Université Libre de Bruxelles

CoDE - SMG

An adaptive questioning procedure for eliciting PROMETHEE IIs weight parameters

CoDE-SMG – Technical Report Series

Stefan Eppe, Yves De Smet

CoDE-SMG – Technical Report Series

Technical Report No.
TR/SMG/2012-002

June 2012

CoDE-SMG – Technical Report Series
ISSN 2030-6296

Published by:
CoDE-SMG, CP 210/01
Université Libre de Bruxelles
Bvd du Triomphe
1050 Ixelles, Belgium

Technical report number TR/SMG/2012-002

An adaptive questioning procedure for eliciting PROMETHEE IIs weight parameters

CoDE-SMG – Technical Report Series

Stefan Eppe stefan.eppe@ulb.ac.be

Yves De Smet yves.de.smet@ulb.ac.be

CoDE-SMG, Université Libre de Bruxelles, Brussels, Belgium

June 2012

An adaptive questioning procedure for eliciting PROMETHEE II's weight parameters

Stefan Eppe* and Yves De Smet

Technical Report -- June 2012

Abstract

For most decision making problems, finding representative parameters of a decision maker's preferences remains a challenging task. We here adapt to the PROMETHEE II outranking method an eliciting procedure named *Q-Eval* that has been developed in the context of multi-attribute utility. It refines the estimation of the preference parameters by iteratively querying the DM through pairwise action comparisons. After formalising an extension of this method to two other types of queries: (i) selection from an action sub-set; (ii) ranking of a sub-set of actions, we propose an adaptive query selection scheme that presents to the DM the most discriminating query type at each step of the process. Simulation results show that this approach improves the efficiency of the eliciting phase in terms of convergence speed.

1 Introduction

In the context of Multicriteria Decision Aid (MCDA), the use of a preference model aims at aggregating the information we have on each action in such a way that it allows to represent the preferences of a decision maker (DM). The main difficulty of so-called preference eliciting procedures is to interactively determine a set of robust parameters that represent the DM's preferences, but without putting too much (cognitive) burden on him with questions that he is not capable and/or willing to answer. It has to be noted that choosing a preference model already represents a decision that impacts the output of the decision making process. Finding a best possible set of preference parameters will therefore always depend on the actually used model.

Aggregration-disaggregation based methods (Bous et al., 2010; Dias et al., 2002; Jacquet-Lagrèze and Siskos, 2011; Mousseau, 2003; Mousseau and Słowiński, 1998) are one of the most widely used approaches to eliciting preferences. The main philosophy of this family of methods is to induce quantitative information about a DM's preferences, i.e., quantitative estimations of the preference parameters, from holistic queries that let him express partial preferences.

In the present paper, we consider the PROMETHEE II method (Section 2) that outputs a complete ranking over the set of considered actions. To the best of our knowledge, only a few approaches for eliciting PROMETHEE II preference parameters exist at present (Eppe et al., 2011; Frikha et al., 2010; Kadziński et al., 2012; Sun and Han, 2010). Our goal is to contribute in this direction by proposing a procedure for eliciting PROMETHEE II's weight parameters, applying the same methodology and extending the results of Eppe and De Smet (2012). Practically, our method adapts *Q-Eval* an aggregation-disaggregation eliciting method that has been proposed in the context of Multi-Attribute Utility Theory (MAUT) by Iyengar et al. (2001). While the original method generates pairwise action comparison queries, i.e., questions that ask the DM to state his preference between two presented actions, our approach handles additional types of queries (Section 3).

We also further investigate the features of the search space (of preference parameters) of the eliciting problem. In particular, we address questions of the type:

*Corresponding author at Computer & Decision Engineering department, Polytechnic School of Brussels, Université Libre de Bruxelles, Belgium. E-mail: `stefan.eppe@ulb.ac.be`

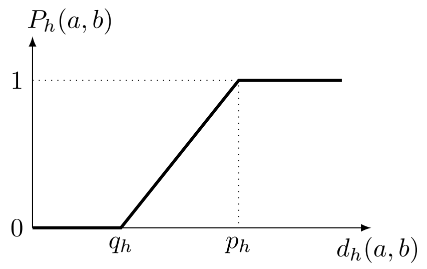


Figure 1: Shape of a PROMETHEE preference function type V, with, for each objective h, an indifference threshold q_h, and a preference threshold p_h. For a maximisation problem and $\mathrm{d}_h(a,b) = f_h(a) - f_h(b) \in [0, q_h]$, both solutions a and b are considered indifferently on criterion h; for a difference greater than p_h, a strict preference (with value 1) of a over b is stated. Between the two thresholds, the preference function evolves linearly with increasing evaluation difference.

- "*For* PROMETHEE *II, what is the impact of the information structure — the type and amount of partial preference information provided by the DM — on the quality of elicited preference parameters?*"
- "*Are there some types of preference information that are contributing better at specific stages of the eleciting process? In other words, is there are preferable way (type, quantity, and order) of questions to the decision maker that optimizes the outcome of an elicitation process?*" ("Query selection criteria", Braziunas (2006))

After briefly presenting the PROMETHEE II preference model (Section 2), we describe the eliciting process we want to investigate (Section 3). Finally, we discuss the results (Section 5).

2 The PROMETHEE II preference model

We consider a set $A = \{a_1 \dots a_n\}$ of $n = |A|$ actions that are evaluated over m criteria by means of m objective functions: $f_h : A \to \mathbb{R} : a \to f_h(a), \forall h \in \{1 \dots m\}$.

PROMETHEE II is a widely used method (Behzadian et al., 2010; Brans and Mareschal, 2005), which provides the DM with a complete ranking over the set A of actions. The net flow $\Phi(a)$ associated with an action $a \in A$ is defined by: $\Phi(a) = \frac{1}{n-1} \sum_{b \in A \setminus a} \sum_{h=1}^{m} w_h \left(P_h(a,b) - P_h(b,a)\right)$, where w_h and $P_h(a,b)$ are respectively the relative weight and the preference function (Figure 1) for criteria $h \in \{1 \dots m\}$. For any pair of actions $(a,b) \in A \times A$, we have one of the following relations: (i) the rank of action a is better than the rank of action b, iff $\Phi(a) > \Phi(b)$; (ii) the rank of action b is better than the rank of action a, iff $\Phi(a) < \Phi(b)$; (iii) action a has the same rank as action b, iff $\Phi(a) = \Phi(b)$. A preference function $P_h : A \times A \to \mathbb{R}_{[0,1]}$, which is defined for each criterion individually, has two main purposes: first, it normalizes the evaluation function, which ensures commensurability of different criteria of different natures; second, it introduces (for most defined preference functions) one ore two threshold values, that allow the decision maker to tune the preference relation on each criterion.

For our present purpose, we rewrite the definition in terms of *unicriterion net flows*, which will ease the formalisation of the linear program (LP) we are going to use for the elicitation process.

$$\begin{aligned}
\Phi(a_i) &= \tfrac{1}{n-1} \sum_{j=1}^{n} \sum_{h=1}^{m} w_h \left(P_h\left(f_h(a_i) - f_h(a_j)\right)\right) - P_h\left(f_h(a_j) - f_h(a_i)\right)) \\
&= \sum_{h=1}^{m} w_h \left[\tfrac{1}{n-1} \sum_{j=1}^{n} \left(P_h\left(f_h(a_i) - f_h(a_j)\right)\right) - P_h\left(f_h(a_j) - f_h(a_i)\right)) \right] \\
&= \sum_{h=1}^{m} w_h \Phi_h(a_i)
\end{aligned}$$

where $\Phi_h(a_i)$ is the *unicriterion net flow* of action a_i considered on criterion h.

