Preference Elicitation and Query Learning

Avrim Blum[1], Jeffrey Jackson[2], Tuomas Sandholm[1], and Martin Zinkevich[1]

[1] Carnegie Mellon University, Pittsburgh PA 15213, USA,
maz@cs.cmu.edu,
WWW home page: http://www.cs.cmu.edu/~avrim,sandholm,maz/
[2] Duquesne University,Pittsburgh PA 15282, USA,
jackson@mathcs.duq.edu,
WWW home page: http://www.mathcs.duq.edu/~jackson/

Abstract. In this paper we initiate an exploration of relationships between "preference elicitation", a learning-style problem that arises in combinatorial auctions, and the problem of learning via queries studied in computational learning theory. Preference elicitation is the process of asking questions about the preferences of bidders so as to best divide some set of goods. As a learning problem, it can be thought of as a setting in which there are multiple target concepts that can each be queried separately, but where the goal is not so much to learn each concept as it is to produce an "optimal example". In this work, we prove a number of similarities and differences between preference elicitation and query learning, giving both separation results and proving some connections between these problems.

1 Introduction

In a combinatorial auction, an entity (the "auctioneer") has a set S of n items that he would like to partition among a set of k bidders. What makes an auction *combinatorial* is that the valuations of the bidders (how much they would be willing to pay for different subsets of items) may not necessarily be linear functions over the items. For instance, if item a is a left shoe and item b is a right shoe, then a bidder might be willing to pay a reasonable amount for the bundle $\{a, b\}$ but very little for just $\{a\}$ or just $\{b\}$. In the other direction, if a and b are each *pairs* of shoes, then a bidder might value $\{a, b\}$ less than the sum of his valuations on $\{a\}$ and $\{b\}$ (especially if he just needs one pair of shoes right now). A standard goal for the auctioneer in such a setting is to determine the allocation of goods that maximizes *social welfare*: this is the sum, over all bidders, of the value that each bidder places on the set of items that he receives. This goal is perhaps most natural if one thinks of the auctioneer as not having a financial interest of its own but simply as an agent acting to help divide up a given set of items in a way that maximizes overall happiness. For example, the case of $k = 2$ can be thought of as a situation in which one of the bidders represents a buyer (with various preferences over bundles of items) and the other bidder represents a marketplace (with various discounts and package-deals), and the auctioneer is acting as an agent to help the buyer decide what subset of items to purchase from the marketplace.[1]

There are a number of issues that arise in the combinatorial auction setting. For example, there is much work on designing protocols (mechanisms) so that bidders will be truthful in reporting their valuations and not want to "game" the system (e.g., [1–3]). But another issue is that even if we can get bidders to be truthful, their valuation functions can be quite complicated. Because of this, bidding in a traditional manner can require an exponential amount of communication. This has led researchers to study the notion of preference elicitation, in which the auctioneer asks questions of the bidders in order to learn (elicit) enough information about their preferences so as to be able to decide on the best (or approximately best) allocation of the items. Because the issues of truthfulness can be handled even in this setting of incremental preference elicitation via known mechanisms [4], much of this previous work—as well as this paper—focuses solely on the elicitation question of how to extract the necessary information needed for allocation.

1.1 Preference Elicitation and Query Learning

We can think of preference elicitation in the context of query learning by thinking of the n items as features, thinking of a subset of items (a "bundle") as an example $x \in \{0,1\}^n$ indicating which items are in the subset, and thinking of the bidder's valuation function as a target function. The standard assumption of "free disposal" (bidders can throw away items for free), means we can assume that these valuation functions are *monotone*, though they typically will not be boolean-valued. Furthermore, one of the natural types of queries studied in preference elicitation, the *value query* (where the auctioneer asks the bidder how much he values some bundle), corresponds exactly with the learning-theoretic notion of a membership query.

On the other hand, a key difference between preference elicitation and query learning is in the goals. In learning, the objective is to exactly or approximately recover the target function. In preference elicitation, however, the goal is more one of finding the "best example". For instance, if there are just two bidders with preference functions f and g, then the goal is to find a partition (S', S'') of the n items to maximize $f(S') + g(S'')$. Thinking in terms of functions over $\{0,1\}^n$, the goal is to find $x \in \{0,1\}^n$ to maximize $f(x) + g(\bar{x})$.

