The Design and Implementation of a Combinatorial Exchange*

Sébastien M. Lahaie  David C. Parkes

Abstract

We describe a preliminary design for an iterative combinatorial exchange. The exchange is an iterated version of a one-shot design that divides surplus so as to satisfy budget-balance and individual-rationality, and minimize incentives to manipulate the mechanism. The exchange implements an efficient outcome with respect to the participants' reported values. It provides feedback to participants with linear prices over the items, and ensures progress with activity rules. Participants communicate with the exchange through proxies. The payment (surplus division) scheme is well specified and has led to highly efficient outcomes in preliminary testing. The linear price-feedback mechanism is also well specified but has not been implemented and tested. The exact structure of the activity rules and proxy agents is still an open question.

1 Introduction

In this document we present the beginnings of a design for a complete combinatorial exchange. A combinatorial exchange combines and generalizes two different mechanisms: double auctions and combinatorial auctions. In a double auction, multiple buyers and sellers are brought together so that the sellers' goods can be appropriately allocated to the buyers [17]. We assume that each seller has a single item up for sale. Similarly, we assume that each buyer only desires one item.¹ Trades and payments must be determined so as to increase the total value. In a combinatorial auction, a single seller has multiple heterogeneous items up for sale [5, 24]. Here buyers may have complementarities or substitutabilities between goods (i.e. a combination of goods may be worth more or less than the sum of its parts). Thus they are allowed to bid not only on single items but also on bundles of goods. Again, the goal is to determine an allocation and payments so as to increase total value.

A combinatorial exchange is a combinatorial double auction, that brings together multiple buyers and sellers to trade multiple heterogeneous goods. Participants, which we will refer to as agents, are able to specify values for trades. For example, in an exchange for wireless spectrum, an agent may declare that she is willing to pay $1 million for a trade where she obtains licenses for New York City, Boston, and Philadelphia, and loses her licence for Washington DC. The exchange may have the ability to define which combinations of goods are valid to trade.

There are three main challenges in designing effective combinatorial exchanges: winner determination, preference elicitation, and incentives. The winner determination problem consists of finding the optimal allocation, once the agents' values for the possible trades are known. This problem is NP-hard by reduction from weighted set-packing, as is the case in combinatorial auctions. In fact,

¹If this is not the case, we can simply construe a single seller as multiple sellers, one for each of her goods. Also, we can assume that buyers' values for goods are not interdependent, so that the value of a bundle is simply the sum of the values of its underlying goods.
winner determination in combinatorial exchanges will likely be harder than in one-sided auctions. Nevertheless, methods from the operations research literature show good promise of tackling this issue effectively in practice.

The preference elicitation problem consists of communicating enough information about values from the agents to the exchange so that an optimal allocation can be computed. Iterative designs can address this challenge by guiding the elicitation process and informing agents of tentative values through prices.

The incentive-compatibility problem arises because agents cannot be forced to reveal their true values. This leads to a central challenge in exchange design known as the bargaining problem. Consider the following scenario, drawn from Myerson [20]. A seller has two goods, A and B, which she values together at 100, but each individually at 0. There are two buyers. The first, buyer 1, only values good A, but this exact value is not known to the exchange. All that is known is that it is either 90 or 30. Meanwhile it is known that buyer 2 only values good B, at 90. Since the total increase in surplus from allocating A to 1 and B to 2 is either 80 or 20, it is clear that this trade should take place.

Now comes the issue of dividing the surplus. Suppose the exchange divides the surplus equally among buyers, and that buyer 1 happens to have a value of 90 for A. If buyer 1 claims truthfully to have value 90, the surplus is 80 so each buyer pays 50 and obtains a surplus of 40. But if buyer 1 claims to have value 30, she only pays 20 and gets 70 of the total surplus. Thus buyer 1 has an incentive to lie, and in this case the exchange does not meet its goal of dividing the surplus equally.

In general the issues of allocation and payments cannot be decoupled in an exchange, because the payment scheme will affect agents' responses, which impact the final allocation. So although the correct allocation would be clear if all true values were known to the exchange, the payment scheme will influence the quality of allocation when values are private.

1.1 Design Overview

The exchange we envision proceeds in two stages: an elicitation stage, and a final allocation stage. The elicitation stage proceeds in rounds, with buyers and sellers making bids and asks on bundles. A bid (ask) is a price offer to buy (sell) a specific good. At each round a provisional allocation is computed, and prices over the items are quoted to give agents feedback as to how they should modify their bids and asks. Eventually this stage halts, and the final allocation and payments are determined, with payments computed so as to mitigate the bargaining problem and fairly divide the surplus. The payments do not necessarily correspond to a price for each individual good. This two-stage design is similar in spirit to Ausubel and Milgrom's clock-proxy auction [3].

Agents communicate their bids and asks through proxies. The proxies maintain upper and lower bounds on agents' values for different bundles, and make bids and asks on the agents' behalf with this partial information. The agents can refine these upper and lower bounds when they find this necessary, given the price feedback. The proxies may possibly guide the elicitation process by requesting specific information about agents' values. The proxies also enforce activity rules that specify the rate at which information must be revealed, to ensure that the exchange always makes good progress.