As already mentioned, PROMETHEE II requires several types of preference parameters: in any case, a weight vector has to be given, that assigns a relative importance for each criterion with respect to the others. Although different preference functions have been proposed be the original authors (Brans and Mareschal (2005)), we restrict ourselves to one of the most widely used ones for actual applications: the "*V-shape*" preference function (Figure 1). For a detailed presentation of the PROMETHEEmethods, we refer the reader to Brans and Mareschal (2005).

3 The preference eliciting process

After stating the problem at hand in general terms and defining the notations used, we mainly describe in this section the types of queries that will be considered in our experiments as well as the constraint their answers generate on the parameter space. The practical details as well as the actually implemented eliciting procedure are described in Section 4.

Stated in a very general way, we consider a preference model $\mathcal{M}$ which aims at aggregating the evaluations of each action in such a way that it reflects the preferences of a decision maker (DM) in the most accurate possible manner. In this paper we focus our attention on the ranking problem: our goal is thus to use an MCDA aggregation process to assign a scalar value to each action that represents its relative preference degree with respect to the other actions. In Multiple Attribute Theory (MAUT), this preference degree is called the *utility* of an action, while being labelled the *net flow* in the PROMETHEE II outranking method. For most preference models, the actual preferences of a DM are represented by a set of parameters, which depend on the chosen model. As already mentioned, determining the values of those parameters is far from trivial, and poses several challenging problems: (i) the cognitive difficulty for the DM to express his preferences; (ii) an eliciting procedure that is "making the most of the provided information"; etc.

Let us denote d the number of parameters that has to be provided by the DM and that depends on the action set A and on the preference model $\mathcal{M}$. For PROMETHEE II (with the "*V-shape*" preference function), a problem on m criteria requires $\mathrm{d} = 3m-1$ independent parameters to be set: $m-1$ weights (the last being dependent, since the weights are assumed to be normalized), m indifference and m preference thresholds. Let $\Omega_0 \subseteq \mathbb{R}^{\mathrm{d}}$ be the unconstrained compatible parameter domain, i.e., the hypervolume of all admissible preference parameters for the given preference model (assuming that all parameters can be represented by real numbers).

When eliciting preferences by an aggregation-disaggregation approach (Mousseau, 2003), the DM is asked to provide (partial) information about his preferences. Based on these, the parameters of the chosen preference model, i.e., the weights of PROMETHEE II in our case, will be approximated.

Practically, eliciting preferences is the search for the parameter set $\omega' \in \Omega_0$ that represents the decision maker's preference in the "*best possible way*". For an MCDA ranking method this means that the found set of parameters ω' should lead to the same ranking $\mathcal{R}'$ as the implicit one, denoted $\mathcal{R}^\star$(that we assume the DM has in his mind and that will serve him as reference to answer the queries). The process iteratively reduces the domain of possible parameter values Ω through the answer to holistic queries that are proposed to the DM (Figure 2). Subsequently, the number of compatible rankings $|\mathfrak{R}|$ also reduces during the process. For each iteration k:

1. ***A query*** $\mathfrak{q}_k$ ***is generated***. It is added to the set of so far generated queries $\mathfrak{Q}_k = \mathfrak{Q}_{k-1} \cup \mathfrak{q}_k$, with $\mathfrak{Q}_0 = \varnothing$. In this paper, we exclusively consider "closed" queries, that is, any query $\mathfrak{q}_k = \{\alpha_{k1} \dots \alpha_{kp}\}$, can be defined as a set of partial preference statements from which the DM has to pick one. For instance, for a pairwise comparison of actions a and b, we would have $\mathfrak{q}_k = \{\text{"}a \succ b\text{"}, \text{"}b \succ a\text{"}\}$: the DM could only select one of these two statements as answer.

2. ***The decision maker makes a statement***. Each answer to a query adds a constraint on the admissible parameter values. The precise form and examples of the constraints are given below (Section 3.1). The set of constraints at iteration w is updated: $\mathfrak{C}_k = \mathfrak{C}_{k-1} \cup \mathfrak{c}_k$, with $\mathfrak{C}_0 = \{\text{"}w_1 \geq 0\text{"}, \text{"}w_2 \geq 0\text{"}, \dots, \text{"}w_m \geq 0\text{"}, \text{"}\sum_{h=1}^{m} w_m = 1\text{"}\}$. $\mathfrak{C}_0$ contains only the basic constraints for weights.

3. ***The compatible parameter domain*** Ω_k ***is computed***. Each statement reduces the size of the compatible parameter domain and converges to the parameter values that should best represent the

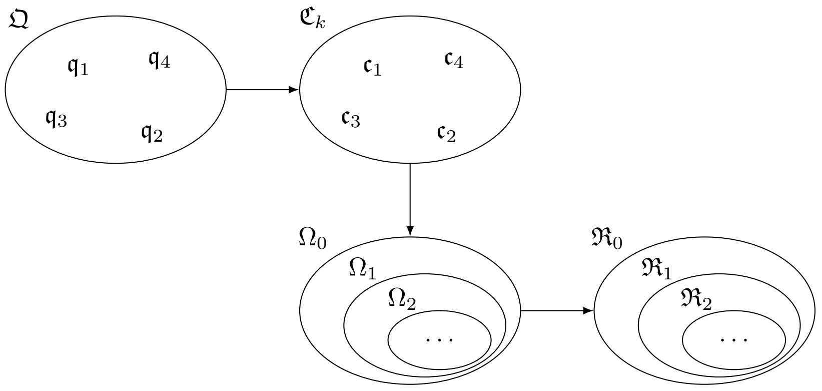
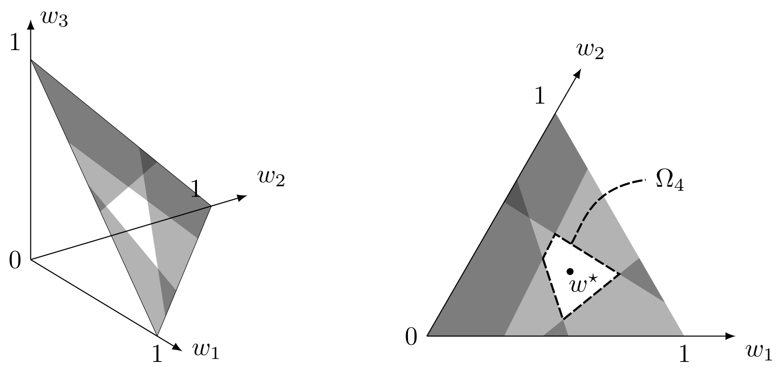


Figure 2: From a query to its consequences on the preference parameter space.



(a) Domain of possible weights for 3 criteria.

(b) Example of constraint satisfaction domain Ω_4 for 4 constraints.

Figure 3: Planar projection of a three-criteria compatible weight domain (a) onto a plane (b). The dot on the right plot represents the reference weight $\omega^\star$. The surrounding white polytope represents the domain of weight vectors Ω_4 that satisfy all constraints at iteration 4. The darker an area, the more constraints are violated inside that area.