Notice that one of the immediate differences between preference elicitation and query learning is that preference elicitation makes sense even if the target functions do not have short descriptions, or even short approximations. We will see some interesting examples later, but as a simple case, if we learn that bidder A will pay \$100 for the entire set of n items but no more than \$50 for any subset of size $n-1$ (she is a collector and wants the whole set), and B will pay a

[1] To think of this as a combinatorial auction, it is easiest to imagine that the auctioneer has pre-purchased *all* the items, and then is deciding which the buyer should keep and which should be returned for a refund.

maximum of \$50 even for the whole lot, then we know we might as well give all items to A, and we do not need to know exactly how much each bidder would have paid for different subsets. On the other hand, it is quite possible for allocation of items to be *computationally* hard, even if the preferences of all the bidders are known. For example, even if each bidder's preferences can be expressed as a simple conjunction (these are called "single-minded" bidders), then if there are many bidders, allocation is equivalent to the NP-hard set-packing problem. For somewhat more complicated preference functions, such as read-once formulas, allocation can be NP-hard even for two bidders [5].

Another difference concerns the types of queries that are most natural in each setting. While value/membership queries are common to both, equivalence queries are quite *un*natural in the context of preference elicitation. On the other hand, the *demand query*, a powerful type of query for preference elicitation introduced by Nisan [6], does not seem to have been studied in query learning.[2]

In this paper, we discuss similarities and differences between the three objectives of exact learning, approximate learning, and preference elicitation. We then give a number of upper and lower bounds for preference elicitation of natural preference (concept) classes. We focus primarily on the case of $k = 2$ bidders, because even this case is quite interesting, both practically (since it models a buyer and a marketplace as mentioned above) and technically. We show that monotone DNF formulas (long known to be hard to learn exactly from membership queries alone but easy to learn approximately) are *hard* for preference elicitation, even with demand queries. However, the hardness we show is $2^{\Omega(\sqrt{n})}$-hard rather than $2^{\Omega(n)}$-hard. On the other hand, $\log(n)$-DNF are *easy* for preference-elicitation, even if the functions have more than polynomially many terms. We also give a number of general statements about when the ability to succeed for one of these goals implies being able to succeed in the others. We then end with a number of open problems.

1.2 Related Work on Combinatorial Auctions

Combinatorial auctions are economically efficient mechanisms for selling n items to multiple 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). Determining the winners in such auctions, given the bids, is a complex optimization problem that has received considerable attention (e.g., [7–9, 2, 10, 11]). Equally important, however, is the problem of communication. There are $2^n - 1$ bundles, and each agent may need to bid on all of them to fully express its preferences. Appropriate bidding languages [8, 12, 9, 2, 13, 14] can address the communication overhead in some cases where the bidder's utility function is compressible. However, they still require the agents to completely determine

[2] In this query, the auctioneer proposes a set of item prices and then asks the bidder what set of items he would choose to buy at those prices. These will be discussed further in Section 2.

and transmit their valuation functions and as such do not solve all the issues. So in practice, when the number of items for sale is even moderate, the bidders cannot bid on all bundles. Instead, they may bid on bundles which they will not win, and they may fail to bid on bundles they would have won. The former problem leads to wasted effort, and the latter problem leads to reduced economic efficiency of the resulting allocation of items to bidders.

Selective *incremental preference elicitation* by the auctioneer was recently proposed to address these problems, and several papers have studied different types of elicitors [4, 15, 16, 6, 17]. On the negative side, if valuations are arbitrary monotone functions, then the worst-case communication complexity to find an (even approximately) optimal allocation is exponential in the number of items, no matter what query types are used [6]. However, experimentally, only a small decreasing fraction of the bidders' preferences can be elicited before the provably optimal allocation is found [16].

Vickrey-Clark-Groves [18–20] schemes provide a method for charging bidders so that each is motivated to tell the truth about its valuations. Briefly, in this scheme the elicitor first finds the optimal allocation OPT. Then, for each bidder i, it finds the optimal allocation OPT_i without bidder i. Bidder i is charged a fee based on the difference between the utility of the other agents in OPT and OPT_i. One then proves that in such a scheme, each bidder is motivated to be truthful. This means that if one can elicit the optimal allocation exactly assuming that agents tell the truth, one can determine the Vickrey payments that make truth-telling a good strategy for the bidders. Because of this, for the remainder of the paper we assume that the bidders are truthful.

