# Computational Decision Support: Regret-based Models for Optimization and Preference Elicitation

Craig Boutilier

## 1 Introduction

Decision making is a fundamental human, organizational, and societal activity, involving several key (and sometimes implicit) steps: the formulation of a set of options or decisions; information gathering to help assess the outcomes of these decisions and their likelihood; some assessment of the relative utility or desirability of the possible outcomes; and an assessment of the tradeoffs involved to determine an appropriate course of action. Advances in information and communication technology have changed the nature of the decisions that face individuals, professionals, and organizations. Sophisticated devices, ubiquitous sensors, and the increasing use of online mechanisms to mediate personal, professional, and business communications and transactions has resulted in an explosion in the data available to support decision making. Apart from data proliferation, the range and complexity of the options facing decision makers has also increased: networked communication has drastically reduced search costs, and sophisticated computational models have greatly expanded the capacity for reasoning about complex decision structures (e.g., policies, configurations of decision variables, etc.).

Computer-aided decision support is vital for several reasons. First, computational methods are critical in helping decision makers wade through of huge volumes of data to extract relevant information. Tremendous strides in information retrieval and machine learning have offered ever-more sophisticated algorithms for data mining and relevance detection. Second, while computers have long been used for decision support, the new richness of data sources and complexity of decision spaces bring with them demands for more sophisticated forms of algorithmic optimization to help decision makers determine suitable courses of action. However, even with sophisticated algorithms at their disposal, decision support tools require considerable information about the decision maker's preferences to function effectively. This is especially true when we consider putting decision support in the hands of the "masses," that is, people who make decisions without having (or needing) a rich understanding of the underlying domain or its dynamics. Many ingredients of a decision scenario can be *fixed*—options, likelihood of outcomes, relevance of information sources, etc., may be the same for each user of a decision support tool. *What varies are the preferences of the users*, each of whom has different goals or objectives in mind. Hence, this information cannot be coded into the system in advance. Unfortunately, no decision support system can recommend decisions without some idea of what these preferences are, giving rise to *the preference bottleneck*: how do we get the preferences of the user (or organization) "into" the decision support system?

The preference bottleneck is one of the greatest impediments to the wide-scale deployment of

decision support tools for "everyday decision making," and a pain point even for sophisticated decision makers. Addressing it satisfactorily requires computational insight, but also relies on the study of human preferences and decision making in behavioral decision theory, behavioral economics, psychology, and a host of other disciplines.

In this article, I briefly overview a specific set of techniques for preference elicitation and robust optimization especially suited to decision problems with large, complex decision spaces. These methods exploit an intuitive concept known as *minimax regret* (see Sec. 3) for two purposes. First, minimax regret is used by the decision support system to recommend decisions when it has incomplete information about the decision maker's true preferences. Indeed, it provides a form of *robust optimization* that offers guarantees on the quality of its decision—how far it can be from optimal—no matter what the true preferences turn out to be. Second, minimax regret can drive the *preference elicitation* process. The solution of the minimax optimization problem can be used to determine which unknown preference information can most improve the quality of the decision. Critically, these techniques do not attempt to reduce preference uncertainty for its own sake; rather they exploit the rich structure of the decision space to focus attention on "relevant" preference information. As such, regret-based optimization can serve as the core of truly interactive decision-support systems.

While I emphasize computational aspects of regret-based optimization and elicitation, the successful adoption of these techniques will require insight from the social, behavioral and cognitive sciences to shape the interaction with decision makers. I conclude the article with a brief discussion of the types of research that are most likely to influence the design of "everyday" interactive decision support tools.

The remainder of the article is organized as follows. Section 2 provides an overview of *multiattribute decision theory*, the framework used throughout. I outline several domains, each with different characteristics, for which preference assessment is complex and burdensome. Section 3 introduces minimax regret as a means of recommending decisions when only partial information is known about the decision maker's preferences, and briefly discusses computation of minimax regret. In Section 4, I describe how minimax regret can be used to determine relevant queries about decision maker preferences. Finally, in Section 5, I make a few remarks regarding the potential for research in the social, behavioral and cognitive sciences to influence next-generation computational decision support systems.

## 2 Basic Formulation and Some Domains

I begin with an informal overview of the framework used throughout the article, that of *multiattribute utility theory (MAUT)* (Keeney and Raiffa, 1976). I outline several domains where the size and complexity of the decision space not only demands software-based decision support, but requires that the preference bottleneck be addressed.

### 2.1 Utility Functions

We assume a decision maker (DM) faced with a set of *options* or *decisions* $D$, which we take to be finite for ease of exposition. A software-based *decision support system (DSS)* helps DM navigate the

We assume a (finite) set of *decisions* $D$; a set $X$ of *attributes* $X_1, X_2, \ldots, X_n$ each with finite domains; a set of *outcomes* $\mathbf{X} = X_1 \times \cdots \times X_n$, consisting of all combinations of attribute values. We denote by $\mathbf{x}$ an arbitrary outcome (combination of attribute values), and use $x_i$ to denote the value of attribute $X_i$ in $\mathbf{x}$. Each decision $d \in D$ gives rise to a probability distribution $P_d$ over the outcome set: $P_d(\mathbf{x})$ denotes the probability that outcome $\mathbf{x}$ will occur if decision $d$ is taken.
Decision maker preferences are represented by a *utility function* $u : \mathbf{X} \mapsto \mathbb{R}$, where $u(\mathbf{x})$ denotes the utility of $\mathbf{x}$. Given these ingredients, an *optimal decision* from $D$ is any $d^*$ with maximum expected utility:

$$d^* = \arg\max_{d \in D} \sum_{\mathbf{x}} P_d(\mathbf{x}) u(\mathbf{x}).$$

If the utility function $u$ satisfies *additive independence* (Keeney and Raiffa, 1976), it can be written as a sum of single-attribute *subutility functions*:

$$u_A(\mathbf{x}) = u_1(x_1) + u_2(x_2) + \ldots + u_n(x_n). \tag{1}$$

Each subutility $u_i : X_i \mapsto \mathbb{R}$ reflects the contribution of $X_i$ to the overall utility of an outcome.
Let $S_1, \ldots, S_m$ be a collection of subsets of attributes, with each attribute $X_i$ in at least one subset. If $u$ satisfies *generalized additive independence (GAI)* with respect to this decomposition, it can be written as a sum of real-valued functions $u_j$ over subsets $S_j$ (Fishburn, 1967):

$$u_{GAI}(\mathbf{x}) = u_1(\mathbf{x}_1) + \ldots + u_m(\mathbf{x}_m).$$

Here $\mathbf{x}_j$ refers to the restriction of outcome $\mathbf{x}$ to the attributes in $S_j$.

**Box 1: Multiattribute Decision Problems**

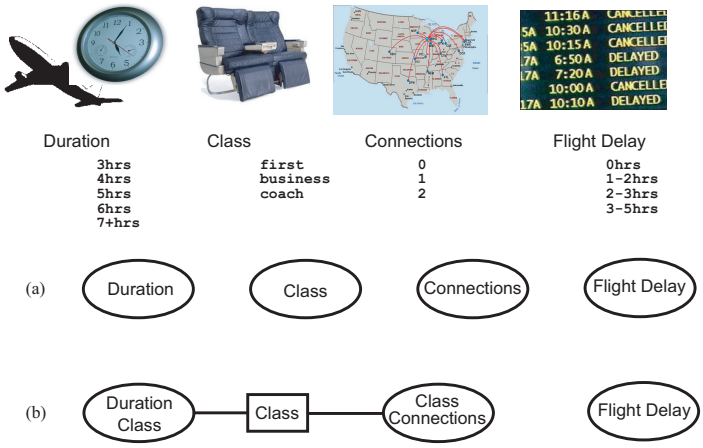



Figure 1: A small example to illustrate a multiattribute utility model (Braziunas and Boutilier, 2008). A graphical representation of the (a) additive and (b) GAI model from our four-attribute example.

space of options, potentially making or recommending specific decisions for DM. We take the task of the DSS in what follows to be the recommendation of a specific option for DM.

The main ingredients of a basic decision scenario are outlined in **Box 1**. I illustrate the concepts with a simple example (taken from Braziunas and Boutilier, 2008). Suppose DM must choose a flight from Toronto to Seattle: the decision set $D$ is the set of all flights on a specific day. The possible *outcomes* of a flight choice are given by a set of *attributes* of interest to DM, for example, flight Class (which takes values `first,business,coach`), number of Connections (`0,1,2`), total flight Duration in hours (`3,4,5,6,7+`), and flight Delay in hours (`<1,1-2,2-3,3-5`). This gives us a total of $5 \cdot 3 \cdot 3 \cdot 4 = 180$ outcomes or combinations of attribute values (see Fig. 1(a)). Any flight $d$ uniquely determines the first three attributes, while the fourth attribute is stochastic, with the probability of each outcome determined by relevant flight statistics. Other deterministic (e.g., cost, airline, nominal arrival time, etc.) and stochastic (e.g., lost luggage, cancellation, actual arrival time) attributes may be included as well.

