}_i^k$ that are monotone in k, the allocation rule that gives the object to the bidder whose bid maximizes $\bar{c}_i^k$ among all bids is truthful. In what follows, we show that for each i, one can construct a monotone sequence $\bar{c}_i^k$ by "flattening out" $c\,_i^k$ so that the resulting auction is optimal. This technique was first introduced in [18], where it was applied to continuous virtual utility functions (which may be nonmonotone as well).

Intuitively, to compute $\bar{c}_i^k$, we construct a piecewise linear function that consist of K segments whose slopes are $c_i^1, \dots, c_i^K$. If the sequence c_i^k is not monotone, this function is not convex; we then compute its lower envelope (which is convex by definition) and set $\bar{c}_i^k$ to be the slope of the segment of the lower envelope that corresponds to c_i^k. More formally, recall that $G_i^k = g_i^1 + \dots + g_i^k$ and let $H_i^k = c_i^1 g_i^1 + \dots + c_i^k g_i^k$. for $k = 1, \dots, K$. Set $G_i^0 = 0, H_i^0 = 0$, and let $L_i(z) : [0,1] \mapsto \mathbf{R}$ be the lower envelope of the set $\left\{(G_i^0, H_i^0), \dots, (G_i^K, H_i^K)\right\} \subseteq \mathbf{R}^2$, that is,

$$L_i(z) = \min_{\substack{0 \le k, l \le K, \alpha \in [0,1] \\ \alpha G_i^k + (1-\alpha) G_i^l = z}} \left\{ \alpha H_i^k + (1-\alpha) H_i^l \right\}.$$

Set $L_i^k = L_i(G_i^k)$; observe that $L_i^0 = H_i^0 = 0$ and $L_i^K = H_i^K$. Define $\bar{c}_i^k = (L_i^k - L_i^{k-1})/g_i^k$.

Lemma 1. *We have $\bar{c}_i^k \le \bar{c}_i^{k+1}$ for all $i = 1, \dots, n$, $k = 1, \dots, K-1$.*

Proof. The value $\bar{c}_i^k$ is the slope of $L_i(z)$ between L_i^{k-1} and L_i^k. Since $L_i(z)$ is a convex function, the sequence $\bar{c}_i^k$ is nondecreasing in k. □

Theorem 2. *Let $\bar{Q}$ be an allocation rule that gives the object to the lexicographically first bidder among the ones with the highest value of $\bar{c}_i^k$. Then $\bar{Q}$ is an optimal valid allocation rule for $\mathcal{G} = (W, \mathcal{D})$.*

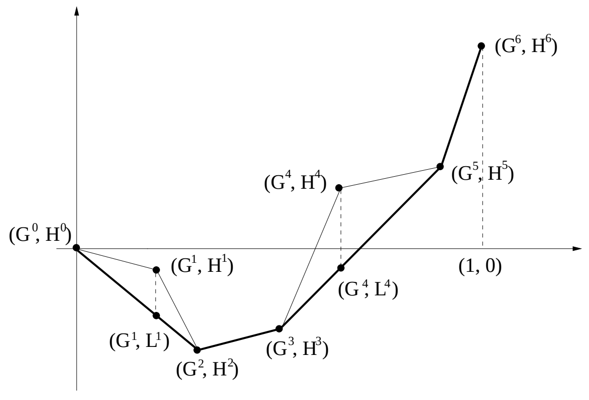

Fig. 1. Nonregular case. Since $c^1 = (H^1 - H^0)/(G^1 - G^0) > (H^2 - H^1)/(G^2 - G^1) = c^2$, we replace both c^1 and c^2 with $\bar{c}^1 = \bar{c}^2 = (H^2 - H^0)/(G^2 - G^0) = (c^1 g^1 + c^2 g^2)/(g^1 + g^2)$.

Proof. Since $\bar{c}_i^k$ are monotone in k, $\bar{Q}$ is a valid allocation rule. To prove optimality, let Q be an arbitrary valid allocation rule on W. Consider the expected payment P_i of the ith bidder under this rule. We have $R_Q = \sum_{i=1}^n P_i$, where

$$P_i = \sum_{k=1}^{K} c_i^k g_i^k q_i^k = \sum_{k=1}^{K} (H_i^k - H_i^{k-1}) q_i^k = -\sum_{k=1}^{K-1} H_i^k (q_i^{k+1} - q_i^k) - H_i^0 q_i^1 + H_i^K q_i^K.$$

Set $\bar{R}_Q = \sum_{i=1}^n \bar{P}_i$, where

$$\bar{P}_i = \sum_{k=1}^{K} \bar{c}_i^k g_i^k q_i^k = -\sum_{k=1}^{K-1} L_i^k (q_i^{k+1} - q_i^k) - L_i^0 q_i^1 + L_i^K q_i^K.$$

Observe that

$$P_i - \bar{P}_i = \sum_{k=1}^{K-1} \left((L_i^k - H_i^k)(q_i^{k+1} - q_i^k) - (H_i^0 - L_i^0) q_i^1 + (H_i^K - L_i^K) q_i^K \right).$$

Since $L_i(z)$ is a lower envelope, $H_i^0 = L_i^0$ and $H_i^K = L_i^K$. Hence,

$$R_Q - \bar{R}_Q = \sum_{i=1}^{n} (P_i - \bar{P}_i) = \sum_{i=1}^{n} \sum_{k=1}^{K-1} (L_i^k - H_i^k)(q_i^{k+1} - q_i^k).$$

Furthermore, for all $i = 1, \ldots, n$, $k = 1, \ldots, K$, we have $L_i^k - H_i^k \leq 0$. Since the sequence q_i^k is monotone in k, this implies that $R_Q \leq \bar{R}_Q$.

Consider the case $Q = \bar{Q}$. By construction, whenever $L_i^k < H_i^k$, i.e., H_i^k does not lie on the lower envelope, the slope of $L_i(z)$ does not change at G_i^k, which means that $\bar{c}_i^k = \bar{c}_i^{k+1}$, and hence $q_i^k = q_i^{k+1}$. Consequently, $R_{\bar{Q}} = \bar{R}_{\bar{Q}}$. Now, let (Q', M') be an arbitrary truthful auction for $(W, \mathcal{D})$. Since Q' must be a valid allocation rule, $R_{Q'} \leq \bar{R}_{Q'}$. On the other hand, $\bar{Q}$ maximizes $\bar{R}_Q$, i.e., $\bar{R}_{Q'} \leq \bar{R}_{\bar{Q}}$. Finally, we have shown that $R_{\bar{Q}} = \bar{R}_{\bar{Q}}$, so $R_{Q'} \leq R_{\bar{Q}}$. Hence, $\bar{Q}$ achieves the maximal revenue for $\mathcal{G} = (W, \mathcal{D})$. □

4 Order-based auctions

In the optimal auction described above, the object is given to the bidder i with the highest flattened virtual utility, who then pays his threshold bid, i.e., $\min\{w_i^k \mid \bar{c}_i^k \geq \bar{c}_j^l\}$, where $\bar{c}_j^l$ is the second highest flattened virtual utility (if j precedes i in the lexicographic ordering, '$\geq$' should be replaced with '$>$'). Alternatively, the optimal auction can be described as follows: all bidders' possible values w_i^k, $i = 1, \ldots, n$, $k = 1, \ldots, K$, are ordered on a line from left to right according to their flattened virtual utilities $\bar{c}_i^k$; also, a cutoff point is selected to separate the points with $\bar{c}_i^k < 0$ from those with $\bar{c}_i^k \geq 0$. The auction then proceeds as follows. All bidders draw their values and report them to the mechanism, which marks the respective points in the ordering; the cutoff point is marked, too. The bidder whose bid corresponds to the rightmost marked point wins; if all bids are to the left of the cutoff point, the item remains unallocated. To determine the winner's payment, the mechanism scans the ordering from left to right starting at the second rightmost marked point till it comes across a (possibly unmarked) point that corresponds to an element of the winner's distribution support; this is the amount that the winner is required to pay.