Driven by the same concerns as preference elicitation in combinatorial auctions, there has been significant recent work on ascending combinatorial auctions (e.g., [21–26]). These are multistage mechanisms. At each stage the auctioneer announces prices (on items or in some cases on bundles of items), and each bidder states which bundle of items he would prefer (that is, which bundle would maximize his valuation minus the price he would have to pay for the bundle) at those prices. The auctioneer increases the prices between stages, and the auction usually ends when the optimal allocation is found. Ascending auctions can be viewed as a special case of preference elicitation where the queries are demand queries ("If these were the prices, what bundle would you buy from the auction?") and the query policy is constrained to increasing the prices in the queries over time. Recently it was shown that if *per-item* prices suffice to support an optimal allocation (i.e., a *Walrasian equilibrium* exists), then the optimal allocation can be found with a polynomial number of queries (where each query and answer is of polynomial size) [6].

Recently, some of us [5], noticing the connection to query learning, showed how the AHK algorithm [27] could be adapted to elicit preferences expressable as read-once-formulas over gates that are especially natural in the context of combinatorial auctions. This work goes on to discuss the computational problem of determining the best allocation once the formulas are elicited. On the negative side, it shows that even for two bidders with read-once-formula preferences, allocation can be NP-hard, but on the other hand, if one of the two bidders has a linear value function, then allocation can be done in polynomial time.

2 Notation and Definitions

Because subset notation is most natural from the point of view of preference elicitation, we will use both subset notation and bit-vector notation in this paper. That is, we will think of the instance space X both as elements of $\{0,1\}^n$ and as the power set of some set S of *n items*. We will also interchangeably call a subset of S a "bundle" or an "example". When discussing preference elicitation, we assume there are k bidders with monotone real-valued preference functions over the instance space. The objective of preference elicitation is to determine a k-way partition $(S_1, \ldots, S_k)$ of S to maximize $f_1(S_1) + f_2(S_2) + \ldots + f_k(S_k)$, where $f_1, \ldots, f_k$ are the k real-valued preference functions. Typically we will assume $k = 2$.

Let C be a class of monotone functions. We will be interested in the learnability of various C in the exact learning, approximate learning, and preference elicitation models given the ability to make various types of queries. By "approximate learning" we mean learning with respect to the uniform distribution on inputs — i.e.., finding a hypothesis function that agrees with the target over almost all of the instance space. While learning algorithms are typically considered efficient if they run in time polynomial in the number of items n and in the length of the representation of the target (and possibly other parameters), we will at times explicitly require run time bounds independent of description length in order to demonstrate a fundamental advantage of preference elicitation for problems involving complex targets. The hardness observations for learning problems when this restriction is in place are therefore not hardness results in the standard learning-theoretic sense.

Query types: A *membership query* or *value query* is a request $x \in \{0,1\}^n$ to an oracle for a target f. The oracle responds with the value $f(x)$ corresponding to x. We can think of these queries as asking the following question of a bidder: "How much are you willing to pay for this bundle of items?"

A *demand query* is a request $w \in (\mathbf{R}^+)^n$ ($\mathbf{R}^+$ here represents non-negative real values) to an oracle for a target f. The oracle responds with an example $x \in \{0,1\}^n$ that maximizes $f(x) - w \cdot x$. We can think of these queries as asking the following question of a bidder: "If these are the costs of items, what would you choose to buy?"

We can illustrate the power of demand queries with the following observation due to Nisan. If one of the bidders has a linear valuation function, and the other is arbitrary, then preference elicitation can be done with $n+1$ queries: n value queries and one demand query. Specifically, we simply ask the linear bidder n value queries to determine his value on each item, and then send the other bidder these values as prices and ask him what he would like to buy. Thus it is interesting that our main lower bounds hold for demand queries as well.

Natural function/representation classes: One of the most natural representation classes of monotone functions in machine learning is that of monotone DNF formulas. In preference elicitation, the analog of this representation is called the "XOR bidding language"[3]. A preference in this representation is a set of bundles (terms) $T = \{T_1, T_2, \ldots, T_m\}$ along with values v_i for each bundle T_i. The value of this preference for all $S' \subseteq S$ is:

$$f_{T,v}(S') = \max_{T_i \subseteq S'} v_i.$$

In other words, the value of a set of items S' is the maximum value of any of the "desired bundles" in T that are contained in S'. We will call this the *DNF representation* of preferences, or "DNF preferences" for short. Our hardness results for this class will all go through for the boolean case (all v_i are equal to 1), but our positive results will hold for general v_i.