The preferences of DM over these outcomes are represented by a *utility function* $u$, which reflects strength of preference. The best flight for DM is that which maximizes her expected utility. There are several reasons to use quantitative utility functions rather than a simple ordinal ranking of outcomes. First, we usually trade off the relative desirability of flights with their price: if Flight $A$ is preferred to $B$ (independent of price), but $B$ is cheaper than $A$, some knowledge of the *degree* to which $A$ is preferred is needed for a DSS to recommend a flight. Second, when outcomes are stochastic, tradeoffs involving strength of preference are needed. For example, suppose $A$ and $B$ differ only on probability of late arrival: $A$ will almost certainly be on time, while $B$ has a 25% chance of being 30 minutes late and a 10% chance of arriving 30 minutes early. Without assessing strength of preference for arrival time, the tradeoff between $A$ and $B$ cannot be made. Finally, our utility elicitation schemes below will not pin down DM utilities with complete precision—quantitative utilities support error/quality estimates when making decisions without precise utilities in a way that is not possible with qualitative preferences.

Computer-aided decision support is vital, even in situations as simple as this one. First, the decision space is large: all flight plans, including those with connections must potentially be considered. In domains discussed below, the decision space exhibits considerable *complexity*, not just size, and it is impossible for the DSS to even enumerate the space of options explicitly. It is imperative that software be used to explore, evaluate and compare options. Second, the "objective" data in the problem is often unknown to DM, and evaluating the expected utility of a decision again requires software support. In our flight example, relevant data would include on-time arrival statistics.

In many applications, specifying the utility $u(\mathbf{x})$ of each outcome $\mathbf{x} \in \mathbf{X}$ is infeasible since the outcome space grows exponentially with the number of attributes. Fortunately, preferences usually possess significant structure. Models that exploit this structure allow one to specify and represent $u$ concisely. Among the most common assumptions in practical systems is that of *additive independence* (Keeney and Raiffa, 1976). In our example, an additive utility function requires the specification of separate subutility functions for each of the attributes Dur, Class, Conn, and Delay, with the utility of any outcome given by their sum, e.g.,

$$u_A(\texttt{4hrs}, \texttt{coach}, \texttt{0}, \texttt{1-2hr}) = u_{Dur}(\texttt{4hrs}) + u_{Class}(\texttt{coach}) + u_{Conn}(\texttt{0}) + u_{Delay}(\texttt{1-2hr})$$

(see Fig. 1(a)). Thus rather than specifying utility values for all 180 outcomes, we specify only subutilities for each of the five values of Dur, three for Class, three for Conn, and four for Delay, for a total of 15 values. In other words, the additive model is specified using 15 *parameters* instead of the 180 required if the model has no structure.

Additive models, while popular in practice, are restrictive because of the assumption of attribute independence. In our flight example, for many users strength of preference for Class will depend on Dur (with preference for Class=`first` increasing with flight duration). This dependence cannot be captured by an additive model. The *generalized additive independence (GAI)* model (Fishburn, 1967; Bacchus and Grove, 1995; Boutilier et al., 2001; Gonzales and Perny, 2004; Boutilier et al., 2006; Braziunas and Boutilier, 2005) is much more flexible, ranging from "flat" utility functions with no structure to fully additive utility functions. Most realistic problems fall somewhere between these extremes. We can capture the fact that preference for Class depends on both Dur and Conn using a GAI model with three factors (or subsets), as illustrated in Fig. 1(b), where the utility of the outcome $(\texttt{5hrs}, \texttt{first}, \texttt{1}, \texttt{<1hr})$ would decompose as:

$$u_{GAI}(\texttt{5hrs}, \texttt{first}, \texttt{1}, \texttt{<1hr}) = u_{Dur,Class}(\texttt{5hrs}, \texttt{first}) + u_{Class,Conn}(\texttt{first}, \texttt{1}) + u_{Delay}(\texttt{<1hr}).$$

One can view these three subsets, or *factors*, as a grouping together preferentially dependent attributes.

While DSSs can help DM navigate the space of options, and incorporate diverse sources of information, making a decision still requires knowledge of DM's preferences, i.e., her utility function $u$. Is a flight that is $200 cheaper than another, but has an extra connection and increased odds of delay, the appropriate recommendation? It may be for user $Z$, but not user $Y$. Since $u$ encodes features of the decision problem that are unique to DM, any DSS must address the preference bottleneck. We examine this issue in the next two sections. But before moving on, we describe several additional domains where the size and complexity of the decision space demand the use of decision-support software.

## 2.2 Product Configuration

Our example of selecting flights illustrates a general class of *product configuration* problems. Consider consumer choice of products such as travel packages, automobiles, financial instruments, real estate, feature-laden consumer electronics, etc. The space of options facing consumers is often bewildering. The richness of attributes that influences choices increases constantly as well: on-time flight statistics to help choose flights; school ratings and crime statistics in real estate choice, etc.

Often, the modeling of uncertainty is ignored. Instead, one equates the decision space and the outcome space (e.g., if I buy a house, the outcome is known) and preferences are articulated directly over (multiattribute) choices. Since not all attribute combinations $\mathbf{X}$ are possible, we limit choices to a *feasible set* $\mathbf{X}_f$. For instance, the feasible set might correspond to the houses on the market in a particular city, so $\mathbf{X}_f$ is represented by a real estate *database*. In other cases, the set $\mathbf{X}_f$ is represented implicitly with a set of *constraints* on the combinations of attributes that are "allowed." For instance, in the automobile domain, a constraint might state that "Any car with engine power over 240 hp. must have fuel economy of under 24 mpg." Optimization can be computationally demanding: given a set of constraints and a *known utility function*, determining the optimal outcome is *NP-hard* (Dechter, 2003), which informally means that it is widely believed that computation time grows exponentially, in the worst case, with the size of the problem (the number of variables/attributes and the domain sizes of each). This is problematic since we're often interested in configuration problems with dozens or hundreds of variables, each with large numbers of possible values. Despite this, state-of-the-art constraint processing methods and optimization technologies (such as integer programming) make these problems solvable in practice.

Consumer DSSs have access to the same objective data for multiple DMs, but each DM will have different preferences. Moreover, users will have a hard time quantifying strength of preference and the required tradeoffs explicitly. Finally, given the number of parameters required to specify a utility function and the numerical precision required by classical elicitation methods, users will not have the patience to fully articulate their utility functions, even if they were able to do so. This means assessment of DM preferences must differ from the textbook approaches in decision analysis.

## 2.3 Corporate Sourcing

A different setting is that of *expressive auctions for corporate sourcing* (Sandholm, 2007). Here the decision maker is a company that is sourcing raw materials, supplies, or services from a collection of suppliers. For instance, Company A requires supply contracts for the coming year to run its manufacturing facilities, and has requirements on the volume, quality, and delivery schedule of different items needed to maintain production. Traditionally, Company A would run a *reverse auction*, allowing suppliers to bid on the items it needs, and accepting the bids of the lowest cost suppliers for each item. Such methods usually lead to economic inefficiencies. Consider a case where Supplier Z can produce and sell item $x_1$ to A for \$5 per unit in isolation, and item $x_2$ for \$7 per unit in isolation; but production synergies allow both items to be produced *jointly* and sold for \$9. If Z is forced to bid on each item separately, the joint cost to A is \$12 per unit-pair if it accepts Z's bids, even though Z would be willing to sell *the pair* for less. Not only A can end up overpaying; but should two different suppliers offer $x_1$ and $x_2$ for \$4 and \$6, respectively, A will not only pay more (\$10) by selecting

these suppliers over Z, but Z would have gladly been willing to supply the pair of items for less per unit ($9): economic efficiency has suffered.

*Combinatorial auctions (CAs)* generalize traditional market mechanisms by allowing the direct specification of bids over *bundles* of items (Cramton et al., 2005). For example, by allowing Z to explicitly offer the pair $(x_1, x_2)$ for $9, both A and Z benefit, as does economic efficiency. Other types of side information can also be incorporated in supplier bids. While offering greater economic efficiency, the problem of *winner determination*, namely, determining the set of winning bids so that A gets its required supplies for minimal cost, is computationally difficult (in fact, NP-hard, see above) (Rothkopf et al., 1998), though these problems are now practically solvable. We refer to the set of winning bids, or allocation of business to suppliers, as an *allocation*.

While CAs have great potential, buyers are interested in more than just minimizing total sourcing cost. Often they sacrifice price to improve other *attributes of the allocation* (Boutilier et al., 2004); in other words, the cost-minimizing allocation is often not the most preferred. For example, Company A may prefer, all else being equal, an allocation in which fewer suppliers are awarded business (e.g., to minimize logistical overhead). Generally, buyers will be concerned with a host of factors that can be traded off against cost. The *optimal allocation* is the one that strikes the appropriate tradeoff between these attributes and total cost.

Recasting the decision problem facing Company A in this light, the set of decisions $D$ becomes the set of *feasible allocations* of business to suppliers that meet the buyer's demands. This decision space is incredibly complex, and cannot be explicitly articulated. Simply determining the feasibility of a single "potential" decision is computationally difficult. The potential outcomes $\mathbf{X}$ are defined by the allocation attributes (of which cost is just one); and the feasible outcomes $\mathbf{X}_f$ are those induced by some feasible allocation. DM's utility function is expressed over the same attributes. While software decision support is vital here, several unique aspects of this problem include: the sheer complexity and size of the decision space; the computational difficulty of optimization; and the fact that DM will often not be able to articulate some relevant attributes until outcomes are revealed that call attention to those attributes.

