# Using Value Queries in Combinatorial Auctions

Benoît Hudson Tuomas Sandholm *
Carnegie Mellon University
Computer Science Department
5000 Forbes Avenue
Pittsburgh, PA 15213, USA
{bhudson,sandholm}@cs.cmu.edu

## ABSTRACT

Combinatorial auctions, where bidders can bid on bundles of items, are known to be desirable auction mechanisms for selling items that are complementary and/or substitutable. However, there are $2^k - 1$ bundles, and each agent may need to bid on all of them to fully express its preferences. We address this by showing how the auctioneer can recommend to the agents incrementally which bundles to bid on so that they need to only place a small fraction of all possible bids. These algorithms impose a great computational burden on the auctioneer; we show how to speed them up dramatically. We also present an optimal elicitor, which is intractable but may be the basis for future algorithms. Finally, we introduce the notion of a *universal revelation reducer*, demonstrate a randomized one, and prove that no deterministic one exists.

The full paper is available in draft form at `http://www.cs.cmu.edu/~sandholm/using_value_queries.pdf`

## Categories and Subject Descriptors

I.2.11 [**Computing Methodologies**]: Distributed Artificial Intelligence—*auctions*

## General Terms

Algorithms, Experimentation, Performance

## Keywords

Combinatorial auctions, electronic auctions, preference elicitation

## 1. INTRODUCTION

Combinatorial auctions, where agents can submit bids on *bundles* of items, are economically efficient mechanisms for selling $k$ items to $n$ bidders, and are attractive when the bidders' valuations on bundles exhibit *complementarity* (a bundle of items is worth more than the sum of its parts) and/or *substitutability* (a bundle is worth less than the sum of its parts). Unfortunately, there are $2^k - 1$ bundles, and each agent may need to bid on all of them to fully express its preferences. In addition, evaluating a bundle's value may require the bidder to solve a computationally expensive problem. Appropriate bidding languages can solve the communication overhead in some cases when the bidder's utility function is compressible. We focus on the case where agents' valuation functions are largely incompressible.

Our work uses a recently proposed scheme for *incremental preference elicitation* [1]. In this framework, the auctioneer elicits information from bidders about their valuation functions, using all replies to previous queries in order to guide further questioning (the auctioneer is thus also known as the *elicitor*). The elicitor stops eliciting when it has gathered enough information to clear the auction. This information acts as a certificate for the optimality of the final allocation.

We designed some algorithms within that scheme, limiting our focus here to value queries (asking a bidder for its valuation of a bundle). These map directly to the traditional notion of a bid in a sealed-bid auction. As is typical in work on combinatorial auctions, we assume the bidders have free disposal. This means that their valuation for a bundle is at least as high as their valuation for any subbundle (because at worst, they could throw away the extraneous items for free).

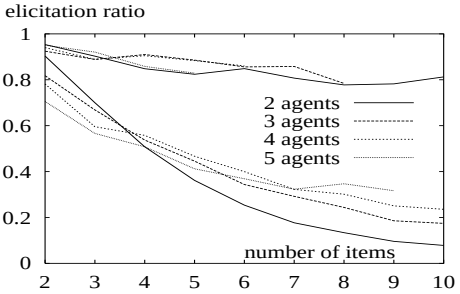


**Figure 1:** *Results of running our elicitors on synthetic data (detailed in the full-length version). Top set of lines: Random elicitation policy. Bottom set of lines: High-value policy. The* elicitation ratio *is the number of bundles elicited divided by the total number of bundles.*

## 2. RANDOM ELICITATION POLICY

The first policy we investigate simply asks random value queries. In the beginning, we generate the set of all $n(2^k - 1)$ value queries. Whenever it is time to ask a query, the policy chooses a random query from the set, ignoring those it has already asked or for which the value can already be inferred. If it is possible to save elicitation, then on average this policy saves elicitation:

PROPOSITION 1. *Let $Q = n(2^k - 1)$ be the total number of queries, and let $q_{min}$ be the number of queries in the shortest certificate. For any problem instance, the expected number of queries that the random elicitation policy asks is at most $\frac{q_{min}}{q_{min}+1}(Q+1)$.*

Figure 1 shows that the elicitation ratio $q/Q$ is indeed less than 1 in the experiments we ran. Also, the ratio slowly falls as the number of items increases. Nevertheless, we would hope to do better: on all the instance sizes we ran on, this algorithm asks more than half of all the queries.

