

Jordi Pascual Fontanilles

**Design and implementation of tools to assist the decision maker
in preference modeling with ELECTRE-H**

TREBALL DE FI DE GRAU

dirigit per: Aïda Valls Mateu i Antonio Moreno Ribas

Grau d'Enginyeria Informàtica



UNIVERSITAT ROVIRA I VIRGILI

Tarragona

2019

Resum.

Els mètodes informàtics d'ajuda a la presa de decisions multi-criteri són cada cop més utilitzats degut a que les noves tecnologies permeten recollir de forma fàcil molta informació sobre qualsevol tema, i això fa que prendre decisions sigui cada cop més complex ja que s'han d'analitzar molts factors alhora.

Aquest és un àmbit d'estudi dins del camp de la Intel·ligència Artificial que vol crear sistemes de suport a la presa de decisions que facilitin aquesta tasca. Un dels mètodes més utilitzats des de fa anys en aquest àmbit és l'ELECTRE. Ara bé, fins fa molt poc, aquest mètode no permetia que els criteris que s'utilitzen per a escollir la millor decisió, estiguin organitzats de forma jeràrquica. Per a solucionar aquest problema, en el grup de recerca ITAKA es va desenvolupar la variant ELECTRE-H. Aquest mètode afegeix tres paràmetres en els criteris intermedis de la jerarquia, els quals afegeixen dificultat a l'usuari a l'hora de crear el model del problema.

En aquest treball fi de grau, partint com a base del programa ELECTRE-H en la seva versió 2.0, s'han dissenyat i implementat noves eines per a facilitar a l'usuari l'elecció dels valors més idonis per aquests paràmetres. Aquestes eines permeten generar automàticament uns primers valors per als paràmetres escollint entre unes configuracions predeterminades. Aquests valors poden servir com a base per fer una exploració guiada de l'espai de possibles valors. Les eines també permeten visualitzar els resultats obtinguts amb diferents paràmetres de forma gràfica, per tal que l'usuari pugui identificar fàcilment quins s'ajusten més a les seves necessitats.

Les eines desenvolupades s'han provat en diferents casos d'estudi, amb resultats satisfactoris. Per això s'ha creat la versió 3.0 de ELECTRE-H que inclou aquestes noves funcionalitats i que es vol distribuir a través de la plataforma Innoget.

Resumen.

Los métodos informáticos de ayuda a la toma de decisiones multicriterio son cada vez más usados debido a que las nuevas tecnologías permiten recoger de forma fácil mucha información sobre cualquier tema, y eso hace que tomar decisiones sea cada vez más complejo ya que se deben analizar muchos factores al mismo tiempo.

Este es un ámbito de estudio dentro del campo de la Inteligencia Artificial que quiere crear sistemas de soporte a la toma de decisiones que faciliten esta tarea. Uno de los métodos más usados desde hace años en este ámbito es el ELECTRE. Hasta hace poco, este método no permitía que los criterios usados para escoger la mejor decisión, estuviesen organizados de forma jerárquica. Para solucionar este problema, en el grupo de investigación ITAKA se desarrolló la variante ELECTRE-H. Este método añade tres parámetros en los criterios intermedios de la jerarquía, los cuales añaden dificultad al usuario en el momento de crear el modelo del problema.

En este trabajo de fin de grado, partiendo como base del programa ELECTRE-H en su versión 2.0, se han diseñado e implementado nuevas herramientas para facilitar al usuario la elección de los valores más idóneos para estos parámetros. Estas herramientas permiten generar automáticamente unos primeros valores para los parámetros escogiendo entre unas configuraciones predeterminadas. Estos valores pueden servir como base para hacer una exploración guiada del espacio de posibles valores. Las herramientas también permiten visualizar los resultados obtenidos con diferentes parámetros de forma gráfica, para que el usuario pueda identificar fácilmente cuales se ajustan más a sus necesidades.

Las herramientas desarrolladas se han probado en diferentes casos de estudio, con resultados satisfactorios. Por eso se ha creado la versión 3.0 de ELECTRE-H que incluye estas nuevas funcionalidades y que se quiere distribuir a través de la plataforma Innoget.

Abstract.

Multi criteria decision aiding methods are increasingly used because new technologies allow easy gathering of information on any topic, which makes decision making more and more complex since many factors have to be analyzed at the same time.

This is a research field in Artificial Intelligence focused to create decision support systems to assist in the decision making task. One of the most used methods for years is ELECTRE. However, until recently, this method did not allow a hierarchical organization of the criteria used in the decision model. To solve this problem, in the ITAKA research group, an extension named ELECTRE-H was developed. This method adds three parameters in the intermediate criteria of the hierarchy, which add difficulty to the user when creating the problem model.

In this final degree work, starting with the ELECTRE-H software in its 2.0 version, new tools have been designed and implemented to help the user to find the most suitable values for these parameters. These tools automatically generate a first set of values for the parameters and the user can choose them from some default linguistic settings. These values can serve as a basis for a guided exploration of the space of possible values. The tools also allow to visualize graphically the results obtained using different parameters, so that the user can easily identify which ones best suit his needs.

The tools developed have been tested in different case studies, with satisfactory results. This is why the version 3.0 of ELECTRE-H, which includes these new features, has been created and will be distributed through the Innoget platform.

Agraïments

En primer lloc, voldria agrair als meus tutors, Aïda Valls i Antonio Moreno, primer de tot per donar-me la oportunitat de treballar en aquest projecte. També, per tot el seu temps i dedicació, els coneixement que m'han transmès i els seus consells, orientació i suport.

També vull donar les gràcies a tots els companys del grup de recerca ITAKA, que en tot moment m'han animat, i m'han ajudat sempre que ho he necessitat.

Finalment, he d'agrar sobretot a la meva família, per la seva ajuda i suport incondicional, sense el qual no hagués pogut realitzar aquest projecte ni aquests estudis.

Table of contents

1	INTRODUCTION	5
1.1	OBJECTIVES.....	6
1.2	STRUCTURE OF THE DOCUMENT.....	6
2	BACKGROUND	7
2.1	BASIC DEFINITIONS	7
2.1.1	<i>Types of decision problems</i>	<i>8</i>
2.1.2	<i>Types of outranking relations.....</i>	<i>8</i>
2.2	THE CLASSIC ELECTRE	8
2.2.1	<i>Parameters of criteria in ELECTRE</i>	<i>9</i>
2.2.2	<i>Classical ELECTRE methodology.....</i>	<i>9</i>
2.2.3	<i>Construction of the outranking relation.....</i>	<i>10</i>
2.2.4	<i>Exploitation types.....</i>	<i>12</i>
2.3	THE ELECTRE-H METHOD.....	14
2.4	ELECTRE-III-H METHOD.....	16
2.5	ELECTRE-H SOFTWARE PACKAGE	18
2.6	PREFERENCE LEARNING	22
3	SPECIFICATIONS	24
4	DESIGN AND METHODOLOGY	25
4.1	DISCRIMINATION POLICIES.....	26
4.2	COMPARE PREORDERS.....	30
4.3	EXPLORATORY ANALYSIS.....	31
5	DEVELOPMENT	35
5.1	DISCRIMINATION POLICIES.....	37
5.2	COMPARE PREORDERS.....	40
5.3	EXPLORATORY ANALYSIS.....	41
6	EVALUATION	45
6.1	THE DATASET	45
6.2	THE INITIAL MODEL.....	48
6.3	TEST OF THE DISCRIMINATION POLICIES	48
6.4	TEST OF THE EXPLORATORY ANALYSIS.....	51
6.5	TEST OF COMPARING PREORDERS	57
6.6	ANALYSIS OF ONE CASE: GEORGIA	60
7	CONCLUSIONS AND FUTURE WORK.....	69
7.1	CONCLUSIONS	69
7.2	FUTURE WORK.....	70
8	REFERENCES.....	71

List of tables

TABLE 1: PREFERENCE RELATIONS MATRIX IN O_I PREORDER AND RANK ORDER VALUE	16
TABLE 2: SAMPLE RANKING PERFORMANCE MATRIX	25
TABLE 3: DISCRIMINATION POLICIES THRESHOLDS.....	38
TABLE 4: EPI CRITERIA	46
TABLE 5: <i>RANKINGEPI2018 USERPOLICY</i>	50

List of figures

FIGURE 1: ELECTRE PHASES	10
FIGURE 2: EXAMPLE OF O PARTIAL PREORDER.....	14
FIGURE 3: CRITERION STRUCTURE WITHOUT HIERARCHY	15
FIGURE 4: CRITERION STRUCTURE WITH HIERARCHY	15
FIGURE 5: EXAMPLE OF O_I PARTIAL PREORDER.....	16
FIGURE 6: ELECTRE-H v 2.0 INITIAL SCREEN	18
FIGURE 7: ELECTRE-H MAIN WINDOW	18
FIGURE 8: ELECTRE-H READER PANEL.....	19
FIGURE 9: GENERAL OVERVIEW LOADED FROM AN EXCEL PROBLEM FILE.....	19
FIGURE 10: ELECTRE-H DISTILLATION RESULTS	20
FIGURE 11: VISUALIZATION OF GROUPS WITH THEIR DEFINITIONS	20
FIGURE 12: VISUALIZATION OF CRITERIA DESCRIPTION	21
FIGURE 13: VISUALIZATION OF THE PERFORMANCE MATRIX.....	21
FIGURE 14: SAVE PREFERENCE RELATIONS OPTION	21
FIGURE 15: PREORDER DISTANCE.....	22
FIGURE 16: STATISTICS PANEL	22
FIGURE 17: SAMPLE RANKING CRITERIA HIERARCHY	25
FIGURE 18: ELECTRE-H v 2.0 (LEFT) AND v 3.0 (RIGHT)	26
FIGURE 19: SAVE THE CURRENT HIERARCHY AS A NEW POLICY.....	27
FIGURE 20: SAMPLE RANKING POLICIES SLIDER	27
FIGURE 21: CUSTOMIZING INTERMEDIATE CRITERION THRESHOLDS USING DISCRIMINATION POLICIES	28
FIGURE 22: STOP THE SOLVING PROCESS.....	29
FIGURE 23: INTERMEDIATE RESULT PANEL	30
FIGURE 24: PREORDERS COMPARE PANEL	31
FIGURE 25: EXPLORATORY ANALYSIS PANEL	32
FIGURE 26: ACCEPTABILITY INDEX CHART	33
FIGURE 27: RELATION INDEX CHART	34
FIGURE 28: ELECTREH.BASE PACKAGES	35
FIGURE 29: ELECTRE.CALCULATION PACKAGES.....	36
FIGURE 30: ELECTREH.GUI PACKAGES.....	36
FIGURE 31: ELECTRE-H HELP PACKAGES.....	37
FIGURE 32: POLICIES CLASS DIAGRAM	38
FIGURE 33: SOLVING SEQUENCE DIAGRAM	40
FIGURE 34: COMPARE PREORDERS SEQUENCE DIAGRAM	41
FIGURE 35: AUTOMATED DISTILLATION SEQUENCE DIAGRAM.....	42
FIGURE 36: RELATION INDEX SEQUENCE DIAGRAM.....	43
FIGURE 37: ACCEPTABILITY INDEX SEQUENCE DIAGRAM.....	44
FIGURE 38: EPI CRITERIA HIERARCHY.....	47
FIGURE 39: <i>RANKINGEPI2018</i> GROUPS AND WEIGHTS	48
FIGURE 40: <i>RANKINGEPI2018</i> HIERARCHY	49
FIGURE 41: <i>RANKINGEPI2018</i> CRITERIA	49
FIGURE 42: <i>RANKINGEPI2018</i> PERFORMANCE MATRIX.....	49
FIGURE 43: <i>RANKINGEPI2018</i> THRESHOLDS CHANGE USING DISCRIMINATION POLICIES.....	50
FIGURE 44: <i>RANKINGEPI2018</i> WITH <i>USERPOLICY</i> POLICY.....	51
FIGURE 45: <i>RANKINGEPI2018</i> EXPLORATORY ANALYSIS	52
FIGURE 46: <i>RANKINGEPI2018</i> ACCEPTABILITY INDEX.....	53
FIGURE 47: <i>RANKINGEPI2018</i> AUSTRALIA RELATION INDEX.....	54
FIGURE 48: <i>RANKINGEPI2018</i> IRELAND RELATION INDEX	55
FIGURE 49: <i>RANKINGEPI2018</i> KENYA RELATION INDEX.....	56
FIGURE 50: <i>RANKINGEPI2018</i> ACCEPTABILITY (LEFT) AND RELATION (RIGHT) INDEXES	57
FIGURE 51: COMPARE <i>EPIRANKING2018 VERY STRICT</i> AND <i>VERY TOLERANT</i> PREORDERS.....	58
FIGURE 52: <i>EPIRANKING2018 VERY STRICT</i> AND <i>VERY TOLERANT</i> BUILT PREORDERS.....	59
FIGURE 53: COMPARE <i>EPIRANKING2018 VERY STRICT</i> AND <i>VERY TOLERANT</i> PREORDERS.....	60
FIGURE 54: <i>ECOSYSTEMVITALITY</i> PREORDERS COMPARISON	61
FIGURE 55: <i>ECOSYSTEMVITALITY VERY STRICT</i> AND <i>VERY TOLERANT</i> PREORDERS.....	62
FIGURE 56: <i>ENVIRONMENTALHEALTH</i> PREORDERS COMPARISON	62
FIGURE 57: <i>ENVIRONMENTALHEALTH VERY STRICT</i> AND <i>VERY TOLERANT</i> PREORDERS.....	63

FIGURE 58: <i>GOAL VERY STRICT</i> AND <i>VERY TOLERANT</i> PREORDERS.....	63
FIGURE 59: GEORGIA RELATION INDEX IN <i>ECOSYSTEMVITALITY</i> COMPARED TO IRAN	64
FIGURE 60: GEORGIA RELATION INDEX IN <i>ENVIRONMENTALHEALTH</i> COMPARED TO IRAN	65
FIGURE 61: GEORGIA RELATION INDEX IN <i>GOAL</i> COMPARED TO IRAN	65
FIGURE 62: GEORGIA RELATION INDEX IN <i>ECOSYSTEMVITALITY</i> COMPARED TO CHILE	66
FIGURE 63: GEORGIA RELATION INDEX IN <i>ENVIRONMENTALHEALTH</i> COMPARED TO CHILE	67
FIGURE 64: GEORGIA RELATION INDEX IN <i>GOAL</i> COMPARED TO CHILE	67
FIGURE 65: GEORGIA ACCEPTABILITY INDEX	68

1 Introduction

Decision-making consists on selecting the best option among a set of multiple possible alternatives. It is a natural process in rational beings that takes place in many situations frequently. Decision can be classified in two main groups: programmed and unprogrammed decisions. The first one refers to decisions that are repetitive or routinary, and the second one to decisions that are new or unusual.

For years, scientific methodologies for decision-making have been studied, but thanks to the technological advancement, it is acquiring more importance. Complex decisions can be found in multiple fields, such as engineering, science, business, environmental pollution, etc. In those problems, we usually have many different performance criteria that must be taken into account together to arrive to the best solution. To analyze and solve these decision problems, a multidisciplinary approach is needed.

Multiple criteria decision aiding (MCDA) are methods of decision-making that help the decision maker in problems with multiple criterion that need to be optimized [7]. The decision is made from the evaluation of the performance of a set of alternatives taking into account the preferences of the decision-maker on each of the criteria. These methods can be applied to programmed and unprogrammed problems, but their common use is in unprogrammed problems, which need the construction of a model of the decision problem adhoc.

As recognized in the literature [7], there are three main approaches to solve decision-making problems:

1. Methods based on the utility theory: they are based on the construction of a utility function for each criterion, the scoring of the alternatives on basis of the utility functions and the aggregation of the scores to find an overall utility degree for each alternative. This approach is known as Functional Model.
2. Methods based on outranking: they are based on comparing the performance of pairs of alternatives with respect to one single criterion at a time. A binary preference relation is built for each criterion and then they are merged do find the overall preference structure among all the alternatives. This is called the Relational Model.
3. Methods based on rules: they obtain a set of decision rules that model the reasoning process of the decision maker. These rules are applied to each alternative so solve the problem. It is known as Logic Model.

In this final degree project, we will focus on the Relational model. We will study one of the most important methods based on outranking: the ELECTRE method. It stands for ELimination Et Choix Traduisant la REalité (Elimination and Choice Translating Reality). It is a method proposed by Bernard Roy in the mid-1960s. Some extensions to this method exist and are still in development. The ITAKA research group at URV has developed the ELECTRE-H version of this method, which is able to deal with criteria organized into a hierarchy (from elementary specific criteria to more general criteria) [2].

The ELECTRE-H software package can be obtained at:

<https://www.innoget.com/technology-offers/4008/electre-h-software-package-a-tool-for-data-analysis-and-decision-aiding-with-hierarchical-criteria>.

The software is constantly being improved, and the goal of this project is the study and implementation of techniques for preference modelling, so we can help the user choose the values for the multiple parameters he has to introduce in order to solve a MCDA problem.

1.1 Objectives

The aim of this final degree project is to add methods that can help the user choose appropriate values for the different parameters ELECTRE methods need. This main goal is divided in the following sub-objectives:

1. Study the ELECTRE methodology, in particular ELECTRE-III-H method for hierarchical data.
2. Allow stopping the ELECTRE-III-H method during the exploitation process in intermediate nodes of the hierarchy to study partial results.
3. Add default policies so the user has some already pre-defined parameters to ease the decision-making.
4. Add a method to visually compare two relation matrix files for the same problem.
5. Allow saving and loading new user defined policies.
6. Add a new toolkit to the software that permits to make an exploratory analysis of several configurations of parameters at the same time providing tools to visualize and compare the results obtained with the different policies.

1.2 Structure of the document

This document is organized as follows. Chapter 2 introduces the ELECTRE method, as well as its extension ELECTRE-H and its implementation as a software package. Chapter 3 specifies the new changes that should be added to the ELECTRE-H software package. Chapter 4 shows the design of the new tools specified in Chapter 3. Chapter 5 explains the implementation of the tools designed in Chapter 4 in the ELECTRE-H software package. Chapter 6 evaluates and tests the design and implementation of the new tools. Finally, Chapter 7 gives a conclusion about the work done, and proposes some work that could be done in the future regarding this work.

2 Background

In this chapter, I explain the elements that are used in the ELECTRE method. Next, the classic ELECTRE method is introduced first, and then ELECTRE-H (for hierarchical data) is briefly explained. Finally, the software implementing ELECTRE-H that was given to me is presented, as it has been the source of this project.

2.1 Basic definitions

In MCDA there are four essential elements, which are:

Alternatives: finite set of options A . The objective is finding the best one for a determined decision-maker. They can be anything, so the decision-making process can be used in different fields.

Criteria: finite set of performance measurable attributes G . They are used to evaluating and compare alternatives from a well-defined point of view. Alternative evaluation is done according to a criterion. It has to represent the user preference for an alternative.

Decision Maker: is the person or group of persons that perform the decision-making. Depending on the decision maker, the final choice of the decision-making process might be different.

A data matrix is constructed with the values that alternatives take in each criterion, $A \times G$. Values are better or worse depending on the criterion, not always a higher value means being better. A direction is defined for each criterion to know if we want to maximize or minimize the criterion value. If we want to maximize, it will be *Gain*, whereas if we want to minimize it will be *Cost*. For example, we would want to maximize a product quality while keeping its price low.

Moreover, not all criterion should influence the results in the same measure. In real life, each decision maker may give more importance to some criterion. In order to make the decision-making more real, criterion can have a weight that gives them more or less importance. In MCDA methods based on outranking, like ELECTRE, the weights indicate the voting power of each criterion with respect to the others.

2.1.1 Types of decision problems

In all problem types, you have a set of alternatives to analyze but the ultimate goal of the decision may be different. There are 3 types of decision problematics that have different goals. They can be described as follows:

- Choice problematic: the objective is to get the best alternative. It relies on getting a small set of alternatives so the best one can be chosen. Not always a single alternative will be selected as optimum, maybe you want only to eliminate most of the alternatives.
- Ranking problematic: you want to rank the alternatives from the best one, to the worse one in a partial or complete ordering. Some of the alternatives may not be comparable.
- Sorting problematic: sort the alternatives in predefined categories, whose preference order may be defined.

2.1.2 Types of outranking relations

The goal of these methods is to build a binary outranking relation S on a set of alternatives $A = \{a, b, c, d, \dots\}$. The meaning of aSb is that “ a is at least as good as b ”. The relation S is satisfied if there are enough arguments to decide alternative a is at least as good as alternative b , while there is no obvious reason to deny the assertion aSb [7].

To find which relations exist when comparing pairs of alternatives, the information of the data matrix is used. Three situations appear when comparing the alternatives with respect to many different criteria:

- **Indifference (I):** Occurs when evident reasons exist to demonstrate there is a total equivalence between two alternatives. Is a reflexive and symmetric binary relation.

Example: $aSb \& bSa \rightarrow aIb \rightarrow a$ is indifferent to b and vice versa.

- **Preference (P):** Occurs when evident and favorable reasons exist towards one of the alternatives. Is a non-reflexive and asymmetrical binary relation.

Example: $aSb \& \neg(bSa) \rightarrow aPb \rightarrow a$ is clearly preferred to b .

- **Incomparability (R):** Occurs when no reasons exist to justify any of the relations above. Is a non-reflexive and symmetrical binary relation.

Example: $\neg(aSb) \& \neg(bSa) \rightarrow aRb \rightarrow a$ is incomparable to b .

2.2 The classic ELECTRE

There are multiple ELECTRE methods, starting with the basic ELECTRE-I, ELECTRE-Is for selecting the best alternatives, ELECTRE-II and ELECTRE-III/IV for ranking problems, and ELECTRE-TRI for sorting problems [7].

These methods have some strengths and weaknesses [2]. The strengths are the following ones:

- ELECTRE methods can take into account the qualitative nature of some criterion. Original data can be used without having to transform it into other defined numerical scales.
- ELECTRE methods work with heterogeneous criterion scales. Many decision-making methods require a homogeneous scale, so normalizing criterion scales is needed when dealing with heterogeneous criterion.
- If in a determined criterion, an alternative is strongly opposed to the *aSb* assertion, you can reject it. This is called right to veto.

The weaknesses are the following ones:

- ELECTRE methods can not calculate an overall score for each alternative.
- Classical ELECTRE assumes all criterion are at the same level, without the possibility of organizing them in a hierarchical structure.

2.2.1 Parameters of criteria in ELECTRE

In outranking methods, the values of an alternative for each criterion $g_j(a_x)$ are not useful by themselves, they are always compared to the values of other alternatives. In this comparison procedure, three parameters are used. The two first ones are related to the discrimination power of a criterion (we can be more strict or flexible when comparing to values, this is related to uncertainty issues). The third one is used to avoid compensations with other criteria. The three thresholds are:

- q_j : **Indifference threshold** for criterion g_j : this value marks the point where an alternative is strictly preferred to another.
- p_j : **Preference threshold** for criterion g_j : this value defines an interval to decide if the preference between two values is significant or not.
- v_j : **Veto threshold** for criterion g_j : this value sets the maximum negative difference we can allow before vetoing.

The following relation must always be fulfilled:

$$0 \leq q_j \leq p_j \leq v_j$$

Depending on the values of the two first thresholds, we can distinguish two types of criteria:

- **True-criterion** when $p = q = 0$.
- **Pseudo-criterion** when $p > q > 0$.

2.2.2 Classical ELECTRE methodology

Classical ELECTRE methodology is divided in two phases. The first one finds all the relations and constructs the outranking matrix. This matrix contains the results of comparing all the pairs of alternatives. The second one does the decision-making using the matrix build in the first phase.

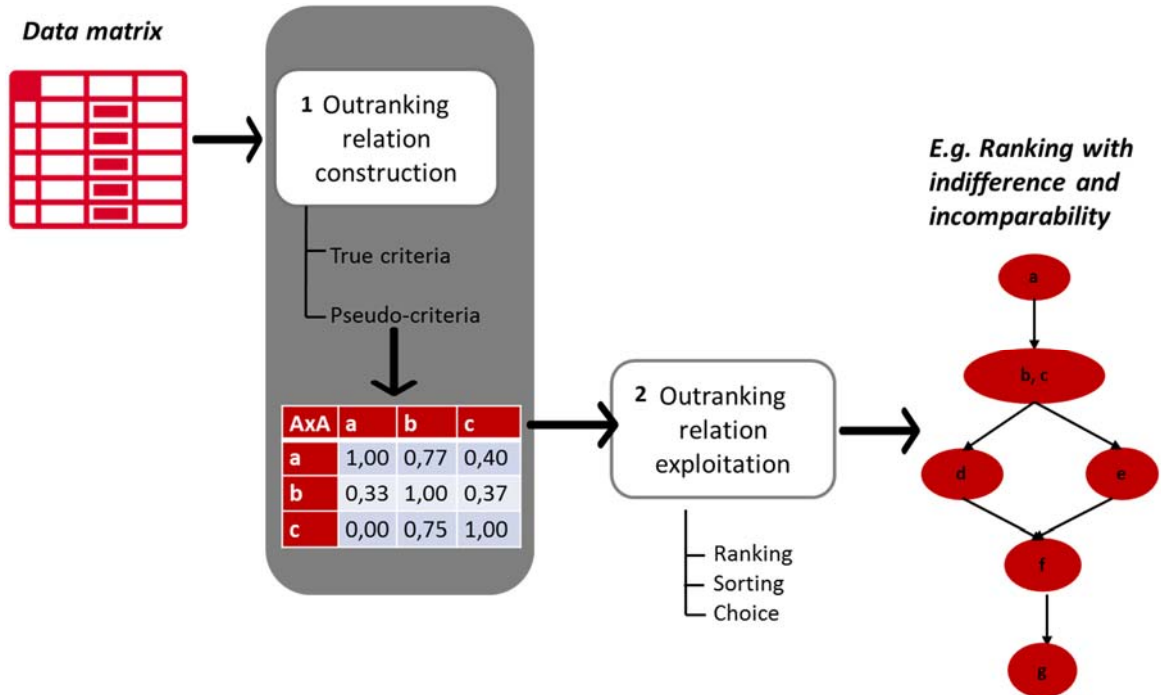


Figure 1: ELECTRE phases

To find if an outranking relation aSb exists between a pair of alternatives (a,b) , ELECTRE has two measures:

- **Concordance:** if most criterion are in favor of the assertion aSb , “ a is at least as good as b ”.
- **Discordance:** checks if any criterion can deny the assertion aSb . It is used to introduce veto right to minorities.

Using these elements, a credibility index is calculated. It references the outranking value in the binary relation between a pair (a,b) of alternatives. It defines the credibility of the aSb assertion. To calculate the credibility index, we need a previous calculation of a concordance and discordance index matrix for each criterion.

- **Concordance matrix:** each pair of alternatives has to be evaluated for each criterion. To do so, we use the preference (p_j) and indifference (q_j) indexes, obtaining partial concordance matrixes. Using these matrixes, we can calculate the concordance matrix.
- **Discordance matrix:** it gives right to veto to the criterion adding a veto threshold (v_j) to the criterion (g_j). For instance, in b to a outranking, if the b value overcomes the a value by a greater value than the veto for that criterion, this outranking relation is vetoed.

2.2.3 Construction of the outranking relation

In the ELECTRE methodology, we have four methods to construct the outranking matrix between all the pairs of alternatives a problem has. They are explained below.

2.2.3.1 Simple majority

This method check if a simple majority is in favor of aSb assertion. For each criterion in favor of the assertion, 1 point is given. When comparing we need to have in mind which is the desired value. The smaller one if the direction of the criterion is Cost or the biggest one if it is Gain.

Ties can happen, and in this case, a point is also given. As *Gain* criterion can be transformed to *Cost* multiplying by negative one, the following formula are reduced to the *Gain* case.

Partial concordance index for the j -th criterion is calculated as follows:

$$c_j = \begin{cases} 1 & g_j(a) \geq g_j(b) \\ 0 & g_j(a) < g_j(b) \end{cases} \quad (2.1)$$

Where $g_j(a)$ is the value of the criterion j for the alternative a .

To calculate the global concordance, all partial concordances are summed up:

$$c(a, b) = \sum_{j=1}^n c_j(a, b) \quad (2.2)$$

Simple majority is surpassed if the final sum overpasses the minimum value set by the user. Then, aSb outranking relation exists and has a value of 1. If the majority is not surpassed, it will have a value of 0.

This process has to be done by each pair of alternatives, getting a rating for each alternative.

2.2.3.2 Simple majority with veto

It has the same basis as Simple majority. The main difference is the right to veto the criterion have. This means that instead of just checking if an alternative surpasses other alternatives, if it has bad results in an important criterion, it will lose many points instead of not being rated. Veto is introduced in the discordance between alternatives in a criterion. Partial concordances will be calculated like in the equation (2.1), and global concordance like (2.2).

We will have enough reasons to refuse an alternative when it overpasses the alternative is being compared to by a value greater than the criterion veto. To know this, the following function is used:

$$d(a, b) = \begin{cases} 1 & \text{if } g_j(b) - g_j(a) > v_j(g_j) \\ 0 & \text{otherwise} \end{cases} \quad (2.3)$$

This way we can get the discordance between two alternatives. Combining concordance and discordance indexes, the outranking relation will be defined as follows:

$$aSb \text{ if } \begin{cases} c(a, b) \geq \hat{c} \\ d(a, b) \leq \hat{d} \end{cases}, \quad (2.4)$$

We can define the concordance threshold as a high value, and the discordance one as a low value.

