Bayesian Interactive Decision Support for Multi-Attribute Problems with Even Swaps

Debarun Bhattacharjya and Jeffrey O. Kephart
Cognitive Computing Research, IBM T. J. Watson Research Center
1101 Kitchawan Rd, Rt. 134, Yorktown Heights, NY 10598, USA
E-mail: debarunb, kephart@us.ibm.com

Abstract

Even swaps is a method for solving deterministic multi-attribute decision problems where the decision maker iteratively simplifies the problem until the optimal alternative is revealed (Hammond et al. 1998, 1999). We present a new practical decision support system that takes a Bayesian approach to guiding the even swaps process, where the system makes queries based on its beliefs about the decision maker's preferences and updates them as the interactive process unfolds. Through experiments, we show that it is possible to learn enough about the decision maker's preferences to measurably reduce the cognitive burden, i.e. the number and complexity of queries posed by the system.

1 INTRODUCTION

In deterministic multi-attribute problems, the decision maker (or DM, for short) chooses among N alternatives, each of which has M attributes. An alternative $\boldsymbol{x}$ is a vector of consequences for each attribute:

$$\boldsymbol{x} = \{x_i : i = 1, \ldots, M\}, \tag{1}$$

where x_i is the consequence for attribute i. This is often represented as a ***consequence table*** such as the one illustrated in Fig 1(a), which displays alternatives and attributes for a hiring problem along its columns and rows respectively.

The DM's preferences for the various attributes can be modeled using a ***value function*** $v(\boldsymbol{x})$. Additive value functions are a popular choice, mainly due to the ease with which they can be elicited:

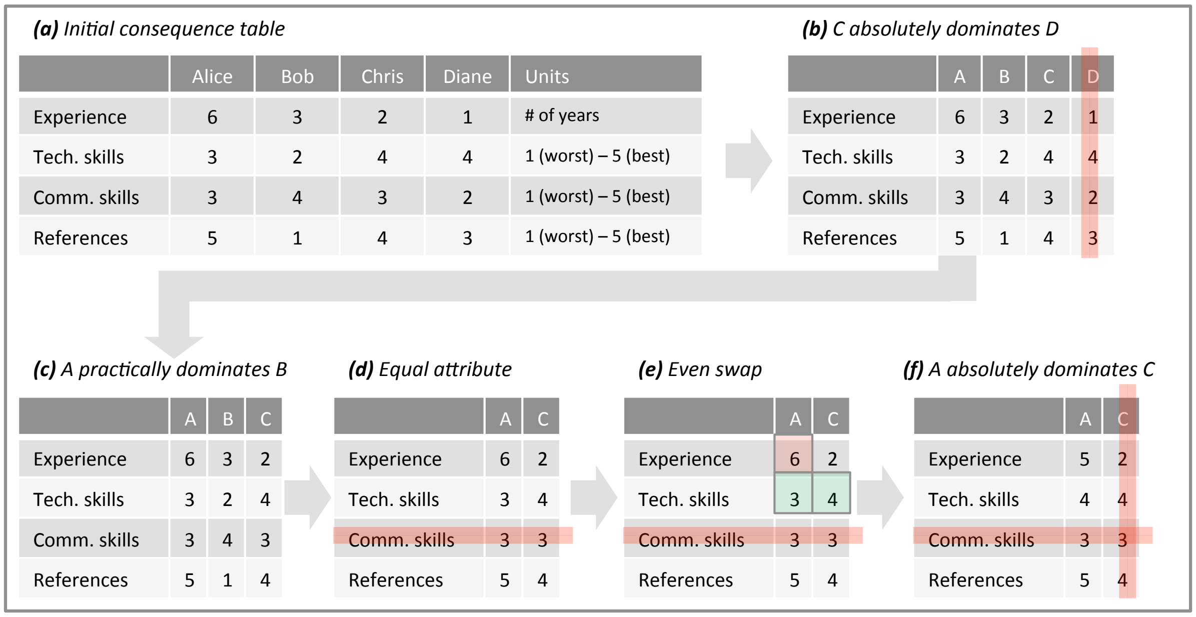
$$v(\boldsymbol{x}) = \sum_{i=1}^{M} w_i v_i(x_i), \tag{2}$$

where attribute weights $\boldsymbol{w} = \{w_i : i = 1, \ldots, M\}$ are non-negative and sum to 1 and the $v_i(x_i)$ represent one-dimensional marginal value functions. Note that we make a distinction between value and utility functions, following Keeney and Raiffa (1976), who reserve the term 'utility function' to characterize preferences under uncertainty.

There are several well-known approaches to eliciting additive value functions. The most popular ones tackle direct elicitation, where the DM reveals their trade-offs by answering questions pertaining to the weights and marginal value functions. von Winterfeldt and Edwards (1986) and Belton and Stewart (2002) review some well known weighting techniques. An alternate approach is that of ***even swaps***, which is an indirect preference elicitation method that simultaneously solves a specific decision problem (Hammond et al. 1998, 1999). Here the DM answers a few simple and pointed queries to iteratively reduce the number of columns and rows in the consequence table until the optimal alternative is revealed. Mustajoki and Hämäläinen (2005, 2007) coined the term ***smart swaps*** to refer to guided even swaps, i.e. using a decision support system to provide process suggestions.

In this paper, we propose a Bayesian approach to guiding the even swaps process, whereby the system makes queries based on its beliefs about the DM's preferences and updates them as the interactive process unfolds. The literature on Bayesian techniques for learning a DM's preferences is vast and varied, spanning domains such as management science, artificial intelligence, expert systems and machine learning (e.g. Eliashberg and Hauser 1985, Jimison et al. 1992, Poh and Horvitz 1993, Chajewska et al. 2000, Anderson and Hobbs 2002, Boutilier 2002, Scott and Shachter 2005).

