# Online Feature Elicitation in Interactive Optimization

**Craig Boutilier** CEBLY@CS.TORONTO.EDU
**Kevin Regan** KMREGAN@CS.TORONTO.EDU
**Paolo Viappiani** PAOLO@CS.TORONTO.EDU
Dept. of Computer Science, University of Toronto, Toronto, ON, CANADA

## Abstract

Most models of utility elicitation in decision support and interactive optimization assume a predefined set of "catalog" features over which user preferences are expressed. However, users may differ in the features over which they are most comfortable expressing their preferences. In this work we consider the problem of *feature elicitation*: a user's utility function is expressed using features whose definitions (in terms of "catalog" features) are unknown. We cast this as a problem of *concept learning*, but whose goal is to identify only enough about the concept to enable a good decision to be recommended. We describe computational procedures for identifying optimal alternatives w.r.t. *minimax regret* in the presence of concept uncertainty; and describe several heuristic query strategies that focus on reduction of *relevant* concept uncertainty.

## 1. Introduction

Many decision support settings are designed to help users effectively explore a space of possible alternatives (products, system configurations, plans, etc.) to find one that is optimal (or at least acceptable) with respect to the user's preferences. The ability to tailor recommendations to the needs and desires of particular users requires the incorporation of *preference or utility elicitation* into the navigation process. Considerable work in AI, decision analysis and operations research has been devoted to effective means of eliciting preferences [4, 5, 7, 19]; much of this work can be viewed as a form of *interactive optimization*.

Typical frameworks for preference elicitation assume that user preferences are specified over a predefined set of attributes. For instance, in consumer product configuration—say, choice of an automobile—preferences are assumed to be defined in terms of product features and specifications (e.g., color, engine size, fuel economy, available options, etc.). We refer to such "universal" attributes in a specific domain as the *catalog features*.[1] However, just as preferences can vary significantly from user to user, so too can the features over which their preferences are most naturally expressed. For example, some users may be concerned about the "degree of safety" of a car, but different users may have different notions of safety in mind, none of which are catalog features. Furthermore, the user-specific *subjectivity* of safety prevents one from adding a new feature to the catalog.

In this paper, we develop a framework for eliciting *subjective features* in interactive optimization. We assume that subjective features can be defined in terms of the set of catalog features and that preferences are defined over both catalog and subjective features. As such, learning subjective feature definitions can be cast as a *concept learning problem* [1, 12, 13]. However, unlike traditional concept learning, our aim is not to learn the concept definition *per se*; rather we want to learn *just enough* about it to make a good or optimal decision on the user's behalf. To illustrate, suppose we have positive and negative examples of safe cars, which constrain but do not fully specify (subjective) safety: as a result, only certain cars are *consistent* with possible realizations of the definition of safety. Other user preference information (e.g., utility tradeoffs between safety, performance and cost) may render any "safety consistent" car so undesirable that we will not recommend a safe car no matter how the concept definition is realized. Further elicitation (i.e., refinement of the version space) is therefore useless.

Our focus in this paper is on feature elicitation. We develop a framework in which other aspects of a user's

Appearing in *Proceedings of the 26$^{th}$ International Conference on Machine Learning*, Montreal, Canada, 2009. 

[1] This is meant merely to be evocative of a product catalog; we do not assume that an explicit catalog exists.

utility function are known and only the feature definition is unspecified. We use *minimax regret* [4, 20] as our decision criterion given concept uncertainty, allowing good or optimal decisions to be made without complete feature specification. We describe an integer program formulation for computation of minimax regret in the case of conjunctive concepts. We also present several heuristic techniques for querying concept definitions that reduce minimax regret quickly. In contrast to standard concept learning models, we aim to reduce "relevant" concept uncertainty w.r.t. the utility model, rather than learn an accurate concept definition for its own sake. While simultaneously querying users about both utility and feature definitions is important, in this work we focus on feature elicitation itself (however, we briefly discuss an extension of our model to the simultaneous case).

## 2. Regret-based Optimization and Elicitation

We assume a decision support task in which our system is charged with recommending an outcome to a user from some feasible set; for concreteness, we will will use terminology associated with consumer product configuration, but our model applies to any multiattribute decision domain with constraints on feasible outcomes. Products are characterized by a set of attributes $\mathcal{X} = \{X_1, ...X_n\}$, each with finite domains $Dom(X_i)$. Let $\mathbf{X} \subseteq Dom(\mathcal{X})$ denote the set of *feasible configurations*. For instance, attributes may correspond to the catalog features of various cars (engine size, fuel economy, cargo capacity, etc.) with $\mathbf{X}$ defined by constraints on attribute combinations, or an explicit database of feasible configurations. The user has a *utility function* $u : Dom(\mathcal{X}) \rightarrow \mathbb{R}$, typically parameterized compactly (e.g, using linear, additive or generalized additive form [6, 10, 16]). The precise form of $u$ is not critical, only that $u(\mathbf{x}; w)$ is linear in the parameters (or weights) $w$.

Our system does not have direct access to the user's utility parameters $w$; but elicitation can be used to refine its knowledge of $w$. However, decisions will still generally be made without full knowledge of $w$ for two key reasons [4]. First, good or optimal decisions can often be made with little utility information. Second, the value of obtaining certain utility information—in terms of its impact on decision quality—is often not worth the cost of obtaining it.