Clearly, one can combine the same procedure with a different ordering of the $nK + 1$ points. For example, it is easy to see that ordering the points according to their actual values would give rise to the second price auction. Moreover, while not all $(nK + 1)!$ orderings correspond to valid auctions, it is easy to see which ones do: a necessary and sufficient condition is that each bidder's values are placed in increasing order, i.e., for any bidder i and any two elements $w_1 < w_2$, $w_1, w_2 \in W_i$, it should be the case that w_1 is placed before w_2. We formalize this idea as follows.

Definition 2. *For any n bidders $b_1, \ldots, b_n$ whose valuations are drawn from the sets $W_1, \ldots, W_n$ and a bijective function $\pi : \{(i, k) \mid i = 1, \ldots, n, k = 1, \ldots, K\} \cup \{\perp\} \mapsto \{0, \ldots, nK\}$ that satisfies $\pi(i, k) < \pi(i, k + 1)$ for all $i = 1, \ldots, n$, $k = 1, \ldots, K - 1$ (we call any such π a* valid ordering*), an* order-based auction *A_π is conducted as follows. Each bidder i submits his bid $\beta_i = (i, k_i)$ to the center; if $k_i \not\in \{1, \ldots, K\}$, the center rejects it and cancels the auction. If no bids are rejected, set $i^* = \mathrm{argmax}\{\pi(\beta_i) \mid i = 1, \ldots, n\}$. If $\pi(\beta_{i^*}) > \pi(\perp)$, the bidder b_{i^*} wins the auction, otherwise, the item remains unallocated. In the former case, define $p_0 = \max\{\pi(\perp), \max\{\pi(\beta_i) \mid i = 1, \ldots, n, i \neq i^*\}\}$; the* payment *$p$ of the winning bidder b_{i^*} is $\min_k\{w_{i^*}^k \mid \pi(i^*, k) > p_0\}$.*

We abuse notation by writing $\pi(w_i^k)$ instead of $\pi(i, k)$; note, however, that even if $w_i^k = w_j^l$, we treat $\pi(w_i^k)$ and $\pi(w_j^l)$ as distinct objects.

Proposition 2. *Any order-based auction is truthful: if $v_i = w_i^k$, then bidder i's dominant strategy is to bid (i, k).*

Proof. Observe that the allocation rule of any order-based auction is monotone in the bidder's value and the winner pays his threshold bid. It has been shown in [13] that any such auction is truthful; the argument is similar to that used by Vickrey in [22] to prove the truthfulness of the second-price auction. $\square$

Once the optimal ordering is chosen, it can be stored as a table of size $nK + 1$; afterwards, the winner and his payment are determined based on integer comparisons only. This suggests that the order-based representation is likely to be useful if the auction is to be run repeatedly and the value distributions remain unchanged, or if we have plenty of time for pre-processing (and hence can construct π) but then have to run the auction in real time. Also, it can be used to implement the optimal auction in a cryptographically secure manner by generalizing the methods of [19] and [17]. Another important application of this concept is in the incomplete information scenario discussed in the next section.

5 Learning the optimal auction

The results of Section 3 provide a polynomial time algorithm for designing the optimal auction when the distributions $\mathcal{D}_1, \dots, \mathcal{D}_n$ are public knowledge. In this section, we focus on the situation when we know the sets $W_1, \dots, W_n$, but not the distributions $\mathcal{D}_1, \dots, \mathcal{D}_n$; however, we have access to an oracle that given descriptions of two order-based auctions, can tell us which of them has higher expected revenue.

Let $R(A)$ denote the total expected revenue of an auction A. We assume that we have a probabilistic procedure $\texttt{Compare}_{\epsilon,\delta}(\pi, \pi')$ that given any two valid orderings π and π' with probability at least $1 - \delta$ satisfies the following: if $R(A_\pi) < R(A_{\pi'}) - \epsilon$, $\texttt{Compare}_{\epsilon,\delta}(\pi, \pi') = 1$; if $R(A_\pi) > R(A_{\pi'}) + \epsilon$, $\texttt{Compare}_{\epsilon,\delta}(\pi, \pi') = 0$; if $|R(A_\pi) - R(A_{\pi'})| \leq \epsilon$, the procedure may return either 0 or 1.

One way to implement such a procedure is by selling sufficiently many copies of the object using each of the candidate auctions and observing the winners' payments; the number of trials needed to estimate the expected revenue of each auction up to a small additive error can be computed using Chernoff–Hoeffding bounds. This approach assumes that each of the n bidders is willing to buy a large number of units, draws the value of each unit independently from his distribution, and is myopic, i.e., treats each auction as a single-shot game rather than considers the impact of his behavior in this auction on subsequent auctions. Clearly, this situation is unlikely to occur in practice; however, it becomes more plausible if instead of n bidders we consider n bidder *types*, where all bidders of the same type have the same value distribution, and in each auction we observe one representative of each type. For instance, when selling airplane tickets, types may correspond to different social groups (students, soldiers, retired people, businessmen), and the bidders may be considered myopic because they are unlikely to be interested in another ticket from the same provider in the nearest future.

Note, however, that our model is independent of how the comparison oracle is implemented; by subcontracting the task of revenue estimation we eliminate the issue of incentive compatibility and can focus on combinatorial aspects of the problem.

5.1 Monotone virtual utilities

In this subsection, we focus on the case when each bidder's virtual utilities satisfy $c_i^k \le c_i^l$ for any $k < l$. To handle the case $\max_i\{\pi(\beta_i)\} < \pi(\perp)$ together with other cases, we introduce a bidder 0 who always bids $\perp$, i.e., we set $W_0 = \{w_0^0\}$; $c_0^0 = 0$. Let $\pi_{[s,t]}$ be an ordering obtained by swapping sth and tth point: $\pi^{-1}(s) = \pi^{-1}_{[s,t]}(t)$, $\pi^{-1}(t) = \pi^{-1}_{[s,t]}(s)$, where $s, t \in \{0, \dots, nK\}$.

Lemma 2. *Suppose that for some $i \neq j$ and $s \le nK - 1$ we have $\pi(w_i^k) = s$, $\pi(w_j^l) = s + 1$. Then $R(A_\pi) < R(A_{\pi_{[s,s+1]}})$ if and only if $c_i^k > c_j^l$, $\pi(\perp) \le s + 1$, and for all $j' \neq j$ we have $\pi(w_{j'}^1) \le s + 1$.*

Proof. Let Q and Q' be the allocation rules associated with π and $\pi_{[s,s+1]}$, respectively. Recall that the expected revenue of an auction with allocation rule Q is

$$R_Q = \sum_{k_1=1}^{K} \dots \sum_{k_n=1}^{K} \left[\sum_{i=1}^{n} Q_i(w_1^{k_1}, \dots, w_n^{k_n}) c_i^{k_i} \right] g_1^{k_1} \dots g_n^{k_n}.$$

