Towards Simple Polynomial-Time Optimal Auctions

Takehiro Oyakawa and Amy Greenwald

1Department of Computer Science, Brown University, Providence, RI
{oyakawa, amy}@cs.brown.edu

July 13, 2016

Abstract

Border’s theorem gives us a way of solving mechanism design problems in polynomial time. This approach requires the use of linear programming machinery, the mechanics of which are significantly more complicated than the pointwise maximization approach that Myerson’s analysis of single-parameter mechanism design problems yields. However, Myerson’s approach, applied naively, yields an exponential-time algorithm. Motivated by the simplicity of Myerson’s analysis, we give a simple method for solving the usual one-good, many-bidders optimal auction problem in polynomial time. This method can be applied to both surplus and revenue maximization, and does not rely on linear programming.

1 Introduction

Optimal auctions for single-parameter settings were studied by Myerson [1981], who showed that a Vickrey auction with reserve prices can be used to maximize seller revenue. Myerson’s analysis revealed that total expected revenue can be determined by transforming bidder valuations into virtual valuations, and then assigning positive allocation probabilities to the bidders with the highest positive virtual valuations. This pointwise procedure is simple and elegant. However, examining every possible bidder profile may take an exponential amount of time. In this paper, we modify Myerson’s pointwise approach to compute total expected revenue in polynomial time in the usual one-good, many-bidders setting. In particular, our result does not rely on linear programming or oracles, which are widely used today to address the issue of computational complexity in mechanism design, such as in Cai et al. [2013].

Related Work  Vickrey [1961] showed that auctions in which the highest bidder wins and pays the second-highest bid incentivize bidders to bid truthfully. Myerson [1981] showed that in the single-parameter setting, with the usual quasi-linear utility function involving linear payments, total expected revenue is maximized by a Vickrey auction with reserve prices. In principle, revenue maximization for some mechanism design problems, such as the single-parameter setting, can be solved using Border’s characterization of interim feasible outcomes (Border [1991]) and an ellipsoid type algorithm with a separation oracle. However, in practice, such mechanisms tend to be expensive and do not have the simplicity and elegance of the pointwise maximization procedure Myerson’s analysis yields. Here, we seek fast, intuitive and simple to understand allocation methods that do not rely on linear programming machinery. We make use of perturbations for handling ties, as in Hartline [2013], which applies perturbations to revenue curves.
2 Model and Background

An auctioneer would like to sell a good to one of \( n \) bidders. Each bidder \( i \in N = \{1, \ldots, n\} \) has a private valuation \( 0 \leq v_i \in T_i \) that is independently drawn from some distribution \( F_i \). Let \( T = T_1 \times \cdots \times T_n \) be the set of all possible valuation vectors (profiles), and let \( F = F_1 \times \cdots \times F_n \) be the distribution over valuation vectors \( \mathbf{v} = (v_1, \ldots, v_n) \in T \). Let \( \mathbf{b} = (b_1, \ldots, b_n) \in \mathbb{R}^n \) be a vector of bids, where the \( i \)th entry \( b_i \) is bidder \( i \)'s bid. For \( \mathbf{y} \in \{ \mathbf{v} \}, \) we use the notation \( \mathbf{y} = (y_i, \mathbf{y}_{-i}) \), where \( \mathbf{y}_{-i} = (y_1, \ldots, y_{i-1}, y_{i+1}, \ldots, y_n) \). Similarly, we let \( T_{-i} = \prod_{j \neq i} T_j \) and \( F_{-i} = \prod_{j \neq i} F_j \).

Given a vector of reports \( \mathbf{b} \), a mechanism determines an allocation rule \( \mathbf{x}(\mathbf{b}) \in [0, 1]^n \), together with a payment rule \( \mathbf{p}(\mathbf{b}) \in \mathbb{R}^n_{\geq 0} \), where bidder \( i \)'s payment is \( p_i(b_i, \mathbf{b}_{-i}) \). We define bidder \( i \)'s quasi-linear utility function as

\[
u_i(b_i, \mathbf{b}_{-i}) = v_i x_i(b_i, \mathbf{b}_{-i}) - p_i(b_i, \mathbf{b}_{-i}).\quad (1)
\]

Next, we formalize the usual constraints imposed on optimal auction design. Because we restrict our attention to incentive compatible auctions, where it is optimal to bid truthfully, hereafter, we write \( x_i(v_i, \mathbf{v}_{-i}) \) instead of \( x_i(b_i, \mathbf{b}_{-i}) \), and \( p_i(v_i, \mathbf{v}_{-i}) \) instead of \( p_i(b_i, \mathbf{b}_{-i}) \).

