Incremental elicitation of choquet capacities for multicriteria choice, ranking and sorting problems

Nawal Benabbou, Patrice Perny, Paolo Viappiani

Incremental Elicitation of Choquet Capacities for Multicriteria Choice, Ranking and Sorting Problems[☆]

Nawal Benabbou, Patrice Perny*, Paolo Viappiani

Sorbonne Universités, UPMC Univ Paris 06, UMR 7606, LIP6
CNRS, UMR 7606, LIP6, F-75005, Paris, France
4 Place Jussieu, 75005 Paris, France

Abstract

This paper proposes incremental preference elicitation methods for multicriteria decision making with a Choquet integral. The Choquet integral is an evaluation function that performs a weighted aggregation of criterion values using a capacity function assigning a weight to any coalition of criteria, thus enabling positive and/or negative interactions among them and covering an important range of possible decision behaviors. However, the specification of the capacity involves many parameters which raises challenging questions, both in terms of elicitation burden and guarantee on the quality of the final recommendation.

In this paper, we investigate the incremental elicitation of the capacity through a sequence of preference queries (questions) selected one-by-one using a minimax regret strategy so as to progressively reduce the set of possible capacities until the regret (the worst-case "loss" due to reasoning with only partially specified capacities) is low enough. We propose a new approach designed to efficiently compute minimax regret for the Choquet model and we show how this approach can be used in different settings: 1) the problem of recommending a single alternative, 2) the problem of ranking alternatives from best to worst, and 3) sorting several alternatives into ordered categories. Numerical experiments are provided to demonstrate the practical efficiency of our approach for each of these situations.

Keywords: multicriteria decision making, Choquet integral, capacity, incremental elicitation, minimax regret, choice, ranking, sorting.

[☆]This paper is an extension of work published at ECAI 2014 [1].

*Principal Corresponding Author: Patrice Perny, Sorbonne Universités, UPMC, LIP6, 4 place Jussieu, 75252 PARIS CEDEX 05, France

Email addresses: nawal.benabbou@lip6.fr (Nawal Benabbou), patrice.perny@lip6.fr (Patrice Perny), paolo.viappiani@lip6.fr (Paolo Viappiani)

1. Introduction

Preferences are pervasive in Artificial Intelligence. In particular, they are used to specify goals or desiderable behaviors for autonomous agents, and to provide automatic recommendations, adapted to the user, in decision-support tools. In various decision contexts, the quality of alternatives is assessed with respect to multiple criteria, possibly conflicting each other. An aggregation function is often used to compare alternatives evaluated on multiple criteria by synthesizing their performances into overall utility values. In this context, there is a need of elicitation methods aiming at specifying preference parameters modeling the relative importance of criteria for the Decision Maker (DM) and possibly their interactions.

Aggregation functions must be sufficiently expressive to fit the DM's preferences, allowing for instance the determinination of his/her preferred alternative. The Choquet integral defines a family of non-linear aggregators that are really appealing for preference modeling. Decision models based on Choquet integrals have been initially introduced in the context of uncertainty with the Choquet Expected Utility theory [2]. The Choquet integral is also used in the field of decision-making under risk (probabilistic uncertainty), for example in the Yaari's model [3] or in the Rank Dependent Utility model proposed by Quiggin [4], to overcome some descriptive limitations of Expected Utility. Choquet integrals are also very popular in the field of multicriteria decision making [5, 6] because they enable to model different kinds of interactions between criteria and include many aggregators as special cases (e.g. linear additive models, min, max and any other order statitistics, leximin and leximax, OWA and WOWA [7, 8]).

The Choquet integral has received much attention in the last decades and is now widely used in practical decision making [9]. Its usefulness has been established in various domains of artificial intelligence. For example, in machine learning, the use of Choquet integrals provides higher predictive capacities than linear models, while offering measures for quantifying the importance of individual predictor variables and the interaction between groups of variables [10]. Moreover, in recommender systems [11], the advantage provided by Choquet integrals is to allow positive and negative synergies between criteria, with enhanced descriptive and prescriptive possibilities. Similarly, in multiagent decision making [12], the Choquet integral is used to aggregate individual preferences using a possibly non-additive measure of the importance of agent coalitions, which allows one to model various notions of social welfare. In information fusion [13], the use of Choquet integrals allows one to model positive or negative reinforcements among sets of observations. Finally, in multiobjective state-space search [14], the use of Choquet integrals allows one to find compromise solutions that could not be obtained using linear aggregators.

However, to compute overall utility values using a Choquet integral, decision support systems need to be able to assess the model's parameters according to DM's preferences. These parameters used to capture the value system of the

DM are characterized by a capacity function defining the weight attached to every subset of criteria. Therefore, they are in exponential number relatively to the number of criteria and their elicitation is a challenging issue. Most of the works aiming at determining a suitable capacity for the Choquet integral consider a static preference database as input, and focus on the determination of capacity values that best fit the available preferences. For example, one can minimize a quadratic error between Choquet values and target utility values prescribed by the DM on a sample of reference alternatives. Alternatively, one can impose some constraints on Choquet values to enforce the decision model to be compatible with a partial or total order available on a subset of alternatives. These approaches are illustrated in many papers see, e.g. [15, 16, 17, 18, 19, 20, 21] and [22] (Chapter 11), some of them being implemented in decision support softwares such as TOMASO [23] and MYRIAD [24].

Departing from these standard approaches, we are proposing an incremental elicitation process for the Choquet integral; in this process, informative preference queries are selected one at the time in order to progressively reduce the set of admissible capacities until a robust recommendation can be made. This active learning process could be continued as long as some uncertainty remains in the specification of the capacity but a complete determination of the model is generally not necessary to make a decision. Moreover, it would require a prohibitive amount of preference queries. Instead, we iterate queries until what we know of the capacity is enough to formulate a recommendation (e.g. a choice, ranking or sorting of the alternatives). The elicitation process is stopped when it can be proved that further specifications of the model cannot seriously challenge the current recommendation.

This approach relies on and extends previous works on the incremental elicitation of linear utility functions, going back to the ISMAUT method [25] and more recently, strategies developed within the artificial intelligence community for preference query selection using the minimax-regret criterion [26, 27, 28, 29]. Regret-based elicitation has been successfully demonstrated with real users in a prototype for decision support (UTPref) and validated in a user study [30]. Adaptation of minimax regret elicitation strategies to Choquet models is not obvious, as in general the number of constraints required to impose that the parameters of the model are valid is exponential in the number of criteria. In this paper, we propose efficient algorithms that avoid this issue by focusing on specific (but intuitive) types of preference queries. Our approach can be used for standard choice problems (where one single alternative is suggested to the DM) but also for ranking and sorting problems. We now briefly review the main differences between these three problem settings.

Choosing is the problem setting where one alternative has to be selected and recommended to the DM. We thoroughly discuss the elicitation of Choquet capacities for choice problems in Section 3. In particular, we discuss computational issues related to minimax regret optimization (Section 3.1, 3.2, 3.3), strategies for generating relevant queries within an incremental elicitation process (Section 3.4) and evaluations with numerical tests (Section 3.5).

Ranking is the problem setting where different alternatives have to be ranked from best to worst. We consider here a special case of ranking, constructed by repeated choices. As we focus on this particular ranking model, we will discuss it within Section 3 devoted to choice problems (see Section 3.6).

Sorting is the problem setting where alternatives need to be assigned to different categories ordered from best to worst. The categories are assumed to be defined a-priori and the assignment of alternatives is based on an *intrinsic* evaluation. Sorting (also called *multipartite ranking* and *instance ranking* in [31]) is used, for instance, in order to assess financial credit demands or to assign specific distinctions to individuals. In this paper, we consider sorting methods with thresholds: one assumes that the utility scale is divided into intervals and assignments are made by looking in which interval the utility scores fall (see Section 4).

In this paper, we propose elicitation methods for each setting and evaluate, with numeric simulations, their practical efficiency.

2. Background and notations

Let X be the set of alternatives (items, products, candidates...) that need to be compared. Any alternative $x \in X$ is evaluated with respect to a set of n criteria denoted $N = \{1, \ldots, n\}$, and is characterized by a performance vector $(x_1, \ldots, x_n)$ where $x_i \in [0, 1]$ represents the utility of x with respect to criterion i for all $i \in N$. We assume that the same utility scale is used for all components so that x_i and x_j can be compared when $i \neq j$. Let us now recall the definitions of Choquet capacities and discrete Choquet integrals. For simplicity, x will indifferently denote the alternative or its performance vector.

2.1. Discrete Choquet integrals

For any alternative $x \in X$, let $(.)$ denote a permutation of $\{1, \ldots, n\}$ that sorts the components of x by increasing order [1], i.e. $x_{(i)} \leq x_{(i+1)}$ for $i \in [\![1, n-1]\!]$.

Definition 1. *The i^{th} level set of x, denoted by $X_{(i)}$, is defined as $X_{(i)} = \{(i), \ldots, (n)\}$.*

Note that $X_{(i+1)} \subseteq X_{(i)}$ for all $i \in [\![1, n-1]\!]$ by definition.

Definition 2. *A normalized Choquet capacity v is a real-valued set-function defined on 2^N such that:*

- $v(\emptyset) = 0$, $v(N) = 1$ (normalization constraints)
- $v(A) \leq v(B)$ *for all* $A \subseteq B \subseteq N$ (monotonicity constraints)

[1] In case of ties among criteria, multiple permutations are possible to sort the components in ascending order. However, the definitions given below are such that, no matter which permutation is chosen, the result is the same.

where $v(A)$ represents the weight attached to coalition A, for any $A \subseteq N$. Note that the capacity value of the first level set of any alternative x is one (i.e. $v(X_{(1)}) = v(N) = 1$) due to normalization constraints; monotonicity constraints can be rewritten as follows: $v(A) \leq v(A \cup \{i\})$ for all $A \subset N$ and all $i \in N \backslash A$.

Definition 3. *The Choquet integral value of alternative $x \in X$ is defined as:*

$$C_v(x) = \sum_{i=1}^{n} \left[x_{(i)} - x_{(i-1)}\right] v(X_{(i)}) \text{ with } x_{(0)} = 0 \tag{1}$$

$C_v(x)$ represents the overall utility of alternative x. Therefore, alternative x is as least as good as y whenever $C_v(x) \geq C_v(y)$.

Example 1. *Consider a problem defined on 3 criteria, i.e. $N = \{1, 2, 3\}$, two alternatives $x = (0.7, 0.6, 1)$ and $y = (0.8, 1, 0.6)$ and the following capacity v defined on $2^{\{1,2,3\}}$:*

	$\emptyset$	$\{1\}$	$\{2\}$	$\{3\}$	$\{1,2\}$	$\{1,3\}$	$\{2,3\}$	$\{1,2,3\}$
v	0	0.1	0.2	0.3	0.5	0.6	0.7	1

Alternatives x and y induce the following level sets:

- $X_{(1)} = \{1, 2, 3\}$, $X_{(2)} = \{1, 3\}$ *and* $X_{(3)} = \{3\}$;
- $Y_{(1)} = \{1, 2, 3\}$, $Y_{(2)} = \{1, 2\}$ *and* $Y_{(3)} = \{2\}$.

The computation of the Choquet value of x and y according to capacity v gives:

$$C_v(x) = 0.6v(\{1, 2, 3\}) + (0.7 - 0.6)v(\{1, 3\}) + (1 - 0.7)v(\{3\}) = 0.75$$
$$C_v(y) = 0.6v(\{1, 2, 3\}) + (0.8 - 0.6)v(\{1, 2\}) + (1 - 0.8)v(\{2\}) = 0.74$$

Hence we have $C_v(x) > C_v(y)$, meaning that x is strictly preferred to y.

In multicriteria decision-making, one needs to ensure that $C_v(x) \geq C_v(y)$ whenever x weakly Pareto-dominates y (i.e. $x_i \geq y_i$ for all $i \in N$). This property holds due to the monotonicity of v with respect to set inclusion, as can be seen from the following equivalent formulation of the Choquet integral:

$$C_v(x) = \sum_{i=1}^{n} x_{(i)} \left[v(X_{(i)}) - v(X_{(i+1)})\right] \text{ with } X_{(n+1)} = \emptyset$$

Due to monotonicity, we have $v(X_{(i)}) - v(X_{(i+1)}) \geq 0$ for all $i \in \{1, \ldots, n\}$ which guarantees that $C_v(x)$ cannot decrease as quantities x_i increase. In many papers on multicriteria optimization with a Choquet integral, the capacity is assumed to be given [32, 33, 34, 35]. This assumes that preference elicitation methods are available to determine the capacity that best fits DM's preferences. The aim of this paper is to introduce an incremental approach for the elicitation

of the capacity in the Choquet integral, following interactive elicitation schemes proposed in [25, 28, 29] for simpler utility models.

In this approach, the elicitation task is seen as a game played with the DM. At every step of the elicitation process, the system generates a preference query, and then the DM reveals a piece of his/her actual preferences. The answer provides new constraints on the set of admissible capacities thus reducing the uncertainty attached to the capacity and therefore to the Choquet values. In this process, both the problem of selecting the next query and the one of generating a recommendation are seen as a decision problem under uncertainty, where the uncertainty is due to the imperfect knowledge of preference parameters (here the capacity). The selection of the query is made so that an effective regret reduction is guaranteed whatever the answer is.

We recall now the standard background on incremental preference elicitation based on the minimax regret criterion (Sections 2.2 and 2.3). This approach is intended for choice problems in which a parameterized utility function f_θ is to be maximized, parameter θ being imprecisely known. Then, we will further specify this approach for the Choquet integral in Section 3, first in the setting of choice and then in the setting of ranking. Finally, we will propose in Section 4 an incremental elicitation method based on an alternative definition of regrets better suited to sorting problems.

2.2. Choice based on the minimax regret criterion

The minimax regret is a decision criterion classically used for optimization under uncertainty [36, 37]; it has been more recently advocated for use in decision-making where the uncertainty is over utility values [29, 38]. Assume that DM's preferences can be modeled by a parameterized utility function f_θ where θ denotes the parameter of this aggregation function. In this setting, alternative $x \in X$ is preferred to alternative $y \in X$ if and only if $f_\theta(x) \geq f_\theta(y)$. We assume here that θ is not known precisely and can initially take any value in an uncertainty set denoted Θ. Let $\mathcal{P}$ be the set gathering some pairs of alternatives (a, b) such that a is preferred to b by the DM. Let $\Theta_\mathcal{P}$ be the subset of Θ containing all parameters θ consistent with information $\mathcal{P}$, i.e. such that $f_\theta(a) \geq f_\theta(b)$ for all $(a, b) \in \mathcal{P}$. The problem is now to determine the most promising alternative under parameter uncertainty $\Theta_\mathcal{P}$. To this end, the minimax regret approach is based on the following definitions.

Definition 4. *The pairwise max regret (PMR) of alternative x with respect to the alternative y is defined as:*

$$\mathrm{PMR}(x, y, \Theta_\mathcal{P}) = \max_{\theta \in \Theta_\mathcal{P}} \Big\{ f_\theta(y) - f_\theta(x) \Big\}$$

In other words, the pairwise max regret of x with respect to y represents the worst-case loss when recommending x instead of y. We can establish a link with the notion of possible and necessary preferences used in other papers (see e.g. [18]): remark that $\mathrm{PMR}(x, y, \Theta_\mathcal{P}) < 0$ means that x is necessarily better than

y whereas $\mathrm{PMR}(x, y, \Theta_{\mathcal{P}}) > 0$ means that y is possibly better than x. Finally $\mathrm{PMR}(x, y, \Theta_{\mathcal{P}}) = 0$ means that x is necessarily as least as good as y.

Definition 5. *The max regret (MR) of alternative $x \in X$ is defined as:*

$$\mathrm{MR}(x, X, \Theta_{\mathcal{P}}) = \max_{y \in X} \mathrm{PMR}(x, y, \Theta_{\mathcal{P}})$$

The max regret of x is the worst-case loss when recommending x instead of one of the adversary's choices (i.e. any element of $\arg\max_{y \in X} \mathrm{PMR}(x, y, \Theta_{\mathcal{P}})$).

Definition 6. *The minimax regret (mMR) is defined as:*

$$\mathrm{mMR}(X, \Theta_{\mathcal{P}}) = \min_{x \in X} \mathrm{MR}(x, X, \Theta_{\mathcal{P}})$$

An optimal solution for the minimax regret criterion is an alternative that achieves the minimax regret (i.e. any element of $\arg\min_{x \in X} \mathrm{MR}(x, X, \Theta_{\mathcal{P}})$). Recommending such an alternative allows one to guarantee that the worst-case loss is minimized.

To determine the best alternative according to the minimax regret criterion, we have to compute PMR for all ordered pairs of distinct alternatives. However, the computational effort can be significantly reduced by using standard pruning rules for min aggregators, as shown in [39] (but of course, the number of PMR computations remains quadratic in the worst-case). When f_θ is a linear or piecewise linear utility function, then $\Theta_{\mathcal{P}}$ is defined by a number of linear constraints approximately equal to the size of $\mathcal{P}$. In that case, PMR-optimizations (and therefore mMR) can be performed exactly quite efficiently by solving simple linear programs. This approach is more tractable than probabilistic models of utility that rely on Bayesian updates that are computationally expensive [27, 26].

2.3. Incremental elicitation for choice problems

Given a particular set of preference statements $\mathcal{P}$, the worst-case loss ensured by the minimax regret criterion might still be at an unacceptable level. By considering additional preferences statements (inducing constraints on the set of admissible parameters), this loss may be decreased. Indeed, we know that, for any set of preference statements $\mathcal{P}' \supseteq \mathcal{P}$, we have $\Theta_{\mathcal{P}'} \subseteq \Theta_{\mathcal{P}}$; therefore, for any $x, y \in X$, we have:

$$\mathrm{PMR}(x, y, \Theta_{\mathcal{P}'}) \leq \mathrm{PMR}(x, y, \Theta_{\mathcal{P}}) \tag{2}$$

$$\mathrm{MR}(x, X, \Theta_{\mathcal{P}'}) \leq \mathrm{MR}(x, X, \Theta_{\mathcal{P}}) \tag{3}$$

$$\mathrm{mMR}(X, \Theta_{\mathcal{P}'}) \leq \mathrm{mMR}(X, \Theta_{\mathcal{P}}) \tag{4}$$

Hence, the minimax regret cannot increase by adding new preference statements; in practice, it often strictly decreases (see [39], pp. 194-202). Therefore, the minimax regret criterion can be used within an incremental elicitation process that progressively asks preference queries to the DM until the minimax regret drops under a given tolerance threshold $\delta \geq 0$. At that time, recommending

any optimal alternative for the mMR criterion ensures that the loss incurred by not choosing the true optimal alternative is bounded above by that threshold.

Different types of queries can be used when designing such an incremental elicitation process. Comparison queries are relatively simple; they require the DM to compare a pair of alternatives and state which one is preferred. Notice however that some queries are more informative than others (e.g. the minimax regret will not decrease when asking to compare an alternative with another that Pareto-dominates the former). Thus, it is important to focus on relevant queries so as to make a good recommendation without asking too many queries. A notion of myopic value of information can be used [40] to evaluate the relevance of a query. Let $\mathcal{Q}$ denote the set of all queries under consideration.

Definition 7. *The worst-case minimax regret (WmMR) of a query $q \in \mathcal{Q}$ is:*

$$\mathrm{WmMR}(q, \Theta_{\mathcal{P}}) = \max_{p \in \mathcal{P}_q} \mathrm{mMR}(X, \Theta_{\mathcal{P} \cup \{p\}})$$

where $\mathcal{P}_q$ denotes the set of all possible answers to the query q.