By replacing π with $\pi_{[s,s+1]}$, we only change the allocation rule at points of the form

$$\mathbf{w} = (w_1^{k_1}, \dots, w_i^k, \dots, w_j^l, \dots, w_n^{k_n})$$

that satisfy $Q_j(\mathbf{w}) = 1$ (in particular, if $\pi(w_j^l) < \pi(\perp)$, there are no such points and the revenue does not change). The contribution of any such point to $R(A_\pi)$ is $c_j^l g_1^{k_1} \dots g_n^{k_n}$. For the modified auction, we have $Q_i'(\mathbf{w}) = 1$ for any such $\mathbf{w}$ and hence this point's contribution to $R(A_{\pi_{[s,s+1]}})$ is $c_i^k g_1^{k_1} \dots g_n^{k_n}$. Hence, this interchange increases the total revenue if and only if $c_i^k > c_j^l$ and there is a non-zero probability that the jth bidder can win when bidding w_j^l. It is easy to see that the latter condition is equivalent to stipulating that $\pi(\perp) \le \pi(w_j^l)$ and $\pi(w_{j'}^1) \le \pi(w_{j'}^l)$ for any $j' \neq j$. $\square$

Given access to $\texttt{Compare}_{\epsilon,\delta}(\cdot,\cdot)$, we can find the optimal ordering by a simple `BubbleSort`-like algorithm presented in Figure 2. Essentially, the algorithm attempts to permute two adjacent points and checks if this leads to a higher revenue; this is repeated until there are no more local improvements. The order in which the pairs of points are considered is the same as in `BubbleSort`.

The input to our algorithm is the space W of all possible values. We use the following notation: $\alpha(t) = i$ iff there exists a $k \in [1, K]$ such that $\pi(w_i^k) = t$, i.e., $\alpha(t)$ is the identity of the bidder who "owns" the tth point.

```
LocalOpt$_{\epsilon,\delta}(W)$:

1. Set $\pi(w_i^1) = i$, $i = 0, \ldots, n-1$, $\pi(\perp) = n$, and extend $\pi$ to a valid ordering in an arbitrary way.
2. For $i = 1, \ldots, n-1$:
3.    if Compare$_{\epsilon,\delta}(\pi, \pi_{[n-1,n]}) = 1$,
        set $\pi' = \pi_{[n-1,n]}$,   $\pi = \pi'_{[n-1,n-i-1]}$.
4. For $i = 1, \ldots, nK$:
5.    For $j = 0, \ldots, nK-1$:
6.       if $\alpha(j) \neq \alpha(j+1)$
7.        if Compare$_{\epsilon,\delta}(\pi, \pi_{[j,j+1]}) = 1$, set $\pi = \pi_{[j,j+1]}$.
8. Output $\pi$.
```

Fig. 2. The procedure `LocalOpt`$_{\epsilon,\delta}(W)$

It is easy to analyze the performance of this procedure for the case $\epsilon = 0$, i.e., assuming that with probability $1 - \delta$, the comparison oracle returns a correct answer no matter how small the difference between $R(A_\pi)$ and $R(A_{\pi'})$ is.

Proposition 3. *Let A_{opt} be the optimal auction for W, and let π be the output of* `LocalOpt`$_{0,\delta}(W)$ *Then with probability at least $1 - ((nK)^2 + n)\delta$, we have $R(A_\pi) = R(A_{opt})$.*

Proof. The procedure `Compare`$_{0,\delta}(\cdot,\cdot)$ is called at most $(nK)^2 + n$ times; with probability at least $1 - ((nK)^2 + n)\delta$, each time it returns the correct result. Suppose that this is indeed the case. During the first phase of the algorithm (lines 1–3), we find $\max\{0, \max\{c_i^1 \mid i = 1, \ldots, n\}\}$ and put the corresponding element in the nth position; suppose that this element is x. At this moment, all elements to the left of x have virtual utilities that are smaller than that of x; it is easy to see that this remains true throughout the algorithm. During the second phase (lines 4–7), we transpose jth point and $(j+1)$st point if and only if $j \geq \pi(x)$ and the virtual utility of $\pi^{-1}(j)$ is greater than the virtual utility of $\pi^{-1}(j+1)$ (this includes the case $\alpha(j) = \alpha(j+1)$: we do not permute the points, and by monotonicity, the virtual utility of $\pi^{-1}(j)$ is less than the virtual utility of $\pi^{-1}(j+1)$). Using a standard proof of correctness for `BubbleSort`, we can see that in the end the points to the right of x (including x itself) are sorted according to their virtual utility. The relative ordering of the points to the left of x does not matter, since irrespective of it, they contribute 0 to the total revenue. Therefore, one can permute these points to transform the ordering produced by the algorithm into the optimal ordering (where *all* points are sorted according to their virtual utility) without changing the total revenue, which means that the ordering produced by the algorithm is optimal. □

This analysis can be extended to the case $\epsilon > 0$.

Theorem 3. *Let A_{opt} be the optimal auction for W, and let π be the output of* `LocalOpt`$_{\epsilon,\delta}(W)$. *Then with probability at least $1 - ((nK)^2 + n)\delta$, we have $R(A_\pi) \geq R(A_{opt}) - 2(nK + 1)\epsilon$.*

Proof. First, we show that `BubbleSort` performs well even given a "faulty" comparison oracle, i.e., it returns an ordering that is not very different from the true one. Next, we estimate the profit of an order-based auction that is "close" to optimal.

Proposition 4. *Let $A = \{a_1, \dots, a_n\}$ and set $a^{(1)} = \min\{a_i \mid a_i \in A\}$, $a^{(2)} = \min\{a_i \mid a_i \in A, a_i \neq a^{(1)}\}$, …, $a^{(n)} = \max\{a_i \mid a_i \in A\}$. Suppose that we attempt to sort A in increasing order using* `BubbleSort`*, but instead of comparing elements of A directly, we use a deterministic procedure $\texttt{Comp}_\epsilon(x, y)$, which returns 1 if $y < x - \epsilon$ and 0 if $y > x + \epsilon$. If $|x - y| \leq \epsilon$, the procedure can return either 0 or 1; however, we require that $\texttt{Comp}_\epsilon(x, y)$ is antisymmetric, i.e., $\texttt{Comp}_\epsilon(x, y) = 1 - \texttt{Comp}_\epsilon(y, x)$. Let $(b^{(1)}, \dots, b^{(n)})$ be the output of $\texttt{BubbleSort}(A, \texttt{Comp}_\epsilon)$. Then for any $i \in [1, n]$ we have $|a^{(i)} - b^{(i)}| \leq n\epsilon$.*

Proof. Let i be the variable used in the outer loop of `BubbleSort`, and let j be the variable used in the inner loop. In what follows, the actions of the algorithm when $i = i_0$ are referred to as the i_0th stage of the algorithm, and the comparison (and, possibly, permutation) of jth and $(j+1)$st element during the ith stage is referred to as the jth step of the ith stage.

We will show that if after i stages of the algorithm the elements are ordered as $(b_1, \dots, b_n)$ then for $i' = n - i + 1, \dots, n$ we have $\texttt{Comp}_\epsilon(b_{i'-1}, b_{i'}) = 0$; the proof proceeds by induction on i.

First, consider the base case $i = 1$. At the $(n-1)$st step, we compare the $(n-1)$st element x with the nth element y. If $x < y - \epsilon$, then $\texttt{Comp}_\epsilon(x, y) = 0$, both elements stay in place, and hence $\texttt{Comp}_\epsilon(b_{n-1}, b_n) = 0$. If $x > y + \epsilon$, then $\texttt{Comp}_\epsilon(x, y) = 1$; when we set $b_n = x$, $b_{n-1} = y$, we obtain $\texttt{Comp}_\epsilon(b_{n-1}, b_n) = 0$. If $|x - y| \leq \epsilon$, we interchange x and y if and only if $\texttt{Comp}_\epsilon(x, y) = 1$, therefore $\texttt{Comp}_\epsilon(b_{n-1}, b_n) = 0$.