We introduce interim allocation and interim payment variables, respectively:
\( \hat{x}_i(v_i, \mathbf{v}_{-i}) = \mathbb{E}_{\mathbf{v}_{-i}}[x_i(v_i, \mathbf{v}_{-i})] \) and \( \hat{p}_i(v_i, \mathbf{v}_{-i}) = \mathbb{E}_{\mathbf{v}_{-i}}[p_i(v_i, \mathbf{v}_{-i})] \). These variables comprise the interim allocation and payment rules, \( \hat{x}(\mathbf{v}) \in [0, 1]^n \) and \( \hat{\mathbf{p}}(\mathbf{v}) \in \mathbb{R}^n_{\geq 0} \).

We call a mechanism (Bayesian) incentive compatible (IC) if utility is maximized by truthful reports in expectation: \( \forall i \in N \) and \( \forall v_i, w_i \in T_i \), \( v_i \hat{x}_i(v_i) - \hat{p}_i(v_i) \geq v_i \hat{x}_i(w_i) - \hat{p}_i(w_i) \). A mechanism is Individually rational (IR) if it insists on non-negative utilities in expectation: \( \forall i \in N \) and \( \forall v_i \in T_i \), \( v_i \hat{x}_i(v_i) - \hat{p}_i(v_i) \geq 0 \). We say a mechanism is ex-post feasible (XP) if it never overallocates: \( \forall \mathbf{v} \in T \), \( \sum_{i=1}^n x_i(v_i, \mathbf{v}_{-i}) \leq 1 \).

Finally, we require that \( 0 \leq x_i(v_i, \mathbf{v}_{-i}), \hat{x}_i(v_i) \leq 1 \), \( \forall i \in N \), \( \forall v_i \in T_i \), and \( \forall \mathbf{v}_{-i} \in T_{-i} \).

2.1 Myerson’s Payment Analysis, and Pointwise Maximization

Myerson’s payment theorem, which takes as a starting point the bidders’ utility functions (i.e., Equation (1)), was expressed assuming that bidders drew valuations from continuous distributions. Here, we apply his analysis to discrete distributions.

Specifically, we assume the distribution of values is drawn from the discrete type space \( T_i = \{z_{i,k} : 1 \leq k \leq M_i\} \), of cardinality \( |T_i| = M_i \), where \( z_{i,j} < z_{i,k} \) for \( j < k \), and we let \( z_{i,T_i+1} = z_{i,T_i} \). We also assume the probability of type \( z_{i,k} \in T_i \) is given by cumulative distribution function \( F_i(z_{i,k}) \) and corresponding probability mass function \( f_i(z_{i,k}) \). In addition, \( f_{-i}(\mathbf{v}_{-i}) \) is the probability mass function of \( \mathbf{v}_{-i} \in T_{-i} \).

**Theorem 2.1** [Myerson 1981]. Assume bidders’ utilities take the form of Equation (1). A mechanism is IC and IR iff for all \( i \in N \): the allocation rule \( \hat{x} \) is monotone, i.e., \( \forall v_i \geq w_i \in T_i \), \( \hat{x}_i(v_i) \geq \hat{x}_i(w_i) \); and the payment rule \( \hat{\mathbf{p}} \) is given by: \( \forall z_{i,\ell} \in T_i \),

\[
\hat{p}_i(z_{i,\ell}) = z_{i,\ell} \hat{x}_i(z_{i,\ell}) - \sum_{j=1}^{\ell-1} (z_{i,j+1} - z_{i,j}) \hat{x}_i(z_{i,j}). \quad (2)
\]

Note that \( \hat{p}_i(z_{i,1}) = z_{i,1} \hat{x}_i(z_{i,1}) \).

Using Myerson’s analysis, we can see that if maximizing total expected surplus, \( \mathbb{E}_{\mathbf{v} \sim F}[\mathbf{v} \cdot \mathbf{x}(\mathbf{v})] \), were the objective, then we can do so pointwise, solving for an optimal allocation for each \( \mathbf{v} \) in turn,
subject only to ex-post feasibility. This allocation rule can be supported with Myerson’s payment rule, thereby ensuring IC and IR. Algorithm 1 describes this pointwise approach in detail, where, when maximizing total expected surplus is the objective, each \( L_i \) is the identity function.

Algorithm 1 Pointwise Maximizer

```plaintext
1: for all \( v \in T \) do
2:     for all \( i \in N \) do
3:         \( x_i(v_i, v_{-i}) \leftarrow 0 \)
4:     end for
5: if any of the \( L_i(v_i) \)'s are positive then
6:     \( w \leftarrow \arg \max_i \{ L_i(v_i) \} \)
7:     for all \( i^* \in w \) do
8:         \( x_i^*(v_i^*, v_{-i^*}) \leftarrow 1/|w| \)
9:     end for
10: end if
11: end for
12: for all \( i \in N \) do
13:     for \( \ell = 1 \) to \( |T_i| \) do
14:         \( \hat{x}_i(z_{i,\ell}) \leftarrow \mathbb{E}_{v_{-i}}[x_i(z_{i,\ell}, v_{-i})] \) \quad \( \triangleright \text{Compute interim allocations} \)
15:         \( \hat{p}_i(z_{i,\ell}) \leftarrow z_{i,\ell}\hat{x}_i(z_{i,\ell}) - \sum_{j=1}^{\ell-1}(z_{i,j+1} - z_{i,j})\hat{x}_i(z_{i,j}) \) \quad \( \triangleright \text{Compute interim payments} \)
16:     end for
17: end for
18: \( S \leftarrow \sum_{i=1}^{n} \mathbb{E}_{z_{i,\ell}}[\hat{x}_i(z_{i,\ell})] \) \quad \( \triangleright \text{Total expected surplus} \)
19: \( R \leftarrow \sum_{i=1}^{n} \mathbb{E}_{z_{i,\ell}}[\hat{p}_i(z_{i,\ell})] \) \quad \( \triangleright \text{Total expected revenue} \)
20: return \( S, R, \hat{x}, \hat{p} \)
```