We review the even swaps method in Section 2. The subsequent three sections summarize our main contributions. In Section 3, we present some properties of even swaps, providing conditions under which they are feasible. In Section 4, we describe our swaps algorithm


***(a)** Initial consequence table*

	Alice	Bob	Chris	Diane	Units
Experience	6	3	2	1	# of years
Tech. skills	3	2	4	4	1 (worst) – 5 (best)
Comm. skills	3	4	3	2	1 (worst) – 5 (best)
References	5	1	4	3	1 (worst) – 5 (best)

***(b)** C absolutely dominates D*

	A	B	C	D
Experience	6	3	2	1
Tech. skills	3	2	4	4
Comm. skills	3	4	3	2
References	5	1	4	3

***(c)** A practically dominates B*

	A	B	C
Experience	6	3	2
Tech. skills	3	2	4
Comm. skills	3	4	3
References	5	1	4

***(d)** Equal attribute*

	A	C
Experience	6	2
Tech. skills	3	4
Comm. skills	3	3
References	5	4

***(e)** Even swap*

	A	C
Experience	6	2
Tech. skills	3	4
Comm. skills	3	3
References	5	4

***(f)** A absolutely dominates C*

	A	C
Experience	5	2
Tech. skills	4	4
Comm. skills	3	3
References	5	4

Figure 1: The even swaps method applied to a hiring problem.

in detail. In Section 5, we discuss the results of some experiments that study the effect of problem size and Bayesian learning on the number and type of queries made to the DM. We are not aware of any previous work with computer experiments that explores the effect of smart swaps on a set of consequence tables and DMs. Finally, we conclude in Section 6.

2 EVEN AND SMART SWAPS

We will explain the even swaps method with the help of the following illustrative example:

A Hiring Example. Figure 1(a) presents the consequence table for a manager Zoe who faces a hiring decision and must choose among four candidates — Alice, Bob, Chris and Diane — across four attributes: Experience (in # of years) and qualitative measures such as Technical Skills, Communication Skills and References, all scored on a scale of 1 (worst) to 5 (best).

Zoe chooses to pursue the even swaps method to determine the optimal hire. First, she recognizes that Chris scores at least as well as Diane on all attributes, and therefore removes Diane from consideration in 1(b). This is an example of ***absolute dominance***. Next, she observes that Alice fares better on most attributes as compared to Bob, except for Technical Skills where Bob scores 1 point higher. Feeling that Alice compensates for this deficit along the other attributes, i.e. that Alice exhibits ***practical dominance*** over Bob, Zoe removes Bob from consideration in 1(c).[1]

Zoe then notices that the remaining candidates, Alice and Chris, have the same score (3) on Communication Skills. Reasoning that she need not be concerned with this attribute in subsequent iterations, as she can make subsequent value judgments conditional on this common score, she greys this attribute out in 1(d). In the even swaps literature, this task is referred to as identifying an irrelevant attribute; for reasons explained in the next section, we prefer the term ***equal attribute***, and say that this attribute has become inactive.

Now Zoe makes the move that gives the ***even swap*** method its name. She observes that Alice fares worse on Technical Skills, but better on the remaining active attributes. She answers the following question, indicated by the three boxes in 1(e): how many years of Experience would she be willing to give up for Alice to improve her Technical Skills score from 3 to 4? An even swap produces a hypothetical equivalent alternative in which a change in the consequences of one attribute balances the change in the consequences of another, and is a specific kind of matching query (Delquié 1993). The DM's response is determined by her value judgments; in this case, she determines that Alice's Experience should change from 6 to 5 years. She then replaces Alice with her hypothetical clone in the consequence table in 1(f). In the final step, she recognizes that Alice absolutely dominates Chris, thereby revealing Alice to be the optimal candidate. □

[1]The original even swaps literature introduced practical dominance as an intuitive but vague notion. Subsequent work on smart swaps proposed a definition with some practical drawbacks. A major contribution of our work is a precise definition and demonstration of its practicality.

The key idea that differentiates indirect methods like even swaps from direct elicitation techniques is that the analyst/system need not have a complete picture of the DM's preferences to find the optimal alternative for a particular decision. It is therefore often beneficial to use such techniques for reducing elicitation burden and potential inaccuracies, as people are highly susceptible to cognitive biases (Lichtenstein and Slovic 2006).

The even swaps method appears to be suitable for small problems where the interactive nature of the method, the access to the alternatives and the (almost) instant gratification from solving the problem appeal to the DM. It is particularly useful for DMs who either find it difficult to answer questions about their trade-offs in terms of weight ratios, or who need to view/consider the alternatives to construct their preferences. Kajanus et al. (2001) provide an application to strategy selection in rural enterprises.

Even swaps was originally intended to be self-guided; Mustajoki and Hämäläinen (2005, 2007) propose a decision support system for smart swaps using preference programming, i.e. by recognizing the feasible region of weights for fixed bounds on marginal value functions. Their model makes the practical dominance notion precise by recommending it through pairwise dominance, which occurs when there is no way an alternative can be most preferred, based on the feasible weight region and bounds. Their method however has several limitations. For instance, there is little the system can do if it proposes a practical dominance query and the DM rejects it, aside from changing bounds midway through the process. Crucially, they are unable to recognize swaps that are not feasible.

Here we propose a Bayesian approach that exploits prior information about the feasible weight region as represented by a prior probability distribution. We introduce the notion of probable dominance as well as a heuristic that recommends even swaps through probabilistic computations. The system easily handles rejection of practical dominance queries. We also present new results about feasibility conditions for even swaps, using them to recognize and adapt to declarations of infeasible swaps. Our approach is particularly adept at providing inexperienced users with specific recommendations. However, as discussed in the conclusions, our method possesses its own set of limitations.

3 PROPERTIES OF EVEN SWAPS

The overarching even swaps method gets its name from the even swap query, which is crucial towards reducing the size (and therefore complexity) of the consequence table. In this section, we specify our assumptions for the class of multi-attribute problems under consideration, and then present some results pertaining to the properties of even swaps.

3.1 ASSUMPTIONS

We address multi-attribute problems where the DM has an additive value function, i.e. of the form in equation (2). This is applicable only when attributes are mutually preferentially independent. As noted by previous authors, this is a common assumption and is widely applied in practice (Keeney and Raiffa 1976, Stewart 1996, Belton and Stewart 2002).