Hence the next query of the elicitation process should be chosen in:

$$\arg\min_{q \in \mathcal{Q}} \mathrm{WmMR}(q, \Theta_{\mathcal{P}})$$

because any optimal solution for the WmMR criterion ensures the best reduction of minimax regret in the answer's worst-case scenario. Note that computing the optimal query for WmMR can be computationally intensive when set $\mathcal{Q}$ under consideration is too large. In that case, one may consider a very efficient query selection strategy (though not optimal in general) called the Current Solution Strategy (CSS) [29] consisting in asking the DM to compare, at each iteration step, an optimal solution x^* for the mMR criterion with its adversarial choice y^* (arbitrarily chosen in $\arg\max_{y \in X} \mathrm{PMR}(x^*, y, \Theta_{\mathcal{P}})$). This elicitation scheme has been successfully used in various contexts, see e.g. [29, 30, 41, 42, 43].

In this paper, we adopt an incremental elicitation procedure based on minimax regret in order to acquire the most relevant information about the Choquet capacity representing the DM's preferences (capacity v taking the role of parameter θ for the Choquet integral C_v). This is done with different, but related, goals: recommending a single alternative, providing a ranking of alternatives and sorting the alternatives by assigning them to predefined ordered categories.

3. Choice and ranking with a Choquet integral

In this section, we focus our discussion on the Choquet integral model; in that case, $\Theta_{\mathcal{P}}$ is the set of all capacities v compatible with $\mathcal{P}$. Any preference statement of type "a is preferred to b" will be interpreted as a constraint of type $C_v(a) \geq C_v(b)$ restricting the initial admissible set of capacities Θ, where C_v is defined by Equation (1). It is important to note that, although $C_v(a)$ is a non-linear function of a for fixed v (e.g. $C_v(a+b) \neq C_v(a) + C_v(b)$ in general),

$C_v(a)$ is linear in v for fixed a (since the permutation of $\{1,\ldots,n\}$ is also fixed). Hence constraint $C_v(a) \geq C_v(b)$ is linear in v for any fixed pair $(a,b) \in \mathcal{P}$. Thus, any preference statement of type "a is preferred to b" will be translated as a linear constraint bounding the set of admissible capacities. As a consequence, the set $\Theta_{\mathcal{P}}$ of admissible capacities under information $\mathcal{P}$ is a convex polyhedron.

In the following subsections, we first address the computation of pairwise max regrets PMR on this polyhedron with (Sections 3.1 et 3.2) or without (Section 3.3) Linear Programming. Then, we provide an efficient query generation strategy enabling the fast determination of the best alternative for the DM (Section 3.4). Finally, we consider the problem of providing a ranking of all alternatives (Section 3.6).

3.1. LP formulations of the PMR*-optimization problem*

Let $v : 2^N \to \mathbb{R}$ be a set-function and v_A be the decision variable representing $v(A)$ for any $A \subseteq N$. Using this notation, v will indifferently denote the set-function or the vector composed of its values in any arbitrary order. Since Choquet integrals are linear with respect to their capacity values, the computation of $\text{PMR}(x, y, \Theta_{\mathcal{P}})$ for any alternatives $x, y \in X$ can be performed by solving the linear program denoted by LP_1 below:

$$
\begin{aligned}
& \max_{v} && C_v(y) - C_v(x) && (5)\\
& s.t. && v_\emptyset = 0 && (6)\\
(\text{LP}_1) \qquad & && v_N = 1 && (7)\\
& && v_A \leq v_{A\cup\{i\}},\ \forall A \subset N,\ \forall i \in N\backslash A && (8)\\
& && C_v(a) \geq C_v(b),\ \forall (a,b) \in \mathcal{P} && (9)
\end{aligned}
$$

Equations (6-8) ensure that v is a normalized capacity (see Definition 2) and Equation (9) ensures that v is compatible with $\mathcal{P}$. Thus, for Choquet integrals, the computation of PMR involves exponentially many variables and monotonicity constraints (8). Note however that, for 2-additive capacities[2], the number of such constraints that are actually needed is polynomial [19] in the number of criteria. However, these subclasses correspond to specific attitudes that do not necessarily match with the observed preferences. Hence, we investigate now the general case without any prior restriction on the admissible set of capacities.

We now introduce a more compact linear programming formulation for the problem modeled by LP_1. For any two performance vectors x and y, let $\mathcal{A}_{(x,y)}$ be the set of all level sets of x and y (see Definition 1), i.e.

$$\mathcal{A}_{(x,y)} = \{X_{(i)}, i \in N\} \cup \{Y_{(i)}, i \in N\}.$$

[2] A capacity v is said to be 2-additive on N when there exist $n(n+1)/2$ coefficients $m_B, B \subseteq N, |B| \leq 2$ such that $v(A) = \sum_{B\subseteq A:|B|\leq 2} m_B$ for all $A \subseteq N$.

Note that sets belonging to $\mathcal{A}_{(x,y)}$ are the only ones that appear in the objective function of PL_1. As a consequence, the objective function involves at most $2n-1$ variables. This specificity can be exploited to simplify the optimization problem associated to the computation of pairwise max regrets. In this paper, the simplification is achieved by limiting the form of preference statements. More precisely, let us consider queries involving the following two types of fictitious alternatives:

- Binary alternatives of type $1A0$, where $1A0$ is an alternative with a top performance on all criteria in $A \subseteq N$ and a bottom one on all others. For example, $1A0 = (1, 0, 1, 0, 0)$ when $A = \{1, 3\}$ and $n = 5$.
- Constant utility profiles of type $\Lambda = (\lambda, \ldots, \lambda) \in [0, 1]^n$.

Note that, for any normalized capacity v and any set $A \subseteq N$, we have:

$$C_v(1A0) = v(A)$$

This is due to the fact that all "bottom" criteria precede all "top" criteria in the permutation, and so the difference of performance between any two successive criteria is equal to zero, except for the last "bottom" criterion and the first "top" criterion, which is equal to one (refer to Definition 3). Moreover, for any normalized capacity v and any $\Lambda \in [0, 1]^n$, we have:

$$C_v(\Lambda) = C_v((\lambda, \ldots, \lambda)) = \lambda$$

This can easily be explained by remarking that the difference of performances between any two successive criteria is zero, except for that between the first criterion and the auxiliary criterion $x_{(0)} = 0$.

We now consider preference queries where the DM is asked to compare a binary alternative with a constant utility profile:

- if $1A0$ is preferred to Λ, then Equation (9) gives the simple constraint $v_A \geq \lambda$, imposing a lower bound on $v(A)$.
- if Λ is preferred to $1A0$, then Equation (9) induces the constraint $v_A \leq \lambda$, imposing an upper bound on $v(A)$.

Consequently, Equation (9) can be replaced by boundary constraints over decision variables; indeed, to ensure that the set-function v is compatible with $\mathcal{P}$, it is sufficient to update the boundaries of an interval $[l_A, u_A]$ whenever a preference involving $1A0$ is inserted in $\mathcal{P}$; note that $l_\emptyset = u_\emptyset = 0$, $l_N = u_N = 1$ due to the normalization constraint, and initially, we have $[l_A, u_A] = [0, 1]$ for all proper subsets A of N.

Moreover, since $\Theta_{\mathcal{P}\cup\{(1A0,\Lambda)\}}$ is the set of all capacities $v \in \Theta_\mathcal{P}$ that satisfy $v(A) \geq \lambda$, and keeping in mind that all capacities are monotonic by definition, we necessarily have $v(B) \geq \lambda$ for all $B \supseteq A$, that means:

$$\Theta_{\mathcal{P}\cup\{(1A0,\Lambda)\}} = \Theta_{\mathcal{P}\cup\{(1B0,\Lambda), B\supseteq A\}}$$

Similarly, since $\Theta_{\mathcal{P}\cup\{(\Lambda,1A0)\}}$ is the set of all capacities $v \in \Theta_{\mathcal{P}}$ such that $v(A) \leq \lambda$, we necessarily have $v(B) \leq \lambda$ for all $B \subseteq A$, that means:

$$\Theta_{\mathcal{P}\cup\{(\Lambda,1A0)\}} = \Theta_{\mathcal{P}\cup\{(\Lambda,1B0),B\subseteq A\}}$$

Therefore, for any $A \subseteq N$, $\lambda \in [l_A, u_A]$, we perform the following updates:

- if the preference $(1A0, \Lambda)$ is inserted in $\mathcal{P}$, meaning that $1A0$ is preferred to Λ, then we set $l_A = \lambda$ and $l_B = \max\{l_B,\ \lambda\}$ for all $B \supset A$;
- if the preference $(\Lambda, 1A0)$ is inserted in $\mathcal{P}$, meaning that Λ is preferred to $1A0$, then we set $u_A = \lambda$ and $u_B = \min\{u_B,\ \lambda\}$ for all $B \subset A$.

It is important to note that restricting the interaction to comparison queries involving a binary alternative $1A0$ and a constant utility profile Λ is sufficient to elicit all preferences of the DM. The constraints derived from the answers indeed allow us to approximate the capacity values $v(A), A \subseteq N$, as close as we want. Hence, the knowledge of v on every subset of criteria completely determines the Choquet integral representing the DM's preferences over the entire set of alternatives. Note also that, by construction, we have $l_A \leq l_B$ and $u_A \leq u_B$ for all $A \subset B$. Then, the following proposition holds:

Proposition 1. *Consider intervals $[l_A, u_A]$, $A \subseteq N$. Assume that $l_A \leq l_B$ and $u_A \leq u_B$ hold for all $A \subset B \subseteq N$. Then, for any $\mathcal{A} \subset 2^N$, the subnetwork of monotonicity constraints $v_A \leq v_B$, where $A, B \in \mathcal{A}$ and $A \subseteq B$, is* globally consistent *[44], i.e. any partial instantiation of variables $v_A, A \in \mathcal{A}$, which is locally consistent (w.r.t. boundary and monotonicity constraints) can be extended to a consistent instantiation of all variables $v_A, A \subseteq N$.*

Proof. Let $\mathcal{A} \subset 2^N$ and let I be an instantiation of all variables in $\{v_A : A \in \mathcal{A}\}$ such that $v_A \in [l_A, u_A]$ for all $A \in \mathcal{A}$ and $v_A \leq v_B$ for all $A, B \in \mathcal{A}$, $A \subset B$ (i.e. I is locally consistent). Consider the following complete extension of I:

$$\forall A \notin \mathcal{A},\ v_A = \max\{l_A,\ \max_{A' \in \mathcal{A}: A' \subset A} v_{A'}\} \tag{10}$$

We assume that $\max_{A' \in \mathcal{A}: A' \subset A} v_{A'} = -\infty$ when $\{A' \in \mathcal{A} : A' \subset A\} = \emptyset$. We want to show that this complete instantiation satisfies boundary and monotonicity constraints.

Boundary constraints: we want to prove $v_A \in [l_A, u_A]$ for all $A \subseteq N$. If $A \in \mathcal{A}$, then boundary conditions follow directly from the hypothesis. If, instead, $A \notin \mathcal{A}$, we necessarily have $v_A \geq l_A$ by definition (see Equation (10)). To derive $v_A \leq u_A$, recall that $u_{A'} \leq u_A$ for all $A' \subset A$. Then, we obtain:

$$v_A = \max\{l_A, \max_{A' \in \mathcal{A}: A' \subset A} v_{A'}\} \leq \max\{u_A, \max_{A' \in \mathcal{A}: A' \subset A} u_{A'}\} \leq u_A$$

Monotonicity constraints: we want to prove $v_A \leq v_B$ for all $A \subset B \subseteq N$. We distinguish the four following cases:

- *Case $A \in \mathcal{A}$ and $B \in \mathcal{A}$:* we can directly infer the result since instantiation I is locally consistent.
- *Case $A \in \mathcal{A}$ and $B \notin \mathcal{A}$:* in that case, we have $A \in \{B' \in \mathcal{A} : B' \subset B\}$. Therefore :

$$v_A \leq \max_{B' \in \mathcal{A}: B' \subset B} v_{B'} \leq \max\{l_B, \max_{B' \in \mathcal{A}: B' \subset B} v_{B'}\} = v_B$$

- *Case $A \notin \mathcal{A}$ and $B \in \mathcal{A}$:* Consider any $A' \subset A$ with $A' \in \mathcal{A}$ (if it exists). Since $A \subset B$, we have $A' \subset B$ and both A' and B are elements of $\mathcal{A}$; hence, we necessarily have $v_{A'} \leq v_B$ since I is locally consistent. We then write the expression of v_A according to Equation (10) and derive:

$$v_A = \max\{l_A, \max_{A' \in \mathcal{A}: A' \subset A} v_{A'}\} \leq \max\{l_B, v_B\} = v_B$$

- Case $A \notin \mathcal{A}$ and $B \notin \mathcal{A}$: Since $A \subset B$, we have $l_A \leq l_B$ by hypothesis. Moreover, we necessarily have $\{A' \in \mathcal{A} : A' \subset A\} \subset \{B' \in \mathcal{A} : B' \subset B\}$. Therefore, using Equation (10), we have:

$$v_A = \max\{l_A, \max_{A' \in \mathcal{A}: A' \subset A} v_{A'}] \leq \max\{l_B, \max_{B' \in \mathcal{A}: B' \subset B} v_{B'}\} = v_B$$

□

Since sets belonging to $\mathcal{A}_{(x,y)}$ are the only ones that appear in the objective function of PL_1, then by applying Proposition 1 to set $\mathcal{A} = \mathcal{A}_{(x,y)}$, we know that $\mathrm{PMR}(x, y, \Theta_{\mathcal{P}})$ can be solved by considering only monotonicity constraints involving variables v_A, $A \in \mathcal{A}_{(x,y)}$. Therefore, we can compute $\mathrm{PMR}(x, y, \Theta_{\mathcal{P}})$ by solving the following simpler linear program:

$$(\mathrm{LP}_2) \quad \max_{v_{X_{(i)}}, v_{Y_{(i)}}, i \in N} \sum_{i=1}^{n} \left[(y_{(i)} - y_{(i-1)}) v_{Y_{(i)}} - (x_{(i)} - x_{(i-1)}) v_{X_{(i)}} \right]$$

$$\begin{aligned}
s.t. \quad & v_{X_{(i+1)}} \leq v_{X_{(i)}}, \forall i \in [\![1, n-1]\!] && (11)\\
& v_{Y_{(i+1)}} \leq v_{Y_{(i)}}, \forall i \in [\![1, n-1]\!] && (12)\\
& v_{X_{(i)}} \leq v_{Y_{(j)}}, \forall i, j \in N \text{ s.t. } X_{(i)} \subset Y_{(j)} && (13)\\
& v_{Y_{(i)}} \leq v_{X_{(j)}}, \forall i, j \in N \text{ s.t. } Y_{(i)} \subset X_{(j)} && (14)\\
& l_{X_{(i)}} \leq v_{X_{(i)}} \leq u_{X_{(i)}}, \forall i \in N && (15)\\
& l_{Y_{(i)}} \leq v_{Y_{(i)}} \leq u_{Y_{(i)}}, \forall i \in N && (16)
\end{aligned}$$

This linear program includes at most $2n - 1$ variables (one per element of $\mathcal{A}_{(x,y)}$) and only a quadratic number of monotonicity constraints.

3.2. A more compact formulation

In this subsection, we show that program LP_2 can be further simplified to obtain a linear programming formulation involving only a linear number of monotonicity constraints. For any set of criteria $A \in \mathcal{A}_{(x,y)}$, let ω_A denote the coefficient of the decision variable v_A in the objective function of linear program LP_2. Then, the objective function can be written $\sum_{A \in \mathcal{A}_{(x,y)}} \omega_A v_A$ where:

$\omega_A = -(x_{(i)} - x_{(i-1)}) \leq 0$, if $A = X_{(i)}$ and $A \neq Y_{(i)}$ for some $i \in N$
$\omega_A = y_{(i)} - y_{(i-1)} \geq 0$, if $A \neq X_{(i)}$ and $A = Y_{(i)}$ for some $i \in N$
$\omega_A = y_{(i)} - y_{(i-1)} - (x_{(i)} - x_{(i-1)})$, if $A = X_{(i)}$ and $A = Y_{(i)}$ for some $i \in N$

Note that we cannot have $X_{(i)} = Y_{(j)}$ if i is different from j since these level sets can only be equal if they have the same cardinality. Note also that all variables $v_A, A \in \mathcal{A}_{(x,y)}$, such that $\omega_A = 0$ have no impact on the objective function of program LP_2. Therefore, using Proposition 1, these variables can be removed from program LP_2 leading to a simplified version of this program. However, to simplify the presentation, we now assume that $\omega_A \neq 0$ for all $A \in \mathcal{A}_{(x,y)}$ [3].

We want to prove that Equation (13) is not required to determine the optimum of program LP_2. Let LP'_2 be the linear program obtained from program LP_2 by removing Equation (13). The following proposition gives us a necessary condition for optimality (the proof is given in the Appendix):

Proposition 2. *Let v be an optimal solution of program LP'_2. For all $A \in \mathcal{A}_{(x,y)}$ such that $\omega_A > 0$, we have:*

$$v_A = \min\{u_A, \min_{A' \in Pa(A)} v_{A'}\} \tag{17}$$

where $Pa(A) = \{A' \in \mathcal{A}_{(x,y)} : A' \supset A \text{ and } \omega_{A'} < 0\}$ is the set of all supersets (parents) A' of set A such that $\omega_{A'} < 0$.

This proposition enables us to derive the following result:

Proposition 3. *Any optimal solution of LP'_2 is optimal for LP_2.*

Proof. Let v be an optimal solution of program LP'_2. We want to prove that solution v necessarily satisfies Equation (13). Let $i, j \in N$ be such that $X_{(i)} \subset Y_{(j)}$. We want to show that $v_{X_{(i)}} \leq v_{Y_{(j)}}$ is necessarily satisfied. Proposition 2 can be applied to $Y_{(j)}$ since $\omega_{Y_{(j)}} > 0$. Two cases may occur:

- *Case $v_{Y_{(j)}} = u_{Y_{(j)}}$:* in that case, since $X_{(i)} \subset Y_{(j)}$, we have $u_{X_{(i)}} \leq u_{Y_{(j)}}$. Moreover, due to Equation (15), we have $v_{X_{(i)}} \leq u_{X_{(i)}}$. Therefore:

$$v_{X_{(i)}} \leq u_{X_{(i)}} \leq u_{Y_{(j)}} = v_{Y_{(j)}}$$

[3] Otherwise, the results can be established using the same arguments on this simplified version of program LP_2 and restricting $\mathcal{A}_{(x,y)}$ to sets A such that $\omega_A \neq 0$.

- *Case* $v_{Y_{(j)}} = \min_{A \in Pa(Y_{(j)})} v_A$*:* if $Pa(Y_{(j)}) = \emptyset$, then $v_{Y_{(j)}} = u_{Y_{(j)}}$ and so the result can be inferred just as in the first case. Otherwise, note that we have $Pa(Y_{(j)}) \subseteq \{A \in \mathcal{A}_{(x,y)} : \omega_A < 0\} \subseteq \{X_{(k)}, k \in N\}$. Therefore, in that case, there exists $k \in N$ such that $v_{Y_{(j)}} = v_{X_{(k)}}$. Then, since $X_{(i)} \subset Y_{(j)}$ (by hypothesis) and $Y_{(j)} \subset X_{(k)}$ (by definition of $Pa(Y_{(j)})$), we necessarily have $X_{(i)} \subset X_{(k)}$. As a consequence, we have $v_{X_{(i)}} \leq v_{X_{(k)}}$ due to Equation (11); hence, we have $v_{X_{(i)}} \leq v_{X_{(k)}} = v_{Y_{(j)}}$.