3 DNF Preferences

Angluin [28] shows that monotone DNF formulas are hard to exactly learn from membership queries alone, but are easy to learn approximately. Angluin's example showing hardness of exact learning can be thought of as follows: imagine the n items are really $n/2$ pairs of shoes. The buyer would be happy with any bundle containing at least one pair of shoes (any such bundle is worth \$1). But then we add one final term to the DNF: a bundle of size $n/2$ containing exactly one shoe from each pair, where for each pair we flip a coin to decide whether to include the left or right shoe. Since the learning algorithm already knows the answer will be positive to any query containing a pair of shoes, the only interesting queries are those that contain no such pair, and therefore it has to match the last term *exactly* to provide any information. Thus even for a randomized algorithm, an expected $2^{n/2-1}$ queries are needed for exact learning of monotone DNF.

We now consider the preference elicitation problem when one or more preferences are represented as monotone DNF expressions, beginning with a few simple observations.

Observation 1 *If f is a* known *DNF preference function with m terms, and g is an* arbitrary *unknown monotone preference function, then preference elicitation can be performed using m value queries.*

Proof. Because g is monotone, the optimal allocation will be of the form $(T_i, S - T_i)$ for some term T_i in f. So, we simply need to query g once for each set $S - T_i$ and then pick the best of these m partitions. $\square$

[3] This terminology is to indicate that the bidder wants only one of his listed bundles and will not pay more for a set of items that contains multiple bundles inside it. This usage is very different from the standard definition of XOR as a sum modulo 2. Therefore, to avoid confusion, we will not use the XOR terminology here.

Observation 2 *If f and g are boolean DNF preferences each containing exactly one term that is not size 2 (the hard case in Angluin's construction) then preference elicitation can be performed using polynomially (in n) many value queries.*

Proof. We begin by finding all terms in f of size 2 by asking n^2 queries. Suppose two of these terms T_1 and T_2 are disjoint. In that case, we query g on $S - T_1$ and $S - T_2$. If one answer is "yes" then we are done. If both answers are "no" then this means all of g's terms intersect both T_1 and T_2. In particular, g can have only a constant number of terms, and therefore *exactly* learning g is easy, after which we can then apply Observation 1 (swapping f and g). On the other hand, if f does not have two disjoint terms of size 2, then the only way f can have more than 3 such terms is if they all share some common item x_i. It is thus now easy to learn the large term in f: if $f(S - \{x_i\})$ is positive, we can "walk downward" from that example to find it, else we can walk downward from the example in which all the *other* items in the small terms have been removed. Once f has been learned, we can again apply Observation 1. □

We now show that even though Angluin's specific example is no longer hard in the preference elicitation model, monotone DNF formulas remain hard for preference elicitation using value queries, even when the preference functions are quite small. We then extend this result to demand queries as well.

Theorem 1. *Preference elicitation of monotone DNF formulas requires $2^{\Omega(\sqrt{n})}$ value queries. This holds even if each bidder's preference function has only $O(\sqrt{n})$ terms.*

Proof. We construct a hard example as follows. There will be $n = m^2$ items, arranged in an m-by-m matrix. Let us label the items x_{ij} for $1 \leq i, j \leq m$. We will call the two preference functions f_R and f_C. Both will be boolean functions. Bidder f_R is happy with any row: that is, $f_R = x_{11}x_{12} \cdots x_{1m} \vee x_{21}x_{22} \cdots x_{2m} \vee \ldots \vee x_{m1}x_{m2} \cdots x_{mm}$. Bidder f_C is happy with any column: that is, $f_C = x_{11}x_{21} \cdots x_{m1} \vee x_{21}x_{22} \cdots x_{m2} \vee \ldots \vee x_{1m}x_{2m} \cdots x_{mm}$. Thus, at this point, it is impossible to make both bidders happy. However, we now add one additional term to each preference function. We flip a coin for each of the n items in S, labeling the item as heads or tails. Let H be the set of all items labeled heads, and T be the set of all items labeled tails. We now add the conjunction of all items in H as one additional term to f_R, and the conjunction of all items in T as one additional term to f_C. Thus now it *is* possible to make both bidders happy, and the optimal allocation will be to give the items in H to the "row bidder" and the items in T to the "column bidder".