In theory, the even swaps method is applicable for all value functions and is not restricted to the additive form. However, the method can be challenging to apply when there is value dependence, in which case the DM would have to consider consequence levels of all attributes while making a judgment about an even swap. In that sense, no attribute would be 'irrelevant' when the DM makes the even swap based on their trade-offs. It is difficult to imagine the method being implemented successfully in such a situation without an analyst in the room to guide the DM. The additive assumption therefore makes an automated decision support system more likely to be used (and perhaps misused).

We also assume that the one-dimensional marginal value functions are continuous, bounded and monotonic. Since they are bounded, these functions can be normalized such that $0 \leq v_i(x_i) \leq 1$, $v_i({x_i}^0) = 0$ and $v_i({x_i}^*) = 1$ for all attributes, where ${x_i}^0$ and ${x_i}^*$ represent the least and most preferred consequences for attribute i. The domain of an attribute is denoted D_i, therefore for an attribute where more is preferred to less, $D_i = \left[{x_i}^0, {x_i}^*\right]$. Monotonic attributes are common in practice; non-monotonic attributes can sometimes be redefined so as to render them monotonic. Furthermore, a discrete attribute can often be approximated as continuous. For instance, in the hiring problem in Figure 1, three of the four attributes are measured as integers on a scale of 1 to 5, but they could easily be approximated as continuous attributes. These assumptions are therefore not too restrictive.

3.2 NOTATION AND PROPERTIES

The even swaps method attempts to guide the DM by simplifying the consequence table. During this interactive process, the DM must carefully consider pairs of alternatives and their consequences along specific attributes. A consequence x_i is deemed to be preferred over y_i if it has higher marginal value:

$$x_i \succ y_i \Leftrightarrow v_i(x_i) > v_i(y_i). \tag{3}$$

Any pair of alternatives $\boldsymbol{x}$ and $\boldsymbol{y}$ can therefore be associated with the following three sets of attributes:

dominating set $D(\boldsymbol{x},\boldsymbol{y}) = \{i : x_i \succ y_i\}$, ***non-dominating set*** $N(\boldsymbol{x},\boldsymbol{y}) = \{i : x_i \prec y_i\}$, and ***equal set*** $E(\boldsymbol{x},\boldsymbol{y}) = \{i : x_i = y_i\}$. Note that $N(\boldsymbol{x},\boldsymbol{y}) = D(\boldsymbol{y},\boldsymbol{x})$.

The task with perhaps the lowest cognitive load for the DM and the lowest computational load for a system is identifying equal attributes. While somewhat more complex for a DM, it is also trivial for a system to discover absolute dominance, denoted $\boldsymbol{x} \succeq^A \boldsymbol{y}$, using non-dominating attribute sets:

$$\boldsymbol{x} \succeq^A \boldsymbol{y} \Leftrightarrow N(\boldsymbol{x},\boldsymbol{y}) = \emptyset. \tag{4}$$

For a replicate pair of solutions, i.e. where both $N(\boldsymbol{x},\boldsymbol{y}) = \emptyset$ and $D(\boldsymbol{x},\boldsymbol{y}) = \emptyset$, either $\boldsymbol{x}$ or $\boldsymbol{y}$ can be removed from the table at random.

Practical dominance comes under consideration when one of the sets $N(\boldsymbol{x},\boldsymbol{y})$ and $D(\boldsymbol{x},\boldsymbol{y})$ has many more elements than the other. While practical dominance claims help remove some solutions, the DM may eventually have to perform an even swap to manipulate the consequence table and make further progress. We denote an even swap as $s(x_i \rightarrow x'_i, x_j \rightarrow x'_j)$, where the alternative $\boldsymbol{x}$ is modified by the DM, such that the change from x_i to x'_i along attribute i is compensated by the change from x_j to x'_j along attribute j.

Consider the even swap in the hiring example, where the DM provided a response to a change from the score 3 to 4 on Technical Skills, along Experience. Note that the swap performed was specifically designed to make consequences identical for Technical Skills. This type of swap is relatively cognitively comfortable for the DM, since they are able to observe the numbers along a specific row. Moreover, ensuring equal consequences simplifies the table and allows for potential ease of elicitation in future tasks. We refer to such a swap as an ***equalizing even swap***, defined as an even swap that makes the consequences of two alternatives equal along an attribute. For any two alternatives $\boldsymbol{x}$ and $\boldsymbol{y}$, $s(x_i \rightarrow y_i, x_j \rightarrow x'_j)$ is an equalizing even swap because it makes attribute i's consequences for both alternatives equal, thereby increasing the set $E(\boldsymbol{x},\boldsymbol{y})$.

Is an even swap always possible? No. The following proposition provides the conditions under which an even swap is feasible, assuming that the DM's response is consistent with their value function.

Proposition 1 (Even Swap Feasibility)**.** *The even swap $s(x_i \rightarrow x'_i, x_j \rightarrow x'_j)$, $i \neq j$, $x_i, x'_i \in D_i$ is feasible only if:*

(i) When $x'_i \succ x_i$: $v_j(x_j) \geq \frac{w_i}{w_j}\left[v_i(x'_i) - v_i(x_i)\right]$

(ii) When $x'_i \prec x_i$: $1 - v_j(x_j) \geq \frac{w_i}{w_j}\left[v_i(x_i) - v_i(x'_i)\right]$

Proof. If $x'_i \succ x_i$, the swap is not feasible when even a response of $x'_j = x^0_j$ cannot compensate for the change, which occurs when $w_j\left[v_j(x_j) - v_j(x^0_j)\right] < w_i\left[v_i(x'_i) - v_i(x_i)\right]$. The result follows after recognizing $v_j(x^0_j) = 0$. The other case is similar. □

The fact that not all swaps are feasible is potentially problematic for a system attempting to guide the process by recommending equalizing even swaps. Since the system is not exactly aware of the DM's preferences during the process, it is possible for the system to propose a swap that is infeasible for the DM. Fortunately, as determined in the following proposition, if the swap $s(x_i \rightarrow y_i, x_j \rightarrow x'_j)$ is not feasible, its ***conjugate swap*** $s(x_j \rightarrow y_j, x_i \rightarrow x'_i)$ must be feasible.