2.2.3.3 Valued with weights

In this method, we need each criterion to have a weight defining its importance. We just need to calculate the concordance index. The right to veto is not considered. To do the global concordance calculation, we will use:

$$c(a, b) = \frac{1}{W} \sum_{j=1}^n w_j c_j(a, b) \quad (2.5)$$

We can express the W weight as:

$$W = \sum_{j=1}^n w_j \quad (2.6)$$

And the partial concordance $c_j(a, b)$ between a and b as:

$$c_j(a, b) = \begin{cases} 1 & \text{if } g_j(a) + q_j(g_j(a)) \geq g_j(b), \\ 0 & \text{if } g_j(a) + p_j(g_j(a)) \leq g_j(b), \\ \frac{p_j(g_j(a)) - (g_j(a) - g_j(b))}{p_j - q_j} & \text{if } g_j(b) - p_j \leq g_j(a) \leq g_j(b) - q_j \end{cases} \quad (2.7)$$

2.2.3.4 Valued with weights and veto

This method adds the right to veto to the previous method. The calculation of concordance indexes is done the same way (2.5 and 2.6). To add the right to veto, we need a discordance index, which in this method will be calculated as follows:

$$d_j(a, b) = \begin{cases} 1 & \text{if } g_j(b) > g_j(a) + v_j(g_j(a)) \\ 0 & \text{if } g_j(b) \leq g_j(a) + p_j(g_j(a)) \\ \frac{g_j(b) - g_j(a) - p_j(g_j(a))}{v_j(g_j(a)) - p_j(g_j(a))} & \text{otherwise} \end{cases} \quad (2.8)$$

Once the indexes are calculated, we can get a non-binary outranking relation with a credibility value $\rho(aSb)$ defined as:

$$\rho(aSb) = \begin{cases} c(a, b) & \text{if } d_j(a, b) \leq c(a, b), \forall j \\ c(a, b) \cdot \prod_{j \in J(a, b)} \frac{1 - d_j(a, b)}{1 - c(a, b)} & \end{cases} \quad (2.9)$$

$J(a, b)$ is the criterion set with a partial concordance value higher than partial discordance. This outranking method has the benefit of being less sensitive to threshold values, because weights have more importance.

2.2.4 Exploitation types

Exploitation is the second phase of the ELECTRE methodology. There are different techniques to be used depending to the type of problem we find:

- Choice problems: ELECTRE with Kernel calculation.
- Ranking problems: we can use Net Flow Score or Distillation.
- Sorting problems: ELECTRE-TRI based on profiles and categories.

In this work, we have focused only in Ranking problems, therefore, the two methods for ranking exploitation are explained below.

2.2.4.1 Net Flow Score

In this method, flows are used to assign a final “preference” score to each alternative. To do so, an entrance and exit flow score is calculated for each alternative.

In a given set of alternatives A evaluated in a set of criteria G the credibility value $\rho(aSb)$ is calculated for each pair of alternatives as explained in 2.2.3, having a value in the $(0,1)$ range. Once we have the outranking relations stored in a matrix $A \times A$, we only take into account the outranking relations above a minimum credibility threshold λ . All relations not surpassing this threshold will be set to 0.

We can represent this matrix as a directed graph. Alternatives are represented as nodes, and outranking relations as arcs (when value above 0).

If an arc goes from a to b , then we can say aSb . In this case, to represent arcs that go from node a_i to other nodes, we use the row represented by the outranking relation of a_{ij} to other alternatives in the credibility matrix. Arcs that enter a_i from other nodes, are represented by the column of the credibility matrix that corresponds to a_{ij} .

To rank alternatives, we calculate the Net Flow Score that corresponds to the balance of arcs entering and leaving each node. To compute this balance for a certain alternative a_i , we use the following formula:

$$NFS(a_i) = \sum_{j=1}^m a_{ij} - \sum_{j=1}^m a_{ji} \quad (2.10)$$

The value obtained for each alternative is used to rank them. The higher value will get the first position, and the lower one will get the last position. If a tie occurs, we can apply the same formula (2.10) with a lower credibility threshold.

Selecting an appropriate value for the credibility threshold is not straightforward and depends on each problem. To avoid this selection, another ranking procedure was proposed: Distillation.

2.2.4.2 Distillation

This method is also known as ELECTRE-III. It is an iterative algorithm that in each iteration selects a subset of alternatives taking into account outranking relations and calculates a NFS with different credibility threshold.

The first step of the process is known as first distillate (D_1). It chooses a subset with the higher scored alternatives. To do so, we need an initial threshold and a score for each alternative $Q(a)$ (i.e. the number of alternatives outranked by a minus the ones which outrank a).

If D_1 only has one alternative, the process is repeated in $A \setminus D_1$. Otherwise, the same process is performed in the D_1 set to obtain the second distillate D_2 . If D_2 is a singleton and is not empty, the process is repeated in $D_1 \setminus D_2$. In other case, it is repeatedly applied in D_2 until D_1 is empty. Then we can begin with $A \setminus D_1$. If two or more alternatives belong to the same distillation because they have the same score, they are indifferent. None of them can be ranked better or worse than the others, and they have the same ranking position.

This process is called descending distillation chain, and produces a first preorder O_1 .

To obtain a second preorder O_2 , ascending distillation chain is used. In this process, alternatives with the lower score are first retained. Intersecting these two preorders O_1 and O_2 , we can obtain a final ranking O . This final ranking is a partial preorder, for any pair of alternatives from set A , one may be preferred, indifferent or incomparable to the other. Incomparability can happen in a pair of alternatives when one, for example a , has a better score than another, for example b , in O_1 preorder, but b has a better score than a in the O_2 preorder.

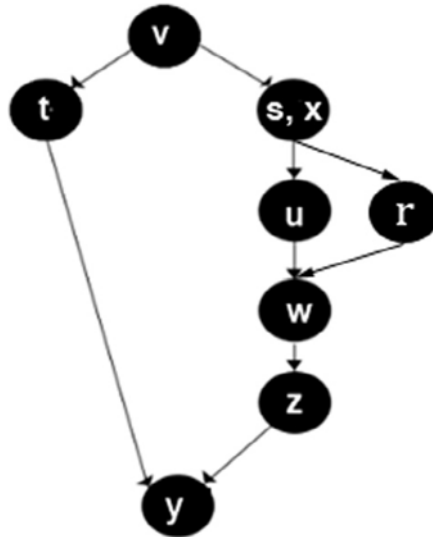


Figure 2: Example of O partial preorder

In this figure we can see, for example, that v is preferred to t , that s and x are indifferent, but that u and r are incomparable. The alternative v is the best and y is the worst.

2.3 The ELECTRE-H method

To address one of the weaknesses of ELECTRE methods, which is not being able to organize criterion hierarchically, ITAKA research group has developed an extended version of this method, with collaboration of some researchers from Poznan University of Technology. This idea arise after seeing the need of organizing criterion in several layers in real life problems. For example, in the analysis of environmental data rank options according to different risk/benefit levels [4][8] or the assessment of Websites using multiple different dimensions [3].

The new methods for hierarchical criteria are called ELECTRE-H [2]. In this hierarchical approach, the user can arrange the criteria in groups representing sub-problems, each of them corresponds to a subset of criteria. Each sub-program is modelled as an intermediate node in a hierarchy of criteria sharing the same root. The ELECTRE method can be then applied to different levels of generality, from the most specific intermediate nodes to the most general ones (being the root the final overall goal). This way, new low-level objectives are solved first and the ranking results are then propagated upwards in the hierarchy until reaching the root. A root criterion must always exist, which is known as Goal.

This would be an example of criterion arranged without a hierarchy:

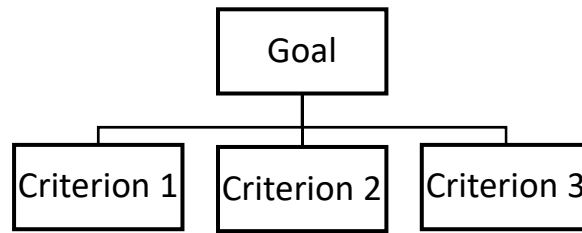


Figure 3: Criterion structure without hierarchy

When adding hierarchy to ELECTRE methods, criterion arrangement could be:

As we can see in Figure 3, the hierarchy causes a determined order to follow when solving a problem, in order to obtain the final result.

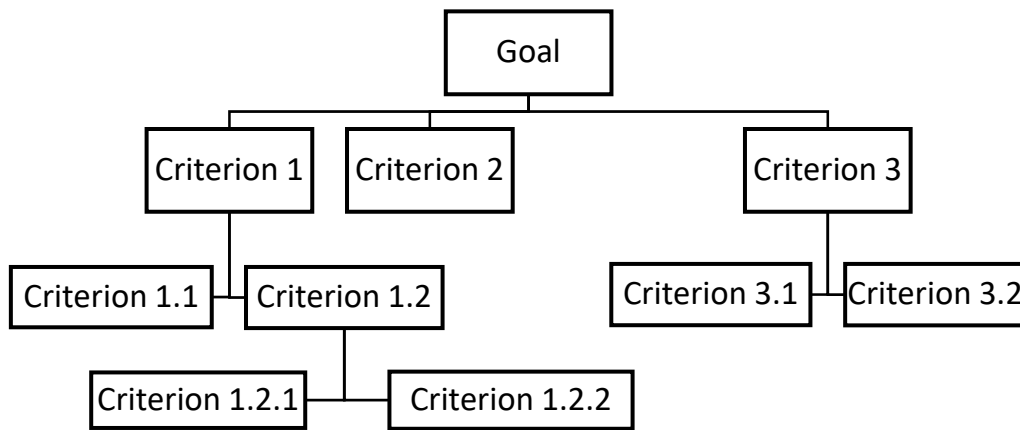


Figure 4: Criterion structure with hierarchy

For example, to solve Criterion 1.2, we need first to solve Criterion 1.2.1 and Criterion 1.2.2, which are in the lowest level of the hierarchy. Then we can solve Criterion 1 by arranging the partial results of Criterion 1.1 and Criterion 1.2. Goal can be solved aggregating Criterion 1, Criterion 2 and Criterion 3 results, once they have been solved.

The ELECTRE-H Software Tool supports Ranking and Sorting problems but not Choice. For sorting problems, it applies the ELECTRE-TRI-B method as defined in the paper [5]. As said before, in this work we focus only on Ranking problems. The software implements the ELECTRE-III-H method for propagating partial pre-orders, which has defined in [2] and the ELECTRE-NFS-H method for propagating net flow scores, defined in [9].

2.4 ELECTRE-III-H method

In the elementary criteria level, classical ELECTRE-III method is applied as explained in section 2.2.2, resulting in a partial preorder. For instance, in Figure 3, we would get a preorder $O_{1.2.1}$ from Criterion 1.2.1 and $O_{1.2.2}$ from Criterion 1.2.2. Then, they must be analysed as two new criteria and compared to generate a new $O_{1.2}$ at the upper level. To achieve this, in intermediate levels, ELECTRE-III-H aggregates partial preorders from lower levels. For example, in Figure 3, ELECTRE-III-H method would aggregate preorders O_1 , O_2 and O_3 to obtain the goal preorder O .

In order to propagate the information of the preorders to an upper level, a new way of calculating the partial concordance and discordance indexes was defined. These indexes are based on the Rank Order Value which, for an alternative a in a O_j partial preorder, is the number of alternatives that are preferred to alternative a in this partial preorder. It is denoted by $\Gamma_j(\cdot)$.

Supposing we have a set $A = \{r, s, t, u, v, w, x, y, z\}$ of alternatives that are structured in a partial preorder O_1 , as shown in Figure 4.

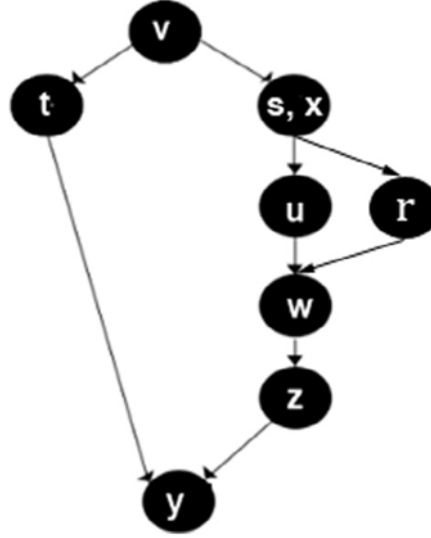


Figure 5: Example of O_1 partial preorder

We can create the relation matrix for this set of alternatives in this preorder with the four types of relations, preference (P), inverse preference (P^-), indifference (I) and incomparability (R). In this relation matrix, a Rank Order Value column can be added as seen in Table 1.

Table 1: Preference relations matrix in O_1 preorder and Rank Order Value

	r	s	t	u	v	w	x	y	z	Γ
r	I	P^-	R	R	P^-	P	P^-	P	P	3
s	P	I	R	P	P^-	P	I	P	P	1
t	R	R	I	R	P^-	R	R	P	R	1
u	R	P^-	R	I	P^-	P	P^-	P	P	3

v	<i>P</i>	<i>P</i>	<i>P</i>	<i>P</i>	<i>I</i>	<i>P</i>	<i>P</i>	<i>P</i>	<i>P</i>	0
w	<i>P</i> ⁻	<i>P</i> ⁻	<i>R</i>	<i>P</i> ⁻	<i>P</i> ⁻	<i>I</i>	<i>P</i> ⁻	<i>P</i>	<i>P</i>	5
x	<i>P</i>	<i>I</i>	<i>R</i>	<i>P</i>	<i>P</i> ⁻	<i>P</i>	<i>I</i>	<i>P</i>	<i>P</i>	1
y	<i>P</i> ⁻	<i>P</i> ⁻	<i>P</i> ⁻	<i>P</i> ⁻	<i>P</i> ⁻	<i>P</i> ⁻	<i>P</i> ⁻	<i>I</i>	<i>P</i> ⁻	8
z	<i>P</i> ⁻	<i>P</i> ⁻	<i>R</i>	<i>P</i> ⁻	<i>P</i> ⁻	<i>P</i> ⁻	<i>P</i> ⁻	<i>P</i>	<i>I</i>	6

As we can see in Table 1, $\Gamma_j(\cdot)$ depends on the number of alternatives that are preferred to the one being evaluated. For example, $\Gamma_j(y) = 8$ because all alternatives are preferred to *y*, and $\Gamma_j(v) = 0$ because no alternative is preferred to *v*.

Depending on the relation type between a pair of alternatives (*a*, *b*), the calculation of partial concordance and partial discordance are different:

- **Preference (*P*) and Indifference (*I*):** in both relation types, the *aSb* assertion is supported, so $c_j(aPb)$ and $c_j(aIb) = 1$, whereas $d_j(aPb)$ and $d_j(aIb) = 0$.
- **Inverse preference (*P*⁻):** in this relation, *aSb* is not supported, but some tolerance is allowed depending on the thresholds:

$$c_j(aP^{-}b) = \begin{cases} 1 & \text{if } \Gamma_j(a) - \Gamma_j(b) \leq q_j \\ 0 & \text{if } \Gamma_j(a) - \Gamma_j(b) > v_j \\ \frac{p_j(a) - (\Gamma_j(a) - \Gamma_j(b))}{p_j - q_j} & \text{otherwise} \end{cases} \quad (2.11)$$

$$d_j(aP^{-}b) = \begin{cases} 1 & \text{if } \Gamma_j(a) - \Gamma_j(b) > v_j \\ 0 & \text{if } \Gamma_j(a) - \Gamma_j(b) \leq p_j \\ \frac{\Gamma_j(a) - \Gamma_j(b) - p_j(a)}{v_j - p_j} & \text{otherwise} \end{cases} \quad (2.12)$$

- **Incomparability (*R*):** this relation type means there is not a direct relation between alternatives *a* and *b*. It is not possible to state if *a* is *P*, *P*⁻ or *I* to *b*, because there is the same probability that it turns into *aPb*, *aP*⁻*b* or *aIb*. In 2/3 of the cases, *aSb* is supported, so we can assume this value for the partial concordance. Hence, the remaining 1/3 is assumed for the partial discordance. Moreover, an α factor is introduced to increase or decrease the partial concordance or discordance based on the magnitude of the difference between the alternatives:

$$\text{if } \Gamma(a) - \Gamma(b) \leq p_j, \text{ then } \begin{cases} c_j(aRb) = \frac{2}{3} + \frac{(\Gamma_j(b) - \Gamma_j(a) - q_j) \times \alpha}{(p_j - q_j) + (n - 2)} \\ d_j(aRb) = 0 \end{cases} \quad (2.13)$$

$$\text{if } \Gamma(a) - \Gamma(b) > p_j, \text{ then } \begin{cases} d_j(aRb) = \frac{1}{3} + \frac{(\Gamma_j(a) - \Gamma_j(b) - v_j) \times \alpha}{(v_j - p_j) + (n - 2)} \\ c_j(aRb) = 0 \end{cases} \quad (2.14)$$

2.5 ELECTRE-H Software Package

This package has been developed by different members of the ITAKA research group, since 2015. It is implemented in Java and provides the methods explained before.

When the program starts, you can select the problem type you want to solve. You can select Ranking, or Sorting. As this work is focused on Ranking problems, all the available options regarding this kind of problems are explained down below.

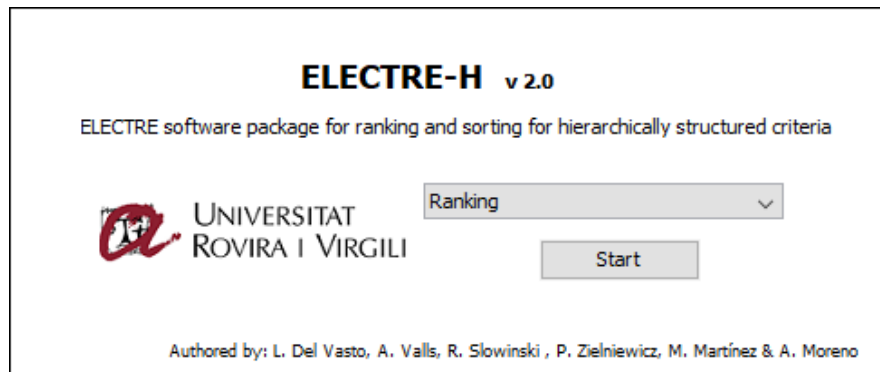


Figure 6: ELECTRE-H v 2.0 initial screen

The main window of the software is the following one.

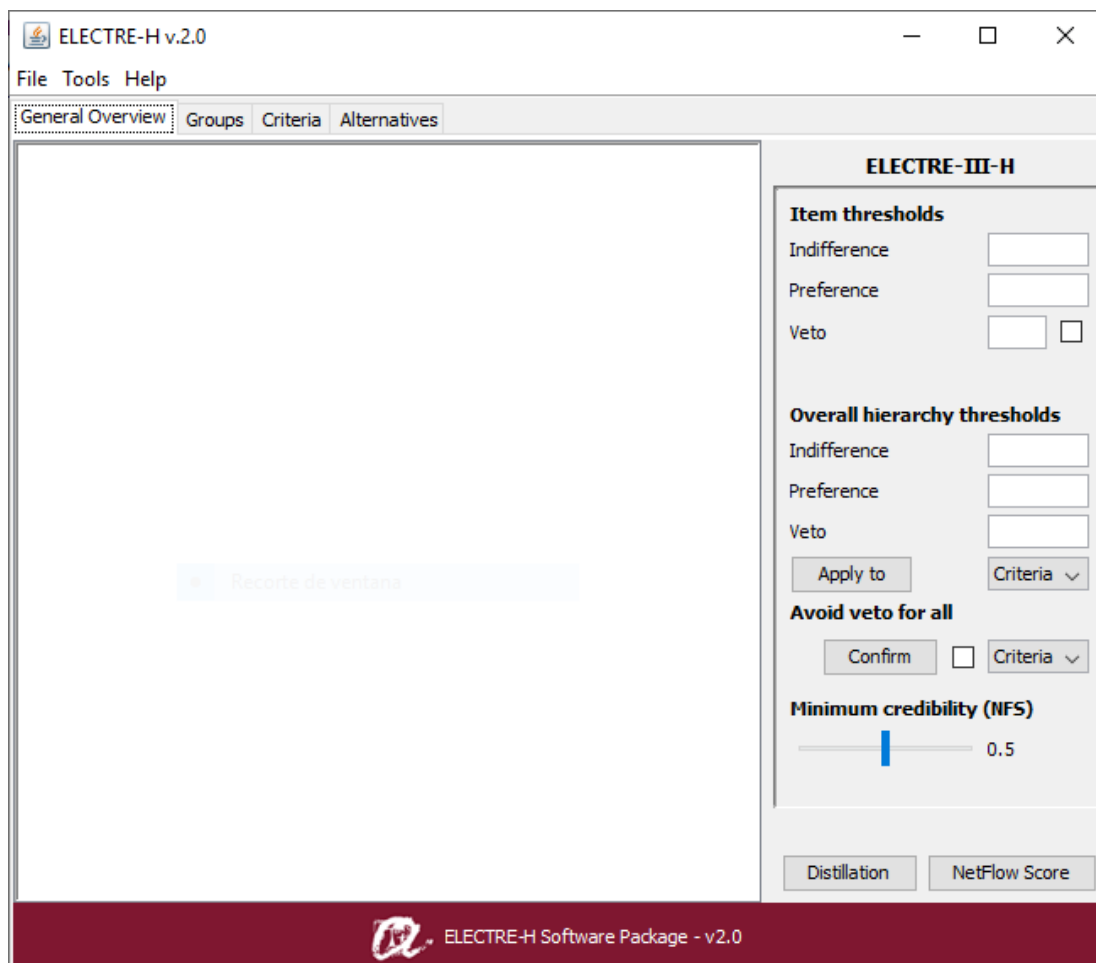


Figure 7: ELECTRE-H main window

Background

To start using the program, a problem has to be loaded from an Excel file, including the criteria hierarchy, and a set of alternatives to be evaluated. To do so, you have to introduce the file location and the rows and columns containing the problem data in the Reader panel that can be found in **File → Read hierarchy from worksheet**.

Figure 8: ELECTRE-H reader panel

Once the problem hierarchy has been loaded, it will be shown in the *General Overview* tab.

Figure 9: General overview loaded from an Excel problem file

Background

Selecting any criterion (Intermediate or elementary), will allow you to see and modify the Indifference, Preference and Veto thresholds for it, as well as enabling or disabling the right to veto.

To apply the same thresholds to all groups or criteria, you have the *Overall hierarchy thresholds* section. In addition, you can enable or disable the right to veto to all groups or criteria in the *Avoid veto for all* section.

You can use Distillation or Net Flow Score exploitation to solve the problem. To use the second one, the user must determine the minimum credibility level using the *Minimum credibility* slider.

In the following figure, you can see the results of solving a problem using Distillation. This panel can be recovered in **Tools → Results**.

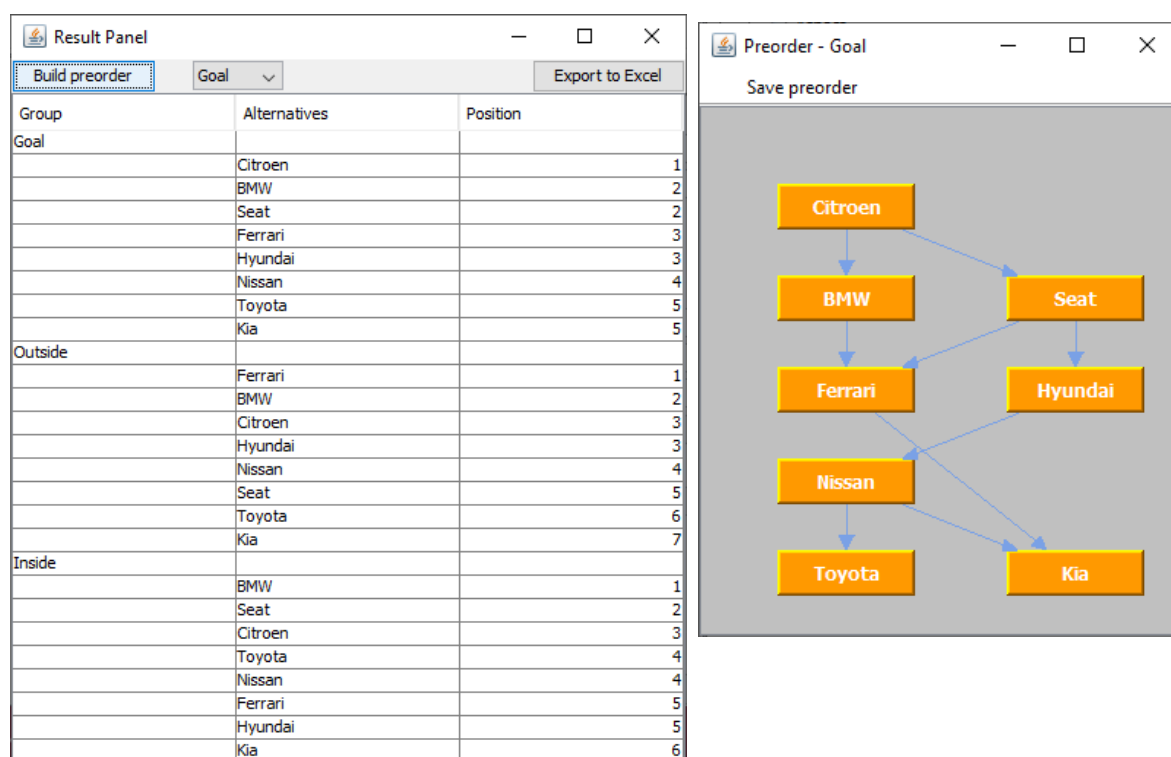


Figure 10: ELECTRE-H Distillation results

You also have the Groups, Criteria, and Alternatives tab, where you can see the groups, criteria and alternatives the problem you have loaded has.

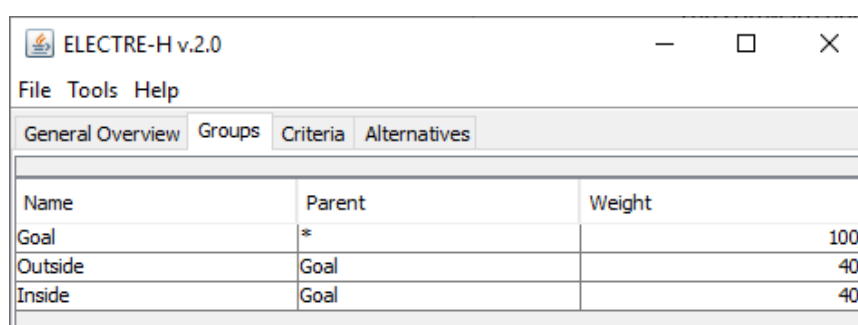
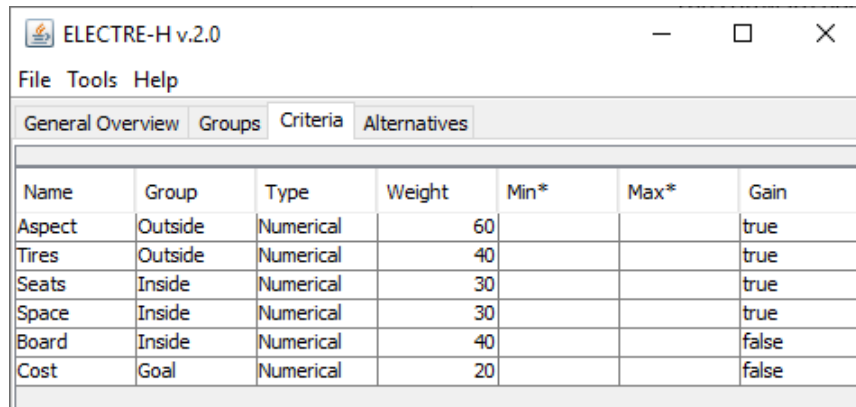
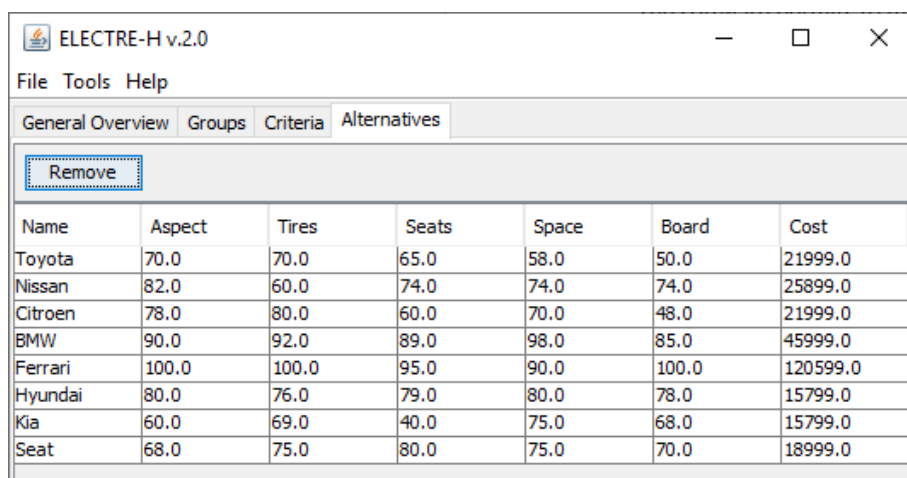


Figure 11: Visualization of groups with their definitions



Name	Group	Type	Weight	Min*	Max*	Gain
Aspect	Outside	Numerical	60			true
Tires	Outside	Numerical	40			true
Seats	Inside	Numerical	30			true
Space	Inside	Numerical	30			true
Board	Inside	Numerical	40			false
Cost	Goal	Numerical	20			false

Figure 12: Visualization of criteria description



Name	Aspect	Tires	Seats	Space	Board	Cost
Toyota	70.0	70.0	65.0	58.0	50.0	21999.0
Nissan	82.0	60.0	74.0	74.0	74.0	25899.0
Citroen	78.0	80.0	60.0	70.0	48.0	21999.0
BMW	90.0	92.0	89.0	98.0	85.0	45999.0
Ferrari	100.0	100.0	95.0	90.0	100.0	120599.0
Hyundai	80.0	76.0	79.0	80.0	78.0	15799.0
Kia	60.0	69.0	40.0	75.0	68.0	15799.0
Seat	68.0	75.0	80.0	75.0	70.0	18999.0

Figure 13: Visualization of the performance matrix

When the problem has been solved, some new options will be available to the user to use. The first one is saving the Relation Matrix into a .matrix file, for a selected group.