## 2.4 Markov Decision Processes

*Markov decision processes (MDPs)* are used to model sequential decision problems under uncertainty in economics, operations research, computer science, and many other areas (Puterman, 1994; Boutilier et al., 1999). Unlike the problems above, where a *single* decision is made, in an MDP one determines a *sequence of decisions/actions* to guide some stochastic system into certain desirable states. Specifically, as various actions are taken, the system moves (stochastically) through various states, which can be more or less desirable. A *reward function* $R$ represents this desirability, where $R(s)$ can be viewed as the "immediate utility" for being in state $s$. However, the true value of being in state $s$ is not simply its immediate reward $R(s)$. The sequential nature of the decision problem means that we must also account for *future opportunities* made available by being in state $s$. For instance, the immediate reward for making some investment (e.g., in education, capital equipment, or practicing violin) may be negative; but that investment opens up future opportunities that may be very rewarding if the right course of action is adopted. Navigating a system in this way requires the appropriate choice of *policy* $\pi$ which dictates which action to take at any system state. An *optimal policy* is one which maximizes

the expected sum of rewards accrued, and can be computed using a variety of dynamic and linear programming methods (Puterman, 1994; Boutilier et al., 1999).

MDPs are routinely used in stochastic control, inventory control, production planning, mobile robotics, and other applications. We illustrate the difficulties of preference assessment using an application in the area of cognitive assistive technologies. COACH (Boger et al., 2005, 2006) is a system designed to help older persons with moderate-to-severe dementia (e.g., Alzheimer's disease) complete routine activities of daily living, such as handwashing. This system monitors the behavior of the user (patient) using computer vision technology. If he seems to be unsuccessful at some point during the task (e.g., confused or forgetful), the system decides between various actions: waiting to see if the user makes progress independently; prompting/reminding the user (in various ways) about the next step; or calling a caregiver (professional or family member) to assist the user. Systems like these can be used to relieve caregivers of the burden of constantly monitoring routine daily activities (Mihailidis et al., 2001).

In COACH, the entire process is modeled as an MDP. While system dynamics (i.e., probabilities of successful completion of task steps) can be learned from data, the key difficulty lies in eliciting the reward function. Rewards in the COACH system represent the preferences of the user or caregiver for actions and outcomes, and one must assess tradeoffs (strength of preference) for features like: successful task completion; successful independent completion of task substeps; prompts of various forms (negative reward, since independent task completion is desirable); calling the caregiver (also negative, since successful completion without caregiver intervention is desirable); time to completion; and other factors. The optimal policy can be very sensitive to the precise reward given to each of these factors; yet caregivers have a hard time quantifying these tradeoffs.

## 2.5 Group Decision Making

Another area in which DSSs play an important role is in group decision support. While DSSs can play many different roles, here we focus on the question of *preference aggregation*: in any group decision scenario, individual stakeholders have preferences over alternatives (outcomes, policies, products, etc.); and the DSS must aggregate those preferences to determine a good group decision, reflecting some notion of consensus, compromise, or group utility. *Social choice* is a branch of social science that has studied the problem of preference aggregation for decades (Arrow et al., 2002), often by considering various forms of *voting rules*. Increasingly, computational technologies make it viable to use such concepts to support group decisions in routine, low-stakes decisions, leading to the burgeoning study of computational social choice (Chevaleyre et al., 2007).

We outline a common formal framework for social choice in **Box 2**. We refer to our users as "voters." The alternatives represent any outcome space over which the voters have preferences (e.g., product configurations, restaurant dishes, candidates for office, public projects, etc.) and for which a single collective choice must be made. Voter preferences, represented as a rankings over alternatives, are aggregated by *voting rules*, which embody some form of consensus or compromise. Dozens of families of voting rules have been studied, with two illustrated in Box 2. *Plurality* asks voters to state only their most preferred alternative, which gives it an "elicitation advantage;" but it also means that it fails to account for relative voter preferences for any alternative other than its top choice. Other

We assume a set of decision makers (or *voters*) $N = \{1, \ldots, n_v\}$ and a set of *alternatives* $A = \{a_1, \ldots, a_m\}$. Each voter $k$ has preferences over the alternatives in $A$, represented by a *ranking* (or linear ordering). Let $v_k(a)$ denote the *rank* of $a$ in $v_k$. Then $k$ prefers $a_i$ to $a_j$, written $a_i \succ_k a_j$, if $v_k(a_i) < v_k(a_j)$. For example, if we have three restaurants, $d, d', d''$, and voter $k$ likes $d$ better than $d'$, and $d'$ better than $d''$, then we write: $v_k(d) = 1, v_k(d') = 2, v_k(d'') = 3$; and $d \succ_k d' \succ_k d''$. A collection of rankings or "votes" $\mathbf{v} = \langle v_1, \ldots, v_n \rangle$ is a *preference profile*. For example, four diners trying to decide among the three restaurants might have the following preference profile:

$$v_1 : d \succ_1 d'' \succ_1 d' \qquad v_2 : d' \succ_2 d \succ_2 d''$$
$$v_3 : d' \succ_3 d \succ_3 d'' \qquad v_4 : d'' \succ_4 d \succ_4 d'$$

A *voting rule* $r$ takes as input a profile $\mathbf{v}$ and outputs a winner $r(\mathbf{v}) \in A$. *Plurality* is one of the most commonly used rules: the alternative with the greatest number of "first place votes" wins. In the example, the plurality score (number of first place votes) for each restaurant is: $s(d') = 2; s(d) = 1; s(d'') = 1$. Hence plurality would recommend $d'$ for this group. The *Borda rule* assigns $m - i$ points to alternative $a$ each time it occurs in position $i$ in some voter ranking: in our three-alternative example, each first-place vote gives $a$ 2 points, second-place 1 point, and third-place zero points. The winner is the alternative with the highest total score. The Borda scores $s_\beta$ of the three restaurants in our example is $s_\beta(d, \mathbf{v}) = 5$, $s_\beta(d', \mathbf{v}) = 4$, $s_\beta(d'', \mathbf{v}) = 3$; hence the Borda rules recommends $d$.

**Box 2: Voting Rules**

schemes produce winners that are much more sensitive to the range of relative preferences, as the Borda rule illustrates.

One obstacle to the use of schemes that require voters to specify full rankings is the informational and cognitive burden they impose on voters. Elicitation of sufficient, but still *partial* information about voter rankings could alleviate some of these concerns. Furthermore, the use of voting rules in lower stakes domains virtually *demands* decision making with partial rankings, since the effort required for users to compare alternatives they know little about, or care little for, will not be worth the impact it has on the final decision. We discuss decision making with partial rankings in the next section.

An important consideration in group decision making is the question of incentives and manipulation: unlike in the case of a single DM, DMs in a group setting may have an incentive to misreport their preferences. In our restaurant example, under the plurality rule, voter 4 might be better off voting for $d$ rather than $d''$: since he prefers $d$ to the winner $d'$, by voting for $d$ he would improve the plurality score for $d$ to 2, tying it with $d'$ (and depending on the tie-breaking protocol, give $d$ a chance to win). The celebrated Gibbard-Satterthwaite theorem (Gibbard, 1973; Satterthwaite, 1975) demonstrates the theoretical impossibility of avoiding such manipulation in general. However, when one considers more market-oriented mechanisms for making group decisions—that is, mechanisms that allow the transfer of payments between participants—*strategyproof mechanisms*—that is, mechanisms in which all parties are incentivized to reveal their preferences truthfully—can be constructed for certain domains. The well-known Vickrey-Clarke-Groves (VCG) mechanism, for instance, defines a set of payments between individuals that ensures truthtelling and a socially optimal outcome (Vickrey, 1961; Groves, 1973; Clarke, 1971). Once again, the implementation of this mechanism requires that all agents fully

reveal their utilities to the system.

# 3 Robust Optimization using Minimax Regret

We've seen above how the preference bottleneck manifests itself in several different settings. There are a number of ways of assessing, eliciting or otherwise estimating DM preferences and utility functions (see next section). However, we take for granted that any reasonable form of preference assessment is going to leave tremendous uncertainty in the DSS's knowledge of the DM's utility function. DM utility functions have a large number of parameters, and full knowledge will require the specification of *each* of these with a high degree of (numerical) precision, placing a great burden on DM. In addition, we will see that very good, even optimal, decisions can be computed even when little utility information is available, as long as it is *relevant* information. In this section, we discuss *robust* decision making in the presence of utility function uncertainty using *minimax regret*.

## 3.1 Utility Function Uncertainty

Assume a class of utility functions $\mathcal{U}$ over some outcome set $\mathbf{X}$, possibly parameterized in some compact way (e.g., $\mathcal{U}$ may be the set of additive utility functions over attribute set $X_1, \cdots, X_n$). In what follows, we abuse terminology by equating a utility function $u$ with its parameter vector.

Suppose our DSS has partial knowledge of DM's utility function $u$. We take this knowledge to be in the form of a subset $U \subseteq \mathcal{U}$, dubbed the *feasible set*. Intuitively, the DSS knows DM's utility function $u$ lies within $U$, and nothing more. Generally, $U$ will be determined by a set of constraints acquired through some elicitation or preference assessment process.

Now suppose the DSS must recommend a decision. Maximizing expected utility relative to $u$ is not possible if $u$ is unknown. A variety of techniques have been proposed for decision making in this context. Bayesian methods quantify uncertainty about preferences probabilistically, using a prior density over $\mathcal{U}$, conditioning on the acquired knowledge, and calculating the utility of any alternative $a \in A$ by taking expectation over $\mathcal{U}$ (Weber, 1987; Chajewska et al., 2000; Boutilier, 2002; Holloway and White, 2003). Other methods are inspired by similar probabilistic intuitions (e.g., by considering the uniform distribution over the space $U$), but are non-Bayesian in their recommendations (Toubia et al., 2003, 2004; Abbas, 2004; Iyengar et al., 2001). Other methods simply attempt to identify Pareto optimal options (i.e., those that are optimal for some feasible utility function) without making a specific recommendation (White et al., 1984; Sykes and White, 1991).