Proposition 2 (Equalizing Even Swap Feasibility)**.** *For any two alternatives $\boldsymbol{x}$ and $\boldsymbol{y}$ that do not dominate each other over attributes i and j, at least one of the equalizing even swaps $s(x_i \rightarrow y_i, x_j \rightarrow x'_j)$ or $s(x_j \rightarrow y_j, x_i \rightarrow x'_i)$ is feasible.*

Proof. Suppose that $y_i \succ x_i$. If the swap $s(x_i \rightarrow y_i, x_j \rightarrow x'_j)$ is not feasible, then from Proposition 1(i), $v_j(x_j) < \frac{w_i}{w_j}\left[v_i(y_i) - v_i(x_i)\right]$. Rearranging, $w_i v_i(x_i) + w_j v_j(x_j) < w_i v_i(y_i)$. For the conjugate swap, by definition, $w_i v_i(x_i) + w_j v_j(x_j) = w_i v_i(x'_i) + w_j v_j(y_j)$. Using the condition from the infeasibility of the original swap, $w_i v_i(x'_i) + w_j v_j(y_j) < w_i v_i(y_i) \implies w_i v_i(x'_i) < w_i v_i(y_i) \implies x'_i \prec y_i$. The conjugate swap is therefore indeed feasible. The other case is similar. □

The implication of these results is that a feasible swap can always be found: if the DM declares that a given even swap is infeasible, then the conjugate swap will be feasible, and the system can recommend it.

Note that both propositions assume that a DM's response is consistent with their value function. However, behavioral research on bi-matching suggests people may provide inconsistent responses between queries pertaining to a swap vs. its conjugate (Delquié 1997, Willemsen and Keren 2003). In the algorithm described in the next section, we assume the DM is willing to make either a swap or its conjugate, but we allow for noise in the response to an even swap query.

4 BAYESIAN SMART SWAPS

In our formulation of guiding an interactive even swap, we assume the system has prior beliefs $p(\boldsymbol{w})$ about the DM's weights in their additive value function. If there is no a priori information available, the system may choose a uniform prior over the weight simplex: $p(\boldsymbol{w}) \sim \text{Dirichlet}(\boldsymbol{\alpha})$ where $\boldsymbol{\alpha}$ is a vector of 1s.

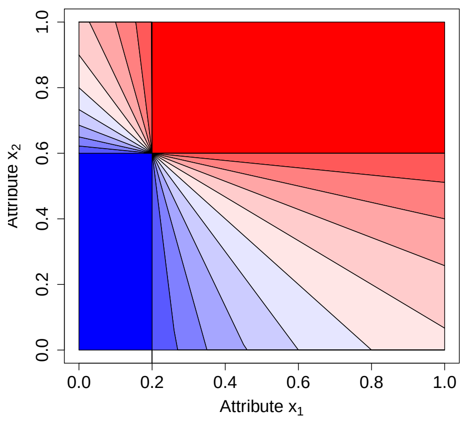


Figure 2: Regions of absolute and practical dominance for the two-attribute example when $w_1 \sim \text{Uniform}(0, 1)$.

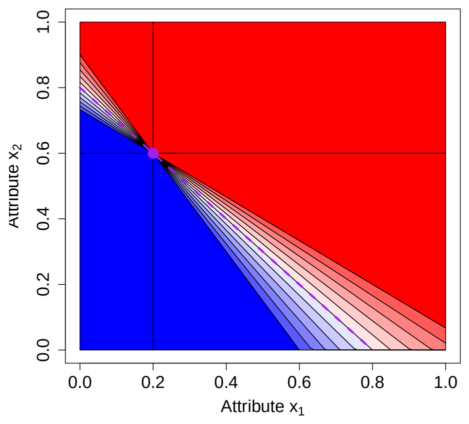


Figure 3: Regions of absolute and practical dominance for the two-attribute example when $w_1 \sim \text{Uniform}(0.4, 0.6)$.

We also assume for now that the system knows the DM's marginal value functions, perhaps through prior assessments. Since these are one-dimensional functions, they are usually easier to elicit than weights that reflect trade-offs. In sub-section 4.4 we briefly outline how our algorithm may be extended to the case of unknown marginal functions.

We explore how a system can cope with uncertainty about the DM's weights, incorporating responses to recommended practical dominance and even swap queries from the DM. In our algorithm, the system gradually learns the user's preferences and exploits it for the sole purpose of reaching the optimal alternative as soon as possible. In the following sub-sections, we describe various aspects of our overall approach.

4.1 ABSOLUTE VS. PROBABLE DOMINANCE

One of the central notions of the original even swaps method is that of practical dominance, according to which an alternative can be discarded if it appears to be *nearly* absolutely dominated by another. In this section, we view practical dominance through a Bayesian lens, with the intent of reducing the cognitive burden of DMs. To motivate our approach, let us first study absolute dominance.

Consider an alternative $\boldsymbol{x}$ whose consequences have been normalized; therefore it lies somewhere in the unit cube. $\boldsymbol{x}$ dominates a proportion of other alternatives given by the volume $\prod_{i=1}^{M} x_i$, and is dominated by a proportion $\prod_{i=1}^{M}(1 - x_i)$. Note that if a family of problems is built by generating alternatives uniformly over the consequence domains, then the probability that any particular alternative dominates another decreases exponentially with the number of attributes M. Therefore absolute dominance does not occur with sufficient frequency to be a basis for a practical decision support algorithm. Moreover, in real-world settings, absolutely dominated alternatives would likely be shelved before reaching the conference room.

A relationship that might be more useful is that of ***probable dominance***, which measures the system's beliefs about whether the DM prefers an alternative to another. The probability that alternative $\boldsymbol{x}$ dominates $\boldsymbol{y}$ is denoted $p_{\boldsymbol{xy}}$, where:

$$p_{\boldsymbol{xy}} = \int_{\boldsymbol{w}} \left(\sum_{i=1}^{M} w_i \left[v_i(x_i) - v_i(y_i) \right] \geq 0 \right) p(\boldsymbol{w}) d\boldsymbol{w}. \tag{5}$$

If the system believes that the DM is likely to prefer an alternative over another, perhaps it can recommend them as a candidate pair for practical dominance. Although the DM makes the eventual judgment, the system recommends the pair in the hope that it will simplify the problem. We therefore propose probable dominance above a certain threshold p_T to recognize potential practical dominance, $p_{\boldsymbol{xy}} \geq p_T$.