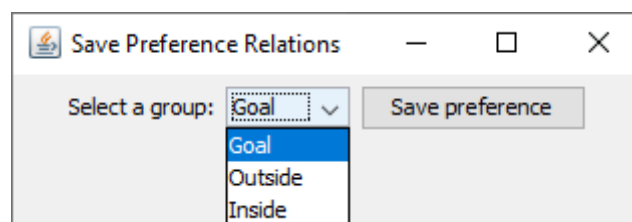


Figure 14: Save preference relations option

This relation matrix has the binary relations of all alternatives. Relations can be Preference, Inverse Preference, Incomparability or Indifference. You can use the Pre-Order Distance panel to compare two results of the same problem, and get the distance between them.

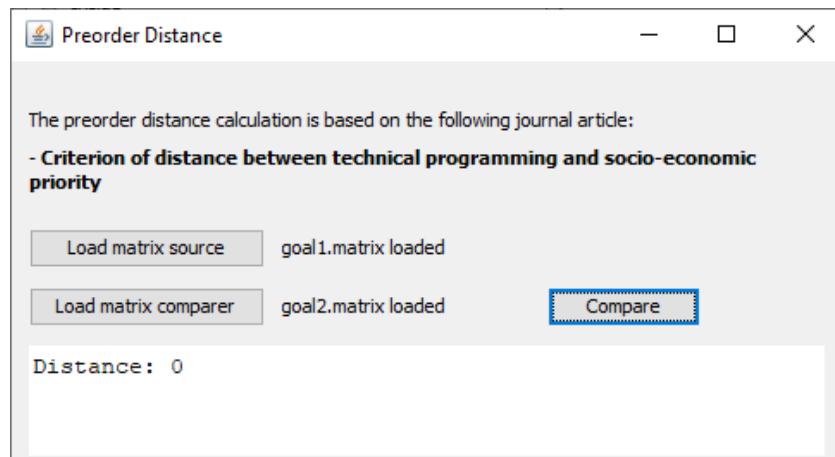


Figure 15: Preorder Distance

To get correlations between two classifications or rankings, you can use the Statistical Panel. You can calculate Spearman, Kendall Tau and Game Coefficient.

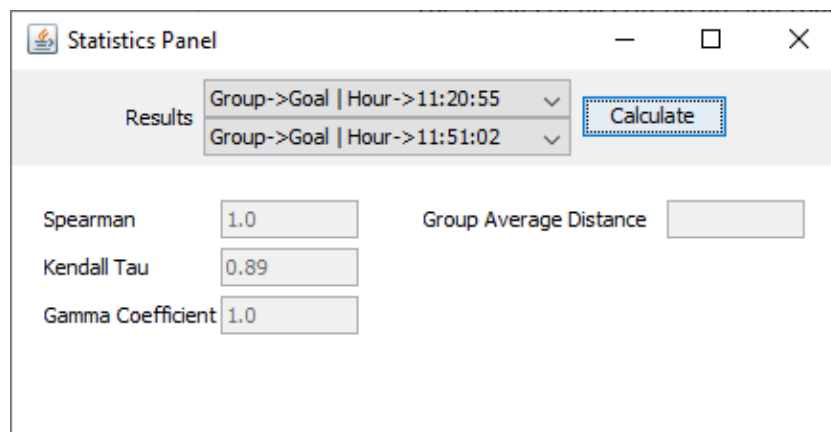


Figure 16: Statistics Panel

2.6 Preference learning

In the literature, we can find four methods to help the decision maker when choosing the parameters that are used in the ELECTRE method.

The first approach is inference-based [6]. This approach consists on finding the veto thresholds by making use of mathematical optimization methods such as linear programming, 0-1 linear programming or separable programming. The constraints introduced to the model are elicited from the information provided by the decision maker. The decision maker is asked to compare a set of pairs of alternatives and establish a preference relation of each of these pairs. Then, using an aggregation-disaggregation approach the mathematical linear program is formulated to find a feasible solution, which consists on a value for some of the ELECTRE-III thresholds (indifference, preference or veto). An extension of the first method, includes the management of inconsistent preference relations given by the user. An algorithm was proposed to solve the inconsistency that occurs when no feasible values of model parameters permit a model to represent decision examples given by the decision maker. To solve this, the decision maker chooses between different given options of inconsistent examples [6].

A second approach is ELECTRE^{GKMS} which applies Robust Ordinal Regression (ROR) in ELECTRE. This approach is, in fact, an extension of the first one. The main difference is that the goal is not to find a single feasible value for the parameters, but to establish the region of feasible values for each of them. This interactive method develops preference models by getting some information of the decision maker, and then using it to infer his preferences. To get this information, the decision maker is provided with some pairwise comparisons, that can be expressed with respect to all criteria, or a particular criterion, and the decision maker chooses his preference [1][10]. This type of results allows to derive some robust conclusions about the possible values of each parameter. The set of possible values is reduced by interactive refinement with new information given by the decision maker at each step.

A third approach is SMAA-III, which simulates and studies the effects produced by the change of the values of parameters on the results. In this method, the weight, criteria performance scores and thresholds are explored. The objective is to find the values that lead to some pre-determined rankings of the set of alternatives. It allows the ELECTRE-III method to work with partially missing information. The weights can be provided using intervals, for example. Some version used in SMAA-III permits to elicit the preferences with one or more decision makers, for example to rank the criteria by importance [11].

However, all those methods are used for classic ELECTRE, which does not consider a hierarchical structure as the one we will study in this work.

3 Specifications

An analysis of the new functionalities that we should add to the ELECTRE-H Software was done taking into account the goals presented in Chapter 1. A list of specification for the design and development was elaborated:

- After loading a Ranking problem, a new thread will solve that problem using distillation. The value of the maximum gamma for each intermediate group will be stored internally.
- It will also calculate the default discrimination values, and load all the sheets in the problem Excel as user defined policies.
- A *Discrimination policies* section will be added to the main screen. It will contain a policies slider, and two buttons *Reset* and *Reset All*.
- The policies slider will initially have the default discrimination policies (Very Tolerant, Tolerant, Neutral, Strict, Very Strict), and No classification, and it will be disabled by default. It will be updated with the user defined policies after they have been loaded.
- When the user selects an intermediate group in the hierarchy tree, the policies slider will be enabled, and it will select the policy that has its thresholds as close as possible to the group ones.
- When the user selects an intermediate group in the hierarchy tree, and changes the policy using the policies slider, the thresholds for that group will change to the ones of the selected policy. Item thresholds text boxes will also update their value.
- When the user selects an elementary criterion, policies slider will default to No classification and will be disabled.
- Two checkboxes will be added to the main screen. A *Stop after each node*, and a *Stop after selected node*.
- When the user solves the problem using any of the exploitation methods, the process will stop after finishing solving all the groups that have been selected using the stop checkboxes, showing an Intermediate Result panel. In this panel, the user will be able to use the discrimination policies.
- A Compare Preorders panel will be accessible under Tools → Statistical Analysis → Compare Preorders. It will allow the user to load to relation matrix of the same problem, and compare their results based on the ranking positions.
- After a problem has been loaded, the user will be able to access the Exploratory Analysis panel, under Tools menu. Using the discrimination policies, it will get multiple different solutions of the problem. These solutions could be compared with the Acceptability and Relation index.

4 Design and methodology

In this section, the new panels and functionalities that will be introduced to ELECTRE-H software, will be explained. To exemplify it, a sample hierarchy included in the software package will be used, Sample Ranking. This problem ranks different cars according to the following criteria hierarchy.

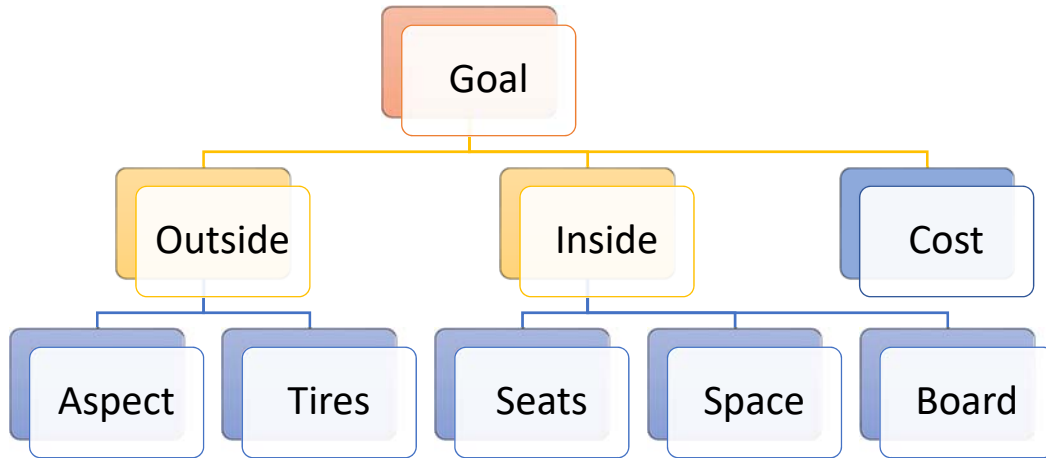


Figure 17: Sample Ranking criteria hierarchy

We can see in red the root group, *Goal*. It has an elementary criterion, *Cost*, marked in blue. It also has two intermediate groups, which are *Outside* and *Inside*, marked in yellow. Both intermediate groups have some elementary criteria, also marked in blue. *Outside* has *Aspect* and *Tires*, and *Inside* *Seats*, *Space* and *Board*.

The alternatives this problem has, and its performance matrix, is shown in the following table.

Table 2: Sample Ranking performance matrix

Name	Aspect	Tires	Seats	Space	Board	Cost
<i>Toyota</i>	70	70	65	58	50	21999
<i>Nissan</i>	82	60	74	74	74	25899
<i>Citroen</i>	78	80	60	70	48	21999
<i>BMW</i>	90	92	89	98	85	45999
<i>Ferrari</i>	100	100	95	90	100	120599
<i>Hyundai</i>	80	76	79	80	78	15799
<i>Kia</i>	60	69	40	75	68	15799
<i>Seat</i>	68	75	80	75	70	18999

4.1 Discrimination policies

First of all, the changes that will be done to the already existing main window, are shown in Figure 18. Then, the explanation of all the functionalities will be done.

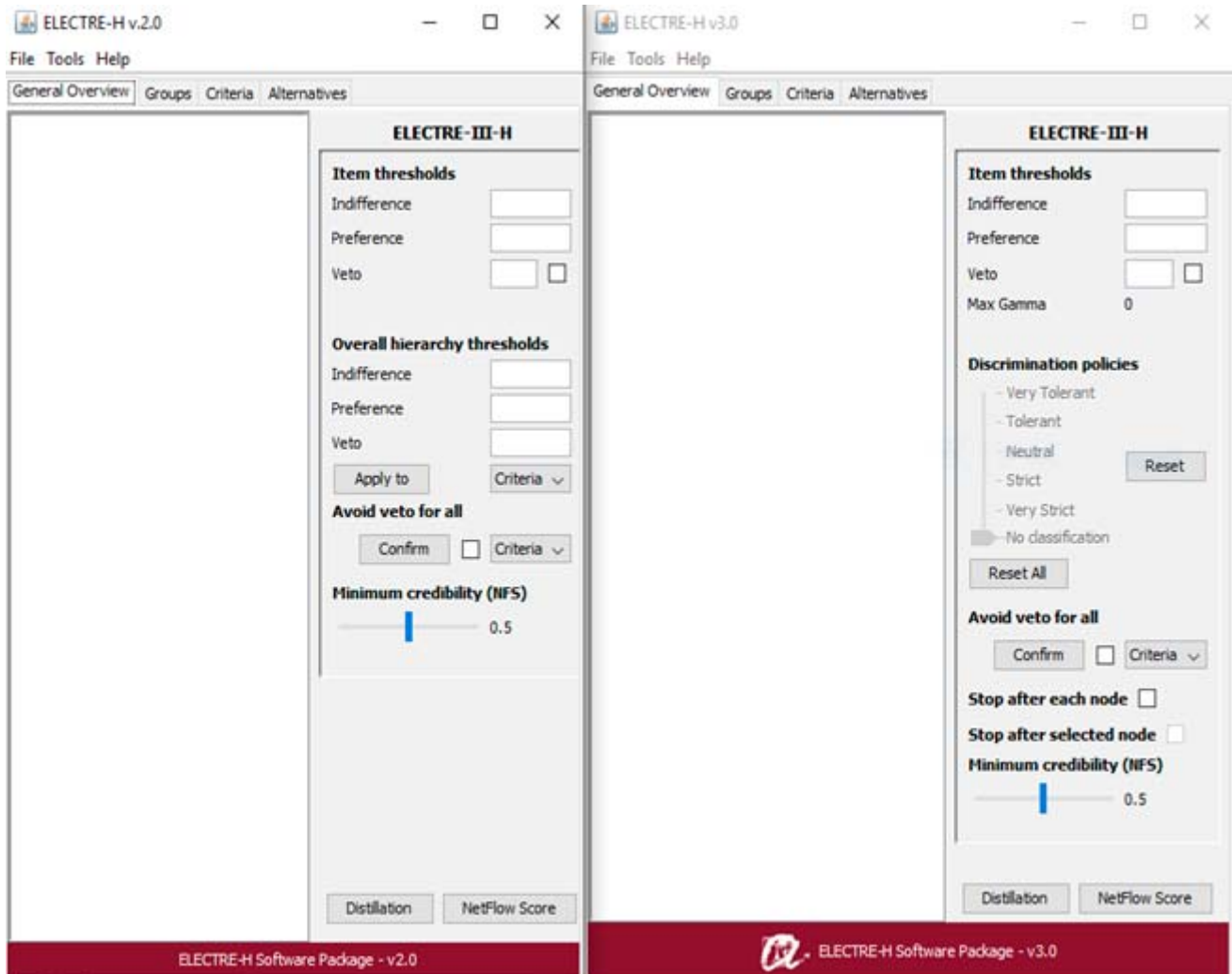


Figure 18: ELECTRE-H v 2.0 (left) and v 3.0 (right)

When an intermediate criterion is selected in the hierarchy, the program will activate the *Discrimination policies* section in the main panel, which consists on a policies slider. It will allow you to use some existing policies to change the thresholds of the selected criterion. You can use the predefined policies, that range between Very Strict, Strict, Neutral, Tolerant, and Very Tolerant, or user defined policies.

This last type of policies are loaded from the problem Excel. When you have finished modifying the thresholds to some you may have interest in, you can save them as a new user defined discrimination policy using the *Save hierarchy as a policy* button, found in the File menu. The program will save the modified data in a new sheet of the loaded Excel file. The name of the sheet will be the name of the policy.

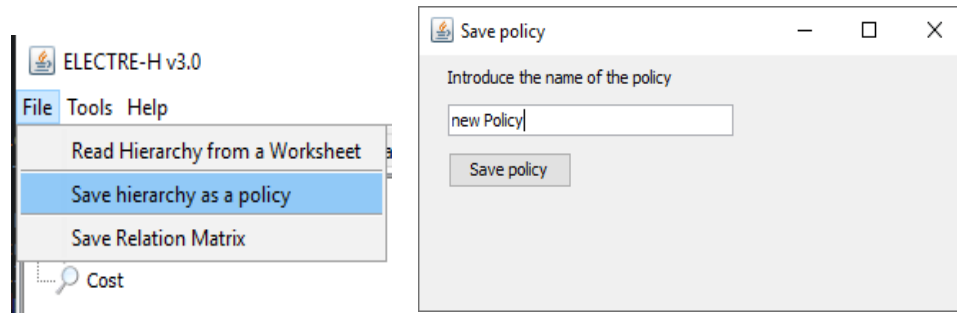


Figure 19: Save the current hierarchy as a new policy

Loading the Sample Ranking problem, we can see that the policies slider has the 5 predefined discrimination policies, and a user defined policy, named *Default*.



Figure 20: Sample Ranking policies slider

Moving the slider will automatically change the thresholds of the selected criterion to the ones that the program has pre-calculated for that policy.

You also have a *Reset* button in case you want to recover the threshold values you loaded from the Excel file at a certain intermediate criterion, or the *Reset All* button to recover all initial thresholds to all intermediate criteria.

In addition, new values for the thresholds can be directly introduced on the fields at *Item Thresholds* section. The slider will be automatically moved to the most similar policy to the values you introduced, including the predefined policies, and user defined policies.

If any policy is similar enough to the introduced values, *No classification* will be chosen.

In the following image, you can see an example of use of the discrimination policies slider. On the left panel you can see that the *Default* user defined policy is selected, because it is the most similar policy to the current threshold values $q_j = 0$, $p_j = 1$ and $v_j = 3$.

If we change the slider to another policy, for example *Strict*, these thresholds will change, as can be seen in the image on the right. For the *Strict* predefined policy, thresholds are in this example $q_j = 0$, $p_j = 0$ and $v_j = 1$.

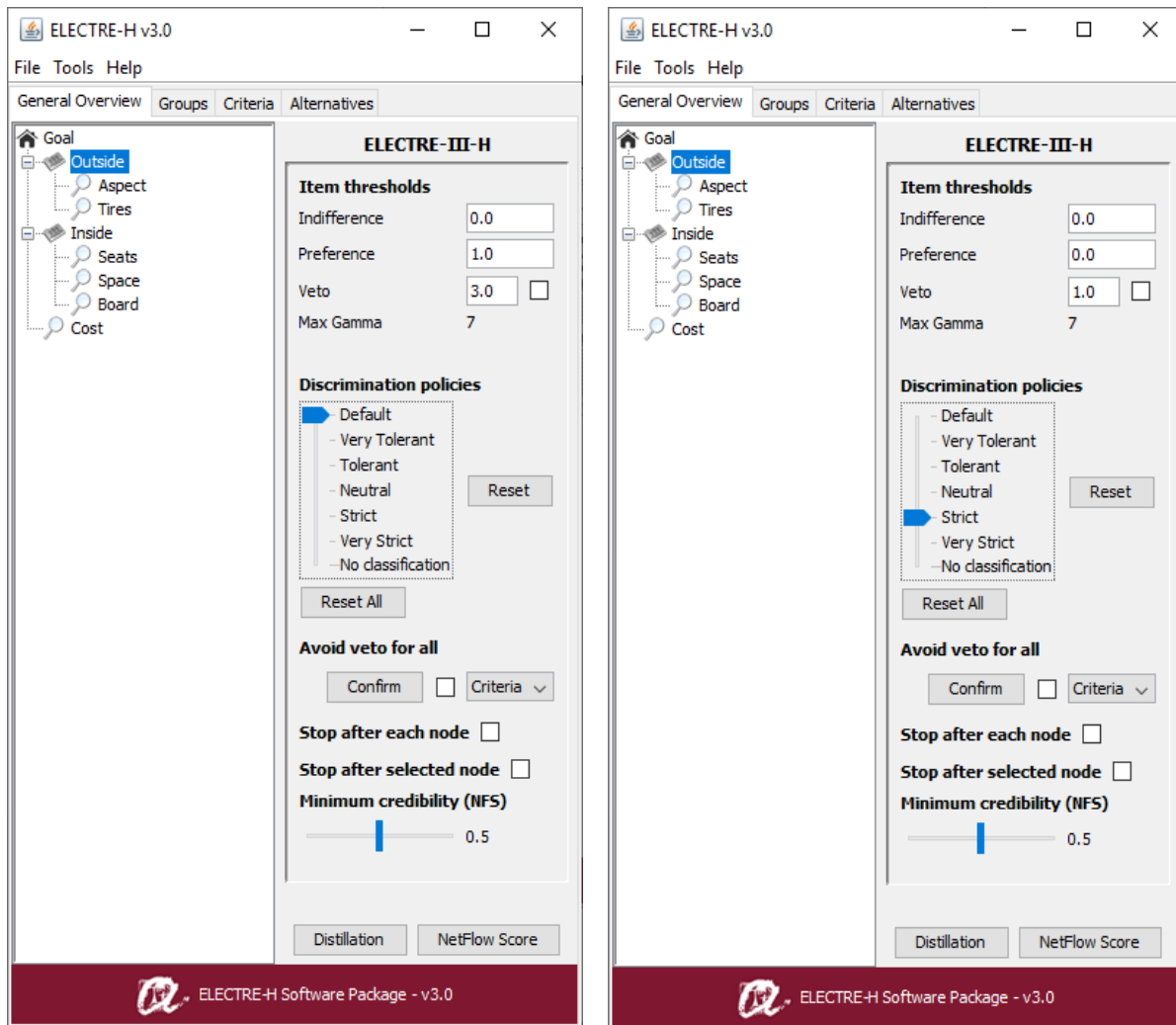


Figure 21: Customizing intermediate criterion thresholds using Discrimination Policies

There is the option to stop the solving process after solving intermediate groups. It can be done by checking the *Stop after each node* checkbox if you want to stop after each intermediate group, or *Stop after selected node* to stop after solving the intermediate group you have selected in the criteria hierarchy.

In the following image, we can see how stopping after solving *Outside* group is done.

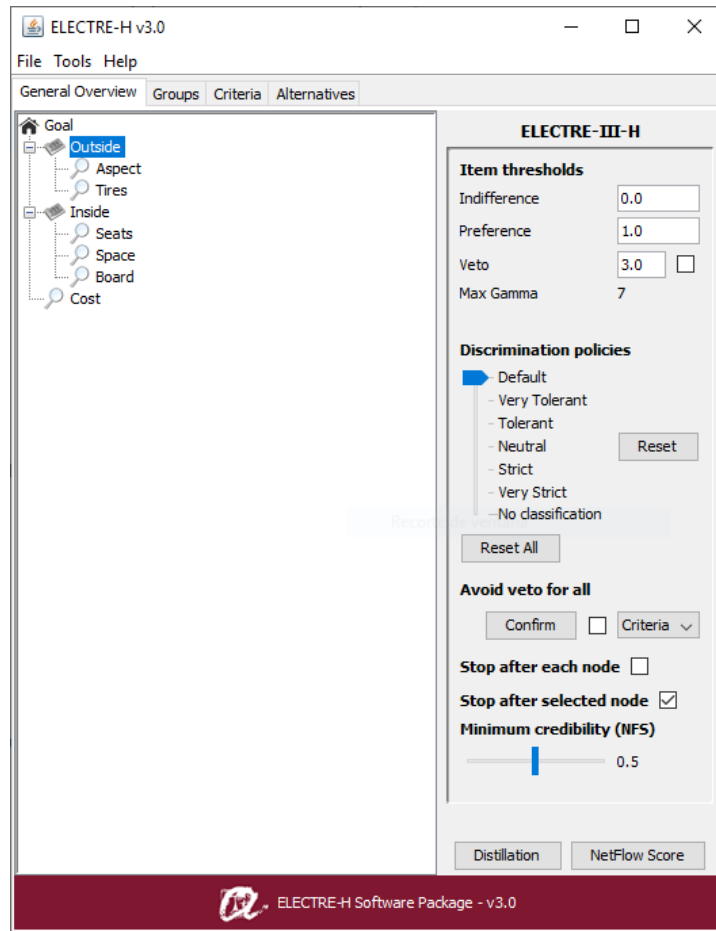


Figure 22: Stop the solving process

When solving the problem by using Distillation or Net Flow Score, after the selected group has been solved, an Intermediate Result Panel is shown. In this panel, we can see the ranking for that group and the Rank Order Value of each alternative. We can save these results by building and saving its preorder, or by saving the intermediate result table as an Excel file.

In addition, you can modify the thresholds for that group before continuing the solving process. If the problem is being solved using Distillation, you can also use the discrimination policies to change the thresholds.

In the following figure, we can see on the left the Intermediate Result Panel of the *Outside* intermediate group. The preorder of this intermediate result is shown on the right.

As the problem is being solved using Distillation, the thresholds for the *Outside* group can be changed using the discrimination policies. We can see that we applied the Very Strict policy. This policy changed the thresholds to $q_j = 0$, $p_j = 0$ and $v_j = 1$, from the original $q_j = 0$, $p_j = 1$ and $v_j = 3$ that can be seen on the preorder.

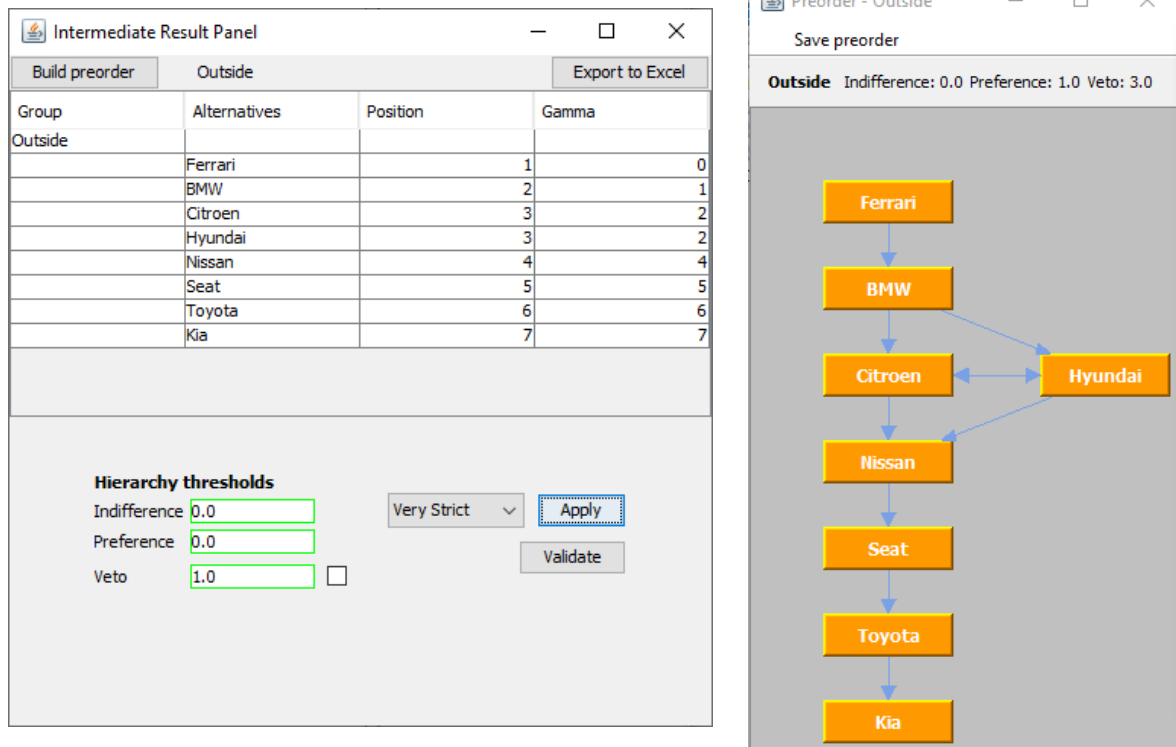


Figure 23: Intermediate Result Panel

4.2 Compare preorders

This tool can be found under **Tools** → **Statistical analysis** → **Compare preorders**. It is used to find significant changes in two preorders using their .matrix files.

This tool compares two preorders, and in order to do this, they must have the same alternatives.

Using the Load first preorder and Load second preorder buttons, we can load the .matrix files we want to compare. Once they are loaded, you can build their preorders using the Show preorders button. The built preorder is the same you get when you solve the problem, the same you can see in Figure 10.

Then, we can compare them by pressing *Compare*. If any difference has been found, it will be shown in the table, as can be seen in Figure 24. Only significant changes will be shown, so an alternative must change its final position from quartile to be shown.

Depending on the number of ranking positions on the .matrix files, 2 or 4 quartiles are made.

After pressing *Compare*, you can see the alternative, the position change, and the quartile change. The quartile cell will be painted depending on the change. Green for a positive change, light green for a slight positive change, light red for a slight negative change, and red for a negative change.