We describe the two stages in reverse: section 3 focuses on the final allocation and the bargaining problem, while section 4 describes the elicitation stage and related design questions. Section 2 provides necessary background.
2 Preliminaries

An ideal combinatorial exchange would have the following three properties:

1. **Budget-balance** (BB). The total payments received by the exchange from agents should be at least the total payments made by the exchange to agents.

2. **Individual-rationality** (IR). No agent should pay more than its net increase in value for the items it trades.

3. **Allocative-efficiency** (EFF). Trade should be executed to maximize the total increase in value over all agents.

In short, (BB) means the exchange does not run at a loss, and (IR) means that agents can only benefit from participating in the exchange. This benefit is measured with respect to their *reported* values over items; agents may in fact leave the exchange worse off if they try to game the system and poorly misrepresent their values. The (EFF) criterion means that items are reallocated to agents that value them most, hence maximizing the total increase in value, or *surplus*. This criterion specifies nothing about the division of this surplus among agents. In fact, this is a core design issue. The surplus-division scheme will impact the extent to which agents will try to game the system, because all agents will wish to extract as much of the surplus as possible for themselves. The exchange divides the surplus by requiring payments from buyers and providing payments to sellers, always ensuring that (BB) and (IR) are met.

As an example, suppose we have three agents A, B, and C. A owns an item that it values at 4. B and C value this item at 10 and 7 respectively. Giving the item to C yields a surplus of 3, while giving it to B yields a surplus of 6, so the latter is the efficient trade. By giving the item to C, the exchange already satisfies (EFF). Now the exchange will give payment $p_A$ to A and receive payment $p_B$ from B (C can be ignored from now on). To satisfy (IR), we must have $p_A \geq 4$ and $p_B \leq 10$. To satisfy (BB) we must have $p_A \leq p_B$. So the prices must satisfy $4 \leq p_A \leq p_B < 10$. Suppose we set $p_A = 5$ and $p_B = 8$. Then out of a surplus of 6, 1 goes to A, 2 goes to B, and 3 goes to the exchange.

In general care should be taken to ensure that the surplus is divided equitably, or at least that any apparent biases in the surplus-division can be justified. The concept of equity is subjective and so is not usually present in mathematical models, but an inequitable exchange design would probably have little chance of actually being implemented. The risks of litigation would be too great for a player like the FCC, for example.

Another convenient property would be incentive-compatibility (IC). Intuitively, this means that it is an optimal strategy for each agent to be truthful about its values for items. We distinguish between (IC) in *dominant strategies*, where is is optimal for an agent to be truthful no matter what actions the others take, and *Bayes-Nash* (IC), where it is optimal for an agent to be truthful given the actions it expects others to take (which the agent deduces from its beliefs over the other agents’ values for items). Incentive-compatibility in dominant strategies is also called *strategy-proofness*. (IC) is desirable for an exchange because it ensures agents will not try to game the system, and an allocation can then be computed that enhances the true surplus, instead of a surplus with respect to false values. Thus an exchange that is hard to manipulate will usually lead to more efficient allocations.

Myerson and Satterthwaite [21] show that there is no exchange that can have all three properties of (BB), (IR), and (EFF), even putting aside (IC). This means that for any exchange design, there is always some possible scenario of agents and values where the exchange will fail on one of
these properties. It does not mean that the exchange will always fail on some property, as our example shows above. Still, one of these properties must be abandoned in any design. Parkes et al. [22] (henceforth PKE) take a constructive approach and design clearing and payment rules so as to maximize surplus subject to the constraints implied by (BB) and (IR). Both (BB) and (IR) are essential because no reasonable exchange can run at a loss, and agents cannot be forced to participate.

The (IC) property can be obtained together with (BB) and (IR). However, to achieve (IC) it may be necessary to implement an allocation that is deliberately inefficient with respect to (truthful) reported values. This might make the exchange appear flawed, which could be politically undesirable (more litigation). Thus PKE propose implementing the surplus-maximizing allocation (subject to (BB) and (IR)), and dividing the surplus so as to mitigate manipulation opportunities.

2.1 The VCG Mechanism

The type of payment scheme we will implement is inspired by the Vickrey-Clarke-Groves (VCG) mechanism. Let \( V^* \) be the maximum surplus that can be obtained in the exchange (ignoring (BB) and (IR) constraints), and let \( V_{-i}^* \) be the maximum surplus that can be obtained when agent \( i \) is not present. Let \( \hat{v}_i^* \) be agent \( i \)'s reported value for the proposed surplus-maximizing allocation. The VCG mechanism sets the payment to agent \( i \) to \( p_{\text{vick},i} = \hat{v}_i^* - (V^* - V_{-i}^*) \). The VCG mechanism has the following properties:

**Proposition 1** The VCG mechanism is efficient and strategy-proof. The VCG mechanism is individual-rational, such that the expected utility to rational agents from participation is also non-negative.