Now, suppose that the statement is true for all $i' < i$. Suppose that at the beginning of the ith stage the set A is ordered as $(a_1, \dots, a_n)$. By the induction hypothesis, we have $\texttt{Comp}_\epsilon(a_{i'-1}, a_{i'}) = 0$ for all $i' \in [n-i+2, n]$. Clearly, the elements $a_{n-i+1}, \dots, a_n$ remain in place till the $(n-i)$th step. Let x be the element in the $(n-i)$th position before the $(n-i)$th step. Set $k = \min\{j \mid n-i \leq j \leq n-1, \texttt{Comp}_\epsilon(u, v) = 0\}$, where u and v are the elements in the jth and $(j+1)$st position before the jth step; if $\texttt{Comp}_\epsilon(u, v) = 0$ for all $j \in [n-i, n-1]$, set $k = n$. Until the kth step, we have $u = x$, $v = a_{j+1}$: since $\texttt{Comp}_\epsilon(x, a_{j+1}) = 1$, we transpose x and a_{j+1}, and therefore at the next step, we are comparing x and a_{j+2}. During the kth step, x and a_{k+1} are not transposed. Therefore, during the subsequent steps we compare a_j and a_{j+1}; by the induction hypothesis, $\texttt{Comp}_\epsilon(a_j, a_{j+1}) = 0$, so we do not transpose them.

Hence, after the end of the ith stage, we have $b_{n-i} = a_{n-i+1}, \dots, b_{k-1} = a_k$, $b_k = x$, $b_{k+1} = a_{k+1}, \dots, b_n = a_n$. By the induction hypothesis, we have $\texttt{Comp}_\epsilon(b_{j-1}, b_j) = 0$ for all $j \in [n-i+1, k-1] \cup [k+2, n]$. Moreover, by construction, $\texttt{Comp}_\epsilon(b_j, b_{j+1}) = 0$ for $j = k, k+1$.

Therefore, for the output of the sorting algorithm we have $\texttt{Comp}_\epsilon(b^{(i)}, b^{(i+1)}) = 0$ for all $i = 1, \dots, n-1$, and hence $b^{(i)} \leq b^{(i+1)} + \epsilon$.

To conclude the proof, we need the following lemma.

Lemma 3. *Suppose $\alpha_1 \leq \cdots \leq \alpha_n$, and let $(\beta_1, \dots, \beta_n)$ be a permutation of the set $A = \{\alpha_1, \dots, \alpha_n\}$ that satisfies $\beta_1 \leq \beta_2 + \epsilon \leq \cdots \leq \beta_n + (n-1)\epsilon$. Then we have $|\alpha_i - \beta_i| \leq n\epsilon$ for any $i = 1, \dots, n$.*

Proof. Consider an element β_i. There are at least i elements of A (namely, $\beta_1, \ldots, \beta_i$) that satisfy $\alpha \leq \beta_i + (i-1)\epsilon$. Therefore, the i smallest elements of A, and, in particular, α_i satisfy this inequality and hence $\alpha_i \leq \beta_i + (i-1)\epsilon$.

Similarly, there are at least $n-i+1$ elements of A (namely, $\beta_i, \ldots, \beta_n$) that satisfy $\alpha + (n-i)\epsilon \geq \beta_i$. Therefore, the $n-i+1$ largest elements of A, and, in particular, α_i satisfy this inequality and hence $\alpha_i \geq \beta_i - (n-i)\epsilon$. $\square$

Applying the lemma to $(a^{(1)}, \ldots, a^{(n)})$ and $(b^{(1)}), \ldots, b^{(n)})$, we obtain the desired result. $\square$

Lemma 4. *Consider an order-based auction A' for bidders $b_1, \ldots, b_n$ that uses an ordering π determined by monotone nondecreasing functions $d_i : W_i \to \mathbf{R}$, $i = 1, \ldots, n$, i.e., for any $i \neq j$ we have $\pi(\perp) < \pi(w_i^k) < \pi(w_j^l)$ if and only if $0 < d_i(w_i^k) < d_j(w_j^l)$, and $\pi(w_i^k) < \pi(w_i^{k'})$ for any $k < k'$. Suppose also that bidder i's virtual utility $c_i(\cdot)$ is monotone and for all $k = 1, \ldots, K$ we have $|c_i(w_i^k) - d_i(w_i^k)| \leq \epsilon$. Then $R(A') \geq R(A) - 2\epsilon$, where A is an optimal order-based auction for $(W, \mathcal{D})$.*

Proof. Let Q be the allocation rule associated with A. We have seen that

$$R(A) = \sum_{k_1, \ldots, k_n = 1}^{K} \left(\sum_{i=1}^{n} Q_i(w_1^{k_1}, \ldots, w_n^{k_n}) c_i^{k_i} \right) g_1^{k_1} \ldots g_n^{k_n}.$$

Fix a bid vector $\mathbf{w} = (w_1^{k_1}, \ldots, w_n^{k_n})$. Set $d_i^k = d_i(w_i^k)$. In the case of the optimal auction, $Q_i(\mathbf{w}) = 1$ if and only if $c_i^{k_i} = \max\{c_1^{k_1}, \ldots, c_n^{k_n}, 0\}$; consequently, if A allocates the object to bidder i, this event's contribution to the total revenue is $c_i^{k_i} g_1^{k_1} \ldots g_n^{k_n}$.

The auction A' may allocate the object to a bidder j, $j \neq i$, if $d_j^{k_j} = \max\{d_1^{k_1}, \ldots, d_n^{k_n}, 0\}$, in which case the contribution to the total revenue is $c_j^{k_j} g_1^{k_1} \ldots g_n^{k_n}$. Then we have $c_j^{k_j} + \epsilon \geq d_j^{k_j} \geq d_i^{k_i} \geq c_i^{k_i} - \epsilon$, and therefore, $c_j^{k_j} \geq c_i^{k_i} - 2\epsilon$. Similarly, if A' does not allocate the object at all, we have $d_i^{k_i} < 0$, $c_i^{k_i} \leq d_i^{k_i} + \epsilon$, and hence $c_i^{k_i} < \epsilon$, and if A' allocates the object to a bidder j who bids $w_j^{k_j}$, but under A the object remains unallocated, we have $c_j^{k_j} \geq d_j^{k_j} - \epsilon$, $d_j^{k_j} \geq 0$; in both of these cases the total loss of revenue is at most $\epsilon g_1^{k_1} \ldots g_n^{k_n}$.

Summing over all possible values of $\mathbf{w}$, we see that using the ordering based on $d_1(\cdot), \ldots, d_n(\cdot)$ rather than $c_1(\cdot), \ldots, c_n(\cdot)$ decreases the total revenue by at most 2ϵ. $\square$

Remark 4. A similar statement can be proved if the error in valuations is multiplicative rather than additive: if the estimated virtual utilities $d_i(w_i^k)$ satisfy $(1-\epsilon)c_i(w_i^k) \leq d_i(w_i^k) \leq (1+\epsilon)c_i(w_i^k)$, we have $R(A') \geq \frac{1-\epsilon}{1+\epsilon} R(A)$.