## 3. HIGH-VALUE ELICITATION POLICY

We examined several other policies that take into account various reasonable heuristics for choosing better than randomly. The

*The material in this paper is based upon work supported by the National Science Foundation under CAREER Award IRI-9703122, Grant IIS-9800994, ITR IIS-0081246, and ITR IIS-0121678.


*EC'03*, June 9–12, 2003, San Diego, California, USA.
ACM 1-58113-679-X/03/0006.

best policy we have found so far – a modification of a policy recently used in a combinatorial exchange setting [6] – attempts to maximize its probability of reducing the upper bound of suboptimal allocations while simultaneously maximizing its possibility of increasing the lower bound of the optimal allocation. In particular, the elicitor finds the set of allocations which, given the information it has elicited so far, may have the highest value of any other allocation. This defines a set of bundle-agent pairs: a pair $(b, i)$ is in the set if one of the high-value allocations allocates bundle $b$ to agent $i$. We further refine the set of pairs to include only those that may reduce the upper bound on the value of the most other bundles in the set (given the free disposal assumption, this roughly corresponds to picking larger bundles).

**Representing allocations implicitly**

In the general algorithm framework [1], the set of allocations is explicitly maintained. However, the size of the set grows as $n^k$ where $n$ is the number of agents and $k$ is the number of items. When $n > 2$, the size of the set of allocations even exceeds the total number of bundles, $n2^k$. We therefore designed a method of implementing the policy described above, while avoiding the explicit representation in order to scale better in the number of agents.

We accomplish this by repeatedly solving an integer program (IP) every time a query is to be selected—rather than explicitly representing the set of allocations. We use the following IP to compute the value of the highest-valued allocation:

$$\begin{array}{lll} \text{maximize} & \sum_{i \in N, b \subseteq K} UB_i(b) x_i(b) & \\ \text{subject to} & x_i(b) \in \{0,1\} & \forall i \in N, \forall b \subseteq K \\ & \sum_{b \subseteq K} x_i(b) \leq 1 & \forall i \in N \\ & \sum_{i \in N} \sum_{b \ni j} x_i(b) \leq 1 & \forall j \in K \end{array}$$

That is, we maximize the sum of the upper bounds we can achieve on the value of each bundle, constrained to picking only non-overlapping bundles and only one bundle per agent.

Upon solving the IP, the elicitor will know the value $UB_{\max}$ of the allocations with greatest upper bound. Then, for each bundle-agent pair $(b, i)$ in turn, we force $x_i(b) = 1$ and solve again. This returns the value $UB_{\max}(b, i)$ of the allocations with greatest upper bound, constrained to only those allocations that allocate $b$ to agent $i$. If $UB_{\max}(b, i) = UB_{\max}$, then $(b, i)$ is in a high-value allocation. The elicitor now has the set of $(b, i)$ that are in high-value allocations, and can proceed as before (picking the bundles that will reduce the value of the largest number of others). Implemented naively, the policy solves the IP for each pair $(b, i)$ every time the elicitor needs to select a query. However, the solution to the IP for $(b, i)$ will often not change between queries; and even if it does, since the IP is finding an upper bound, the solution value can only fall. We therefore cache old solutions and thus avoid recomputing the IP in many cases.

Our experiments show that in our implementation, the implicit representation of allocations is faster than the explicit one already with three agents ($n = 3$). With 5 agents, the implicit approach is several orders of magnitude faster.