DM's preferences. If we denote Ω_k the compatible domain after the DM has answered the k-th query, we have $\Omega_0 \supseteq \Omega_1 \ldots \supseteq \Omega_k$.

4. ***The set of compatible rankings*** $\mathfrak{R}_k$ ***is deduced***. The set of rankings $\mathfrak{R}_k$ that can be generated from all the weights inside the k-th compatible parameter domain Ω_k is finite. Depending on the instance size, this set can nevertheless be very large. During the eliciting process, the goal is to reduce this number as much as possible, striving to have only one ranking left at the end, that hopefully equals the implicit ranking we assume exists in the DM's head. As for the successive compatible weight domains, we have $\mathfrak{R}_0 \supseteq \mathfrak{R}_1 \ldots \supseteq \mathfrak{R}_k$.

Let us stress that, contrary to the approach proposed in Eppe et al. (2011), we assume that the DM answers the queries in a consistent way (no mistakes, no doubts, etc.).

In the following and for the sake of clarity, we will illustrate features of the compatible weights domain Ω in the case of $m = 3$ criteria. In particular, we show its projection of the three dimensional constrained hyperplane (Figure 3). However, the results and descriptions are, unless mentioned otherwise, not dependent of the number of criteria.

Although we only consider "closed queries", can the DM still be queried in many different ways: pairwise comparisons of actions, rankings of sub-sets of actions and/or weights, etc. It is intuitively clear that all queries do not have the same "value" to the process. Indeed, some queries could be very simple to answer (e.g. pairwise comparison of actions), but could require a lot of likewise queries to get sufficently robust values for the parameters. Other questions, on the contrary, could be very demanding for the DM, but have a strong "eliciting power". An extreme example of the latter would be to ask explicitely for the implicit ranking of the DM (the one we assume he "knows" without being able to formally express it). Two conflicting objectives in particular should be taken into account in this context: minimizing the cognitive difficulty of questions and/or the quantity of information required from the DM while maximizing the added value of each query in the eliciting process. The information given by the DM is thus characterized by both its *nature* and its *quantity* (e.g. the number of pairwise comparisons).

For the k-th query $\mathfrak{q}_k$, let us define its discriminating power Δ_k as follows:

$$\Delta_k = 1 - \frac{\max_{\alpha \in \mathfrak{q}_k} |\Omega_{k-1} \cap \Omega_\alpha|}{|\Omega_{k-1}|},$$

where Ω_{k-1} represents the domain of possible parameters of the constraint sequence $\mathfrak{C}_{k-1}$, and the numerator is the size of the biggest possible domain of compatible weights at the k-th query for the set of possible answers to query $\mathfrak{q}_k$. Ω_α is the the domain that is compatible with the sole constraint induced by answer α. In other words, by his answer to each new query (for instance, one action from a set of two or more actions, etc.), the DM implicitely adds a constraint from a set of possible constraints. Each of these constraints should reduce the size of the compatible domain. The numerator is representing the least favorable case in the sense of compatible domain reduction: it corresponds to the maximum domain size associated with each possible choice at a given stage. As for the information structure, the discriminating power Δ_k of a query $\mathfrak{q}_k$ depends on the previous queries' history $\mathfrak{Q}_k = \{\mathfrak{q}_1, \ldots, \mathfrak{q}_k\}$. It is thus practically not computable beforehand when given a concrete problem. Quite obviously, the aim of an eliciting procedure is thus to select, at each step, a query with the highest possible discriminating power.

3.1 Types of queries

For this work we consider three types of queries:

PAC Pairwise action comparison (allowing only strict preference of one action over another) (Siraj, 2011). It is one of the moste widely used and simplest ways of expressing partial preferences.
Example: Considering two actions, $a, b \in A$, we have $\mathfrak{q}_{\mathsf{PAC}} = \{$"$a \succ b$", "$b \succ a$"$\}$; the DM can answer either that he prefers a to b, or b to a.

POS Preferred action from a sub-set of actions $A' \subseteq A$. (Binshtok et al., 2009; Domshlak et al., 2006)
Example: We consider a subset of three actions $a, b, c \in A$. $\mathfrak{q}_{\mathsf{POS}} = \{$"$a \succ b \wedge a \succ c$", "$b \succ a \wedge b \succ c$", "$c \succ a \wedge c \succ b$"$\}$; the DM selects one action: a, b, or c as the most preferred action from the subset $\{a, b, c\}$.

ASR Action sub-rankings on a (sub-)set $A' \subseteq A$. As for POS, the DM is given more than 2 actions that he is asked to rank from the best to the worst.
Example: Again, we consider three actions $a, b, c \in A$, but here we look for complete rankings: $\mathfrak{q}_{\mathsf{ASR}} = \{$"$a \succ b \succ c$", "$a \succ c \succ b$", "$b \succ a \succ c$", "$b \succ c \succ a$", "$c \succ a \succ b$", "$c \succ b \succ a$"$\}$; the DM select one of these rankings.

Let us first develop the PAC type of queries. Considering the thresholds q_h and p_h as given for each criterion h, the eliciting of weight parameters by means of pairwise action comparisons yields a set of linear constraints. Without loss of generality, let us assume that the answer to the k-th query $\mathfrak{q}_k = \{$"$a_k \succ b_k$", "$b_k \succ a_k$"$\}$ is inducing the constraint $\mathfrak{c}_k$: "$a_k \succ b_k$", which is expressed as follows in terms of action net flows:

$$a_k \succ b_k \iff \Phi(a_k) > \Phi(b_k) \iff \sum_{h=1}^{m} w_h \left[\, \Phi_h(a_k) - \Phi_h(b_k) \right] \, > 0$$

And, thus finally, the compatible weight domain Ω_k after k answered queries is defined by:

$$\Omega_k : \begin{cases} 0 \leq w_h \leq 1 & ,1 \leq h \leq m-1 \\ \sum_{h=1}^{m} w_h = 1 & \\ \sum_{h=1}^{m} w_h \left[\, \Phi_h(a_l) - \Phi_h(b_l)\right] \; > 0 & ,1 \leq l \leq k \end{cases}$$

The effect of pairwise comparison constraints on the compatible parameter space are illustrated on Figure 3(b).

For the other two considered types of queries, POS and ASR, determining the compatible domain is done in a very similar way. It suffices to notice that both can be expressed as a set of PAC constraints. For instance, for action sub-sets of three actions (a, b, c), given for a k-th POS query, $\mathfrak{q}_k = \{$ "a_k is preferred", "b_k is preferred", "c_k is preferred" $\}$, which is equivalent to $\mathfrak{q}_k = \{$ "$a \succ b \wedge a \succ c$", "$b \succ a \wedge b \succ c$", "$c \succ a \wedge c \succ b$" $\}$. Each constraint can thus be transformed into a combination of PAC constraints. Although the number of PAC constraints derived from both POS and ASR are the same, the latter provides more information than POS, since it induces a complete ranking, even if only on a sub-set of actions.

Hence, there are several decisions to be taken: what type of query to generate and which actions to use for the chosen type?

3.2 Query generation

Although queries have been generated randomly in Eppe and De Smet (2012), this has to be seen as a bottom-line approach that only serves as reference to compare different eliciting procedures and query generation schemes.