We now argue that no query algorithm can find this allocation in less than $\frac{1}{2}2^{\sqrt{n}} - 2$ queries in expectation. Let us enforce that the last two questions of the query protocol are the values of the actual allocation. That is, if the elicitor assigns the items in H to the row agent and T to the column agent, it must ask the row agent the value of H and the column agent the value of T. This constraint only increases the length of the protocol by at most 2 questions.

Let us assume that the elicitor knows in advance the structure of the problem, the row sets and the column sets, and the only information the elicitor does not know are the sets H and T. In this case, we can assume without loss of generality that the elicitor never asks the row bidder about any bundle containing a row (because he already knows the answer will be "yes") and similarly never asks the column bidder about any bundle containing a column.

We now argue as follows. If the elicitor asks a query of the row bidder, the query must be missing at least one item in each row, and if the elicitor ask a query of the column bidder, it must be missing at least one item in each column. However, notice that in the first case, the answer will be positive only if all missing items are in T, and in the second case, the answer will be positive only if all missing items are in H. Therefore, for any given such query, the probability that the answer will be positive taken over the random coin flips is at most $2^{-\sqrt{n}}$. Thus, for any elicitation strategy, the probability the elicitor gets a positive response in the first k queries is at most $k2^{-\sqrt{n}}$ and therefore the expected number of queries is at least $\frac{1}{2}2^{\sqrt{n}}$. □

We now show that preference elicitation remains hard for DNF preferences even if we allow demand queries.

Theorem 2. *Even if both demand queries and value queries are allowed, preference elicitation of monotone DNF formulas requires $2^{\Omega(\sqrt{n})}$ queries. This holds even if each bidder's preference function has only $O(\sqrt{n})$ terms.*

Proof. We use the same example as in the proof of Theorem 1. As in that proof, we can insist that the last question be a demand query where the agent responds with the set H or T respectively. Let us without loss of generality consider a sequence of demand queries to the "row bidder". What we need to calculate now is the probability, for any given cost vector w, that the set H happens to be the cheapest term in his DNF formula. The intuition is that this is highly unlikely because H is so much larger than the other terms.

Specifically, for a given query cost vector w, let w_i be the total cost of the ith row. Thus, the cheapest row has cost $\min(w_1, \ldots, w_m)$ and the *expected* cost of H is $\frac{1}{2}(w_1 + \ldots + w_m)$. One simple observation that helps in the analysis is that if we define h_i as the cost of the items in H that are in the ith row, then $\Pr(h_i \geq w_i/2) \geq 1/2$. That is because if any particular subset of the ith row has cost less than $w_i/2$, its complement in the ith row must have cost greater than $w_i/2$. Furthermore, these events are independent over the different rows.

So, we can reduce the problem to the following: we have m independent events each of probability at least $1/2$. If at least two of these events occur, the elicitor gets no information (H is not the cheapest bundle because it is not cheaper than the cheapest row). Thus, the probability the elicitor *does* get some information is at most $(m+1)2^{-m}$ and the expected number of queries is at least $\frac{1}{2(m+1)}2^m$. □

Open Problem 1 *Can preferences expressible as polynomial-size DNF formulas be elicited in $2^{O(\sqrt{n})}$ value queries or demand queries?*

3.1 log(n)-DNF Preferences

In the previous problem, even though there were only $O(\sqrt{n})$ terms in each preference function, the terms themselves were fairly large. What if all of the terms are small, of size no more than $\log n$? Observe that there are $\binom{n}{\log n}$ possible terms of size $\log n$, so some members of this class cannot be represented in poly(n) bits.

Theorem 3. *If f and g are DNF-preferences where no term is of size more than $\log_2 n$, then preference elicitation can be performed in a number of value queries polynomial in n.*

Proof. We begin by giving a randomized construction and then show a derandomization.

For convenience let us put an empty term T_0 of value 0 into both f and g. With this convention we can assume the optimal allocation satisfies some term $T' \in f$ and some term $T'' \in g$.

We now simply notice that since T' and T'' are both of size at most $\log_2 n$, a random partition (S', S'') has probability at least $1/n^2$ of satisfying $S' \supseteq T'$ and $S'' \supseteq T''$. So, we simply need to try $O(n^2 \log \frac{1}{\delta})$ random partitions and take the best one, and with probability at least $1 - \delta$ we will have found the optimal allocation.