Note that a VCG mechanism is not necessarily budget-balanced, so it is not appropriate for our purposes. Nonetheless because it is strategy-proof, it does provide insight into the design of mechanisms that are hard to manipulate. Note that the surplus allocated to agent \( i \) in a VCG mechanism is \( (V^* - V_{-i}^*) \), which is always non-negative.

**Definition 1** The Vickrey discount to agent \( i \) is \( \Delta_{\text{vick},i} = V^* - V_{-i}^* \).

The Vickrey discount can be interpreted as the increase in total possible surplus brought about by the agent’s participation in the mechanism. In a VCG mechanism, an agent receives as surplus exactly the surplus it generates by participating in the mechanism. This effectively removes any incentive for the agent to misrepresent its values.

3 Allocation Stage and Payments

In this section we describe a one-shot design for an exchange that focuses on mitigating the bargaining problem, leaving aside the issue of the computational complexity of winner determination. This one-shot design can be repeated in rounds to yield an iterative exchange. The general design is as follows:

- Collect bids
- Compute \( V^* \), the value of the surplus-maximizing trade given all bids.
- Implement this surplus-maximizing trade.
• Compute $V^*_{-i}$, the value of the surplus-maximizing trade without bids from agent $i$.

• Divide the surplus $\sum_i \pi_i = V^*$ among the participants so as to mitigate the bargaining problem.

Here $\pi_i$ is the part of the surplus allocated to agent $i$. The key component in this design is the surplus-division rule.

### 3.1 The Threshold Rule

Ideally every agent in an exchange could be given its Vickrey discount, and this would remove any gaming incentives. However, we have seen that this would not satisfy (BB) in some cases. Let $\Delta_{\text{vick}} = (\Delta_{\text{vick},1}, \ldots, \Delta_{\text{vick},m})$ be the vector Vickrey discounts corresponding to the $m$ agents in the exchange, and let $\Delta = (\Delta_1, \ldots, \Delta_m)$ be the actual discounts allocated to the agents. We would like the latter to satisfy (BB) and (IR), yet deviate as little as possible from the Vickrey discounts in order to mitigate manipulation opportunities. We call a mechanism with such goals a VCG-based mechanism. Thus we solve the following linear program:

\[
\begin{align*}
\min_{\Delta} & \quad L(\Delta, \Delta_{\text{vick}}) \\
\text{s.t.} & \quad \sum_{i \in I^*} \Delta_i \leq V^* \\
& \quad \Delta_i \leq \Delta_{\text{vick},i} \quad \text{for all } i \in I^* \\
& \quad \Delta_i \geq 0 \quad \text{for all } i \in I^* 
\end{align*}
\]

where $V^*$ denotes the available surplus when the exchange clears, and $I^*$ denotes the set of agents that trade. Here $L$ denotes a suitable distance function between the distance vectors. PKE propose to adapt

\[
L_{\infty}(\Delta, \Delta_{\text{vick}}) = \max_i |\Delta_i - \Delta_{\text{vick},i}|
\]

which is the maximum deviation on any component. Rather than solving the LP explicitly, the solution to this problem is characterized by a threshold rule $\Delta^*_i = \max(0, \Delta_{\text{vick},i} - C)$, with $C$ chosen such that (BB) still holds. We call this a threshold rule because only those agents whose Vickrey discounts are greater than $C$ will receive any discount, or surplus. With such a rule, the agents with highest Vickrey discounts receive the highest actual discounts (so this rule is arguably fair, because recall that an agent’s Vickrey discount is effectively the surplus that it brings to the exchange). Note also that if $C = 0$, the rule reduces to the VCG payment scheme, because the Vickrey discount is never negative.

The degree of possible manipulation is in some sense limited for any agent when using the threshold rule. An agent will have an incentive to manipulate when it finds that truthful bidding does not extract the maximum possible surplus from the mechanism. This maximum surplus is in fact the Vickrey discount. Intuitively then, when an agent receives its Vickrey discount, it has no incentive to manipulate. The threshold $C$ is chosen to be minimal, keeping actual discounts as close as possible to Vickrey discounts, and hence keeping low the possible gains from manipulation. See appendix A for a rigorous presentation of these arguments.

Although the threshold rule mitigates manipulation, any degree of manipulation is of concern in game-theoretic analysis. There is the risk that agents will not only manipulate, but also attempt to anticipate other agents’ possible manipulation schemes. In these cases it can be very hard to ensure a reasonable equilibrium outcome. To alleviate these concerns one could also analyse the
difficulty of manipulating the mechanism. Manipulation could very well be NP-hard in an exchange. Showing this would at least provide the assurance that there is (probably) no efficient, systematic way of gaming the system. NP-hardness analyses have been performed for the manipulation of voting schemes and other mechanisms \[\].

As mentioned, this one-shot design can be iterated. At each round, the allocation and payments are computed as with the one-shot exchange, but they remain provisional. Agents can then modify their bids and asks in response if they are not satisfied with the allocation. However, the discounts may not give the agents precise information as to how to modify their bids and asks. Discounts only apply to specific bundles. Linear prices (where each item is priced individually), are usually more intuitive to agents \[8\], so they are more appropriate as price-feedback. This question and related issues are addressed in the elicitation stage.