As already mentioned, one of the most widely used type of query is to provide the DM with two actions and to ask him to state his preference of one of them over the other (PAC). However, the discriminating power of each consecutive pairwise comparison made by the DM during the elicitation process decreases. This is an illustration of the Principle of Increasing Irrelevance of Preference Information (Rosinger, 1991). To alleviate this effect, we will adapt to PROMETHEE II an adaptive query selection scheme that has been proposed for MAUT. Furthermore, we investigate other complementary query types that may efficiently contribute to reducing the size of the compatible weights polytope. Although strongly discriminating questions may be hard to answer for the DM, proposing, at each stage of the process, several queries (and a possible indication on their respective potential "informational gain") that he may choose to answer could improve the process.

Because of the burden that it represents for the DM, the questioning process should be optimized in order to minimize the amount of information required from the DM. We have decided to base our approach on the *Q-Eval* method (Iyengar et al., 2001), although it has been proposed in the context of MAUT. In this paragraph, we briefly introduce the *Q-Eval* method and adapt it to PROMETHEE II before proposing an extension that integrates limited versions of the different query types presented in the previous section.

Original *Q-Eval* procedure adapted to Promethee II. The main idea of Iyengar et al. (2001)'s original iterative approach is to select the next query (a pair of actions to be presented to the DM, asking to state which one he prefers) in such a way, that it splits as evenly as possible the hyper-volume of admissible weights, i.e., the weights that are compatible with all preceding queries. Practically, this means selecting the query (the pair of actions) that maximizes, at step k, the discriminating power Δ_k. After reducing the dimensionality $m' = m-1$ of the problem by integrating the fact that $\sum_{h=1}^{m} w_h = 1$ into the relations, we obtain a base set of m constraints that defines the compatible weights domain Ω_0: $w_h \geq 0$, $\forall h \in \{1, \ldots, m'\}$ and $\sum_{h=1}^{m'} w_h \leq 1$. At each iteration, we compute the center of that polytope. For the base set of constraints, at the first iteration, the center is trivially given by $\boldsymbol{w} = \{\frac{1}{m}, \ldots, \frac{1}{m}\}$. At each iteration k, the pair of actions that is presented to the DM is selected in a three-stage procedure (Figure 4):

(a) the actions with the β best corresponding net flows (based on the current analytic center w_k^{c} of the compatible weight domain Ω_{k-1}) are selected;

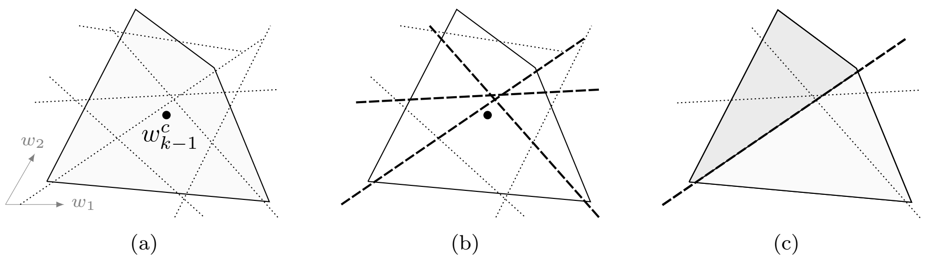


Figure 4: Schematic representation of the three major steps of *Q-Eval*'s iterative process, as proposed by Iyengar et al. (2001). The above plots represent iteration k of the process.

(b) the γ pairwise comparison constraint hyperplanes (using only the β best actions) that are closest to the analytic center are kept;

(c) the action pair whose constraint hyperplane most equally splits the current polytope into two sub-polytopes is selected as next query.

Here, β and γ are two parameters that have to be set. For our experiments, we take $\beta = 30$ and $\gamma = 10$. Indeed, as shown in the original paper, the number of actions and hyperplanes to compare may be very high. Reducing the number of considered actions to the best ones and considering only the hyperplanes that are closest to the analytical center have shown by Iyengar et al. to be an efficient "heuristic".

The reference ranking $\mathcal{R}^\star$ allows to answer the query and by doing so, impose a new constraint on the parameter space. The analytical center of the new polytope is then computed (as described in Bous et al. (2010)) and the whole process is repeated until the required number of queries is reached or there is no further discriminating pair of actions to compare.

At this stage, the only modification with respect to the original version is the hypervolume evaluation technique. To the very basic method proposed in the original paper, that mainly suits the purpose of speed, we have substituted a Monte Carlo sampling based hypervolume approximation. This choice is motivated by our goal to analyse the compatible weight domain Ω: at our level, we are more interested by precision (even limited) than in speed.

Extended *Q-Eval* to query types POS-3 and ASR-3. The original paper generates exclusively pairwise action comparison (PAC) queries. We now extend *Q-Eval* to POS and ASR query types. However, because of the computational overhead of computing such queries for bigger sub-sets, we limit ourselves to triples of actions. Let us first describe the method for POS:

(i) as before, based on the current analytic center w^c, the actions with the β best corresponding net flows are selected;

(ii) as opposed to the original procedure, we now consider all sub-sets of three actions from the set of β selected actions. Let us denote one such triple as (a, b, c). As already mentioned, $\mathfrak{q}_{\mathsf{POS}} = \{$ "a is preferred", "b is preferred", "c is preferred" $\}$ is equivalent to the set of PAC query pairs: $\{$ "$a \succ b \wedge a \succ c$", "$b \succ a \wedge b \succ c$", "$c \succ a \wedge c \succ b$" $\}$. Let us further denote $d_{\mathcal{H}_{ab}} = |\sum_{h=1}^{m} w^c_h \left[\Phi_h(a) - \Phi_h(b)\right]|$ the distance of the PAC's query hyperplane to the analytical center, and $d_{\mathcal{H}_{\max}(a,b,c)} = \max\left(d_{\mathcal{H}ab}, d_{\mathcal{H}ac}, d_{\mathcal{H}bc}\right)$ the maximum distance. We keep the γ action triples with the smallest values of $d_{\mathcal{H}\max}$;

(iii) for each of the γ best triples we compute the standard deviation of the distribution of hypervolumes for the three possible outcomes of the query: "a_k is preferred", "b_k is preferred", or "c_k is preferred". We keep the triple that minimizes that standard deviation, because this means that, on average, it is the best split of the compatible polytope, whatever the DM's choice.

This procedure can be adapted to ASR-3 without difficulty. Here again, the choice is based on the evaluation of all possible outcomes (6 of them, for ASR-3). Although the method described here could be further extended to more actions without any formal difficulty, the computation times, on the contrary, would increase strongly.

Adaptive *Q-Eval* method. As will be shown in the results section, the different query types lead to complementary behaviours in the search for preference parameters. An adaptive method has thus been tested, with the ability to select, at each step of the process, i.e., for each query, the best suited next query from PAC, POS-3, and ASR-3. The selection is based on the discriminating power of the next query: $\mathfrak{q}_k = \arg\max_{\mathfrak{q}' \in \mathcal{Q}_k} \Delta_{\mathfrak{q}'}$, where $\mathcal{Q}_k = \{\mathfrak{q}_{k,\mathsf{PAC}}, \mathfrak{q}_{k,\mathsf{POS-3}}, \mathfrak{q}_{k,\mathsf{ASR-3}}\}$.