We can now derandomize this algorithm using the (n, k)-universal sets of Naor and Naor [29]. A set of assignments to n boolean variables is (n, k)-universal if for every subset of k variables, the induced assignments to those variables covers all 2^k possible settings. Naor and Naor [29] give efficient explicit constructions of such sets using only $2^{O(k)} \log n$ assignments. In our case, we can use the case of $k = 2\log_2 n$, so the construction is polynomial time and size. Each of these assignments corresponds to a partition of the items, and we simply ask f and g for their valuations on each one and take the best. □

4 General Relationships

In this section we describe some general relationships between query learning and preference elicitation. We begin with an example in which preference elicitation is easy but exact learning is hard, even though the function has a small description. We then show that in certain circumstances, however, the ability to elicit does imply the ability to learn with queries.

4.1 Almost-Threshold Preferences

We now define a class of preference functions that we call *almost-threshold*. This class will be used to show that, even if all of the functions in a class have representations of size polynomial in n, we can still separate exact learning and preference elicitation with respect to membership queries.

An "almost threshold" preference function is defined by specifying a single set S'. This set in turn defines a preference function that is 1 for any set of size greater than or equal to $|S'|$, except for S' itself, and is 0 otherwise. Formally, for any $S' \neq \emptyset$, define:

$$h_{S'}(S'') = \begin{cases} 1 \text{ if } S'' \neq S' \text{ and } |S''| \geq |S'| \\ 0 \text{ otherwise} \end{cases}$$

The class H_{AT} of almost-threshold preference functions is then $H_{AT} = \{h_{S'}\}$.

Observation 3 *It requires at least $\binom{n}{\lceil n/2 \rceil - 1}$ membership queries to exactly learn the class H_{AT}.*

Theorem 4. *If $f, g \in H_{AT}$ then the optimal allocation can be elicited in $4 + \log_2 n$ membership queries.*

Proof. Assume $|S| > 2$ and suppose $f = h_{S'}$. The first step is to determine $|S'|$. We can do this in $\log_2 n + 1$ queries using binary search. We next use two more queries to find two sets T, T' of size $|S'|$ such that $f(T) = f(T') = 1$. This can be done by just picking three arbitrary sets of size $|S'|$ and querying the first two: if either has value 0 then the third has value 1. Then, we test if $g(S \backslash T) = 1$. If it is, then $T, S \backslash T$ is an optimal allocation. Otherwise, $T', S \backslash T'$, regardless of its value, is an optimal allocation. □

4.2 Positive Results

We now show that in certain circumstances, however, the ability to elicit does imply the ability to learn with queries. In particular, we will show that in certain cases, the ability to perform preference elicitation will provide us with a Superset Query oracle, which together with membership queries can allow us to to learn concept classes not learnable by membership queries alone.

Definition 1. *A* **superset query oracle** *for a concept class H takes in a function $f \in H$ as input. If f is a superset of the target f^*, that is, $\{x : f(x) = 1\} \supseteq \{x : f^*(x) = 1\}$, then the query returns "true". Otherwise the query produces a counterexample: an x such that $f(x) = 0$ but $f^*(x) = 1$.*

Notice that Angluin's algorithm [28] for learning Monotone DNF can use superset queries instead of equivalence queries, because the hypothesis is always a subset of the target function. Furthermore, any subclass of Monotone DNF that is closed under removal of terms can be learned from superset queries and membership queries by the same algorithm.

What makes this interesting is the following relationship between preference elicitation and superset queries. First, for any boolean function f, let us define its "dual"

$$\hat{f}(S') = 1 - f(S \backslash S').$$

Or, in other words, $\hat{f}(x) = \bar{f}(\bar{x})$. Given a hypothesis space H, define $\hat{H} = \{\hat{f} : f \in H\}$. For example, the dual of $\log(n)$-DNF preferences is $\log(n)$-CNF preferences. The set of monotone functions is closed under dual.