As Figure 1 shows, this elicitation policy asks only a small fraction of the queries before the optimal allocation is found, and the elicitation ratio decreases with the number of items in the auction. So, incremental preference elicitation is a promising avenue to tackle the combinatorial auction problem in practice, although it is known that the worst case communication complexity is exponential (even to find an approximately optimal allocation) [4].

# 4. OPTIMAL ELICITATION ALGORITHM

We can easily design an optimal elicitation policy. The elicitor would like to incrementally choose the query that, given all the information it knows, brings it closest to having a certificate (on average). To do this, the optimal algorithm checks, for each possible query, the expected number of queries the elicitor would have to make before it found an optimal allocation. That is, for each possible query, and for each possible answer to that query, how many queries would be left on average?

Unfortunately, a full tree search is infeasible: at the root, the branching factor of the algorithm is $n(2^k - 1)$ queries. At the next level, the branching factor is equal to the number of possible answers, which is large (for bounded integer valuations) or even infinite (for general integer valuations, or continuous valuations). For most of those answers, the algorithm branches on $n(2^k - 1) - 1$ queries, and so on. Even with very aggressive pruning (namely, cheating by always "guessing" the correct answer and thus only branching on the queries), we can only solve this problem exactly on tiny instances ($n = k = 3$).

# 5. UNIVERSAL REVELATION REDUCERS

So far we have presented elicitation policies that, on average over instances, save a large amount of preference revelation. Now we ask the question: Do there exist *universal* elicitors, that is, elicitors that save revelation on all instances (excepting those where even an elicitor guided by an oracle would reveal everything)?

DEFINITION 1. *A universal revelation reducer is an elicitation policy with the following property: given a problem instance, it can guarantee (always in the deterministic case; in expectation over the random choices in the randomized case) saving some elicitation over full revelation—provided the shortest certificate is shorter than full revelation.*

PROPOSITION 2. *The unrestricted random elicitation policy is a universal revelation reducer.*

PROPOSITION 3. *No deterministic value query policy is a universal revelation reducer.*

The proof is based on a fooling set. For any deterministic algorithm, at least one of the elements of the fooling set will cause the algorithm to ask all the questions, even though each element of the fooling set could be cleared with fewer questions.

# 6. FUTURE RESEARCH

Future research includes studying additional query type for elicitation in combinatorial auctions [1–3], identifying special classes of valuations for which elicitation can be done in polynomial time [7], and better understanding the relationship between this work and ascending auctions (e.g. [5]), which can be viewed as preference elicitation using demand queries.

# 7. REFERENCES


[1] Wolfram Conen and Tuomas Sandholm. Preference elicitation in combinatorial auctions: Extended abstract. In *ACM-EC*, pp. 256–259, 2001. A more detailed description of the algorithmic aspects appeared in the IJCAI-2001 Workshop on Economic Agents, Models, and Mechanisms, pp. 71–80.

[2] Wolfram Conen and Tuomas Sandholm. Differential-revelation VCG mechanisms for combinatorial auctions. In *AAMAS-02 workshop on Agent-Mediated Electronic Commerce*, 2002.

[3] Benoit Hudson and Tuomas Sandholm. Effectiveness of preference elicitation in combinatorial auctions. In *AAMAS-02 workshop on Agent-Mediated Electronic Commerce*, 2002. Extended version: Carnegie Mellon University, Computer Science Department, CMU-CS-02-124, March. Also: Stanford Institute for Theoretical Economics workshop (SITE-02).

[4] Noam Nisan and Ilya Segal. The communication complexity of effi cient allocation problems Draft. March, 2002.

[5] David C Parkes. iBundle: An effi cient ascending price bundle auction. In *ACM-EC*, pp. 148–157, 1999.

[6] Trey Smith, Tuomas Sandholm, and Reid Simmons. Constructing and clearing combinatorial exchanges using preference elicitation. In *AAAI-02 workshop on Preferences in AI and CP: Symbolic Approaches*, 2002.

[7] Martin Zinkevich, Avrim Blum, and Tuomas Sandholm. On polynomial-time preference elicitation with value queries. To appear *ACM-EC*, 2003.