3.3 Metrics for analysis

For our investigations, we are interested in the features of the compatible domain Ω_k at every stage k of the eliciting process, as well as the characteristics of the rankings that are induced by the compatible weights. We now introduce the metrics that we use in the following for that.

Number of compatible rankings $\boldsymbol{\rho}$. Although the domain of compatible weights is continous, only a finite number of different compatible rankings can be induced. This number gives information about the convergence of the eliciting process: the less different compatible rankings exist for a weight domain the easier it should be for a decision maker to choose one that corresponds to his preferences.

Comparison of rankings. Different metrics for comparing rankings have been proposed in the literature. In a recent paper Eppe and De Smet (2012) have used the well-known Kendall's correlation coefficient (τ_{K}) for its simplicity and its intuitive interpretation. However, this metric comes with some drawbacks (Carterette, 2009), that we would like to address in this paper.

For the present work, we use the generalized Kendall's τ_{K} correlation coefficient (Kumar and Vassilvitskii, 2010), but focus on its ability to take into account the similarity of elements' evaluations: a weight is assigned to each pair of actions (a_i, a_j), depending on their similarity. The closer the values, the smaller the impact of a possible rank inversion has on the computed value. The element similarities matrix D is defined as the mean value of the evaluation function's difference between actions a_i and a_j on both rankings. Let $\Phi^{(1)}$ and $\Phi^{(2)}$ be the net flow vectors that we want to compare with this metric: since $\Phi_i^{(1)}, \Phi_i^{(2)} \in [0,1]$, we can define the element similarity weight by $D_{ij} = \frac{1}{2}(|\Phi_i^{(1)} - \Phi_j^{(1)}| + |\Phi_i^{(2)} - \Phi_j^{(2)}|)$.

We would like to propose a generalized approach that may be applied to higher dimensions. With this concern in mind, the analytical approach to determine all possible rankings that can be generated with the weights of a compatible parameter domain is not realistic. We therefore choose to sample the domain. This is done with the *hit an run* algorithm (Lovász and Vempala, 2004) that generates a uniform distribution of samples inside a multi-dimensional convex domain defined by linear constraints.

4 Experimental set-up

Our aim is to investigate the general features of the proposed eliciting algorithm. With this goal in mind, we want to eliminate the inevitable influence of a DM's hesitations and possible inconsistencies, as well as being able to analyse results over a representative set of repeated runs. Therefore, we have chosen a simulation based approach instead of conducting experiments with real decision makers.

The detailed workflow of the experimental process is described in Algorithm 1. It shows how the different elements discribed in the previous section are articulated to produce the results presented in the next one. In particular, the algorithm uses the following functions:

`calcRanking` computes the rank of each action, based on its unicriterion net flow (that is fixed, once the threshold values are set) and a weight vector. This can be the reference weight $w_i^\star$, or any other weight (e.g. sampled compatible weight from the current iteration).

Algorithm 1: Standard experimental process

```
Input: $n$, $m$, $k$, $q$, $p$, $N_{\text{trials}}$, $N_{\text{samples}}$
for $i = 1 \ldots N_{\text{trials}}$ do
    randomly generate a set of actions $A_i$ and a reference weight vector $w_i^\star$;
    compute the unicriterion netflow matrix $\Phi_i(A_i, q, p)$;
    compute the corresponding reference ranking $R_i^\star = \texttt{calcRanking}(\Phi_i, w_i^\star)$;
    initialize set of constraints $\mathfrak{C}_i = \varnothing$;
    for $j = 1 \ldots k$ do
        $\mathfrak{q}_{ij} = \texttt{nextQuery}(\Phi_i, \mathfrak{C}_i)$;
        $\mathfrak{C}_i = \mathfrak{C}_i \cup \texttt{calcAnswer}(R_i^\star, \mathfrak{q}_{ij})$;
        sample weights $\boldsymbol{w} = \{w_1 \ldots w_{N_{\text{samples}}}\} = \texttt{genSamples}(\mathfrak{C}_i, N_{\text{samples}})$;
        $\underline{\tau}_i = 1$;
        for $s = 1 \ldots N_{\text{samples}}$ do
            compute the ranking $R_{js} = \texttt{calcRanking}(\Phi_i, w_s)$;
            $\underline{\tau}_i = \min\{\tau_{\min}, \texttt{generalizedTau}(R_{js}, R_i^\star)\}$;
```

Table 1: Parameter values used for the tests. When different values are given for one parameter, the one in bold represents the default value.

Parameter		Value
Number of actions	n	10, **20**, 50, 100, 200
Number of criteria	m	3, **5**, 7, 10
Indifference thresholds Preference thresholds	q_h p_h	$\begin{pmatrix} \mathbf{0.05} \\ \mathbf{0.10} \end{pmatrix}, \begin{pmatrix} 0.00 \\ 0.50 \end{pmatrix}, \begin{pmatrix} 0.20 \\ 1.00 \end{pmatrix}$
Number of trials for each run	n_T	50
Sampling size for quality estimation	n_S	500

`nextQuery` generates the next query according to the (extended) *Q-Eval* procedure. It uses the unicriterion net flows and the current set of constraints that results from the history of already answered queries.

`calcAnswer` uses the reference ranking $R_i^\star$ to answer the current query (which is a finite set of possible answers, the DM being asked to select his most preferred answer). The answer is returned as an additional constraint to be added to the set of previous constraints.

`genSamples` randomly generates $N_{samples}$ weights that are uniformly distributed on the compatible weight domain defined by the constraint set $\mathcal{C}_i$.

Concerning the randomly generated test data: In the present paper, we randomly generate the evaluations for each action on each criterion inside the interval [0,1], based on a uniform distribution.

5 Results

Eppe and De Smet (2012) have proposed a first approach that generates the queries in a purely random way: this gives us a "bottom line" result that we would like to compare our current results with.

Unless mentioned otherwise, all results presented in this section are based on default parameter values provided in bold in Table 1. Results for other parameter combinations of that table confirm the observations made for the default configuration and are thus not presented here explicitly. Where considered appropriate, other results have been given too, to show how the algorithm scales. As already mentioned, we use fixed values for the threshold parameters. We justify this approach by the fact that these parameters, with their relatively understandable intra-criterion meaning, could be provided by a decision maker more easily.

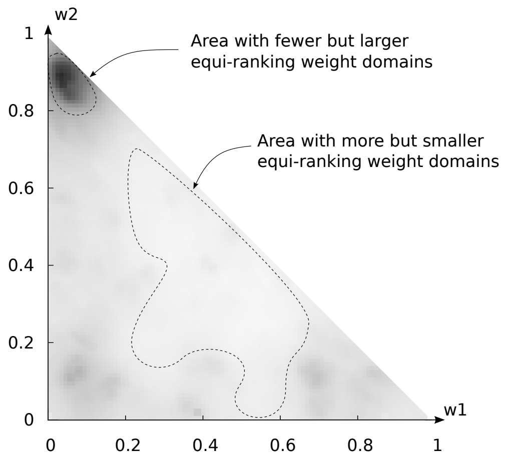


Figure 5: This map shows, for each weight, the total number of sampled weights that induce de thame ranking: the darker the color, to more weights induce the same ranking. The presented map is an example for one randomly generated instance. Each instance will be associated with another, yet relatively similar, map (Figure 6).