□

This proposition enables us to conclude that none of the constraints (13) are required to find the optimum.

Note that some constraints given by Equation (14) are unnecessary since they are redundant (they are implied by the other constraints). Indeed, if there exist $i, j \in N$ such that $Y_{(i)} \subset X_{(j)}$, then we also have $Y_{(i)} \subset X_{(k)}$ for all $k \in \{1, \ldots, j-1\}$ which creates redundant constraints with respect to Equation (11). Thus, it is sufficient to impose $v_{Y_{(i)}} \leq v_{X_{(j)}}$ when $Y_{(i)} \subset X_{(j)}$ and $Y_{(i)} \not\subseteq X_{(j+1)}$. Moreover, we also have $Y_{(l)} \subseteq X_{(j)}$ for all $l \in \{i+1, \ldots, n\}$, which creates a redundancy with Equation (12). Finally, it is sufficient to impose:

$$v_{Y_{(i)}} \leq v_{X_{(j)}} \text{ when } Y_{(i)} \subset X_{(j)},\ Y_{(i)} \not\subseteq X_{(j+1)} \text{ and } Y_{(i-1)} \not\subseteq X_{(j)}$$

This leads to the following simplified formulation:

$$\begin{aligned}
\max_{v_{X_{(i)}}, v_{Y_{(i)}}, i \in N} \quad & \sum_{i=1}^{n} \Big[(y_{(i)} - y_{(i-1)}) v_{Y_{(i)}} - (x_{(i)} - x_{(i-1)}) v_{X_{(i)}} \Big] & \\
s.t. \quad & v_{X_{(i+1)}} \leq v_{X_{(i)}}, \forall i \in [\![1, n-1]\!] & (18)\\
& v_{Y_{(i+1)}} \leq v_{Y_{(i)}}, \forall i \in [\![1, n-1]\!] & (19)\\
(\mathrm{LP}_3) \qquad & v_{Y_{(i)}} \leq v_{X_{(j)}}, \forall i, j \in N, \text{ s.t. } \left\{ \begin{array}{l} Y_{(i)} \subset X_{(j)} \\ Y_{(i)} \not\subseteq X_{(j+1)} \\ Y_{(i-1)} \not\subseteq X_{(j)} \end{array} \right. & (20)\\
& l_{X_{(i)}} \leq v_{X_{(i)}} \leq u_{X_{(i)}}, \forall i \in N & (21)\\
& l_{Y_{(i)}} \leq v_{Y_{(i)}} \leq u_{Y_{(i)}}, \forall i \in N & (22)
\end{aligned}$$

Hence, we now use at most $2n-1$ variables (one per element of $\mathcal{A}_{(x,y)}$) linked by at most $3(n-1)$ monotonicity constraints.

Example 2. *Consider a problem defined on 5 criteria (i.e.* $N = \{1, 2, 3, 4, 5\}$*), and two alternatives* $x = (1, 0.8, 0.4, 0.5, 0.1)$ *and* $y = (0.8, 1, 0.6, 0.2, 0.4)$*. The*

compact linear program associated with $\mathrm{PMR}(x, y, \Theta_{\mathcal{P}})$ *computation is:*

$$\max_{v} \quad 0.2(v_{\{1,2,3,5\}}+v_{\{1,2,3\}}+v_{\{2\}}-v_{\{1\}})+0.1(v_N-v_{\{1,2\}}-v_{\{1,2,4\}})-0.3v_{\{1,2,3,4\}}$$

$$s.t. \quad v_{\{1\}} \leq v_{\{1,2\}} \leq v_{\{1,2,4\}} \leq v_{\{1,2,3,4\}} \leq v_N \tag{23}$$

$$v_{\{2\}} \leq v_{\{1,2\}} \leq v_{\{1,2,3\}} \leq v_{\{1,2,3,5\}} \leq v_N \tag{24}$$

$$v_{\{1,2,3\}} \leq v_{\{1,2,3,4\}} \tag{25}$$

$$l_{\{1\}} \leq v_{\{1\}} \leq u_{\{1\}},\ l_{\{1,2\}} \leq v_{\{1,2\}} \leq u_{\{1,2\}},\ l_{\{1,2,4\}} \leq v_{\{1,2,4\}} \leq u_{\{1,2,4\}}$$

$$l_{\{1,2,3,4\}} \leq v_{\{1,2,3,4\}} \leq u_{\{1,2,3,4\}},\ l_N \leq v_N \leq u_N,\ l_{\{2\}} \leq v_{\{2\}} \leq u_{\{2\}}$$

$$l_{\{1,2,3\}} \leq v_{\{1,2,3\}} \leq u_{\{1,2,3\}},\ l_{\{1,2,3,5\}} \leq v_{\{1,2,3,5\}} \leq u_{\{1,2,3,5\}}$$

where the constraints in (23) are those given by Equation (18), the constraints in (24) correspond to Equation (19) and the constraint in (25) is given by Equation (20); the other constraints correspond to Equations (21-22). Thus, there are only nine monotonicity constraints.

3.3. Efficient optimization of the compact formulation

Although a state-of-the-art solver easily solves any instance of linear program LP_3, the computation of minimax regret may require a quadratic number of calls to LP_3. To speed up computations of minimax regrets, we present now a faster solution method for LP_3 without resorting to linear programming.

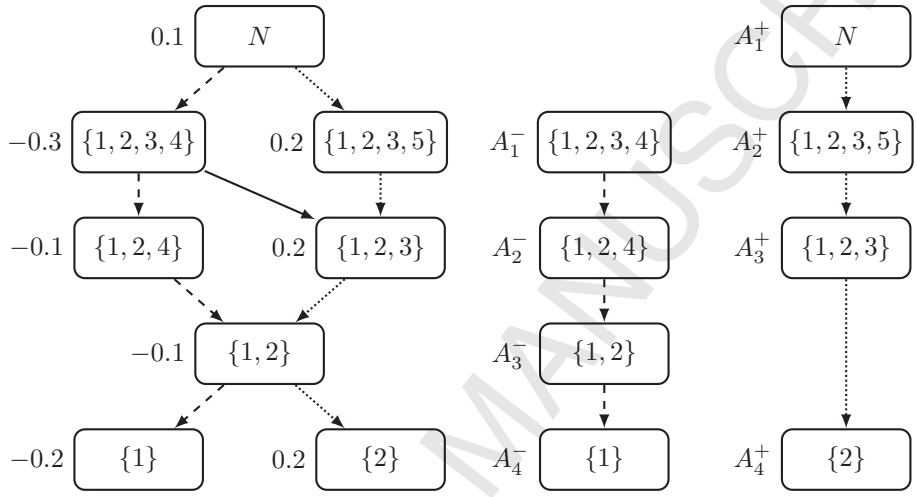
Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be the directed graph that represents the binary constraints of program LP_3. More precisely, $\mathcal{G}$ is defined as follows:

- $\mathcal{V} = \{X_{(i)}, i \in N\} \cup \{Y_{(i)}, i \in N\}$ is composed of all sets $A \subseteq N$ associated to decision variables v_A in LP_3.
- $\mathcal{E}$ is the set of arcs (A, B) such that constraint $v_A \geq v_B$ appears in LP_3.

Within this graph, the arcs corresponding to Equations (18-19) form two paths characterized by the following lists of nodes: $P_x = (X_{(1)}, \ldots, X_{(n)})$ and $P_y = (Y_{(1)}, \ldots, Y_{(n)})$. Let A^- (resp. A^+) be the subsequence of P_x (resp. P_y) composed of all nodes A such that $\omega_A < 0$ (resp. $\omega_A > 0$) where ω_A is the coefficient of v_A in the objective function. By definition, sequences A^- and A^+ include together all the nodes in $\mathcal{V}$ and have no common node; in the sequel, variables attached to nodes in A^- (resp. A^+) will be referred to as *negative variables* (resp. *positive variables*) since they have a negative (resp. positive) impact on the objective function.

Note that, by definition of level sets, we necessarily have $A_i^+ \supset A_{i+1}^+$ for all $i \in \{1, \ldots, |A^+| - 1\}$ and $A_i^- \supset A_{i+1}^-$ for all $i \in \{1, \ldots, |A^-| - 1\}$, where A_i^+ (resp. A_i^-) denote the i^{th} node in sequence A^+ (resp. A^-). Note also that, for all $i \in \{1, \ldots, |A^-|\}$, there exists at most one arc of type (A_i^-, A_j^+) in the constraint graph (see Equation (20)). An example of constraint graph $\mathcal{G}$ with sequences A^+ and A^- is given below, based on Example 2:

Example 2 (continued). *In this example, we have* $N = \{1, 2, 3, 4, 5\}$, $x = (1, 0.8, 0.4, 0.5, 0.1)$ *and* $y = (0.8, 1, 0.6, 0.2, 0.4)$. *The graph* $\mathcal{G}$ *is depicted on Figure 1 (left part). Within this graph,* $P_x = (N, \{1, 2, 3, 4\}, \{1, 2, 4\}, \{1, 2\}, \{1\})$ *and* $P_y = (N, \{1, 2, 3, 5\}, \{1, 2, 3\}, \{1, 2\}, \{2\})$. *Hence the relevant sequences are* $A^- = (\{1, 2, 3, 4\}, \{1, 2, 4\}, \{1, 2\}, \{1\})$ *and* $A^+ = (N, \{1, 2, 3, 5\}, \{1, 2, 3\}, \{2\})$ *also depicted on Figure 1 (right part).*



(a) Constraint graph associated with the problem; the dashed (resp. dotted) arcs represent path P_x (resp. P_y); value ω_A is given at the left side of each node A.

(b) Sequences A^- and A^+ associated with the problem; successive elements of sequence A^- (resp. A^+) are linked by dashed (resp. dotted) arcs.

Figure 1: The constraint graph and the sequences A^+ and A^- associated to Example 2.

Since sequence A^+ (resp. A^-) includes all sets A such that variable v_A has a positive (resp. negative) impact on the objective unction, we want to maximize (resp. minimize) the variable v_A for all $A \in A^+$ (resp. for all $A \in A^-$) so as to maximize the objective function. Thus, if there exists no arc of type (A_i^-, A_j^+) in $\mathcal{E}$, then the optimum of program LP_3 can be easily obtained; it is indeed sufficient to set v_A to its lower bound l_A for all $A \in A^-$ and v_A to its upper bound u_A for all $A \in A^+$. Otherwise, for all arcs of type (A_i^-, A_j^+) in $\mathcal{E}$, we need to decide whether to set variable $v_{A_i^-}$ to its lower bound $l_{A_i^-}$ (at the expense of constraining variable $v_{A_j^+}$) or to assign $v_{A_i^-}$ to an higher value. Moreover, for all $k \in \{j+1, \ldots, |A^+|\}$, we implicitly impose $v_{A_i^-} \geq v_{A_k^+}$ since $A_k^+ \subset A_j^+$ (by definition of sequence A^+). As a consequence, we need to decide whether to assign variable $v_{A_i^-}$ to its lower bound $l_{A_i^-}$ or to set this variable to an higher value in order to preserve the variables in $\{v_{D_m^i}, 1 \leq m \leq |D^i|\}$, where D^i is the subsequence of A^+ composed of all descendants of node A_i^- in the graph and D_m^i denotes the m^{th} element in sequence D^i.

In order to make this decision, we need first to check whether we should set $v_{A_i^-}$ to the upper bound of its first positive descendant D_1^i or at a lower value. We are in the former case when the sum of the values $|\omega_A|$, for all sets $A \in A^-, A \supset D_1^i$, is strictly smaller than $\omega_{D_1^i}$. In this case, $v_{A_i^-}$ is set to the upper bound of D_1^i to protect $v_{D_1^i}$ which better contributes to the objective function. Otherwise, the test must be iterated on the second descendant D_2^i and so on; if the test fails for all descendants of node A_i^-, variable $v_{A_i^-}$ is set to its lower bound. This principle is implemented in Algorithm 1. Before establishing the validity of Algorithm 1, let us remark that it has a quadratic time complexity since $|A^-| \leq n$ and $|D^i| \leq |A^+| \leq n$.

Algorithm 1: Iterative optimization of PMR

Input: Two alternatives $x, y \in X$
Output: instantiation of variables v_A, $A \in \mathcal{A}_{(x,y)}$

Construction of A^- and A^+ from the constraint graph
for $A \in A^+$ **do** $v_A \leftarrow u_A$
for $i = 1 \ldots |A^-|$ **do**
  $v_{A_i^-} \leftarrow l_{A_i^-}$
  $j \leftarrow 1$
  $\omega^+ \leftarrow 0$
  $\omega^- \leftarrow 0$
  while $j \leq |D^i|$ *and* $v_{D_j^i} > l_{A_i^-}$ **do**
    $\omega^+ \leftarrow \omega^+ + \omega_{D_j^i}$
    $\omega^- \leftarrow \omega^- + \sum_{k \in \{i,\ldots,|A^-|\}: D_1^k = D_j^i} \omega_{A_k^-}$
    if $\omega^+ + \omega^- > 0$ **then**
      $v_{A_i^-} \leftarrow v_{D_j^i}$
      break
    end
    $j \leftarrow j + 1$
  end
  for $j = 1 \ldots |D^i|$ **do** $v_{D_j^i} \leftarrow \min\{v_{D_j^i}, v_{A_i^-}\}$
end
return v

The following proposition will be used to prove the correctness of Algorithm 1 (the proof is given in the Appendix).

Proposition 4. *At the end of any step i of the 'for' loop, we have:*

$$\forall A \in A^+,\ v_A = \min\{u_A, \min_{A' \in Pa_i(A)} v_{A'}\}$$

where $Pa_i(A) = \{A_k^-, 1 \leq k \leq i : A \subset A_k^-\}$ is the set of all supersets (parents) A' of set A such that $\omega_{A'} < 0$ and $A' \supseteq A_i^-$.

Proposition 5. *Algorithm 1 returns a solution such that Equation (17) is satisfied for all $A \in A^+$.*

Proof. The result directly follows from Proposition 4 (with $i = |A^-|$) since $Pa_{|A^-|}(A) = Pa(A)$ by definition. □

Thus, Equation (17) actually gives us an operational way to set positive variables, when negative variables are set first. The following proposition shows the correctness of our algorithm (the proof is given in the Appendix):

Proposition 6. *Algorithm 1 returns an optimal solution of program LP_3.*

For the sake of illustration, we present the execution of Algorithm 1 on the two alternatives considered in Example 2.

Example 2 (continued). *Recall that, in Example 2, we have $N = \{1, 2, 3, 4, 5\}$, $x = (1, 0.8, 0.4, 0.5, 0.1)$ and $y = (0.8, 1, 0.6, 0.2, 0.4)$. Let us assume that, at a given step of the elicitation procedure, the corresponding intervals $[l_A, u_A]$, $A \in \mathcal{A}_{(x,y)}$, are given here below:*

$v_{\{1\}}$	$v_{\{1,2\}}$	$v_{\{1,2,4\}}$	$v_{\{1,2,3,4\}}$	$v_{\{2\}}$	$v_{\{1,2,3\}}$	$v_{\{1,2,3,5\}}$	v_N
$[0, 0.5]$	$[0.2, 0.7]$	$[0.3, 0.9]$	$[0.6, 1]$	$[0.1, 0.4]$	$[0.4, 0.8]$	$[0.5, 0.9]$	$[1, 1]$

We show how Algorithm 1 computes $\mathrm{PMR}(x, y, \Theta_{\mathcal{P}})$. *The algorithm starts with the execution of line 1 where the structure depicted on Figure 1 is built. Then, according to line 2, we set $v_A = u_A$ for all $A \in A^+$: $v_N = 1$, $v_{\{1,2,3,5\}} = 0.9$, $v_{\{1,2,3\}} = 0.8$ and $v_{\{2\}} = 0.4$.*
At line 3, we start the execution of the first iteration of the outer for loop, examining $A_1^- = \{1, 2, 3, 4\}$; its positive descendants are given by $D^1 = (\{1, 2, 3\}, \{2\})$. When executing line 4, we set $v_{\{1,2,3,4\}}$ to its minimum feasible value, i.e. $v_{\{1,2,3,4\}} = l_{\{1,2,3,4\}} = 0.6$. Then, we enter the while loop (line 8):

- *During the first iteration step of the while loop, we consider the descendant $D_1^1 = \{1, 2, 3\}$. Since $v_{\{1,2,3\}} = 0.8 > 0.6 = l_{\{1,2,3,4\}}$, we compute $\omega^+ = \omega_{\{1,2,3\}} = 0.2$ and $\omega^- = \omega_{\{1,2,3,4\}} = -0.3$. This iteration step stops since the condition $\omega^+ + \omega^- > 0$ does not hold.*
- *Then, we consider the descendant $D_2^1 = \{2\}$. Since $v_{\{2\}} = 0.4 \leq 0.6 = l_{\{1,2,3,4\}}$, the second part of the condition of the while loop (line 8) is not satisfied (the while loop immediately stops).*

We perform the following updates (line 17): $v_{\{1,2,3\}} = \min\{v_{\{1,2,3\}}, v_{\{1,2,3,4\}}\} = \min\{0.8, 0.6\} = 0.6$ and $v_{\{2\}} = \min\{v_{\{2\}}, v_{\{1,2,3,4\}}\} = \min\{0.4, 0.6\} = 0.4$.

In the second iteration step of the outer for loop, we inspect the node $A_2^- = \{1, 2, 4\}$ with only one positive descendant: $D^2 = (\{2\})$. We initially set $v_{\{1,2,4\}}$ to its minimum feasible value (line 4), i.e. $v_{\{1,2,4\}} = l_{\{1,2,4\}} = 0.3$.

- *We consider the descendant* $D_1^2 = \{2\}$. *Since* $v_{\{2\}} = 0.4 > 0.3 = l_{\{1,2,4\}}$, *we compute* $\omega^+ = \omega_{\{2\}} = 0.2$ *and* $\omega^- = \omega_{\{1,2,4\}} + \omega_{\{1,2\}} = -0.2$. *As a consequence, the condition in line 11 is not verified at this step. Then, the while loop stops since* $|D^2| = 1$.

We perform the following update: $v_{\{2\}} = \min\{v_{\{2\}}, v_{\{1,2,4\}}\} = \min\{0.4, 0.3\} = 0.3$.

At the third iteration step, we inspect the node $A_3^- = \{1, 2\}$ *whose descendants are given by* $D^3 = (\{2\})$. *At line 4, we set* $v_{\{1,2\}}$ *to its minimum feasible value, i.e.* $v_{\{1,2\}} = l_{\{1,2\}} = 0.2$.

- *We consider the descendant* $D_1^3 = \{2\}$. *Since* $v_{\{2\}} = 0.3 > 0.2 = l_{\{1,2\}}$, *we compute* $\omega^+ = \omega_{\{2\}} = 0.2$ *and* $\omega^- = \omega_{\{1,2\}} = -0.1$. *Here,* $\omega^+ + \omega^- > 0$ *holds, and so we need to update the value of* $v_{\{1,2\}}$ *as follows:* $v_{\{1,2\}} = v_{\{2\}} = 0.3$. *Finally, the while loop stops due to the "break" command.*

We perform the following update: $v_{\{2\}} = \min\{v_{\{2\}}, v_{\{1,2\}}\} = \min\{0.3, 0.3\} = 0.3$.