Algorithm 1 is exponential in runtime. Let \( \mathfrak{R} = \max\{|T_i| \mid i \in N\} \). There are \( O(\mathfrak{R}^n) \) valuation vectors in \( T \). For each valuation vector, determining allocations is \( O(n) \), so determining all allocations is \( O(n\mathfrak{R}^n) \). Each \( \hat{x}_i(z_{i,\ell}) \) is computed in \( O(\mathfrak{R}^{n-1}) \), so determining all interim allocations takes \( O(n\mathfrak{R}^n) \). Each \( \hat{p}_i(z_{i,\ell}) \) is computed in \( O(\mathfrak{R}) \), so determining all interim payments takes \( O(n\mathfrak{R}^2) \). Computing \( S \) is done in \( O(n\mathfrak{R}) \). Computing \( R \) is done in \( O(n\mathfrak{R}) \). Therefore, the complexity of Algorithm 1 is \( O(n\mathfrak{R}^n) \).

### 2.2 Applying Pointwise Maximization to Revenue

In his seminal work on optimal auction design, Myerson proved that expected revenue can be expressed as something called expected **virtual surplus**, which he defined in terms of virtual valuations. Using our notation, we define **virtual valuations** for discrete distributions as follows:

\[
\psi_i(z_{i,k}, z_{i,k+1}) = z_{i,k} - (z_{i,k+1} - z_{i,k}) \left( 1 - \frac{F_i(z_{i,k})}{f_i(z_{i,k})} \right). \tag{3}
\]

We use the shorthand \( \psi_i(v_i) \equiv \psi_i(z_{i,k}, z_{i,k+1}) \), where \( v_i = z_{i,k} \) for some \( 1 \leq k \leq |T_i| \). We assume our problem is regular, as in Myerson [1981], so that \( \psi_i(z_{i,k+1}) > \psi_i(z_{i,k}) \) whenever \( z_{i,k+1} > z_{i,k} \).

Using the discrete version of Myerson’s payment formula (Theorem 2.1), and following a similar analysis to that of Myerson [1981], we arrive at the following theorem:
Similarly, we know that bidder $i$’s revenue is the objective, in turn. This procedure is again given by Algorithm 1, where, when maximizing total expected revenue is the objective, $L_i$ is the virtual valuation function of bidder $i$.

In maximizing Equation (5), bidders with negative virtual valuations will never be allocated. Since bidders must place a bid that maps to a non-negative virtual valuation in order to be allocated, it means that each bidder $i$ has a reserve price, $\psi_i^{-1}(0)$, which is the smallest bid she may place in order to possibly be allocated, and the smallest possible payment she must make when allocated.

3 On Ties

In this section, we describe how the expected revenue from a bidder can be described by a partition of calculations. This partitioning does not change the fact that calculating the expected revenue takes an exponential amount of time. However, in the very special case of no ties, partitioning in this way leads to a polynomial-time calculation of virtual surplus (revenue).

Bidder $i$ can only be allocated when the following criteria are met:

- $i$ has placed a bid larger than $i$’s reserve price: $\psi_i(v_i) \geq \psi_i^{-1}(0)$, and
- $i$’s virtual valuation is at least as large as everyone else’s: $\psi_i(v_i) = \max \{ \psi_j(v_j) : j \in N \}$.

Let $w(v)$ be the set of bidders that can potentially be allocated:

$$w(v) = \{ i : \psi_i(v_i) \geq \psi_i^{-1}(0), \psi_i(v_i) = \max \{ \psi_j(v_j) : \forall j \in N \}, \forall i \in N \}. \quad (6)$$

The probability that a bidder is allocated depends on whether there are any ties or not:

$$x_i(v_i, v_{-i}) = \begin{cases} 1/|w(v)| & \text{if } i \in w(v) \\ 0 & \text{otherwise}. \end{cases} \quad (7)$$