Assume a decision must be made, but the system only knows that $w \in W$, i.e., the user's utility function lies in some space $W$. We use the *minimax-regret decision criterion* for making decisions in the face of such utility function uncertainty. Minimax regret [20] has been advocated as a means for robust optimization in the presence of data uncertainty [17], and has more recently been used for decision making with utility uncertainty [4, 5, 19]. Following [4]) we have:

**Definition 1** *Given utility space $W$, define the* max regret *of $\mathbf{x} \in \mathbf{X}$, the* minimax regret *of $W$ and the* minimax optimal configuration *as follows:*

$$MR(\mathbf{x}; W) = \max_{w \in W} \max_{\mathbf{x}' \in \mathbf{X}} u(\mathbf{x}'; w) - u(\mathbf{x}; w)$$
$$MMR(W) = \min_{\mathbf{x} \in \mathbf{X}} MR(\mathbf{x}, W)$$
$$\mathbf{x}^*_W = \arg\min_{\mathbf{x} \in \mathbf{X}} MR(\mathbf{x}, W)$$

Intuitively, $MR(\mathbf{x}, W)$ is the worst-case loss associated with recommending configuration $\mathbf{x}$; i.e., by assuming an adversary will choose the user's utility function $w$ from $W$ to maximize the difference in utility between the optimal configuration (under $w$) and $\mathbf{x}$. The minimax optimal configuration $\mathbf{x}^*_W$ minimizes this potential loss. $MR(\mathbf{x}, W)$ bounds the loss associated with $\mathbf{x}$, and is zero iff $\mathbf{x}$ is optimal for all $w \in W$.

Minimax regret has been applied successfully to robust optimization given utility uncertainty in additive models [5] and generalized additive models [4, 6] for decision problems involving large-scale mixed integer programs (MIPs). While regret-optimization requires the solution of a minimax problem with a quadratic objective, the application of Benders' decomposition, constraint generation, and various reformulations renders the problem feasible, converting it to a (linear) MIP [4, 5]. We will adapt these techniques below.

Minimax regret has also proven to be effective as a means of utility elicitation [4, 5, 6]. One especially effective heuristic strategy is the *current solution strategy*, where preference queries are asked that involve the current minimax-optimal configuration $\mathbf{x}^*_W$ and/or the adversarial configuration (or *witness*). Unlike volumetric-based approaches to elicitation, regret-based elicitation reduces utility uncertainty only in the relevant regions of utility space, exploiting knowledge of which configurations are actually feasible. In contrast to Bayesian elicitation [3, 7], regret models require no prior over utility functions, nor expensive, approximate probabilistic inference.

## 3. Subjective Feature Elicitation

In many cases, some of the attributes over which a user forms her preferences will not coincide with system catalog features. We consider *subjective features* that are "objectively" definable using catalog attributes,

but where the definition varies from user to user.[2] For instance, the notion of a *safe* car may be different for a parent with small children, a young, single professional interested in high-performance vehicles, and a family that takes frequent trips to the mountains. The *concept* in question, namely, safety, has *personalized definitions*. The user also has *preferences* for safety (as she does for other car attributes) represented in the form of a utility function: it is both the user's utility function over this extended attribute space, as well as her personal definition of safety, that determines the optimal vehicle. As such, the decision support system must engage in both preference elicitation and *feature elicitation* to make a suitable recommendation.

This leads to interesting tradeoffs in elicitation. One could engage in feature elicitation using well-known concept learning techniques [1, 14], and then move to preference elicitation (e.g., using techniques mentioned above); but this could be wasteful. For instance, suppose we learn that safety requires attribute $X_i$ to be true (e.g., have side airbags) but know nothing else about the concept. If we engaged in preference elicitation simultaneously and ascertained that no cars in the user's price range satisfy $X_i$—or that other more important features must be sacrificed to attain $X_i$—then the full concept definition is not needed for optimal allocation. For similar reasons, full preference elicitation followed by feature elicitation may be undesirable. This suggests that *interleaved* feature and utility elicitation can be much more effective.

In this paper, we focus on the problem of feature elicitation in isolation, setting aside utility uncertainty. Apart from forming the foundation for work on joint utility and feature elicitation, this problem is interesting in its own right, as an extension of concept/query learning. In the remainder of this section, we describe a regret-based framework for *pure feature elicitation*, assuming a known utility function. We briefly discuss a more general framework that incorporates both utility and feature uncertainty in Sec. 5.

### 3.1. Feature Elicitation Model

We have features $\mathcal{X} = \{X_1, ...X_n\}$, which we take to be Boolean for ease of exposition, and a feasible set $\mathbf{X} \subseteq Dom(\mathcal{X})$. A known *reward* $r(\mathbf{x})$ for each $\mathbf{x} \in \mathbf{X}$ reflects user utility for $\mathbf{x}$ w.r.t. known product features. The user also has preference for configurations satisfying some target concept $c$. Concept $c$ is an unknown Boolean function over $\mathcal{X}$: $c(\mathbf{x}) = c(\mathbf{x}_1, \ldots, \mathbf{x}_n)$.[3] We suppose that $c$ is drawn from a particular function class/hypothesis space $H$. We treat identification of $c$ as a problem of concept learning [1, 12, 14], with some query set $Q$ that can be used to refine the target concept. For instance, membership queries would be quite natural ("do you consider the following car to be safe?").[4] Finally, a known *bonus* $p$ is associated with any $\mathbf{x}$ s.t. $c(\mathbf{x})$ holds, representing user utility for concept satisfaction.