At the fourth iteration step, we have $A_4^- = \{1\}$, $D^4 = ()$. *We set* $v_{\{1\}}$ *to its minimum feasible value, i.e.* $v_{\{1\}} = l_{\{1\}} = 0$. *Then, this iteration step stops since* D^4 *is empty.*

Finally, the algorithm returns the following instantiation:

$v_{\{1\}}$	$v_{\{1,2\}}$	$v_{\{1,2,4\}}$	$v_{\{1,2,3,4\}}$	$v_{\{2\}}$	$v_{\{1,2,3\}}$	$v_{\{1,2,3,5\}}$	v_N
0	0.3	0.3	0.6	0.3	0.6	0.9	1

3.4. A query generation strategy for Choquet capacity elicitation

In the previous subsections, we have shown that, assuming that preference statements are of types $(1A0, \Lambda)$ or $(\Lambda, 1A0)$, we are able to efficiently compute minimax regret values. We introduce now a query generation strategy determining the most informative query involving a binary alternative and a constant profile. Our query generation strategy uses the WmMR criterion presented in Definition 7. Since at each iteration step the DM is asked to compare a binary alternative 1A0 to a constant profile $\Lambda = (\lambda, \ldots, \lambda)$, an optimal query for this criterion is defined by a pair $(A \subseteq N, \lambda \in [l_A, u_A])$ that reduces the minimax regret in the worst-case as much as possible. More precisely, to find such a pair, for all sets $A \subseteq N$, we have to determine the following value:

$$\begin{aligned}\lambda_A &= \arg \min_{\lambda \in [l_A, u_A]} \mathrm{WmMR}((A, \lambda), \Theta_{\mathcal{P}}) \\ &= \arg \min_{\lambda \in [l_A, u_A]} \max \Big\{ \mathrm{mMR}(X, \Theta_{\mathcal{P} \cup \{(1A0, \Lambda)\}}),\ \mathrm{mMR}(X, \Theta_{\mathcal{P} \cup \{(\Lambda, 1A0)\}}) \Big\}\end{aligned}$$

and then, we arbitrary select a set A^* in $\arg\min_{A \subseteq N} \mathrm{WmMR}((A, \lambda_A), \Theta_{\mathcal{P}})$; thus, the pair $(A^*,\ \lambda_{A^*})$ defines an optimal query for the WmMR criterion.

Given a set $A \subseteq N$, determining λ_A amounts to minimizing over $\lambda \in [l_A, u_A]$ the maximum between:

- $\mathrm{mMR}(X, \Theta_{\mathcal{P}\cup\{(1A0,\Lambda)\}})$ which is a weakly decreasing function of λ, and
- $\mathrm{mMR}(X, \Theta_{\mathcal{P}\cup\{(\Lambda,1A0)\}})$ which is a weakly increasing function of λ.

Similarly to what is observed for utility functions over consequences [28], these two functions necessarily intersect since they have the same maximum, i.e. $\mathrm{mMR}(X, \Theta_{\mathcal{P}})$. This intersection gives the value of λ_A and can easily be computed by a standard bisection algorithm. An example of this phenomenon is pictured on Figure 2.

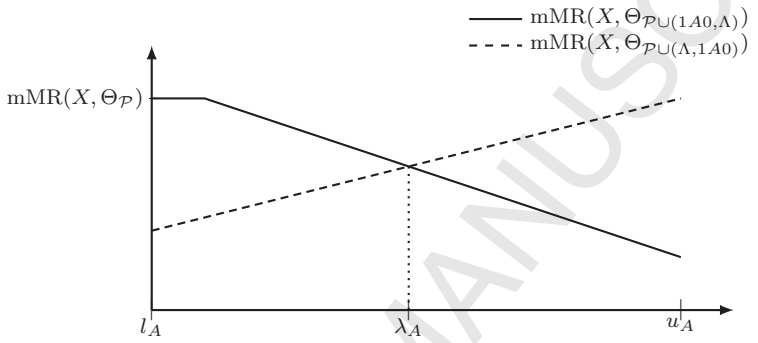


Figure 2: Determination of threshold λ_A.

It may happen that the WmMR value of the optimal query is equal to $\mathrm{mMR}(X, \Theta_{\mathcal{P}})$, which means that the optimal question will not necessarily induce a regret reduction. In such cases, we propose to choose a set $A \subseteq N$ that minimizes the expected value of minimax regret over the two possible answers with an uniform distribution hypothesis over $[l_A, u_A]$, setting $\lambda = (l_A + u_A)/2$.

Note that the determination of the next query implies to select A^* within the $2^n - 2$ possible proper subsets of N, a number which increases significantly with the number of criteria. To make this query generation step more efficient, as a heuristic we propose to focus on sets directly involved in the computation of $\mathrm{PMR}(x^*, y^*, \Theta_{\mathcal{P}})$, where x^* is an optimal alternative for the mMR criterion knowing $\mathcal{P}$, and y^* is one of its adversary's choices. These sets are those in

$$\mathcal{A}_{(x^*,y^*)} = \{X^*_{(i)}, i \in N\} \cup \{Y^*_{(i)}, i \in N\}$$

where $X^*_{(i)}$ and $Y^*_{(i)}$ respectively denote the i^{th} level set of x^* and y^*. Thus, the heuristic will further constrain parameters inducing the current minimax regret. According to this heuristic, at most $2n - 1$ sets are investigated (the elements of $\mathcal{A}_{(x^*,y^*)}$) instead of exactly $2^n - 2$.

Let us remark that this incremental elicitation procedure enforces both the consistency of the set $\mathcal{P}$ of preference statements collected so far, and the consistency of the induced constraints defining $\Theta_{\mathcal{P}}$, under the assumption that preferences are representable by a Choquet integral. Hence $\Theta_{\mathcal{P}}$ cannot become

empty at some step of the elicitation process. This important feature can be explained as follows: let $[l_A, u_A]$ be the range of admissible values for $v(A)$ at a given step of the elicitation procedure (resulting from $\mathcal{P}$). At this stage, for any $\lambda \in [l_A, u_A]$, both statements $(1A0, (\lambda, \ldots, \lambda))$ and $((\lambda, \ldots, \lambda), 1A0)$ are compatible with the known part of preferences. Asking the DM to compare the binary alternative $1A0$ to the constant utility profile $\Lambda = (\lambda, \ldots, \lambda)$ cannot generate any contradiction because the answer will induce one of the two constraints "$v(A) \leq \lambda$" or "$v(A) \geq \lambda$". Such constraints will contribute to further reduce the interval $[l_A, u_A]$ into $[\lambda, u_A]$ or $[l_A, \lambda]$, depending on the answer, but the interval remains non-empty in both cases.

3.5. Numerical tests involving choice problems

The first numerical tests aim to evaluate the efficiency of the incremental elicitation procedure presented in Section 3.4 in terms of number of queries needed to make a decision. Recall that each generated query involves two fictitious alternatives carefully chosen: one binary alternative of type $1A0$, $A \subseteq N$, and one constant utility profile of type $\Lambda = (\lambda, \ldots, \lambda)$, $\lambda \in [l_A, u_A]$. Hence, as a baseline for comparison, we consider the query generation strategy that consists in choosing, at each iteration step of the elicitation procedure, both set $A \subseteq N$ and $\lambda \in [l_A, u_A]$ at random.

Starting from an empty set of preferences statements, simulated DMs answer to queries according to randomly generated Choquet integrals. At each iteration step of the elicitation procedures, we compute both the minimax regret and the *real regret*, the latter being the actual loss of utility associated with the choice of the minimax regret optimal alternative instead of the true preferred alternative (the utility is measured by the hidden Choquet integral modeling the DM's preferences). These regrets are expressed on a normalized scale assigning value 1 to the initial minimax regret (computed before collecting any preference information) and value 0 when the true preferred alternative is detected.

Figure 3 shows the results obtained by averaging over 100 runs for instances with 10 criteria and 1000 alternatives; performance vectors of alternatives are generated by randomly picking Pareto-optimal solutions in a multiobjective knapsack problem so as to obtain alternatives which are uncomparable using Pareto-dominance and require additional preference information from the DM.

First, we can see that the minimax regret reduces much more quickly with our query strategy than with the random generation strategy. For instance, after about 20 queries on average, the minimax regret is under 10% of the initial regret with our strategy while remaining above 70% percent with the random generation strategy. The same observation applies to the real regret. Note also that the real regret is much smaller than the minimax regret (a fact that has already been observed in regret-based elicitation in other contexts [28, 40, 29]). After about 8 queries on average, the real regret observed when using our eliciation procedure is under 10% of the initial regret (while the minimax regret is still around 30%).

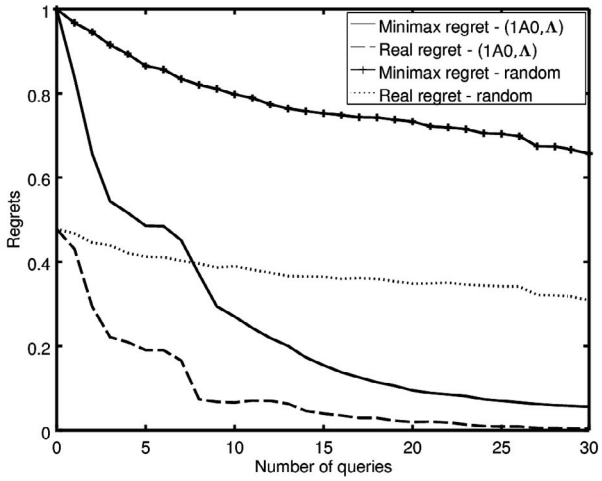


Figure 3: Comparison with the random generation strategy ($n = 10$, 1000 alternatives).

The next experiments aim at evaluating the efficiency of our query generation strategy in terms of regret reduction. To this end, it is compared with the standard elicitation method named Current Solution Strategy (CSS) [29] which is based on the comparison of two actual alternatives in the dataset (see Section 2.3). Results obtained by averaging over 100 runs are given in Figure 4.

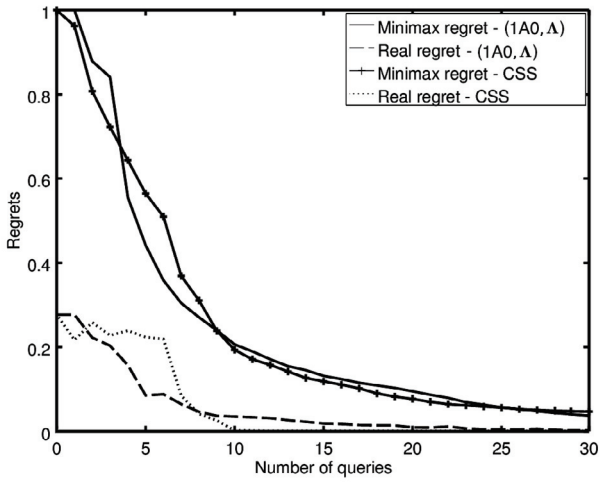


Figure 4: Comparison with the Current Solution Strategy ($n = 5$, 150 alternatives).

We can see that our elicitation procedure provides performance very similar to the CSS in terms of regret reduction. For instance, after about 20 queries, both elicitation procedures recommend an alternative with a max regret under 10 percent of the initial regret on average. The real advantage of our approach will appear when looking at the computation times. We compare now computation times of mMR calculations within the following elicitation methods:

- $\mathbf{M}_1$: queries are generated according to the CSS. Pairwise max regrets are computed by solving LP_1 (Section 3.1) using an LP-solver.
- $\mathbf{M}_2$: queries are generated according to our elicitation procedure presented in Section 3.4. The optimization of pairwise max regrets is performed by solving LP_3 (see Section 3.2) using an LP-solver.
- $\mathbf{M}_3$: queries are here also generated according to our elicitation procedure presented in Section 3.4, but Algorithm 1 presented in Section 3.3 is used to solve LP_3 (instead of using a LP-solver).

In our experiments, we use the Gurobi[4] solver, called from the main program written in Java (using the Gurobi-Java interface). In order to evaluate how the available preference information impacts on computation times, we report the results obtained after respectively 0, 10 and 20 iteration steps of the elicitation procedures. Computation times are obtained by averaging over 50 runs.

Table 1: Computation times of minimax regret computations (in seconds).

n	size	step	M_1	M_2	M_3
5	150	0	4.473	3.846	0.005
5	150	10	4.833	2.594	0.001
5	150	20	4.805	2.820	0.001
5	1000	0	35.944	32.161	0.007
5	1000	10	25.471	24.129	0.004
5	1000	20	26.792	24.131	0.003
10	150	0	26.357	3.864	0.002
10	150	10	9.504	2.586	0.001
10	150	20	12.901	2.087	0.001
10	1000	0	276.067	57.795	0.014
10	1000	10	140.062	38.721	0.008
10	1000	20	218.528	34.499	0.007

In Table 1, we can see that computation times with M_1 drastically increase with not only the number of alternatives, which directly impacts on the number

[4]http://www.gurobi.com/.

of pairwise max regret optimizations, but with the number of criteria due to the exponential number of monotonicity constraints in LP_1 formulation.

As expected, we can see that M_2 is faster than M_1 overall. Moreover, the number of criteria has much more impact on the computation time for the latter method as it deals with an exponential number of monotonicity constraints (where LP_3 has only a linear number of them).

Interestingly enough, M_3 is significantly faster than M_1 and M_2 (about five orders of magnitude) and its computations times seem to be very weakly impacted by the number of alternatives and the number of criteria. In fact, M_3 generates fifty queries in a few minutes for 1000 alternatives, while M_1 requires about six hours for a number of alternatives lower by an order of magnitude.

In conclusion, by restricting queries to the comparison of binary alternatives with constant profiles (rather than comparing real alternatives, as in the CSS), we obtain a drastically faster incremental elicitation procedure for Choquet integrals while preserving the efficiency of the elicitation process in terms of number of queries needed to make a decision.

3.6. Ranking by iterated choices

Ranking is a topic that has received considerable attention (from the community of machine learning [45, 46, 47], as well from psychological choice modeling [48], management science and decision aid communities [49, 50, 51, 52, 53, 54]). It is a much more complex issue than choosing a single alternative; it requires to be able to compare all pairs of alternatives instead of just identifying the best one. Although adapting an incremental elicitation scheme based on minimax regret computations to the case of ranking seems theoretically possible, minimax-regret optimizations would probably require prohibitive computation times. There are indeed factorially many possible orders on a given finite set of elements and computing explicitly pairwise regrets for all pairs of possible rankings does not seem feasible.

For this reason, we address here the problem of ranking as one of repeated choice, assuming that the DM first chooses the first alternative, then the second, and so forth. This is a very natural decision process and iteration can be interrupted at any step k if only the top k elements are of interest. When using iterated choices, constructing a ranking appears as a kind of extension of the choice problem and the tools developed for choosing may be used.

As in the choice problem, we reason with a partially specified capacity used in the Choquet integral. In this framework, an alternative x is necessarily at least as good as an alternative y when $C_v(x) \geq C_v(y)$ for all admissible capacities $v \in \Theta_{\mathcal{P}}$, that is when $\mathrm{PMR}(x, y, \Theta_{\mathcal{P}}) < 0$. In order to save queries, we might assume that x is preferred to y when $\mathrm{PMR}(x, y, \Theta_{\mathcal{P}}) \leq \delta$ where $\delta \geq 0$ is a (small) tolerance threshold. We will first generate preference queries and use minimax regret computations to determine a top alternative with a max regret lower than threshold δ. Then, this alternative is deleted and the selection process is iterated on the remaining set of alternatives with the same tolerance threshold.

The selected alternative in this second stage will be the second best alternative in the ranking and so forth (see Algorithm 2).

Algorithm 2: Interactive ranking by iterated choices

Input: X: set of alternatives
Output: L: list representing the whole ranking

$L \leftarrow ()$
$Z \leftarrow X$
while $Z \neq \emptyset$ **do**
  while $\text{mMR}(Z, \Theta_{\mathcal{P}}) > \delta$ **do**
    Ask a preference query to the DM and insert the answer in $\mathcal{P}$
    Update $\Theta_{\mathcal{P}}$ accordingly
  end
  Select z^* in $\arg\min_{z \in Z} \text{MR}(z, Z, \Theta_{\mathcal{P}})$
  Append(L, z^*)
  Remove z^* from Z
end
return L

This interactive ranking algorithm satisfying the following nice property:

Proposition 7. *For any pair of alternatives x and y such that x is ranked before y in the final ranking, we have $\text{PMR}(x, y, \Theta_{\mathcal{P}}) \leq \delta$ where $\mathcal{P}$ is the set of all preference statements collected to construct the whole ranking.*

Proof. Let $\mathcal{P}'$ be the set of preference statements collected until the insertion of x in the ranking. Since y is ranked below x, we know that $\text{PMR}(x, y, \Theta_{\mathcal{P}'}) \leq \delta$. Moreover, since $\mathcal{P}' \subseteq \mathcal{P}$, then we have $\Theta_{\mathcal{P}'} \supseteq \Theta_{\mathcal{P}}$ and (refer to Equation (2)) therefore $\text{PMR}(x, y, \Theta_{\mathcal{P}}) \leq \text{PMR}(x, y, \Theta_{\mathcal{P}'}) \leq \delta$. □

This property allows us to give a guarantee of the quality of the ranking obtained with our procedure; threshold δ represents the worst-case loss (in terms of utility) that we may incur by choosing x instead of any other alternative y that has a lower position in the ranking. In particular, if $\delta = 0$, then the true ranking (sorting the alternatives from the best to the worst according to their Choquet value C_v) has been identified with certainty.

Since the ranking is obtained by a sequence of choice problems, we can use at each step the same elicitation strategy presented in Section 3.4 for the Choquet integral. Moreover, at any step of the ranking procedure, we start from the set of preference statements collected so far to determine the next preferred element. This obviously saves a lot of preference queries as will be seen in the next paragraph dedicated to numerical tests.

Numerical tests. We now present some experimental results about our incremental ranking method. It consists in iteratively selecting the best alternative

and then removing it from the dataset of alternatives; the incremental elicitation procedure presented in Section 3.4 is used to determine the best alternative at each iteration step. We want to estimate the amount of additional preference information that is needed to rank all alternatives (instead of just determing the best one). Hence, we ran tests with different tolerance thresholds δ (0.05, 0.1 and 0.15) so as to study its impact on the number of queries. Figure 5 shows the results obtained by averaging over 50 runs for instances with 1000 alternatives and 5 criteria; performance vectors are generated as described in Section 3.5.

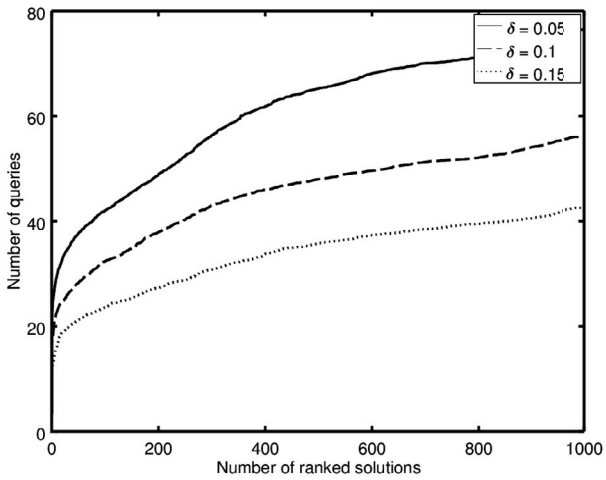


Figure 5: Performance (in terms of number of queries) of the interactive ranking algorithm ($n = 5$, 1000 alternatives).

