Stability and Incentive Compatibility in a Kernel-Based Combinatorial Auction

Sébastien Lahaie
Yahoo! Research
New York, NY 10018
laahaies@yahoo-inc.com

Abstract

We present the design and analysis of an approximately incentive-compatible combinatorial auction. In just a single run, the auction is able to extract enough value information from bidders to compute approximate truth-inducing payments. This stands in contrast to current auction designs that need to repeat the allocation computation as many times as there are bidders to achieve incentive compatibility. The auction is formulated as a kernel method, which allows for flexibility in choosing the price structure via a kernel function. Our main result characterizes the extent to which our auction is incentive-compatible in terms of the complexity of the chosen kernel function. Our analysis of the auction’s properties is based on novel insights connecting the notion of stability in statistical learning theory to that of universal competitive equilibrium in the auction literature.

Introduction

The purpose of an auction is typically to determine an efficient allocation of resources. To achieve this, the auction must not only compute the value maximizing allocation, but also incentivize buyers to truthfully report their values so that the maximization is performed over the right objective. In the process of computing an efficient allocation, auctions—especially iterative auctions—often derive clearing prices that balance supply and demand. However, when one deals with combinatorial auctions over multiple items, these prices may not induce truthful reporting. Instead, one must perform further computation to obtain truth-inducing payments. Due to a variety of appealing properties, the Vickrey-Clarke-Groves (VCG) payment scheme is often the method of choice to achieve incentive compatibility (Clarke 1971; Groves 1979; Vickrey 1961).

This situation has led to much research into combinatorial auctions that derive VCG payments as a by-product of the allocation and clearing process, and into characterizations of the conditions under which prices can match VCG payments. Bikhchandani and Ostroy (2002) demonstrate that the latter is the case when valuations satisfy a substitutes condition. Ausubel (2006) and de Vries et al. (2007) develop linear and nonlinear price iterative auctions that compute VCG payments under this substitutes condition. Mishra and Parkes (2007) develop an ascending-price iterative auction for general valuations that terminates with sufficient information to compute VCG payments.

The sealed-bid VCG mechanism itself can be run to compute incentivizing payments. This requires solving the allocation problem once with all buyers present and \(n\) more times each with one agent removed, where \(n\) is the number of agents, for an \(n\)-fold increase in computation over simple efficient allocation. In fact, all the auctions mentioned above also introduce an \(n\)-fold increase in computation, either implicitly or explicitly, by requiring multiple price paths or increasing the computation in each round—their advantages lie not in their computational savings but elsewhere (e.g., ascending prices, incremental elicitation). The question of how much increase in computation is needed to compute VCG payments goes back to the work of Nisan and Ronen (1999), who posed it specifically in the context of routing on a network.

In this work, we study the extent to which clearing prices computed by an auction can capture the information needed to derive VCG payments after just a single run of the allocation process. We adapt the original formulation of the combinatorial allocation problem given by Bikhchandani and Ostroy (2002) in two ways, drawing on ideas from kernel methods in machine learning. First, following Lahaie (2009), we introduce some flexibility in the choice of price structure via a kernel function that specifies how bundles are priced. Second, we introduce a penalty term on the magnitude of the clearing prices, akin to a regularization term in machine learning. This formulation of the allocation problem can form the basis of single-shot or iterative combinatorial auctions, using standard techniques from the literature (Bikhchandani et al. 2001).

Our main result characterizes how well VCG payments can be approximated using the price information computed by our auction formulation, in terms of the price structure and penalty term. Our analysis draws a connection between the notion of stability (Bousquet and Elisseeff 2002) in statistical learning theory and the notion of a universal competitive equilibrium (Mishra and Parkes 2007) in the auction literature. Informally, stability ensures that a learning algorithm is not too sensitive to any particular input in the training data, so that the algorithm can therefore avoid overfitting and generalize well. Under a universal competitive equilib-
rium, the clearing prices remain clearing even if an agent is removed; thus, the prices are not sensitive to the presence of any one agent. This does not mean that we achieve incentive-compatibility with such prices, but that they capture an agent’s marginal contribution to the economy, which is the information needed to compute VCG payments.