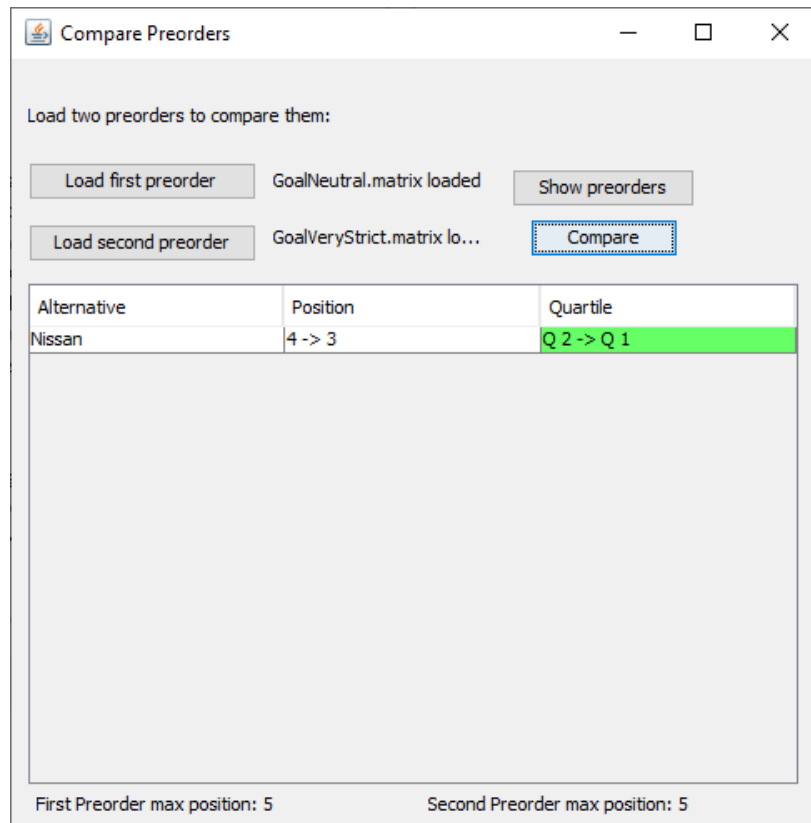


Figure 24: Preorders compare panel

In the example shown in this figure, two .matrix files are loaded. The first one is the solution of Sample Ranking applying the Neutral discrimination policy in all intermediate nodes. The second one is the solution of Sample Ranking, but applying the Very Strict policy in all intermediate nodes. The result of this comparison, is a slight positive change of alternative Nissan, which in the second preorder has a third position, instead of the fourth it got in the first preorder.

4.3 Exploratory Analysis

This panel can be found in **Tools** → **Exploratory Analysis**. It has some tools to analyze the results of each policy. The aim is to help you to find the most appropriate policy for each of the intermediate criteria, especially for large hierarchies of criteria. This tool applies a certain policy to all the hierarchy tree and then you can compare the results using the visualization tools available in this window, as will be explained here.

The system uses the predefined discrimination policies, Very Strict, Strict, Neutral, Tolerant and Very Tolerant, as well as the user defined ones (and saved). All the sheets found in the problem Excel will be loaded as new policies.

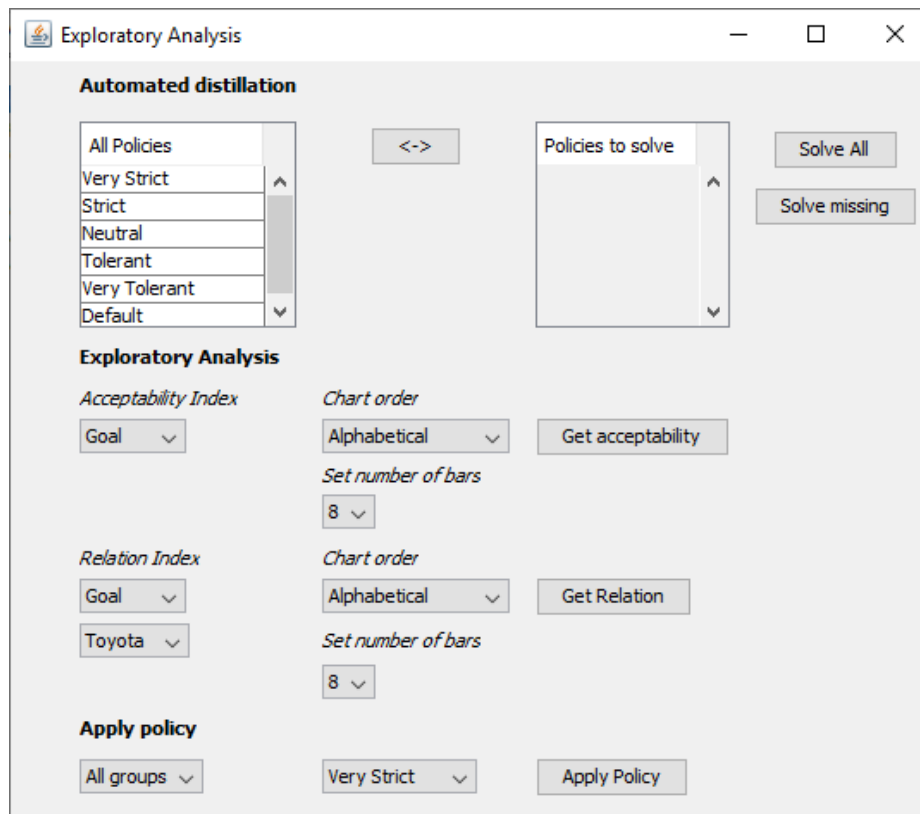


Figure 25: Exploratory Analysis panel

The first tool you can find is *Automated distillation*. In it, you can select three or more policies, and solve the problem automatically for all of them with their corresponding values on the parameters. The program will apply each selected policy to all groups, and then will solve the problem using Distillation. All policies are available to be used, the 5 predefined policies, and the user defined ones. In Figure 25, we can see that apart from the predefined policies, we have one user defined one, *Default*.

To select the policies, you have to select them in the *All Policies* table, and click the button <->. This button changes the selected policies from the table where they are, to the other one.

The *Solve All* button will solve all the policies in the *Policies to solve* table, and the *Solve missing* button will only solve the policies in *Policies to solve* that have not been solved previously. In both options, a Result Panel for each policy solved will appear.

The second tool is the *Exploratory Analysis* of the results obtained when solving the problem using the selected policies. It uses the policies on *Policies to solve* table to create two different charts.

The first one is the *Acceptability Index*. First, it divides the alternatives into two or four quartiles, depending on the number of positions in the ranking.

Then it shows using a stacked bar chart, how many policies assign the alternative in each of the quartiles. Before creating the chart, you can select the order of the alternatives, and the number of bars that will appear in the chart without moving the horizontal slider. You can see which percentage of policies belong to each quartile, and which policies are. You can know that by placing the mouse in a bar, so a tooltip showing it will appear. With the *Get*

policies per quartile button you can also see this for the selected alternative, as can be seen in the following image.

You can save the image of the acceptability index or save the data as an Excel file.



Figure 26: Acceptability Index chart

In this example, Acceptability Index is calculated in the Goal group. We can see that policies Very Strict, Strict, Neutral and Default make the alternative Ferrari finish in the first quartile. By contrast, Tolerant and Very Tolerant, make Ferrari finish in the second quartile, so we can conclude that applying one of these policies will make Ferrari have a worse ranking position than the other ones.

The second tool for exploration is the *Relation Index*. Before generating the chart, you can select an alternative, which will be compared to the others. The chart will show the performance between the selected alternative, and the other ones. The possible performance relations are Preference, Indifference, Incomparability, and Inverse Preference.

You will know which performance has each alternative when applying each policy.

Before creating the chart, you can also select the order of the alternatives, and the number of bars that will appear in the chart without moving the horizontal slider.

Moreover, you will be able to save an image of the chart as it is on the screen, and to generate an Excel with the data that created the chart.

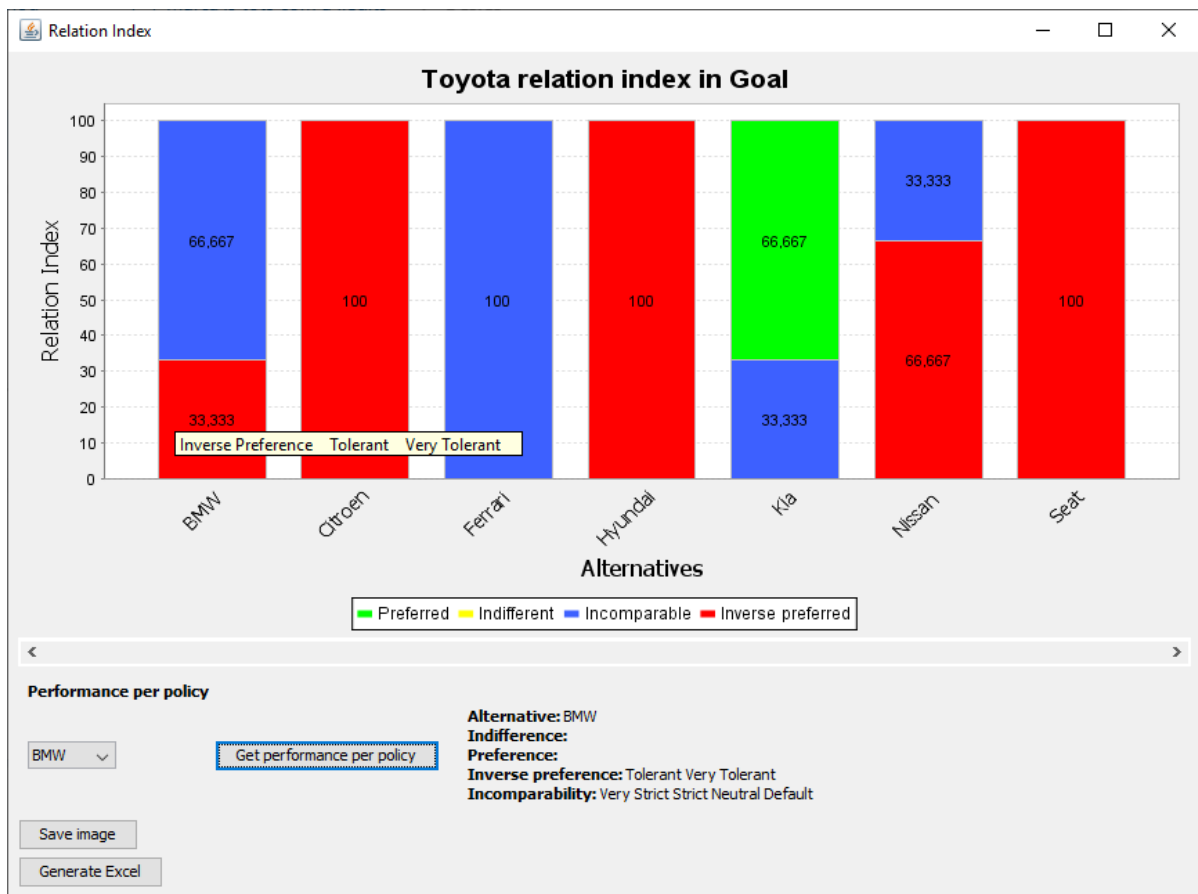


Figure 27: Relation Index chart

In this example, you can see that Toyota is compared to the other alternatives in the Goal group. Using Tolerant and Very Tolerant policies, we can see that Toyota is Inverse Preferred to BMW, and using Very Strict, Strict, Neutral or Default, Toyota is incomparable to BMW.

The last tool you can find is *Apply policy*. You can apply a policy to all groups, or to a selected one. This change will be applied to the hierarchy in the main window. When you solve again the problem using Distillation or Net Flow Score, it will use the new thresholds selected here.

5 Development

In this section, the changes done to the ELECTRE-H v 2.0 software package are explained. This program is coded in Java. As it is an object oriented programming language, the code is organized in different classes and packages. First, a brief introduction to the software organization is done. The classes that have a yellow background were modified during the development, the ones with a green background were created from scratch, and the other ones remained as they were.

The first three packages we find in the program, are the base ones, which mainly contain the classes that store all the data needed by the program and some functions related to that data. They are *electreh.base*, *electreh.base.complex*, and *electreh.base.exception*. The latter, has exceptions related to the other base packages. The classes they contain can be seen in the following diagram.

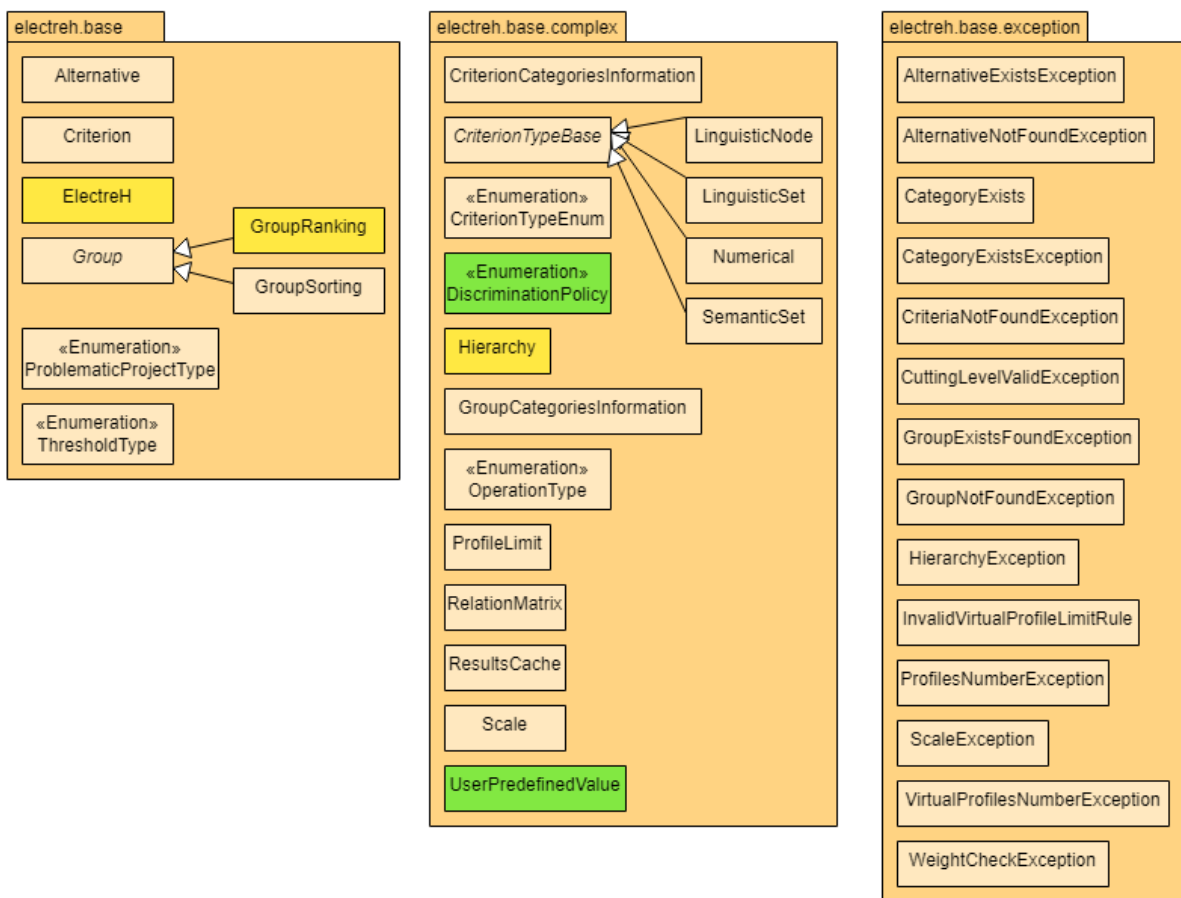


Figure 28: electreh.base packages

The next 5 packages, *electreh.calculation*, *electreh.calculation.ranking outranking*, *electreh.calculation.ranking.exploitation*, *electreh.calculation.sorting outranking* and *electreh.calculation.sorting.exploitation*, contain the classes that have the functions to perform the implementations of outranking and exploitation methods.

Both ranking and sorting problems have a package for their implementation of outranking and exploitation method, including the classical ELECTRE method, as well as the hierarchical version.

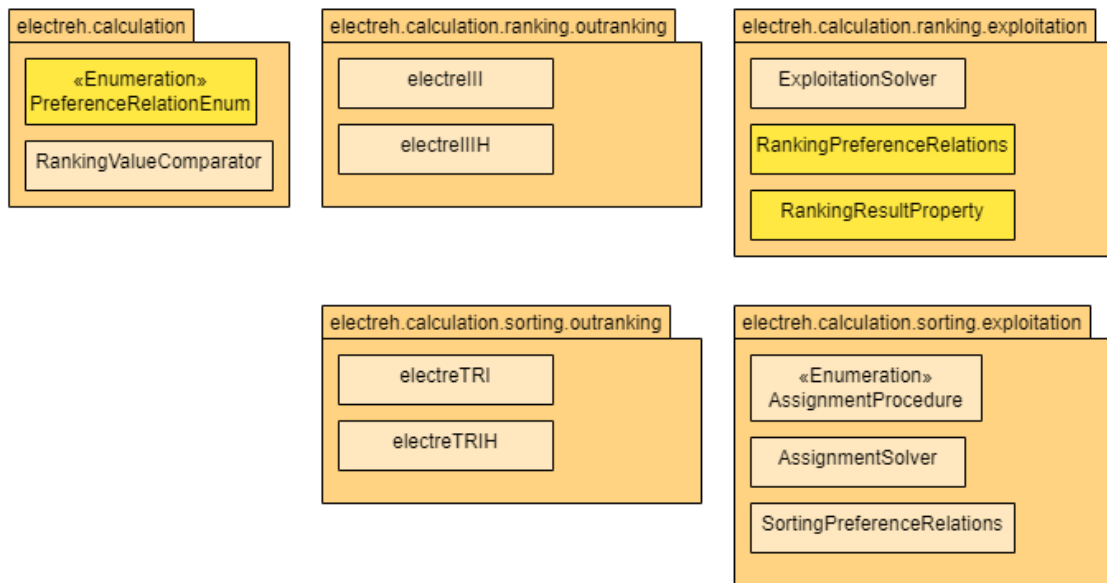


Figure 29: electreh.calculation packages

In the program, we also find two packages that contain all the GUI related classes, *electreh.gui* and *electreh.gui.controls*.

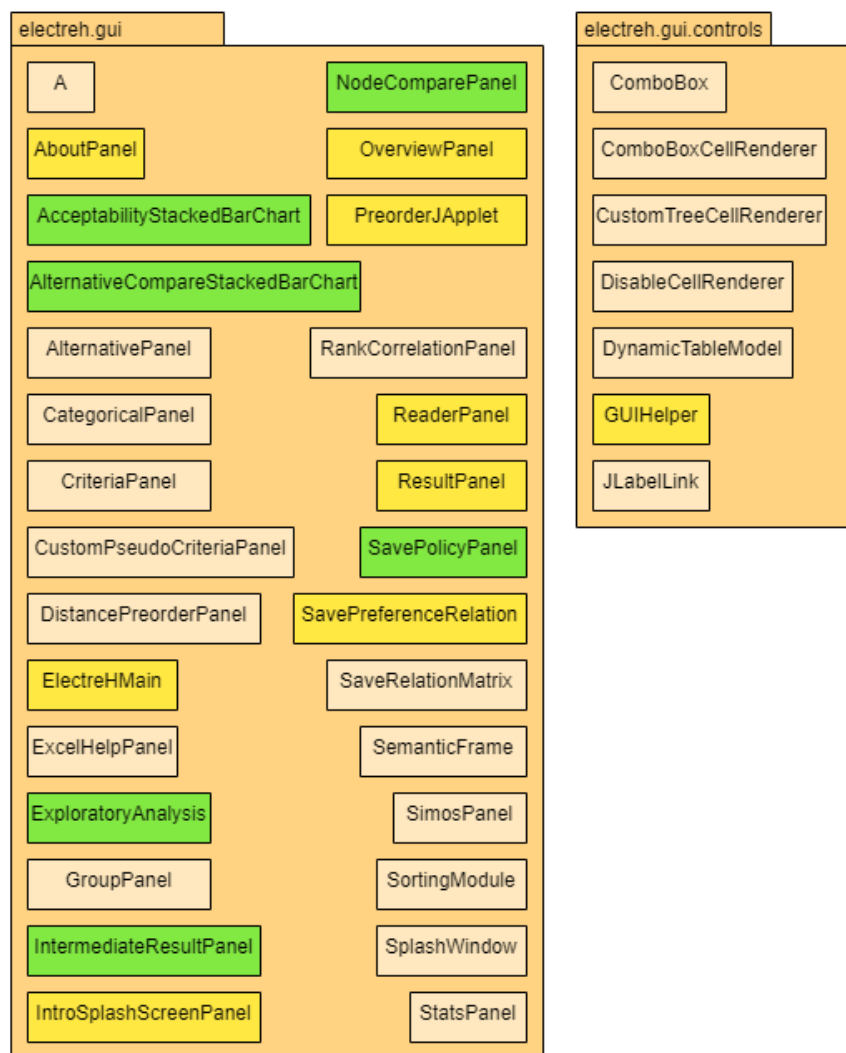


Figure 30: electreh.gui packages

Finally, there are some packages with additional functions needed by the program. For instance, *electreh.excel* to handle Excel file reading and writing, *electreh.help* with some additional classes, and *electreh.io* to perform input/output to save data that has to be persistent to files.

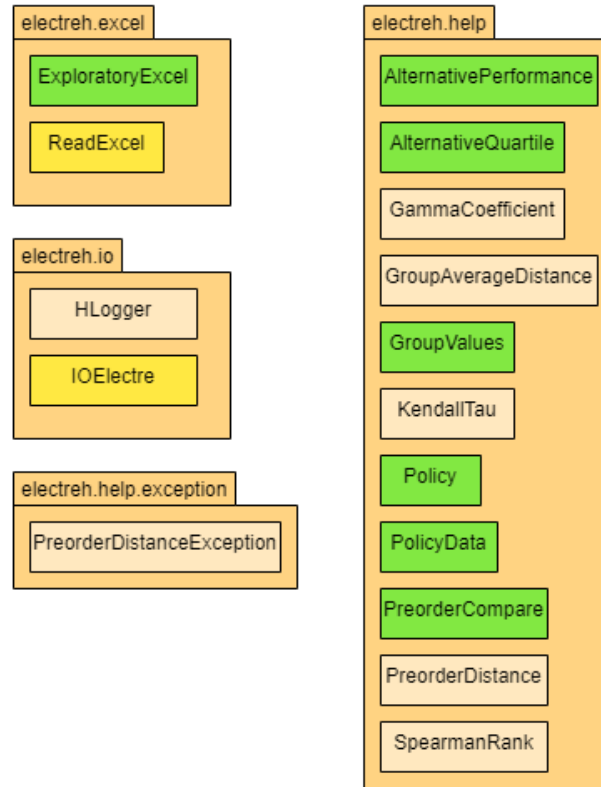


Figure 31: ELECTRE-H help packages

5.1 Discrimination policies

The first thing done was the creation of some classes to store the policies. The first one was *GroupValues*, which stores the 3 thresholds, preference, indifference and veto.

The second one was *Policy*, which has the data of a concrete policy. It saves its complete name, and a policy name that is used internally. This policy name is the complete name but with the blank spaces trimmed, and all letters in upper case.

GroupValues is an associating class, which associates a *Policy* with an intermediate group, saving its thresholds.

Finally, *PolicyData* is a class that contains and manages all the policies. In the following class diagram, you can see the relations between the different classes that handle the policies information.

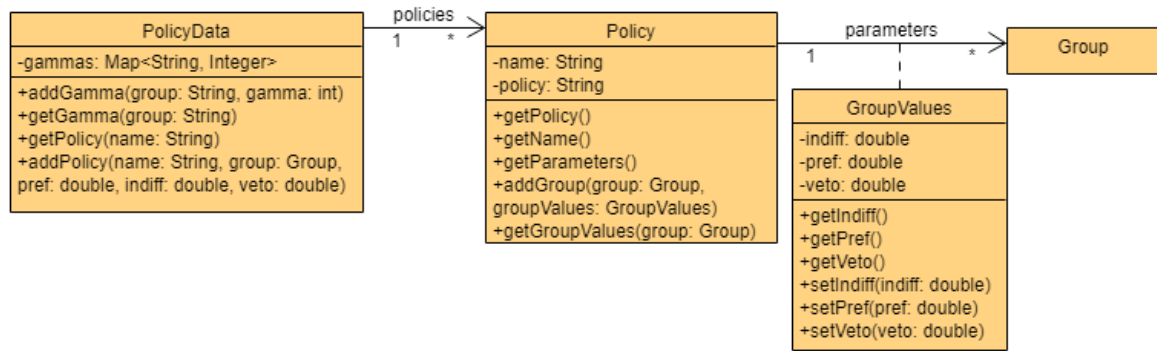


Figure 32: Policies class diagram

The *ReaderPanel* class was modified in order to automatically solve a problem using Distillation after it has been loaded in a background process. Once the problem has been solved, a new method created in *ElectreH* is called, *setPolicyData*. This method adds to *PolicyData* the 5 predefined policies initializing them, reads the problem Excel file to load the user defined policies into *PolicyData* and saves the Rank Order Value of each group.

User defined policies are given by the user, but predefined policies needed a way to be calculated. On the basis that the relation $0 \leq q_j \leq p_j \leq v_j$ had to be fulfilled, and looking at similar works [3] [8], a percentage of the maximum Rank Order Value of the intermediate group was used to calculate the 3 thresholds. The chosen percentages are shown in the following table.

Table 3: Discrimination Policies thresholds

Discrimination policy	Indifference (%)	Preference (%)	Veto (%)
<i>Very Strict</i>	0	0	10
<i>Strict</i>	0	5	20
<i>Neutral</i>	5	10	30
<i>Tolerant</i>	10	20	40
<i>Very Tolerant</i>	15	25	50

The most strict policies had to have a smaller value than the less restrictive ones. With small thresholds, a little difference between alternatives would be considered as significant for the decision, whereas with big thresholds, you have larger interval of uncertainty when calculating when a criterion is in concordance or discordance. For example, the larger the value of veto, the less vetoing situations will be produced, relying more the decision on the majority opinion.

To help doing this calculus, an Enumeration class called *DiscriminationPolicy* was created. It has the 5 predefined discrimination policies and methods to get the values according to Table 3. Using these values, *PolicyData* could be initialized after solving the current problem using the distillation exploitation method, because the Rank Order Value would already be calculated.

When policies are selected in the Policies Slider in the main panel, or in the intermediate result panel, the program gets the corresponding thresholds from *PolicyData*, and replaces the current thresholds by these ones.

The *Hierarchy* class was modified to save the thresholds loaded from the Excel file into a new list of *UserPredefinedValue* classes. When using the *Reset* or *Reset All* buttons, the program gets the thresholds of this list, and replaces the current ones.

When thresholds are changed by the user, the program tries to find a similar policy and selects it on the slider. To check the similarities between the user thresholds and the policies thresholds, the following formula is applied for each policy, being q_u , p_u and v_u the thresholds of the user and q_p , p_p and v_p the thresholds of a certain policy.

$$similarity = |q_u - q_p| + |p_u - p_p| + |v_u - v_p| \quad (5.1)$$

The policy with the lowest similarity value is selected, as long as this value is lower than the maximum Rank Order Value in that group divided by two. If the similarity is not low enough, any policy is considered enough similar, and *No classification* is chosen on the slider.

The Intermediate Result Panel was created having the Result Panel as a baseline. The first one has the same functions as the second one, with the addition of the possibility to change the thresholds manually or using the Discrimination Policies.

The *ElectreH* class was modified to allow stopping the solving process and showing this new panel. As the solving algorithm is recursive, before calling the recursive function again, the program checks if the process has to be stopped for the group it has finished solving. If this is the case, it creates and shows the Intermediate Result Panel, and tries to acquire a semaphore previously initialized at 0. It will not be able to acquire this semaphore, so it will stop the solving thread, but the panel will render and work because it is processed using the AWT Thread, in charge of rendering the panels.

When the user wants to continue the solving process, she can Validate the thresholds values currently displayed at the Intermediate Result Panel, and then the AWT Thread will release the semaphore the solving process is stopped at, and the solving process will continue.

In the following sequence diagram, you can see this process.

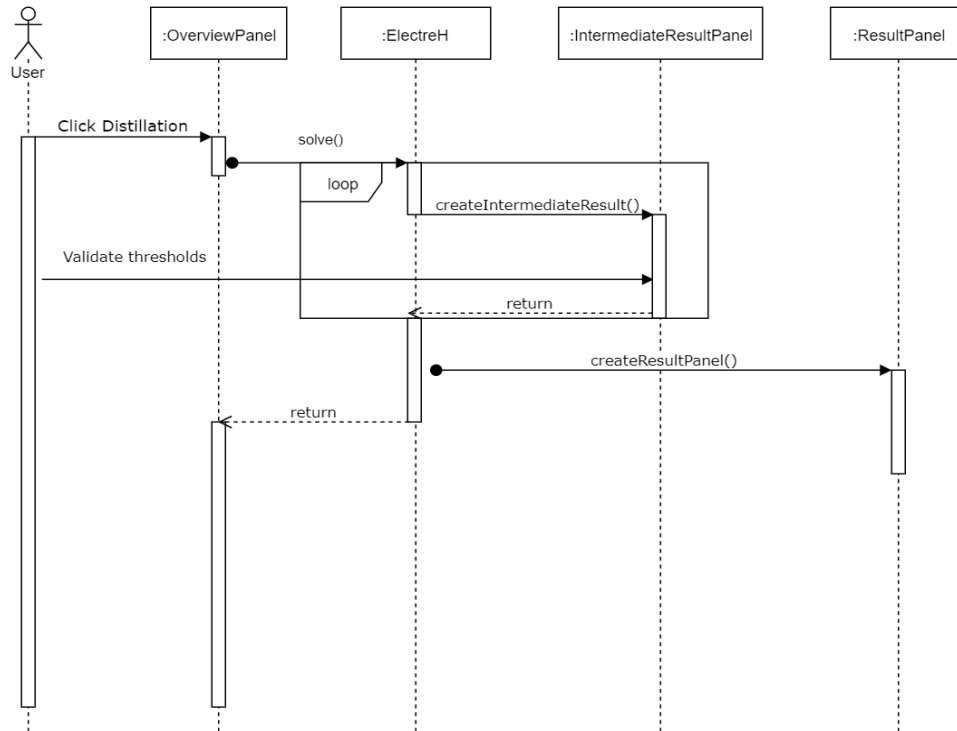


Figure 33: Solving sequence diagram

5.2 Compare preorders

The first thing done is the creation of a new panel called *NodeComparePanel*. This class has two buttons to load the two *.matrix* files which have the data to be compared, a button to show the preorders built using the *.matrix* data, a button to start the comparison, a table to show the results, and some labels with information about what had been loaded.