The rest of the proof is similar to that for the case $\epsilon = 0$. The only difficulty is that in Proposition 4 it is assumed that the comparison procedure is deterministic and antisymmetric, while $\texttt{Compare}_{\epsilon,\delta}(\pi, \pi')$ has neither of these properties (and, indeed, if it is based on a Monte Carlo algorithm, these properties cannot be guaranteed). This can be resolved by caching the results of the previous calls: given some π and $\pi' = \pi_{[j,j+1]}$, where $\pi^{-1}(j) = x$, $\pi^{-1}(j+1) = y$, we check whether $\texttt{Compare}_{\epsilon,\delta}()$ has been called before on some π_1 and $\pi'_1 = \pi_{[k,k+1]}$ such that $\pi_1^{-1}(k) = x$, $\pi_1^{-1}(k+1) = y$, or $\pi_1^{-1}(k) = y$, $\pi_1^{-1}(k+1) = x$. In the former case, we return $\texttt{Compare}_{\epsilon,\delta}(\pi_1, \pi'_1)$; in the latter case, we return $1 - \texttt{Compare}_{\epsilon,\delta}(\pi_1, \pi'_1)$. If no such π_1, π'_1 were found, we call $\texttt{Compare}_{\epsilon,\delta}(\pi, \pi')$.

Also, as in Proposition 3, the relative order of the elements that end up to the left of x does not matter, and the virtual utility of any such element cannot exceed the virtual utility of x by more than ϵ. In particular, we can permute these points without affecting the total revenue so that the condition of Lemma 3 is satisfied. □

Even though `BubbleSort` is not among the fastest sorting algorithms, we chose to focus on a `BubbleSort`-based procedure, because it provides a better model for learning in a real-life scenario: an unsophisticated seller is likely to prefer a greedy algorithm, which allows him to search for a good auction by local improvement. Showing that the optimal auction can be found in this manner is an argument for practical applicability of our model.

If, on the other hand, we care about computational efficiency, we can achieve a better running time by using `MergeSort`: by monotonicity, we can assume that each bidder's points are already sorted, and all that we have to do is to merge these n arrays of size K each. To merge arbitrary sorted arrays of bids, we need to be able to compare c_i^k and c_j^l for all $i < j, i, j = 1, \dots, N$ and $k, l = 1, \dots, K$. This can be done by constructing a valid ordering π in which c_i^k and c_j^l are adjacent and $q_i^k, q_j^l \neq 0$. Using this approach, we can find the optimal ordering using $O(nK \log n)$ comparisons.

5.2 Nonmonotone virtual utilities

If some bidders' utilities are not monotone in k, we may be unable to compare some of the elements: if $\alpha(j) = \alpha(j+1)$, then $\pi_{[j,j+1]}$ is not a valid ordering, so we cannot call $\texttt{Compare}_{\epsilon,\delta}(\pi, \pi_{[j,j+1]})$, and we are not guaranteed that the virtual utility of $\pi^{-1}(j)$ is at most the virtual utility of $\pi^{-1}(j+1)$. Fortunately, it turns out that given access to $\texttt{Compare}_{\epsilon,\delta}(\pi, \pi')$ we can design a procedure that is capable of comparing flattened virtual utilities $\bar{c}_i^k$, $\bar{c}_j^l$ as long as $i \neq j$; if $i = j$, then by definition $\bar{c}_i^k \leq \bar{c}_i^{k'}$ as long as $k < k'$. Therefore, we can apply any sorting algorithm that is based on pairwise comparisons, e.g., `MergeSort`.

In the rest of this section, we explain how to compare $\bar{c}_i^k$ and $\bar{c}_j^l$ for arbitrary $i \neq j$ using $\texttt{Compare}(\pi, \pi') := \texttt{Compare}_{0,0}(\pi, \pi')$ as an oracle; the construction can be generalized to the case $\epsilon, \delta > 0$ using the techniques developed in the previous subsection.

First, we need to generalize Lemma 2 to the situation when we move around more than two points. To this end, for any bidder i and any $1 \leq k_1 \leq k_2 \leq K$, we define

$$c_i^{[k_1,k_2]} = \frac{c_i^{k_1} g_i^{k_1} + \dots + c_i^{k_2} g_i^{k_2}}{g_i^{k_1} + \dots + g_i^{k_2}} = \frac{H_i^{k_2} - H_i^{k_1-1}}{G_i^{k_2} - G_i^{k_1-1}}.$$

This quantity is naturally related to bidder i's flattened virtual valuation.

Lemma 5. *Suppose that*

$$\bar{c}_i^{k_1-1} < \bar{c}_i^{k_1} = \cdots = \bar{c}_i^{k_2} < \bar{c}_i^{k_2+1}. \tag{6}$$

Then $c_i^{[k_1,k_2]} = \bar{c}_i^{k_1} = \cdots = \bar{c}_i^{k_2}$*. Also, if* $c_i^{[k_1,k_2]} > c_i^{[k_2+1,k_3]}$ *for some* $k_1, k_2, k_3 \in [1, K]$*, and* $\bar{c}_i^{k_1} = \cdots = \bar{c}_i^{k_2}$, $\bar{c}_i^{k_2+1} = \cdots = \bar{c}_i^{k_3}$ *then* $\bar{c}_i^k = \bar{c}_i^l$ *for all* $k, l \in [k_1, k_3]$.

Proof. Condition (6) means that the slope of $L_i(z)$ changes at $G_i^{k_1-1}$ and $G_i^{k_2}$, but remains constant between these two points. Hence, for $k \in [k_1, k_2]$, the value of $\bar{c}_i^k$ is the slope of the line that passes through $(G_i^{k_1-1}, H_i^{k_1-1})$, and $(G_i^{k_2}, H_i^{k_2})$ i.e., $c_i^{[k_1,k_2]}$. To prove the second statement, note that if $c_i^{[k_1,k_2]} > c_i^{[k_2+1,k_3]}$, then $(G_i^{k_2}, H_i^{k_2})$ lies above the line that connects $(G_i^{k_1-1}, H_i^{k_1-1})$ and $(G_i^{k_3}, H_i^{k_3})$. Therefore it cannot be a vertex of the lower envelope, i.e., the slope of $L_i(z)$ does not change at $(G_i^{k_2}, H_i^{k_2})$. $\square$

Lemma 6. *Suppose that for some* $i \neq j$ *and* $s, r, t \leq nK$ *we have* $\pi(w_i^k) = s, \ldots, \pi(w_i^{k+r-1}) = s + r - 1$, $\pi(w_j^l) = s + r, \ldots, \pi(w_j^{l+t}) = s + r + t$*. Let* π' *be an ordering obtained from* π *by swapping the groups* $(w_i^k, \ldots, w_i^{k+r-1})$ *and* $(w_j^l, \ldots, w_j^{l+t})$*. Let* Q *be the allocation rule associated with* π*. Then* $R(A_\pi) < R(A_{\pi'})$ *iff* $c_i^{[k,k+r-1]} > c_j^{[l,l+t]}$ *and* $q_j^l > 0$*.*

Proof. Let Q' be the allocation rule associated with π'. Recall that the expected revenue of an auction with allocation rule Q is

$$R_Q = \sum_{k_1=1}^{K} \cdots \sum_{k_n=1}^{K} \left[\sum_{i=1}^{n} Q_i(w_1^{k_1}, \ldots, w_n^{k_n}) c_i^{k_i} \right] g_1^{k_1} \ldots g_n^{k_n}.$$

By changing the ordering from π to π', we only changed the allocation rule at points $\mathbf{w}$ such that $Q_j(\mathbf{w}) = 1$ and $\mathbf{w} = (w_1^{k^1}, \ldots, w_i, \ldots, w_j, \ldots, w_n^{k^n})$, where $w_i \in \{w_i^k, \ldots, w_i^{k+r-1}\}$ and $w_j \in \{w_j^l, \ldots, w_j^{l+t}\}$. Let W^0 be the set of all such points. Let p_0 be the probability that all bidders $b_{i'}$, $i' \neq i, j$, have values $v_{i'}$ that satisfy $\pi(v_{i'}) < \pi(w_i^k)$; clearly, $p_0 > 0$ if and only if $q_j^l > 0$.