The remainder of the paper is organized as follows. We first present the model, explaining the distinction between prices and payments in auctions. We then provide our kernel-based auction formulation. The following sections contain our results, connecting the concepts of stability, universal competitive equilibrium, VCG payments, and incentive compatibility. We conclude with a discussion of our bounds and avenues for future research.

The Model
There are \( n \) buyers and \( m \) distinct indivisible items held by a single seller. A bundle is a subset of the items. We associate each bundle with its indicator vector, and denote the set of bundles by \( X = \{0, 1\}^m \). We write \( x \leq x' \) to denote that bundle \( x \) is contained in bundle \( x' \) (the inequality is understood component-wise).

Buyer \( i \) has a value function \( v_i : X \rightarrow \mathbb{R}_+ \), denoting how much it is willing to pay for each bundle. For the sake of simplicity, we assume that each buyer is single-minded, meaning that there is a bundle \( x_i \) such that \( v_i \) is entirely determined by the bundle-value pair \( (x_i, v_i(x_i)) \) as follows:

\[
v_i(x) = \begin{cases} v_i(x_i) & \text{if } x \geq x_i \\ 0 & \text{otherwise} \end{cases}
\]

with the requirement that \( v_i(\emptyset) = 0 \). In words, buyer \( i \) would like to acquire all the items in bundle \( x_i \), but is not interested in any others.\(^1\)

A selection of buyers is simply a subset \( I \subseteq \{1, \ldots, n\} \). A selection if feasible if \( \sum_{i \in I} x_i \leq 1 \); this means that the desired bundles of the buyers in the selection do not intersect. We denote the set of feasible selections by \( \mathcal{F} \). A selection \( I \) is efficient if \( I \in \arg\max_{I' \in \mathcal{F}} \sum_{i \in I'} v_i(x_i) \). In this paper we will typically denote the allocation of items corresponding to a selection by a vector of bundles \( y = (y_1, \ldots, y_n) \), where \( y_i \) is the bundle acquired by buyer \( i \); we have \( y_i = x_i \) if \( i \in I \) and \( y_i = 0 \) otherwise. Together with an efficient selection, we wish to determine transfers from the buyers to the auctioneer. Transfers can serve different purposes.

Prices In the first instance, transfers are specified as prices \( p : X \rightarrow \mathbb{R} \) defined over bundles; we will assume that prices are normalized so that \( p(\emptyset) = 0 \). The purpose of prices is to balance supply and demand. Prices are competitive with respect to a selection \( I \) if the following conditions hold:

\[
\begin{align*}
v_i(x_i) - p(x_i) & \geq 0 & \text{for } i \in I \\
v_i(x_i) - p(x_i) & \leq 0 & \text{for } i \notin I \\
\sum_{i \in I} p(x_i) & \geq \sum_{i \notin I} p(x_i) & \text{for all } I' \in \mathcal{F}
\end{align*}
\]

This means that each buyer in the selection willingly acquires its associated bundle, the other buyers willingly receive nothing, and the selection maximizes the seller’s revenue. In this sense, the prices balance supply and demand, assuming the agents act as pure price-takers. If prices \( p \) are competitive with respect to a selection \( I \), we say that they support the selection and the associated allocation \( y \).

It is well-known that if prices \( p \) are competitive with respect to a selection, then the selection is efficient (Bikhchandani and Ostroy 2002). It is also known that with single-minded bidders, competitive prices exist to support any efficient selection (Lahaie and Parkes 2009); however, existence may not hold if we impose a specific structure on the prices, such as linearity.\(^2\) Part of our goal in designing our auction is to allow for flexibility in the price structure, in order to study the interplay between price structure and incentives.