After creating the panel using an absolute layout setting the bounds for all the components, a new auxiliary class was created to do the comparison, *PreorderCompare*. This class should be able to compare two preorders extracted from the *.matrix* files. These files contain the results of the ascending and descending distillation, as well as the final ranking, and are created by serializing the class *RankingPreferenceRelations*, which contains this data. Deserializing these files, the *RankingPreferenceRelations* class for the two preorders to be compared could be recovered, and passed to the *PreorderCompare* class to get the comparison results.

The first thing *PreorderCompare* had to do was checking that both preorders had the same alternatives. If that was not the case, a *PreorderDistanceException* was thrown to be caught at the panel class, and shown as an error message to the user.

To allow an easy way to see the differences between a pair of preorders, instead of comparing the final position of an alternative, we compare the quartile it belongs to. The number of quartiles is determined by the total of positions in the ranking. If there are 8 or less positions only 2 quartiles are made, and otherwise 4 quartiles are made. The way of handling cases where the number of positions has a remainder when dividing it with integer division to the number of quartiles to be made, is adding the remaining positions ($total_{positions} \% total_{quartiles}$) one at a time to the quartiles, starting with the first one.

For instance, if you have a total of 5 positions, they should be partitioned in two quartiles. Each quartile should have two positions ($5 / 2 = 2$), but there is a remaining position ($5 \% 2 = 1$). This remaining position is assigned to the first quartile, ending with the first quartile

having the 3 first positions, and the second one having the 2 last positions. In case you have a total of 26 positions, each of the four quartiles will have 6 positions ($6 \times 4 = 24$) and the remaining 3 will be assigned to quartiles 1 and 2 consecutively.

To summarize, only when the final position changes from quartile, the change is considered significant enough, and it is shown in the result table.

To easily see the change of quartile in the result table, the Quartile column cells are painted depending on the quartile change. It is green or light green if the change improves the position by two or one quartile respectively and red or light red if it worsens the position by two or one quartile respectively.

This process is shown in the following sequence diagram.

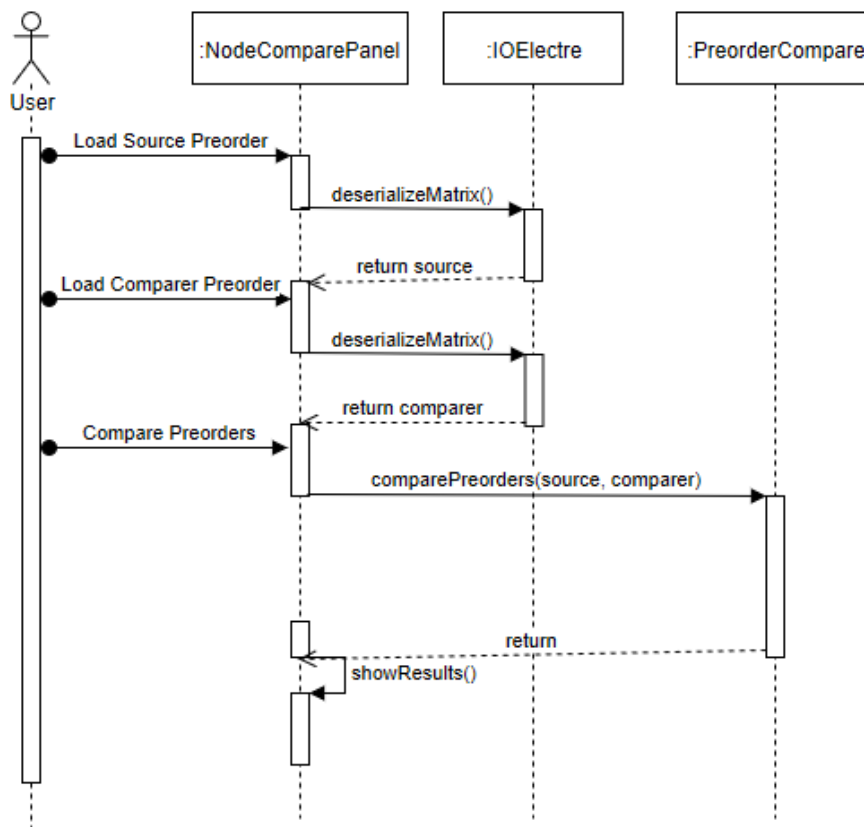


Figure 34: Compare Preorders sequence diagram

5.3 Exploratory Analysis

This panel was created from scratch in the *ExploratoryAnalysis* class. It cannot be launched until a problem has been loaded, including loading all the policies, predefined, and user defined.

It uses three layouts to create the panel. A mig layout contains the first 3 columns of the panel, all except the solve buttons. These buttons are in a box layout arranged in the vertical axis. Both layouts are in the panel layout, which is a border layout. The mig layout is on the center of the border layout, because it contains the main options, and the box layout is on the east.

To solve the problem using the different policy thresholds, a new method to solve the problem using distillation was created. The method is the same as the normal distillation one, but this one receives a list of policies extracted from *PolicyData* to be solved.

This method is computed in a new working thread. The first operation it does, is saving the current hierarchy thresholds in the *Hierarchy* class using the auxiliary class *UserPredefinedValue*, in order to recover them later. Then, for each policy to be solved, it changes the current thresholds for the ones provided by the policy, and solves the problem by using Distillation. To keep the results from the multiple solutions, *GroupRanking* was modified adding an array to save *RankingPreferenceRelations* and another array to save *RankingResultProperty* of each policy. These two classes, after the solving process finishes, would contain the results of each policy needed to calculate both indexes. Finally, the thresholds saved before starting the solving process are restored.

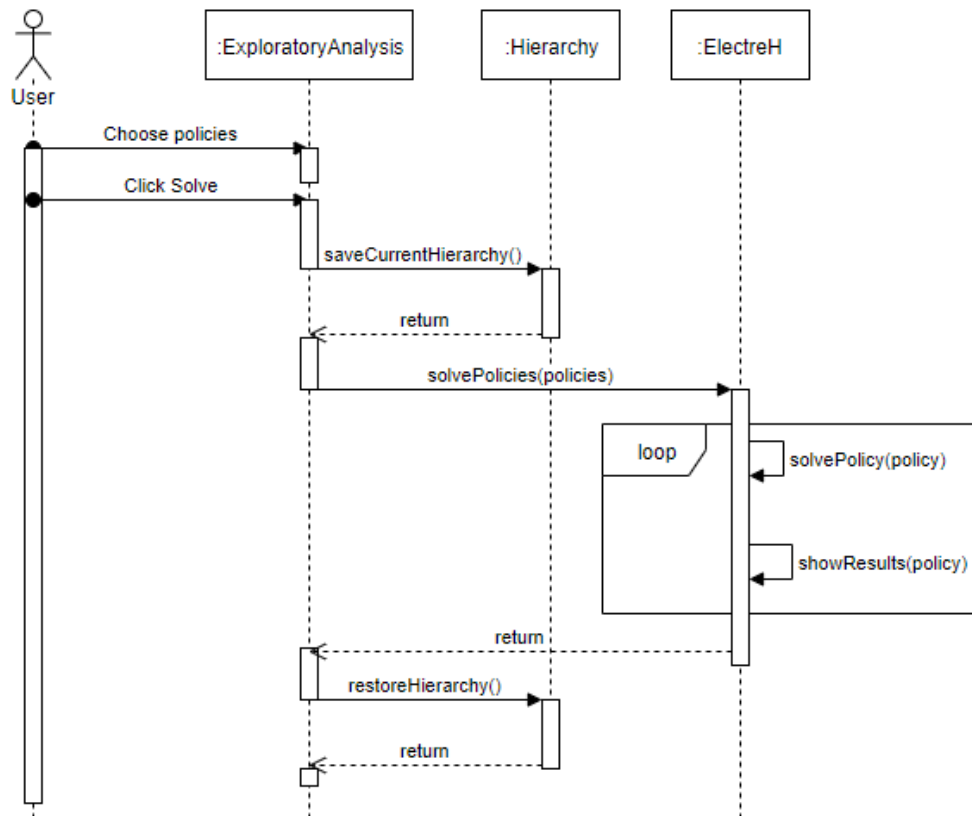


Figure 35: Automated distillation sequence diagram

To show the Acceptability and Relation index charts, two new classes per index were created. First, a class to store the data and operations needed by the chart. Acceptability index has *AlternativeQuartile*, and Relation index *AlternativePerformance*. Second, a class to create and show the chart using the auxiliary class with the data needed. *AcceptabilityStackedBarChart* for Acceptability index, and *AlternativeCompareStackedBarChart* for Relation index. Both chart classes use the JFreeChart open source framework to create the charts.

An *AlternativeQuartile* object is created for each alternative, and it contains an acceptability matrix. There are as many rows as policies exist, and each row corresponds to a policy, and always 4 columns exist. Every column represents a quartile. *ExploratoryAnalysis*, for each alternative and for each policy, gets the quartile that alternative ended the same way as

explained in 5.2. and adds 1 to the corresponding position of the matrix. For example, if the policy Very Strict is in the row 0 and the alternative ended in the first quartile, the current value of the position [0, 0] of the matrix is added the value of 1.

By reducing this matrix to just one row, we can get the total number of times the alternative ended in each quartile, which divided by the number of policies and multiplied by 100, is the percentage used in the chart. Each alternative has a bar in the chart, and each quartile is a stack of that bar.

Using the acceptability matrix and the auxiliary class *ExploratoryExcel*, all the data represented in the chart can be exported in an Excel file.

AlternativePerformance works in a similar way. It has a relation matrix, but its columns represent the relation between it and the alternative is compared. The relations are in the *PreferenceRelationEnum*, and can be of preference, indifference, incomparability or inverse preference.

The reduced matrix to only one row is also used to represent the stacks and bars of the chart the same way as the acceptability index chart.

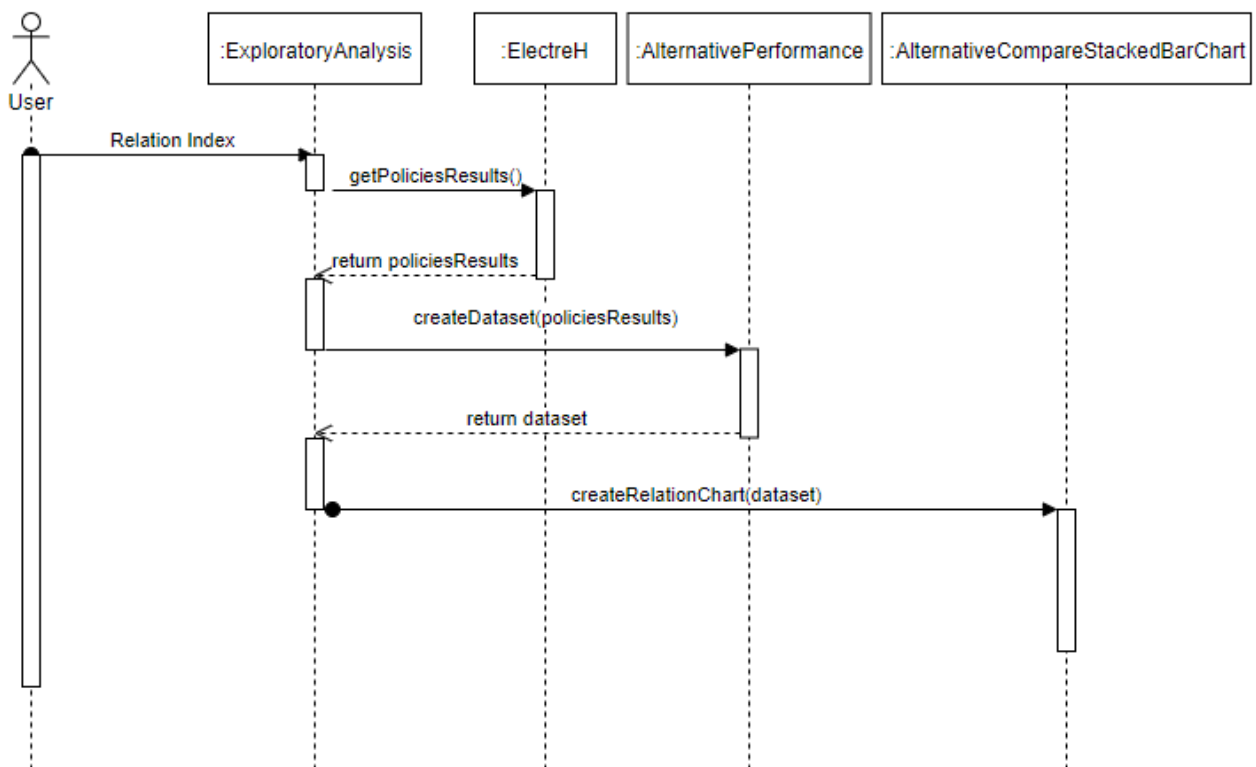


Figure 36: Relation Index sequence diagram

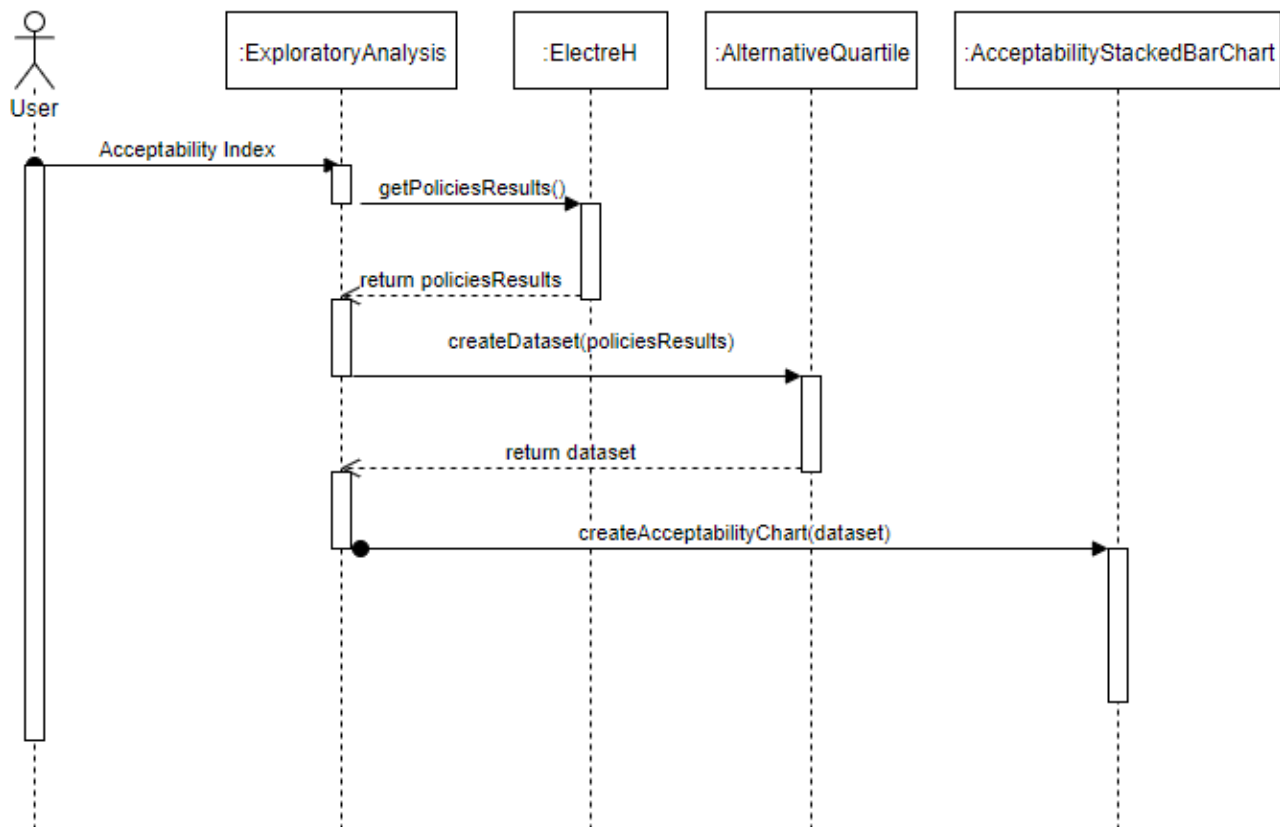


Figure 37: Acceptability Index sequence diagram

The last function, *Apply policy*, gets the thresholds for the selected policy from *PolicyData* and applies them to the selected groups. This change is reflected in the hierarchy of the main panel.

6 Evaluation

To evaluate the new tools added to the ELECTRE-H software package, a test using a real problem with a big number of alternatives and criteria is explained in this chapter.

We have focused on an important issue in the agenda of many governments: environmental sustainability. To meet the ambitious targets outlined in the United Nations 2015 Sustainable Development Goals (SDGs) and the Paris Climate Agreement, countries must integrate environmental performance metrics across a range of pollution control and natural resources targets. It is thus, of outmost importance to have decision aiding tools that permit to make an analysis of the current achievements in environmental protection on different countries, in order to detect good and bad points. This study is interesting both at a global level as well as at country level.

6.1 The dataset

The Environmental Performance Index (EPI) provides a huge number of data about 180 countries on 24 different indicators covering environmental health and ecosystem vitality. These metrics permit to see how close countries are to established environmental policy goals. The EPI thus offers a scorecard that highlights leaders and laggards in environmental performance, gives insight on best practices, and provides guidance for countries that aspire to be leaders in sustainability. [12]

The EPI is produced jointly by Yale University and Columbia University in collaboration with the World Economic Forum. The 2018 EPI was produced with generous support from the McCall MacBain Foundation and Mark T. DeAngelis.

In this work we are using the EPI values of 2018. Criteria are organized in a hierarchy, and also have a weight representing the importance of each indicator. All the countries have a score for each indicator, so the data can be formatted in the way ELECTRE-H requires it.

Before creating the problem, the criteria hierarchy provided by EPI has to be analyzed to see if all criteria are of interest to do this evaluation.

In the following table we can see the hierarchy of criteria of EPI with weights and three-letter acronyms for each indicator.

Table 4: EPI criteria

Policy Objective				Issue Category			Indicator		
Title	TLA	Weight	Title	TLA	Weight	Title	TLA	Weight	
EPI	Environmental Health	HLT	40%	Air Quality	AIR	65%	Household Solid Fuels	HAD	40%
							PM _{2.5} Exposure	PME	30%
							PM _{2.5} Exceedance	PMW	30%
				Water & Sanitation	H2O	30%	Drinking Water	UWD	50%
							Sanitation	USD	50%
				Heavy Metals	HMT	5%	Lead Exposure	PBD	100%
	Ecosystem Vitality	ECO	60%	Biodiversity & Habitat	BDH	25%	Marine Protected Areas	MPA	20%
							Biome Protection (National)	TBN	20%
							Biome Protection (Global)	TBG	20%
							Species Protection Index	SPI	20%
							Representativeness Index	PAR	10%
							Species Habitat Index	SHI	10%
							Forests	FOR	10%
				Fisheries	FSH	10%	Fish Stock Status	FSS	50%
							Regional Marine Trophic Index	MTR	50%
				Climate & Energy	CCE	30%	CO ₂ Emissions – Total	DCT	50%
							CO ₂ Emissions – Power	DPT	20%
							Methane Emissions	DMT	20%
							N ₂ O Emissions	DNT	5%
							Black Carbon Emissions	DBT	5%
				Air Pollution	APE	10%	SO ₂ Emissions	DST	50%
							NO _x Emissions	DXT	50%
				Water Resources	WRS	10%	Wastewater Treatment	WWT	100%
				Agriculture	AGR	5%	Sustainable Nitrogen Management	SNM	100%

Some intermediate groups in this hierarchy have a single sub criterion with a 100% weight. The case of interest is having multiple criteria in an intermediate group, so this intermediate groups can turn into final criteria. This happens in Heavy Metals, Forests, Water Resources and Agriculture. The other criteria do not need any modification.

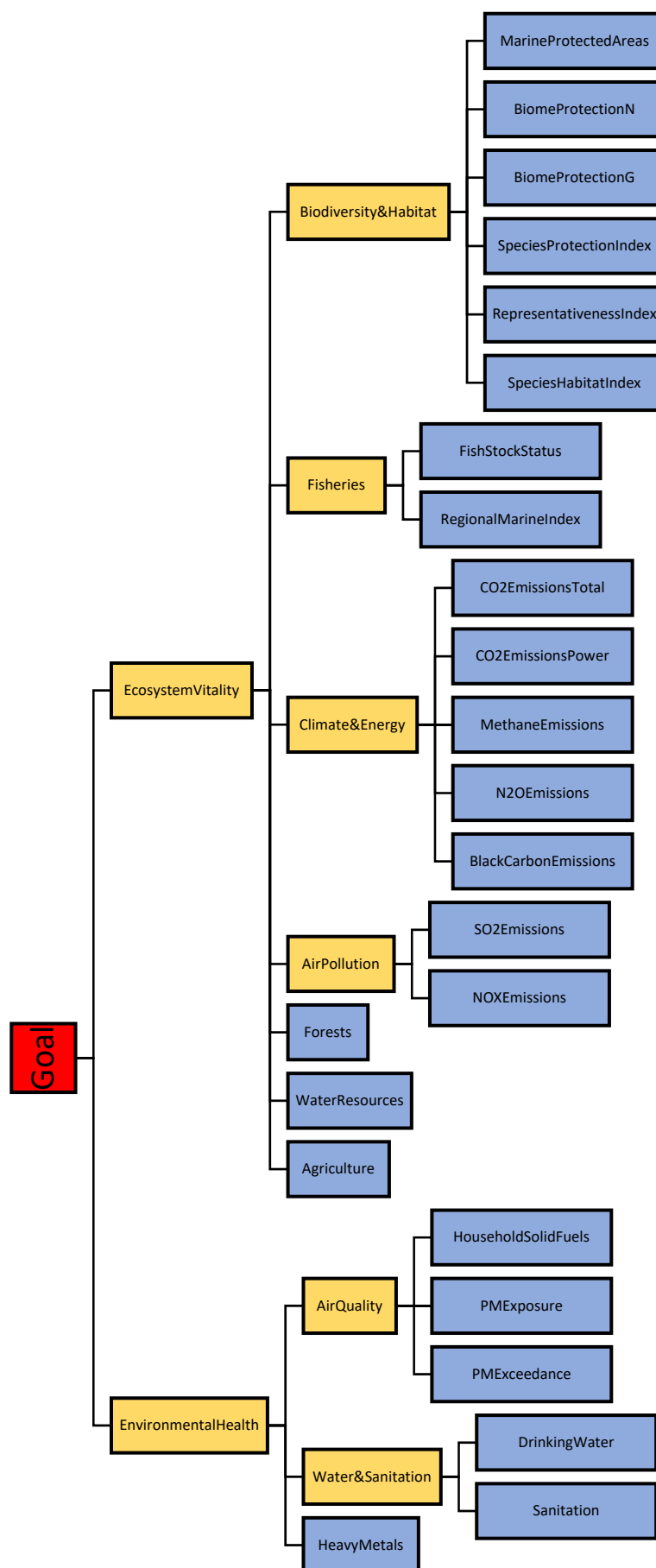


Figure 38: EPI criteria hierarchy

The complete name without blank spaces will be used as the name of intermediate groups and criteria, and the same weights will be used too. The only change to the hierarchy will be the one already explained.

In the hierarchy, you can see the root group in red, the intermediate groups in yellow, and the criteria in blue.

The other thing needed is the performance matrix values. This is, the score of each alternative for each criterion. The EPI ranking provides this score in a numerical way, in the $[0, 100]$ range, being 0 the worse score and 100 the best one. This data will be directly used specifying in the problem Excel that criteria are numerical, and that the objective is Gain to try getting the higher value possible in all criteria.

6.2 The initial model

To complete the definition of the problem, we have to define the criteria thresholds. All criteria will have the same values, $q_j = 20$, $p_j = 50$ and $v_j = 54$. All groups will also have the same initial thresholds, $q_j = 9$, $p_j = 18$ and $v_j = 20$.

The Excel sheet will have the name *Default*, so when loaded by the program we will know it is loaded as a policy named *Default*. Then, this problem will be loaded in the program, and we will test the Discrimination Policies and Exploratory Analysis, to assure they work as intended.

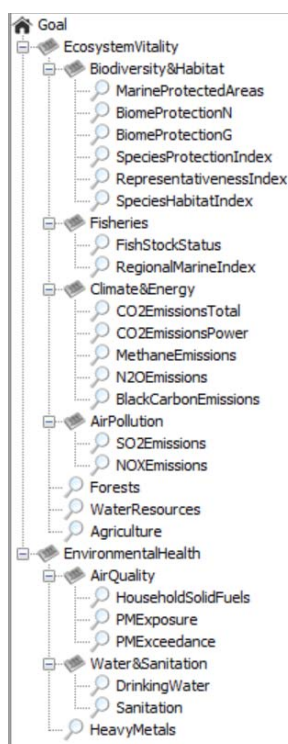
6.3 Test of the discrimination policies

First of all, the Excel with the EPI data was created as *RankingEPI2018*. Then it was loaded into the ELECTRE-H software. The data was checked in the program to assure it was correctly loaded.

In the following figures you can see the data, that corresponds to the EPI data of 2018 as explained in the previous section.

Name	Parent	Weight
Goal	*	100
EcosystemVitality	Goal	60
Biodiversity&Habitat	EcosystemVitality	25
Fisheries	EcosystemVitality	10
Climate&Energy	EcosystemVitality	30
AirPollution	EcosystemVitality	10
EnvironmentalHealth	Goal	40
AirQuality	EnvironmentalHealth	65
Water&Sanitation	EnvironmentalHealth	30

Figure 39: *RankingEPI2018* groups and weights



Name	Group	Type	Weight	Min*	Max*	Gain
HouseholdSolidFuels	AirQuality	Numerical	40			true
PMExposure	AirQuality	Numerical	30			true
PMExceedance	AirQuality	Numerical	30			true
DrinkingWater	Water&Sanitation	Numerical	50			true
Sanitation	Water&Sanitation	Numerical	50			true
HeavyMetals	EnvironmentalHealth	Numerical	5			true
MarineProtectedAreas	Biodiversity&Habitat	Numerical	20			true
BiomeProtectionN	Biodiversity&Habitat	Numerical	20			true
BiomeProtectionG	Biodiversity&Habitat	Numerical	20			true
SpeciesProtectionIndex	Biodiversity&Habitat	Numerical	20			true
RepresentativenessIndex	Biodiversity&Habitat	Numerical	10			true
SpeciesHabitatIndex	Biodiversity&Habitat	Numerical	10			true
Forests	EcosystemVitality	Numerical	10			true
FishStockStatus	Fisheries	Numerical	50			true
RegionalMarineIndex	Fisheries	Numerical	50			true
CO2EmissionsTotal	Climate&Energy	Numerical	50			true
CO2EmissionsPower	Climate&Energy	Numerical	20			true
MethaneEmissions	Climate&Energy	Numerical	20			true
N2OEmissions	Climate&Energy	Numerical	5			true
BlackCarbonEmissions	Climate&Energy	Numerical	5			true
SO2Emissions	AirPollution	Numerical	50			true
NOXEmissions	AirPollution	Numerical	50			true
WaterResources	EcosystemVitality	Numerical	10			true
Agriculture	EcosystemVitality	Numerical	5			true

Figure 40:
RankingEPI2018

Figure 41: *RankingEPI2018* criteria

Name	Household...	PMExposure	PMExceed...	DrinkingW...	Sanitation	HeavyMetals	MarineProt...	BiomeProt...	BiomeProt...	SpeciesPro...	Represent...	Spec
Afghanistan	0.0	71.78	77.11	26.62	24.88	0.0	0.0	0.61	1.26	3.07	7.0	90.6
Albania	30.95	88.9	88.08	65.97	67.15	62.89	86.85	94.46	94.07	32.09	56.94	81.81
Algeria	87.42	100.0	96.72	56.17	64.36	35.6	57.46	43.41	46.6	53.93	10.06	80.45
Angola	14.77	60.72	68.46	8.99	9.42	40.08	23.48	34.14	37.07	34.63	45.09	88.18
Antigua and Barbuda	72.82	100.0	100.0	51.96	54.69	52.04	70.01	100.0	100.0	0.0	79.39	0.0
Argentina	65.25	100.0	94.9	69.9	76.24	59.78	89.14	43.46	55.09	44.35	23.41	68.93
Armenia	36.71	60.04	69.52	65.02	58.86	69.06	0.0	99.99	100.0	58.27	27.75	92.27
Australia	100.0	100.0	100.0	96.88	100.0	68.34	100.0	83.66	77.19	89.41	58.72	87.12
Austria	86.88	77.05	81.13	98.85	90.4	91.08	0.0	100.0	100.0	77.44	94.06	84.54
Azerbaijan	30.57	53.14	64.22	51.28	45.77	63.09	0.0	50.68	61.95	83.61	16.85	95.91
Bahamas	60.47	100.0	100.0	53.67	57.13	46.81	78.71	99.41	99.42	65.69	76.36	99.39
Bahrain	73.54	62.48	65.5	59.62	66.5	45.62	83.47	15.84	15.84	0.0	2.53	0.0
Bangladesh	10.31	0.0	0.0	28.92	28.02	14.72	94.02	19.35	81.37	23.58	6.54	92.57
Barbados	100.0	100.0	100.0	54.93	56.35	60.69	45.6	6.08	6.08	0.0	74.46	0.0
Belarus	68.5	69.55	71.47	71.2	66.13	72.8	0.0	62.03	62.03	72.6	40.37	91.21
Belgium	96.74	79.83	86.6	95.93	96.26	58.6	100.0	100.0	100.0	100.0	82.9	74.09
Belize	36.82	100.0	100.0	45.42	48.06	49.86	100.0	100.0	100.0	99.88	100.0	34.99
Benin	6.23	81.35	79.22	6.62	6.13	30.23	0.0	100.0	100.0	98.42	67.73	86.21
Bhutan	24.94	41.52	48.2	34.9	30.3	30.28	0.0	100.0	100.0	99.55	74.78	96.51
Bolivia	30.12	100.0	96.22	44.18	44.96	43.51	0.0	97.22	97.06	86.0	73.36	74.69
Bosnia and Herzeg...	28.68	80.39	82.59	68.63	74.44	63.39	0.0	8.1	13.54	11.13	61.68	88.18
Botswana	24.29	52.48	59.58	22.09	21.5	49.65	0.0	100.0	100.0	95.29	100.0	95.91
Brazil	55.19	100.0	89.01	48.37	43.46	49.19	94.88	99.88	99.49	89.38	69.8	44.69
Brunei Darussalam	100.0	100.0	95.87	90.93	98.88	79.92	59.2	100.0	100.0	99.89	18.34	79.24
Bulgaria	41.64	85.44	87.33	66.18	76.35	74.15	98.4	99.87	99.79	100.0	54.48	83.33
Burkina Faso	4.63	100.0	99.99	8.71	8.19	29.78	0.0	92.32	92.32	88.94	51.69	85.45
Burundi	4.25	54.01	63.38	0.77	0.95	28.7	0.0	46.52	46.86	57.45	37.09	90.6
Côte d'Ivoire	7.72	62.51	66.9	36.09	34.43	39.84	68.07	100.0	100.0	99.04	45.6	25.44
Cabo Verde	7.42	79.1	79.47	13.7	13.2	45.84	0.0	57.64	64.0	63.44	40.3	85.15
Cambodia	100.0	100.0	97.59	100.0	89.38	91.49	82.22	100.0	100.0	71.31	44.02	80.91
Cameroon	27.03	100.0	100.0	41.42	39.9	53.47	0.0	13.26	13.26	25.56	2.69	99.85
Canada	0.0	68.91	74.2	0.0	0.0	25.01	0.0	100.0	100.0	84.15	49.99	98.48
Central African Rep...	2.52	100.0	99.36	0.38	0.25	34.67	0.0	68.09	77.36	83.64	28.36	91.21
Chad	65.17	70.97	72.86	79.81	79.41	97.15	100.0	61.65	43.29	67.31	65.19	88.18
Chile	25.85	1.05	12.44	69.66	66.82	38.02	89.28	67.65	83.05	72.21	29.46	71.81
China	44.78	100.0	97.01	56.12	63.79	60.12	88.43	82.84	78.16	84.27	77.84	68.48

Figure 42: *RankingEPI2018* performance matrix

Now, the first step will be manually changing the problem groups thresholds using the Discrimination Policies slider. *EcosystemVitality* and *EnvironmentalHealth* will be assigned

to *Neutral*. *Biodiversitat&Habitat*, *Fisheries* and *AirQuality* will be assigned to *Tolerant*, and *Climate&Energy*, *AirPollution* and *Water&Sanitation* will be assigned to *Strict*.