As expected, we can see that the number of queries reduces as the value of δ increases (since the requirement on the performance guarantee is weakened); for instance, the number of queries needed to obtain the complete ranking is halved when the performance threshold increases from 5% to 15% of the initial regret. Moreover, we observe that completing the ranking after the determination of the top alternative can be achieved at a reasonable marginal cost (no more than two times the cost of determining the best alternative). Most of preference queries appears during the first iteration (when the first item in the ranking is identified); in fact, we empirically observed that the number of alternatives has a relatively low impact on the number of queries required for completion. In fact, the marginal amount of preference queries which are necessary to find the next element decreases as the rank of the selected alternative increases.

4. Sorting methods with thresholds on Choquet values

Sorting problems require to assign alternatives to categories. When assessing a utility score $f_\theta(x)$ for each alternative $x \in X$, it is natural to sort alternatives with respect to their score by considering thresholds. This is a standard approach in multricriteria decision-making [55, 56, 57, 58]. This is also typical in binary classification problems where algorithms compute a numerical value and then assign alternatives to categories by checking if the value exceeds a given threshold. When considering ordered categories, the machine learning community uses the term *multipartite ranking* [59, 60, 61] or *instance ranking* [31].

Assume that the utility scale is divided into q intervals $[\alpha_\ell, \alpha_{\ell-1}]$, where $\alpha_0 \geq \ldots \geq \alpha_q$. Assignments are made by looking in which interval the utility scores fall. More precisely, the method proposed in this section consists in assigning alternative $x \in X$ to category K_ℓ if $f_\theta(x) \in [\alpha_\ell, \alpha_{\ell-1}]$. In the case that $f_\theta(x)$ is exactly the value of threshold α_ℓ, we consider that the DM is indifferent between assigning x to category K_ℓ or $K_{\ell+1}$ (the two assignments are equally valid). This view of sorting is somewhat natural; it can be seen as a "discretization" of utility into categories for situations where we want to provide an informative summary about the utilities of the alternatives.

As utility function f_θ is not known precisely, we need to define some measure of regret for possible assignments. Hence, in this section, we first propose an incremental elicitation approach for sorting problems with thresholds (Section 4.1), before addressing the associated regret-optimization problem (Section 4.2). Then, we focus on difficulties encountered when f_θ is a Choquet integral, presenting optimization techniques based on linear programming or not (Section 4.3) and an efficient query generation strategy (Section 4.4). Finally, we present evaluations with simulations (in Section 4.5).

4.1. An incremental elicitation approach for sorting with thresholds

When sorting with thresholds, categories are associated with intervals in the utility scale, whose extreme are the thresholds. Assignment of alternatives to categories is determined by which intervals enclose the alternatives' utility values. More precisely, given parameter θ, we will assign x to the category K_ℓ such that $f_\theta(x) \in [\alpha_\ell, \alpha_{\ell-1}]$.

For fixed θ, the actual regret (or loss) of assigning alternative x to a category K_ℓ will then be 0 if $f_\theta(x)$ lies between α_ℓ and $\alpha_{\ell-1}$. When the assignment is made incorrectly, it is natural to assume that the regret will be higher for categories delimited by thresholds that are further away from $f_\theta(x)$. We further assume that the regret will scale linearly with the displacement of $f_\theta(x)$ from the nearest of the two thresholds $\alpha_\ell, \alpha_{\ell-1}$ delimiting category K_ℓ. More precisely, if $f_\theta(x) \geq \alpha_{\ell-1}$, then we define the regret to be $f_\theta(x) - \alpha_{\ell-1}$; similarly, if $f_\theta(x) \leq \alpha_\ell$, then the regret is equal to $\alpha_\ell - f_\theta(x)$. This is represented by the

following expression:

$$\mathrm{R}(x, K_\ell, \theta) = \max\Big\{ f_\theta(x) - \alpha_{\ell-1},\ \alpha_\ell - f_\theta(x),\ 0 \Big\}$$

Note that our formulation of regret R is consistent with the case in which $f_\theta(x)$ is exactly the value of one of the thresholds; for example, if $f_\theta(x) = \alpha_\ell$ then we have both $\mathrm{R}(x, K_\ell, \theta) = 0$ and $\mathrm{R}(x, K_{\ell+1}, \theta) = 0$.

In general, in our setting, parameter θ is not known precisely and we want to be able to sort alternatives under utility uncertainty. Proceeding in a way similar to the case of choice problems (see Section 2.2), we define the notion of *max regret* (and subsequently *minimax regret*), as follows:

Definition 8. *The max regret (MR) of $x \in X$ with respect to category K_ℓ is:*

$$\begin{aligned}\mathrm{MR}(x, K_\ell, \Theta_\mathcal{P}) &= \max_{\theta \in \Theta_\mathcal{P}} \mathrm{R}(x, K_\ell, \theta) \\ &= \max_{\theta \in \Theta_\mathcal{P}} \max\Big\{ f_\theta(x) - \alpha_{\ell-1},\ \alpha_\ell - f_\theta(x),\ 0 \Big\}.\end{aligned}$$

$\mathrm{MR}(x, K_\ell, \Theta_\mathcal{P})$ is the maximal possible utility gap between $f_\theta(x)$ and the interval $[\alpha_\ell, \alpha_{\ell-1}]$ defining category K_ℓ.

We now define the notion of *minimax regret* and the MR-*optimal category* associated to an alternative $x \in X$:

Definition 9. *The minimax regret (mMR) of $x \in X$ is:*

$$\mathrm{mMR}(x, \Theta_\mathcal{P}) = \min_{\ell \in \{1,\ldots,q\}} \mathrm{MR}(x, K_\ell, \Theta_\mathcal{P}).$$

The decision rule is that of assigning x to the category that minimizes $\mathrm{MR}(x, K_\ell, \Theta_\mathcal{P})$; this category is called the MR-*optimal category* of x. The mMR represents the maximal utility gap between $f_\theta(x)$ and the interval defining the mMR-optimal category.

In sorting problems, each alternative is associated with a minimax regret value (mMR). Therefore, we have a vector of $|X|$ minimax regret values (instead of a single mMR value for choice problems); sorting can be viewed as simultaneously solving several decision problems, one for each alternative that we need to assign to one of the categories. We now need to define an aggregate measure to evaluate the overall quality (with respect to regret values) of a complete assignment. We adopt the notion of *maximum minimax regret* (MmMR), defined as follows:

Definition 10.

$$\mathrm{MmMR}(X, \Theta_\mathcal{P}) = \max_{x \in X} \mathrm{mMR}(x, \Theta_\mathcal{P})$$

Note that one could consider other criteria, for example the average of the minimax regret values (allowing to compensate high regret values with lower

ones); here we focus on the maximum, as it is a choice consistent with the pessimistic notion of max regret MR and it provides a performance guarantee with respect to the worst-case.

In sorting problems, MmMR plays the role of measuring the current decision quality (as mMR did in choice problems). For a given set of preferences, it might be the case that the aggregate value MmMR is still too large according to the DM. Therefore, we can conceive incremental elicitation strategies that, as in choice problems, iteratively ask questions to the DM until the MmMR value drops below a given tolerance threshold $\delta \geq 0$. Indeed, we have $\Theta_{\mathcal{P}'} \subseteq \Theta_{\mathcal{P}}$ for any set of preference statements $\mathcal{P}' \supseteq \mathcal{P}$; then, for any $x \in \mathcal{X}$ and any $\ell \in \{1, \ldots, q\}$, we have:

$$\begin{aligned} \mathrm{MR}(x, K_\ell, \Theta_{\mathcal{P}'}) &\leq \mathrm{MR}(x, K_\ell, \Theta_{\mathcal{P}}) \\ \mathrm{mMR}(x, \Theta_{\mathcal{P}'}) &\leq \mathrm{mMR}(x, \Theta_{\mathcal{P}}) \\ \mathrm{MmMR}(X, \Theta_{\mathcal{P}'}) &\leq \mathrm{MmMR}(X, \Theta_{\mathcal{P}}) \end{aligned}$$

and so, the maximum minimax regret cannot increase by adding new preference statements; in the subsection devoted to numerical tests, we will see that, in practice, it strictly decreases when queries are chosen in a reasoned way.

In order to evaluate the relevance of a query q, one can make use of a notion of myopic value of information defined as follows:

Definition 11. *The worst-case maximum minimax regret (WMmMR) of q is:*

$$\mathrm{WMmMR}(q, \Theta_{\mathcal{P}}) = \max_{p \in \mathcal{P}_q} \mathrm{MmMR}(\Theta_{\mathcal{P} \cup \{p\}})$$

where $\mathcal{P}_q$ denotes the set of all possible answers to query q.

Then, the next query should be chosen in:

$$\arg\min_{q \in Q} \mathrm{WMmMR}(q, \Theta_{\mathcal{P}})$$

where Q denotes the set of all possible queries, because it ensures the best reduction of maximum minimax regret (MmMR) in the answer's worst-case scenario. However, when the number of queries under consideration is too large, the computation of the optimal query can be computationally intensive; hence, we may want to consider heuristics of the WMmMR criterion.

To achieve this, one possibility is to ask the DM to compare well chosen alternatives. Nevertheless, the optimal assignment (i.e. such that each alternative is assigned to its MR-optimal category) does not directly suggest the choice of the comparison query; the "semantics" of sorting problems is radically different from that of choice problems (there is neither a notion of adversarial choice for an individual alternative, nor for a complete assignment).

However, we can design strategies for generating queries in sorting problems that are similar to the Current Solution Strategy in choice problems (see Section 2.3). We can ask the DM, at each iteration, to classify one alternative associated to the highest minimax regret value mMR.

4.2. Determination of the optimal assignment

The computation of max regrets (and therefore minimax regret mMR) can be efficiently performed by exploiting the following property of MR.

Proposition 8. *For any alternative $x \in X$, it holds*

$$\mathrm{MR}(x, K_\ell, \Theta_\mathcal{P}) = \max\Big\{f_x^\top - \alpha_{\ell-1},\ \alpha_\ell - f_x^\perp,\ 0\Big\}$$

where $f_x^\top = \max\limits_{\theta\in\Theta_\mathcal{P}} f_\theta(x)$ *and* $f_x^\perp = \min\limits_{\theta\in\Theta_\mathcal{P}} f_\theta(x)$.

Proof. For any solution $x \in X$:

$$\begin{aligned}
\mathrm{MR}(x, K_\ell, \Theta_\mathcal{P}) &= \max_{\theta\in\Theta_\mathcal{P}} \mathrm{R}(x, K_\ell, \theta)\\
&= \max_{\theta\in\Theta_\mathcal{P}} \max\Big\{f_\theta(x) - \alpha_{\ell-1},\ \alpha_\ell - f_\theta(x),\ 0\Big\}\\
&= \max\Big\{\max_{\theta\in\Theta_\mathcal{P}}\{f_\theta(x) - \alpha_{\ell-1}\}, \max_{\theta\in\Theta_\mathcal{P}}\{\alpha_\ell - f_\theta(x)\},\ 0\Big\}\\
&= \max\Big\{\max_{\theta\in\Theta_\mathcal{P}}\{f_\theta(x)\} - \alpha_{\ell-1},\ \alpha_\ell - \min_{\theta\in\Theta_\mathcal{P}}\{f_\theta(x)\},\ 0\Big\}\\
&= \max\Big\{f_x^\top - \alpha_{\ell-1},\ \alpha_\ell - f_x^\perp,\ 0\Big\}.
\end{aligned}$$

□

This means that, in order to determine the MR-optimal category for x, it is sufficient to compute $\max_{\theta\in\Theta_\mathcal{P}} f_\theta(x)$ and $\min_{\theta\in\Theta_\mathcal{P}} f_\theta(x)$, and then compute $\mathrm{MR}(x, K_\ell, \Theta_\mathcal{P})$ according to Proposition 8 for each category K_ℓ, $\ell \in \{1, \ldots, q\}$. Actually, the next property allows to simplify even more the optimization task.

Proposition 9. *For any $x \in X$, the category K_ℓ such that*

$$\frac{f_x^\top + f_x^\perp}{2} \in [\alpha_\ell, \alpha_{\ell-1}]$$

is the MR*-optimal category for x, where* $f_x^\top = \max\limits_{\theta\in\Theta_\mathcal{P}} f_\theta(x)$ *and* $f_x^\perp = \min\limits_{\theta\in\Theta_\mathcal{P}} f_\theta(x)$.

Proof. We want to prove that $\mathrm{MR}(x, K_\ell, \Theta_\mathcal{P}) \leq \mathrm{MR}(x, K_k, \Theta_\mathcal{P})$ holds for any $k \in \{1, \ldots, q\}\backslash\{\ell\}$. First, we prove that $\mathrm{MR}(x, K_\ell, \Theta_\mathcal{P}) \leq (f_x^\top - f_x^\perp)/2$ holds.

On the one hand, since $(f_x^\top + f_x^\perp)/2 \leq \alpha_{\ell-1}$ by definition of category K_ℓ, we have $f_x^\top - \alpha_{\ell-1} \leq f_x^\top - (f_x^\top + f_x^\perp)/2 = (f_x^\top - f_x^\perp)/2$. On the other hand, since we also have $\alpha_\ell \leq (f_x^\top + f_x^\perp)/2$ by definition of category K_ℓ, we have $\alpha_\ell - f_x^\perp \leq (f_x^\top + f_x^\perp)/2 - f_x^\perp = (f_x^\top - f_x^\perp)/2$. Hence, we can deduce that we have $\mathrm{MR}(x, K_\ell, \Theta_\mathcal{P}) \leq (f_x^\top - f_x^\perp)/2$ from Proposition 8.

Therefore, to prove that $\mathrm{MR}(x, K_\ell, \Theta_\mathcal{P}) \leq \mathrm{MR}(x, K_k, \Theta_\mathcal{P})$ holds for any $k \in \{1, \ldots, q\}\backslash\{\ell\}$, it is sufficient to prove that $\mathrm{MR}(x, K_k, \Theta_\mathcal{P}) \geq (f_x^\top - f_x^\perp)/2$

holds for any $k \in \{1, \ldots, q\} \backslash \{\ell\}$. For any $k > \ell$, we have:

$$\begin{aligned}
\mathrm{MR}(x, K_k, \Theta_{\mathcal{P}}) &\geq f_x^{\top} - \alpha_{k-1} \quad \text{by Proposition 8} \\
&= f_x^{\top} - \alpha_{\ell} + \alpha_{\ell} - \alpha_{k-1} \\
&\geq f_x^{\top} - (f_x^{\top} + f_x^{\perp})/2 + \alpha_{\ell} - \alpha_{k-1} \quad \text{by definition of } K_{\ell} \\
&= (f_x^{\top} - f_x^{\perp})/2 + \alpha_{\ell} - \alpha_{k-1} \\
&\geq (f_x^{\top} - f_x^{\perp})/2 \quad \text{since } \ell \leq k-1
\end{aligned}$$

For any $k < \ell$, we have:

$$\begin{aligned}
\mathrm{MR}(x, K_k, \Theta_{\mathcal{P}}) &\geq \alpha_k - f_x^{\perp} \quad \text{by Proposition 8} \\
&= \alpha_k - \alpha_{\ell-1} + \alpha_{\ell-1} - f_x^{\perp} \\
&\geq \alpha_k - \alpha_{\ell-1} + (f_x^{\top} + f_x^{\perp})/2 - f_x^{\perp} \quad \text{by definition of } K_{\ell} \\
&= \alpha_k - \alpha_{\ell-1} + (f_x^{\top} - f_x^{\perp})/2 \\
&\geq (f_x^{\top} - f_x^{\perp})/2 \quad \text{since } \ell - 1 \geq k
\end{aligned}$$

Hence, we have $\mathrm{MR}(x, K_k, \Theta_{\mathcal{P}}) \geq (f_x^{\top} - f_x^{\perp})/2$ for any $k \in \{1, \ldots, q\} \backslash \{\ell\}$. Therefore, category K_{ℓ} is the MR-optimal category for x. □

In order to determine the MR-optimal category, thanks to Proposition 9, it is sufficient to compute $(f_x^{\top} + f_x^{\perp})/2$ and then to identify the category K_{ℓ} whose thresholds enclose this value. Note that if f_{θ} is a linear function in θ, then any preference of type "$x \in K_{\ell}$", denoted (x, K_{ℓ}), induces two linear constraints on the parameter space: $f_{\theta}(x) \geq \alpha_{\ell}$ and $f_{\theta}(x) \leq \alpha_{\ell-1}$. Similarly, any preference of type "a is preferred to b" impose the linear constraint $f_{\theta}(a) \geq f_{\theta}(b)$. In that case, set $\Theta_{\mathcal{P}}$ is described by linear constraints, and so the two optimization problems $f_x^{\top} = \max_{\theta \in \Theta_{\mathcal{P}}} f_{\theta}(x)$ and $f_x^{\perp} = \min_{\theta \in \Theta_{\mathcal{P}}} f_{\theta}(x)$ can be formulated as linear programs (and therefore solved very quickly for simple utility models such as linear utilities; we will discuss the use of Choquet integrals in the next section).

4.3. Application to Choquet integrals

In the following, we assume that the DM's preferences can be modeled by a Choquet integral; i.e. the aggregation of criteria is performed by computing the Choquet integral with a capacity v:

$$f_{\theta}(x) := C_v(x)$$

where the uncertain capacity function v takes the role of θ and the space of parameters $\Theta_{\mathcal{P}}$ consists in the set of all normalized capacities compatible with the observed preferences $\mathcal{P}$.

In Section 4.1, we introduced the notion of maximum minimax regret for sorting problems, that represents an overall aggregate score to assess the quality

of an assignment. According to Proposition 9 (see Section 4.2), in order to compute the maximum minimax regret MmMR, one needs to compute $f_x^\top$ and $f_x^\perp$ for each alternative $x \in X$. The optimization of $f_x^\top$ for Choquet integrals can be performed by solving the following linear program:

$$
\begin{aligned}
& \max_{v} && C_v(x) && (26)\\
& s.t. && v_\emptyset = 0 && (27)\\
(\mathrm{LP}_4) \qquad & && v_N = 1 && (28)\\
& && v_A \leq v_{A\cup\{i\}},\ \forall A \subset N,\ \forall i \in N\backslash A && (29)\\
& && C_v(a) \geq C_v(b),\ \forall (a,b) \in \mathcal{P} && (30)\\
& && \alpha_\ell \leq C_v(a) \leq \alpha_{\ell-1},\ \forall (a, K_\ell) \in \mathcal{P} && (31)
\end{aligned}
$$

Similarly, the optimization of $f_x^\perp$ for Choquet integrals can performed by solving linear program LP_4 where the objective function has to be minimized. Note that Equations (27-29) ensure that v is a normalized capacity and Equations (30-31) ensure that v is compatible with $\mathcal{P}$. We remark that the number of variables and constraints in LP_4 are exponential in the number of criteria, due to the monotonicity constraints. Nevertheless, we can again obtain a more compact formulation by considering preference queries involving binary alternatives of type $1A0$, with $A \subseteq N$, and constant utility profiles of type $\Lambda = (\lambda, \ldots, \lambda)$.