For given a target concept $c$ and $\mathbf{x} \in \mathbf{X}$, define the utility of $\mathbf{x}$ under concept $c$ to be:

$$u(\mathbf{x}; c) = r(\mathbf{x}) + pc(\mathbf{x})$$

(We treat $c(\mathbf{x})$ as an indicator function for concept satisfaction). In other words, the utility of $\mathbf{x}$ is its reward, plus the bonus $p$ if $\mathbf{x}$ satisfies $c$. The optimal configuration is $\mathbf{x}^* = \arg\max u(\mathbf{x}; c)$.

### 3.2. Minimax Regret over Concepts

If we do not know the target concept, we cannot generally identify $\mathbf{x}^*$; but we can still make a decision with partial concept information. Let version space $V \subseteq H$ represent our current set of consistent hypotheses [18]. We define minimax regret w.r.t. feature uncertainty:

**Definition 2** *Given version space $V$, the* max regret *of $\mathbf{x} \in \mathbf{X}$, the* minimax regret *of $V$ and the* minimax optimal configuration *are:*

$$MR(\mathbf{x}; V) = \max_{c \in V} \max_{\mathbf{x}' \in \mathbf{X}} u(\mathbf{x}'; c) - u(\mathbf{x}; c) \quad (1)$$

$$MMR(V) = \min_{\mathbf{x} \in \mathbf{X}} MR(\mathbf{x}, V) \quad (2)$$

$$\mathbf{x}^*_V = \arg\min_{\mathbf{x} \in \mathbf{X}} MR(\mathbf{x}, V) \quad (3)$$

This provides a simple model of subjective feature uncertainty that abstracts away utility uncertainty. An adversary can cause us to regret recommendation $\mathbf{x}$ through suitable choice of concept $c \in V$. If $MMR(V) = 0$, then $\mathbf{x}^*_V$ is optimal for any realization of the concept $c \in V$.

We ask queries from query class $Q$ to refine knowledge of $c$ to a point where minimax regret $MMR(V)$ is reduced to some acceptable tolerance $\varepsilon$ (possibly zero). Our goal is very different from that of classical concept learning. We aim not to learn the concept $c$ itself, but simply learn enough about it to make a good decision. The reward model and feasibility constraints on $\mathbf{X}$ often allow us to make optimal recommendations with relatively weak concept knowledge.

[2] Other subjective features may not be so definable (e.g., visual features); for this, data-intensive collaborative filtering techniques are more appropriate.

[3] Allowing multivalued concepts is straightforward.

[4] Other query types (e.g., equivalence queries) are less natural in this domain, but may play a role in others.

We can characterize the minimax optimal solution in terms of the structure of the version space. We use the standard general-specific version space lattice [18]: concept $c$ is *at least as general as* $c'$ ($c \geq c'$) iff $c' \subseteq c$ (we treat a concept $c$ and its extension on $\mathbf{X}$ indistinguishable where no confusion arises). For any concept $c$, let $\mathbf{X}_c = \mathbf{X} \cap c$ be the set of feasible outcomes satisfying $c$. Define the *best outcome satisfying* $c$:

$$\mathbf{x}_c^* = \arg\max_{\mathbf{x} \in \mathbf{X}_c} r(\mathbf{x}); \text{ and } r_c^* = \max_{\mathbf{x} \in \mathbf{X}_c} r(\mathbf{x}).$$

Clearly $r_c^* \geq r_{c'}^*$ if $c \geq c'$. The measure $r_c^*$ induces a natural ordering $\geq_r$ on $V$ that respects the general-specific ordering:

**Observation 1** *If $c \geq c'$ then $r_c^* \geq r_{c'}^*$.*

We say $c^* \in V$ is *reward optimal* iff $r_{c^*}^* \geq r_c^*, \forall c \in V$. Define $\mathbf{X}^+ \subseteq \mathbf{X}$ to be the set of configurations with highest reward (independent of $V$) and $r^+ = r(\mathbf{X}^+)$. Finally, let $r_1 > r_2 \ldots > r_m$ be an ordering of the $m$ (distinct) values in $\{r_c^* : c \in V\}$. Define $C_i = \{c \in V : r_c^* = r_i\}$ and $S_i = \cap C_i$; i.e., $S_i$ is the set of configurations in the intersection of all concepts in $V$ with the $i$th-largest $r^*$ value ($S_i$ may be empty).