In the following image as an example, the group *EcosystemVitality* is changed to the *Neutral* policy. The parameters given by the policies and the Rank Order Value used to calculate them are given in Table 5.

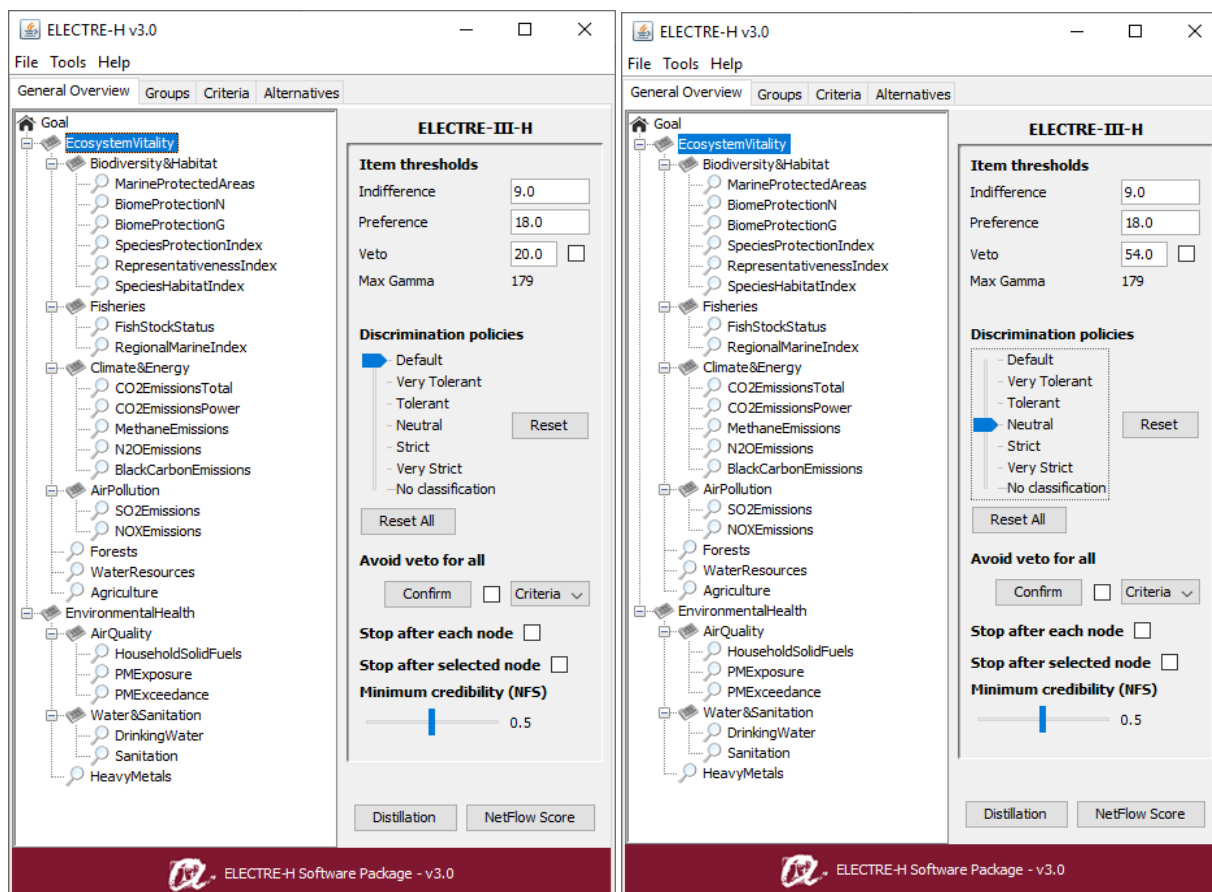


Figure 43: *RankingEPI2018* thresholds change using discrimination policies

Table 5: *RankingEPI2018* UserPolicy

Group	Policy applied	Indifference	Preference	Veto	Rank Order Value
EcosystemVitality	Neutral	9	18	54	179
Biodiversity&Habitat	Tolerant	18	36	72	179
Fisheries	Tolerant	14	27	54	135
Climate&Energy	Strict	0	9	36	179
AirPollution	Strict	0	9	36	179
EnvironmentalHealth	Neutral	9	18	54	179
AirQuality	Tolerant	18	35	71	177
Water&Sanitation	Strict	0	9	36	179

Once all parameter changes were done, they were saved as a new policy *UserPolicy*. A new sheet named *UserPolicy* was created in the *RankingEPI2018* Excel file. Because of this, the program recognized it as a policy, and loaded it.

Because the current values of the program were the same ones loaded from the policy, the discrimination policies slider automatically set itself to the *UserPolicy* policy as can be seen in the following image.

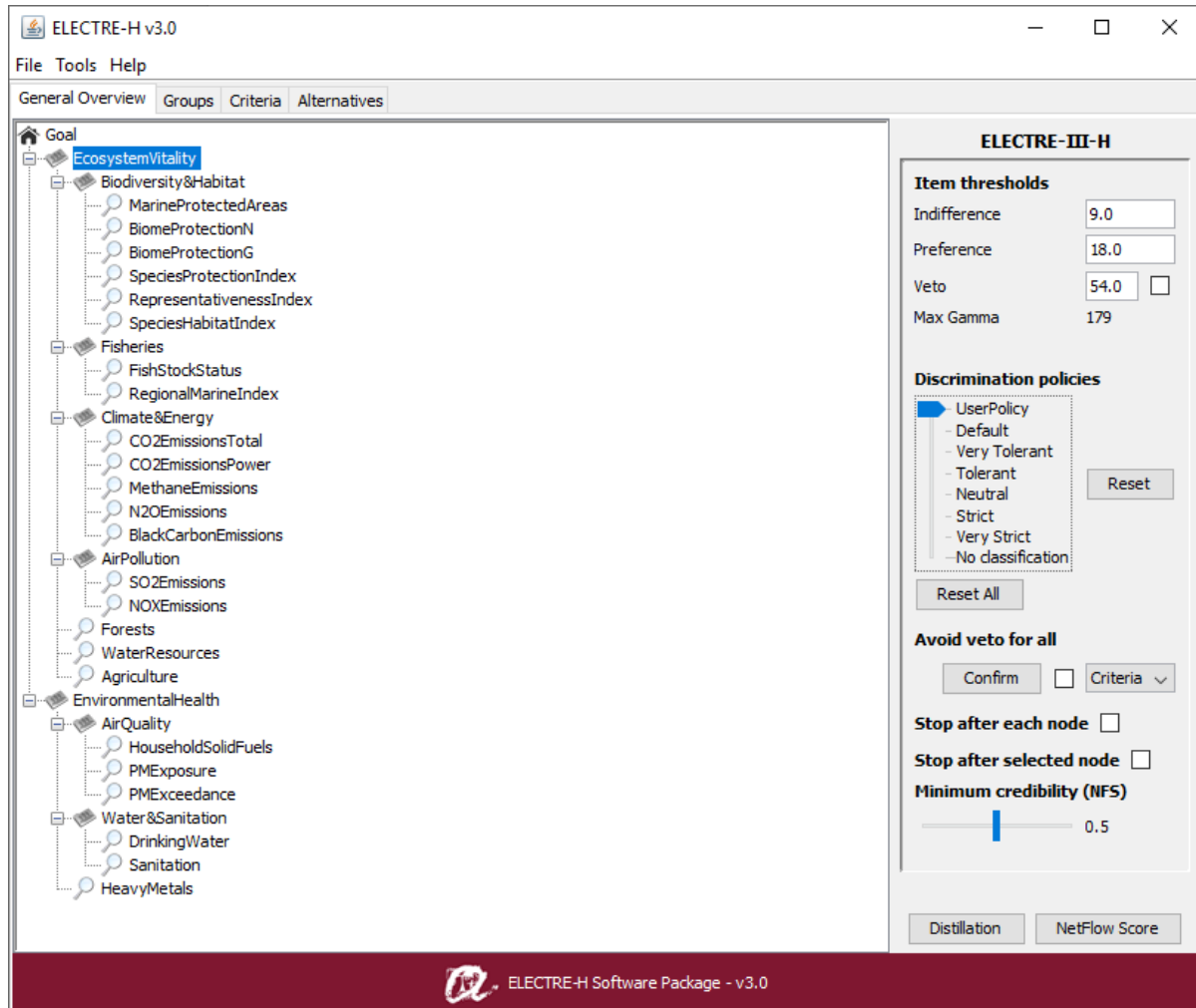


Figure 44: *RankingEPI2018* with *UserPolicy* policy

6.4 Test of the exploratory analysis

For the exploratory analysis test, the policies *Very Strict*, *Very Tolerant* and *UserPolicy* were selected to be used, and first of all were solved. The 3 results were shown in 3 result panels, and the relation matrixes were saved using the *Save relation matrix* button to be compared afterwards. In the following image, the exploratory analysis panel and the 3 policy results are shown.

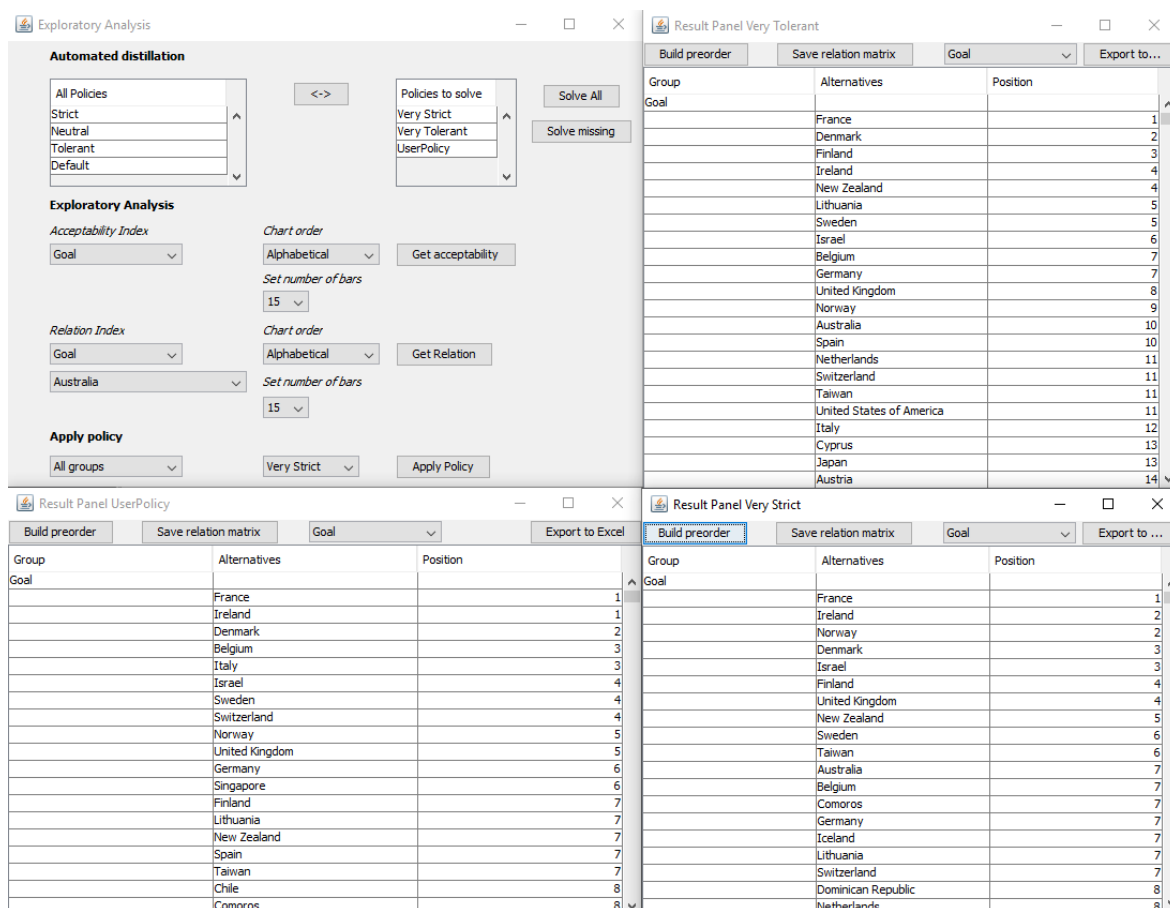


Figure 45: RankingEPI2018 exploratory analysis

Then, acceptability index chart could be created. The group selected was Goal, the chart order was alphabetical, and 15 alternatives were set to appear at a time.

The second alternative, Albania, has better results when using *UserPolicy* or *Very Strict* than when using *Very Tolerant*. It ends in the second quartile instead than in the first one.

In other cases, like Afghanistan, Azerbaijan or Belarus, the alternative always ends in the second quartile, no matter the policy applied.

Using this chart, the decision maker can easily see which policy makes each alternative end in a significant better or worse position. Knowing this, the decision maker can start the decision making process using as initial values the ones from the policy that gives results he was expecting.

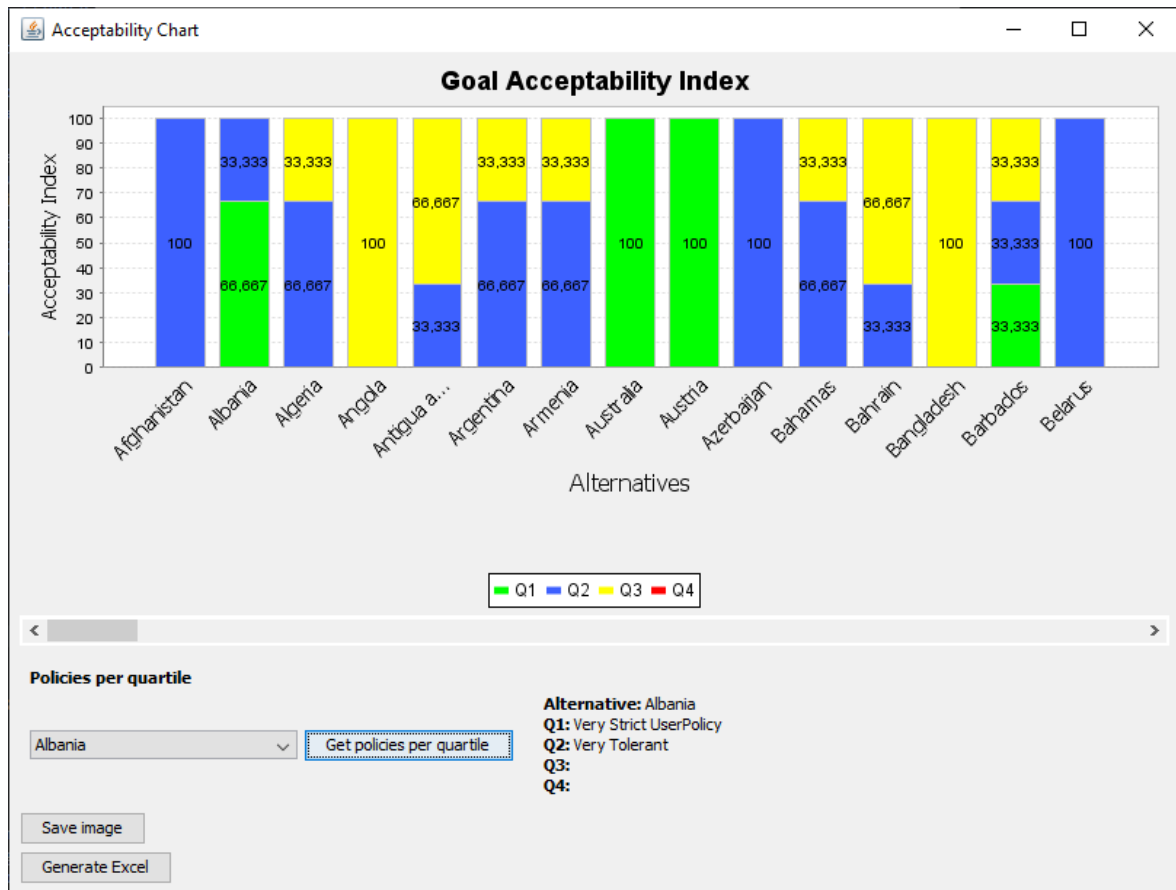


Figure 46: *RankingEPI2018* Acceptability Index

Relation Index could also be created, and was done using the same options as the acceptability index, and selecting Australia as the alternative to be compared to the other ones.

In the chart we can see how Australia is preferred over 12 of the 15 alternatives that appear in screen, no matter the policy used. Albania, Austria and Belgium are an exception. For example, when using the policy *Very Tolerant*, Australia is preferred to Austria, whereas using *Very Strict* and *UserPolicy*, it is incomparable to it.

If the decision maker does not want Australia to be incomparable to Austria, he could use the *Very Tolerant* policy parameters as a base.

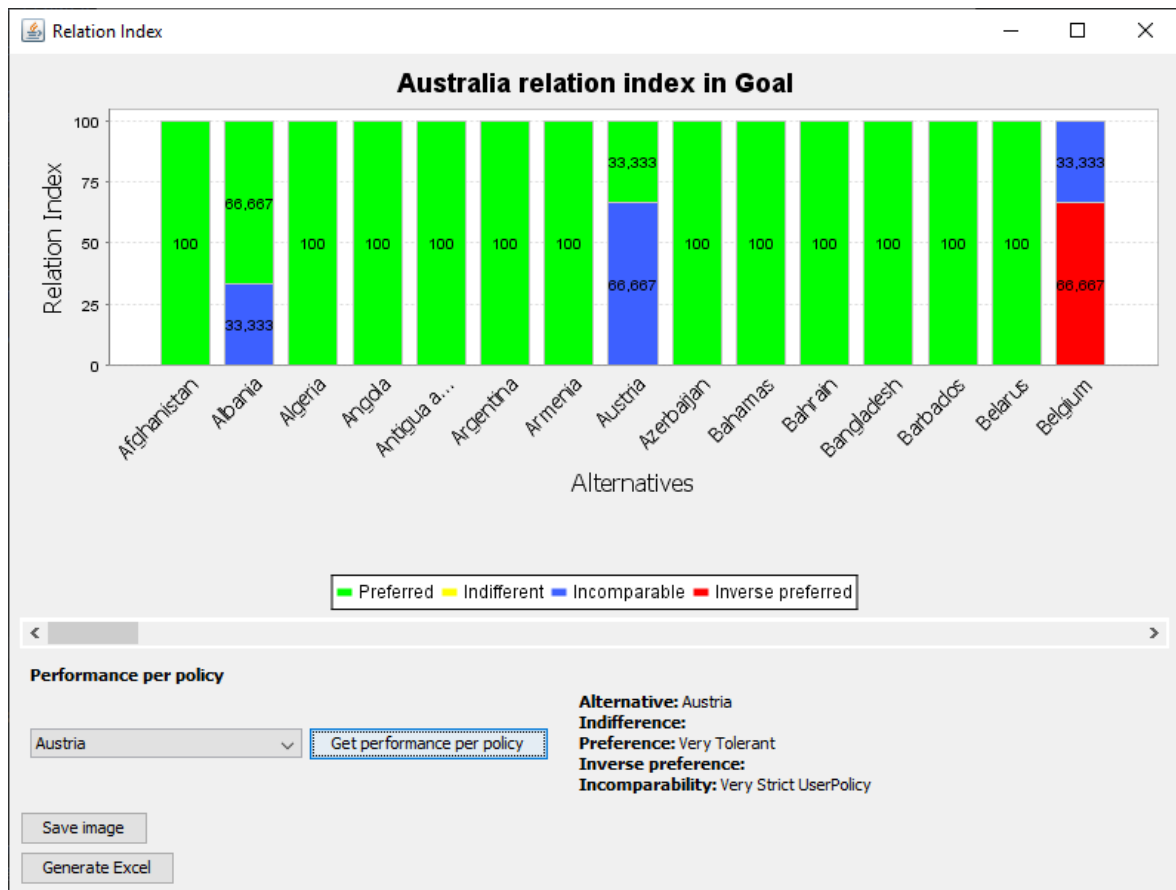


Figure 47: *RankingEPI2018* Australia Relation Index

Another Relation Index chart for the Goal group was created, but this time showing 40 alternatives in an ascending order for the country Ireland.

As can be seen with a quick look in the resulting chart in Figure 48, Ireland has very good results in the ranking. The only alternative Ireland is not preferred to in any case, is France. Moreover, is not preferred to Denmark, Finland, Norway and New Zealand only in 1 out of the 3 policies used for this analysis.

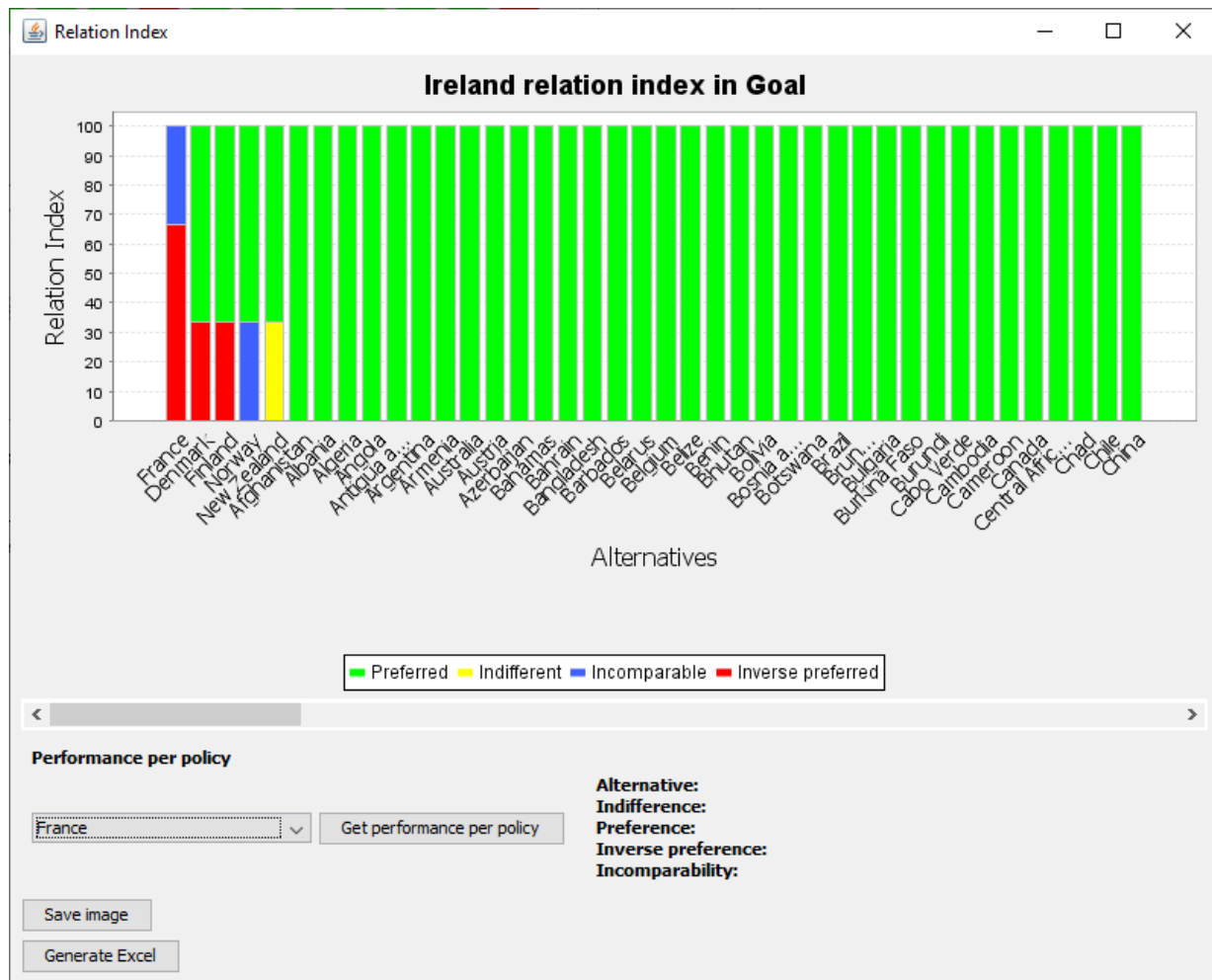


Figure 48: *RankingEPI2018* Ireland Relation Index

Finally, a last Relation Index chart was created for the Goal group showing all the 180 alternatives in a descending order for the country Kenya.

It can be seen in Figure 49 that more than half of the alternatives are preferred to Kenya no matter the policy used. Also, Kenya is preferred to about a quarter of the alternatives in all policies, and some alternatives have different relations to Kenya depending on the policy used.

If the decision maker wanted to know the results for a particular alternative, he could get this information by using the *Get performance per policy* button. Alternatively, if he wanted to have all this information, he could extract the data used to create the chart in an Excel file.