The presented results are structured in the following way: first, we provide some insights into the structure of the search space of the PROMETHEE II weight parameters, which validates our empirical approach for the following. We then show the limitations of query generation with *Q-Eval* (at least when applied to PROMETHEE II), before presenting some results about the quality measure we propose for the eliciting process. Finally, we give the improved results that we have obtained with an extended version of *Q-Eval*.

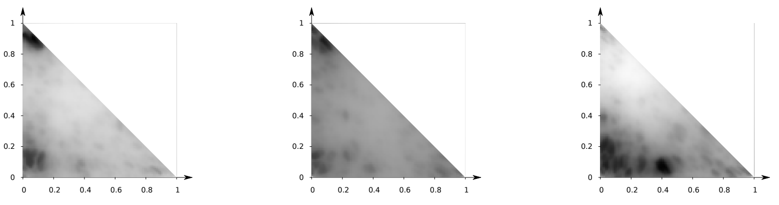
5.1 Structure of the search space

Having some knowledge about the parameter space helps in the design of well performing eliciting methods. In this section, we propose some basic empirical insights that validate our approach presented further in the text. Indeed, for any given set of actions, only a subset of all possible permutations can be reached by PROMETHEE II, whatever the weight parameters. Within the compatible domain, the number of weights that induce a given ranking, i.e., the For a given action set A, the set of actually reachable rankings is small with respect to the theoretical set of permutations ($n!$).

Furthermore, the number of possible rankings is discrete and finite, as opposed to the parameter space which is a continuous domain. This results in having, for each possible ranking, a domain $\omega_r \in \Omega$ which leads to a ranking r. Obviously, we have that $\bigcup_{r \in R} \omega_r = \Omega$, where R is the set of all possible rankings. However, the set of all possible rankings R is not accessible to investigation, as the number of rankings increases factorially. Figure 5 shows an approximation of the size of the weight-domain that induce the same ranking. It gives an idea on the degree of homegeneity of the parameter space. Practically, based on the sampling of the whole weight domain, for each sampled weight, the darkness is proportional to the number of other sampled weights that induce the same ranking. This representation is almost unique for each set of generated action, but it has been observed that, generally, the central region of the weight-domain Ω has a high density of small domains ω_r, while the peripheral areas (the "corners") have fewer but bigger domains. The main practical implication to us is that for central, more compromise-oriented areas of the weight domain, the distribution of different rankings seems not to vary strongly. This suggests that the approach based on a geometrical comparison of domain sizes to determine the next query (Section 3) makes sense, since the number of different rankings should be approximatively proportional to the domain's respective sizes.

the weights that are elicited are less robust when the DM, as could be considered as more desirable, is looking for a compromise solution, i.e., does not favor one criterion and neglect the other.

The presented results also suggest that the parameter space's structure is not too sensitive to variations of the threshold values q and p (Figure 6). Thus, choosing different threshold values should not modify these



(a) q = 0.00, p = 0.50 (b) q = 0.05, p = 0.10 (c) q = 0.20, p = 1.00

Figure 6: For one same set of randomly generated actions ($n = 20, m = 3$), these three maps show that choosing “reasonable” threshold values (plots a and b) does not have a strong impact on the homogeneity of the weight parameter space. However, for more radical yet possible threshold values (c), the parameter space has a different structure.

Table 2: Maximum number of PAC queries that could be generated on average with *Q-Eval* for 30 requested queries for 50 randomly generated instances. Note that only queries with a discriminating power $\Delta_k > 0$ are generated.

	m			
n	3	5	7	10
10	8.2	12.7	15.4	17.4
20	12.3	20.5	26.3	29.5
50	15.9	28.4	30.0	30.0
100	17.3	29.5	30.0	30.0
200	18.0	29.8	30.0	30.0

conclusions.

We consider these first observations as sufficient for the scope of this paper, but further investigations on the search space could most certainly allow to shed more light on the properties of the parameter space of the preference models.

5.2 Number of queries that can be generated with *Q-Eval*

The number of queries that can be generated on average with *Q-Eval* is strongly related to the instance size (Table 2). Depending on the willingness and/or ability of a DM to answer queries, the instance size could thus represent a ”methodological” limit. In addition, the parameters β and γ that, for speed-up reasons, respectively limit the number of considered actions and the number of hypervolume comparisons (Section 3), have also an impact on the number of queries that can be generated. For this work, as already mentioned, we have arbitrarily fixed the values as follows: $\beta = 30$ and $\gamma = 10$.

Notwithstanding the possible effects on preference eliciting processes, this limitation has an impact on our experimental study. Indeed, many results proposed hereafter are showing the evolution of the results when the number of queries increases. However, with ”stagnating” number of generated *Q-Eval* queries (e.g., Figure 9 for $n = 20$ and $m = 3$), some results may be represented erroneously. In order to prevent this phenomenon from artificially altering the results, we will cut-off results as soon as the number of actually generated constraints with *Q-Eval* is less than the number of required constraints. For instance, based on Table 2, all results for instances of $n = 20$ actions and $m = 5$ criteria will be shown only for up to 20 constraints.

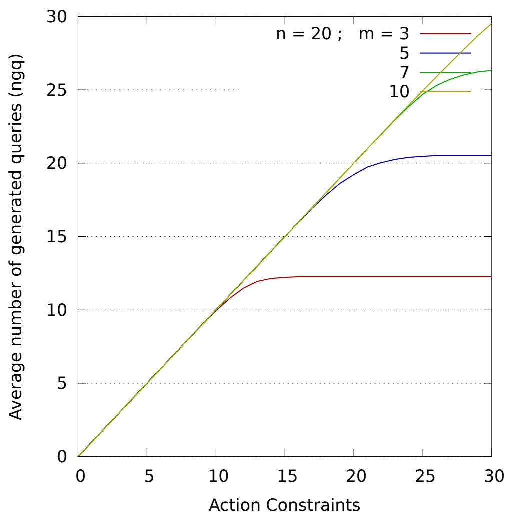
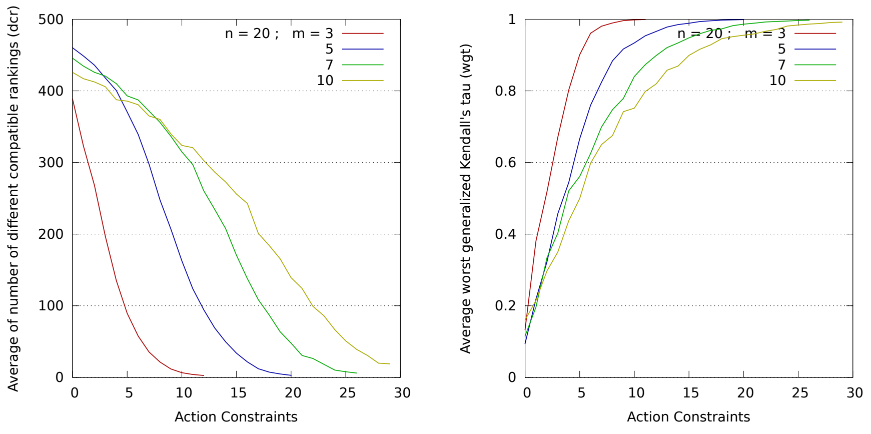


Figure 7: Evolution of the average number of constraints that can be generated with the *Q-Eval* method depending on the instance size (number m of alternatives) for a given number of alternatives. Plots for a constant value of m and different number of alternatives n are very similar.



