



International Institute for
Applied Systems Analysis
Schlossplatz 1
A-2361 Laxenburg, Austria

Tel: +43 2236 807 342
Fax: +43 2236 71313
E-mail: publications@iiasa.ac.at
Web: www.iiasa.ac.at

Interim Report

IR-09-22

**User Guide to MCA:
Multiple Criteria Analysis of Discrete Alternatives
with a Simple Preference Specification**

Marek Makowski (marek@iiasa.ac.at)

Janusz Granat (J.Granat@itl.waw.pl)

Hongtao Ren (renh@iiasa.ac.at)

Approved by

Detlof von Winterfeldt (detlof@iiasa.ac.at)

Director, IIASA

December 2009

Interim Reports on work of the International Institute for Applied Systems Analysis receive only limited review. Views or opinions expressed herein do not necessarily represent those of the Institute, its National Member Organizations, or other organizations supporting the work.

Foreword

Practically all important decisions involve analysis of several (or even many), typically conflicting, criteria. Analysis of trade-offs between criteria is difficult because such trade-offs for most problems are practically impossible to be defined a-priori even by analysts experienced in Multi-Criteria Analysis (MCA). Therefore the trade-offs emerge during an interactive MCA which actually supports a learning process about the trade-offs. Hence, effective MCA methods are important for actual support of decision-making processes, especially those related to policy-making.

IIASA has been developing novel methods for MCA since mid 1970s, and successfully applying them to many practical problems in various areas of applications. However, there are new practical problems for which the existing MCA methods (developed not only at IIASA but also by many researchers all over the world) are not satisfactory. In particular, discrete decision problems with a large number of criteria and alternatives (the latter making pairwise comparisons by the users impracticable) demand new methods. For example, MCA analysis of future energy technologies involves over 60 criteria and over 20 discrete alternatives; a careful requirement analysis of this application has proven that none of the existing MCA methods is suitable for an effective analysis of the corresponding problem. Moreover, this analysis has been done by a large number of stakeholders with diverse backgrounds and preferences; most of them have no analytical skills, therefore the specification of preferences needed to be simple but still provide effective and intuitive analysis of the Pareto set.

The dedicated Web-based application developed for supporting stakeholders in multi-criteria analysis of future energy technologies has been enhanced to support specification and analysis of any discrete alternative choice problem. The enhanced application (called MCA) has been made available for research and educational purposes. This report serves as the user guide and tutorial for the MCA Web-site.

Abstract

This report provides detailed information about using the MCA, which is the Web-based application for multiple criteria analysis of discrete alternatives with a simple preference specification. MCA in its currently available version supports analysis of a set of discrete alternatives, defined either interactively or through a prepared file. Each alternative is characterized by several attributes (indicators) that serve for specification of criteria. MCA has been designed for problems having large numbers of alternatives and of criteria. However, it can also be used for problems with small numbers of criteria and/or alternatives.

The report is composed of two main parts. The first one provides an overview of the MCA and summarizes the methodology of multiple criteria analysis implemented in the MCA. The second part consists of the step-by-step detailed tutorial to the MCA organized into three stages corresponding to (1) basic functions of the MCA, (2) specification of problems, instances, and analyses, and (3) interactive multicriteria analysis process.

Acknowledgments

The authors gratefully acknowledge help of Bartosz Kozłowski of IIASA, who over the period of four years has systematically contributed his expertise in several areas, including data basis, Web-based applications, and software development. Moreover, he has provided the so-called *Framework* composed of software modules supporting implementation of Web-based applications.

The authors thank Stefan Hirschberg, Warren W. Schenler, and Peter Burgherr of the Laboratory for Energy Systems Analysis, Paul Scherrer Institute, Villigen, Switzerland for numerous discussions during several meetings, many lengthy phone-calls, and countless email exchanges. We also thankfully acknowledge comments of members of other teams participating in the Research Stream 2b of the NEEDS project provided during testing of the MCA-NEEDS version of the application. Moreover, we thank Dan Dolk of the Naval Postgraduate School, Monterey, USA, who has provided detailed comments and suggestions on an early prototype of the MCA.