Let $MMR(V)$ be the current minimax regret level, $\mathbf{x}_V^*$ a minimax optimal solution, $\mathbf{x}^w$ the adversarial witness outcome, and $c^w$ the witness concept. Intuitively, minimax regret can be viewed as a game between a player choosing $\mathbf{x}^*$ and an adversary choosing $\mathbf{x}^w$ and $c^w$. It is not hard to see that if $\mathbf{x}^*$ is not consistent with $V$ (i.e., $\mathbf{x} \notin c$ for all $c \in V$), then $\mathbf{x}^* \in X^+$. So restrict attention to $V$-consistent configurations. Suppose the player chooses some $V$-consistent $\mathbf{x}^*$. If $\mathbf{x}^* \notin S_1$, the adversary can choose a $c^w \in C_1$ that excludes $\mathbf{x}^*$ and $\mathbf{x}^w = \mathbf{x}_{c_w}^*$, thus maximizing regret by taking a highly rewarding configuration, obtaining the bonus, and preventing the player from getting the bonus.

Regret will be strictly lower if the player chooses $\mathbf{x}^* \in S_1$ itself: for the adversary to get the bonus and prevent the player from getting it, it must choose a concept outside $C_1$ (hence a less rewarding $\mathbf{x}^w$). The player can further limit the adversary by ensuring $\mathbf{x}^* \in S_2$ (forcing the adversary to move to $C_3$ if it wants to get the bonus while the player does not). Of course, this comes at a price: the player gets a lower reward by moving to an $\mathbf{x}^* \in S_1 \cap S_2$ (if indeed this set is nonempty); in the game, this sacrifice is traded off against the benefit of further restricting the adversary. This informal line of reasoning can be formalized to prove:

**Proposition 1** *If $\mathbf{x}_V^*$ is not consistent with $V$, then $\mathbf{x}_V^* \in X^+$ (and all $\mathbf{x} \in X^+$ have identical max regret).*

**Proposition 2** *If $\mathbf{x}_V^* \notin X^+$, then: (a) $\mathbf{x}_V^*$ is consistent with $V$; (b) $\mathbf{x}_V^* \in \arg\max\{r(\mathbf{x}) : \mathbf{x} \in S_1 \cap \ldots \cap S_i\}$ for some $i \geq 1$; and (c) either $c^w \in C_1$, or $c^w \in C_{i+1}$.*

We also have:

**Observation 2** *$\mathbf{x}_V^* \in c^w$ only if $\mathbf{x}^w \in c^w$.*

**Observation 3** *If $r_{c^*}^* \leq r^+ - p$, then $MMR(V) = 0$ and $\mathbf{x}^* = \mathbf{x}^w = \mathbf{x}^+$.*

Thus if we have reduced $V$ so that no $V$-consistent $\mathbf{x} \in \mathbf{X}$ has reward within $p$ of $r^+$, then the (true) optimal choice does not satisfy the target concept.

### 3.3. Computing Regret: Conjunctive Concepts

We assume that the underlying configuration problem is represented as a MIP $\max_{\mathbf{x} \in \mathbf{X}} r(\mathbf{x})$. When subjective feature uncertainty exists, minimax regret computation can often be directly incorporated into the MIP for particular concept and query classes. We illustrate this in the case of (nonmonotone) conjunctive concepts with membership queries.

Assume target $c$ is a conjunction of literals over variables $X_j$. Memberships queries ask whether $\mathbf{x} \in c$ for a particular configuration. Let $E^+$ be the set of positive examples, $E^-$ the set of negative examples, and (nonempty) $V$ the induced version space. Instead of representing $V$ using most general and most specific concepts, we encode $E^+$ and $E^-$ directly in our MIP below (see, e.g., Hirsch [15] who uses negative examples to represent most general concepts).

**Constraint Generation:** We formulate the minimax problem Eq. 2 as a minimization with $O(|V|)$ constraints. Let $\mathbf{x}_c = \arg\max_{\mathbf{x} \in \mathbf{X}} u(\mathbf{x}; c)$ and constant $p(\mathbf{x}, c) = p$ if $c(\mathbf{x})$ and 0 otherwise. Let indicator variable $I^c$, for each $c \in V$, denote that configuration $(X_1, \cdots, X_n)$ satisfies $c$; and write $x^j \in c$ ($\overline{x}_j \in c$) to denote that variable $X_j$ occurs positively (negatively) in $c$. Then $MMR(V)$ is:

$$\begin{aligned}
&\min \quad \delta \\
&\text{s.t. } \delta \geq r(\mathbf{x}_c) - r(X_1, \cdots X_n) + p(\mathbf{x}_c, c) - pI^c \quad \forall c \in V \quad (4)\\
&\qquad I^c \leq X_j \quad \forall c \in V, \forall x_j \in c \quad (5)\\
&\qquad I^c \leq 1 - X_j \quad \forall c \in V, \forall \overline{x}_j \in c \quad (6)
\end{aligned}$$

For any *fixed* concept $c$, the adversary maximizes the regret of $(X_1, \cdots, X_n)$ with witness $\mathbf{x}_c$. The MIP above chooses a configuration that minimizes against the "worst-case" choice of the adversary (with (4) ensuring MMR is as great as regret given any $c \in V$; and (5, 6) encoding whether $(X_1, \cdots, X_n)$ satisfies $c$). Regret constraints for most $c \in V$ will be inactive, so we use constraint generation to search through the

space of adversarial concepts.[5] Let $Gen \subseteq V$; we solve a relaxed MIP using only $c \in Gen$. Let $\delta^*$ and $\mathbf{x}^*$ be the solution to the relaxed MIP. We test for violated constraints by solving the max regret problem $MR(\mathbf{x}^*, V)$, detailed below. If $MR(\mathbf{x}^*, V) > \delta^*$, concept $c'$ (produced as a witness in $MR$ computation) offers larger regret for $\mathbf{x}^*$ than any $c \in Gen$. So we add $c'$ to $Gen$ and resolve. If $MR(\mathbf{x}^*, V) = \delta^*$, $\mathbf{x}^*$ is the optimal solution to $MMR(V)$.