(a) Number of different rankings ρ (b) Worst generalized Kendall's τ_{G}

Figure 8: For $n = 20$ actions and different numbers m of criteria, shows the average evolution of: *a)* the number of different rankings found in the sampling of the compatible parameter domain Ω; *b)* the worst value of the generalized Kendall's correlation coefficient τ_G.

5.3 The *Q-Eval* eliciting process for pairwise action comparisons

We are using the generalized Kendall's correlation coefficient, denoted τ_{G}, to assess the quality of a ranking $\mathcal{R}$ (induced by a set of parameters ω) with respect to a reference ranking $\mathcal{R}^{\star}$(induced by a set of reference parameters $\omega^{\star}$). As has been done in Eppe and De Smet (2012), we also take a pessimistic point of view by selecting the worst value of τ_{G} among the sampled weights that are compatible with the so far provided constraints. In addition to this first metric, we also wish to study how many different rankings could be generated that are compatible with the set of constraints. Let us note that this number of different rankings, denoted $\underline{\rho}$ in the following, depends both on the instance size and the sampling size. This means that the comparison makes the most sense between results of same instance and sampling sizes. For the sake of comparability, the results presented in detail in the following are therefore limited to mainly one type of instances. Practically, we choose to focus on instances of $n = 20$ actions.

Table 3: This table represents the average number of different rankings in the samples for each trial after the last PAC query. Note that the sample size of $n_S = 500$ has a clear impact on the result, in particular in the upper range of values. The actual number of different rankings inside the compatible domain is probably higher in those cases.

	m			
n	3	5	7	10
10	1.1	1.1	2.4	3.9
20	1.0	1.5	2.2	14.1
50	4.8	18.6	204.4	358.8
100	44.6	226.7	430.8	465.9
200	187.9	432.1	477.8	490.5

Table 4: This table represents the average value of the worst Kendall's correlation coefficient τ_G for the sampling of the compatible parameter domain Ω for each trial after the last PAC query. Note that the values are rounded to the third digit. As the values of τ_G are very high, some values are rounded to 1, although there are more than different compatible rankings as shown in Table 3.

	m			
n	3	5	7	10
10	0.999	0.998	0.988	0.988
20	1.000	1.000	0.999	0.994
50	1.000	1.000	0.998	0.987
100	1.000	1.000	0.997	0.983
200	1.000	1.000	0.998	0.982

As expected, increasing the number m of criteria gives the eliciting procedure a harder time to reach a good quality of elicited weight parameters (Figure 8, Table 3, Table 4). Comparing these results with those previously obtained by Eppe and De Smet (2012), clearly shows the better efficiency of the *Q-Eval* method.

5.4 Extended *Q-Eval* method to other query types

In this section we present the results obtained with an extension of *Q-Eval* to queries of the following forms: POS-3 and ASR-3, which are respectively POS and ASR queries for sub-sets of 3 actions.

Both of these query types may seem very similar. This is due to the purposeful limitation to three actions. As the computational overhead of selecting three actions is already significant, we have not considered the case of a higher number of actions to be compared simultenaously by the DM. Let us note that already the here presented query types are computationally more demanding than the original procedure that is limited to pairwise action comparisons. Although this aspect should be taken into account for a real applicaton, we have chosen to neglect the time required to generate the next query in this study. At the end of this section, we will combine the three types of queries (PAC, POS-3, and ASR-3) into an adaptive querying scheme.

Based on the quantity of information contained in each query type's answer, the queries can be ordered in the following way: PAC, POS-3, ASR-3, from the least to the most information containing query type. It could therefrom be expected that ASR-3 queries would be the most efficient ones, followed by POS-3, and, finally, PAC queries. The conclusion would be, that only ASR-3 queries should be proposed to the DM. However, this neglects the already mentioned limitation inherent to the *Q-Eval* procedure (Section 5.2). Thus, as could be expected, less queries are required to reach an equivalent eliciting quality with ASR-3 than with POS-3, and than with PAC (Table 5). Considering Table 6 however, it becomes clear that queries with high informational content are not able to reach the highest quality, i.e. the smallest number of different queries that are compatible with all constraints. To express this in terms of the discriminating power introduced earlier, we can say that the discrimating power of ASR-3 is higher than POS-3 and PAC

Table 5: Number of queries that has to be answered by a DM to reach the desired level of quality $\underline{\tau_G}$ for randomly generated $n = 20$ action sets on $m = 5$ criteria. The extended *Q-Eval* method significantly outperforms the bootom-line approach (Eppe and De Smet, 2012) and *Q-Eval*.

		$\underline{\tau_G}$						
		0.50	0.60	0.70	0.80	0.90	0.95	0.99
random	PAC	13	17	21	25	35	47	
Q-Eval	PAC	4	5	6	7	9	11	17
	POS-3	2	3	4	4	6	8	
	ASR-3	2	3	4	4	4	5	
	adaptive	2	2	3	3	4	5	10

Table 6: Number of pairwise action comparisons that has to be given by a DM to reduce to a desired number of different compatible rankings $\underline{\rho}$ for randomly generated $n = 20$ action sets on $m = 5$ criteria. The extended *Q-Eval* method significantly outperforms the bootom-line approach (Eppe and De Smet, 2012) and *Q-Eval*.

		$\underline{\rho}$						
		100	50	20	10	5	2	1
random	PAC	59						
Q-Eval	PAC	12	14	16	17	18	20	
	POS-3	8	10					
	ASR-3	5	6	9				
	adaptive	5	6	7	8	9	10	

for the first constraints, but then rapidly drops down to a value near to zero. It is not possible anymore to find a triple of actions whose ranking would reduce the domain of constraint compatible weights. On the contrary, PAC queries have a smaller but "longer lasting" discrimative power. POS-3's behaviour lies somewhere in between.

These results suggest that an adaptive query selection scheme could be of interest. At each step of the eliciting process, it would select the query type that has the highest discriminative power, when compared to the other. Average results for this approach (Figure 9) show that:

- Indeed, ASR-3 queries are exclusively used at the beginning of the eliciting process and tend to be progressively less selected as the process goes on;
- Contrary to what could be expected, the ASR-3 queries are predominantly replaced by PAC queries and not POS-3 queries, at least during a "transition phase" (until 10 queries);
- For a relatively small number of randomly generated instances however, POS-3 queries do represent the best possible next query.

The execution time required to compute a query depends on its type and on the instance size (Table 7). The times given refer to a non-optimized implementation in Octave and are purely informative. They do not aim at representing any benchmark, but rather to show that the times for a decision maker to wait for the next query seem reasonable for most cases in an MCDA context.

Globally, the adaptive extension significantly outperforms the original *Q-Eval* approach, allowing to turn down the information asked to the DM to be reduced to as few as 10 queries.

Table 7: Average query generation time (in seconds) for two extreme instance sizes and for each query type.