We know that $i$ will not be allocated when she bids less than her reserve price; only valuations at least as large as the reserve price may be allocated. Let $\tau_{i}^{\geq 0}$ be the set of bidder $i$’s valuations that at least match her reserve price:

$$\tau_{i}^{\geq 0} = \{ z_{i,k} : \psi_i(z_{i,k}) \geq 0, \forall z_{i,k} \in T_i \}. \quad (8)$$

Similarly, we know that bidder $i$ is guaranteed not to be allocated when there exists any other bidder with virtual valuation larger than $i$’s. Specifically, in order for $i$ to win with valuation $z_{i,k}$, the other bidders virtual valuations must be such at most $\psi_i(z_{i,k})$. Let $\tau_{\leq \psi_i(z_{i,k})}$ be the set of profiles other bidders may have where $i$’s virtual valuation, $\psi_i(z_{i,k})$, is at least as large as any other:

$$\tau_{\leq \psi_i(z_{i,k})} = \{ v_{-i} : \max v_{-i}(v_{-i}) \leq \psi_i(z_{i,k}), \forall v_{-i} \in T_{-i} \}. \quad (9)$$
We can now express the interim allocation \( \hat{x}_i(z_{i,k}) \) as follows:

\[
\hat{x}_i(z_{i,k}) = \begin{cases} 
0 & \text{if } \psi_i(z_{i,k}) < 0 \\
\sum_{v_{-i} \in \tau_{-i}} f_i(v_{-i}) & \text{if } \psi_i(z_{i,k}) \geq 0 \\
\sum_{v_{-i} \in \tau_{-i}} \psi_i(z_{i,k}) \sum_{v_{-i} \in \tau_{-i}} x_i(v_i, v_{-i}) f_i(v_{-i}) & \text{otherwise.}
\end{cases}
\] (10)

Furthermore, we can partition the calculation of total expected revenue from bidder \( i \) as follows:

\[
\sum_{k=1}^{\left| T_i \right|} \psi_i(z_{i,k}) f_i(z_{i,k}) \hat{x}_i(z_{i,k}) = \sum_{z_{i,k} \in \tau_{i}^{\geq 0}} \psi_i(z_{i,k}) f_i(z_{i,k}) \sum_{v_{-i} \in \tau_{-i}} x_i(v_i, v_{-i}) f_i(v_{-i}).
\] (11)

This total expected revenue formula involves only a subset of \( T \). However, as indicated by Equation (11), this subset is still exponential in size. Next, we observe that when there are no ties, computing total expected revenue can be done in polynomial time.

Assuming no ties, when \( i \) wins, \( x_i(v_i, v_{-i}) = 1 \). Let \( \tau_{-i}^{<\psi_i(z_{i,k})} \) be the set of profiles other bidders may have where \( i \)'s virtual valuation, \( \psi_i(z_{i,k}) \), is strictly larger than any other:

\[
\tau_{-i}^{<\psi_i(z_{i,k})} = \{ v_{-i} : \max_{\psi_{-i}(v_{-i})} < \psi_{i,k}(z_{i,k}), \forall v_{-i} \in T_{-i} \}.
\] (12)

In this strict case, Equation (10) becomes

\[
\hat{x}_i(z_{i,k}) = \sum_{v_{-i} \in \tau_{-i}^{<\psi_i(z_{i,k})}} f_i(v_{-i}),
\] (13)

and Equation (11) becomes

\[
\sum_{k=1}^{\left| T_i \right|} \psi_i(z_{i,k}) f_i(z_{i,k}) \hat{x}_i(z_{i,k}) = \sum_{z_{i,k} \in \tau_{i}^{\geq 0}} \psi_i(z_{i,k}) f_i(z_{i,k}) \sum_{v_{-i} \in \tau_{-i}^{<\psi_i(z_{i,k})}} f_i(v_{-i}).
\] (14)

Finally, since we are working with independent and identically distributed random variables, the interim allocation calculation is no longer exponential:

\[
\sum_{v_{-i} \in \tau_{-i}^{<\psi_i(z_{i,k})}} f_i(v_{-i}) = \prod_{j \neq i} \Pr(\psi_i(z_{i,k}) > \psi_j(z_{j,k})) = \prod_{j \neq i} \sum_{z_{j,k} : \psi_i(z_{i,k}) > \psi_j(z_{j,k})} f_j(z_{j,k}).
\] (15)

In other words, assuming no ties, interim allocations can be computed in polynomial time. Consequently, we can also compute total expected revenue in polynomial time. We describe this procedure in Algorithm 2, where \( L_i \) is \( i \)'s virtual valuation function.

Algorithm 2 is polynomial in runtime. For each bidder, setting \( \hat{x}_i(z_{i,k}) \) to 0 for all \( z_{i,k} \in T_{i} \setminus \tau_{i,L_i}^{\geq 0} \) is \( O(R) \). Computing \( \hat{x}_i(z_{i,k}) \) is done in \( O(nR) \). Doing so for all \( z_{i,k} \in \tau_{i,L_i}^{\geq 0} \) is \( O(nR^2) \). Computing every \( \hat{p}_i(z_{i,k}) \) is \( O(R^2) \). Therefore, determining interim allocations takes \( O(n^2R^2) \), and determining interim payments takes \( O(n^2R^2) \). Computing \( S \) is done in \( O(nR) \). Computing \( R \) is done in \( O(nR) \). Therefore, the complexity of Algorithm 2 is \( O(n^2R^2) \).