Payments In the second instance, transfers are specified as payments \( q = (q_1, \ldots, q_n) \in \mathbb{R}^n \) from the buyers to the seller. The purpose of payments is to align the buyers’ incentives with that of efficiency. A rational buyer will realize that both the selection and its payment in an auction will depend on the value function it reports, and will therefore choose its report to maximize its own surplus (i.e., it may lie about its value function), where surplus refers to value minus price. We seek a payment rule such that an agent’s potential gain from misreporting its value is small.

Formally, let \( (y, q) \) be the allocation and payments computed by the auction when buyer \( h \) reports its value function truthfully, holding the other buyers’ values fixed, and let \( (y', q') \) be the allocation and payments computed when it reports some other value function. The auction is \( \epsilon \)-incentive compatible (in dominant strategies), where \( \epsilon \geq 0 \), if it is always the case that

\[
v_h(y_h) - q_h + \epsilon \geq v_h(y'_h) - q'_h.
\]

If \( \epsilon \) is not explicitly mentioned, it is implied that \( \epsilon = 0 \). A natural approach would be to charge \( q_i = p(y_i) \) for each buyer \( i \), but with general or even single-minded valuations this may not be \( \epsilon \)-incentive compatible for any small \( \epsilon \) (Ausubel and Milgrom 2006), no matter what competitive prices are chosen. A standard way to achieve incentive-compatibility is to use the Vickrey-Clarke-Groves payment scheme, to be defined in the next section.

The Auction
Kernels To achieve flexibility in the price structure, we follow Lahaie (2009) and draw on ideas from kernel methods in machine learning. To compute nonlinear competitive prices, we view these as linear prices in a higher-dimensional space to which we map the bundles via a mapping \( \phi : X \rightarrow \mathbb{R}^M \), where usually \( M \gg m \). Given

\(^1\)All our results extend to arbitrary valuations over bundles, but the auction formulation and analysis would be needlessly complex. Only Lemma 1 would have to be generalized, the remaining proofs would remain unchanged.

\(^2\)When generalizing our results to arbitrary valuations, we may need personalized prices to ensure the existence of competitive prices (i.e., we may need \( p_i(x) \neq p_j(x) \) for two distinct buyers \( i \) and \( j \)). This can be handled in our model by considering bundle-agent pairs \((x, i)\) instead of just bundles, and redefining valuations so that an agent’s value for a bundle-value pair is zero unless the bundle is earmarked for that agent.
\( p \in \mathbb{R}^M \), the price of bundle \( x \) is then \( \langle p, \phi(x) \rangle \). These linear prices in \( \mathbb{R}^M \) translate into nonlinear prices over the original bundle space \( X \). (With a slight abuse of notation, we will often write \( p(x) \) to mean \( \langle p, \phi(x) \rangle \), keeping \( \phi \) implicit.) The issue here is that the dimension \( M \) may be very large (even infinite), so it may be infeasible to work explicitly with prices in \( \mathbb{R}^M \).

The trick used to address this in kernel methods is to formulate the relevant problem (e.g., classification, regression) as a mathematical program that relies only on the inner products \( \langle \phi(x_i), \phi(x_j) \rangle \). What makes this practical is that, for many kinds of mappings, the inner products can be efficiently evaluated in time that does not depend on \( M \). The inner products are given via a kernel function \( k \).

We will assume that the mapping \( \phi \) is defined as \( k(x, x') = \langle \phi(x), \phi(x') \rangle \). Rather than imposing a hard constraint that the outcome is sufficiently complex to allow it (Lahaie 2009). With the identity kernel, for instance, there is an integer solution (Bikhchandani and Ostroy 2002).