**Generating Violated Constraints:** We can compute the maximally violated constraint for MMR computation above by solving the max regret problem $MR(\mathbf{x}, V)$ for the current relaxed solution $\mathbf{x}^*$. This too can be formulated as a MIP. For each configuration variable $X_j$, let binary indicator variables $I(x_j)$ (resp. $I(\overline{x}_j)$) denote that $X_j$ is positive (resp. negative) in concept $c$. We also introduce binary variables $B^x$ and $B^w$ indicating that $\mathbf{x}$ and the witness allocation satisfy the chosen concept. Using $\mathbf{x}[j]$ to denote the $j$th literal of $\mathbf{x}$, this MIP gives $MR(\mathbf{x}; V)$:

$$\max \quad r(X_1, \cdots, X_n) - r(\mathbf{x}) + pB^w - pB^x$$

$$\text{s.t. } B^w + I(x_j) \leq X_j + 1.5 \quad \forall j \leq n \tag{7}$$

$$B^w + I(\overline{x}_j) \leq (1 - X_j) + 1.5 \quad \forall j \leq n \tag{8}$$

$$B^x \geq 1 - \sum_{j:\mathbf{x}[j] \text{ positive}} I(\overline{x}_j) - \sum_{j:\mathbf{x}[j] \text{ negative}} I(x_j) \tag{9}$$

$$\sum_j I(\neg \mathbf{y}[j]) = 0 \quad \forall \mathbf{y} \in E^+ \tag{10}$$

$$\sum_j I(\neg \mathbf{y}[j]) \geq 1 \quad \forall \mathbf{y} \in E^- \tag{11}$$

$$(X_1, \cdots, X_n) \in \mathbf{X} \tag{12}$$

Apart from the binary indicator variables $B^w, B^x, I(x_j), I(\overline{x}_j)$ described above, we have configuration variables $X_j$. (12) ensures that the configuration is feasible, and notation $r(X_1, \cdots, X_n)$ loosely denotes the reward of any configuration (which is encoded using a linear parameterization of utility). (7) and (8) ensure that the adversary does not get the concept bonus $p$ (i.e., cannot set $B^w = 1$) unless $(X_1, \cdots, X_n)$ satisfies the concept dictated by the $I$-variables. Similarly, (9) ensures that the input configuration $\mathbf{x}$ cannot be denied the bonus (i.e., the adversary cannot set $B^x = 0$) unless $\mathbf{x}$ violates at least one conjunct in the chosen concept. Finally, (10, 11) restrict the conjunctive concept to be consistent with all positive and negative examples.[6]

[5] $V$ is exponential in $|\mathcal{X}|$ with conjunctive concepts; constraint generation is even more important in other hypothesis spaces, since they can have doubly exponential size.

[6] The version space for conjunctions of literals can, of course, be encoded more succinctly, i.e., without constraints for each example.

### 3.4. Query Strategies

We now consider several heuristic query strategies that can quickly reduce $MMR(V)$. We focus on membership queries, since these are most natural in preference assessment. Query strategies from concept learning can be used directly to refine feature uncertainty. However, the motivation for many query strategies is very dependent on the learning model in question. For instance, the mistake-bound model [1] provides strategies for certain hypothesis classes, such as conjunctions of literals, which offer a polynomial mistake bound. However, these minimize mistakes in prediction, not queries: even a mistake-free prediction requires user concept feedback. Our desire to provide worst-case guarantees on performance without distributional information over potential user-defined features prevents us from adopting a PAC-model [12]. The most natural connection to our model is with work on exact concept learning that attempts to minimize queries [14]. However, results for these models tend to be rather weak. For instance, conjunctive concepts cannot be learned without exponentially many membership queries in the worst case [14].

A halving strategy can be adapted to our utility-based, choice-oriented concept learning model: we ask random memberships queries until a positive example is found (in the mistake-bound model negative predictions would be used); then queries are asked by negating literals one by one in the (unique) most specific conjunctive hypothesis. Once a positive example is found, this converges to the true conjunctive concept using a number of queries linear in $|\mathcal{X}|$. We need not identify the concept exactly however; we terminate once minimax regret reaches an acceptable level. We call this the *halving strategy*. Despite its exponential worst-case for *exact* concept learning, the requirement to simply reduce regret should give rise to better performance in our model.

One problem with halving is its lack of regard for reward or configuration feasibility: it wastes effort learning about parts of a concept definition that are irrelevant to making a good "choice." So we consider an alternative heuristic based on using information in the current solution: either the minimax optimal $\mathbf{x}^*_V$ or the witness $\mathbf{x}^w$ [4]. Intuitively, our best hope for (immediate) reduction in regret is to change the version space so that the status of at least one of $\mathbf{x}^*$ or $\mathbf{x}^w$ changes w.r.t. $V$. We define the *current solution strategy (CSS)* by considering three distinct cases for suggesting membership queries.