We instead use the notion of *minimax regret*, a concept first described by Savage (1951) in the context of uncertainty over world states, and since advocated for robust decision making with partial utility functions (Boutilier et al., 2001; Salo and Hämäläinen, 2001; Boutilier et al., 2006). The formal definition of minimax regret is given in **Box 3**. The approach requires that the DSS recommend the option $\mathbf{x}^*$ that minimizes maximum regret, essentially minimizing the worst-case loss for DM relative to all possible realizations of her utility function (consistent with $u \in U$).

We illustrate the concept with a small air travel example (Braziunas and Boutilier, 2008), with three flights and two possible DM utility functions (cast in monetary terms for simplicity):[1]

[1] In general, the feasible set $U$ will be a polytope with infinitely many possibilities, not a discrete set.

The *pairwise max regret* of choosing $\mathbf{x}$ instead of $\mathbf{x}'$ given feasible utility set $U$ is

$$R(\mathbf{x}, \mathbf{x}', U) = \max_{u \in U} u(\mathbf{x}') - u(\mathbf{x}).$$

This defines how much worse alternative $\mathbf{x}$ could be than an alternative $\mathbf{x}'$. The *maximum regret* of choosing outcome $\mathbf{x}$ is

$$MR(\mathbf{x}, U) = \max_{\mathbf{x}' \in \mathbf{X}} R(\mathbf{x}, \mathbf{x}', U).$$

This specifies how far $\mathbf{x}$ could be from optimal for DM. The *minimax optimal decision* is the outcome that minimizes max regret:

$$\mathbf{x}^* = \arg\min_{\mathbf{x} \in \mathbf{X}} \mathit{MR}(\mathbf{x}, U).$$

Define $\mathit{MMR}(U) = \min_{\mathbf{x} \in \mathbf{X}} \mathit{MR}(\mathbf{x}, U)$ to be the minimax regret level of feasible utility set $U$.

**Box 3: Minimax Regret**

| | Flight $A$ | Flight $B$ | Flight $C$ |
|---|---|---|---|
| $u_1$ | \$400 | \$200 | \$340 |
| $u_2$ | \$350 | \$650 | \$600 |

The max regret of choosing Flight A is \$300: if we recommend A, the user may in fact have utility $u_2$, in which case A is \$300 worse than B (the optimal flight given $u_2$). Similarly, Flight B has a max regret of \$200, and Flight C \$60. Thus C is the *minimax optimal choice*, as it has minimum max regret. Flight A and $u_1$ together serve as the *adversarial witness* that "prove" how much we could regret recommending C (since under $u_1$, A is \$60 better than C).

We contrast minimax regret with the oft-used *maximin* criterion (Wald, 1950), which recommends an option whose worst-case utility is greatest. In our example, maximin recommends flight A, with a guaranteed minimum utility of \$350. Notice the difference with minimax regret: while flight A ensures utility of \$350 if DM's utility function turns out to be $u_2$, flight C (the minimax optimal alternative) would have been a *far* better choice in that case.

Minimax regret has been adopted as a optimization criterion for problems in which there is data or objective function uncertainty (Kouvelis and Yu, 1997; Averbakh, 2000; Aissi et al., 2009), but only recently has been proposed as a means for accounting for a DSS's uncertainty regarding DM's utility (Boutilier et al., 2001; Salo and Hämäläinen, 2001; Wang and Boutilier, 2003; Boutilier et al., 2004, 2006; Braziunas and Boutilier, 2007).[2] Minimax regret has several advantages over Bayesian models from a practical perspective. First, we circumvent the need for Bayesian priors, which can be difficult to assess. Second, since most useful priors are not closed under the types of preference queries we consider in the next section, the required probabilistic computations are generally intensive and approximate; by contrast, bounds on utility function parameters are usually easy to elicit and minimax regret can be computed more effectively. Finally, minimax regret provides bounds on the degree of suboptimality associated with any decision. This can be especially important when priors are difficult

[2]We should distinguish regret as used here from its use to explain human violations of the principles of expected utility maximization (Loomes and Sugden, 1982; Bell, 1982).

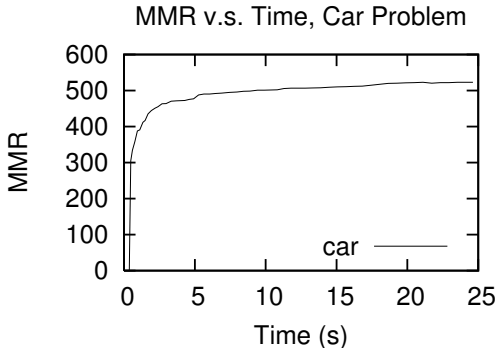



Figure 2: Lower bound on minimax regret on car-rental problem as a function of computation time (Boutilier et al., 2006).

to assess, since it provides robustness guarantees that are not possible by conjecturing, say, a uniform prior.

The computation of minimax regret is difficult in general. In many of these settings, optimization with a *known* utility function $u$ is already difficult combinatorial optimization problem (NP-hard), e.g., in product configuration with a set of constraints, or combinatorial auctions, both of which are typically formulated as *integer (linear) programs*. From a mathematical programming perspective, adding utility function uncertainty changes the problem from a standard linear optimization to a minimax optimization whose objective contains quadratic terms. We do not provide details here, but simply hint at the various classes of techniques that can be used in the following subsections.

### 3.2 Product Configuration and Corporate Sourcing

With utility function uncertainty, various mathematical programming decompositions (e.g., Benders' decomposition) and iterative procedures (e.g., constraint generation) can be used to render the problem practical (Boutilier et al., 2004, 2006; Braziunas and Boutilier, 2007). Generally, the space $U$ of feasible utility functions is represented by a polytope, or set of linear constraints on the parameters of DM's utility function. Methods for computing minimax regret take advantage of the additive or generalized additive structure of the utility function, allowing effective computation of minimax regret for configuration problems involving dozens of attributes and potentially billions of feasible alternatives. Moreover, many of these procedures have good *anytime properties*: approximate solutions can be returned whose quality improves with the computation time available, with the greatest gains coming earlier rather than later. For instance, Fig. 2 shows the anytime performance of minimax regret computation on a car rental domain with 26 variables, 61 billion feasible configurations, and a GAI utility model with 36 factors and 435 utility parameters (taken from Boutilier et al., 2006). While computing MMR exactly takes roughly 25 seconds in this example, a tight lower bound on MMR is found in under 5 seconds (upper bounds are also available). This has important consequences for preference elicitation as we discuss below. Search-based techniques are more appropriate when the space of possible options (e.g., products) is given by a database (Braziunas and Boutilier, 2007, 2010).

### 3.3 Markov Decision Processes

Reward function uncertainty in MDPs (see Sec. 2.4 for an overview of MDPs) has been addressed recently using a formalism known as *imprecise-reward MDPs (IRMDPs)* (Delage and Mannor, 2007; Regan and Boutilier, 2009; Xu and Mannor, 2009). Uncertainty over the parameters of the MDP reward function is captured by a set $\mathcal{R}$ of *feasible reward functions*, again represented by a polytope over reward space (i.e., linear constraints on the values of specific reward terms $R(s)$). *Minimax regret* has been proposed as an attractive robustness criterion for IRMDPs (Regan and Boutilier, 2009, 2010; Xu and Mannor, 2009). While solving IRMDPs using this measure is NP-hard (Xu and Mannor, 2009), techniques have been developed that allow the effective solution of IRMDPs of moderate size. Most of these methods exploit the linear programming formulations of standard MDPs with fully known rewards, and extend them to the uncertain case. Techniques include *constraint generation* to incrementally enumerate "adversarial policies" (Regan and Boutilier, 2009), methods that exploit so-called *non-dominated policies* (Regan and Boutilier, 2010; Xu and Mannor, 2009; Regan and Boutilier, 2011b), as well as techniques that take advantage of properties of the elicitation process (see the next section) to make computation even more tractable (Regan and Boutilier, 2011b).

### 3.4 Social Choice and Voting

In group decision making, we'd like to reduce the burden on voters by asking for only partial information about their preferences rather than asking them to rank all alternatives. Such a *partial ranking* is represented by a collection of *pairwise comparisons*, where each comparison takes the form $a_i \succ_k a_j$ (i.e., voter $k$ prefers $a_i$ to $a_j$). The responses to most natural queries (e.g., pairwise comparison, positional, top-$t$, and other queries) can be represented this way. A *partial profile* is set of partial rankings, one per voter. Of course, each voter has an (unknown) true preference ranking that consistently extends or *completes* the (known) partial ranking.

Somewhat surprisingly, little study of robust decision making with partial vote profiles exists. Concepts like *necessary* and *possible winners* (Konczak and Lang, 2005) have been proposed: the former considers whether an alternative $a$ wins no matter how we complete the voters' partial preferences; the latter whether $a$ wins under some such completion. Unfortunately, necessary winners don't generally exist, and the concept of possible winners provides no guidance for actual *selecting* an alternative. Minimax regret has been proposed as a robustness criterion for just such a scenario (Lu and Boutilier, 2011).[3] It is based on the observation that most voting rules are defined using a natural *scoring function* $s(a, \mathbf{v})$ that measures the quality of each alternative $a$ given a preference profile $\mathbf{v}$, with $a$ winning if it has the highest score (see Box 2). Given a partial profile, $a$ can have a range of different scores depending on how the partial profile is completed. As above, we define the maximum regret of $a$ by allowing the "adversarial" selection of a (complete) profile $\mathbf{v}$ to maximize the loss (difference in score) between our chosen alternative $a$ and the true winner under $\mathbf{v}$. We then chose an alternative $a^*$ that minimizes this regret.