The dual of the quadratic program is the following, denoted (D):

\[
\min_{\pi \geq 0, \pi \geq 0, p} \sum_{i=1}^{n} \pi_i \geq \langle p, \phi(x_i) \rangle \quad i = 1, \ldots, n
\]

\[
\text{s.t. } \pi_i \geq v_i(x_i) - \langle p, \phi(x_i) \rangle \quad i = 1, \ldots, n
\]

\[
\bar \pi \geq \left\langle p, \sum_{i \in I} \phi(x_i) \right\rangle \quad I \in \mathcal{F}
\]

We see that this program computes competitive prices. By complementary slackness, if \( \alpha_i = 1 \), then agent \( i \) obtains \( y_i = x_i \) and \( \pi_i = v_i(x_i) - \langle p, \phi(x_i) \rangle \geq 0 \); whereas if \( \alpha_i = 0 \), then \( y_i = \emptyset \) and \( 0 \geq v_i(x_i) - \langle p, \phi(x_i) \rangle \). Similarly, the selection corresponding to \( \bar \alpha \) maximizes the seller’s revenue. Of course, if the primal does not find an integer solution, then the dual solution is not useful. By the relationship between the primal and dual solutions, the prices \( p \) can be evaluated as

\[
p(x) = \frac{1}{\lambda} \sum_{i=1}^{n} \left( \alpha_i - \sum_{I \ni i} \bar \alpha_i \right) k(x_i, x).
\]
Incentive Compatibility

We show in this section that our auction is approximately incentive compatible when it charges payments \( r \). Let \( \kappa \) be a constant such that \( \|\phi(x)\| \leq \kappa \) for all \( x \in X \). This constant can be seen as a crude measure of the "complexity" of the associated price structure: smaller \( \kappa \) will correspond to more complex price structures. For instance, we have \( \kappa = \sqrt{m} \) for the linear kernel and \( \kappa = 1 \) for the identity kernel. Some proofs in this section appeal to results from convex analysis that can be found in (Bertsekas 1999), abbreviated (B) in the proofs.

Stability

We begin by showing that for large enough \( \lambda \), the presence or absence of a buyer \( h \) does not impact the optimal dual solution (the price vector) significantly. We first formulate the dual problem in a slightly more convenient form. The surplus functions of buyer \( i \) and the seller are defined as follows:

\[
\pi_i(p) = \max \{ v_i(x_i) - p_x_i, 0 \} \\
\bar{\pi}(p) = \max_{i \in I} \sum_{i \in I} p(x_i)
\]

These functions are convex, being the maximum of linear functions of \( p \). We have the following bound on their subgradients.

Lemma 1 For a single-minded buyer \( i \), we have \( \|\pi_i'\| \leq \|\phi(x_i)\| \) for any \( p \) and \( \pi_i' \in \partial \pi_i(p) \).

Proof. Let \( f(p_i) = p_i \) for \( p_i \in \mathbb{R} \) and let \( g(p) = f((\phi(x_i), p)) \) for \( p \in \mathbb{R}^M \). Note that \( \pi_i(p) = \max \{ g(p), 0 \} \).

The crucial point is that each payment \( r_h \) is solely a function of the prices derived after solving (P) once. This distinguishes our approach from all the auctions mentioned in the introduction, which implicitly solve their underlying allocation problem \( n + 1 \) times to obtain VCG payments.

Computation

Although the purpose of this paper is to analyze the properties of solutions to (P), let us discuss briefly how the program could be solved in practice—different ways of solving the program would correspond to different ways of running the auction (e.g., a single-shot or iterative auction). The kernel trick has made the number of constraints manageable, but the number of \( \alpha_i \) variables is far too large to formulate (P) explicitly even for moderate \( m \). A classic way to solve programs with large numbers of variables is to apply delayed column generation (Bertsekas 1999). In fact, a now standard way to design iterative combinatorial auctions is to formulate the selection problem with variables for each allocation as in (P), and apply a primal-dual or subgradient algorithm with delayed column generation to reach a solution (Bikhchandani et al. 2001). This approach has proven effective in practice (Parkes and Ungar 2000). The same issue arises with support vector machines over large datasets. There, cutting plane algorithms have been developed to perform tasks such as support vector regression (Joachims 2006). Given the strong resemblance between our problem and support vector machines, it should be possible to leverage ideas from that literature as well. This is the subject of ongoing work.