**Remark 3.1.** When there are no ties, Algorithm 1 and Algorithm 2 both give the same output.

**Remark 3.2.** When there are no ties, and \( L_i \) is the identity function, Algorithm 2 provides a way of maximizing total expected surplus in polynomial time.
4 Perturbations to Virtual Valuations

At this point, we have observed that the total expected revenue calculation is polynomial-time when there are no ties. It remains, then, to show how ties can also be handled in polynomial-time. This is the subject of the rest of the paper.

Similar to [Hartline 2013], our strategy for handling ties will be to transform any set of virtual valuations into one that is guaranteed to have no ties, without changing the allocation function too much. In this section, we show that we can modify virtual valuations slightly so that (i) any virtual valuation larger (smaller) than any other virtual valuation continues to remain larger (smaller), and (ii) any virtual valuation above (below) the reserve price remains above (below) the reserve price.

Lemma 4.1. There exists an $\epsilon_A \in \mathbb{R}_{>0}$ such that for all $r_{i,k} \in [0, 1]$, if $\psi_i(z_{i,k}) > \psi_j(z_{j,\ell})$, then $\psi_i(z_{i,k}) + r_{i,k} \epsilon_A > \psi_j(z_{j,\ell}) + r_{j,\ell} \epsilon_A$, for all $i,j \in N$, $1 \leq k \leq |T_i|$, $1 \leq \ell \leq |T_j|$.

Proof. Consider any set of virtual valuations $\psi_1 > \psi_2 > \psi_3$. For $c,d \in \{1,2,3\}$, let $\delta_{c,d} = (\psi_c - \psi_d)$ for $c < d$. Let $r_c$ be any number in $[0,1]$. The difference between any $r_c$ and $r_d$ cannot exceed 1, so $\delta_{c,d} = (\psi_c - \psi_d) \geq (r_d - r_c) \delta_{c,d}$.

Let $0 < \epsilon_A < \epsilon_A^U = \min\{\delta_{c,d} | c,d \in \{1,2,3\}, c < d\}$. Then we have $(\psi_c - \psi_d) > (r_d - r_c) \epsilon_A$, so $\psi_c + r_c \epsilon_A > \psi_d + r_d \epsilon_A$. See Figure 1.

Figure 1: Graphical depiction of the proof of Lemma 4.1. Any change of the virtual valuations by $\epsilon_A$ preserves the ordering of virtual valuations.

A procedure for finding an $\epsilon_A$ is given in Algorithm 2. Constructing $\lambda$ takes $O(nR)$ time. Constructing $\Lambda$ by creating a sorted list takes $O(nR \log(nR))$ time. Therefore, the complexity of Algorithm 2 is $O(nR \log(nR))$.

Lemma 4.2. There exists an $\epsilon_B \in \mathbb{R}_{>0}$ such that for all $r_{i,k} \in [0, 1]$, if $\psi_i(z_{i,k}) \geq 0$, then $\psi_i(z_{i,k}) + r_{i,k} \epsilon_B \geq 0$, and if $\psi_i(z_{i,k}) < 0$, then $\psi_i(z_{i,k}) + r_{i,k} \epsilon_B < 0$, for all $i,j \in N$, $1 \leq k \leq |T_i|$, $1 \leq \ell \leq |T_j|$.
Algorithm 3 $\epsilon_A$Finder

1: $\lambda \leftarrow \cup_{i \in N} \{\psi_i(z_{i,k}) \mid \forall z_{i,k} \in T_i\}$  \> All unique virtual valuations
2: if $|\lambda| = 1$ then  \> Everyone has the same virtual valuation
3: $\epsilon_A^U = 1$
4: else
5: $\epsilon_A^U \leftarrow \infty$
6: Let $\Lambda$ be an ascending sequence of virtual valuations in $\lambda$
7: for all $\psi_k, \psi_{k+1} \in \Lambda$ do
8: $\epsilon_A^U \leftarrow \min\{\epsilon_A^U, (\psi_{k+1} - \psi_k)\}$
9: end for
10: end if
11: $\epsilon_A \leftarrow \epsilon_A^U / 2$  \> This satisfies $0 < \epsilon_A < \epsilon_A^U$
12: return $\epsilon_A$

Proof. Let $\epsilon_B^U$ be the minimum of the absolute value of the set of all non-zero virtual valuations: $\epsilon_B^U = \min\{|\psi_i(z_{i,k})| : \psi_i(z_{i,k}) \neq 0, \forall i \in N, \forall z_{i,k} \in T_i\}$. Let $0 < \epsilon_B < \epsilon_B^U$. Any virtual valuation $\psi_i(z_{i,k}) \geq 0$ can have $\epsilon_B$ added to it and remain non-negative. Similarly, any virtual valuation $\psi_i(z_{i,k}) < 0$ can have $\epsilon_B$ added to it and remain negative, since $\epsilon_B < |\psi_i(z_{i,k})|$. See Figure 2.