Let us study the following simple example to compare the occurrence of absolute and probable dominance:

A Two-attribute Example. Suppose $M = 2$ and that the DM's marginal value functions are linear and normalized to between 0 and 1. As a reference, suppose that the DM's trade-offs are accurately captured by weights $w_1 = 0.5$ and $w_2 = 0.5$.

Figure 2 illustrates the regions of absolute and practical dominance with respect to a chosen alternative $\boldsymbol{x} = (0.2, 0.6)$ (represented as a purple dot) when the system believes that $w_1 \sim \text{Uniform}(0, 1)$. Alternatives that absolutely dominate $\boldsymbol{x}$ are shown in dark red, while those that are absolutely dominated by $\boldsymbol{x}$ are shown in dark blue. The regions of potential practical dominance (as determined by probable dominance) for various values of the probability p appear as bands of lighter red and blue, in increments of 0.1, ranging from $p = 0.9$ to 1.0 (almost deep red) down to $p = 0$ to 0.1 (almost deep blue).

Figure 3 illustrates almost the same situation except now the system believes that $w_1 \sim \text{Uniform}(0.4, 0.6)$, possibly by learning from responses to previous queries. It is immediately apparent that the reduced uncertainty yields enlarged regions in which there is a high certainty that $\boldsymbol{x}$ dominates or is dominated. These regions now appear as large triangles flanking the rectangular regions of absolute dominance. From a practical perspective, such regions are effectively equivalent to those of absolute dominance.

Relative to Fig. 2, the region of uncertainty is much more tightly clustered around the diagonal line $x_1 + x_2 = 0.8$ that represents the boundary between $x \succ y$ and $y \succ x$. This is due to the reduced uncertainty about the true value of $\boldsymbol{w}$, and illustrates the benefits of reducing the uncertainty: it allows the system to be more confident in suggesting potential practical dominance to the DM. □

The simple example illustrates that a pair of alternatives chosen at random is more likely to exhibit probable rather than absolute dominance, making it more useful in practice. Further numerical simulations were performed, demonstrating that the probability for a given vector $\boldsymbol{x}$ to practically dominate a given vector $\boldsymbol{y}$ above a given threshold p_T is insensitive to the number of attributes M. This suggests that probable dominance may be a useful concept in practice.

Algorithm 1 summarizes the system's approach to recommending practical dominance and updating beliefs. We assume that the DM responds accurately to this query based on their preferences, since this is a comparison question that is typically associated with low cognitive load. This implies that the polytope of the weight region can be updated to incorporate the following condition:

$$\sum_{i=1}^{M} w_i \left[v_i(x_i) - v_i(y_i)\right] \geq (\leq)\ 0, \qquad (6)$$

depending on whether the user responds 'yes' or 'no' to the question: do you prefer $\boldsymbol{x}$ over $\boldsymbol{y}$? The implications and potential limitations of assuming an accurate response to this query are discussed in the conclusions.

4.2 EVEN SWAPS

Recommending an effective even swap is more challenging than computing practical dominance. In Section 3, we explored some properties, discovering that not all swaps are feasible. The notion of an equalizing even swap was introduced as a practical means of forming a simpler consequence table. To make an equalizing even swap $s(x_i \to y_i, x_j \to x'_j)$, the system needs an alternative pair $\boldsymbol{x}$, $\boldsymbol{y}$ and an attribute pair i, j. Moreover, the system should be able to handle an infeasible swap.

Algorithm 1 Practical Dominance Query

```
**Input:** N alternatives, threshold $p_T$, prior $p(\boldsymbol{w})$
Initialize $p_D^{max} = 0$
**for** each pair of vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ **do**
    Compute $p_{\boldsymbol{xy}}$ from equation (5)
    **if** $p_{\boldsymbol{xy}} \geq \max(p_T, p_D^{max})$ **then**
        Store pair $\boldsymbol{x}, \boldsymbol{y}$; $p_D^{max} = p_{\boldsymbol{xy}}$
    **end if**
**end for**
**if** $p_D^{max} \neq 0$ **then**
    Recommend potential practical dominance for $\boldsymbol{x}, \boldsymbol{y}$, inquiring whether $\boldsymbol{x} \succeq \boldsymbol{y}$
    Update $p(\boldsymbol{w})$ in accordance with DM's response, using equation (6)
**else**
    There is no candidate pair
**end if**
```

We present a heuristic for recommending an even swap that identifies the most suitable alternative and attribute pairs. There are two main steps involved. In the first step, the system identifies alternatives $\boldsymbol{x}$, $\boldsymbol{y}$ where it believes $\boldsymbol{x}$ might be preferred over $\boldsymbol{y}$. It is natural to use probable dominance to quantify this belief. In the second step, the system identifies attributes $i \in N(\boldsymbol{x}, \boldsymbol{y})$ and $j \in D(\boldsymbol{x}, \boldsymbol{y})$ such that swap $s(x_i \to y_i, x_j \to x'_j)$ is likely to decrease $|N(\boldsymbol{x}, \boldsymbol{y})|$.

The intuition behind the heuristic is that an even swap query potentially pushes a pair of alternatives towards dominance of some sort, making it eventually evident to both the system and the DM. Focusing on a pair where one is likely to dominate the other and reducing the non-dominated attribute set ensures that this occurs. As shown in Section 3, an infeasible swap always possesses a feasible conjugate swap, so the heuristic is guaranteed to make progress (from a normative perspective). Note that the cognitive effort expended by the user in trying to respond to the original (failed) swap will be helpful in responding to its conjugate. Note also that the heuristic is myopic in that it attempts to find the 'best' swap at the current moment, without regard to long-term savings. It can be viewed as a dominance-focused heuristic, as it tries to drive alternative pairs towards dominance.