In linear program LP_4 allowing to compute $f_x^\top$ and $f_x^\perp$ for a given x, we notice that the set of variables involved in the objective function is the set $\mathcal{A}_x$ of all level sets of x, i.e.

$$\mathcal{A}_x = \{X_{(i)}, i \in N\}.$$

Therefore, in a way analogous to what we have done in Section 3.1, we are able to reformulate this linear program in a way that the numbers of variables and monotonicity constraints are drastically reduced. More precisely, we have seen that if we restrict ourself to preference queries that ask to compare a binary alternative with a constant utility profile, then preference constraints (Equations (30-31)) can be replaced by boundary constraints of type

$$l_A \leq v_A \leq u_A$$

for all subsets of criteria $A \subseteq N$, where $l_A \leq l_B$ and $u_A \leq u_B$ for all $A \subset B \subseteq N$. Using Proposition 1 with $\mathcal{A} = \mathcal{A}_x$, from the linear program we can remove all monotonicity constraints involving a variable v_A that is not present in the objective function of Equation (26) (i.e. $v_A \notin \mathcal{A}_x$), as they play no role in the optimization; we then obtain a more compact formulation with only n variables and $n-1$ constraints given here below:

$$
\begin{aligned}
& \max_{v_{X_{(i)}}, i\in N} && \sum_{i=1}^{n} (x_{(i)} - x_{(i-1)}) v_{Y_{(i)}} && \\
(\mathrm{LP}_5) \qquad & s.t. && v_{X_{(i+1)}} \leq v_{X_{(i)}},\ \forall i \in [\![1, n-1]\!] && (32)\\
& && l_{X_{(i)}} \leq v_{X_{(i)}} \leq u_{X_{(i)}},\ \forall i \in N &&
\end{aligned}
$$

We now show that this compact formulation can be optimized without using a LP-solver by exploiting their specific structure. This allows to compute $f_x^\top$ and $f_x^\perp$ in a more efficient way.

Proposition 10. *The solution $v_A = u_A$ for all $A \in \mathcal{A}_x$ is the optimal solution of the max version of LP$_5$. The solution $v_A = l_A$ for all $A \in \mathcal{A}_x$ is the optimal solution of the min version of LP$_5$.*

Proof. We will only explicitely prove the first point as the proof of the second is very similar. For all $A \in \mathcal{A}_x$, let w_A denote the coefficient of decision variable v_A in linear program LP$_5$; the objective function is then equal to $\sum_{A\in\mathcal{A}_x} w_A v_A$. Recall that $w_A = x_{(i)} - x_{(i-1)} \geq 0$ for all $A \in \mathcal{A}_x = \{X_{(i)}, i \in N\}$ by definition of Choquet integrals. Since $w_A \geq 0$ and $v_A \in [l_A, u_A]$ for all $A \in \mathcal{A}_x$, the value $\sum_{A\in\mathcal{A}_x} w_A u_A$ is an upper bound of the optimal solution of the max version of LP$_5$. Now, we want to prove that $v_A = u_A$ holds for all $A \in \mathcal{A}_x$ is a feasible solution of LP$_5$, i.e. we want to prove that all monotonicity constraints (32) are satisfied by this solution. Note that these constraints are satisfied if and only if we have $u_{X_{(i+1)}} \leq u_{X_{(i)}}$ for all $i \in [\![1, n-1]\!]$. Therefore, since we know that $u_A \leq u_B$ holds for all $A \subseteq B \subseteq N$ by construction of all intervals $[l_A, u_A]$, $A \subseteq N$, the latter condition is true and establishes the result. □

As a consequence, linear program LP$_5$ can be solved without using a LP-solver. More precisely, to solve the maximization (resp. minimization) problem, it is enough to consider the instantiation where $v_A = u_A$ for all $A \subseteq N$ (resp. $v_A = l_A$ for all $A \subseteq N$), and to calculate the Choquet integral of x (i.e. $\sum_{A\in\mathcal{A}_x} w_A v_A$).

Example 3. *Consider a problem defined on 5 criteria and 5 categories delimited by the following thresholds: $\alpha_0 = 1$, $\alpha_1 = 0.9$, $\alpha_2 = 0.7$, $\alpha_3 = 0.4$, $\alpha_4 = 0.2$ and $\alpha_5 = 0$. We want to determine the* MR*-optimal category of alternative $x = (1, 0.8, 0.4, 0.5, 0.1)$; intervals $[l_A, u_A]$, $A \in \{X_{(i)}, i \in N\}$, are the same as those in Example 2. The computation of $f_x^\top = \max_{v\in\Theta_\mathcal{P}} C_v(x)$ can be performed by solving linear program LP$_5$ given here below:*

$$\begin{array}{ll} \max\limits_{v} & 0.1v_N + 0.3v_{\{1,2,3,4\}} + 0.1v_{\{1,2,4\}} + 0.3v_{\{1,2\}} + 0.2v_{\{1\}} \\ s.t. & v_{\{1\}} \leq v_{\{1,2\}} \leq v_{\{1,2,4\}} \leq v_{\{1,2,3,4\}} \leq v_N \\ & 0 \leq v_{\{1\}} \leq 0.5,\ 0.2 \leq v_{\{1,2\}} \leq 0.7,\ 0.3 \leq v_{\{1,2,4\}} \leq 0.9, \\ & 0.6 \leq v_{\{1,2,3,4\}} \leq 1,\ 1 \leq v_N \leq 1 \end{array}$$

We can compute $f_x^\perp = \min_{v\in\Theta_\mathcal{P}} C_v(x)$ by solving the min version of the latter program. These optimizations can be performed without using a LP-solver since the associated optimal solutions are given in Proposition 10. More precisely:

$$\begin{aligned} f_x^\top &= 0.1\ u_N + 0.3\ u_{\{1,2,3,4\}} + 0.1\ u_{\{1,2,4\}} + 0.3\ u_{\{1,2\}} + 0.2\ u_{\{1\}} \\ &= 0.1 \times 1 + 0.3 \times 1 + 0.1 \times 0.9 + 0.3 \times 0.7 + 0.2 \times 0.5 \\ &= 0.8 \end{aligned}$$

$$
\begin{aligned}
f_x^{\perp} &= 0.1\, l_N + 0.3\, l_{\{1,2,3,4\}} + 0.1\, l_{\{1,2,4\}} + 0.3\, l_{\{1,2\}} + 0.2\, l_{\{1\}} \\
&= 0.1 \times 1 + 0.3 \times 0.6 + 0.1 \times 0.3 + 0.3 \times 0.2 + 0.2 \times 0 \\
&= 0.37
\end{aligned}
$$

Since we have $(f_x^{\top} + f_x^{\perp})/2 = 0.585 \in [\alpha_3, \alpha_2]$*, category* K_3 *is the* MR*-optimal category of alternative* x *(see Figure 6 for illustration).*

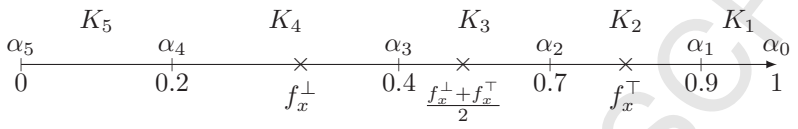


Figure 6: A characteriztion of Example 3 showing the thresholds and the values $f_x^{\top}$ and $f_x^{\perp}$.

4.4. A query generation strategy for Choquet capacity elicitation

In the previous subsection, we have shown that the computation of MmMR can be performed efficiently when $\mathcal{P}$ is a set of preference statements of types $(1A0, \Lambda)$ or $(\Lambda, 1A0)$. Assuming that the DM is only asked to compare binary alternatives to constant profiles, we now have to identify the pair (A, λ) that yield the most informative query. To this end, we propose a query generation strategy based on the WMmMR criterion presented in Definition 11. According to this criterion, an optimal query is defined by a pair $(A \subseteq N, \lambda \in [l_A, u_A])$ that minimizes the maximum minimax regret (MmMR) in the worst-case scenario with respect to all possible answers. In other words, to determine this pair, for all sets $A \subseteq N$ we have to determine the following value:

$$
\begin{aligned}
\lambda_A &= \arg \min_{\lambda \in [l_A, u_A]} \mathrm{WMmMR}((A, \lambda), \Theta_{\mathcal{P}}) \\
&= \arg \min_{\lambda \in [l_A, u_A]} \max \Big\{ \mathrm{MmMR}(X, \Theta_{\mathcal{P} \cup \{(1A0, \Lambda)\}}), \mathrm{MmMR}(X, \Theta_{\mathcal{P} \cup \{(\Lambda, 1A0)\}}) \Big\}
\end{aligned}
$$

and then, we arbitrarily select A^* in $\arg\min_{A \subseteq N} \mathrm{WMmMR}((A, \lambda_A), \Theta_{\mathcal{P}})$; by doing so, pair $(A^*,\ \lambda_{A^*})$ defines an optimal query for the WMmMR criterion.

In fact, for a given set $A \subseteq N$, the determination of the optimal value λ_A can be easily performed using a bisection algorithm, applying similar arguments to those of Section 3.4. The functions $\mathrm{MmMR}(X, \Theta_{\mathcal{P} \cup \{(1A0, \Lambda)\}})$ and $\mathrm{MmMR}(X, \Theta_{\mathcal{P} \cup \{(\Lambda, 1A0)\}})$ are indeed, respectively, weakly decreasing and weakly increasing, while achieving the same maximum. Hence, in order to determine the optimal pair $(A^*,\ \lambda_{A^*})$ at each iteration step of the elicitation procedure, we need to use the bisection algorithm considering that A could be any of the $2^n - 2$ proper subsets of N. However, the number of these sets grows exponentially as the number of criteria increases. Therefore, as a heuristic, we propose to consider only the subsets of N that are involved in the computation of $\mathrm{mMR}(x, \Theta_{\mathcal{P}})$, where x is an alternative with the highest minimax regret mMR given the current $\mathcal{P}$ (the alternative responsible of the current MmMR value). These sets

are the n level sets of x, i.e. $\mathcal{A}_x = \{X_{(i)}, i \in N\}$. In this way, the heuristic will further constrain the parameters involved in the computation of $\text{mMR}(x, \Theta_{\mathcal{P}})$ which may reduce the current MmMR value.

4.5. Numerical tests

The first experiments aim at evaluating the efficiency of the query generation strategy presented in Section 4.4 and based on comparison queries. This strategy is named CQ hereafter. We do not report computation times since we have already seen that optimization with preference statements of types $(1A0, \Lambda)$ or $(\Lambda, 1A0)$ is very efficient, due to the reduced number of monotonicity constraints. This is even more striking witout making use of a LP-solver.

Recall that CQ relies on queries involving binary alternatives $1A0$ and constant utility profiles $\Lambda = (\lambda, \ldots, \lambda)$. Hence, for a baseline comparison, we consider the random generation strategy where both $A \subseteq N$ and $\lambda \in [l_A, u_A]$ are selected at random at each iteration step of the elicitation procedure. Starting from an empty set of preference statements, simulated DMs answer queries according to a randomly generated Choquet integral. At each iteration step, in addition to the maximum minimax regret MmMR, we compute the *maximum real regret*, that is the largest actual loss of utility associated with assigning an alternative to its regret-optimal category instead of its true preferred category (the utility is measured by a latent Choquet integral modeling the DM's preferences). As for choice problems, regrets are normalized to belong to the unit interval. The results averaged over 100 runs are given in Figure 7 for datasets with 1000 alternatives; performance vectors of alternatives are uniformly drawn in $[0, 1]^n$ and categories are defined by dividing the utility scale $[0, 1]$ uniformly.

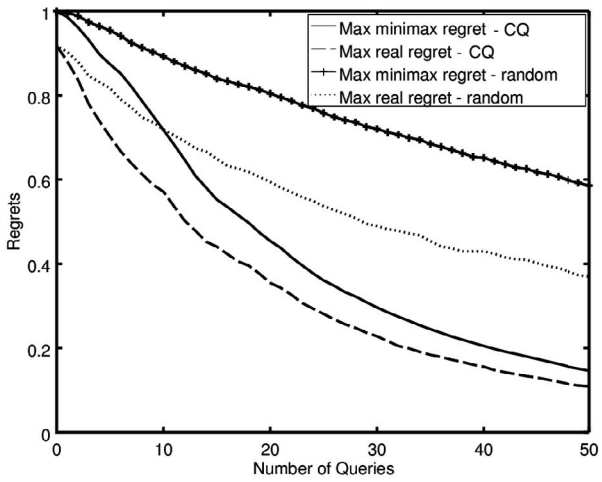


Figure 7: Comparison with the random strategy ($n = 5$, 1000 alternatives, 10 categories).

In Figure 7, we can see that the maximum minimax regret MmMR reduces much faster with CQ than with the random generation satrategy; the same observation applies when considering the maximum real regret with the two strategies. After about 30 queries on average, the maximum real regret observed with CQ is under 20% of the initial regret while still being larger than 50% with the random strategy. Note that the maximum real regret reduces less drastically than the real regret in choice problems (see Figure 3). However, we observe that the average of real regrets (averaged over all alternatives that need to be classified) is below 10% after only 5 queries on average, meaning that most alternatives are well assigned reasonably quickly.

Recall that CQ focuses on alternative x inducing the current MmMR value by generating a preference query involving one of its level sets. Hence, we propose an alternative query generation strategy (named AQ for assignment queries) asking the DM which among all the categories suits most alternative x; for this strategy, max regrets are performed by optimizing the general linear programming formulations LP_4 presented in Section 4.3 because the compact formulation LP_5 does not apply due to the type of questions used. Figure 8 compares strategy CQ with AQ (the results are averaged over 100 runs).

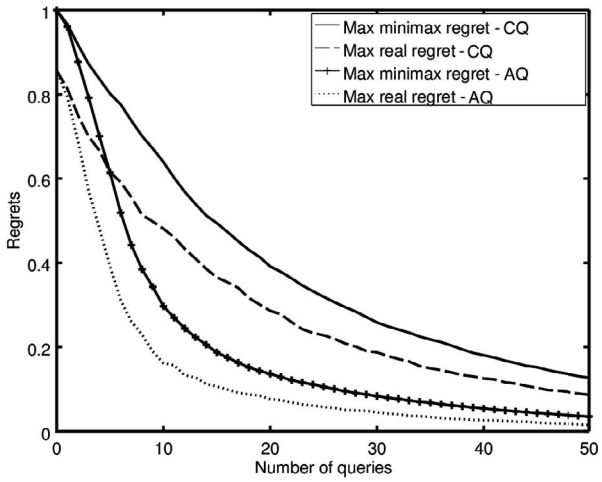


Figure 8: Comparison with the strategy asking the category of the alternative associated with maximal minimax regret ($n = 5$, 150 alternatives, 10 categories).

In Figure 8, we can see here that strategy AQ is more informative than CQ since both the maximum minimax regret and the maximum real regret reduce more quickly. Note however that AQ only applies to small sorting problems involving a few criteria (so that monotonicity constraints can be handled efficiently) and a few hundred alternatives (due to the number of LP optimizations). For larger problems, strategy CQ is more appropriate as it is computationally

much less demanding (due to the reduced number of monotonicity constraints) while reducing regret reasonably fast.

5. Discussion and conclusions

In this paper, we discussed the problem of interactively eliciting the parameters of a Choquet integral in the context of multicriteria decision-making. We presented how an interactive elicitation approach can be applied to the following problems: recommending a single alternative to a decision maker (choice problems), producing a ranking of top-k alternatives (ranking problems) and assigning alternatives to a number of ordered categories (sorting problems). We have adopted the minimax regret approach of Boutilier et. al. [29] allowing robust recommendations under uncertainty with guarantees with respect to the worst-case loss, and formalized ranking and sorting problems in terms of regret.

Using minimax regret in combination with a utility model based on a Choquet integral poses a number of technical difficulties; these are related to the number of parameters needed to characterize a Choquet capacity and the number of constraints required to characterize the space of admissible capacity functions. We have shown that, assuming that preferences are stated in a particular form (involving a binary alternative of type 1A0 and a constant utility profile of type $\Lambda = (\lambda, \ldots, \lambda)$), minimax regret optimizations can be performed efficiently in various settings (choice, ranking, sorting): we presented both a linear programming formulation and an even faster iterative algorithm maintaining lower and upper bounds. We presented experimental results validating the practical efficiency of our incremental approaches in terms of computation times, number of queries and quality of decisions. Recall that we stop when it can be proved that further specifications of the model cannot seriously challenge the current recommendation. That is where we really save time with respect to approaches that fully elicit the utility function, making elicitation feasible in practice. In our opinion, this is the specificity and the contribution of incremental elicitation for decision-making.

Our work differentiates from previous works on Choquet integrals in the focus on incremental elicitation and on the ability to provide robust recommendations using minimax regret without making any restrictive assumption on the capacity. Notably, Ah-Pine et al. [62] assess a feasible capacity for a Choquet integral given some preferential information that maximize the margin of the induced constraints (in a fashion similar to SVM classifiers). However this kind of "pointwise" estimation ignore the specificity of the available alternatives (while regret-based approach can focus elicitation on the "useful" part of the utility). Moreover, it does not directly provide a natural strategy for choosing the query to ask within an incremental elicitation setting.

The problem of minimizing regrets to derive a robust recommendation with a Choquet integral and an imprecise capacity has also been addressed in [63]. In this work, another approach to compute $\mathrm{MR}(x, \mathcal{X}, \Theta_{\mathcal{P}})$ is proposed, relying

on the fact that $\mathrm{MR}(x, \mathcal{X}, \Theta_{\mathcal{P}}) = \max_{v \in \Theta^*_{\mathcal{P}}} \max_{y \in \mathcal{X}} \{C_v(y) - C_v(x)\}$ and $\Theta^*_{\mathcal{P}}$ is the (finite) set of extreme points in $\Theta_{\mathcal{P}}$. However, the amount of extreme points seems to be prohibitively large for enumeration methods. Hence, using this approach within an incremental elicitation procedure seems unfeasible for general capacities. This approach is however interesting for two-additive capacities because, in this case, the size of $\Theta^*_{\mathcal{P}}$ is quadratic in the number of criteria. The advantage of our approach, beside the fact that it is incremental, is that it applies to any monotone capacity, without any prior restriction.

A first direct continuation of this work is to extend the elicitation procedure for set recommendation, following the work on setwise minimax regret [40]. Indeed, while most emphasis of works in recommender systems and decision aid is on providing a single recommendation, it is often appropriate to provide a set of alternatives. The approach we have proposed in this paper extends naturally to sets but is computationally more demanding.

A second direction concerns the development of new strategies to search for highly informative queries. We are interested in the sequential evaluation of the informative value of a query; note, however, that sequential optimization of the value of a query will usually be prohibitive in most cases. Alternatively, it will be interesting to test highly informative local search approaches like the *query-iteration strategy* [64, 40] in our settings.

In this paper, we also addressed the problem of sorting using thresholds on the overall utility scale. These thresholds are assumed to be defined in a preliminary step with the DM, independently of the set of alternatives. For instance one may want to construct an overall intrinsic evaluation scale using the unit interval (e.g. for evaluating students, projects...) where 0.5 is the neutral point separating good and bad alternatives; one may also want that excellent alternatives receive at least 0.8 and that very bad alternatives receive at most 0.2. Using such thresholds, we have studied the set of capacities that are compatible with the assignment examples obtained from the DM and developed an incremental elicitation method to progressively reduce this set. It would be possible to learn or approximate thresholds in the same time as capacity values which would possibly provide greater flexibility to describe an assignment (but would prevent us to use thresholds prescribed by the DM). The variable thresholds approach has been proposed for instance in [58] for linear aggregators and could be adapted for Choquet integrals. This would however make thresholds and capacities interdependent and both could be influenced by the learning set. Moreover, in our approach, the fact that thresholds are known allows us to considerably simplify the optimization of regrets (see Section 4.3), which speeds-up computation and provides good interaction possibilities.

In the literature on multicriteria sorting, another prominent approach for preference-based sorting has been proposed and widely used in the context of multicriteria evaluation: *sorting with reference profiles* as proposed by Roy in the Electre TRI method [65] and used in multiple variants see, e.g., [66, 67, 57, 68, 69]. In such methods, we are given multicriteria *profiles* describing

the "desiderata" (in terms of criteria) for each category; the alternatives are compared to these profiles on each criterion to derive preference indices that are then aggregated to establish the overall preference. Profiles act as multicriteria boundaries of categories used to make preference-based assignements. Various elicitation procedures have been proposed to assess some parameters in these models (e.g., weight of criteria, reference profiles), see e.g., [66, 70, 71]. Recently, the elicitation of Choquet capacities has been also studied in this context [72] but the proposed approach is not incremental. The approach we are proposing here for the incremental elicitation of capacities could easily be adapted to multicriteria sorting models based on comparisons with reference profiles.