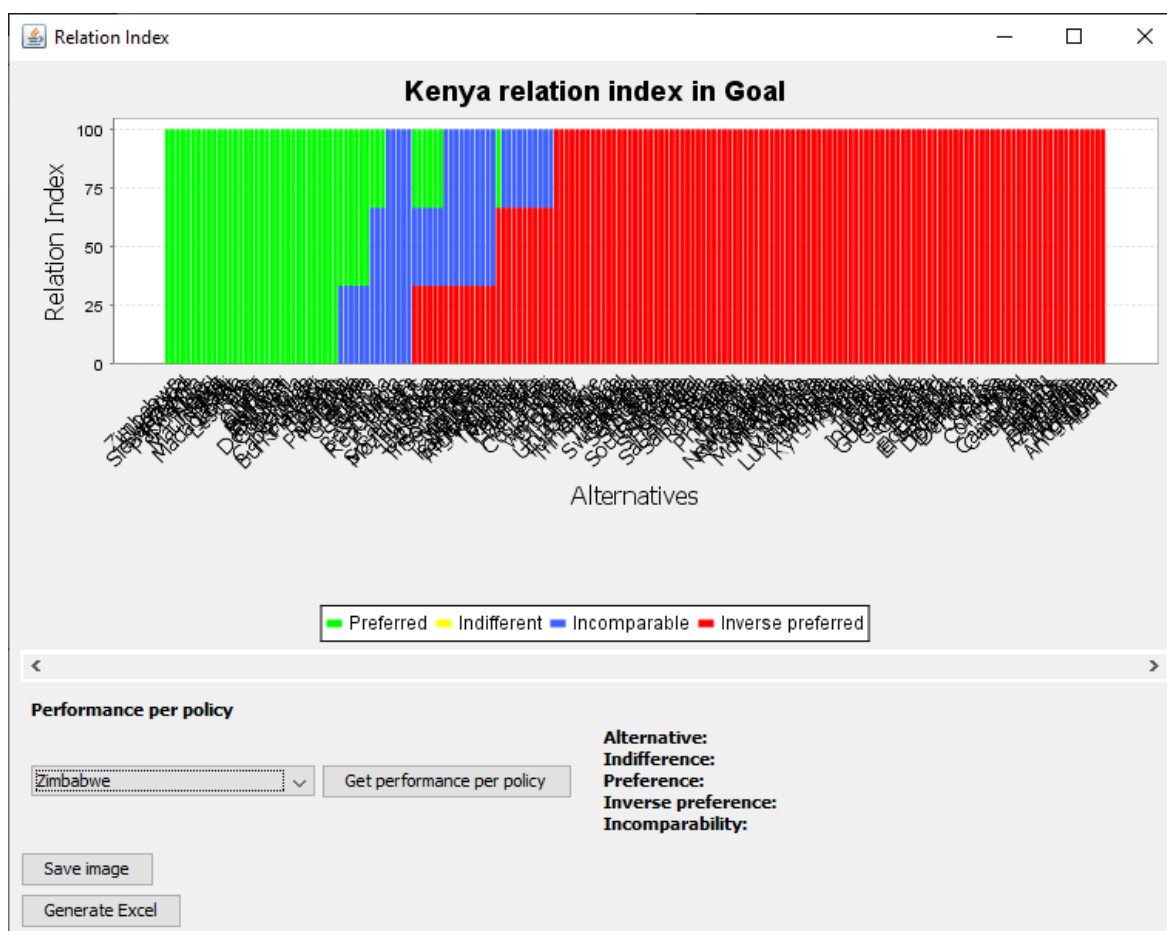


Figure 49: *RankingEPI2018* Kenya Relation Index

In both kinds of charts, the section of the chart that is seen on screen could be saved as an image file, and also all the data used to create the chart could be exported as an Excel file.

In Figure 50 you can see an example of how the data is exported onto the Excel file. In both indices, the first column has all the alternatives, except in Relation Index that the alternative being compared to the other ones is not shown. In addition, there is a column for each policy used to create the charts. In this example, the policies used were Very Strict, Vert Tolerant and UserPolicy.

In Acceptability Index, for each alternative and each policy, the quartile in which the alternative ended is written as Q followed by the quartile (Q1, Q2, Q3 or Q4). By contrast, in Relation Index, the relation of the alternative selected to be compared, and the other ones, is written for each policy. This relation can be preference (P), indifference (I), incomparability (R) or inverse preference (P⁻). In this example, Australia was selected to be compared.

	A	B	C	D		A	B	C	D
1	Alternative/Policy	Very Strict	Very Tolerant	UserPolicy	1	Alternative/Policy	Very Strict	Very Tolerant	UserPolicy
2	Afghanistan	Q2	Q2	Q2	2	Afghanistan	P	P	P
3	Albania	Q1	Q2	Q1	3	Albania	P	P	R
4	Algeria	Q2	Q3	Q2	4	Algeria	P	P	P
5	Angola	Q3	Q3	Q3	5	Angola	P	P	P
6	Antigua and Barbuda	Q2	Q3	Q3	6	Antigua and Barbuda	P	P	P
7	Argentina	Q2	Q3	Q2	7	Argentina	P	P	P
8	Armenia	Q3	Q2	Q2	8	Armenia	P	P	P
9	Australia	Q1	Q1	Q1	9	Austria	R	P	R
10	Austria	Q1	Q1	Q1	10	Azerbaijan	P	P	P
11	Azerbaijan	Q2	Q2	Q2	11	Bahamas	P	P	P
12	Bahamas	Q2	Q2	Q3	12	Bahrain	P	P	P
13	Bahrain	Q2	Q3	Q3	13	Bangladesh	P	P	P
14	Bangladesh	Q3	Q3	Q3	14	Barbados	P	P	P
15	Barbados	Q1	Q3	Q2	15	Belarus	P	P	P
16	Belarus	Q2	Q2	Q2	16	Belgium	R	P-	P-
17	Belgium	Q1	Q1	Q1	17	Belize	P	P	P
18	Belize	Q3	Q3	Q3	18	Benin	P	P	P
19	Benin	Q4	Q4	Q4	19	Bhutan	P	P	P
20	Bhutan	Q3	Q4	Q3	20	Bolivia	P	P	P
21	Bolivia	Q3	Q3	Q3	21	Bosnia and Herzegovina	P	P	P
22	Bosnia and Herzegovina	Q3	Q3	Q3	22	Botswana	P	P	P
23	Botswana	Q3	Q3	Q3	23	Brazil	P	P	P
24	Brazil	Q2	Q2	Q2	24	Brunei Darussalam	P	P	P

Figure 50: *RankingEPI2018* Acceptability (left) and Relation (right) Indexes

6.5 Test of comparing preorders

After performing this exploratory analysis, let us suppose the decision maker has interest in using *Very Tolerant* or *Very Strict* policy for solving the problem. As relation matrices for the solutions provided in the group Goal for both policies were saved, *Compare Preorders* panel could be opened to compare them, and ease the decision maker choose which one fits best his needs.

The first thing done was loading both .matrix files. The *Very Strict* as the first one, and *Very Tolerant* as the second one. By comparing them, the first we saw was that *Very Strict* had a total of 60 positions, and by contrast *Very Tolerant* had 71 as it can be seen in Figure 51. This means that more alternatives are indifferent or incomparable in *Very Strict* than in *Very Tolerant*. Also, the graph is flatter in the first preorder than in the second one.

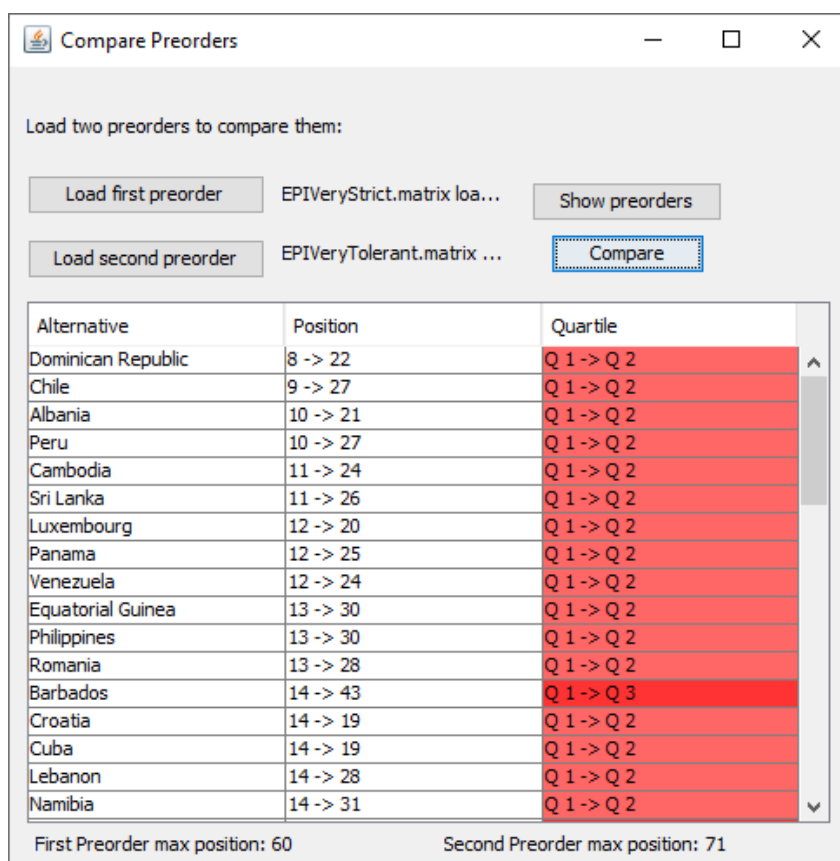


Figure 51: Compare *EPIRanking2018 Very Strict* and *Very Tolerant* preorders

This could be easily seen by showing both preorders as can be seen in Figure 52.

At this point, the decision maker could choose which kind of hierarchy he liked the most.

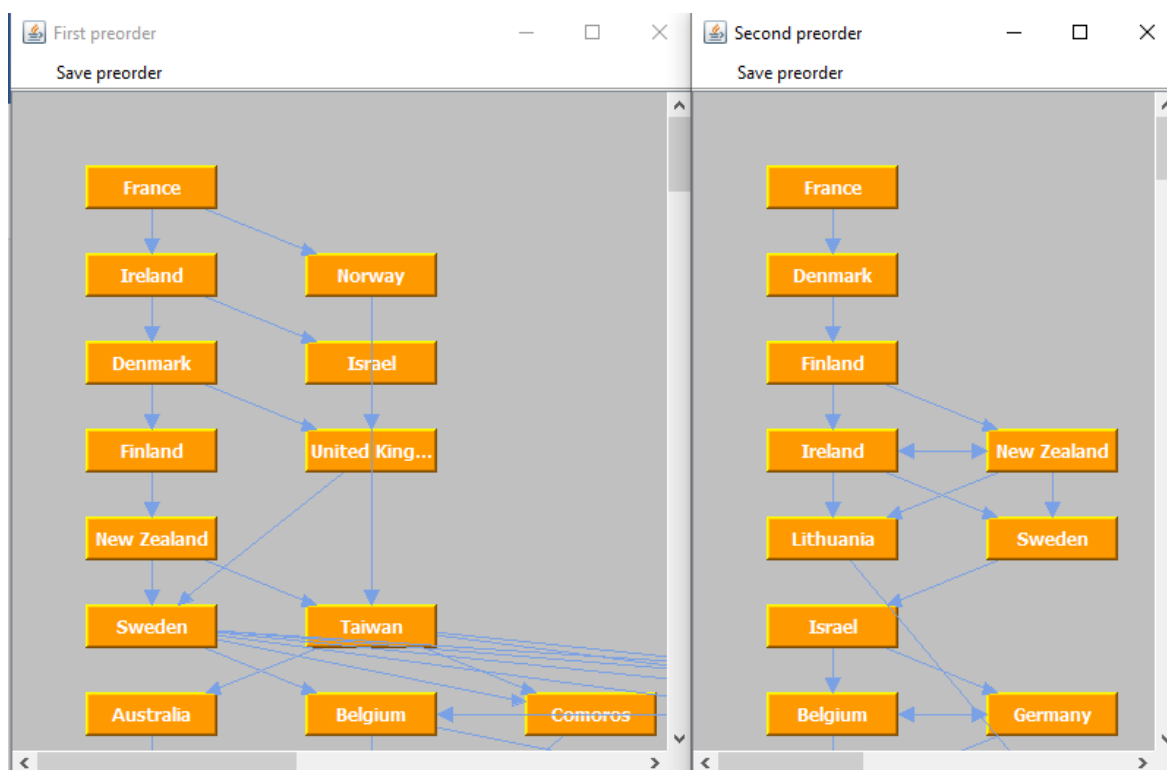


Figure 52: *EPIRanking2018 Very Strict* and *Very Tolerant* built preorders

Finally, the decision maker can look at the alternatives and their change in position or quartile. With a quick look at the table, was easy to see that most alternatives that changed at least from one quartile, had a negative change.

For example, Luxembourg, Panama and Venezuela in *Very Strict* all had the same position in the first quartile, 11. By contrast, in *Very Tolerant*, they are not in the same position, each one had a different position, but in the second quartile.

If the decision maker is interested in having an alternative in a better or worse position, he can see the difference of the desired alternative easily in the compare preorders table.

Scrolling the table down, as can be seen in Figure 53, two alternatives can be seen with a positive change, Iran and Georgia. Even though Iran maintained the same final position, 17, it changed from the second quartile to the first one. This is due to the difference in the maximum ranking position each preorder has. In the first preorder it has the position 17 out of 60, but in the second one it has 17 out of 71.

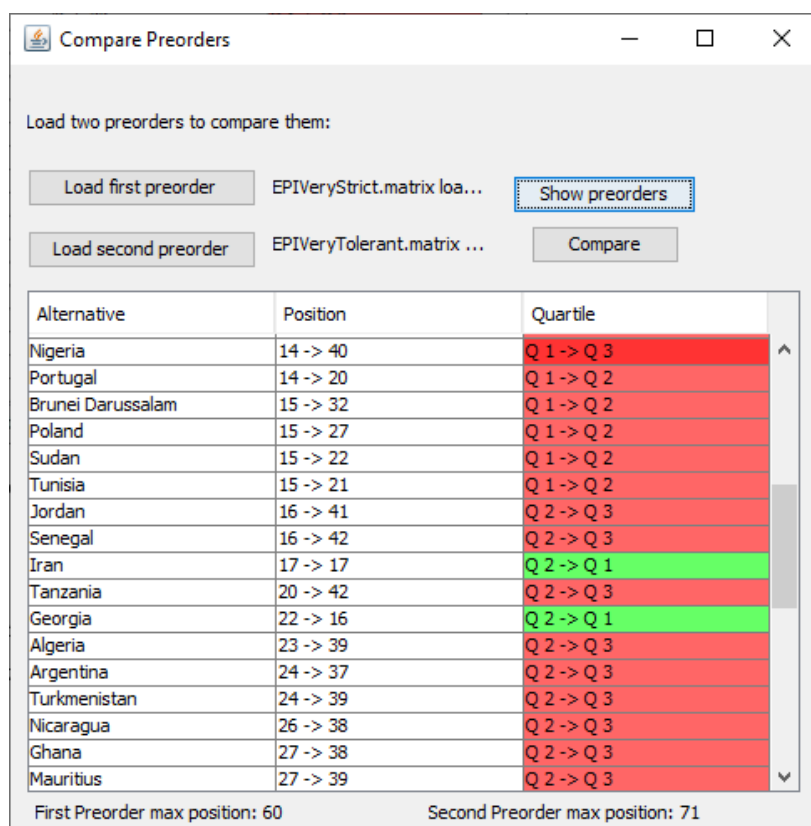


Figure 53: Compare *EPIRanking2018 Very Strict* and *Very Tolerant* preorders

6.6 Analysis of one case: Georgia

A more extensive analysis of a determined alternative can be performed using the Compare Preorders panel. As a prerequisite, the relation matrix for each group has to be saved, in order to be compared.

In this example, Georgia was chosen to be analyzed when applying *Very Strict* or *Very Tolerant* policies. The lower level groups always have the same ranking because the group parameters are applied when solving the upper levels, so the group relation matrices needed were *EcosystemVitality*, *EnvironmentalHealth* and *Goal*.

The first group loaded in the Compare Preorders panel was *EcosystemVitality*, as can be seen in Figure 54. In this group, using *Very Strict* configuration, Georgia gets the 16th position, whereas using *Very Tolerant*, it gets the 5th one.

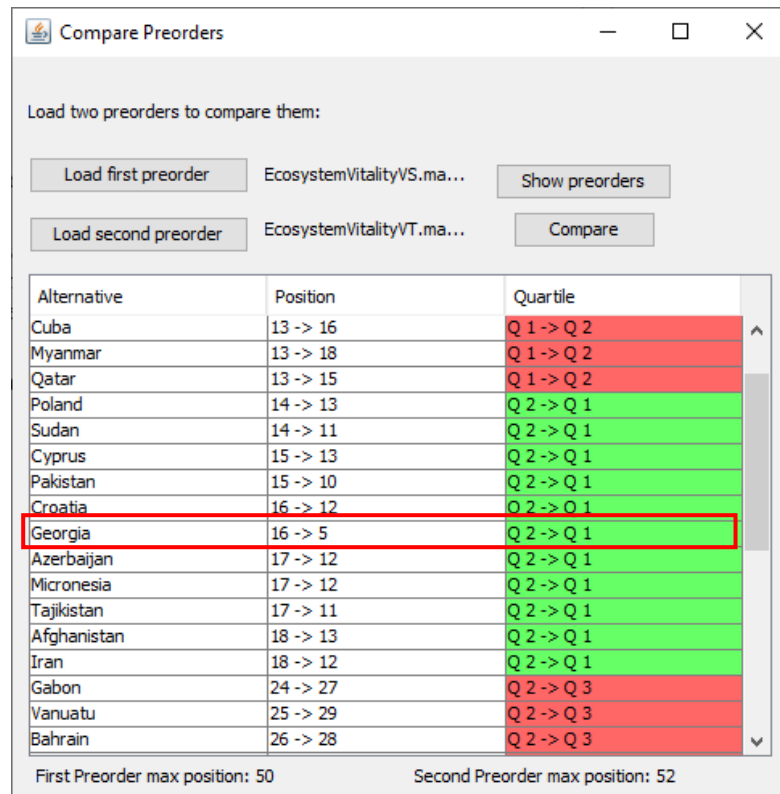


Figure 54: *EcosystemVitality* preorders comparison

For a better analysis, the preorders can be shown (Figure 55). Using *Very Strict* (left graph) much more alternatives have the same ranking position than using *Very Tolerant* (right graph). We can see a lot of incomparabilities in both cases.

For Ecosystem Vitality in the *Very Strict* setting, we can see a ranking (branch) formed by Spain > Bulgaria > South Korea > Greece > Georgia. We can also see that Georgia is outranks to Azerbaijan and Iceland, and transitively to Afghanistan and Iran.

However, for the *Very Tolerant* setting, Georgia does not outrank any other country. We find the ranking formed by France > Taiwan > Peru > Chile > Georgia. Therefore, Georgia is incomparable to all the rest of countries. It does not seem a good situation in order to extract conclusions, probably because the parameters have led to a too uncertain model with low degree of veto.

The *Very Strict* configuration seems better than the *Very Tolerant* one.

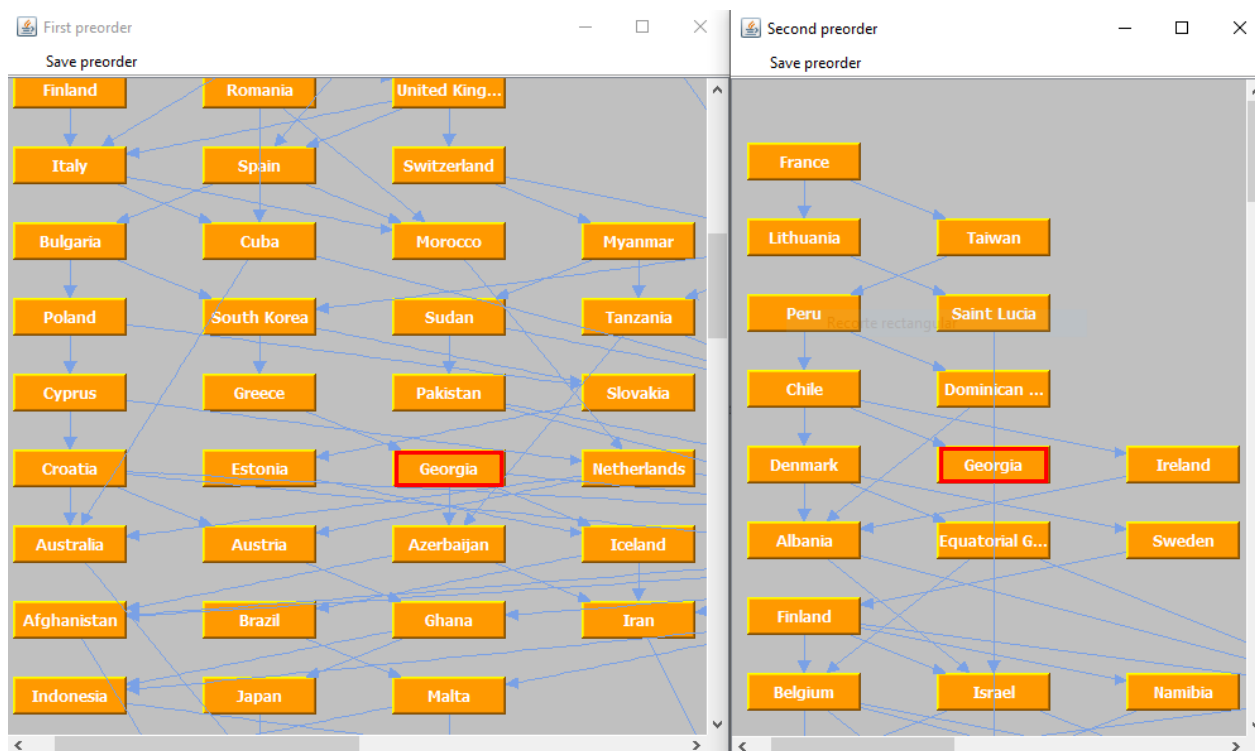


Figure 55: *EcosystemVitality* Very Strict and Very Tolerant preorders

Then, *EnvironmentalHealth* matrices could be loaded. Both policies have worse results than the ones in the other group. *Very Strict* gets the 34th position, and *Very Tolerant* the 46th.

Compare Preorders

Load two preorders to compare them:

Load first preorder: EnvironmentalHealthVS.... Show preorders

Load second preorder: EnvironmentalHealthVT.... Compare

Alternative	Position	Quartile
Bulgaria	19 -> 21	Q 2 -> Q 1
Oman	19 -> 21	Q 2 -> Q 1
Comoros	20 -> 17	Q 2 -> Q 1
Seychelles	20 -> 21	Q 2 -> Q 1
Turkey	20 -> 16	Q 2 -> Q 1
Lithuania	22 -> 18	Q 2 -> Q 1
Chile	30 -> 47	Q 2 -> Q 3
Uzbekistan	31 -> 47	Q 2 -> Q 3
Georgia	34 -> 46	Q 2 -> Q 3
Romania	34 -> 46	Q 2 -> Q 3
Armenia	35 -> 48	Q 2 -> Q 3
Saint Vincent and the Gren...	36 -> 42	Q 3 -> Q 2
Fiji	39 -> 41	Q 3 -> Q 2
Kenya	51 -> 64	Q 3 -> Q 4
Liberia	54 -> 62	Q 4 -> Q 3
Tanzania	54 -> 60	Q 4 -> Q 3
Guinea-Bissau	55 -> 61	Q 4 -> Q 3
Madagascar	55 -> 62	Q 4 -> Q 3

First Preorder max position: 69 Second Preorder max position: 83

Figure 56: *EnvironmentalHealth* preorders comparison

In the built preorders, *Very Strict* leads to having much more alternatives with similar ranking positions than using *Very Tolerant*. It can be seen that Chile that is preferred to Georgia when using *Very Tolerant*, and it is inversely preferred when using *Very Strict*. This illustrates the importance of analyzing also the partial preorder graphs.

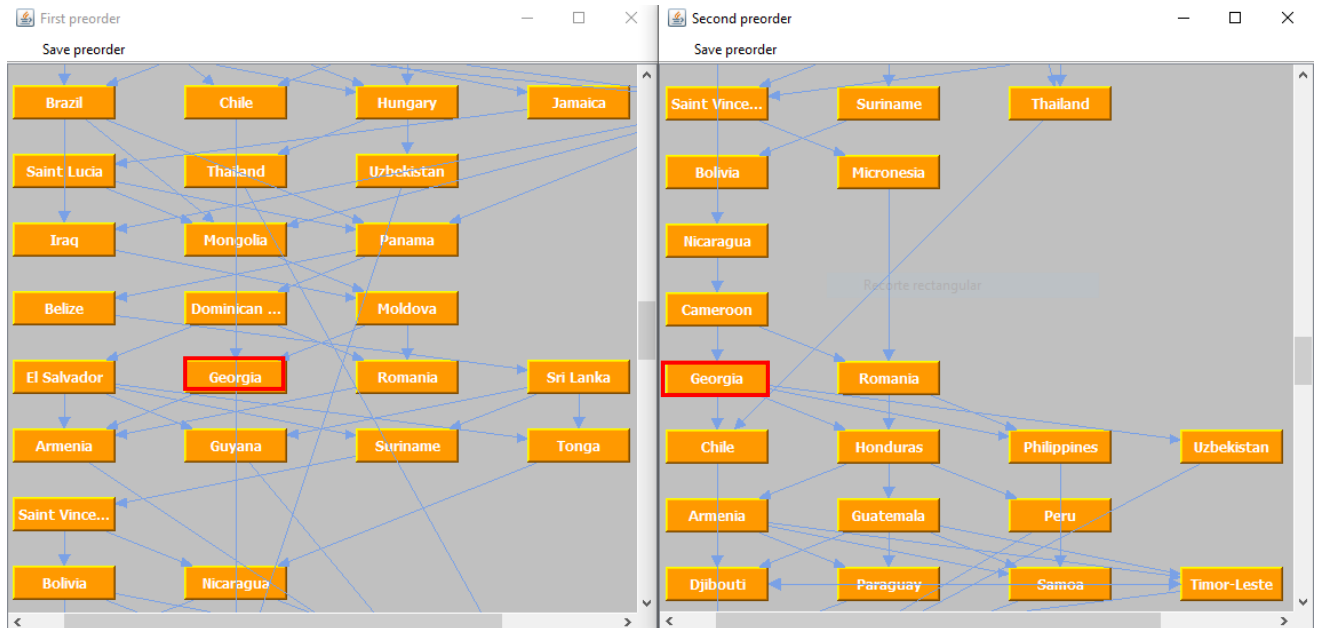


Figure 57: *EnvironmentalHealth Very Strict and Very Tolerant preorders*

The preorder comparison of *Goal* has already been seen in Figure 53, with a change from Q2 to Q1. Now in Figure 58 you can see the preorders built at the goal level

In *Very Strict* as already explained, the hierarchy is very flat. Georgia is preferred to some other alternatives, as for example Algeria, Ecuador, Egypt, Malaysia or Montenegro, whereas in *Very Tolerant* is preferred to Iran. So, we can conclude that the outranking relations that generate the ranking are very sensitive to the parameters used in the different levels.

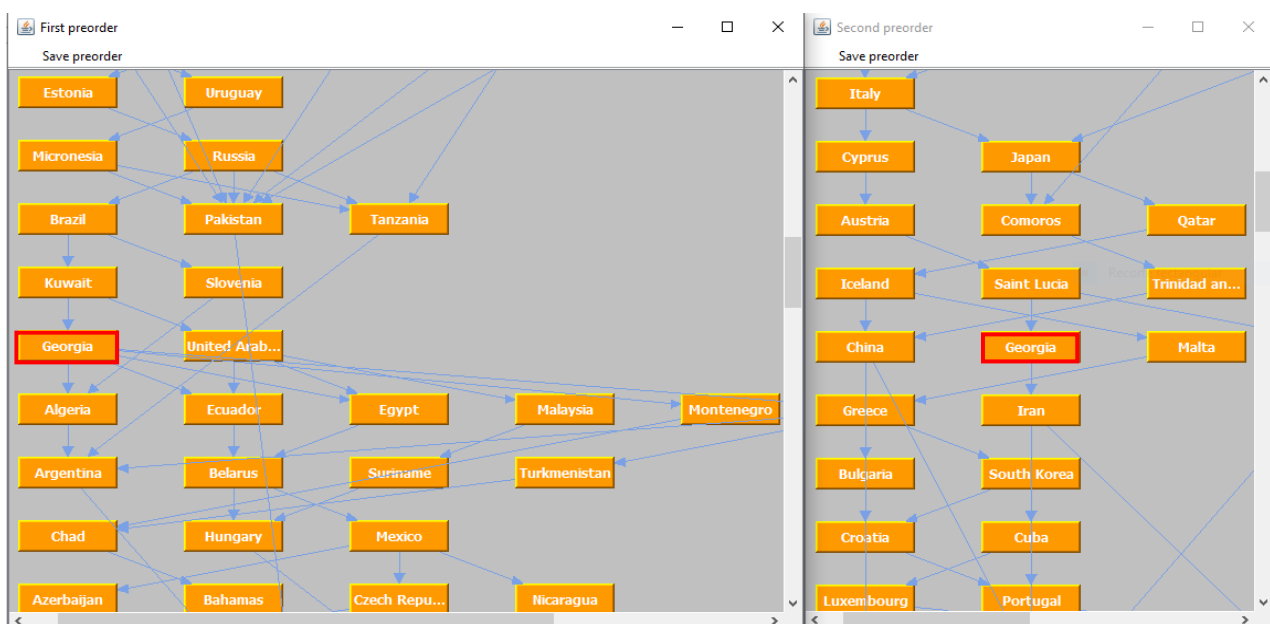


Figure 58: *Goal Very Strict and Very Tolerant preorders*

If the decision maker wanted to do a deeper analysis of the relation between Georgia and other countries when using a certain policy, Relation Index could be used.

Let's use the Relation Index of Georgia to compare Georgia to Iran in the *EcosystemVitality* (Figure 59), *EnvironmentalHealth* (Figure 60) and *Goal* (Figure 61) groups.

As can be seen, Georgia has different relations to Iran depending on the criteria being evaluated. In *EcosystemVitality*, Georgia is always preferred to Iran, whereas on *EnvironmentalHealth* is always inverse preferred. In *Goal*, Georgia is preferred to Iran if using the *Very Tolerant* policy, but incomparable if using the *Very Strict* one. A conclusion the decision maker could get to, is that to get a better result in the final ranking, the environmental health index of the country should be better.