Suppose that the system is considering the equalizing even swap $s(x_i \to y_i, x_j \to x'_j)$ based on the aforementioned heuristic. By definition:

$$v_j(x'_j) = \frac{w_i\left(v_i(x_i) - v_i(y_i)\right) + w_j v_j(x_j)}{w_j}, \qquad (7)$$

if it is feasible, i.e. satisfies Proposition 1.

Algorithm 2 Even Swap Query

```
Input: N alternatives, swap response noise $\delta$, prior $p(\boldsymbol{w})$
Set threshold $p_T = 0$ and find alternative pair $\boldsymbol{x}, \boldsymbol{y}$ from Algorithm 1
Initialize $p_S^{max} = 0$
for each pair of attributes $i$ in $N(\boldsymbol{x}, \boldsymbol{y})$ and $j$ in $D(\boldsymbol{x}, \boldsymbol{y})$ do
    Compute $p_S$ from equation (8)
    if $p_S \geq p_S^{max}$ then
        Store pair $i, j$; $p_S^{max} = p_S$
    end if
end for
Recommend the swap $s(x_i \to y_i, x_j \to x'_j)$
if Response is $x'_j$ then
    Update $p(\boldsymbol{w})$ with conditions from equation (9)
else if DM declares swap is infeasible then
    Recommend conjugate swap $s(x_j \to y_j, x_i \to x'_i)$
    Update $p(\boldsymbol{w})$ using equation (9), after swapping $i$ and $j$
end if
```

Algorithm 3 Bayesian Smart Swaps

```
Input: N alternatives, threshold $p_T$, swap response noise $\delta$, prior $p(\boldsymbol{w})$
while more than 1 solution and 1 active attribute remain in table do
    Remove absolutely dominated solutions, if any
    Mark any attributes with equal consequences across alternatives as inactive, if any
    Identify potential practical dominance using Algorithm 1
    if practical dominance detected then
        Recommend it and update $p(\boldsymbol{w})$ from response
    else
        Recommend an even swap using Algorithm 2 and update $p(\boldsymbol{w})$ from response
    end if
end while
if single attribute remains then
    Find the optimal alternative $\boldsymbol{x}$
end if
Return $\boldsymbol{x}$
```

Suppose that i and j are both attributes where more is preferred to less. Then x_i is increased to y_i for the swap (because $i \in N(\boldsymbol{x}, \boldsymbol{y})$), therefore x_j is decreased to x'_j if the swap is feasible. The probability that this swap will decrease the non-dominated set p_S is:

$$p_S = P(x'_j \geq y_j) = P(v_j(x'_j) \geq v_j(y_j)) = \int_{\boldsymbol{w}} \left(\sum_{k=i,j} w_k \left[v_k(x_k) - v_k(y_k) \right] \geq 0 \right) p(\boldsymbol{w}) d\boldsymbol{w}, \tag{8}$$

where the final step is a result of integration after rearranging (7). For the sake of brevity, we have notationally omitted specifying that p_S is a function of the swap; it should be inferred that it is associated with swap $s(x_i \to y_i, x_j \to x'_j)$. Also, note that although equation (8) applies only when i and j have monotonically increasing marginal value functions, it is easy to generalize it to include all other cases.

Algorithm 2 summarizes the system's approach to recommending even swaps and updating beliefs. Since an even swap is associated with a significant cognitive load, the system treats the response to lie within a noise band measured using the ***swap response noise*** δ. For instance, if a user responds to an even swap query with normalized consequence 0.6 and $\delta = 0.2$, then the system forms a lower bound $L_\delta = 0.5$ and upper bound $U_\delta = 0.7$. This noise band is subject to the other constraints posed on a response, i.e. it must lie within the domain. Therefore a response of 0.05 with $\delta = 0.2$ results in $L_\delta = 0$ and $U_\delta = 0.15$. If more of attribute j is preferred to less, the polytope of the weight region can be updated with conditions from two inequalities involving w_i and w_j:

$$L_\delta \leq \frac{x'_j - x^0_j}{x^*_j - x^0_j} \leq U_\delta, \tag{9}$$

where L_δ and U_δ are bounds on normalized consequences ($L_\delta, U_\delta \in [0, 1]$) that depend on the DM's response and δ as described above, and x'_j is a function of the weights and marginal values as in equation (7). This equation could be easily modified for the case where less of attribute j is preferred.

4.3 HIGH-LEVEL ALGORITHM

Now that we have outlined the two main sub-routines – practical dominance and even swaps – Algorithm 3 provides the high-level routine for our proposed interactive even swaps method. The algorithm identifies absolute dominance and equal attributes, recommends practical dominance when it is confident enough, and recommends an equalizing even swap based on a dominance focused heuristic. The algorithm terminates when the optimal alternative is revealed.

4.4 EXTENSION: UNKNOWN MARGINAL VALUE FUNCTIONS

In the algorithm described here, we assumed that the system already knew the marginal value functions, perhaps through initial assessments. There is a natural extension to the case of unknown marginal value functions along the lines of Mustajoki and Hämäläinen

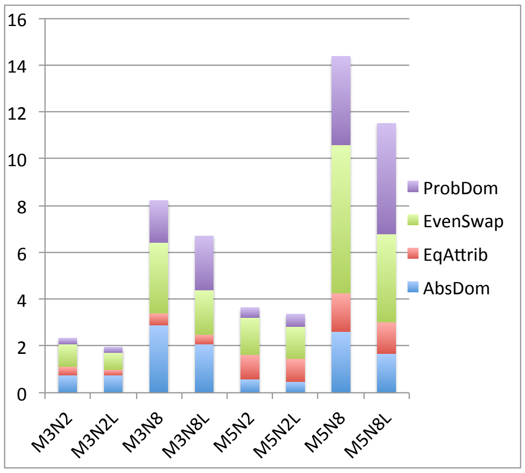


Figure 4: The effect of learning upon the number and type of queries and events. Average number of absolute dominance and equal attribute events, as well as probable dominance and even swap queries per scenario, for $M = \{3, 5\} \times N = \{2, 8\}$, with learning turned on (L) and off.