It is worth noting that the absence of any redundancy in preference queries is a key aspect to obtain efficient questionnaires. We remind indeed that the DM is asked to compare a pair of alternatives (x, y) only when both answers "x preferred to y" and "y preferred to x" are consistent with the preferences collected so far. Hence, we implicitly enforce consistency of stated preference statements and inferred preferences. However, we do not check the internal consistency of the DM during the elicitation process nor the adequacy of the Choquet model to her preferences. Revisiting incremental approaches to manage possible internal inconsistencies of the DM (noisy preferences) and to offer the possibility to falsify the decision model (testing consistency of the observed preferences w.r.t. the Choquet model) while keeping fast elicitation sequences would be an interesting but challenging line for further research.

Finally, an interesting direction of research, departing from the regret-based approach, would be incremental elicitation of Choquet capacities using a Bayesian approach (following works on Bayesian utility elicitation [26, 73]), adopting a less conservative criterion for selecting preference queries under uncertainty. It would not provide the same guarantee on the robustness of decisions but could possibly reduce the average number of preference queries in the elicitation process by considering the expected value of information instead of performing a worst case analysis.

6. Acknowledgments

We wish to thank the reviewers for their very detailed feedback and their useful recommendations. This work has been supported by the French National Research Agency through the Idex Sorbonne Universités under grant ANR-11-IDEX-0004-02.

Appendix

Proof of Proposition 2. Let v^* be an optimal solution of program LP'_2. Assume that v^* do not satisfy Equation (17) for some $A_0 \in \mathcal{A}_{(x,y)}$ such that $\omega_{A_0} > 0$. Let $\hat{v}$ be the solution defined by:

- $\hat{v}_A = v^*_A$ for all $A \in \mathcal{A}_{(x,y)}$ such that $\omega_A < 0$.
- $\hat{v}_A = \min\{u_A, \min_{A' \in Pa(A)} \hat{v}_{A'}\}$ for all $A \in \mathcal{A}_{(x,y)}$ such that $\omega_A > 0$.

Hence, solution $\hat{v}$ satisfies Equation (17) for all $A \in \mathcal{A}_{(x,y)}$ such that $\omega_A > 0$. First, we want to prove that $\hat{v}$ is also a feasible solution of program LP'_2. To do so, we just need to prove that Equations (11), (12), (14), (15) and (16) are satisfied by $\hat{v}$:

- *Equation (11):* this equation is verified if $\hat{v}_A \leq \hat{v}_B$ for all $A, B \in \mathcal{A}_{(x,y)}$ such that $A \subset B$, $\omega_A < 0$ and $\omega_B < 0$. Let $A, B \in \mathcal{A}_{(x,y)}$ be such that $A \subset B$, $\omega_A < 0$ and $\omega_B < 0$. Since $A \subset B$ and v^* is a feasible solution, we necessarily have $v^*_A \leq v^*_B$ due to Equation (11). Then, the result is simply obtained by using the fact that we have $\hat{v}_C = v^*_C$ for all $C \in \mathcal{A}_{(x,y)}$ such that $\omega_C < 0$.
- *Equation (12):* this equation is satisfied if $\hat{v}_A \leq \hat{v}_B$ for all $A, B \in \mathcal{A}_{(x,y)}$ such that $A \subset B$, $\omega_A > 0$ and $\omega_B > 0$. Let $A, B \in \mathcal{A}_{(x,y)}$ be such that $A \subset B$, $\omega_A > 0$ and $\omega_B > 0$. Since $A \subset B$, we necessarily have $u_A \leq u_B$ and $Pa(B) \subseteq Pa(A)$. Therefore, using Equation (17), we obtain:

$$\hat{v}_A = \min\{u_A, \min_{A' \in Pa(A)} \hat{v}_{A'}\} \leq \min\{u_B, \min_{B' \in Pa(B)} \hat{v}_{B'}\} = \hat{v}_B$$

- *Equation (14):* this equation is verified if $\hat{v}_A \leq \hat{v}_B$ for all $A, B \in \mathcal{A}_{(x,y)}$ such that $A \subset B$, $\omega_A > 0$ and $\omega_B < 0$. The result directly follows from Equation (17) since B is an element of $Pa(A)$.
- *Equations (15) and (16)* : we want to prove $l_A \leq \hat{v}_A \leq u_A$ for all $A \in \mathcal{A}_{(x,y)}$. Note that, for all $A \in \mathcal{A}_{(x,y)}$ such that $\omega_A < 0$, we necessarily have $l_A \leq \hat{v}_A \leq u_A$ since $\hat{v}_A = v^*_A$ and v^* is a feasible solution. Let $A \in \mathcal{A}_{(x,y)}$ be such that $\omega_A > 0$. By definition, we have $\hat{v}_A \leq u_A$ (see Equation (17)). Therefore, we just need to prove $\hat{v}_A \geq l_A$. Since $l_{A'} \geq l_A$ for all $A' \in Pa(A)$, we have:

$$\hat{v}_A = \min\{u_A, \min_{A' \in Pa(A)} \hat{v}_{A'}\} \geq \min\{l_A, \min_{A' \in Pa(A)} l_{A'}\} \geq l_A$$

Thus, $\hat{v}$ is a feasible solution of program LP'_2. Now, we want to prove that $\hat{v}$ is strictly better than v^*, i.e. we want to show that the following inequality holds:

$$\sum_{A \in \mathcal{A}_{(x,y)}} \omega_A \hat{v}_A > \sum_{A \in \mathcal{A}_{(x,y)}} \omega_A v^*_A$$

Since $v^*_A = \hat{v}_A$ for all $A \in \mathcal{A}_{(x,y)}$ such that $\omega_A < 0$, it is sufficient to prove that $v^*_A \leq \hat{v}_A$ for all $A \in \mathcal{A}_{(x,y)}$ such that $\omega_A > 0$ and that $v^*_A < \hat{v}_A$ for some $A \in \mathcal{A}_{(x,y)}$ such that $\omega_A > 0$. Note that, since v^* is a feasible solution, we necessarily have $v^*_A \leq u_A$ (due to Equation (16)) and $v^*_A \leq \min_{A' \in Pa(A)} v^*_{A'}$ (due to Equation (14)). Therefore:

$$v^*_A \leq \min\{u_A, \min_{A' \in Pa(A)} v^*_{A'}\}$$

Then, since $v^*_{A'} = \hat{v}_{A'}$ for all $A' \in Pa(A)$, we obtain:

$$v^*_A \leq \min\{u_A, \min_{A' \in Pa(A)} v^*_{A'}\} = \min\{u_A, \min_{A' \in Pa(A)} \hat{v}_{A'}\} = \hat{v}_A$$

Note that the first inequality is necessarily strict if solution v^* do not satisfy Equation (17). By hypothesis, there exists some set $A \in \mathcal{A}_{(x,y)}, \omega_A > 0$, such that v^* do not verify Equation (17). Therefore, we conclude that $\hat{v}$ is strictly better than v^*, which contradicts the hypothesis: v^* cannot be an optimal solution of program LP'_2. □

Proof of Proposition 4. Let us prove by induction that the following statement, denoted by $P(i)$, holds at the end of step $i \in \{0, \ldots, |A^-|\}$:

$$\forall A \in A^+, v_A = \min\{u_A, \min_{A' \in Pa_i(A)} v_{A'}\}$$

For step $i = 0$ (before entering the loop), statement $P(i)$ obviously holds since $Pa_i(A) = \emptyset$ and v_A is initialized to u_A for all $A \in A^+$. Assume $P(i-1)$ holds for some $i \in \{1, \ldots, |A^-| - 1\}$. We want to prove that $P(i)$ is necessarily true in that case. Let $A \in A^+$. At the beginning of step i, we know that $v_A = \min\{u_A, \min_{A' \in Pa_{i-1}(A)} v_{A'}\}$ by induction hypothesis. Let us remark that none of the variables $v_{A'}$, $A' \in Pa_{i-1}(A)$, is modified during this step. If $A \notin A^-_i$, then variable v_A is not modified during step i. Moreover, since $Pa_i(A) = Pa_{i-1}(A)$ in that case, we can directly infer the result. Assume now that $A \subset A^-_i$. During step i, variable v_A is modified as follows (see line 17):

$$\begin{aligned} v_A &= \min\{\min\{u_A, \min_{A' \in Pa_{i-1}(A)} v_{A'}\}, v_{A^-_i}\} \\ &= \min\{u_A, \min_{A' \in Pa_{i-1}(A) \cup \{A^-_i\}} v_{A'}\} \\ &= \min\{u_A, \min_{A' \in Pa_i(A)} v_{A'}\} \end{aligned}$$

Thus, statement $P(i)$ holds. Therefore, for all $i \in \{0, \ldots, |A^-|\}$, we have $v_A = \min\{u_A, \min_{A' \in Pa_i(A)} v_{A'}\}$ for all $A \in A^+$. □

Proof of Proposition 6. Let us prove that Algorithm 1 constructs an optimal solution of LP_3. To do so, it is sufficient to prove by induction that the following statement, denoted by $P(i)$, holds at the end of any step $i \in \{0, \ldots, |A^-|\}$ of the 'for' loop:

"The instantiation of the variables in $\{v_{A_k^-}, 1 \leq k \leq i\}$, denoted by I_i, can be extended to an optimal solution of program LP_3."

If statement $P(i)$ is true for $i = |A^-|$, then it enables us to establish the result: the instantiation of all negative variables (i.e. $\{v_A : A \in A^-\}$) can be extended to an optimal instantiation, while the remaining variables, which are those with a positive impact on the objective function (i.e. $\{v_A : A \in A^+\}$), verify the necessary condition for optimality (due to Proposition 5).

For iteration step $i = 0$ (i.e. before entering the loop), statement $P(i)$ obviously holds since set $\{v_{A_k^-}, 1 \leq k \leq i\}$ is empty. Assume now that $P(i-1)$ holds for some $i \in \{1, \ldots, |A^-|\}$. We want to prove that $P(i)$ is true. Note that none of the variables in $\{v_{A_k^-}, 1 \leq k \leq i-1\}$ is updated during step i. Moreover, the instantiation I_{i-1} can be extended to an optimal solution of program LP_3 by induction hypothesis. Therefore, we just need to prove that variable $v_{A_i^-}$ is instantiated in such a way that instantiation I_i can still be extended to an optimal solution of LP_3.

At iteration step i, a while loop is used to determine the value of $v_{A_i^-}$ which iterates over the positive descendants of node A_i^- in decreasing order of size. Two cases may occur: either the value of $v_{A_i^-}$ is fixed at line 12 or at line 4. To simplifly the proof, we will only consider the first case, since the second one can be proved using very similar arguments (we will come back to this point later). Thus, we assume here that the while loop stops at some step $j \in \{1, \ldots, |D^i|\}$ due to the condition in line 11. In that case, we know that $v_{A_i^-}$ is set to u_j^i, where u_j^i denotes the value of variable $v_{D_j^i}$ at the beginning of iteration step i (see line 12). Note that we have:

$$u_j^i = \min\{u_{D_j^i}, \min_{A \in Pa_{i-1}(D_j^i)} v_A\}$$

where $Pa_{i-1}(D_j^i) = \{A_k^-, 1 \leq k \leq i-1 : D_j^i \subset A_k^-\}$ (due to Proposition 4). Therefore, u_j^i actually represents the maximum feasible value of variable $v_{D_j^i}$ given the instantiation of the variables in $\{v_A : A \in Pa_{i-1}(D_j^i)\}$. Note that we have $\{v_A : A \in Pa_{i-1}(D_j^i)\} \subseteq \{v_{A_k^-} : 1 \leq k \leq i-1\}$. As a consequence, since instantiation I_{i-1} can be extended to an optimal solution of program LP_3 (by induction hypothesis), we can impose $v_{D_j^i} \leq u_j^i$ while still being able to find an optimal solution of program LP_3. Hence, we can find an optimal solution of LP_3 by solving the following problem:

$$\max_{u \in [l_{D_j^i}, u_j^i]} f(u, \mathcal{A}_{(x,y)}) \tag{33}$$

where $f(u, \mathcal{A}_{(x,y)})$ denotes the optimal solution of the following subproblem:

$$\begin{aligned} \max_{v_A, A \in \mathcal{A}_{(x,y)}} \quad & \sum_{A \in \mathcal{A}_{(x,y)}} \omega_A v_A \\ \text{s.t.} \quad & v_{D_j^i} = u \\ & \text{Equations } (18-22) \end{aligned} \tag{34}$$

We now aim to show that that there exists an optimal solution of program LP_3 that extends instantiation I_{i-1} while verifying $v_{A_i^-} = v_{D_j^i} = u_j^i$. With this aim in mind, we want to prove that $f(u, \mathcal{A}_{(x,y)})$ is maximized for $u = u_j^i$.

Let us focus on the computation of $f(u, \mathcal{A}_{(x,y)})$ for some fixed $u \in [l_{D_j^i}, u_j^i]$. We define the following sets:

- $S(D_j^i) = \{A \in \mathcal{A}_{(x,y)} : A \supseteq D_j^i\}$: the supersets of D_j^i restricted to those in $\mathcal{A}_{(x,y)}$. Each element of $S(D_j^i)$ is associated to a variable that is now bounded below by u (due to the constraint $v_{D_j^i} = u$).

- $P(D_j^i) = \{A \in \mathcal{A}_{(x,y)} : A \subset D_j^i\}$: the powerset of D_j^i restricted to set $\mathcal{A}_{(x,y)}$. Each element of $P(D_j^i)$ corresponds to a variable that is now bounded above by u (due to the constraint $v_{D_j^i} = u$).

- $I(D_j^i) = \{A \in \mathcal{A}_{(x,y)} : A \not\supseteq D_j^i, A \not\subset D_j^i\}$: the sets that are incomparable to D_j^i. Note that we necessarily have $I(D_j^i) \cap A^+ = \emptyset$ since $D_j^i \in A^+$ and A^+ is an embedded sequence. Therefore, we have $I(D_j^i) \subseteq A^-$ which means that $\omega_A < 0$ for all $A \in I(D_j^i)$; hence, the objective function of problem (34) increases as variable v_A decreases. We want to prove that there exists an optimal solution of problem (34) such that $v_A \leq \max\{l_A, u\}$ for all $A \in I(D_j^i)$. Let $A \in I(D_j^i)$. Assume first that there exists no set $A' \in \mathcal{A}_{(x,y)}$ such that $A' \subset A$. In that case, Equations (18-22) do not include any constraint of type $v_{A'} \leq v_A$. Therefore, since the objective function increases as variable v_A decreases, we know that there exists an optimal solution such that $v_A = l_A \leq \max\{l_A, u_A\}$. Assume now that there exists $A' \in \mathcal{A}_{(x,y)}$ such that $A' \subset A$. First, we want to prove that imposing $v_A \leq \max\{l_A, u\}$ does not impact on variable $v_{A'}$. Two cases may occur: either $\omega_{A'} > 0$ or $\omega_{A'} < 0$. Let $A' \in \mathcal{A}_{(x,y)}$ be such that $A' \subset A$ and $\omega_{A'} > 0$ (it it exists). Note that $A' \subset D_j^i$ must hold, as otherwise we would have $D_j^i \subset A$ and $A \in I(D_j^i)$ (which yields a contradiction). As a consequence, we necessarily have $A' \in P(D_j^i)$; hence, $v_{A'}$ is bounded above by u. Therefore, imposing $v_A \leq \max\{l_A, u\}$ does not impact on variable $v_{A'}$. Let $A' \in \mathcal{A}_{(x,y)}$ be such that $A' \subset A$ and $\omega_{A'} < 0$ (it it exists). If $A' \in P(D_j^i)$, then $v_{A'}$ is bounded above by u and so imposing $v_A \leq \max\{u, l_A\}$ does not impact on variable $v_{A'}$. Otherwise, we necessarily have $A' \in I(D_j^i)$ and so the result can be obtained by iterating this complete reasoning on A': $v_{A'} \leq \max\{l_{A'}, u\} \leq \max\{l_A, u\}$. Hence, we can impose $v_A \leq \max\{u, l_A\}$ without impacting on variable

$v_{A'}$. As a consequence, since the objective function increases as variable v_A decreases, we know that there exists an optimal solution such that $v_A \leq \max\{l_A, u\}$.

Thus, set $\mathcal{A}_{(x,y)}$ can be decomposed into three disjoint sets $S(D_j^i)$, $P(D_j^i)$ and $I(D_j^i)$ such that:

$$f(u, \mathcal{A}_{(x,y)}) = h(u, P(D_j^i), I(D_j^i)) + g(u, S(D_j^i))$$

where $h(u, P(D_j^i), I(D_j^i))$ denotes the optimum of the following problem:

$$\max_{\substack{v_A, A \in P(D_j^i) \cup I(D_j^i) \\ \forall A \in P(D_j^i), v_A \leq u \\ \forall A \in I(D_j^i), v_A \leq \max\{l_A, u\}}} \sum_{A \in P(D_j^i) \cup I(D_j^i)} \omega_A v_A$$

and $g(u, S(D_j^i))$ denotes the optimal value of the following problem:

$$\max_{\substack{v_A, A \in S(D_j^i) \\ \forall A \in S(D_j^i), v_A \geq u \\ v_{D_j^i} = u}} \sum_{A \in S(D_j^i)} \omega_A v_A$$

Equations (18-22) are here omitted to simplify the presentation. Note that these two subproblems have no variable in common; hence, Equations (18-22) are actually restricted to the variables that are involved in the considered subproblem.

We can observe that $h(u, P(D_j^i), I(D_j^i))$ is an increasing function of u since constraints of types "$v_A \leq u$" or "$v_A \leq \max\{l_A,\ u\}$" become less constraining as u increases. Therefore, $h(u, P(D_j^i), I(D_j^i))$ is maximized for $u = u_j^i$. As a consequence, in order to prove that $f(u, \mathcal{A}_{(x,y)})$ is maximized for $u = u_j^i$, it is sufficient to show that $g(u, S(D_j^i))$ is maximized for $u = u_j^i$.

First, let us simplify the optimization problem. By induction hypothesis, all variables in $\{v_{A_k^-} : 1 \leq k \leq i-1\}$ have already been "correctly" instantiated. Therefore, we can remove all these variables from the optimization and we just have to enforce consistency of the next instantiations. Moreover, given the instantiation of these variables, we can derive the optimal value of all variables in $\{v_A, A \in A^+ : A \not\subset A_i^-\}$ from the necessary condition (17); hence, these variables can also be removed from the optimization problem. We now want to show that $g(u, s(D_j^i))$ is maximized for $u = u_j^i$ where:

$$s(D_j^i) = S(D_j^i) \backslash (\{A_k^- : 1 \leq k \leq i-1\} \cup \{A \in A^+ : A \not\subset A_i^-\})$$

We first assume that u is a feasible value of all variables v_A, $A \in s(D_j^i)$, i.e. $l_A \leq u \leq u_A$ for all $A \in s(D_j^i)$. On that assumption, we aim to prove that there exists $\alpha > 0$ such that $g(u, s(D_j^i)) = \alpha \times u$. We decompose $s(D_j^i)$ into two disjoints sets $s(D_j^i)^+$ and $s(D_j^i)^-$ defined as follows:

- $s(D_j^i)^+ = A^+ \cap s(D_j^i) = \{D_m^i : 1 \leq m \leq j\}$. This set is composed of the j first positive descendants of node A_i^-.
- $s(D_j^i)^- = A^- \cap s(D_j^i) = \{A_k^- : i \leq k \leq K\}$, where K is the largest value k such that $i \leq k \leq |A^-|$ and $A_k^- \supset D_j^i$. This set is composed of all $A \in A^-$ such that $D_j^i \subset A \subseteq A_i^-$.