Incentive Compatibility

We show in this section that our auction is approximately incentive compatible when it charges payments \( r \). Let \( \kappa \) be a constant such that \( \|\phi(x)\| \leq \kappa \) for all \( x \in X \). This constant can be seen as a crude measure of the “complexity” of the associated price structure: smaller \( \kappa \) will correspond to more complex price structures. For instance, we have \( \kappa = \sqrt{m} \) for the linear kernel and \( \kappa = 1 \) for the identity kernel. Some proofs in this section appeal to results from convex analysis that can be found in (Bertsekas 1999), abbreviated (B) in the proofs.

Stability

We begin by showing that for large enough \( \lambda \), the presence or absence of a buyer \( h \) does not impact the optimal dual solution (the price vector) significantly. We first formulate the dual problem in a slightly more convenient form. The surplus functions of buyer \( i \) and the seller are defined as follows:

\[
\pi_i(p) = \max \{ v_i(x_i) - p_x_i, 0 \} \\
\bar{\pi}(p) = \max_{i \in I} \sum_{i \in I} p(x_i)
\]

These functions are convex, being the maximum of linear functions of \( p \). We have the following bound on their subgradients.

Lemma 1 For a single-minded buyer \( i \), we have \( \|\pi_i'\| \leq \|\phi(x_i)\| \) for any \( p \) and \( \pi_i' \in \partial \pi_i(p) \).

Proof. Let \( f(p_i) = p_i \) for \( p_i \in \mathbb{R} \) and let \( g(p) = f((\phi(x_i), p)) \) for \( p \in \mathbb{R}^M \). Note that \( \pi_i(p) = \max \{ g(p), 0 \} \).

The crucial point is that each payment \( r_h \) is solely a function of the prices derived after solving (P) once. This distinguishes our approach from all the auctions mentioned in the introduction, which implicitly solve their underlying allocation problem \( n + 1 \) times to obtain VCG payments.

Computation

Although the purpose of this paper is to analyze the properties of solutions to (P), let us discuss briefly how the program could be solved in practice—different ways of solving the program would correspond to different ways of running the auction (e.g., a single-shot or iterative auction). The kernel trick has made the number of constraints manageable, but the number of \( \alpha_i \) variables is far too large to formulate (P) explicitly even for moderate \( m \). A classic way to solve programs with large numbers of variables is to apply delayed column generation (Bertsekas 1999). In fact, a now standard way to design iterative combinatorial auctions is to formulate the selection problem with variables for each allocation as in (P), and apply a primal-dual or subgradient algorithm with delayed column generation to reach a solution (Bikhchandani et al. 2001). This approach has proven effective in practice (Parkes and Ungar 2000). The same issue arises with support vector machines over large datasets. There, cutting plane algorithms have been developed to perform tasks such as support vector regression (Joachims 2006). Given the strong resemblance between our problem and support vector machines, it should be possible to leverage ideas from that literature as well. This is the subject of ongoing work.

Incentive Compatibility

We show in this section that our auction is approximately incentive compatible when it charges payments \( r \). Let \( \kappa \) be a constant such that \( \|\phi(x)\| \leq \kappa \) for all \( x \in X \). This constant can be seen as a crude measure of the “complexity” of the associated price structure: smaller \( \kappa \) will correspond to more complex price structures. For instance, we have \( \kappa = \sqrt{m} \) for the linear kernel and \( \kappa = 1 \) for the identity kernel. Some proofs in this section appeal to results from convex analysis that can be found in (Bertsekas 1999), abbreviated (B) in the proofs.

Stability

We begin by showing that for large enough \( \lambda \), the presence or absence of a buyer \( h \) does not impact the optimal dual solution (the price vector) significantly. We first formulate the dual problem in a slightly more convenient form. The surplus functions of buyer \( i \) and the seller are defined as follows:

\[
\pi_i(p) = \max \{ v_i(x_i) - p_x_i, 0 \} \\
\bar{\pi}(p) = \max_{i \in I} \sum_{i \in I} p(x_i)
\]

These functions are convex, being the maximum of linear functions of \( p \). We have the following bound on their subgradients.