Notice that this use of minimax regret—by measuring the potential loss in the selection of an alternative by its score difference with the optimal candidate—implicitly treats the scoring function

[3] Minimax regret has been used in voting contexts in different ways than discussed here, most notably to explain behavior of individual voters to account for voter participation in the face of Downs paradox (Ferejohn and Fiorina, 1974).

$s$ as a surrogate for "societal utility" or *social welfare*. This is not uncontroversial, and may not be suitable in all settings (e.g., selecting winners in political elections). But in many, if not most, preference aggregation problems, specific voting rules are often viewed in this way.

Minimax regret can be computed for some voting rules relatively effectively (in *polynomial time*) (Lu and Boutilier, 2011), though for other rules the problem can be computationally difficult. Unlike configuration problems, where the underlying problem even in the presence of complete utility information is hard, most voting rules can be computed efficiently with complete ranking information. An interesting distinction between voting and product configuration lies in the fact that configuration problems tend to have very large numbers of alternatives (feasible combinations of attributes) and relative few utility parameters (e.g., the weights in an additive utility model); by contrast, typical voting situations involve relatively few alternatives, but a large number of "utility" parameters (i.e., the preferences of many different voters).

In market-based mechanisms like VCG, group members are required to reveal utility functions (not just rankings) over alternatives. One can use minimax regret to compute the minimax optimal group choice (specifically, the alternative that has minimax regret with respect to social welfare). Unfortunately, an important component of the VCG scheme is the use of group member preferences to determine payments that incentivize truthful reporting. The incentive properties of VCG are destroyed by approximation of the outcome using minimax regret (Nisan and Ronen, 2000); however, approximate variants of these payments can be devised that bound the incentive (limit the gain) for members to misreport their preferences in a way that is tightly tied to the minimax regret level (Hyafil and Boutilier, 2006, 2007).

# 4 Preference Elicitation using Minimax Regret

Minimax regret allows a DSS to make robust decisions in the face of uncertainty about DM's utility function. However, if the minimax regret level $\mathit{MMR}(U)$, given knowledge $U$ of DM's utility function, is too great, the recommended alternative will have unacceptably high (potential) loss. This can only be reduced if more information is obtained about DM's true utility $u$, thus reducing the uncertainty in $U$. Therefore, we must couple decision making with techniques for eliciting additional preference information.

Many techniques have been developed in decision analysis, econometrics, artificial intelligence, and other disciplines for assessing preferences. We consider here methods for *active preference elicitation*, in which a DSS explicitly asks DM specific queries about her preferences: the aim is to discover "just enough" about her utility function to recommend a good or optimal decision, measuring decision quality using minimax regret. We outline a broad class of techniques for elicitation within the minimax regret framework known as *current solution (CS)* methods. These reflect a general heuristic principle that has proven to be very effective in many domains, as we outline below.

## 4.1 The Current Solution Heuristic

Our general elicitation framework assumes some class of queries $Q$ used to elicit information about DM's utility function. We discuss different types of queries below, but generally, the queries are

categorized into different "types." For example, a *global comparison query* may ask DM to compare two alternatives and select the one she most prefers. Thus all unordered pairs of alternatives comprise the set $Q$ of comparison queries. Some query types admit a continuous parameterization; e.g., a *global bound query* asks DM whether she would be willing to pay at least price $p$ for some alternative. Thus both an alternative and a price $p$ must be selected. Each query $q \in Q$ has a set of possible responses $R(q)$, each revealing information about DM's utility function. Comparison and bound queries have binary responses, and are generally easy to answer; but some queries, such as *value* queries ("what price $p$ would you be willing to pay for alternative $\mathbf{x}$?") have continuous responses (and are cognitively demanding for DM). All query types discussed below have (small) finite response spaces, and the response to any query can be represented by a linear constraint on DM's utility parameters; thus the feasible utility set $U$ at any stage is a polytope.

The DSS interacts with DM, each interaction eliciting input from DM that imposes additional constraints on $U$. Thus we shrink $U$ iteratively until minimax regret is reduced to a level acceptable to DM. In rough sketch, elicitation proceeds as follows:

1. Compute minimax regret *MMR* for $U$.
2. Repeat until *MMR* $< \tau$ (for some threshold $\tau$):
   (a) Request input from DM (e.g., ask query $q$).
   (b) Update the constraints over utility function parameters to reflect user input (e.g., response to $q$), giving new feasible utility set $U'$.
   (c) Recompute *MMR* with respect to $U'$.

In quasi-linear settings, minimax regret (and the quality threshold $\tau$) may be expressed in dollar terms. If minimax regret is reduced to zero, then the true optimal decision is found (even if DM's utility function has not been completely specified).

The efficacy of the general elicitation strategy described above depends crucially on the ability to select good queries. A number of methods have been developed for assessing DM preferences in areas ranging from decision analysis (Keeney and Raiffa, 1976; French, 1986) to (aggregate) consumer choice modeling (Louviere et al., 2000; Toubia et al., 2004), but most rely on explicitly attempting to elicit or discover a good model of DM preferences. Since our goal is simply to recommend a good alternative from the set $\mathbf{X}$, we do not reduce utility uncertainty for its own sake, but rather reduce minimax regret as quickly as possible.

The *current solution (CS)* heuristic chooses queries that do just this, based on the following intuition. Given polytope $U$, recall that minimax regret is defined as (see **Box 3**):

$$\mathit{MMR}(U) = \min_{\mathbf{x}\in\mathbf{X}} \max_{\mathbf{x}'\in\mathbf{X}} \max_{u\in U} u(\mathbf{x}') - u(\mathbf{x}).$$

Specifically, $\mathit{MMR}(U)$ is the pairwise max regret $R(\mathbf{x}, \mathbf{x}', U)$ between the minimax optimal recommendation $\mathbf{x}$ and an adversarially chosen configuration $\mathbf{x}'$. If we ask a query of DM whose response fails to further constrain the utility of either $\mathbf{x}$ or $\mathbf{x}'$, pairwise max regret between the two will not

change; so unless the response changes the minimax optimal decision $\mathbf{x}$, minimax regret is not reduced. The CS heuristic requires that the only queries that can be asked must provide information about utility function parameters that determine the utility of either $\mathbf{x}$ or $\mathbf{x}'$ (or both). For example, if our queries are global comparison queries, a natural application of this heuristic would be to ask DM: "Do you prefer $\mathbf{x}$ or $\mathbf{x}'$?"

The CS method serves as an effective filter, restricting attention to a much smaller set of queries than the full set $Q$, queries that have direct potential to reduce minimax regret. Since the computation of even myopically optimal queries is difficult (see below), this heuristic has computational benefits as well: query selection is based on information made available by the computation of the minimax optimal recommendation. We now consider its instantiation in several different domains.

## 4.2 Product Configuration

In a product configuration problem with attributes $X_1, \cdots, X_n$, there are a variety of query types. We describe several of these and discuss how the CS heuristic can be used to choose suitable queries of each type:

- *Global comparison queries* ask DM to compare two complete configurations $\mathbf{x}$ and $\mathbf{x}'$. In quasi-linear environments, the price can (optionally) be included: Fig. 3(a) illustrates such a comparison (with prices). CS requires that the comparison be made between the minimax optimal configuration $\mathbf{x}$ and the adversarial witness $\mathbf{x}'$.

- *Local comparison queries* take advantage of an additive decomposition of $u$, asking DM to compare two values $x_i$ and $x_i'$ of a single attribute $X_i$.[4] For each attribute $X_i$, CS considers only the comparison between the two attribute values $x_i$ and $x_i'$ that occur in $\mathbf{x}$ and $\mathbf{x}'$. The selection of the attribute $X_i$ to be queried uses an estimate of the degree to which MMR will be reduced (Braziunas and Boutilier, 2007).

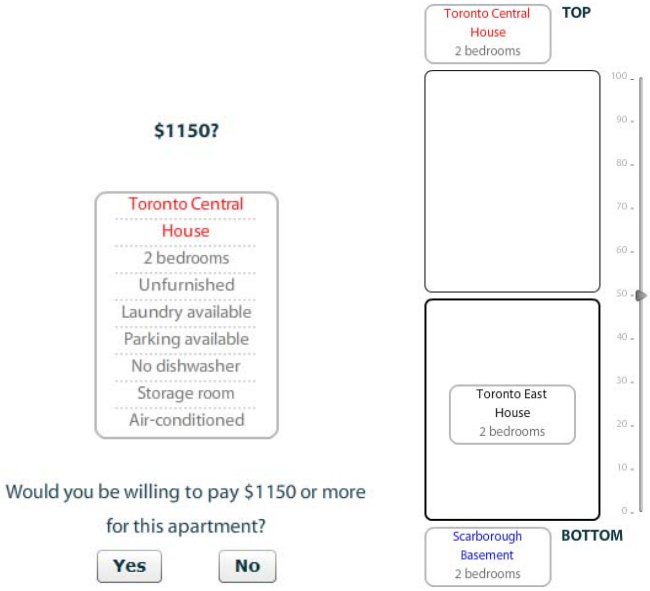
  *Local sorting* interfaces are quite natural, and obviate the need for local comparisons: DM can often specify a natural preference ordering on attributes values *a priori*. A sorting interface (shown for a GAI model) is illustrated in Fig. 3(b).