4 Elicitation Stage

In this section we present the elicitation component of the exchange design. The ingredients of this component are a proxy agent, a price feedback mechanism, and activity rules. Each agent will have a proxy agent that sets bids and asks on its behalf. The proxy agents maintain partial information about the agents’ valuations in the form of upper and lower bounds on values for various items. The central exchange mechanism communicates with proxies for bids and asks; these proxies then communicate with their respective agents to refine their information if necessary. At each round a provisional allocation and payments are computed, and price-feedback information is fed to the proxies so that they may decide which bids or asks to put forth next. Activity rules require that proxies (and hence agents) continuously refine the upper and lower bounds on valuations so that the exchange may move forward at a reasonable pace.

4.1 Proxies

The purpose of proxy agents is to facilitate and expedite communication between the central exchange mechanism and agents \[23\]. A proxy agent maintains partial information on the valuations of its respective agent. This can be achieved simply by having the agent reveal some information of its choosing to the proxy. The proxy may also guide the revelation process by making specific queries on the agent’s values. On the exchange side, a proxy sets bids and asks on behalf of the agent.

A typical example is the proxy bidder from eBay.com \[\]. Here bidders provide a lower bound on their value for a single item to the proxy. The proxy however views this value as an upper bound, and bids on the item on behalf of the bidder until the item is either won, or the upper bound is reached (the bids are set by adding a small increment to the current price). If the latter occurs, the actual bidder is informed that her lower bound is insufficient to win. She may then increase this lower bound, if it does not already match her true value for the item.

Our proxies will maintain upper and lower bounds on the agents' values for each item or bundle of items. For a buyer, a lower bound \( \underline{b}(S) \) on a bundle of items \( S \) indicates that the buyer would necessarily be willing to buy \( S \) if its price were under \( \underline{b}(S) \). An upper bound \( \overline{b}(S) \) indicates that the buyer would not be willing to buy the item for any price above \( \overline{b}(S) \). Similarly, a lower bound \( \underline{g}(S) \) for a seller means the seller would not be willing to sell \( S \) for any price under \( \underline{g}(S) \), and an upper bound \( \overline{g}(S) \) means the seller would always be willing to sell for a price above \( \overline{g}(S) \). In short, we always have \( v_i(S) \in [\underline{b}(S), \overline{b}(S)] \) for every agent \( i \) and bundle \( S \) for which upper and lower bounds have been specified.
Currently, we envision passive proxies that simply wait for refinements on upper and lower bounds for bundles. We might also implement more active proxies that query agents for specific information. Such queries may include, for example:

1. **Value queries**: The proxy presents a bundle $S$ and the agent responds with its value $v(S)$ for this bundle.

2. **Demand queries**: The proxy indicates what bundles the agent will trade, and the prices that it will pay or receive for the trade. The agent then either accepts this trade, or indicates a trade that it would prefer at the quoted prices.

3. **Ordinal queries**: The proxy presents two bundles $S_1$ and $S_2$ and the agent indicates which bundle it prefers, if any.

These queries are just a sample of the type of partial information that proxies could actively elicit from agents. Designing proxies that judiciously make queries in order to efficiently elicit the required information is a non-trivial problem; one way to approach it might be through learning theory methods [12].

As currently specified our bidding language only allows agents to reveal values over bundles, not actual trades. It might be the case though that some agents will be both buyers and sellers, and will only be interested in selling if they can obtain other goods in return (to maintain some kind of market presence, for example). Thus an agent might want to specify that she is willing to sell bundle AB in return for item C plus $10. This would be convenient for the agents, but it might make the exchange significantly harder to clear. Also, the price-feedback mechanism described above is not adapted to this generalized bidding language. It is still an open question whether to allow this type of combined bid and ask.

### 4.2 Activity Rules

Activity rules ensure both consistency and progress throughout rounds [33]. In the simple upper and lower bound scheme described above, consistency means that the bounds on a given bundle $S$ must tighten as the auction progresses. Letting $[\hat{b}^i_t(S), \tilde{b}^i_t(S)]$ be the bounds for agent $i$ on bundle $S$ at round $t$, we must then have $\hat{b}^i_t(S) \leq \hat{b}^i_{t+1}(S)$ and $\tilde{b}^i_t(S) \geq \tilde{b}^i_{t+1}(S)$. We will also usually assume that the agents’ valuations satisfy the free-disposal assumption, namely that $v_i(S) \leq v_i(S')$ for agent $i$ when $S \subseteq S'$. In this case consistency would also require that $\hat{b}_i(S) \leq \hat{b}_i(S')$ and $\tilde{b}_i(S) \leq \tilde{b}_i(S')$ for bundles $S \subseteq S'$. If any further assumptions are made on agent valuations, they might imply more consistency rules, so these should be checked.

Activity rules need to ensure progress along two dimensions: the slack between the bounds on bundles must tighten, and the number of additional bundles that can be introduced must be reduced in later stages. How to measure the total slack between bounds is still an open design question. The total slack over all bundles could be computed, or perhaps some weighted measure of slack would be more appropriate. Other design questions include: Should slack always be reduced on all bundles, or only on a subset? If the latter, how should this subset be chosen? By what factor must slack be reduced? Will this factor vary as the exchange progresses?