Lemma 1 For a single-minded buyer \( i \), we have \( \|\pi_i'\| \leq \|\phi(x_i)\| \) for any \( p \) and \( \pi_i' \in \partial \pi_i(p) \).

Proof. Let \( f(p_i) = p_i \) for \( p_i \in \mathbb{R} \) and let \( g(p) = f((\phi(x_i), p)) \) for \( p \in \mathbb{R}^M \). Note that \( \pi_i(p) = \max \{ g(p), 0 \} \).

The crucial point is that each payment \( r_h \) is solely a function of the prices derived after solving (P) once. This distinguishes our approach from all the auctions mentioned in the introduction, which implicitly solve their underlying allocation problem \( n + 1 \) times to obtain VCG payments.

Computation

Although the purpose of this paper is to analyze the properties of solutions to (P), let us discuss briefly how the program could be solved in practice—different ways of solving the program would correspond to different ways of running the auction (e.g., a single-shot or iterative auction). The kernel trick has made the number of constraints manageable, but the number of \( \alpha_i \) variables is far too large to formulate (P) explicitly even for moderate \( m \). A classic way to solve programs with large numbers of variables is to apply delayed column generation (Bertsekas 1999). In fact, a now standard way to design iterative combinatorial auctions is to formulate the selection problem with variables for each allocation as in (P), and apply a primal-dual or subgradient algorithm with delayed column generation to reach a solution (Bikhchandani et al. 2001). This approach has proven effective in practice (Parkes and Ungar 2000). The same issue arises with support vector machines over large datasets. There, cutting plane algorithms have been developed to perform tasks such as support vector regression (Joachims 2006). Given the strong resemblance between our problem and support vector machines, it should be possible to leverage ideas from that literature as well. This is the subject of ongoing work.
Subtracting (6) from (5), then taking the inner product with 
\( p - p^{-h} \) and rearranging, we have
\[
0 = \sum_{i \neq h} \langle \pi_i'(p) - \pi_i'(p^{-h}), p - p^{-h} \rangle + \langle \pi_h'(p), p - p^{-h} \rangle + \lambda \| p - p^{-h} \|^2.
\]
By the monotonicity property of subgradients (an easy con-
sequence of their definition) we have \( \langle \pi_i'(p) - \pi_i'(p^{-h}), p - p^{-h} \rangle \geq 0 \) for each buyer \( i \neq h \) and the same for the seller term. Thus we have \( \langle \pi_h'(p), p - p^{-h} \rangle + \lambda \| p - p^{-h} \|^2 \leq 0 \).

Continuing,
\[
\lambda \| p - p^{-h} \|^2 \leq \langle \pi_h'(p), p - p^{-h} - p \rangle \\
\leq \| \pi_h'(p) \| \| p - p^{-h} - p \| \\
\leq \| \phi(x_h) \| \| p - p^{-h} \| \\
\leq \kappa \| p - p^{-h} \|.
\]
The third inequality follows from Lemma 1, and the sec-
ond follows from the Cauchy-Schwarz inequality. This com-
pletes the proof. □

It may seem curious that Proposition 1 does not depend on
\( \phi_h(x_h) \). After all, if buyer \( h \) has a high value, then its ab-
scence should have a higher impact on the objective and thus
on the solution than the absence of low value buyers. This
reasoning is misleading for the following reason. Consider
the dual objective with a very large \( \lambda \) relative to the values.
In this case the quadratic program will essentially try to mini-
imize \( \| p \| \), no matter what the values. Thus the proximity of
\( p \) and \( p^{-h} \) is expressed in terms of \( \lambda \) alone. We see from
the proof though that the size of bundle \( x_h \) does impact the
proximity of the two price vectors, and this is the origin of
the \( \kappa \) term.