- *Global bound queries* ask DM to specify bounds on the utility of a complete configuration $\mathbf{x}$. This can be viewed as variant of the classic standard gamble, but in which the tradeoff probability is fixed, giving rise to a simple binary choice. In quasi-linear environments, the utility of $\mathbf{x}$ can be measured in terms of price or willingness to pay, so a bound on price can be elicited: see Fig. 3(c). With additive (and GAI) models, global queries can largely be avoided (Keeney and Raiffa, 1976; Fishburn, 1967; Braziunas and Boutilier, 2005), and are only needed for global calibration.

- *Local bound queries* ask DM to specify bounds on the local utility of some attribute value. A natural means of eliciting such bounds is to ask DM whether value $x_i$ of attribute $X_i$ rates

[4] With generalized additive models, one must provide a *conditioning context*, asking DM to compare two attribute values assuming a (usually small) subset of other attributes are fixed at their reference values (Braziunas and Boutilier, 2005, 2007).


(a) Global comparison (with prices) (b) Local sorting (local comparison)




(c) Global bound (d) Local bound

Figure 3: Four different query types (Braziunas and Boutilier, 2010).

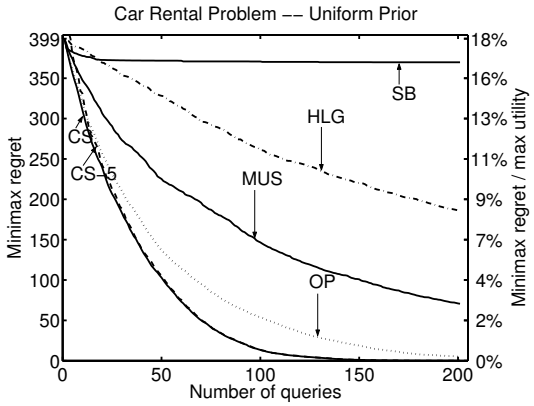



Figure 4: Average minimax regret on car-rental problem (45 instances) as a function of number of queries (Boutilier et al., 2006).

higher or lower than some point (e.g., 50) on a predefined scale (e.g., 0–100), where the best attribute value $x_i^{\top}$ is at the top of the scale and the worst value $x_i^{\perp}$ is at the bottom. This is illustrated (using a GAI model) in Fig. 3(d); in this interface, DM can actually tighten or loosen the bound depending on her comfort level with the response.

The response to a query of any of these types imposes linear constraints on the parameters of an additive (or GAI) utility function. The effectiveness of the CS heuristic has been demonstrated in several domains, both in simulation and with live users. It provides excellent simulated results over randomly drawn GAI and additive utility functions in product configuration domains involving dozens of attributes and billions of feasible configurations (Boutilier et al., 2006; Braziunas and Boutilier, 2007). For instance, Fig. 4 shows the performance of CS elicitation (denoted CS) compared to several other strategies in a car rental domain with 26 variables, 61 billion feasible configurations, and a GAI utility model with 36 factors and 435 parameters (Boutilier et al., 2006).

Interestingly, in experiments where (myopically) optimal queries are computed by exhaustive enumeration, CS reduces regret just as effectively as myopically optimal queries without the computational overhead. Furthermore, Boutilier et al. (2006) show that approximation of the minimax optimal solution provides almost no sacrifice in elicitation quality. For example, Fig. 4 shows that the method CS-5—identical to CS except that minimax regret computation is limited to 5 seconds, at which point the approximate answer is used to select a query (rather than the minimax optimal solution)—performs nearly indistinguishably from CS. Note that exact MMR computation takes 83 seconds on average in this domain, while the time bound of 5 seconds between queries is sufficient to support real-time interactive response. Several strategies designed explicitly to reduce utility uncertainty perform rather poorly by comparison (see, e.g., MUS and HLG in Fig. 4, the latter of which is similar to polyhedral methods proposed in conjoint analysis (Toubia et al., 2004).)

A study with users searching for rental accommodation (from an apartment database) have confirmed the effectiveness of minimax regret-based elicitation in practice (Braziunas and Boutilier, 2010). This study showed that users were generally quite comfortable with the notion of minimax

regret and found regret-based elicitation using the queries discussed above (and shown in Fig. 3) to be intuitive. Furthermore, even in a small database of only 100 apartments, regret-based elicitation allowed users to find high quality apartments in significantly less time than was required by a manual search (even when this search was supported by sorting and categorization tools). Finally, though access to true user utility generally requires full elicitation, the study showed that the apartments recommended by the minimax regret method were, for almost all users, optimal or near-optimal.

Notice that comparison queries can be generalized to more than two options; we can ask DM "Which of these $k$ options do you like most?" for some $k > 2$, a common form of query in survey data (Louviere et al., 2000; Toubia et al., 2004). One can also generalize the form of recommendations: suppose that rather than recommending a single configuration, we propose a set of $k$ choices to DM, from which she selects her most preferred option (e.g., we present four different, potentially "desirable" apartments). Presenting multiple alternatives can be valuable, especially if they reflect some "diversity" in DM preferences. One can generalize the notion of maximum regret to such *recommendation sets*.[5] Somewhat surprisingly, the optimal recommendation set of size $k$ is also a myopically optimal choice query (Viappiani and Boutilier, 2009, 2010). This has important implications in "conversational" recommendation, where the interaction involves making a series of recommendations which DM can either accept, thus terminating the process, or "critique," which provides further preference constraints and leads to another round of recommendation.

Many other forms of queries and interactions can be considered in product recommendation. For example, in "high frequency" consumer domains, where users purchase multiple variants of the same product type over time (e.g., books, music, movies), one can learn much about a user's preferences based on passive observations of purchase history using *collaborative filtering* techniques (Konstan et al., 1997; Hofmann and Puzicha, 1999; Marlin and Zemel, 2007).

### 4.3 Corporate Sourcing

The assessment of tradeoffs in corporate sourcing applications can proceed much as it does in product configuration. Often pairwise comparisons are presented to the sourcing team (the buyer): which of two allocations $(\mathbf{x}, c)$ and $(\mathbf{x}', c')$ is preferred, where $\mathbf{x}$ and $\mathbf{x}'$ are feature vectors associated with the allocations (e.g., number of suppliers awarded business, percentage of business awarded to specific suppliers, etc.) and $c$ and $c'$ are the total allocation costs. Using the CS strategy to determine appropriate sequences of comparisons, one can quickly determine the optimal allocation of business without assessing the precise value of the utility parameters (Boutilier et al., 2004).

Interestingly, when intelligent elicitation of this form is not available, sourcing teams routinely use winner determination software to assess these preferences anyway. For instance, the buyer might limit the number of suppliers to five and see how allocation cost changes (increases) relative to the unconstrained problem. If the increase is too great, the buyer may relax the constraint (limiting the number of suppliers to six) and see how this reduces cost. This process is repeated for many combinations of constraints; when it ends, the buyer chooses a generated allocation as making the right tradeoff between cost and relevant non-price attributes. Intelligent preference elicitation makes this exploration of different scenarios much more efficient, and provides quality guarantees on the final

[5] Naturally, allowing more options to be recommended can only reduce max regret.

results.

### 4.4 Markov Decision Processes

The space of queries available to elicit reward information in an MDP tends to be much richer than in the situations above because of the sequential nature of the decision process. One can distinguish MDPs with unstructured states from those with "factored" state spaces (Boutilier et al., 1999) in which each state is defined as an instantiation of certain variables. In the former case, one can ask queries about the specific reward of a state, e.g., bound queries (Regan and Boutilier, 2009) or precise reward queries (Delage and Mannor, 2007); or one can ask for a comparison of two states (e.g., comparison queries). In the factored case, one can assume an additive or generalized additive reward function over the state variables, and elicit the reward function exactly as in the case of multi-attribute, single-shot decision problems like product configuration (Regan and Boutilier, 2011a).

It is also natural in an MDP to ask queries about *state trajectories (sequences of states)* or distributions over trajectories. One can ask DM which of two trajectories represents more desirable behavior. One can also summarize a trajectory (or distribution) using *events counts* if the states themselves have specific reward-bearing attributes. In the COACH system (see Sec.2.4), a trajectory can be summarized by presenting the relative number of occurrences of specific events of interest (e.g., number of prompts, task completion time, caregiver interruption, etc.). This provides a convenient means of comparing (the effects of) two policies (Regan and Boutilier, 2011a).

For query selection, the CS strategy is readily adapted to the MDP setting. We refer to Regan and Boutilier (2009) for a discussion of CS in the case of MDPs with unstructured states, and to Regan and Boutilier (2011a) for a discussion of CS for factored MDPs with additive reward models.

### 4.5 Social Choice and Voting

The question of effective partial vote elicitation has received surprisingly little attention. Work in statistical and behavioral social choice has considered the question of estimating properties of elections (e.g., majority-consistent preferences) from random samples of the electorate, as well as determining appropriate sample sizes (see, e.g., Regenwetter et al., 2006). Unfortunately, winners can't be determined in many voting schemes without a large amount of information in the worst case (Conitzer and Sandholm, 2002, 2005). Nonetheless, the development of elicitation schemes that work well in practice has been addressed recently. For instance, Kalech et al. (2011) develop several heuristic strategies for vote elicitation, using the concept of possible and necessary winners to determine termination of the elicitation process.