An agent will naturally not provide bounds on every possible bundle; we assume that it is uninterested in bundles for which it provides no bounds. However, an agent is allowed to reveal interest in certain bundles throughout the exchange by providing bounds. The number of bundles that can be introduced in this way must diminish as rounds progress. Currently we have no fixed
function in mind for this, but this is an important design issue because it will impact both the speed of the exchange and the behavior of the agents throughout time.

At some point, a final stage will be reached. This halting stage must occur late enough that enough information has been gathered from agents to determine a reasonably efficient allocation, but not so late as to make the exchange too lengthy. The conditions that characterize the final stage are not currently specified and remain an open design question. An auction typically ends when a “fixed-point” has been reached where all agents are satisfied with their allocation and do not wish to bid any further. In our case we might choose to end the exchange when the total slack in agent’s values has dipped below some threshold. It should be clear to agents in advance when the final stage is expected to occur.

4.3 Price-Feedback

At each stage, a “high” and “low” outcome will be computed. The “high” outcome is the efficient allocation when taking the buyers’ upper bounds on their values as their bids, and the sellers’ lower bounds on their values as their asks. Prices that support (or approximately support) this allocation can provide useful feedback in the early stages. Conversely, the “low” outcome is the efficient allocation when taking the buyers’ lower bounds as their bids, and the sellers’ upper bounds as their asks. Prices that support this low outcome will provide more useful feedback in the later stages. Ultimately, the low outcome at the final round is implemented.

We will only consider linear prices, in the sense that a price is quoted for each item, and the price of a bundle is the sum of the prices of its underlying items; as opposed to nonlinear prices where a price may be quoted for each bundle. Providing a price for each bundle of interest may be infeasible due to the exponential number of bundles. Also, linear prices are much simpler for agents to interpret [8].

At each stage there will be four separate sets of anonymous prices, for the buy and sell-sides of the high and low outcomes. Prices are anonymous if all agents face the same set of prices; that is, an agent will never be required to pay more than another agent for the same set of items. However, the sell-price and buy-price of an item may differ, within each different outcome.

On the buy-side of the high outcome, prices are computed according to the following program. Let $\overline{p}_{bid,j}$ be the price for item $j$ in this scenario. Let $S^i$ be the bundle allocated to buyer $i$. If a buyer is allocated a nonempty bundle, we call her a “winner”, otherwise a “loser”.

$$\min_{\overline{p}_{bid}, \delta \geq 0} \delta$$

s.t. 

$$\overline{g}_i(S^i) - \sum_{j \in S^i} \overline{p}_{bid,j} \geq 0$$

for each winner $i$ \hspace{1cm} (6)

$$\overline{h}_i(S) - \sum_{j \in S} \overline{p}_{bid,j} \leq \overline{h}_i(S^i) - \sum_{j \in S^i} \overline{p}_{bid,j} + \delta$$

for each winner $i$, bundle $S$ \hspace{1cm} (7)

$$\overline{g}_i(S) - \sum_{j \in S} \overline{p}_{bid,j} \leq \delta$$

for each loser $i$, bundle $S$ \hspace{1cm} (8)

To better understand the above, note that

$$\overline{g}_i(S) - \sum_{j \in S} \overline{p}_{bid,j}$$

represents the surplus to buyer $i$ if she obtains bundle $S$ at the high-outcome buy-side prices. First suppose that $\delta$ is 0 in the program above. Then constraint (6) ensures that each winner obtains a bundle that gives her non-negative surplus. Constraint (7) ensures further that this surplus is maximal among all possible bundles. Constraint (8) ensures that no bundle could provide a loser
with positive surplus. We thus have an instance of a *competitive equilibrium* with linear prices over the items. In most cases there will be no linear prices that can achieve a \( \delta \) of 0, so we relax the program above and minimize \( \delta \) so as to find the best possible linear prices, in some sense.

The resulting prices provide feedback that can help winners tighten their upper bounds on bundles. By constraint (6) each winner can lower her upper bound to the price of her won bundle, and still keep it. Similarly, the buy-side prices of the low outcome can provide feedback to losers as to how to increase their lower bounds. The corresponding program is:

\[
\begin{align*}
\min_{\mathbf{p}_{\text{bid}}, \delta \geq 0} & \quad \delta \\
\text{s.t.} & \quad b_i(S) - \sum_{j \in S} p_{\text{bid},j} \leq 0 \quad \text{for each loser } i, \text{ bundle } S \quad (9) \\
& \quad b_i(S^i) - \sum_{j \in S^i} p_{\text{bid},j} \geq \delta \quad \text{for each winner } i \quad (10) \\
& \quad b_i(S) - \sum_{j \in S} p_{\text{bid},j} \leq b_i(S^i) - \sum_{j \in S^i} p_{\text{bid},j} + \delta \quad \text{for each winner } i, \text{ bundle } S \quad (11)
\end{align*}
\]

The constraints here have the same interpretation as before. However, constraint (9) now suggests by how much a buyer must raise its lower bound on a bundle to have a chance of winning it (i.e., to lower bound must be raised to at least the bundle’s price).