**Universal Competitiveness** The connection between sta-
bility and approximate incentive compatibility comes via the
concept of universal competitive equilibrium, introduced by
Mishra and Parkes (2007). Prices \( p \) are universally com-
petitive if they support not only the efficient allocation, but
also any efficient allocation \( y^{-h} \) that arises when buyer \( h \)
is removed, for any buyer \( h \). We show that our program
computes approximately universally competitive prices. We
believe this result is interesting in its own right.

**Proposition 2** Let \( p \) be the supporting prices computed by
the dual. Let \( y^{-h} \) be the efficient allocation computed when
buyer \( h \) is removed, and let \( \hat{y}^{-h} \) be an arbitrary allocation
among the agents with \( h \) removed. Then for each buyer \( i \neq h \) and for the seller, we have
\[
\begin{align*}
v_i(y^{-h}_i) - p(y^{-h}_i) + \frac{\kappa^2}{\lambda} & \quad \geq \quad v_i(y'_i) - p(y'_i) \\
\sum_{i \neq h} p(y^{-h}_i) + \frac{(n-1)\kappa^2}{\lambda} & \quad \geq \quad \sum_{i \neq h} p(y'_i)
\end{align*}
\]

**Proof.** We have the following derivation.
\[
\begin{align*}
\nu_i(y'_i) - p(y'_i) \\
= \quad v_i(y'_i) - p^{-h}(y'_i) + p^{-h}(y'_i) - p(y'_i) \\
\leq \quad v_i(y^{-h}_i) - p^{-h}(y^{-h}_i) + p^{-h}(y'_i) - p(y'_i) \\
= \quad v_i(y^{-h}_i) - p(y^{-h}_i) + p(y^{-h}_i) - p^{-h}(y^{-h}_i) \\
& \quad + p^{-h}(y'_i) - p(y'_i) \\
\leq \quad v_i(y^{-h}_i) - p(y^{-h}_i) + (p^{-h} - p, \phi(y^{-h}_i) - \phi(y^{-h}_i)) \\
\leq \quad v_i(y^{-h}_i) - p(y^{-h}_i) + \| p^{-h} - p \| \| \phi(y'_i) - \phi(y^{-h}_i) \| \\
\leq \quad v_i(y^{-h}_i) - p(y^{-h}_i) + \frac{\kappa^2}{\lambda}.
\end{align*}
\]
The first inequality follows from the fact that prices \( p^{-h} \) sup-
port the allocation \( y^{-h}, \hat{y}^{-h} \). The second follows from the
Cauchy-Schwarz inequality, and the last from Proposition 1.
The line of argument is completely analogous for the seller’s
revenue bound. □

Mishra and Parkes (2007) showed that universally compet-
itive prices, besides being competitive, also contain all the
information necessary to compute VCG payments. From the
approximately universally competitive prices \( p \) computed by
our auction, we will therefore be able to derive approximate
VCG payments and achieve approximate incentive compati-

**VCG Payments** We are now ready to prove our main re-
result, characterizing how well payments \( r \) approximate VCG
payments.

**Proposition 3** For each buyer \( i \), the payments \( q \) and \( r \) sat-
sify
\[
r_i \geq q_i \geq v_i - \frac{2(n-1)\kappa^2}{\lambda}.
\]

**Proof.** Let \( y \) be the efficient allocation computed by the
auction together with prices \( p \), let \( y^{-h} \) be the efficient allo-
cation computed when buyer \( h \) is removed, and let \( \hat{y}^{-h} \) be a
revenue-maximizing allocation among buyers \( i \neq h \) with
respect to \( p \). Note that for any buyer \( i \neq s \),
\[
\sum_{i \neq h} \left[ v_i(y^{-h}_i) - v_i(y'_i) \right] + \left[ p(y^{-h}_i) - p(y'_i) \right] \\
\geq \quad p(y^{-h}_i) - p(y'_i) \\
\geq \quad \frac{(n-1)\kappa^2}{\lambda}.
\]

where (7) follows from Proposition 2 and (8) by the fact that
\( p \) supports \( y \). Summing (8) over all \( i \neq h \) combined with
the fact that \( \hat{y}^{-h} \) maximizes revenue at prices \( p \) (among
\( i \neq h \)) yields the upper bound. Summing (7) over all \( i \neq h \)
combined with the seller bound in Proposition 2 yields the
lower bound. □

Approximate incentive compatibility now follows almost
immediately.
Theorem 1 The auction that charges payments $r$ is $\epsilon$-incentive compatible for $\epsilon = \frac{2(n-1)\kappa^2}{\lambda}$.