First consider the case where $\mathbf{x}^*, \mathbf{x}^w \in c^w$: we know $c^w \in C_1$ must be reward-optimal and $\mathbf{x}^w$ must be a

$V$-consistent configuration with greatest reward. CSS asks a membership query $\mathbf{x}^w$ (does $\mathbf{x}^w$ satisfy the concept), which has great potential value: if the answer is positive, then we know $\mathbf{x}^w$ is, in fact, optimal whatever the concept (and we reduce $MMR(V)$ to zero); and if the answer is negative, we rule out a reward-optimal concept $c^w$ from the version space, potentially reducing minimax regret, and strictly reducing max regret (ignoring ties in reward) for all $\mathbf{x} \notin c^w$. (Asking a membership query $\mathbf{x}^*$ has less impact in this case).

Second suppose $\mathbf{x}^* \notin c^w, \mathbf{x}^w \in c^w$: we either have $\mathbf{x}^* \in \mathbf{X}+$, in which case $c^w \in C_1$; or $\mathbf{x}^*$ is $V$-consistent, in which case $c^w \in C_{i+1}$ (where $\mathbf{x}^*$ is chosen from $S_1 \cap \ldots \cap S_i$). In this case, CSS asks a membership query $\mathbf{x}^*$. A positive response reduces pairwise max regret between $\mathbf{x}^*$ and $\mathbf{x}^w$ (and hence often reduces $MMR(V)$). A negative response does not reduce pairwise regret, but it rules out all concepts from the version space that include $\mathbf{x}^*$ and all concepts more general (if any) than $c^w$, giving us additional flexibility in the choice of allocation without losing value.

Finally, suppose $\mathbf{x}^*, \mathbf{x}^w \notin c^w$ (recall, we cannot have $\mathbf{x}^* \in c^w, \mathbf{x}^w \notin c^w$). If $\mathbf{x}^w$ is not $V$-consistent, we know $\mathbf{x}^w \in \mathbf{X}^+$, so CSS asks a membership query $\mathbf{x}^*$ (the rationale is as in case 2 above). If $\mathbf{x}^w$ is $V$-consistent, then $\mathbf{x}^w \notin c^w$ only if $\mathbf{x}^w \in c$ implies $\mathbf{x}^* \in c$, for all $c \in V$ (in which case, choosing a concept that satisfies $x^w$ has no impact on $MMR(V)$). In this case, CSS asks a membership query $x^w$ (as in case 1 above).[7]

## 4. Empirical Evaluation

We experimented with the two heuristic strategies, Halving and CSS, as well as Random, which asks random membership queries. Configuration problems with 30 Boolean variables were generated, each with random binary constraints to reflect the realistic assumption that the space of feasible products is relatively sparse (each constraint rules out (on average) half of the configurations). Rewards were generated using a linear utility function: each variable $X_i$ was randomly assigned a reward $r_i \sim U[0, 10]$ and a parity (if positive, $x_i$ gets reward $r_i$ and $\overline{x}_i$ reward 0; if negative, the opposite). Reward $r(\mathbf{x})$ is the sum of the variable rewards. The bonus $p$ for concept satisfaction was fixed at a certain percentage of the maximum reward. Conjunctive concepts were randomly drawn from a pool of ten variables, with each variable (independently) occurring in the concept positively (probability 0.25), negatively (0.25), or not at all (0.50). Minimax regret computation was reasonably fast (under 1 sec. using CPLEX 11 on a quad-core Intel machine).

[7] If CSS recommends $\mathbf{x}^*$ or $\mathbf{x}^w$ when membership (or nonmembership) is logically certain given $V$, the other query is asked. (*Both* can't be certain unless MMR is 0.)

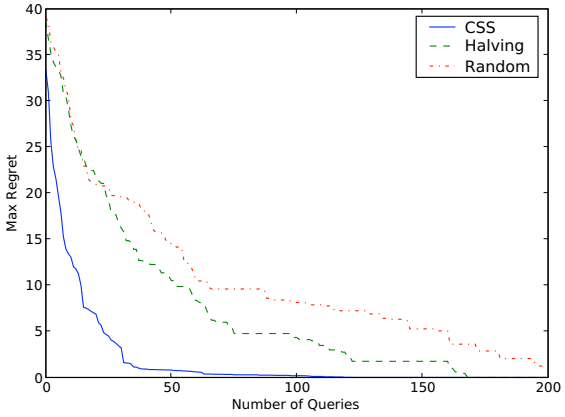


*Figure 1.* Max regret vs. number of queries (20 constraints, $p = 25\%$), 50 runs

We first illustrate the performance of our three main strategies using some typical parameter settings: 20 binary constraints and a bonus $p$ set at 25% of the maximum reward value. Fig. 1 shows reduction in MMR as a function of the number of membership queries by each strategy. The performance of CSS is considerably better than that of Halving. Key to the performance difference is the ability to exploit the current solution to identify which concepts have the greatest impact on performance given feasibility restrictions and reward values (in particular, $r^*$ values).