The sell-side pricing scheme is entirely symmetric. Linear prices computed for the high outcome suggest to the winning sellers (those that actually sell some of their items) by how much they can increase their ask price lower bounds and still sell their items. Note that the sellers’ lower bounds are used to compute the high outcome. The program is:

\[
\begin{align*}
\min_{\mathbf{p}_{\text{ask}}, \delta \geq 0} & \quad \delta \\
\text{s.t.} & \quad \sum_{j \in S^i} p_{\text{ask},j} - b_i(S^i) \geq 0 \quad \text{for each winner } i \quad (12) \\
& \quad \sum_{j \in S} p_{\text{ask},j} - b_i(S) \leq \sum_{j \in S^i} p_{\text{ask},j} - b_i(S^i) + \delta \quad \text{for each winner } i, \text{ bundle } S \quad (13) \\
& \quad \sum_{j \in S} p_{\text{ask},j} - b_i(S) \leq \delta \quad \text{for each loser } i, \text{ bundle } S \quad (14)
\end{align*}
\]

To understand why these prices are informative specifically to winning sellers, note that constraint (8) does not depend on \( \delta \). Therefore winning sellers can know how to make this constraint tight (i.e. hold with equality) simply with knowledge of the prices, not of \( \delta \).

Finally, linear prices for the low outcome suggest to the losing sellers by how much they should decrease their upper bounds to sell something, by the same principle.

\[
\begin{align*}
\min_{\mathbf{p}_{\text{ask}}, \delta \geq 0} & \quad \delta \\
\text{s.t.} & \quad \sum_{j \in S} \mathbf{p}_{\text{ask},j} - \bar{b}_i(S) \leq 0 \quad \text{for each loser } i, \text{ bundle } S \quad (15) \\
& \quad \sum_{j \in S^i} \mathbf{p}_{\text{ask},j} - \bar{b}_i(S^i) \geq \delta \quad \text{for each winner } i \quad (16) \\
& \quad \sum_{j \in S} \mathbf{p}_{\text{ask},j} - \bar{b}_i(S) \leq \sum_{j \in S^i} \mathbf{p}_{\text{ask},j} - \bar{b}_i(S^i) + \delta \quad \text{for each winner } i, \text{ bundle } S \quad (17)
\end{align*}
\]

As noted above, the low outcome is ultimately implemented. What incentives do buyers and sellers then have to refine their upper and lower bounds respectively, namely those bounds that are relevant to the high outcome? Recall that the activity rule will require that slack be reduced throughout the rounds. Agents may find it more convenient to satisfy this rule by adjusting either upper and lower bounds, (reducing the run-time from days to hours)
4.4 Key Problems

There are three key elicitation problems that must be addressed by the exchange. As explained above, the bargaining problem is omnipresent and must be dealt with by appropriate distribution of surplus. There is also the problem of price discovery: there may be no trade in the initial stages, and as a result no prices are set. Some means must be devised to start the price discovery process, perhaps through tentative initial prices.

Finally, there is the problem of item discovery. The scope of the exchange may not be initially known: sellers might not know which items are specifically demanded by buyers, and buyers may not be conscious of all items that are up for sale. To aid in item discovery, tentative prices can be posted to signal the presence of items to buyers and sellers.

On the buy-side prices are needed for items offered on the sell-side. One possible price signal might be $0.5 \left( p_{\text{ask}, j} + (1 - s) v(S) \right)$. For sell-side items on the buy-side, a possible signal might be $0.5 \left( (1 + s) v(S) + p_{\text{bid}, j} \right)$. Finding appropriate price signals is still an open design question.

5 Strategic Analysis of One-Shot Exchange

It is very difficult to analytically derive the Nash equilibria that might be played in a mechanism like the one in section 3. Therefore, we turn to computer experiments to find potential equilibria. There is a substantial literature on the complexity of finding Nash equilibria in normal-form games. For two-person normal-form games, the Lemke-Howson algorithm [15] efficiently computes mixed Nash equilibria in practice. For more general $n$-person games, simplicial subdivision algorithms like that of Scarf [30] are typically used. For overviews of algorithms for computing equilibria, see McKelvey et al. [18].

Here we present a heuristic method developed by Krych [10] precisely for the purpose of finding equilibria in the one-shot exchange mechanism described in section 3. This method searches for \emph{ex ante} Bayes-Nash equilibria, that is, equilibria where agents play their optimal strategy given their beliefs about the other agents' values and potential actions. This is as opposed to \emph{ex post} Nash equilibria, in which each agent is playing her optimal strategy given exact knowledge of the other players' actions.

To keep the search space tractable, the method assumes a linear strategy space where agents change their bids or asks by a constant fraction of the true value. Thus a strategy for a buyer consists of choosing some $s \in [-1, 1]$ and bidding $(1 - s) v(S)$ for bundle $S$ when $v(S)$ is the agent's true value. Similarly a seller will ask $(1 + s) v(S)$ for a bundle $S$ with true value $v(S)$. The players' valuations are assumed to all be drawn from the same distribution. As a result the Bayes-Nash equilibria will be symmetric, and all players will choose the same optimal $s$ (as both a decrease in their bids and an increase in their asks).