Fix $x \in [k, k+r-1], y \in [l, l+t]$ and consider the set of all points W^{xy} that satisfy $Q_j(\mathbf{w}) = 1$, $w_1 = w_i^x$, $w_2 = w_j^y$; clearly, $W^0 = \cup_{x \in [k,k+r-1], y \in [l,l+t]} W^{xy}$. Let $p_{xy} = \mathbf{Pr}[\mathbf{w} \in W^{xy}]$; it is easy to see that $p_{xy} = p_0 g_i^x g_j^y$. Under Q, the contribution of all $\mathbf{w} \in W^{xy}$ to the revenue is $c_j^y p_{xy}$; under Q', for all such $\mathbf{w}$ we have $Q_i(\mathbf{w}) = 1$ and hence the contribution of these points is $c_i^x p_{xy}$. The total change in revenue is therefore equal to

$$\sum_{x=k}^{k+r-1} \sum_{y=l}^{l+t} (c_i^x - c_j^y) p_{xy} = \sum_{x=k}^{k+r-1} \sum_{y=l}^{l+t} c_i^x p_{xy} - \sum_{x=k}^{k+r-1} \sum_{y=l}^{l+t} c_j^y p_{xy} =$$

$$= p_0 \sum_{x=k}^{k+r-1} \left(c_i^x g_i^x \sum_{y=l}^{l+t} g_j^y \right) - p_0 \sum_{y=l}^{l+t} \left(c_j^y g_j^y \sum_{x=k}^{k+r-1} g_i^x \right) = \frac{p_0 (c_i^{[k,k+r-1]} - c_j^{[l,l+t]})}{(g_j^l + \cdots + g_j^{l+t})(g_i^k + \cdots + g_i^{k+r-1})}.$$

Clearly, this expression is positive if and only if $c_i^{[k,k+r-1]} > c_j^{[l,l+t]}$ and $p_0 > 0$. $\square$

Lemma 5 implies that if $c_i^k > c_i^{k+1}$, then $\bar{c}_i^k = \bar{c}_i^{k+1}$, and therefore we can assume that in the optimal ordering these two elements appear together. Hence, we can combine them into a single element $w_i^{[k,k+1]}$ that has probability $g_i^k + g_i^{k+1}$ and virtual utility $c_i^{[k,k+1]}$. By Lemma 6, this element behaves identically to the pair (w_i^k, w_i^{k+1}) with respect to all comparisons. This reasoning also applies to combinations of three or more consecutive elements with identical flattened virtual utilities. Extending our notation, we set $\texttt{Compare}(w_i^{[k,k+r-1]}, w_j^{[l,l+t]}) = \texttt{Compare}(\pi, \pi')$, where π and π' are defined as in Lemma 6 with the additional restriction that $q_j^l \neq 0$ (i.e., $\pi(\perp) < \pi(w_i^k)$ and $\pi(w_{i'}^1) < w_i^k$ for all $i' \neq i$).

We start by describing a procedure $\texttt{Insert}(x, \mathcal{L})$ that given a bidder i's combined value $w_i^{[k_1,k_2]}$, the list $\mathcal{L}$ of bidder j's values $(w_j^1, \ldots, w_j^K)$, and access to $\texttt{Compare}(w_i^{[k_1,k_2]}, w_j^{[l_1,l_2]})$ can find a position t in $\mathcal{L}$ such that $c_i^{[k_1,k_2]} > \bar{c}_j^l$ for all $l \leq t$ and $c_i^{[k_1,k_2]} \leq \bar{c}_j^l$ for all $l > t$. To simplify notation, we set $x = w_i^{[k_1,k_2]}$ and omit the index j.

We assume that the list $\mathcal{L} = (w^1, \ldots, w^K, \$)$ has the structure of a double linked list, where $\$$ denotes the last element of this list, and $\texttt{Next}(u)$ and $\texttt{Prev}(u)$ are the standard linked list operations. Also, $\texttt{Merge}(w^{[l_1,l_2]}, w^{[l_2+1,l_3]})$ is a procedure that given two adjacent elements of the list, replaces them with an element $w^{[l_1,l_3]}$ and repairs the linked list structure.

```
Insert($x, \mathcal{L}$)

$u = w^1$; Z=1;
While ($u \neq \$$):
   $X = \texttt{Compare}(x, u)$;
   if $Z = 1$ and $X = 1$, set $u = \texttt{Next}(u)$;
   if $Z = 1$ and $X = 0$, set $u = \texttt{Next}(u)$, $Z = 0$;
   if $Z = 0$ and $X = 0$, set $u = \texttt{Next}(u)$;
   if $Z = 0$ and $X = 1$, set $u = \texttt{Merge}(\texttt{Prev}(u), u)$,
            $Z = \texttt{Compare}(x, \texttt{Prev}(\texttt{Prev}(u)))$;
Suppose that in the end, $\mathcal{L} = (w^{[a_1,b_1]}, \ldots, w^{[a_t,b_t]}, \$)$.
Let $w^{[a_s,b_s]}$ be the last element of $\mathcal{L}$
such that $\texttt{Compare}(x, w^{[a_s,b_s]}) = 1$;
Output $b_s$.
```

Fig. 3. The procedure $\texttt{Insert}(x, \mathcal{L})$

Lemma 7. *If* $\texttt{Insert}(c_i^{[k_1,k_2]}, \mathcal{L})$ *outputs* b_s*, we have* $\bar{c}_j^l < c_i^{[k_1,k_2]}$ *if and only if* $l \leq b_s$.

Proof. To simplify notation, assume $k_1 = k_2 = k$; the proof remains valid in the general case.

Clearly, the algorithm only merges $w^{[a,b]}$ and $w^{[a',b']}$ if $a' = b + 1$, so we can assume that when the algorithm terminates, we have $\mathcal{L} = (w^{[0,l_1]}, w^{[l_1+1,l_2]}, \ldots, w^{[l_{t-1}+1,K]}, \$)$ and $a_s = l_{s-1} + 1$,

$b_s = l_s$. Next, we prove by induction that we only merge points with identical flattened virtual utilities. To see this, note that the variable Z indicates whether the last element of the list seen so far was less than x (with respect to `Compare`). Therefore, the situation $Z = 0$, $X = 1$ arises when $\texttt{Prev}(u) = w^{[l_r+1,l_{r+1}]}$, $u = w^{[l_{r+1}+1,l_{r+2}]}$, and $c_i^k \le c_j^{[l_r+1,l_{r+1}]}$, $c_i^k > c_j^{[l_{r+1}+1,l_{r+2}]}$. By inductive assumption, it follows from Lemma 5 that all $\bar{c}_j^l$, $l = l_r + 1, \dots, l_{r+2}$, are equal. In other words, if $\mathcal{L}$ contains $w^{[l_{r-1}+1,l_r]}$, the slope of the lower envelope does not change between $G_j^{l_{r-1}}$ and $G_j^{l_r}$.

Also, it is easy to check by induction that when $\texttt{Insert}(c_i^k, \mathcal{L})$ terminates, any u that appears before $w^{[a_s,b_s]}$ in the list $\mathcal{L}$ satisfies $\texttt{Compare}(x, u) = 1$, and by construction, any u that appears after $w^{[a_s,b_s]}$ satisfies $\texttt{Compare}(x, u) = 0$.