The relative density of feasible configuration space has a significant impact on the relative performance of Halving versus CSS. Intuitively, with a sparser solution space, fewer configurations satisfy a given concept $c$ (hence reducing $r^*_c$), and the value of considering most concepts is diminished. In such a case, CSS has an advantage over Halving. Conversely, with dense solution spaces, narrowing down the version space to get close to full concept identification becomes more important, making Halving potentially more attractive. Fig. 2 shows the number of membership queries needed to reduce minimax regret to 80% of its original level (prior to any queries) while varying the number of binary constraints. More constraints give a sparser solution space; and with 20 (or more) constraints, CSS has a significant advantage.

Similarly, the relative magnitude of the feature bonus $p$ to overall reward can have a dramatic impact on the number of required queries and the need to narrow down the feature definition more or less precisely. Intuitively, when $p$ is relatively small, it has a relatively small impact on utility and far fewer queries

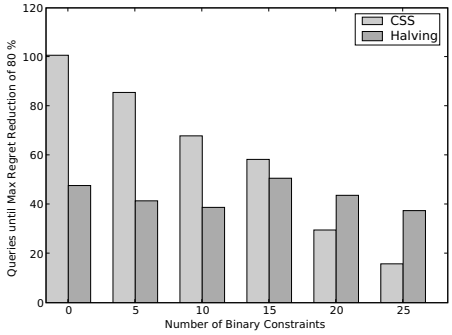


*Figure 2.* Sensitivity to number of binary constraints; bonus $p = 25\%$, 50 runs.

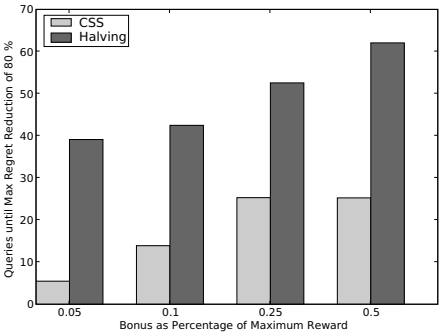


*Figure 3.* Sensitivity to value of bonus $p$ (20 random constraints), 50 runs.

will be needed to reduce regret to zero (since even a loose feature definition will lead to little error). As the magnitude of the bonus increases, a greater need for more precise concept constraints will emerge. Fig. 3 shows this effect. We show the number of membership queries needed to reduce minimax regret to 80% of its original level (prior to any queries), varying the relative size of the bonus from 5% to 50% of $r^+$. While an average of roughly five queries suffice for the CSS strategy at $p = 5\%$, about 25 queries are needed at $p = 50\%$. By way of contrast, Halving needs significantly more queries (from nearly 30 to over 60).

Finally, in certain elicitation settings, it is reasonable to assume that a user can provide a positive concept example (e.g., a "safe" car configuration). In the exact learning model, conjunctive concept identification is hampered by a potentially exponential number of membership queries with negative responses. Once a positive response is obtained, halving can be utilized. Fig. 4 shows a comparison of Halving in which an initial positive instance is assumed, with Halving (no initial seed) and CSS (with initial seed). We plot regret reduction as a function of the number of queries. The availability of a positive seed has a strong impact on regret, reducing the initial minimax regret to 15 from over 35. Halving is able to reduce regret to 0 in just 10

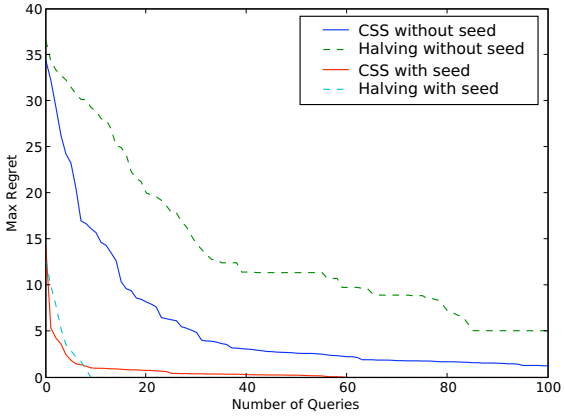


*Figure 4.* Max regret vs. number of queries with/without an initial positive seed (25 constraints, $p = 25\%$), 50 runs.

queries (cf. no initial seed, where it needs 100 queries to get to regret 5). The CSS strategy, dominant in the setting without seeding, still compares favorably to Halving if a positive example is available.

Overall, these results suggest minimax regret has great value as a solution concept for handling feature uncertainty, and can greatly reduce the number of queries needed to make a good decision relative to exact concept identification. In our setting, asymptotic performance takes a back seat to practical performance. Reducing the number of queries one asks of a user from, say, 30 to 20, can have a huge impact on the acceptance of a decision support system. Our framework allows one to make robust decisions with limited information.

## 5. Concluding Remarks

We have presented a model of subjective feature elicitation for use in preference elicitation. The use of minimax regret allows us to focus user queries on those aspects of a feature or concept that have the greatest impact on decision quality, thus reducing the effort required to learn the *relevant* parts of a concept.