Figure 2: Graphical depiction of the proof of Lemma 4.2. Any change of the virtual valuations by $\epsilon_B$ does not change whether virtual valuations are negative or not.

Algorithm 4 $\epsilon_B$Finder

1: $\chi \leftarrow \cup_{i \in N} \{\psi_i(z_{i,k}) \mid \forall z_{i,k} \in T_i\} \setminus \{0\}$  \> Set of non-zero virtual valuations of all bidders
2: $\epsilon_B^U \leftarrow 1$
3: for all $\psi \in \chi$ do
4: $\epsilon_B^U \leftarrow \min\{\epsilon_B^U, |\psi|\}$
5: end for
6: $\epsilon_B \leftarrow \epsilon_B^U / 2$  \> This satisfies $0 < \epsilon_B < \epsilon_B^U$
7: return $\epsilon_B$

A procedure for finding an $\epsilon_B$ is given in Algorithm 4. The complexity of Algorithm 4 is $O(nR)$, as constructing $\chi$ takes $O(nR)$ time.

We say that an $\epsilon$ is a valid $\epsilon$ if it satisfies the properties named in Lemmas 4.1 and 4.2 (and restated in Theorem 4.3). The existence of a valid $\epsilon$ is given by the following theorem:

Theorem 4.3. There exists a valid $\epsilon \in \mathbb{R}_{>0}$ such that for all $r_{i,k} \in [0,1]$, the following properties hold:

- if $\psi_i(z_{i,k}) > \psi_j(z_{j,\ell})$, then $\psi_i(z_{i,k}) + r_{i,k}\epsilon > \psi_j(z_{j,\ell}) + r_{j,\ell}\epsilon$,
- if $\psi_i(z_{i,k}) \geq 0$, then $\psi_i(z_{i,k}) + r_{i,k}\epsilon \geq 0$, and
- if $\psi_i(z_{i,k}) < 0$, then $\psi_i(z_{i,k}) + r_{i,k}\epsilon < 0$,

for all $i, j \in N$, $1 \leq k \leq |T_i|$, $1 \leq \ell \leq |T_j|$. 
Proof. Lemma 4.1 shows the existence of an \( \epsilon_A \) which satisfies the first property, and Lemma 4.2 shows the existence of an \( \epsilon_B \) which satisfies the latter properties. These values are not unique: for any \( \epsilon_A, \epsilon_B \), we can construct an \( \epsilon'_A < \epsilon_A \) and \( \epsilon'_B < \epsilon_B \). Thus, the minimum of \( \epsilon_A \) and \( \epsilon_B \) satisfies all three properties of the theorem.

Algorithm 5 \( \epsilon \)-Finder

1: \( \epsilon_A \leftarrow \epsilon_\text{Finder}(\cdot) \)
2: \( \epsilon_B \leftarrow \epsilon_\text{Finder}(\cdot) \)
3: \( \epsilon \leftarrow \min\{\epsilon_A, \epsilon_B\} \)
4: \textbf{return} \( \epsilon \)

A procedure for finding a valid \( \epsilon \) is given in Algorithm 5. The complexity of Algorithm 5 is \( O(n\mathcal{R}\log(n\mathcal{R})) \), because this is the complexity of Algorithm 4 which is larger than the complexity of Algorithm 3, \( O(n\mathcal{R}) \).

Theorem 4.3 tells us that unique virtual valuations may be changed without affecting their ordering among other virtual valuations, and non-negative (negative) virtual valuations can likewise be changed and remain non-negative (negative). Thus, perturbing a unique virtual valuation will not alter its corresponding allocation.

Even more interesting, any tied virtual valuations which are perturbed will no longer be so, provided that changes to virtual valuations are unique. We use this observation to our advantage in order to compute total expected revenue in polynomial time.

Let the perturbed virtual valuation function \( \tilde{\psi}_i : T_i \rightarrow \mathbb{R} \) be defined as

\[
\tilde{\psi}_i(z_{i,k}) = \psi_i(z_{i,k}) + r_{i,k}\epsilon, \quad \forall z_{i,k} \in T_i,
\]  (16)

where all \( r_{i,k} \) variables are drawn independently from a \( U(0,1) \) (continuous) distribution. Since all \( r_{i,k} \) variables are being drawn from a continuous distribution, the probability that \( r_{i,k} = r_{j,\ell} \) is 0, so each bidder’s perturbed virtual valuations should be unique.