Now, let $L_1 = L_1(z)$ be the lower envelope of the set $\{(G_j^0, H_j^0), (G_j^{l_1}, H_j^{l_1}) \dots, (G_j^{l_s}, H_j^{l_s}\}$, let $L_2 = L_2(z)$ be the lower envelope of the set $\{(G_j^{l_s}, H_j^{l_s}), (G_j^{l_s+1}, H_j^{l_s+1}), \dots, (G_j^K, H_j^K)\}$, and let $L = L(z)$ be the lower envelope of $\{(G_j^l, H_j^l) \mid l = 0, \dots, K\}$. Clearly, $L(z) \le L_1(z)$ for any $z \le G_j^{l_s}$ and $L(z) \le L_2(z)$ for any $z \ge G_j^{l_s}$; we would like to show that, in fact, $L = L_1 \cup L_2$. To see that, note that the slope of any segment of L_1 is less than c_i^k, since it is obtained by taking a weighted average of some $c_j^{[l_{r-1}+1,l_r]}$ for $r \le s$. and $c_j^{[l_{r-1}+1,l_r]} < c_i^k$ for all such r. Similarly, the slope of any segment of L_2 is at least c_i^k. Therefore, $L_1 \cup L_2$ is a convex curve. Moreover, any (G_j^l, H_j^l), $l = 0, \dots, K$, lies on or above $L_1 \cup L_2$, because otherwise the slope of L would change at some G_l, $l \ne l_1, \dots, l_t$. Hence, $L = L_1 \cup L_2$. Consequently, for any $z \le G_j^{l_s}$ the slope of L at z is less than c_i^k, and for any $z > G_j^{l_s}$ the slope of L at z is at least c_i^k. □

If we knew k_1 and k_2 such that $\bar{c}_i^k = c_i^{[k_1,k_2]}$, we could use $\texttt{Insert}(c_i^{[k_1,k_2]}, \mathcal{L})$ to compare $\bar{c}_i^k$ and $\bar{c}_j^l$. Unfortunately, these k_1 and k_2 might be impossible to determine. Nevertheless, it turns out that we can use $\texttt{Insert}(x, \mathcal{L})$ as a subroutine to determine the relative order of the elements of $\mathcal{L}' = (w_i^1, \dots, w_i^K, \$)$ and $\mathcal{L} = (w_j^1, \dots, w_j^K, \$)$.

The new algorithm attempts to insert each of the elements of $\mathcal{L}'$ into $\mathcal{L}$ using `Insert`. If `Insert` suggests inserting w_i^k and w_i^{k+1} after $w_j^{l_1}$ and $w_j^{l_2}$ respectively, and $l_2 < l_1$, this means that $w_i^k > w_j^{l_1} \ge w_i^{k+1}$, and therefore w_i^k and w_i^{k+1} should be merged; the algorithm merges them and uses `Insert` to find the appropriate position for $w_i^{[k,k+1]}$. The algorithm uses a stack S to keep track of the elements of $\mathcal{L}'$ that have been inserted prior to the current element; for each element, we record its position with respect to $\mathcal{L}$, so that in the end we know the relative order of the elements of $\mathcal{L}$ and $\mathcal{L}'$.

Proposition 5. *Suppose that when* $\texttt{Combine}(\mathcal{L}, \mathcal{L}')$ *terminates, the contents of the stack is*

$$(w_i^{[k_1,k_1']}, t_1), (w_i^{[k_2,k_2']}, t_2), \dots, (w_i^{[k_s,k_s']}, t_s).$$

Then $k_1 = 1$, $k_s' = K$, *and* $k_r \le k_r'$, $k_{r+1} = k_r' + 1$ *for all* $1 \le r < s$. *Finally, if* $k \in [k_r, k_r']$, *then* $\bar{c}_j^{t_r} < \bar{c}_i^k \le \bar{c}_j^{t_r+1}$.

```
Combine($\mathcal{L}, \mathcal{L}'$)

$v = w_i^1$;  $t =$ Insert($v, \mathcal{L}$);  Push($S, (v, t)$);
While ($v \neq \$$):
  $(u, t) =$ Pop($S$);  $t' =$ Insert($v, \mathcal{L}$);
  if $t' \geq t$
    Push($S, (u, t)$);  Push($S, (v, t')$);  $v =$ Next($v$);
  if $t' < t$
    $v =$ Merge($u, v$);
```

Fig. 4. The procedure `Combine`$(\mathcal{L}, \mathcal{L}')$

Proof. It is easy to verify by induction that $k_1 = 1$, $k'_s = K$, and $k_r \leq k'_r$, $k_{r+1} = k'_r + 1$, and $t_1 \leq \cdots \leq t_s$. Similarly to the proof of Claim 7, we can show that we only merge $w_i^{[l_1,l_2]}$ and $w_i^{[l_3,l_4]}$ if $l_3 = l_2 + 1$, and we only merge points that have identical flattened virtual utilities. Now, consider all elements on the stack that are of the form $(w_i^{[k_r,k'_r]}, t)$ for a fixed value of t; suppose that these elements are $w_i^{[x,x']}, \ldots, w_i^{[y,y']}$. By construction, for any such $w_i^{[k_r,k'_r]}$ we have $t = \texttt{Insert}(w_i^{[k_r,k'_r]}, \mathcal{L})$ and therefore $\bar{c}_j^t < c_i^{[k_r,k'_r]} \leq \bar{c}_j^{t+1}$. Let L' be the lower envelope of $\{(G_i^k, H_i^k) \mid k = 0, \ldots, K\}$. Repeating the argument in the proof of Proposition 7, we can conclude that the slopes of all segments of L' between G_i^{x-1} and $G_i^{y'}$ were obtained by taking a weighted average of $c_i^{[x,x']}, \ldots, c_i^{[y,y']}$ and therefore $\bar{c}_j^t < \frac{dL}{dz}(\xi) \leq \bar{c}_j^{t+1}$ for any $\xi \in (G_i^{x-1}, G_i^{y'})$. In particular, this is true for $\xi \in (G_i^{k-1}, G_i^k)$, which means that $\bar{c}_j^t < \bar{c}_i^k \leq \bar{c}_j^{t+1}$. □

Running times It is easy to see that both `Insert`$(x, \mathcal{L})$ and (indirectly) `Combine`$(\mathcal{L}, \mathcal{L}')$ make a polynomial number of calls to `Compare`(u, v); in this section we derive somewhat more precise bounds.

Proposition 6. *The procedure* `Insert`$(x, \mathcal{L})$ *makes at most* $2K$ *calls to* `Compare`(x, u). *The procedure* `Combine`$(\mathcal{L}, \mathcal{L}')$ *makes at most* $2K$ *calls to* `Insert`$(v, \mathcal{L})$.

Proof. Whenever `Compare`(x, u) is called, the algorithm also calls either `Next`(x) to obtain an element of $\mathcal{L}$ it has not seen before or `Merge(Prev`$(u), u)$ to merge two groups of elements of $\mathcal{L}$. Clearly, each of these actions can be executed at most K times. Similarly, whenever `Insert`$(v, \mathcal{L})$ is called, the algorithm also calls either `Next`(v) to obtain an element of $\mathcal{L}'$ it has not seen before or `Merge`(u, v) to merge two groups of elements of $\mathcal{L}'$, and each of these actions can be executed at most K times. □

Corollary 1. *For any* $i \neq j$, *the relative ordering of all* $\bar{c}_i^k$ *and* $\bar{c}_j^l$, $k, l = 1, \ldots, K$ *can be determined by* $4K^2$ *comparisons. Hence, using at most* $2(nK)^2$ *comparisons, we can construct an oracle that compares any two flattened virtual utilities in unit time. After this oracle is constructed, we can find an optimal ordering using any sorting algorithm, e.g.,* `MergeSort`*; the running time of* `MergeSort` *will be the same as in the regular case, i.e.,* $O(n \log nK)$.