First, we want to prove that all variables in $\{v_A : A \in s(D_j^i)^-\}$ can be removed from the optimization problem. Note that, for all $k \in \{i, \ldots, K\}$, D_1^k the first positive descendant of A_k^- is necessarily included in $s(D_j^i)^+$ since D_j^i is positive and $D_j^i \subset A_k^- \subseteq A_i^-$. Therefore, we can decompose set $s(D_j^i)^-$ into j disjoint sets $s(D_j^i)_1^-, \ldots, s(D_j^i)_j^-$, defined as follows:

$$\forall m \in \{1, \ldots, j\},\ s(D_j^i)_m^- = \{A_k^-, i \leq k \leq K : D_1^k = D_m^i\}$$

Note that $\omega_A < 0$ for all $A \in s(D_j^i)_m^-$, $m \in \{1, \ldots, j\}$; therefore, $g(u, s(D_j^i))$ increases as v_A decreases. Moreover, by definition of set $s(D_j^i)_m^-$, we have:

$$D_m^i \subset A \text{ and } \nexists A' \in A^+, D_m^i \subset A' \subset A$$

Therefore, due to the monotonicity constraints, $v_A = \max\{l_A, v_{D_m^i}\}$ must hold at the optimum point. Then, since $v_{D_m^i} \geq u$ (by definition of $S(D_j^i)$) and $u \geq l_A$ (by hypothesis), we can conclude $v_A = v_{D_m^i}$ at the optimum point. As a consequence, variables $v_A, A \in s(D_j^i)_m^-$, can be substituted by $v_{D_m^i}$ in the optimization problem. More precisely, $g(u, s(D_j^i))$ can be computed by solving the following problem:

$$\max_{\substack{v_{D_m^i}, m \in \{1,\ldots,j\} \\ \forall m \in \{1,\ldots,j-1\}: v_{D_m^i} \geq u \\ v_{D_j^i} = u}} \sum_{m=1}^{j} \left[\left(\omega_{D_m^i} + \sum_{A \in s(D_j^i)_m^-} \omega_A \right) v_{D_m^i} \right] \tag{35}$$

Note that this optimization problem only involves the positive variables $v_{D_m^i}, m \in \{1, \ldots, j\}$. We want to prove that there exists an optimal solution of this sub-problem such that $v_{D_1^i} = \ldots = v_{D_j^i}$. To do so, we first study the term associated with $v_{D_1^i}$ in the objective function defined by Equation (35). Note that its coefficient is actually equal to $\omega^+ + \omega^-$, where ω^+ and ω^- are defined at the first step of the while loop. Since the while loop stops at step j due to line 11, we know that $\omega^+ + \omega^- \leq 0$ at the first step (unless $j = 1$ of course). Therefore, we know that the objective function defined by Equation (35) increases as $v_{D_1^i}$ decreases. Moreover, since $v_{D_1^i}$ must be larger than or equal to $v_{D_2^i}$ due to Equation (19), then we know that there exists an optimal solution such that $v_{D_1^i} = \max\{l_{D_1^i}, v_{D_2^i}\}$; then, since $v_{D_2^i} \geq u$ (by definition of g) and $u \geq l_{D_1^i}$ (by hypothesis), we can conclude $v_{D_1^i} = v_{D_2^i}$. Therefore, we can substitute $v_{D_1^i}$ by

$v_{D_2^i}$ in the problem, which leads to the following objective function:

$$\Big(\sum_{m=1}^{2}\omega_{D_m^i}+\sum_{A\in s(D_j^i)_1^-\cup s(D_j^i)_2^-}\omega_A\Big)v_{D_2^i}+\sum_{m=3}^{j}\Big[(\omega_{D_m^i}+\sum_{A\in s(D_j^i)_m^-}\omega_A)v_{D_m^i}\Big]$$

By iterating this reasoning, we can conclude that:

$$g(u,s(D_j^i))=\max_{v_{D_j^i}=u}\Big\{\Big(\sum_{A\in s(D_j^i)}\omega_A\Big)v_{D_j^i}\Big\}=\alpha\times u$$

where $\alpha=\sum_{A\in s(D_j^i)}\omega_A$. This equality only holds under the assumption that $u\geq l_A$ for all $A\in s(D_j^i)$; otherwise, some values $v_A, A\in s(D_j^i)$, have been authorized to decrease below their minimal feasible value (to improve the objective function), which only provides the following inequality: $g(u,s(D_j^i))\leq\alpha\times u$. However, this equality is true for $u=u_j^i$: we indeed have $u_j^i\geq l_{A_i^-}$ due to line 8 and $l_{A_i^-}\geq l_A$ for all $A\in s(D_j^i)$ since $A_i^-\supset A$. Moreover, since the while loop stops at step j due to line 11, we know that $\alpha>0$ since $\omega^+ + \omega^- = \alpha = \sum_{A\in s(D_j^i)}\omega_A$ at step j. As a consequence, for all $u\neq u_j^i$, we have:

$$g(u,s(D_j^i))\leq\alpha\times u<\alpha\times u_j^i=g(u_j^i,s(D_j^i))$$

This shows that function $g(u,s(D_j^i))$ is maximized for $u=u_j^i$; therefore, function $f(u,\mathcal{A}_{(x,y)})$ is also maximized for $u=u_j^i$. Thus, when $v_{A_i^-}$ is set to u_j^i (due to the condition in line 11), there exists an optimal solution extending I_{i-1} such that $v_{A_i^-}=v_{D_1^i}=v_{D_j^i}=u_j^i$, which means that $P(i)$ holds in that case. When $v_{A_i^-}$ is set to $l_{A_i^-}$ (due to line 4), two cases may occur:

- If $D^i=\emptyset$ or $v_{D_1^i}\leq l_{A_i^-}$, then no positive variable conflicts with $v_{A_i^-}$. Moreover, since $\omega_{A_i^-}<0$, we know that the objective function strictly increases as $v_{A_i^-}$ decreases. Therefore, $v_{A_i^-}$ must be set to its lower bound $l_{A_i^-}$ in that case.
- Otherwise, we can use the same arguments as those we used when $v_{A_i^-}$ is set to u_j^i (due to line 12). More precisely, a similar reasoning with $j=M$ establishes the result, where M is the largest value m such that $1\leq m\leq|D^i|$ and $v_{D_m^i}>l_{A_i^-}$. The main difference is that we obtain a decreasing function of u at the end (since all tests in line 11 fail) and so variable $v_{A_i^-}$ must be set to its lower bound $l_{A_i^-}$ instead.

Therefore, $P(i)$ holds in all cases. Hence, our algorithm is valid. □

References

[1] N. Benabbou, P. Perny, P. Viappiani, Incremental elicitation of Choquet capacities for multicriteria decision making, in: Proceedings of ECAI'14, 2014, pp. 87–92.

[2] D. Schmeidler, Integral representation without additivity, Proceedings of the American Mathematical Society 97 (2) (1986) 255–261.

[3] M. E. Yaari, The dual theory of choice under risk, Econometrica 55 (1987) 95–115.

[4] J. Quiggin, Generalized Expected Utility Theory, Kluwer Academic Publishers, 1989.

[5] M. Grabisch, The application of fuzzy integrals in multicriteria decision making, European Journal of Operational Research 89 (3) (1996) 445–456.

[6] M. Grabisch, J.-L. Marichal, R. Mesiar, E. Pap, Aggregation Functions (Encyclopedia of Mathematics and Its Applications), Cambridge University Press, New York, NY, USA, 2009.

[7] V. Torra, The weighted OWA operator, International Journal of Intelligent Systems 12 (1997) 153–166.

[8] R. Yager, On Ordered Weighted Averaging aggregation operators in multicriteria decision making, IEEE Transactions on Systems, Man and Cybernetics 18 (1998) 183–190.

[9] M. Grabisch, C. Labreuche, A decade of application of the Choquet and Sugeno integrals in multi-criteria decision aid, Annals of Operations Research 175 (1) (2010) 247–286.

[10] A. F. Tehrani, W. Cheng, K. Dembczynski, E. Hüllermeier, Learning monotone nonlinear models using the choquet integral, Machine Learning 89 (1-2) (2012) 183–211.

[11] G. Beliakov, T. Calvo, S. James, Aggregation functions for recommender systems, in: Recommender Systems Handbook, 2015, pp. 777–808.

[12] J. Dubus, C. Gonzales, P. Perny, Choquet optimization using GAI networks for multiagent/multicriteria decision-making, in: Algorithmic Decision Theory, First International Conference, ADT 2009, Venice, Italy, October 20-23, 2009. Proceedings, 2009, pp. 377–389.

[13] V. Torra, Y. Narukawa, Modeling decisions - information fusion and aggregation operators, Springer, 2007.

[14] L. Galand, P. Perny, Search for choquet-optimal paths under uncertainty, in: UAI 2007, Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence, Vancouver, BC, Canada, July 19-22, 2007, 2007, pp. 125–132.

[15] M. Grabisch, H. Nguyen, E. Walker, Fundamentals of Uncertainty Calculi, with Applications, Encyclopedia of Mathematics and its Applications, Kluwer Academic Publishers, 1995.

[16] J.-L. Marichal, M. Roubens, Determination of weights of interacting criteria from a reference set, European Journal of Operational Research 124 (3) (2000) 641–650.

[17] P. Meyer, M. Roubens, On the use of the Choquet integral with fuzzy numbers in multiple criteria decision support, Fuzzy Sets and Systems 157 (7) (2006) 927–938.

[18] S. Greco, V. Mousseau, R. Slowinski, Ordinal regression revisited: Multiple criteria ranking using a set of additive value functions, European Journal of Operational Research 191 (2) (2008) 416–436.

[19] A. F. Tehrani, W. Cheng, K. Dembczynski, E. Hüllermeier, Learning monotone nonlinear models using the Choquet integral, Machine Learning 89 (1-2) (2012) 183–211.

[20] E. Hüllermeier, A. Fallah Tehrani, Efficient learning of classifiers based on the 2-additive Choquet integral, in: Computational Intelligence in Intelligent Data Analysis Studies in Computational Intelligence Volume, Vol. 445, 2013, pp. 17–29.

[21] S. Greco, V. Mousseau, R. Slowinski, Robust ordinal regression for value functions handling interacting criteria, European Journal of Operational Research 239 (3) (2014) 711–730.

[22] M. Grabisch, J.-L. Marichal, R. Mesiar, E. Pap, Aggregation Functions, Encyclopedia of Mathematics and its Applications, Cambridge University Press, New-York, 2009.

[23] J.-L. Marichal, P. Meyer, M. Roubens, Sorting multi-attribute alternatives: the TOMASO method, Computers & Operations Research 32 (2005) 861–877.

[24] F. Huédé, M. Grabisch, C. Labreuche, P. Savéant, Integration and propagation of a multi-criteria decision making model in constraint programming, Journal of Heuristics 12 (4-5) (2006) 329–346.

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

[26] U. Chajewska, D. Koller, R. Parr, Making Rational Decisions Using Adaptive Utility Elicitation, in: Proceedings of AAAI'00, 2000, pp. 363–369.

[27] C. Boutilier, A POMDP Formulation of Preference Elicitation Problems, in: Proceedings of AAAI'02, 2002, pp. 239–246.

[28] T. Wang, C. Boutilier, Incremental Utility Elicitation with the Minimax Regret Decision Criterion, in: Proceedings of IJCAI'03, 2003, pp. 309–316.

[29] C. Boutilier, R. Patrascu, P. Poupart, D. Schuurmans, Constraint-based Optimization and Utility Elicitation using the Minimax Decision Criterion, Artifical Intelligence 170 (8–9) (2006) 686–713.

[30] D. Braziunas, C. Boutilier, Assessing regret-based preference elicitation with the utpref recommendation system, in: Proceedings 11th ACM Conference on Electronic Commerce (EC-2010), 2010, pp. 219–228.

[31] J. Fürnkranz, E. Hüllermeier, Preference learning: An introduction, in: J. Fürnkranz, E. Hüllermeier (Eds.), Preference Learning, Springer Berlin Heidelberg, 2011, pp. 1–17.

[32] L. Galand, P. Perny, O. Spanjaard, Choquet-based optimisation in multiobjective shortest path and spanning tree problems, European Journal of Operational Research 204 (2) (2010) 303–315.

[33] M. Timonin, Maximization of the Choquet integral over a convex set and its application to resource allocation problems, Annals of Operations Research 196 (2012) 543–579.

[34] L. Galand, J. Lesca, P. Perny, Dominance rules for the Choquet integral in multiobjective dynamic programming, in: Proceedings of IJCAI'13, 2013, pp. 538–544.

[35] J. Lesca, M. Minoux, P. Perny, Compact versus noncompact LP formulations for minimizing convex Choquet integrals, Discrete Applied Mathematics 161 (1-2) (2013) 184–199.

[36] L. J. Savage, The Foundations of Statistics, Wiley, New York, 1954.

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

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

[39] D. Braziunas, Decision-theoretic elicitation of generalized additive utilities, Ph.D. thesis, University of Toronto (2011).

[40] P. Viappiani, C. Boutilier, Regret-based optimal recommendation sets in conversational recommender systems, in: Proceedings of the third ACM conference on Recommender systems, ACM, 2009, pp. 101–108.

[41] T. Lu, C. Boutilier, Robust approximation and incremental elicitation in voting protocols, in: Proceedings of IJCAI'11, 2011, pp. 287–293.

[42] J. Drummond, C. Boutilier, Elicitation and approximately stable matching with partial preferences, in: Proceedings of IJCAI'13, 2013, pp. 97–105.

[43] N. Argyris, A. Morton, J. R. Figueira, CUT: A multicriteria approach for concavifiable preferences, Operations Research 62 (3) (2014) 633–642.

[44] R. Dechter, From local to global consistency, Artificial intelligence 55 (1992) 87–107.

[45] W. W. Cohen, R. E. Schapire, Y. Singer, Learning to order things, Journal of Artificial Intelligence Research 10 (1) (1999) 243–270.

[46] R. Herbrich, T. Graepel, K. Obermayer, Large Margin Rank Boundaries for Ordinal Regression, MIT Press, 2000, Ch. 7, pp. 115–132.

[47] E. Hüllermeier, J. Fürnkranz, W. Cheng, K. Brinker, Label ranking by learning pairwise preferences, Artificial Intelligence 172 (16–17) (2008) 1897 – 1916.

[48] D. E. Critchlow, M. A. Fligner, J. S. Verducci, Probability models on rankings, Journal of Mathematical Psychology 35 (3) (1991) 294 – 318.

[49] P. Fishburn, Utility theory for decision making, Publications in operations research, Wiley, 1970.

[50] T. Saaty, The Analytic Hierarchy Process, Planning, Piority Setting, Resource Allocation, McGraw-Hill, New york, 1980.

[51] E. Jacquet-Lagrèze, Y. Siskos, Assessing a set of additive utility functions for multicriteria decision making: the UTA method, European Journal of Operational Research 10 (1982) 151–164.

[52] J.-P. Brans, P. Vincke, A preference ranking organisation method: (the PROMETHEE method for multiple criteria decision-making), Management science 31 (6) (1985) 647–656.

[53] R. L. Keeney, H. Raiffa, Decisions with multiple objectives: preferences and value trade-offs, Cambridge university press, 1993.

[54] B. Roy, Multicriteria Methodology for Decision Analysis, Kluwer Academic Publishers, 1996.

[55] E. Jacquet-Lagreze, An application of the UTA discriminant model for the evaluation of R&D projects, in: Advances in Multicriteria Analysis, Nonconvex Optimization and Its Applications, Springer US, 1995, pp. 203–211.

[56] C. Zopounidis, M. Doumpos, A multicriteria decision aid methodology for sorting decision problems: The case of financial distress, Computational Economics 14 (3) (1999) 197–218.

[57] C. Zopounidis, M. Doumpos, Multicriteria classification and sorting methods: A literature review, European Journal of Operational Research 138 (2) (2002) 229–246.

[58] S. Greco, V. Mousseau, R. Slowinski, Multiple criteria sorting with a set of additive value functions, European Journal of Operational Research 207 (3) (2010) 1455–1470.

[59] J. Fürnkranz, E. Hüllermeier, S. Vanderlooy, Binary decomposition methods for multipartite ranking, in: W. Buntine, M. Grobelnik, D. Mladenić, J. Shawe-Taylor (Eds.), Machine Learning and Knowledge Discovery in Databases, Vol. 5781 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2009, pp. 359–374.

[60] J. Quevedo, E. Montañés, O. Luaces, J. del Coz, Adapting decision DAGs for multipartite ranking, in: J. Balcázar, F. Bonchi, A. Gionis, M. Sebag (Eds.), Machine Learning and Knowledge Discovery in Databases, Vol. 6323 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2010, pp. 115–130.

[61] K. Uematsu, Y. Lee, Statistical optimality in multipartite ranking and ordinal regression, Pattern Analysis and Machine Intelligence, IEEE Transactions on 37 (5) (2015) 1080–1094.

[62] J. Ah-Pine, B. Mayag, A. Rolland, Identification of a 2-additive bi-capacity by using mathematical programming, in: Algorithmic Decision Theory, Springer, 2013, pp. 15–29.

[63] M. Timonin, Robust optimization of the choquet integral, Fuzzy Sets and Systems 213 (2013) 27–46.

[64] P. Viappiani, C. Boutilier, Recommendation sets and choice queries: There is no exploration/exploitation tradeoff!, in: Proceedings of AAAI'11, 2011, pp. 1571–1574.

[65] B. Roy, A multicriteria analysis for trichotomic segmentation problems, in: Multiple Criteria Analysis, P. Nijkamp and J. Spronk (eds), Gaver, 1981, pp. 245–257.

[66] V. Mousseau, R. Slowinski, Inferring an ELECTRE-TRI model from assignment examples, Journal of Global Optimization 12 (2) (1998) 157–174.

[67] P. Perny, Multicriteria filtering methods based onconcordance and nondiscordance principles, Annals of operations Research 80 (1998) 137–165.

[68] D. Bouyssou, T. Marchant, An axiomatic approach to noncompensatory sorting methods in MCDM, I: The case of two categories, European Journal of Operational Research 178 (1) (2007) 217 – 245.

[69] D. Bouyssou, T. Marchant, An axiomatic approach to noncompensatory sorting methods in MCDM, II: More than two categories, European Journal of Operational Research 178 (1) (2007) 246 – 276.

[70] V. Mousseau, R. Slowinski, P. Zielniewicz, A useroriented implementation of the ELECTRE-TRI method integrating preference elicitation support, Computers and Operations Research 27 (2000) 757–777.

[71] O. Sobrie, V. Mousseau, M. Pirlot, Learning a majority rule model from large sets of assignment examples, in: Proceedings of ADT'13, 2013, pp. 336–350.

[72] O. Sobrie, V. Mousseau, M. Pirlot, Learning the parameters of a non compensatory sorting model, in: Proceedings of ADT'15, 2015, pp. 153–170.

[73] P. Viappiani, C. Boutilier, Optimal Bayesian recommendation sets and myopically optimal choice query sets, in: Advances in Neural Information Processing Systems 23 (NIPS), 2010, pp. 2352–2360.