Remark 4.4. Due to machine precision issues, it may be the case that there are non-unique \( r_{i,k} \) terms in an actual implementation of the perturbed virtual valuation function. In such a setting, a check may be implemented to see if this is the case. A new set of \( r_{i,k} \) values may be drawn again if non-uniqueness is observed. Drawing \( O(n\mathcal{R}) \) random numbers and checking for uniqueness is \( O(n\mathcal{R}) \), so this should not greatly affect runtime. (One may check for uniqueness by inserting each \( r_{i,k} \) into a hash table, keeping track of the number of times each \( r_{i,k} \) is seen.)

5 Revenue Maximization in Polynomial Time

We now show that drawing \( r_{i,k} \) from a \( U(0,1) \) distribution is akin to picking a winner at random when there are ties. This means that we can compute total expected revenue without any error despite using perturbed virtual valuations when determining allocations.

Theorem 5.1. Total expected revenue can be computed in polynomial time using perturbed virtual valuation functions.

Proof. For any valuation vector \( v \in T \), Algorithm 2 will allocate only to bidders with the highest non-negative virtual valuation. As defined earlier, let \( w(v) \) be the set of bidders with the highest
virtual valuation. Suppose instead of virtual valuations, we used perturbed virtual valuations. Let \( \tilde{w}(v) \) be the set of bidders with the highest perturbed virtual valuations:

\[
\tilde{w}(v) = \left\{ i : \tilde{\psi}_i(v_i) \geq \tilde{\psi}_i^{-1}(0), \tilde{\psi}_i(v_i) = \max \left\{ \tilde{\psi}_j(v_j) : \forall j \in N \right\}, \forall i \in N \right\}. \tag{17}
\]

Using a valid \( \epsilon \) guarantees that the intersection of \( w(v) \) and \( \tilde{w}(v) \) is nonempty. If there are no ties, then \( w(v) = \tilde{w}(v) \).

The interesting case is when there are ties. Since all perturbed virtual valuations are unique, \( |w(v) \cap \tilde{w}(v)| = 1 \), and the unique bidder \( i^* \in w(v) \cap \tilde{w}(v) \) contributes \( \psi_{i^*}(v_{i^*}) \) to the total expected virtual surplus. The probability that \( i \in \tilde{w}(v) \) is allocated depends on the perturbations. Since perturbations are drawn independently and uniformly at random, the \( r_{i,k} \) values act as tie-breaking rules, where the probability that any \( j \in w(v) \) wins is uniform over the cardinality of \( w(v) \), just as in Algorithm \([1]\). The maximum virtual surplus attained from any convex combination of winners in \( w(v) \) where \( \sum_{j \in w(v)} x_j(v_j, v_{-j}) = 1 \) is \( \sum_{j \in w(v)} \psi_j(v_j) x_j(v_j, v_{-j}) = \max \{ \psi_j(v_j) : \forall j \in w(v) \} \), which is the outcome of Algorithm \([1]\). In Algorithm \([2]\) the virtual surplus given by \( v \) is \( \psi_{i^*}(v_{i^*}) \).

Since \( \max \{ \psi_j(v_j) : \forall j \in w(v) \} = \psi_{i^*}(v_{i^*}) \), the contribution any \( v \in T \) has on total expected revenue is equivalent in both algorithms.

\[ \square \]

**Remark 5.2.** Our choice of \( \epsilon \) and \( r_{i,k} \) ensures that virtual valuations are perturbed upwards. While it may be possible to perturb virtual valuations down with an appropriate choice of \( \epsilon \) and values drawn from, say, a \( U(-1,1) \) distribution, so that \( \tilde{\psi}_i(z_{i,k}) < \tilde{\psi}_i(z_{i,k}) \) is possible, any virtual valuation equal to zero may become negative. Our choice of \( \epsilon \) and \( U(0,1) \) was made to avoid this possibility, so that a bidder with a zero virtual valuation may be allocated.

The analysis and remark given is not specific to virtual valuations, and can be applied to valuations as well. We can construct a perturbed valuation function, \( \tilde{\psi}_i : T \rightarrow \mathbb{R} \), where \( \tilde{\psi}_i(z_{i,k}) = z_{i,k} + r_{i,k} \epsilon' \) for all \( i \in N \), \( \forall z_{i,k} \in T_i \), with a valid \( \epsilon' \) computed using valuations instead of virtual valuations, so we have the following corollary:

**Corollary 5.3.** Total expected surplus can be computed in polynomial time using perturbed valuation functions.

### 6 Interim Allocations and Payments

Let \( v \) be a bidder profile where multiple bidders have the highest virtual valuation. Ordinarily, each bidder \( i \in w(v) \) has probability \( p = |w(v)|^{-1} \) of being allocated: i.e., \( x_i(v_i, v_{-i}) = p \). However, by using perturbed virtual valuations, allocations \( \tilde{x}_i(v_i, v_{-i}) \) are either 0 with probability \( 1 - p \), or 1 with probability \( p \). This means \( \mathbb{E}[\tilde{x}_i(v_i, v_{-i})] = 1(p) + 0(1-p) = p \). While \( x_i(v_i, v_{-i}) \neq \tilde{x}_i(v_i, v_{-i}) \), it holds that \( x_i(v_i, v_{-i}) = \mathbb{E}[\tilde{x}_i(v_i, v_{-i})] \).