Lu and Boutilier (2011) use minimax regret for vote elicitation, adapting the CS heuristic to the group setting. They consider two types of queries, pairwise comparisons (i.e., asking a voter if she prefers alternative $a$ to $b$), or top-$t$ queries (i.e., asking a voter for their first-ranked alternative, asking for their second-ranked alternative *only* if the first has already been specified, asking for the third only if the first two have been specified, etc.). CS is applied by selecting a voter-query pair that has the greatest potential to reduce minimax regret. Intuitively, this can be accomplished by considering the minimax optimal alternative $a^*$ and the adversarial witness $a'$, and finding the voter whose incomplete ranking offers the greatest potential to decrease the adversarial advantage of $a'$ over $a^*$ (i.e., decrease

the difference of their scores) in the minimax optimal solution. On both randomly generated data sets, and real ranking data sets (including electoral data and product preference data), CS has been shown to dramatically reduce the amount of preference information required to determine true winners relative to full elicitation; furthermore, near-optimal alternatives can be found with very little preference information.

We refer to Hyafil and Boutilier (2006, 2007) for a discussion of regret-based approaches to mechanism design, specifically, the use of minimax regret to guide the elicitation of partial utility functions in an approximate version of the VCG mechanism.

# 5 Interdisciplinary Opportunities

The models and results presented above are largely motivated by a fairly precise, mathematical formulation of decision problems as commonly adopted in decision analysis and operations research. The computational perspective is, of course, greatly facilitated by such precise, formal specifications; and the stylized nature of the interactions and queries used to elicit preferences reflects this. In many applications, such as those involving corporate sourcing (Boutilier et al., 2004), or many others, it is generally safe to assume that decision makers can answer such precise queries without difficulty. Indeed, in some applications, the "preferences" are monetary or involve some other objective measure, and computer simulations and optimization are used to answer these queries without the intervention of a human DM at all (Boutilier et al., 2003; Patrascu et al., 2005).

However, preference elicitation often involves interaction with non-expert human DMs, who are subject to the many foibles, biases in judgement, heuristic decision rules, and time pressures that motivate the use of computational DSSs in the first place. As such, the broader reach of computational decision support is predicated on methods that are sensitive to such factors. Research in many of the social and behavioral sciences should inform many of the next steps in the evolution of DSSs. We outline a few (of many) such possibilities for interaction in this section.

## 5.1 Models of Human Choice Behavior

Work in econometrics, psychometrics, statistics and other disciplines has proposed models of human choice behavior. Marden (1995), for example, provides an overview of various statistical models of preference and ranking data, many of which are inspired by psychometric accounts of choice. Such models of preferences can be exploited in many ways. If regularities exist in individual choice behavior, these should manifest themselves in statistical regularities in population preferences, which can in turn be exploited to make preference elicitation more effective.

The regret-based methods discussed above do not exploit distributional information (though we briefly mentioned alternative probabilistic approaches to elicitation, some Bayesian and some non-Bayesian in their approach to recommendation). It is, however, possible to combine the probabilistic viewpoint for elicitation with the minimax regret approach to decision making (Wang and Boutilier, 2003). Furthermore, much research in machine learning has tackled the effective learning of rankings from partially observed ranking data, especially in web search and online ratings systems (Cohen et al., 1999; Hüllermeier et al., 2008).

### 5.2 Cognitive Biases, Heuristics and Costs

One of the most important factors in human decision making, ignored in the techniques discussed above, are human biases and heuristics used in the assessment or comparison of alternatives. Research in psychology, cognitive science, behavioral decision theory and behavioral economics have identified a number of common biases involving framing, anchoring, endowment effects, risk and ambiguity aversion, discounting behavior, altruism and reciprocity, and numerous other phenomena that must ultimately influence the design of interactive decision support systems.[6] These phenomena must ultimately influence the way in which queries are framed and presented. However, it may be possible to account for systematic biases (possibly induced by the DSS itself) and "correct" for these by developing suitable "noise models."

Another important consideration in the design of DSSs is the cognitive cost associated with answering specific types of queries: this should be explicitly modeled and factored into the development of suitable elicitation strategies. This is another rich area in which the interaction of computer science, cognitive science and psychology can play a valuable role.

### 5.3 Group Choice

Research in social choice and economics has had a strong influence on computational models and tools for group decision support (see, e.g., Nisan et al., 2007, for an overview of research in computational game theory). Both topics have been discussed briefly above. But a topic that deserves more attention is the design of preference elicitation techniques that work more directly within more collaborative, discussion-oriented, argumentation-based, consensus-based or other more-involved group decision support processes. Research in both formal and informal group and organizational decision making (e.g., Raiffa, 2002), as well as computational systems designed to support such processes (e.g., Kraus, 2001) is of vital importance here, and the design of coherent elicitation schemes that can be blended into such processes would be of tremendous value.

## References

Abbas, A. (2004). Entropy methods for adaptive utility elicitation. *IEEE Transactions on Systems, Science and Cybernetics*, 34(2):169–178.

Aissi, H., Bazgan, C., and Vanderpooten, D. (2009). Min-max and min-max regret versions of combinatorial optimization problems: A survey. *European Journal of Operational Research*, 197(2):427–438.

Arrow, K. J., Sen, A. K., and Suzumura, K., editors (2002). *Handbook of Social Choice and Welfare*, volume 1. North Holland, Amsterdam.

Averbakh, I. (2000). Minmax regret solutions for minimax optimization problems with uncertainty. *Operations Research Letters*, 27:57–65.

[6] See, for example, the excellent collections by Camerer et al. (2003), Kahneman and Tversky (2000), and Bell et al. (1988).

Bacchus, F. and Grove, A. (1995). Graphical models for preference and utility. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence (UAI-95)*, pages 3–10, Montreal.

Bell, D. E. (1982). Regret in decision making under uncertainty. *Operations Research*, 30:961–981.

Bell, D. E., Raiffa, H., and Tversky, A., editors (1988). *Decision Making: Descriptive, Normative and Prescriptive Interactions*. Cambridge University Press, Cambridge.

Boger, J., Poupart, P., Hoey, J., Boutilier, C., Fernie, G., and Mihailidis, A. (2005). A decision-theoretic approach to task assistance for persons with dementia. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI-05)*, pages 1293–1299, Edinburgh.

Boger, J., Poupart, P., Hoey, J., Boutilier, C., Fernie, G., and Mihailidis, A. (2006). A planning system based on Markov decision processes to guide people with dementia through activities of daily living. *IEEE Transactions on Information Technology in Biomedicine*, 10(2):323–333.

Boutilier, C. (2002). A POMDP formulation of preference elicitation problems. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-02)*, pages 239–246, Edmonton.

Boutilier, C., Bacchus, F., and Brafman, R. I. (2001). UCP-Networks: A directed graphical representation of conditional utilities. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence (UAI-01)*, pages 56–64, Seattle.

Boutilier, C., Das, R., Kephart, J. O., Tesauro, G., and Walsh, W. E. (2003). Cooperative negotiation in autonomic systems using incremental utility elicitation. In *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence (UAI-03)*, pages 89–97, Acapulco.

Boutilier, C., Dean, T., and Hanks, S. (1999). Decision theoretic planning: Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research*, 11:1–94.

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

Boutilier, C., Sandholm, T., and Shields, R. (2004). Eliciting bid taker non-price preferences in (combinatorial) auctions. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence (AAAI-04)*, pages 204–211, San Jose.

Braziunas, D. and Boutilier, C. (2005). Local utility elicitation in GAI models. In *Proceedings of the Twenty-first Conference on Uncertainty in Artificial Intelligence (UAI-05)*, pages 42–49, Edinburgh.

Braziunas, D. and Boutilier, C. (2007). Minimax regret-based elicitation of generalized additive utilities. In *Proceedings of the Twenty-third Conference on Uncertainty in Artificial Intelligence (UAI-07)*, pages 25–32, Vancouver.

Braziunas, D. and Boutilier, C. (2008). Elicitation of factored utilities. *AI Magazine*, 29(4):79–92.

Braziunas, D. and Boutilier, C. (2010). Assessing regret-based preference elicitation with the UTPREF recommendation system. In *Proceedings of the Eleventh ACM Conference on Electronic Commerce (EC'10)*, pages 219–228, Cambridge.

Camerer, C. F., Loewenstein, G., and Rabin, M., editors (2003). *Advances in Behavioral Economics*. Princeton University Press, Princeton.

Chajewska, U., Koller, D., and Parr, R. (2000). Making rational decisions using adaptive utility elicitation. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-00)*, pages 363–369, Austin.

Chevaleyre, Y., Endriss, U., Lang, J., and Maudet, N. (2007). A short introduction to computational social choice. In *Proceedings of the 33rd Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM-07)*, pages 51–69, Harrachov.

Clarke, E. H. (1971). Multipart pricing of public goods. *Public Choice*, 11(1):17–33.

Cohen, W. W., Schapire, R. E., and Singer, Y. (1999). Learning to order things. *Journal of Artificial Intelligence Research*, 10:243–270.

Conitzer, V. and Sandholm, T. (2002). Vote elicitation: Complexity and strategy-proofness. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-02)*, pages 392–397, Edmonton.

Conitzer, V. and Sandholm, T. (2005). Communication complexity of common voting rules. In *Proceedings of the Sixth ACM Conference on Electronic Commerce (EC'05)*, pages 78–87, Vancouver.