Remark 5. The running time of our algorithm for finding an optimal ordering is dominated by the time it takes to construct the comparison oracle. One can reduce the running time somewhat by combining the two components of our algorithm: even though in this paper we opted for a more modular presentation for the ease of exposition, a practical algorithm would intertwine merging the bidders' arrays and determining the relative ordering of $\bar{c}_i^k$ and $\bar{c}_j^l$. However, the quadratic dependence on K appears to be inherent to our approach, and one will need completely new ideas to eliminate it.

6 Conclusions

We have shown how to construct an optimal auction for finite support distributions. While such distributions provide a better model for many real life scenarios than continuous ones, a rigorous analysis of this case was absent from the literature; this paper fills this gap. Also, we believe that the concept of order-based auctions introduced in this paper may have applications beyond those considered here. The second main contribution of this paper is in demonstrating that the optimal auction can be learned under fairly harsh conditions. Moreover, if the distributions in question are regular, this can be done by a simple greedy algorithm, which can be viewed as an argument for practical applicability of our construction.

In practice, the mechanism designer cannot expect that the output of the comparison oracle will be always correct, and we show that our learning algorithm is robust to errors in the oracle's reports. Further relaxing this model (e.g., to the case when the information about distribution supports is erroneous or imprecise) is an interesting challenge. Another important question is learning the optimal auction in the continuous case; we hope that techniques and intuition developed in this paper will prove useful here. Also, designing and learning the optimal finite support auction when the bidders' valuations are not independent is an open problem. While Cremer and McLean [9] show that one can extract full surplus if the dependencies are strong enough, they provide no answer for the general case and their mechanism is not *ex post* individually rational. On the other hand, it is not clear if the lower bounds proved in [20, 21] are optimal or whether one can get positive results in this framework for special classes of joint distributions. We propose investigating the problem of finding the best order-based auction for this scenario: while this problem is clearly in NP (assuming the expected revenue oracle), it would be interesting to see a hardness result or a polynomial-time algorithm (note that the existence of the latter is not precluded by the results in [21], since we are considering a restricted model).

It is not clear if any of our results may be applicable to the more interesting problem of finding an optimal multi-unit auction. However, our work suggests that when the bidders' valuations are discrete, one might try to characterize a class of combinatorial structures (e.g., a generalization of order-based auctions) containing the optimal solution and use the properties of this class to limit the search space. This topic is a subject of ongoing research.

Acknowledgments The author would like to thank Amit Sahai, Ken Steiglitz, Sergei Izmalkov, Sergei Anisov, Paul Goldberg, and anonymous referees for useful discussions and comments on an earlier version of this paper.

References

1. G. Aggarwal and J. Hartline, Knapsack auctions. In *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1083–1092, 2006
2. M.-F Balcan, A. Blum, J. Hartline, and Y. Mansour, Mechanism design via machine learning. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, pages 605–614, 2005
3. A. Blum and J. Hartline, Near-optimal online auctions. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1156–1163, 2005
4. Z. Bar-Yossef, K. Hildrum, and F. Wu, Incentive-compatible online auctions for digital goods. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 964–970, 2002
5. D. Bergemann and M. Pesendorfer, Information structures in optimal auctions. *Cowles Foundation Discussion Papers 1323, Cowles Foundation, Yale University*, 2001
6. A. Blum, V. Kumar, A. Rudra, and F. Wu, Online learning in online auctions. In *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 202–204, 2003
7. J. Bulow and J. Roberts, The simple economics of optimal auctions. *The Journal of Political Economy*, 97:1060–90, 1989
8. H. Cheng, Optimal auction design with discrete bidding. Working paper, May 2004
9. J. Cremer and R. McLean, Full extraction of the surplus in bayesian and dominant strategy auctions. *Econometrica*, 56:1247–1257, 1988
10. E. Elkind, Computational Issues in Optimal Auction Design. PhD thesis, Princeton University, 2005
11. E. Elkind, Optimal auctions with finite support. In *DIMACS Workshop on Computational Issues in Auction Design*, 2004
12. A. Fiat, A. Goldberg, J. Hartline, and A. Karlin, Competitive generalized auctions. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computation*, pages 72–81, 2002
13. A. Goldberg, J. Hartline, and A. Wright, Competitive auctions and digital goods. In *Proceedings of the 12th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 735–744, 2001
14. M. Hajiaghayi, R. Kleinberg, and D. Parkes, Adaptive limited-supply online auctions. In *Proceedings of the 5th ACM Conference on Electronic Commerce*, pages 71–80, 2004
15. R. Kleinberg, F. Leighton, The value of knowing a demand curve: bounds on regret for online posted-price auctions. In *44th Annual IEEE Symposium on Foundations of Computer Science*, pages 594–605, 2003
16. V. Krishna, Auction Theory. Academic Press, 2002
17. H. Lipmaa, N. Asokan, and V. Niemi, Secure Vickrey auctions without threshold trust. In *Proceedings of the 6th Annual Conference on Financial Cryptography*, pages 87–101, 2002
18. R. Myerson, Optimal auction design. *Mathematics of Operations Research*, 6:58–73, 1981
19. M. Naor, B. Pinkas, and R. Sumner, Privacy preserving auctions and mechanism design. In *ACM Conference on Electronic Commerce* pages 129–139, 1999
20. A. Ronen, On approximating optimal auctions. In *ACM Conference on Electronic Commerce*, pages 11–17, 2001
21. A. Ronen and A. Saberi, On the hardness of optimal auctions. In *Proceedings of the 43rd Symposium on Foundations of Computer Science*, pages 396–405, 2002
22. W. Vickrey, Counterspeculation, auctions, and competitive sealed tenders. *Journal of Finance*, 16:8–37, 1961

A Bidders with finite support value distributions

We describe several settings, in which it is natural to assume that the bidders draw their values from finite support distributions.

Example 1. Suppose that we are selling a plane ticket from New York to Boston, and we have two potential buyers Alice and Bob. Besides buying a ticket from us, both Alice and Bob can take a bus, which costs $\$10$, drive a car, which costs $\$40$, take a train, which costs $\$50$, or buy a plane ticket from someone else for $\$100$. Also, we know that Alice is a student, so she will pick the cheapest available option (i.e., the bus), unless she has to be in Boston early in the morning, in which case she needs to buy a plane ticket (either from us or from another company), and with probability 0.9, Alice has to be in Boston early. On the other hand, Bob is a software engineer, who finds buses uncomfortable, but does not want to pay more than $\$50$, and with probability 0.5, Bob does not own a car. In this situation, we can conclude that Alice's valuation for the ticket is $\$100$ with probability 0.9 and $\$10$ with probability 0.1, while Bob's valuation for the ticket is $\$40$ with probability 0.5 and $\$50$ with probability 0.5.

This example generalizes naturally to the case when instead of buying the object being auctioned, the buyers can purchase one of the similar products available in the market for a fixed price; however, depending on their circumstances (which are not known to the auctioneer), each buyer's selection may be restricted to a subset of these products.

Example 2. Consider an auction that sells bulk goods to retailers that put them into individual packages of a predetermined size and resell them. Each bidder's value for the lot is determined by how many individual packages he expects to sell, and this information is private to the bidder, while the market price of an individual package and its size are common knowledge.

Example 3. Suppose that we are selling an object (e.g., a car) with a number of add-ons; the buyer either values each feature at the (known) market rate, or is indifferent about it, and this information is private to the buyer.