Observe that \( \tilde{x}_i(v_i, v_{-i}) \) is a Bernoulli random variable with mean \( p \) and variance \( p(1-p) \). Assuming multiple runs of Algorithm \([2]\), let \( \tilde{x}_i^d(v_i, v_{-i}) \) be an allocation of the \( d \)th run. By Chebyshev’s inequality, for a fixed \( a > 0 \),

\[
\Pr \left( \left| \frac{\sum_{d=1}^{D} \tilde{x}_i^d(v_i, v_{-i})}{D} - x_i(v_i, v_{-i}) \right| \geq a \right) \leq \frac{p(1-p)}{a^2 D^2}. \tag{18}
\]

This suggests that for a fixed \( v \), we can recover a good estimate of the allocation \( x_i(v_i, v_{-i}) \) obtained by Algorithm \([1]\) with high probability after a few runs of Algorithm \([2]\).
Interim allocations \( \hat{x}_i(v_i) \) rely on the allocations of \( x_i(v_i, v_{-i}) \) for every \( v_{-i} \in T_{-i} \). To obtain a good estimate of an interim allocation \( \hat{x}_i(v_i) \) given by Algorithm 1, we can use a union bound to determine how many times we should run Algorithm 2:

\[
\sum_{v_{-i} \in T_{-i} : |w(v_i, v_{-i})| > 1} \Pr \left( \left| \frac{\sum_{d=1}^{D} \tilde{x}_d(v_i, v_{-i})}{D} - x_i(v_i, v_{-i}) \right| \geq a \right) \leq \sum_{v_{-i} \in T_{-i} : |w(v_i, v_{-i})| > 1} \left| w(v_i, v_{-i}) \right|^{-1} \left( 1 - \left| w(v_i, v_{-i}) \right|^{-1} \right) \frac{a^2 D^2}{1 - \left| w(v_i, v_{-i}) \right|^{-1}}.
\]

In words, Chebyshev’s inequality tells us that the number of times we should run Algorithm 2 is directly proportional to the number of ties there are. When there are no ties, only one run is sufficient to obtain the result given by Algorithm 1. An exponential number of ties would require an exponential number of runs to recover a good estimate of interim allocations given by Algorithm 1. With a good estimate of interim allocations, a good estimate of interim payments can then be computed by using Myerson’s payment formula, Equation (2).

Remark 6.1. As in Myerson [1981], we assumed our problem is regular, so that \( \psi_i(z_{i,k+1}) > \psi_i(z_{i,k}) \) whenever \( z_{i,k+1} > z_{i,k} \). However, some works, such as Hartline [2013], also use regularity to describe distributions, so that virtual valuations are not strictly increasing: i.e., \( \psi_i(z_{i,k+1}) \geq \psi_i(z_{i,k}) \) whenever \( z_{i,k+1} > z_{i,k} \). By adding small random perturbations, we cannot guarantee that \( \tilde{\psi}_i(z_{i,k+1}) \geq \tilde{\psi}_i(z_{i,k}) \) whenever \( \psi_i(z_{i,k+1}) = \psi_i(z_{i,k}) \). In expectation, whenever \( \psi_i(z_{i,k+1}) = \psi_i(z_{i,k}) \), \( \mathbb{E} \left[ \tilde{\psi}_i(z_{i,k+1}) \right] = \mathbb{E} \left[ \tilde{\psi}_i(z_{i,k}) \right] \), so averaging the result across multiple runs of Algorithm 2 preserves virtual valuation monotonicity, and hence allocation monotonicity, in expectation.

7 Conclusion

We show how to solve for optimal revenue and surplus in the classic auction setting (one good, many bidders) in polynomial time without resorting to linear programming. Instead, our algorithm works much like Myerson’s classic optimal auction. First, we point out that expediting Myerson’s approach is easy when there are no ties. In the case of ties, we provide a method for altering virtual valuations slightly so that 1. ties are eliminated, and 2. any virtual valuation with a non-negative probability of winning maintains a very similar non-negative probability of winning. We then show how to use these modified virtual valuations to compute total expected revenue (or surplus, if applied to valuations) in polynomial time. Finally, we show how such a procedure can be used to obtain estimates of interim allocations and interim payments. This analysis reveals that, while computing total expected revenue (or surplus) is always polynomial using our approach, recovering interim allocations and interim payments may take exponential time (in the case of exponentially-many ties). In the future, we would like to investigate how our analysis can be extended to other single-parameter settings, such as knapsack auctions and sponsored search.

8 Acknowledgments

This research was supported by NSF Grant #1217761.
References