The authors gratefully acknowledge the long-term collaboration with, and help of Andrzej P. Wierzbicki of the National Institute of Telecommunications, Warsaw, Poland, who over three decades greatly influenced the developments of the multiple criteria methodology.

All these inputs have considerably helped to design the MCA functionality and interface.

The research reported in this paper was partly financially supported by the EC-funded Integrated Project NEEDS (project no: 502687), and by the Austrian Federal Ministry of Science and Research.

About the authors

Marek Makowski leads the IIASA Integrated Modeling Environment Project. His research interests focus on model-based support for solving complex problems, which incorporates three interlinked areas. First, integration of interdisciplinary knowledge and its representation by mathematical models. Second, creation of knowledge by comprehensive model analysis, including multicriteria methods. Third, tailoring the modeling process to meet the needs of decision-making processes. Thus Marek's research interests cover a cluster of areas relevant to the adaptation (whenever possible) or development (when needed) of methodology, algorithms, and software for model-based decision-making support. This includes more specific topics in Operations Research such as: multicriteria problem analysis, large scale optimization, optimization of badly conditioned problems, use of database management systems for complex models, decision analysis and support, user interfaces in decision support systems, effective treatment of uncertainty and risk.

Marek has published over 130 papers and book-chapters, co-edited four books, coordinated or led several scientific projects, and has been twice guest editor of the European Journal of Operational Research.

Janusz Granat is a leader of a Division of Advanced Information Technology at the National Institute of Telecommunications. He also lectures on decision support systems and management information systems at the Warsaw University of Technology. His scientific interests include decision support systems, multi-criteria analysis, modeling, data mining, event mining, techno-economic analysis and the design of the telecommunications network. He has been involved in various industrial and scientific projects e.g., data warehousing and decision support systems for telecommunication industry, building data mining models for marketing departments, development of decision support systems for energy management.

Hongtao Ren is a researcher in the IIASA Integrated Modeling Environment Project (IME). His research interests include modeling environments, computerized support for knowledge creation, integrated system for scientific creativities, knowledge engineering, semantic web, and text mining. He has developed in collaboration with the IIASA colleagues diverse Web-based applications, including Multiple Criteria Analysis of Discrete Alternatives, and Multiple Criteria Analysis of Future Energy Technologies. He has been participating in various activities of the Center of Excellence at Japan Advanced Institute of Science and technology, and in particular developed Creative Environments for supporting scientific research.

Contents

1	Introduction	1
2	Overview of MCA	2
2.1	Access to the MCA	2
2.2	Specification of multicriteria problem analysis	2
2.2.1	Problem specification	3
2.2.2	Problem instance specification	4
2.2.3	Analysis specification	4
3	Methodology implemented in MCA	4
3.1	Preliminaries	4
3.2	Main elements of multiple criteria analysis in MCA	6
3.3	Selection of a parent iteration	6
3.4	Analysis of results of a current iteration	7
3.4.1	Analysis in the criteria space	7
3.4.2	Auxiliary analysis in the alternative space	9
3.5	Selection of a solver	11
3.6	Specification of preferences	11
3.6.1	Relative importance of criteria	12
3.6.2	Relative importance in hierarchy of criteria	13
3.6.3	Selecting criteria for improvement and compromising	14
3.7	Pareto alternative for the specified preferences	15
4	MCA basic structures and functions	15
4.1	Contact, help and documentation	16
4.2	Overview of the data structure	17
4.2.1	Pre-loaded problems	17
4.3	Login to MCA and MCA-Needs	18
4.3.1	Registration of new users	18
4.3.2	Managing PIN-codes	19
4.4	Navigation through the MCA	19
4.4.1	Upper control panel	20
5	Problems, instances, and analyses	20
5.1	Problem specification	20
5.1.1	The working area	21
5.1.2	Bottom control panel	22
5.1.3	Specification of a new problem	22
5.2	Instance specification and selection	23