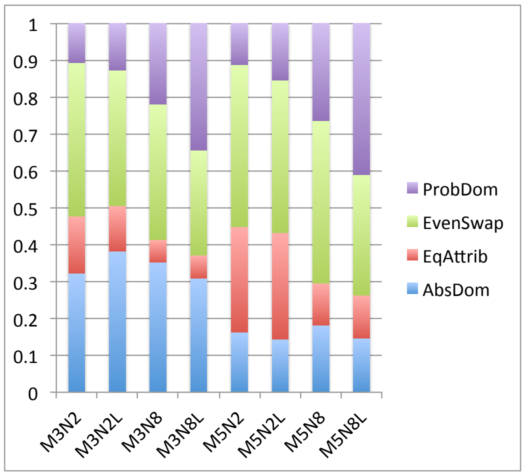


Figure 5: Same data as in Figure 4, except that the scales are normalized to 1 to illustrate the relative contributions of the different types queries and events.

(2005, 2007), using previously determined lower and upper bounds on the marginal value functions. Subsequently, for all probabilistic computations — in this case those pertaining to computing probable dominance and the probability that the swap will decrease the non-dominated set — the system could use probability bounds for making recommendations and update its beliefs based on inequalities from these bounds. The current algorithm could be updated to incorporate these changes.

5 EXPERIMENTAL RESULTS

A first set of experiments were conducted to assess the degree to which learning reduces the number and complexity of queries directed to the DM. The high-level algorithm described in Section 4 was applied to a set of 100 randomly generated scenarios, each involving a randomly generated set of N alternatives with M attributes. Each of the NM values in the consequence table was generated from a Uniform distribution $(0, 1)$. The user's true weights were drawn uniformly from the $(M - 1)$-dimensional unit simplex, and the prior was the same uniform distribution over the simplex. For simplicity, the marginal value functions were assumed to be linear and ranging from 0 to 1.

For each scenario, the probability threshold for a probable dominance query was set relatively high (0.9) to ensure that the queries might not be too onerous for real humans to answer. The probability that $\boldsymbol{x} \succ \boldsymbol{y}$ for the DM was computed by randomly generating at least 10000 weight vectors uniformly in the $(M - 1)$-dimensional unit simplex. First, rejection sampling was employed, i.e. the randomly generated weight vectors were reduced to a set that satisfied any constraints introduced during the interactive process. Then the probable dominance probability was computed as the fraction of weight vectors for which $\boldsymbol{x} \succ \boldsymbol{y}$. Particularly in cases where several even swaps had been applied, the weights were pinned down so precisely that the number of samples satisfying the constraints dropped below 100, in which case more points were generated to ensure that the probable dominance probability was computed from reasonable statistics. To simulate the DM answering a probable dominance or even swaps query, the true weight vector was used to generate the response that the DM would have generated. The DM's noise about the swap value was modeled using a modest swap response noise of $\delta = 0.2$.

For each scenario, the number of absolute dominance and equal-attribute events (accomplished purely through system computations) were recorded. The number of probable dominance and even swap queries (including both regular and conjugate swaps) were recorded as well; these are queries that must be answered by the DM and therefore entail some cognitive burden. Eight sets of 100 scenarios were run, with the number of attributes set to $M = \{3, 5\}$, the number of solutions set to $N = \{2, 8\}$, and the method's learning element both turned on and turned off. The results are summarized in Figure 4.

For the smallest scenarios ($(M, N) = (3, 2)$), an average of just two queries and/or events is required, and typically there is one absolute dominance event and one even swap, with probable dominance and elimination by virtue of equal attributes playing a relatively minor role. Due to the small number of queries and/or events, learning has little impact. The average number of queries and/or events decreases from 2.33 ± 0.11 to 1.96 ± 0.11 — a drop that is of marginal statistical significance. On the other hand, when the number of solutions is increased from 2 to 8, the average number of queries and/or events rises to 8.22 ± 0.34 without learning and 6.7 ± 0.23 with learning — a statistically significant decrease of 18%. When the number of attributes is increased from 3 to 5, a similar trend is observed. For $N = 2$ solutions, the number of queries and/or events is 3.63 ± 0.27 without learning and 3.35 ± 0.24 with learning — an insignificant difference — whereas for $N = 8$ solutions the number of queries and/or events is 14.37 ± 0.57 and 11.51 ± 0.41 — a statistically significant drop of 20%.

Figure 5 provides another view of the same data, in which the relative contribution of the various types of queries and events is obtained by normalization. As anticipated, the impact of absolute dominance decreases as the number of solutions N increases from 2 to 8. This is a consequence of the exponential decrease in the probability for any given vector to absolutely dominate another with the number of attributes. Another trend evident here is that as N increases, the relative impact of probable dominance queries grows stronger. Moreover, for larger problems, the effect of learning is to further increase the relative importance of probable dominance over even swap queries.

Having established that learning can substantially reduce the number of queries and/or events required to identify the optimal alternative, and moreover that it shifts the balance more from even swaps to probable dominance queries as the problem size grows, a second series of experiments were conducted with learning turned on. The objective of these experiments was to chart in greater detail how the number and type of queries and/or events change as the number of attributes and alternatives are varied. The results depicted in Figure 6 demonstrate the same basic trends, including the waning importance of absolute dominance as the number of attributes M grows and the ascendancy of probable dominance as N grows.

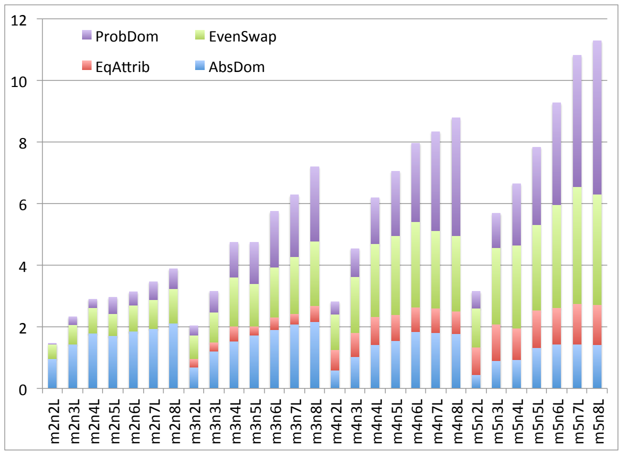


Figure 6: The effect of M and N on the number and type of queries and events. Average number of queries/events of each type, from left to right, for $M = \{2, 3, 4, 5\}$ and $N = \{2, 3, 4, 5, 6, 7, 8\}$.