Before we examine the algorithm, consider a naive approach that discretizes the strategy-space and exhaustively searches through all possible strategy combinations to find Bayes-Nash equilibria. We discretize the strategy space $[-1, 1]$ to increments of 0.01. With 5 buyers, 5 sellers, 20 items, and 10 bids/asks per agent, exhaustive search requires 2.5 days.

The algorithm is as follows:

1. Choose a set of strategies $S$ on which to search from the range $[-1, 1]$. This can be done by discretizing the space to some level of granularity, for example, or simply randomly choosing strategies from this range. Choose an $s_0$ randomly from $S$. 

10
2. Generate \( m \) valuation instances based on the assumed initial distribution of values, and randomly choose an agent \( a \) to follow.

3. Select a set \( S' \) of possible alternate strategies for \( a \). The sets \( S \) and \( S' \) may or may not be closely related.

4. Fix the strategies of all agents other than \( a \) to \( s_0 \). Find the strategy \( s^* \) in \( S' \) that maximizes \( a \)'s surplus when the one-shot exchange is run. (Note this step is computationally intensive, involving \(|S'| \) runs of the exchange, including winner and payment determination).

5. Set \( s_0 \leftarrow f(s_0, s^*) \), where the function \( f \) typically moves \( s_0 \) towards the best-response \( s^* \). Repeat steps 2 to 5 until convergence.

The free parameters in this algorithm are the choice of \( S \) and \( S' \), and the choice of update function \( f \). Krych chose to set \( S \) as \([0, 1] \), and \( S' \) is chosen to be a discretization of the set \( \{ s : |s_0 - s| \leq d \} \), where the parameter \( d \) is updated at each iteration. The parameters \( s_0 \) and \( d \) are updated as follows:

\[
s_0 \leftarrow \frac{(s^* + 2s_0)}{3}
\]

and

\[
d \leftarrow \begin{cases} 
4|s_0 - s^*| & \text{if } s_0 \neq s^* \\
3d/4 & \text{otherwise}
\end{cases}
\]

This algorithm is not guaranteed to converge in practice, and if it does the outcome is not guaranteed to be a global best-response. Thus when the algorithm terminates, it should be checked that the final \( s_0 \) is indeed a best-response to itself. In practice, this algorithm runs much faster than exhaustive search, and has always converged to a global best-response. The run-time for 5 buyers, 5 sellers, 20 goods, and 10 bundles per agent is 2.5 days in order to achieve 1% accuracy. The winner determination step is performed by commercial mixed integer programming (MIP) software [2, 7, 9].

When testing the various exchange rules with these simulations, we find that the Threshold rule is the most efficient \( ex \ post \) individual-rational payment scheme. So the experimental results agree with the theoretical findings mentioned in section 3.1. Since these initial experiments are limited in the type of strategies they consider, there is still much testing that could be done, if more efficient equilibrium-solving algorithms are developed.

The strategies considered above apply to a one-shot mechanism. In an iterative exchange there is the added dimension of time, and agents could attempt to manipulate the outcome by timing their bids and asks, perhaps by sniping (where an agent only bids in the very last rounds). In general, an agent’s strategy will specify how values are reported (with possible mark ups or mark downs), and how these values are revealed throughout the rounds. These parameters need not be fixed beforehand; they may be adapted as the agents observe the impact of their strategies on provisional prices and allocations.

We distinguish between heuristic and equilibrium analyses in our testing. A heuristic analysis would simply involve running the exchange with the artificial bidding strategies to examine the latter’s performance over multiple trials. In an equilibrium analysis like the one described in this section, we assume that agents have beliefs about each others’ values and will therefore play some kind of equilibrium strategy based on these beliefs. This is much harder since it involves identifying equilibria, and even then it may not be clear which equilibria agents would typically play. This approach also requires a reasonable model of agent beliefs, and assumes that agents are sophisticated enough themselves to compute possible equilibria.
We will perform heuristic analyses on our final exchange implementation. Rothkopf [28, 27, 29] presents simple strategies that can be used as inspiration for early bidding agents. For more sophisticated agents, see for example the literature out of the Trading Agents Competition (TAC) [32] and work by Csisrik [4], Wellman [31] and Wurman [34]. Also, it would be useful to study strategies that have been observed in actual auctions for spectrum [1].

6 Domain Modeling

In this section we discuss the issues related with modeling two possible application domains for our exchange: the FCC’s market for wireless spectrum and the FAA’s market for airport take-off and landing slots.

6.1 Wireless Spectrum

The FCC is interested in reallocating wide bands of wireless spectrum to increase economic efficiency [11, 6, 19]. Currently the spectrum is inefficiently allocated among a number of different incumbents, ranging from television stations to the DoD. Furthermore, the spectrum is defined as a set of technical specifications on transmitters, i.e. it is defined in terms of “inputs”. This makes trading spectrum very difficult, because firms will not necessarily have the same inputs, although many different inputs could really be allowed on a certain band of spectrum. To make the market more flexible, the FCC needs to redefine the spectrum in terms of “outputs”, such as power limits and geographic areas.