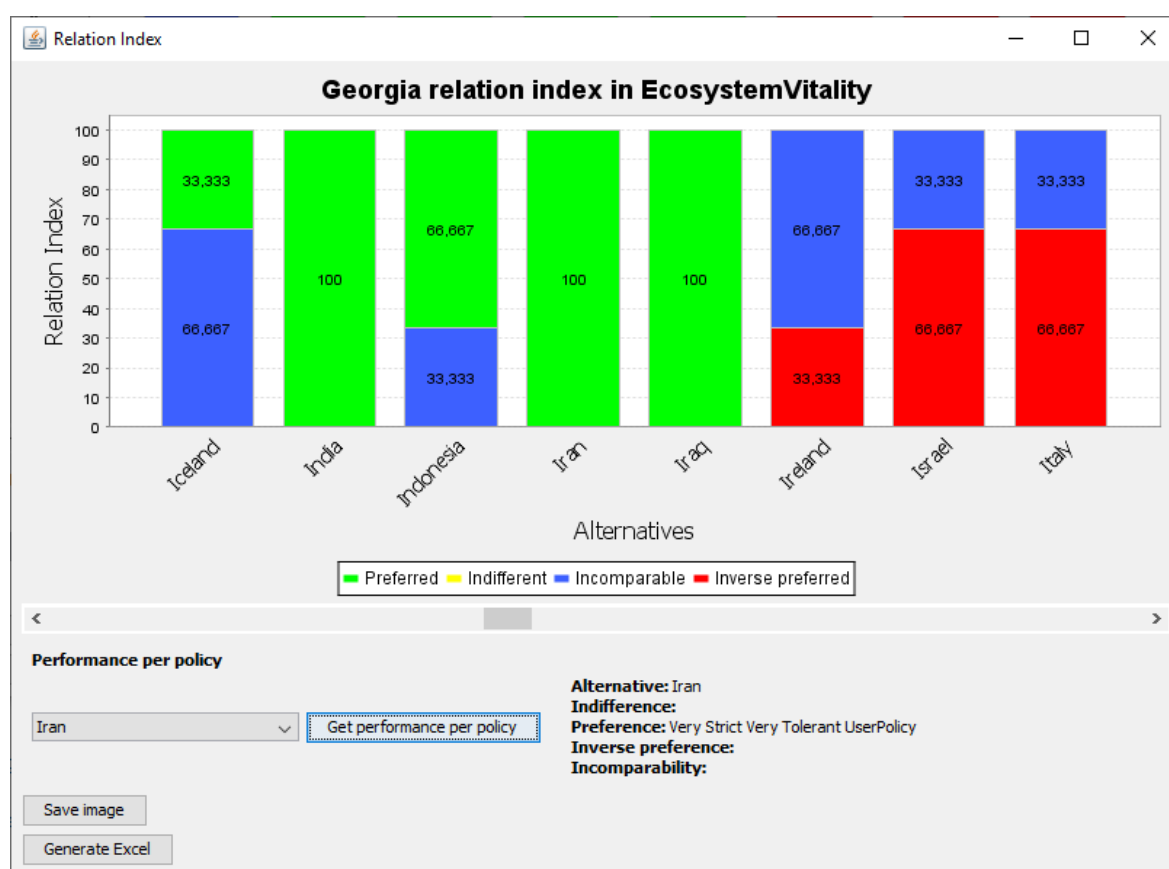


Figure 59: Georgia Relation Index in *EcosystemVitality* compared to Iran

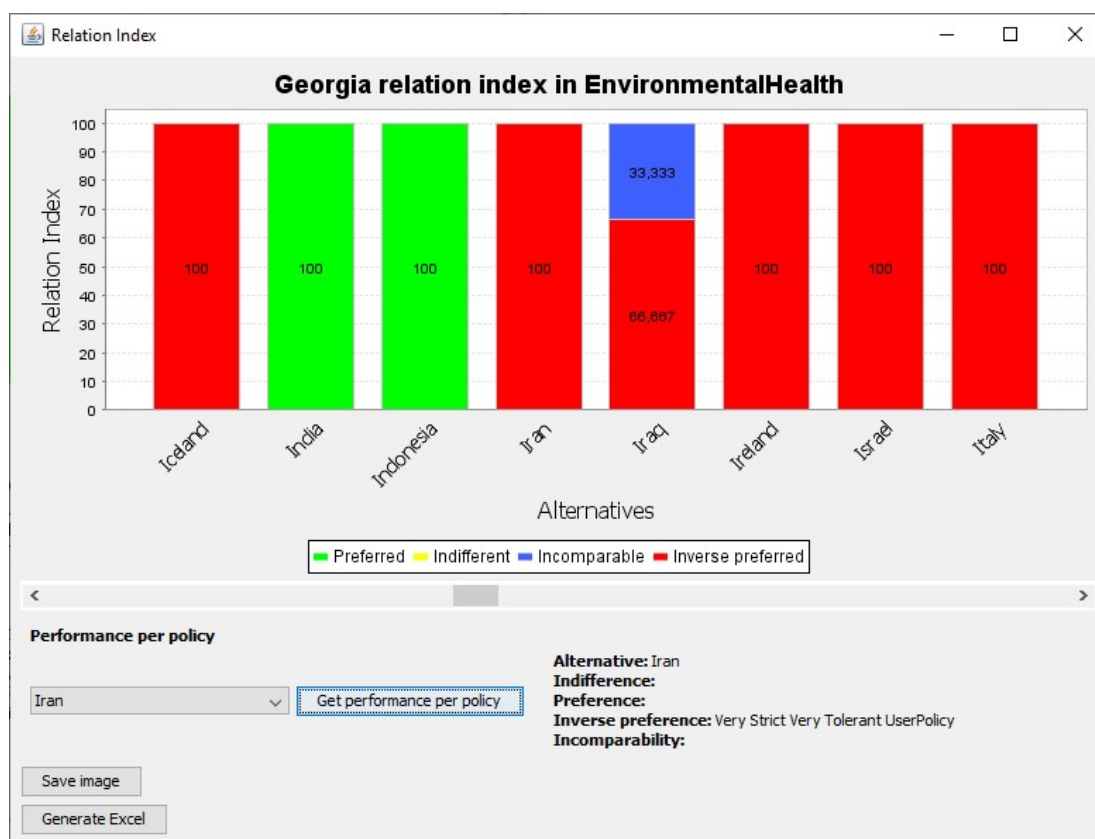


Figure 60: Georgia Relation Index in *EnvironmentalHealth* compared to Iran

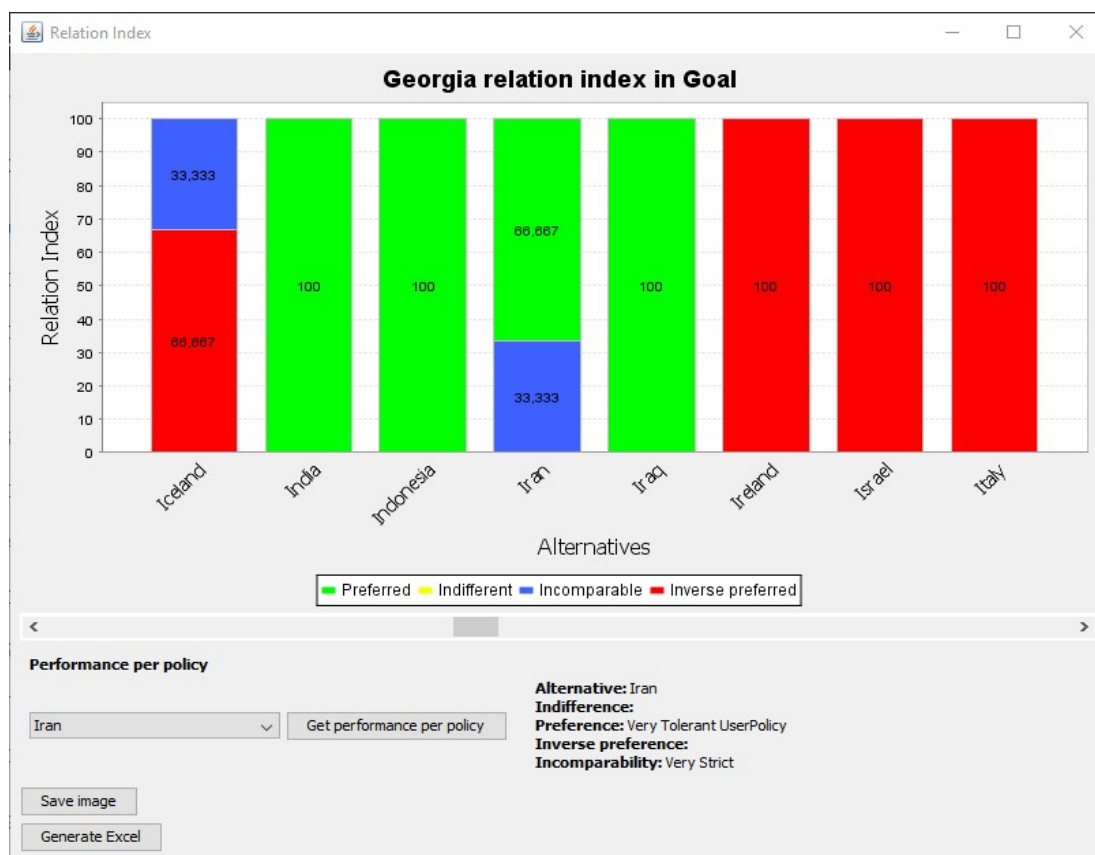


Figure 61: Georgia Relation Index in *Goal* compared to Iran

Now, let's do the same comparison, but with another country, Chile, by analyzing the results shown in figures 62, 63 and 64. In this example, Georgia in *EcosystemVitality* for both *Very Strict* and *Very Tolerant* policies, gets inverse preference compared to Chile. In *EnvironmentalHealth*, is preferred with *Very Tolerant* and incomparable with *Very Strict*. Finally, in *Goal* is preferred using *Very Tolerant* and inverse preferred to Chile using *Very Strict*. As a conclusion, the decision maker can see that to get better results compared to Chile, Georgia should prioritize getting a better performance in *EcosystemVitality*.

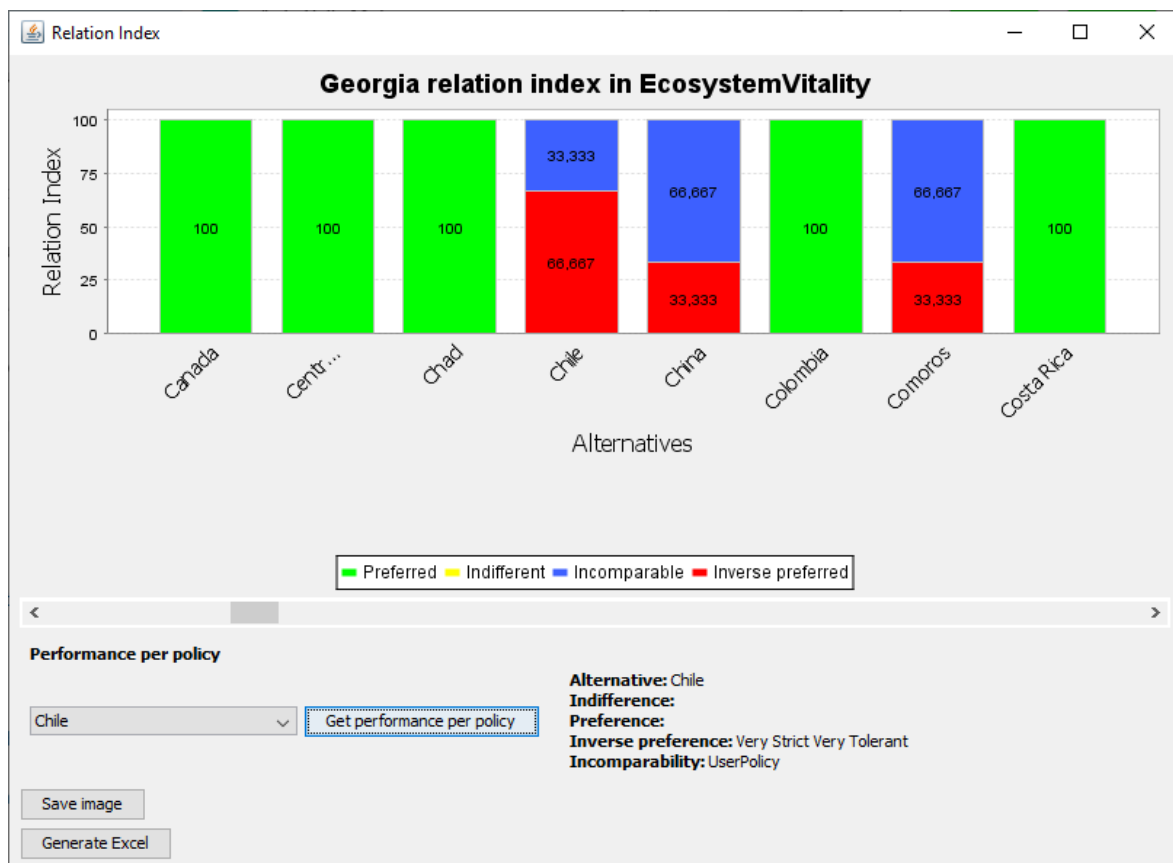


Figure 62: Georgia Relation Index in *EcosystemVitality* compared to Chile

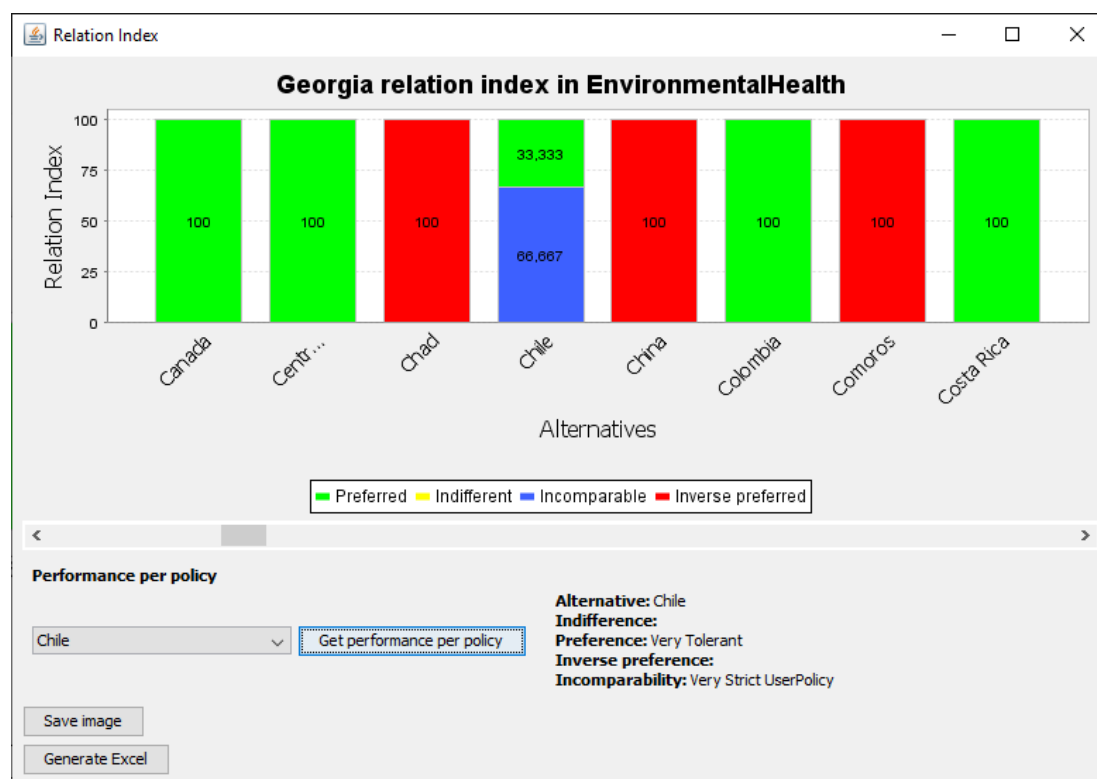


Figure 63: Georgia Relation Index in *EnvironmentalHealth* compared to Chile

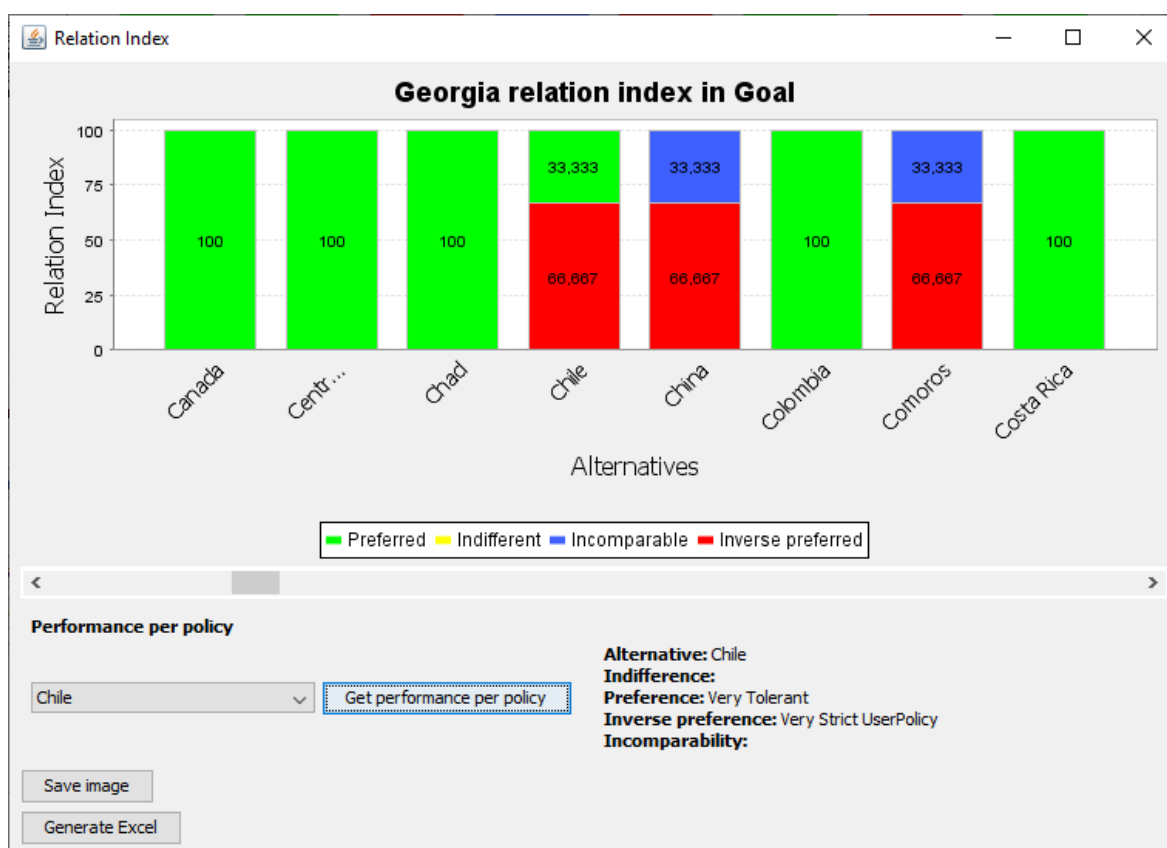


Figure 64: Georgia Relation Index in *Goal* compared to Chile

Another option available to the decision maker to evaluate the results of Georgia when using the policies is the Acceptability Index (Figure 65).

As expected after analyzing the Relation Index, using Very Tolerant gives better results to Georgia than using Very Strict. In this case, the difference is being in the first quartile instead of being in the second one.

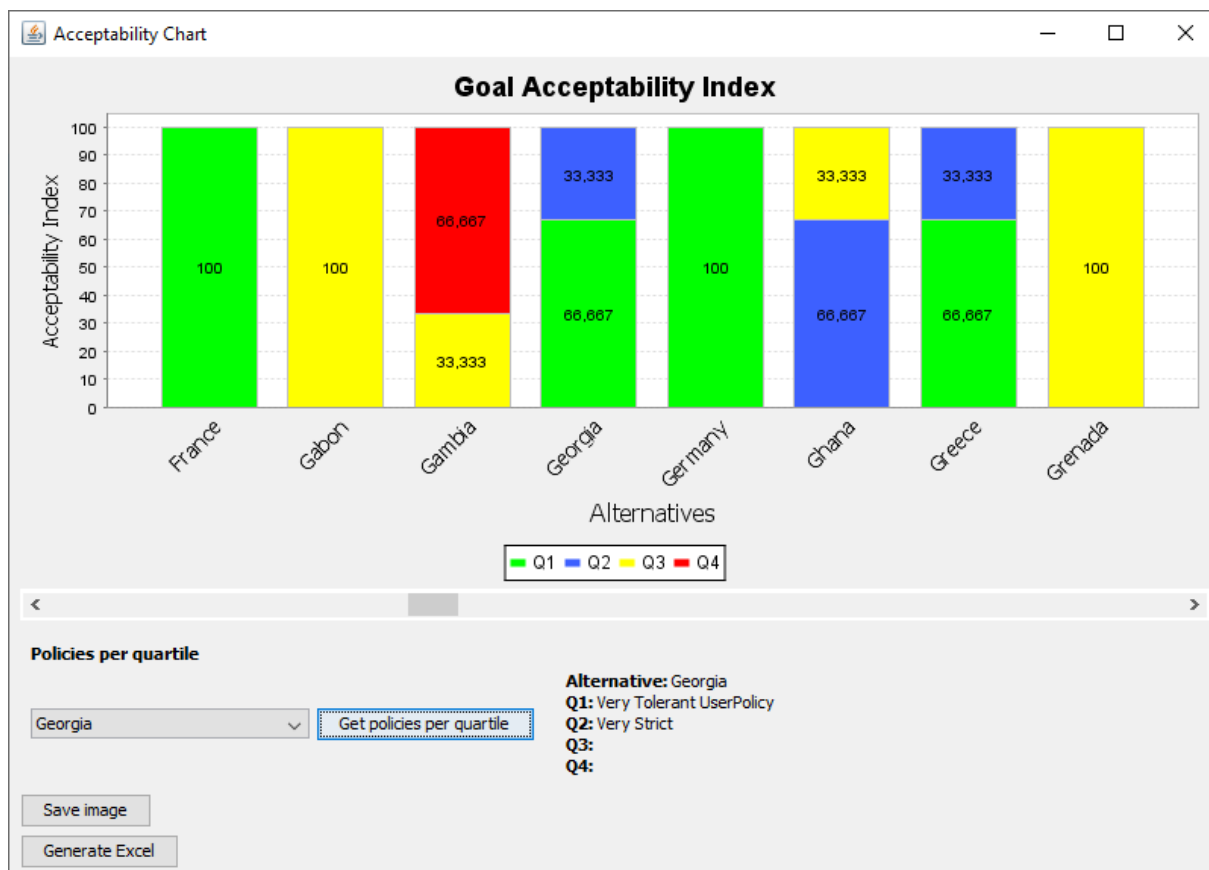


Figure 65: Georgia Acceptability Index

7 Conclusions and future work

7.1 Conclusions

With the work already been completed, and looking at the objectives aimed to be accomplished, I think they have been done successfully. The software now provides many new features that help the decision maker to analyze the problem from different perspectives.

First of all, the user can stop the calculation of the results at intermediate levels of the hierarchy. This is particularly interesting when the hierarchy is complex, with many levels of imbricated criteria. If the number of alternatives is large, this functionality is also important because the calculation of the credibility matrices and its exploitation is quite time and resource consuming (it is of quadratic cost). Therefore, the user can execute the process in small steps and obtain results at intermediate levels with less time.

Second, the new tools designed to help a decision maker using the program by easing the parameter choosing process permit to have a set of 5 predefined policies. Using them, the decision maker can immediately get results of any problem without having to manually modify all the thresholds. In large and complex hierarchies, lots of parameters are needed, and using any of the policies, allows an easy way of having some initial values that give useful results to the decision maker. Then, the decision maker can modify those initial values to get the desired model, and will be less time consuming than not having any initial values to begin with. Moreover, there are 5 different policies to be chosen for each criterion, from the strictest *Very Strict* to the most tolerant *Very Tolerant*, giving the possibility of being more or less strict in a determined criterion, without having to test a lot of values.

Also, the new methods designed to analyze the results the program gives, allow the decision maker to do a quick analysis of multiple results of a problem, so it is much easier to find which result is closer to the model desired. This is very useful to quickly find a set of initial values for the problem. These tools also allow a deeper analysis of different results, allowing comparing them in a visual way.

In conclusion, these methods should allow a non-expert decision maker use the program, and an expert decision maker have more tools to easily find the parameters needed to build the model wanted.

After being designed, they were implemented in the already existing software package ELECTRE-H in its version 2.0, creating a new version 3.0. To test the implementation, an already existing dataset provided with the software package was used. Once all the new functionalities were tested to be working as designed, a more complex dataset was needed to test that the methods proposed were giving the results we were aiming at. It was created by using a huge amount of real data, to simulate a real life problem consisting in ranking countries regarding some environmental criteria. We have simulated the case of a governmental decision maker that wants to analyze the situation of a country. The different charts and tables that can now be generated enabled to detect the preference relations among countries as well as its overall position in 3 different criteria. The case of the country of Georgia has been used to illustrate this analysis. These results showed how Georgia performs compared to other countries, knowing which criteria need an improvement in order to have a better overall ranking position on each of the different dimensions.

7.2 Future work

The immediate work will consist on registering the ELECTRE-H 3.0 version to an office of Intellectual Property and make it available to the Innoget platform for commercialization, as it was done with the previous versions.

The ITAKA research group is interested in testing these new tools with some real decision problem. In the past, they have been using this software in collaboration with other research groups and institutions in problems related to environmental assessment or tourism applications. However, at the moment there is no project ongoing, so this has not been possible. If some users contact the group after the release of this new version, a test in a real setting will be possible.

Regarding future research work derived from the one done in this project, we devise two main tasks:

1. The values of the parameters when the ranking is done with Net Flow Score are calculated with the preorders obtained from Distillation. Although the value of gamma obtained will not be very different to the maximum NFS score, it should be better to adapt the values proposed to the different discrimination policies distinguishing the two types of ranking methods.
2. The ELECTRE-H software also permits to solve decision problems using the ELECTRE-TRI-B method, focused on sorting problems. In this case, the method does not calculate a ranking of the alternatives, instead, it classifies the alternatives into a small set of predefined categories. This assignment is done at each intermediate criterion, as well as on the goal. The propagation of assignments is done with a score that is similar to the Rank Order Value (gamma), but it is called Category Order Value and it is calculated in a different way. The techniques and tools proposed in this work could be also adapted to this case.

8 References

- [1] Corrente, S., Greco, S., Kadzinsky, M., Slowinski, R., “Robust ordinal regression in preference learning and ranking”, *Machine Learning*, vol. 93, pp. 381-422, 2013.
- [2] Del Vasto, L., Valls, A., Slowinski, R., Zielniewicz, P., “The ELECTRE-III-H method for making decisions based on outranking relations with hierarchically structured criteria”, *Expert Systems with Applications*, vol. 42(11), pp. 4910-4926, 2015.
- [3] Del Vasto, L., Fernández-Cavia, J., Huertas, A., Moreno, A., Valls, A., “Official Tourist Destination Websites: Hierarchical Analysis and Assessment with ELECTRE-III-H”, *Tourism Management Perspectives*, vol. 15, pp. 16-28, 2015.
- [4] Del Vasto, L., Kumar, V., Chao, T-C., Valls, A., “A decision support system to find the best water allocation in a Mediterranean river basin in future scenarios of global change”, *Journal of Experimental & Theoretical Artificial Intelligence (JETAI)*, Taylor-Francis Pub., vol.28(1-2) pp.331-350, 2016.
- [5] Del Vasto, L., Valls, A., Zielniewicz, P., Borràs, J., “A hierarchical multi-criteria sorting approach for recommender systems”, *Journal of Intelligent Information Systems*, 46, pp. 313-346, 2016.
- [6] Figueira, J., Greco, S., Roy, B., Slowinski, R., “An Overview of ELECTRE Methods and their Recent Extensions”, *Journal of multi-criteria decision analysis*, vol. 20, pp. 61-85, 2013.
- [7] Figueira, J., Greco, S., Ehrgott, M., “Multiple criteria decision analysis: State of the Art Surveys”, Springer, 2016.
- [8] Fourati, M.A., Valls, A., Ben Ayed, M., Kamoun, A., Moreno, A., “Decision support system for the irrigation scheduling optimization”, *13th International Conference on Modeling Decisions for Artificial Intelligence*, pp. 1-12, 2016.
- [9] Griño, M., del Vasto-Terrientes, L., Valls, A., “Integration of a rating function based on rank positions for the management of hierarchical structures of criteria in ELECTRE”, *Report de Recerca de la Universitat Rovira i Virgili*, DEIM-RR-13-002, 2013.
- [10] Kadzinski, M., Greco, S., Slowinsky, R., “Selection of a representative set of parameters for robust ordinal regression outranking methods”, *Computers & Operations Research*, vol. 39(11), pp. 2500-2519, 2012.
- [11] Tervonen, T., Figueira, J., Lahdelma, R., Salminen, P., “Towards robust ELECTRE III with simulation: theory and software of SMAA-III”, 2007.
- [12] Wendling, Z. A., Emerson, J. W., Esty, D. C., Levy, M. A., de Sherbinin, A., et al. ,*Environmental Performance Index*. New Haven, CT: Yale Center for Environmental Law & Policy, 2018.
<https://epi.yale.edu/>