Proof. Let $(y, r)$ be the allocation and payments computed when buyer $h$ reports its value function truthfully, and let $(y', r')$ be the resulting allocation and payments when it reports some other value. We have:

$$v_h(y_h) - r_h \geq v_h(y_h) - q_h - \frac{2(n-1)\kappa^2}{\lambda}$$
$$\geq v_h(y'_h) - q'_h - \frac{2(n-1)\kappa^2}{\lambda}$$
$$\geq v_h(y'_h) - r'_h - \frac{2(n-1)\kappa^2}{\lambda}.$$

The first inequality follow from the lower bound in Proposition 3, the second from the incentive compatibility of the VCG payment scheme, and the third from the upper bound in Proposition 3.

Discussion

The error $\epsilon$ in Theorem 1 can be reduced in two ways: by using a price structure with high complexity (small $\kappa$) or a large penalty term (large $\lambda$). But note that both cannot be improved in tandem. With more complex price structures, the allocation problem becomes increasingly difficult and one needs an increasingly small $\lambda$ to achieve feasibility within a reasonable tolerance (Lahaie 2009).

Holding $k$ and hence $\kappa$ fixed, one way to maximize $\lambda$ is to begin by setting it large, and then solve (P) over rounds using updates $\lambda \leftarrow \tau \lambda$, where $\tau \in [0, 1]$, until the requisite tolerance on feasibility is achieved. This is precisely the approach taken by penalty methods for solving programs such as ours (Bertsekas 1999). Taking $\tau$ close to 1 will improve the minimization, but also increase the computation time.

To minimize the error in Theorem 1, one could then take the following approach. Consider price structures of increasing complexity, each time minimizing $\lambda$ as just described to obtain the best possible error $\epsilon$. There are common kernels, such as the Gaussian and polynomial kernels (Schölkopf and Smola 2001), that have parameters allowing one to adjust their complexity. Eventually one will reach a point of diminishing returns, settling on the optimal trade-off between the complexity and penalty terms. This approach is analogous to that of structural risk minimization in statistical learning theory (Vapnik 1998), where one trades off the ability of a kernel to fit training data with its ability to generalize.

Finally, let us examine how closely the price $p(y_h)$ buyer $h$ faces can match its approximate VCG payment $r_h$. Note that $p_h - r_h = \sum_i p(y_i) - \sum_{i \neq h} p(y_i^{-h}) \geq 0$, so an agent’s payment never exceeds its price (recall that $p$ supports $y$). By the Cauchy-Schwarz inequality, this difference is bounded by $\|p\|$ multiplied by the term $\delta = \left\| \sum_i \phi(y_i) - \sum_{i \neq h} \phi(y_i^{-h}) \right\|$. The latter quantity can be seen as an indicator of the distance between prices and payments. For instance, note that with the linear kernel, we have $\sum_i \phi(y_i) = \sum_{i \neq h} \phi(y_i^{-h}) = 1$, so $\delta = 0$. This means that if our auction can successfully find an integer solution with this kernel, then the resulting prices satisfy $p(y_h) = r_h$ and correspond to approximate VCG payments. On the other hand, we see that $\delta$ can be as large as $\sqrt{2n - 1}$ with the identity kernel. This suggests that prices and payments diverge with increasingly complex price structures. In general, the development of new complexity measures to characterize economic properties of kernels appears to be a stimulating avenue for future research.

References