An exchange could therefore expedite the transition to a flexible market for spectrum. The exchange would ideally meet three goals: 1) efficiently reallocating encumbered spectrum, 2) efficiently allocating unencumbered spectrum currently held by the FCC, and 3) redefining spectrum licenses in terms of outputs rather than inputs, to create a flexible after-market. To achieve this last goal, the exchange would probably have to present different specifications for goods on the buy and sell-sides (buy-side spectrum is defined by its inputs, sell-side spectrum by its outputs). This redefinition might involve aggregating small blocks of spectrum into larger blocks.

Thus a key challenge here is clearly modeling the structure of the items and bundles in this domain. A further challenge is modeling values over these bundles that participants might exhibit, and modeling distributions over possible values. Finally, the model will need to take into account the fact that the FCC is a special player in this exchange, and reliably depict the FCC’s preferences. As both the exchange manager and a participant, the FCC cannot be seen as acting strategically to increase its revenues, for risk of litigation.

6.2 Landing Slots

The market for take-off and landing slots represents another scenario in which a combinatorial exchange could significantly increase the efficiency of allocations. Allocating airport time slots provided motivation for some of the first combinatorial auctions [25]. Currently there is in fact no market at all for these slots; except for the case of four high-traffic US airports, landing slots are simply provided following a first-come first-served basis [13]. This is very inefficient because airports naturally become crowded during peak periods and underused at other times. Additional inefficiency is introduced by the fact that small aircraft must wait longer to take-off after large aircraft than vice-versa (because large aircraft leave a “wake”). Therefore the sequencing of planes impacts an allocation’s efficiency.
The challenge here is to again clearly the define the items and bundles in play. The problem has a combinatorial nature since airlines will be interested in acquiring take-off and landing slots together. There will also be complementarities between pairs of slots because airlines will want to coordinate their connecting flights. We currently have take-off and landing data for Hartsfield International Airport in Atlanta. Perhaps some information over possible values or distributions over values can be extrapolated from this data.

7 Future Work

The ultimate goal is to put together all components of the exchange and run simulations with artificial agents to test performance. The initial tests would involve simple bidding agents, and more sophisticated bidding agents can be introduced as the exchange is refined [14, 16]. The testing must ensure that the exchange scales well in the number of agents and items involved. It might also be worthwhile to test the exchange’s performance when actual people are involved, in an experimental economics labs [26].

8 Conclusion

We have described the preliminary design of an iterative combinatorial exchange. This design is based on a one-shot exchange that maximizes efficiency subject to the constraints of budget-balance and individual-rationality. This one-shot exchange determines payments according to a threshold rule, which attempts to keep actual payments as close as possible to Vickrey payments in order to minimize any incentives to manipulate. This design has been tested against simple strategies where agents mark up or down all reported values by a chosen factor. It achieves good efficiency against such strategies.

The one-shot design is iterated, with a provisional allocation and payments computed at each round. Linear prices are also computed at each round to provide agents with feedback. This feedback guides the agents as to how they should refine the upper and lower bounds on their values for bundles. The effectiveness of this price-feedback scheme still needs to be tested.

A few significant open design problems remain. We need activity rules to ensure consistency and progress as rounds proceed. The slack between bounds on values must tighten at a reasonable rate, and the number of new values that can be revealed in later rounds must be restricted. The proxy agents that enforce these activity rules and guide the elicitation process must also be designed.

Finally, we need good models for target domains like the markets for wireless spectrum and airport landing slots. Given these models, we need to develop both naïve and sophisticated artificial bidding strategies to stress-test the final design.

References


### A. Degree of Manipulation

In this appendix we present some formal results on the degree of possible manipulation in VCG-based schemes. Recall that the idea in VCG-based mechanisms is to stay as close as possible to Vickrey discounts in order to mitigate manipulation. This motivates the following definition:

**Definition 2** Fix the bids and asks from other agents. The residual degree of manipulation freedom for agent $i$, $\text{RDMF}(i)$, is the maximal amount that the agent can increase its utility with some bid $\hat{v}_i \neq v_i$, in comparison to its utility from the outcome when it bids truthfully.

We have $\text{RDMF}(i) = 0$ in a VCG mechanism, because it is optimal to be truthful in such mechanisms. We have the following two important propositions from PKE:

**Proposition 2** The difference $\Delta_{\text{vick},i} - \Delta_i$ is an upper-bound on the RDMF($i$) in a VCG-based exchange mechanism.

**Proposition 3** No VCG-based exchange mechanism can achieve a RDMF($i$) better than $\Delta_{\text{vick},i} - \Delta_i$.

Since the threshold rule is optimal for distance function (5), it follows from Proposition 3 that this rule minimizes the maximal RMDF amongst all Vickrey-base (IR) and (BB) payment schemes. The RDMF with this rule is at most the value of the threshold parameter $C$ for any agent.