6 CONCLUSIONS

In this paper, we have presented a method for guiding the DM through the even swaps process using an overall Bayesian approach with a dominance focused heuristic. We have demonstrated through experiments that one can effectively learn about the DM's preferences in the course of a single session to guide them quickly to a final choice. A potential next step is to implement a tool and test its efficacy through experiments with real human subjects. Belton et al. (2005) conduct some user experiments involving even swap queries but such studies remain few and far between.

Our approach appears to be practical for modest-sized decision problems ($N < 10$). One could argue that direct elicitation techniques might be appropriate for large N (~ 100); however, if the DM prefers to use even swaps (for reasons highlighted in Section 2), it may be prudent to focus on learning the DM's preferences rather than myopically trying to find the most likely pair of alternatives such that one might dominate the other.

Here we used a simple model for incorporating potential noise in a DM's response to an even swap query; it was chosen to enable conditions represented as inequalities. In future research, we envision more nuanced noise models accounting for cognitive effects such as attribute conflict (e.g. Fischer et al. 2000, Delquié 2003). Also, although we have used probable dominance to measure practical dominance, it remains unclear how the metric would work for large problems because it may be difficult for a DM to compare any two arbitrary alternatives; note that comparison queries are also subject to various cognitive biases in general (e.g. Tversky et al. 1988, Tversky and Kahneman 1991). Finally, regarding the computations for simulation, rejection sampling is sufficient when only a few queries are asked in a single setting. If several queries are asked back-to-back, efficient methods such as hit and run sampling may be more effective.

Acknowledgments

We thank three anonymous reviewers for their valuable feedback.

References

R. M. Anderson and B. F. Hobbs (2002) Using a Bayesian approach to quantify scale compatibility bias. *Management Science*, **48**(12):1555–1568.

V. Belton and T. Stewart (2002) *Multiple Criteria Decision Analysis: An Integrated Approach.* Norwell, MA: Kluwer Academic Publishers.

V. Belton, G. Wright and G. Montibeller (2005) When is swapping better than weighting? An evaluation of the even swaps method in comparison with multi attribute value analysis. Working Paper, 2005/19, Department of Management Science, University of Strathclyde, UK.

C. Boutilier (2002) A POMDP formulation of preference elicitation problems. In *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI)*, 239–246, AAAI Press.

U. Chajewska, D. Koller and R. Parr (2000) Making rational decisions using adaptive utility elicitation. In *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI)*, 363–369, AAAI Press.

P. Delquié (1993) Inconsistent trade-offs between attributes: New evidence in preference assessment biases. *Management Science*, **39**(11):1382–1395.

P. Delquié (1997) "Bi-matching": A new preference assessment method to reduce compatibility effects. *Management Science*, **43**(5):640–658.

P. Delquié (2003) Optimal conflict in preference assessment. *Management Science*, **49**(1):102–115.

J. Eliashberg and J. Hauser (1985) A measurement error approach for modeling consumer risk preference. *Management Science*, **15**(1):1–25.

G. W. Fischer, J. Jia and M. F. Luce (2000) Attribute conflict and preference uncertainty: The RandMAU model. *Management Science*, **46**(5):669–684.

J. S. Hammond, R. L. Keeney and H. Raiffa (1998) Even swaps: A rational method for making trade-offs. *Harvard Business Review*, **76**(2):137–149.

J. S. Hammond, R. L. Keeney and H. Raiffa (1999) *Smart Choices: A Practical Guide to Making Better Decisions.* Harvard Business School Press.

H. B. Jimison, L. M. Fagan, R. D. Shachter and E. H. Shortliffe (1992) Patient-specific explanation in models of chronic disease. *Artificial Intelligence in Medicine*, **4**(3):191-205.

M. Kajanus, J. Ahola, M. Kurttila and M. Pesonen (2001) Application of even swaps for strategy selection in a rural enterprise. *Management Decision*, **39**(5):394–402.

R. L. Keeney and H. Raiffa (1976) *Decisions with Multiple Objectives: Preferences and Value Tradeoffs.* New York, NY: John Wiley and Sons, Inc.

S. Lichtenstein and P. Slovic (2006) *The Construction of Preference.* Cambridge, UK: Cambridge University Press.

J. Mustajoki and R. P. Hämäläinen (2005) A preference programming approach to make the even swaps method even easier. *Decision Analysis*, **2**(2):110–123.

J. Mustajoki and R. P. Hämäläinen (2007) Smart-Swaps - A decision support system for multicriteria decision analysis with the even swaps method. *Decision Support Systems*, **44**:313–325.

K. L. Poh and E. J. Horvitz (1993) Reasoning about the value of decision-model refinement: Methods and application. In *Proceedings of the 9th Conference on Uncertainty in Artificial Intelligence (UAI)*, 174–182, Morgan Kauffman.

G. C. Scott and R. D. Shachter (2005) Individualizing generic decision models using assessments as evidence. *Journal of Biomedical Informatics*, **38**(4):281–297.

T. Stewart (1996) Robustness of additive value function methods in MCDM. *Multi-Criteria Decision Analysis*, **5**(4):301–309.

A. Tversky and D. Kahneman (1991) Loss aversion in riskless choice: A reference-dependent model. *The Quarterly Journal of Economics*, **106**(4):1039–1061.

A. Tversky, S. Sattath and P. Slovic (1988) Contingent weighting in judgment and choice. *Psychological Review*, **95**(3):371–384.

D. von Winterfeldt and W. Edwards (1986) *Decision Analysis and Behavioral Research.* Cambridge, UK: Cambridge University Press.

M. C. Willemsen and G. Keren (2003) The meaning of indifference in choice behavior: Asymmetries in adjustments embodied in matching. *Organizational Behavior and Human Decision Processes*, **90**(2):342–359.