Theorem 5. *If, given $f \in H$ and $g \in \hat{H}$, one can elicit the optimal allocation S, S' using M value queries, then one can perform a superset query on H using $M + 2$ membership queries.*

Proof. Suppose that f^* is the target concept and one wants to perform a superset query with $g \in H$. First, compute $\hat{g} \in \hat{H}$. Then, perform preference elicitation on $f^*, \hat{g}$. If this procedure returns an allocation satisfying both parties, this means we have an x such that $f^*(x) = 1$ and $\hat{g}(\bar{x}) = 1$. But, $\hat{g}(\bar{x}) = \bar{g}(x)$ so this means that x is a counterexample to the superset query. On the other hand if the elicitation procedure fails to do so, then this means no such x exists so the superset query can return "true". □

Corollary 1. *If H is a subclass of monotone DNF that is closed under removal of terms, and if one can perform preference elicitation for $(H, \hat{H})$, then H is learnable from membership queries alone.*

5 Conclusions and Open Problems

In machine learning, one's objective is nearly always to learn or approximately learn some target function. In this paper, we relate this to the notion of preference elicitation, in which the goal instead is to find the optimal partitioning of some set of items (to find an example x maximizing $f(x) + g(\bar{x})$.)

We now describe several open problems left by this work. We begin with a problem stated above in Section 3.

Open Problem 1 *Can preferences expressible as polynomial-size DNF formulas be elicited in $2^{O(\sqrt{n})}$ value queries or demand queries?*

A somewhat fuzzier question related to our results on $\log(n)$-DNF is the following. Our algorithm in this case was non-adaptive: the questions asked did not depend on answers to previous questions. It seems natural that for some classes adaptivity should help. In fact, it not hard to generate artificial examples in which this is the case. However, we know of no natural example having this property.

Open Problem 2 *Are there natural classes of functions for which exact learning is information-theoretically hard, preference elicitation via a non-adaptive algorithm is hard (i.e., one in which the questions can all be determined in advance) but elicitation by an adaptive algorithm is easy.*

One of the oldest techniques for preference elicitation is an ascending auction. An ascending auction can be considered to be a sequence of increasing demand queries, where if one asks a query w' after a query w, then it must be the case that for all i, $w'_i \geq w_i$. One interesting open question is:

Open Problem 3 *Does there exist a preference elicitation problem that is hard (or impossible) to elicit using an ascending auction but easy to elicit using demand queries?*

Acknowledgements

This material is based upon work supported under NSF grants CCR-0105488, ITR CCR-0122581, CCR-0209064, ITR IIS-0081246, and ITR IIS-0121678. Any opinion, findings, conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. Sandholm, T.: eMediator: A next generation electronic commerce server. Computational Intelligence **18** (2002) 656–676 Special issue on Agent Technology for Electronic Commerce. Early versions appeared in the Conference on Autonomous Agents (AGENTS-00), pp. 73–96, 2000; AAAI-99 Workshop on AI in Electronic Commerce, Orlando, FL, pp. 46–55, July 1999; and as a Washington University, St. Louis, Dept. of Computer Science technical report WU-CS-99-02, Jan. 1999.
2. Nisan, N.: Bidding and allocation in combinatorial auctions. In: Proceedings of the ACM Conference on Electronic Commerce (ACM-EC), Minneapolis, MN (2000) 1–12
3. Lehmann, D., O'Callaghan, L.I., Shoham, Y.: Truth revelation in rapid, approximately efficient combinatorial auctions. Journal of the ACM (2003) To appear. Early version appeared in ACMEC-99.
4. Conen, W., Sandholm, T.: Preference elicitation in combinatorial auctions: Extended abstract. In: Proceedings of the ACM Conference on Electronic Commerce (ACM-EC), Tampa, FL (2001) 256–259 A more detailed description of the algorithmic aspects appeared in the IJCAI-2001 Workshop on Economic Agents, Models, and Mechanisms, pp. 71–80.
5. Zinkevich, M., Blum, A., Sandholm, T.: On polynomial-time preference elicitation with value queries. In: Proceedings of the ACM Conference on Electronic Commerce (ACM-EC), San Diego, CA (2003)
6. Nisan, N., Segal, I.: The communication complexity of efficient allocation problems (2002) Draft. Second version March 5th.
7. Rothkopf, M.H., Pekeč, A., Harstad, R.M.: Computationally manageable combinatorial auctions. Management Science **44** (1998) 1131–1147
8. Sandholm, T.: Algorithm for optimal winner determination in combinatorial auctions. Artificial Intelligence **135** (2002) 1–54 First appeared as an invited talk at the First International Conference on Information and Computation Economies, Charleston, SC, Oct. 25–28, 1998. Extended version appeared as Washington Univ., Dept. of Computer Science, tech report WUCS-99-01, January 28th, 1999. Conference version appeared at the International Joint Conference on Artificial Intelligence (IJCAI), pp. 542–547, Stockholm, Sweden, 1999.
9. Fujishima, Y., Leyton-Brown, K., Shoham, Y.: Taming the computational complexity of combinatorial auctions: Optimal and approximate approaches. In: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI), Stockholm, Sweden (1999) 548–553
10. Andersson, A., Tenhunen, M., Ygge, F.: Integer programming for combinatorial auction winner determination. In: Proceedings of the Fourth International Conference on Multi-Agent Systems (ICMAS), Boston, MA (2000) 39–46