Cramton, P., Shoham, Y., and Steinberg, R., editors (2005). *Combinatorial Auctions*. MIT Press, Cambridge.

Dechter, R. (2003). *Constraint Processing*. Morgan Kaufmann, San Francisco.

Delage, E. and Mannor, S. (2007). Percentile optimization in uncertain Markov decision processes with application to efficient exploration. In *Proceedings of the Twenty-fourth International Conference on Machine Learning (ICML-07)*, pages 225–232, Corvallis.

Ferejohn, J. A. and Fiorina, M. P. (1974). The paradox of not voting: A decision theoretic analysis. *The American Political Science Review*, 68(2):525–536.

Fishburn, P. C. (1967). Interdependence and additivity in multivariate, unidimensional expected utility theory. *International Economic Review*, 8:335–342.

French, S. (1986). *Decision Theory*. Halsted Press, New York.

Gibbard, A. (1973). Manipulation of voting schemes: A general result. *Econometrica*, 41(4):587–601.

Gonzales, C. and Perny, P. (2004). GAI networks for utility elicitation. In *Proceedings of the Ninth International Conference on Principles of Knowledge Representation and Reasoning (KR2004)*, pages 224–234, Whistler, BC.

Groves, T. (1973). Incentives in teams. *Econometrica*, 41:617–631.

Hofmann, T. and Puzicha, J. (1999). Latent class models for collaborative filtering. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 688–693, Stockholm.

Holloway, H. A. and White, III, C. C. (2003). Question selection for multiattribute decision-aiding. *European Journal of Operational Research*, 148:525–543.

Hüllermeier, E., Fürnkranz, J., Cheng, W., and Brinker, K. (2008). Label ranking by learning pairwise preferences. *Artificial Intelligence*, 172(16-17):1897–1916.

Hyafil, N. and Boutilier, C. (2006). Regret-based incremental partial revelation mechanisms. In *Proceedings of the Twenty-first National Conference on Artificial Intelligence (AAAI-06)*, pages 672–678, Boston.

Hyafil, N. and Boutilier, C. (2007). Mechanism design with partial revelation. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI-07)*, pages 1333–1340, Hyderabad.

Iyengar, V. S., Lee, J., and Campbell, M. (2001). Q-Eval: Evaluating multiple attribute items using queries. In *Proceedings of the Third ACM Conference on Electronic Commerce*, pages 144–153, Tampa.

Kahneman, D. and Tversky, A., editors (2000). *Choices, Values, and Frames*. Cambridge University Press, Cambridge.

Kalech, M., Kraus, S., Kaminka, G. A., and Goldman, C. V. (2011). Practical voting rules with partial information. *Journal of Autonomous Agents and Multi-Agent Systems*, 22(1):151–182.

Keeney, R. L. and Raiffa, H. (1976). *Decisions with Multiple Objectives: Preferences and Value Trade-offs*. Wiley, New York.

Konczak, K. and Lang, J. (2005). Voting procedures with incomplete preferences. In *IJCAI-05 Workshop on Advances in Preference Handling*, pages 124–129, Edinburgh.

Konstan, J. A., Miller, B. N., Maltz, D., Herlocker, J. L., Gordon, L. R., and Riedl, J. (1997). Grouplens: Applying collaborative filtering to usenet news. *Communications of the ACM*, 40(3):77–87.

Kouvelis, P. and Yu, G. (1997). *Robust Discrete Optimization and Its Applications*. Kluwer, Dordrecht.

Kraus, S. (2001). *Strategic Negotiation in Multiagent Environments*. MIT Press, Cambridge.

Loomes, G. and Sugden, R. (1982). Regret theory: An alternative theory of rational choice under uncertainty. *Economic Journal*, 92:805–824.

Louviere, J. J., Hensher, D. A., and Swait, J. D. (2000). *Stated Choice Methods: Analysis and Application*. Cambridge University Press, Cambridge.

Lu, T. and Boutilier, C. (2011). Robust approximation and incremental elicitation in voting protocols. In *Proceedings of the Twenty-second International Joint Conference on Artificial Intelligence (IJCAI-11)*, pages 287–293, Barcelona.

Marden, J. I. (1995). *Analyzing and modeling rank data*. Chapman and Hall, London.

Marlin, B. M. and Zemel, R. S. (2007). Collaborative filtering and the missing at random assumption. In *Proceedings of the Twenty-third Conference on Uncertainty in Artificial Intelligence (UAI-07)*, pages 50–54, Vancouver.

Mihailidis, A., Fernie, G. R., and Barbanel, J. C. (2001). The use of artificial intelligence in the design of an intelligent cognitive orthosis for people with dementia. *Assistive Technology*, 13:23–39.

Nisan, N. and Ronen, A. (2000). Computationally feasible VCG mechanisms. In *Proceedings of the Second ACM Conference on Electronic Commerce (EC'00)*, pages 242–252, Minneapolis.

Nisan, N., Roughgarden, T., Tardos, E., and Vazirani, V. V., editors (2007). *Algorithmic Game Theory*. Cambridge University Press, Cambridge.

Patrascu, R., Boutilier, C., Das, R., Kephart, J. O., Tesauro, G., and Walsh, W. E. (2005). New approaches to optimization and utility elicitation in autonomic computing. In *Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI-05)*, pages 140–145, Pittsburgh.

Puterman, M. L. (1994). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley, New York.

Raiffa, H. (2002). *Negotiation Analysis: The Science and Art of Collaborative Decision Making*. Harvard University Press, Cambridge.

Regan, K. and Boutilier, C. (2009). Regret-based reward elicitation for Markov decision processes. In *Proceedings of the Twenty-fifth Conference on Uncertainty in Artificial Intelligence (UAI-09)*, pages 454–451, Montreal.

Regan, K. and Boutilier, C. (2010). Robust policy computation in reward-uncertain MDPs using nondominated policies. In *Proceedings of the Twenty-fourth AAAI Conference on Artificial Intelligence (AAAI-10)*, pages 1127–1133, Atlanta.

Regan, K. and Boutilier, C. (2011a). Eliciting additive reward functions for markov decision processes. In *Proceedings of the Twenty-second International Joint Conference on Artificial Intelligence (IJCAI-11)*, pages 2159–2164, Barcelona.

Regan, K. and Boutilier, C. (2011b). Robust online optimization of reward-uncertain MDPs. In *Proceedings of the Twenty-second International Joint Conference on Artificial Intelligence (IJCAI-11)*, pages 2165–2171, Barcelona.

Regenwetter, M., Grofman, B., Marley, A. A. J., and Tsetlin, I. (2006). *Behavioral Social Choice: Probabilistic Models, Statistical Inference, and Applications*. Cambridge University Press, Cambridge.

Rothkopf, M., Pekeč, A., and Harstad, R. (1998). Computationally manageable combinatorial auctions. *Management Science*, 44(8):1131–1147.

Salo, A. and Hämäläinen, R. P. (2001). Preference ratios in multiattribute evaluation (PRIME)–elicitation and decision procedures under incomplete information. *IEEE Trans. on Systems, Man and Cybernetics*, 31(6):533–545.

Sandholm, T. (2007). Expressive commerce and its application to sourcing: How we conducted $35 billion of generalized combinatorial auctions. *AI Magazine*, 28(3):45–58.

Satterthwaite, M. A. (1975). Strategy-proofness and arrow's conditions: Existence and correspondence theorems for voting procedures and social welfare functions. *Journal of Economic Theory*, 10:187–217.

Savage, L. J. (1951). The theory of statistical decision. *Journal of the American Statistical Association*, 46(253):55–67.

Sykes, E. A. and White, III, C. C. (1991). Multiobjective intelligent computer-aided design. *IEEE Transactions on Systems, Man and Cybernetics*, 21(6):1498–1511.

Toubia, O., Hauser, J. R., and Simester, D. I. (2004). Polyhedral methods for adaptive choice-based conjoint analysis. *Journal of Marketing Research*, 41(1):116–131.

Toubia, O., Simester, D. I., Hauser, J. R., and Dahan, E. (2003). Fast polyhedral adaptive conjoint estimation. *Marketing Science*, 22(3):273–303.

Viappiani, P. and Boutilier, C. (2009). Regret-based optimal recommendation sets in conversational recommender systems. In *Proceedings of the 3rd ACM Conference on Recommender Systems (RecSys09)*, pages 101–108, New York.

Viappiani, P. and Boutilier, C. (2010). Optimal bayesian recommendation sets and myopically optimal choice query sets. In *Advances in Neural Information Processing Systems 23 (NIPS)*, Vancouver.

Vickrey, W. (1961). Counterspeculation, auctions, and competitive sealed tenders. *Journal of Finance*, 16(1):8–37.

Wald, A. (1950). *Statistical Decision Functions*. Wiley, New York.

Wang, T. and Boutilier, C. (2003). Incremental utility elicitation with the minimax regret decision criterion. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-03)*, pages 309–316, Acapulco.

Weber, M. (1987). Decision making with incomplete information. *European Journal of Operational Research*, 28:44–57.

White, III, C. C., Sage, A. P., and Dozono, S. (1984). A model of multiattribute decision-making and trade-off weight determination under uncertainty. *IEEE Transactions on Systems, Man and Cybernetics*, 14(2):223–229.

Xu, H. and Mannor, S. (2009). Parametric regret in uncertain Markov decision processes. In *48th IEEE Conference on Decision and Control*, pages 3606–3613, Shanghai.