**Related Work** Our work has connections to work in concept learning and active learning. Our model differs from the mistake-bound model in our need to minimize queries (not prediction error), while the PAC-model emphasis on probabilistic correctness w.r.t. a fixed data distribution renders it inapplicable to our setting.[8] Exact query learning [14] bears the closest

[8] Another difference is in the emphasis on computational tractability: we generally deal with underlying optimization problems, like configuration, that are inherently NP-hard. While the computation methods in this paper are very fast (all queries generated in well under a second),

link to our model; but ours differs from all concept models in the aim to learn just enough about a concept to make a good decision, not the entire concept. However, query strategies from these models (like halving) can be adapted to the regret setting as we have seen. Connections between concept learning and preference elicitation have been explored by [2], who deal with exact learning of a combinatorial valuation function. While their focus is on exact learning differs from ours, their valuation model should lend itself to approximation and regret analysis.

Our work can also be viewed as a form of active learning [8, 9, 11]. Indeed, the focus on regret reduction (and termination when regret reaches some $\varepsilon$) is a non-Bayesian analog of the value of information criterion that underlies much work on active learning. We assume a realizable setting in this work, and our membership query selection technique falls within the framework of [8] (we only query points on which two $c \in V$ disagree). While most techniques from active learning rely on probabilistic data models [9, 11], there is certainly potential to adapt active query strategies to our setting (where the data is not drawn randomly, but the set $\mathbf{X}$ is implicitly available). The key will be to bias queries in a way that accounts for the contribution of the reward function to error (minimax regret).

**Joint Preference and Feature Elicitation** Space precludes a full discussion, but the MIP approach above can be generalized to account for simultaneous uncertainty in both feature definition and utility (i.e., reward function and bonus $p$). We assume general linear constraints relating the utility of two outcomes (e.g., a user comparison of two products will tell us that the utility of one is greater than the other). A key complication, compared to standard elicitation models [4], is that we do not simply get linear constraints on utility parameters $w$. We must encode a set of "conditional constraints" that relate the difference in reward between the two products to whether either or both satisfy the unknown concept. These can be linearized and encoded in single MIP. We defer a full investigation to future work.

**Future Directions** A number of important directions for future research remain. Among these are exploring whether existing active learning models and query strategies can be adapted to our choice-based, regret model. We are currently investigating new query types and practical models for richer hypothesis spaces. Generalizing the form of catalog and subjective features to real-valued domains is of interest, as is investigation of the conceptually straightforward extension to discrete, non-Boolean features. Finally, extending our approach to non-additive utility models (e.g., adapting techniques for GAI models [6]) is essential.

our primary concern is minimizing user burden.

## References


[1] D. Angluin. Queries and concept learning. *Mach. Learn.*, 2(4):319–342, 1987.

[2] A. Blum, J. Jackson, T. Sandholm, and M. Zinkevich. Preference elicitation and query learning. *J. Mach. Learn. Res.*, 5:649–667, 2004.

[3] C. Boutilier. A POMDP formulation of preference elicitation problems. *AAAI-02*, 239–246, 2002.

[4] C. Boutilier, R. Patrascu, P. Poupart, and D. Schuurmans. Constraint-based optimization and utility elicitation using the minimax decision criterion. *Artif. Intelligence*, 170(8–9):686–713, 2006.

[5] C. Boutilier, T. Sandholm, and R. Shields. Eliciting bid taker non-price preferences in (combinatorial) auctions. *AAAI-04*, 204–211, 2004.

[6] D. Braziunas and C. Boutilier. Minimax regret-based elicitation of generalized additive utilities. *UAI-07*, 25–32, 2007.

[7] U. Chajewska, D. Koller, and R. Parr. Making rational decisions using adaptive utility elicitation. *AAAI-00*, 363–369, 2000.

[8] D. Cohn, L. Atlas, and R. Ladner. Improving generalization with active learning. *Mach. Learn.*, 25:201–221, 1994.

[9] S. Dasgupta, D. Hsu, and C. Monteleoni. A general agnostic active learning algorithm. *NIPS 20*, 353–360, 2008.

[10] P. Fishburn. Interdependence and additivity in multivariate, unidimensional expected utility theory. *Intl. Econ. Rev.*, 8:335–342, 1967.

[11] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Mach. Learn.*, 28:133–168, 1997.

[12] D. Haussler. Quantifying inductive bias: Ai learning algorithms and valiant's learning framework. *Artif. Intel.*, 36(2):177–221, 1988.

[13] D. Haussler. Learning conjunctive concepts in structural domains. *Mach. Learn.*, 4:7–40, 1989.

[14] L. Hellerstein, K. Pillaipakkamnatt, V. Raghavan, and D. Wilkins. How many queries are needed to learn? *JACM*, 43(5):840–862, 1996.

[15] H. Hirsh. Polynomial-time learning with version spaces. *AAAI-92*, 117–122, 1992.

[16] R. Keeney, H. Raiffa. *Decisions with Multiple Objectives: Preferences and Value Trade-offs*. Wiley, 1976.

[17] P. Kouvelis and G. Yu. *Robust Discrete Optimization and Its Applications*. Kluwer, Dordrecht, 1997.

[18] T. Mitchell. Version spaces: A candidate elimination approach to rule learning. *IJCAI-77*, 305–310, 1977.

[19] A. Salo and R. Hämäläinen. Preference ratios in multiattribute evaluation (PRIME)–elicitation and decision procedures under incomplete information. *IEEE Trans. Sys., Man, Cyber.*, 31(6):533–545, 2001.

[20] L. Savage. *The Foundations of Statistics*. Wiley, 1954.