11. Sandholm, T., Suri, S., Gilpin, A., Levine, D.: CABOB: A fast optimal algorithm for combinatorial auctions. In: Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI), Seattle, WA (2001) 1102–1108
12. Sandholm, T.: eMediator: A next generation electronic commerce server. In: Proceedings of the Fourth International Conference on Autonomous Agents (AGENTS), Barcelona, Spain (2000) 73–96 Early version appeared in the AAAI-99 Workshop on AI in Electronic Commerce, Orlando, FL, pp. 46–55, July 1999, and as a Washington University, St. Louis, Dept. of Computer Science technical report WU-CS-99-02, Jan. 1999.
13. Hoos, H., Boutilier, C.: Bidding languages for combinatorial auctions. In: Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI), Seattle, WA (2001) 1211–1217
14. Sandholm, T., Suri, S.: Side constraints and non-price attributes in markets. In: IJCAI-2001 Workshop on Distributed Constraint Reasoning, Seattle, WA (2001) 55–61
15. Conen, W., Sandholm, T.: Differential-revelation VCG mechanisms for combinatorial auctions. In: AAMAS-02 workshop on Agent-Mediated Electronic Commerce (AMEC), Bologna, Italy (2002)
16. Hudson, B., Sandholm, T.: Effectiveness of preference elicitation in combinatorial auctions. In: AAMAS-02 workshop on Agent-Mediated Electronic Commerce (AMEC), Bologna, Italy (2002) Extended version: Carnegie Mellon University, Computer Science Department, CMU-CS-02-124, March. Also: Stanford Institute for Theoretical Economics workshop (SITE-02).
17. Smith, T., Sandholm, T., Simmons, R.: Constructing and clearing combinatorial exchanges using preference elicitation. In: AAAI-02 workshop on Preferences in AI and CP: Symbolic Approaches. (2002) 87–93
18. Vickrey, W.: Counterspeculation, auctions, and competitive sealed tenders. Journal of Finance **16** (1961) 8–37
19. Clarke, E.H.: Multipart pricing of public goods. Public Choice **11** (1971) 17–33
20. Groves, T.: Incentives in teams. Econometrica **41** (1973) 617–631
21. Parkes, D.C.: Optimal auction design for agents with hard valuation problems. In: Agent-Mediated Electronic Commerce Workshop at the International Joint Conference on Artificial Intelligence, Stockholm, Sweden (1999)
22. Parkes, D.C.: iBundle: An efficient ascending price bundle auction. In: Proceedings of the ACM Conference on Electronic Commerce (ACM-EC), Denver, CO (1999) 148–157
23. Ausubel, L.M., Milgrom, P.: Ascending auctions with package bidding. Technical report (2001) Draft June 7th.
24. Wurman, P.R., Wellman, M.P.: AkBA: A progressive, anonymous-price combinatorial auction. In: Proceedings of the ACM Conference on Electronic Commerce (ACM-EC), Minneapolis, MN (2000) 21–29
25. Bikhchandani, S., de Vries, S., Schummer, J., Vohra, R.V.: Linear programming and Vickrey auctions (2001) Draft.
26. Bikhchandani, S., Ostroy, J.: The package assignment model. UCLA Working Paper Series, mimeo (2001)
27. Angluin, D., Hellerstein, L., Karpinski, M.: Learning read-once formulas with queries. In: Journal of the ACM. Volume 40. (1993) 185–210
28. Angluin, D.: Queries and concept learning. Machine Learning **2** (1988) 319–342
29. Naor, J., Naor, M.: Small-bias probability spaces: Efficient constructions and applications. In: Proc. 22nd Annual ACM Symposium on Theory of Computing, Baltimore (1990) 213–223