	Query type		
Instance size	PAC	POS-3	ASR-3
$n = 10, m = 3$	0.2	0.4	0.7
$n = 200, m = 10$	1.6	7.0	13.4

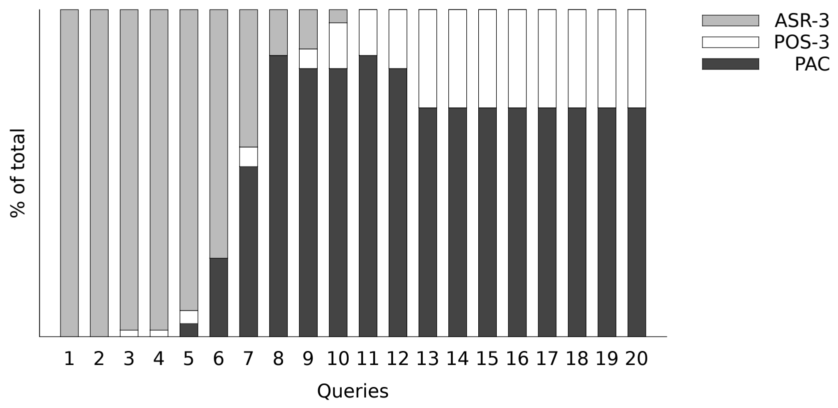


Figure 9: Evolution of the mean ratio of each *Q-Eval* query-type when applying the *adaptive query selection scheme*. Each type of query plays a role and has its usefulness in the process at different stages of the eliciting process.

6 Conclusions

In this paper we have proposed an approach to elicit Promethee II's weight parameter space. We have compared the *Q-Eval* elicting method (based on pairwise action comparisons) with the random, bottom-line approach proposed by Eppe and De Smet (2012). We have quantified the expected but nevertheless strong improvement of the results, mainly in terms of number of queries required to reach a given quality. In order to quantify different query types, we have introduced the discriminating power associated with each query. We have used it to compare the preceeding results with two other types of queries (for the use of which we have adapted *Q-Eval*): "prefered action of a sub-set of 3 actions" (POS-3) and "ranking on a sub-set of 3 actions" (ASR-3). Although these query types generally have a higher discriminating power, they also seem to be unable to reach the highest quality results. Therefore, we have finally proposed an adaptive eliciting process that picks the following query from a set of queries (one of each type) at each step. The results of this adaptive query selection scheme are qualitatively very encouraging. However, they should be studied in more depth and compared with other eliciting methods proposed in the literature. In particular, the determination of the number of queries that have to be answered in order to reach a desired quality seem to be an interesting perspective from which to compare different methods. Finally, several improvements/extensions of the method could be investigated in future works: *i)* elicit Promethee II's preference *thresholds*; *ii)* take DM's possible inconsistencies into consideration; *iii)* generate artificial actions used to build queries with high discriminating power; *iv)* find heuristics to determine β and γ parameters of *Q-Eval*.

References

Behzadian, M., Kazemzadeh, R., Albadvi, A., Aghdasi, M., 2010. PROMETHEE: A comprehensive literature review on methodologies and applications. European Journal of Operational Research 200 (1), 198–215.

Binshtok, M., Brafman, R. I., Domshlak, C., Shimony, S. E., 2009. Generic preferences over subsets of structured objects. Journal of Artificial Intelligence Research 34, 133–164.

Bous, G., Fortemps, P., Glineur, F., Pirlot, M., 2010. ACUTA: A novel method for eliciting additive value functions on the basis of holistic preference statements. European Journal of Operational Research 206 (2), 435–444.

Brans, J.-P., Mareschal, B., 2005. PROMETHEE methods. In: Figueira, J. R., Greco, S., Ehrgott, M. (Eds.), Multiple Criteria Decision Analysis, State of the Art Surveys. Springer, Ch. 5, pp. 163–195.

Braziunas, D., 2006. Computational approaches to preference elicitation. Tech. rep., Department of Computer Science, University of Toronto, Canada.

Carterette, B., 2009. On rank correlation and the distance between rankings. In: Allan, J., Aslam, J. A., Sanderson, M., Zhai, C., Zobel, J. (Eds.), Proceedings of the 32nd Annual International ACM SIGIR Conference, on Research and Development in Information Retrieval, SIGIR 2009. ACM press, New York, NY, pp. 436–443.

Dias, L. C., Mousseau, V., Figueira, J. R., Clímaco, J. N., April 2002. An aggregation/disaggregation approach to obtain robust conclusions with ELECTRE TRI. European Journal of Operational Research 138 (2), 332–348.

Domshlak, C., Brafman, R. I., Shimony, S. E., Silver, Y., 2006. Preferences over sets. In: Proceedings of the 21st Conference on Artifical Intelligence (AAAI '06).

Eppe, S., De Smet, Y., 2012. Studying the impact of information structure in the PROMETHEE II preference elicitation process: A simulation based approach. In: Greco, S., Bouchon-Meunier, B., Coletti, G., Fedrizzi, M., Matarazzo, B., Yager, R. R. (Eds.), Information Processing and Management of Uncertainty, 14th International Conference, IPMU2012. Vol. 300 of Communications in Computer and Information Science. Berlin / Heidelberg, pp. 383–392.

Eppe, S., De Smet, Y., Stützle, T., 2011. A bi-objective optimization model to eliciting decision maker's preferences for the PROMETHEE II method. In: Brafman, R. I., Roberts, F., Tsoukiàs, A. (Eds.), Algorithmic Decision Theory, Third International Conference, ADT 2011. Vol. 6992 of Lecture Notes in Computer Science. Springer, Heidelberg, Germany, pp. 56–66.

Frikha, H., Chabchoub, H., Martel, J.-M., 2010. Inferring criteria's relative importance coefficients in PROMETHEE II. International Journal of Operational Research 7 (2), 257–275.

Iyengar, V. S., Lee, J., Campbell, M., 2001. Q-Eval: Evaluating multiple attribute items using queries. In: Proceedings of 3rd ACM Conference on Electronic Commerce, EC 2001. ACM press, New York, NY, pp. 144–153.

Jacquet-Lagrèze, E., Siskos, Y., 2011. Preference disaggregation: 20 years of MCDA experience. European Journal of Operational Research 130 (2), 233–245.

Kadziński, M., Greco, S., Słowiński, R., 2012. Extreme ranking analysis in robust ordinal regression. OMEGA 40 (4), 488–501.

Kumar, R., Vassilvitskii, S., 2010. Generalized distances between rankings. In: Rappa, M., Jones, P., Freire, J., Chakrabarti, S. (Eds.), Proceedings of the 19th International Conference on World Wide Web, WWW 2010. ACM press, New York, NY.

Lovász, L., Vempala, S., 2004. Hit-and-run from a corner. In: Proceedings of the 36th ACM Symposium on Theory of Computing, STOC'04. ACM press, New York, NY, pp. 310–314.

Mousseau, V., 2003. Elicitation des préférences pour l'aide multicritère à la décision. Ph.D. thesis, Université Paris-Dauphine, Paris, France.

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

Rosinger, E. E., 1991. Beyond preference information based multiple criteria decision making. European Journal of Operational Research 53 (2), 217–222.

Siraj, S., 2011. Preference elicitation from pairwise comparisons in multi-criteria decision making. Ph.D. thesis, Faculty of Engineering and Physical Sciences, University of Manchester, United Kingdom.

Sun, Z., Han, M., 2010. Multi-criteria decision making based on PROMETHEE method. In: Proceedings of the 2010 International Conference on Computing, Control and Industrial Engineering. IEEE Computer Society Press, Los Alamitos, CA, pp. 416–418.