5.3	Analysis specification and selection	24
6	Analysis process	25
6.1	Controlling the analysis process	25
6.1.1	Selection and analysis of the current iteration	26
6.1.2	Creating a new iteration	27
6.1.3	Finishing the analysis	28
6.1.4	Messages from the solver and the user interface	28
6.2	Tutorial example	28
6.3	Accessing the tutorial analysis	29
6.4	Basic analysis	30
6.4.1	Setting the stage	30
6.4.2	Simple trade-offs in criteria space	31
6.4.3	Using the chart options and notes	32
6.4.4	Analysis in the alternative space	33
6.4.5	Final steps of the basic analysis	34
6.4.6	Towards preliminary conclusions	35
6.5	Advanced analysis	35
6.5.1	Preparation of the analysis	35
6.5.2	Preliminary considerations	35
6.5.3	Initial analysis of realistic alternatives	37
6.5.4	Using a Pareto solution as the RFP	38
6.5.5	Analysis in the alternative space	38
6.6	Carrying forward	40
7	Tips for the MCA	41
	References	42
A	Structure of the csv-file with problem and instance definition	44
A.1	Car selection problem example	44
A.2	Defining a new problem	44
A.3	Problem instance specification	45

List of Tables

1	A simple example of Pareto alternatives.	5
2	Attributes of candidates for our new home.	29
3	Nicknames and the corresponding descriptions of alternatives for our new home.	29
4	Illustrative example of fictitious cars' attributes.	44

List of Figures

1	The components of problem analysis.	3
2	Criteria and selected solutions chart.	7
3	Characteristics of alternatives (for problems with less than 11 criteria). . .	9
4	Characteristics of alternatives (for problems with more than 10 criteria). .	9
5	Alternatives and their strengths/weaknesses.	10
6	Specification of relative importance of criteria.	12
7	Specification of relative importance of criteria defined within a hierarchi- cal structure.	13
8	Specification of preferences: relative importance combined with selecting the criteria to be improved, relaxed, stabilized, and freed.	14
9	The MCA login form.	18
10	Navigation through MCA: instance specification screen.	19
11	Problem specification and selection screen.	21
12	Icons used for controlling the actions.	21
13	Specification of a new problem.	22
14	Instance specification form.	24
15	The main window of the analysis.	25
16	Control panel of the analysis main window.	26
17	Initial iteration of the basic analysis.	30
18	Initial iteration of the basic analysis.	31
19	Criteria chart and preferences for the fourth iteration.	33
20	Comparison of expensive alternatives.	33
21	Initial iteration of the advanced analysis.	36
22	Comparing the four top alternatives from iteration 7.	39
23	Same comparison as in Figure 22 but with the <i>Park</i> as the reference alter- native.	40
24	A simple example of criteria hierarchy,	46

User Guide to MCA: Multiple Criteria Analysis of Discrete Alternatives with a Simple Preference Specification

Marek Makowski^{*} (*marek@iiasa.ac.at*)

Janusz Granat^{** ***} (*J.Granat@itl.waw.pl*)

Hongtao Ren^{*} (*renh@iiasa.ac.at*)

1 Introduction

This User Guide provides detailed information about using the MCA, which is the Web-based application for multiple criteria analysis of discrete alternatives with a simple preference specification. MCA in its currently available version supports analysis of a given set of discrete alternatives, each characterized by more than one criterion. The set of alternatives can be either defined interactively, or uploaded from a file. MCA has been designed for problems having large numbers of alternatives and of criteria. However, it can also be used for problems with small numbers of criteria and/or alternatives.

The motivation for the development of the MCA and the corresponding requirement analysis as well as the methodological background for, and detailed description of the implemented methods for multiple criteria analysis can be found in [2], [3], [5], and [6].

This guide covers the following topics:

- Overview of the MCA, including access information, is provided in Section 2.
- Methodology of multiple criteria analysis implemented in the MCA is summarized in Section 3.
- Step-by-step tutorial to the basic functions of the MCA in Section 4, and to specification of problems, instances, and analyses is presented in Section 5.
- Detailed tutorial to interactive multicriteria analysis is available in Section 6.

Although it is possible to use MCA without knowledge on Multi-Criteria Analysis methodology, we encourage the users to become familiar with this methodology. In this guide we provide only an outline of methodological background. We refer readers interested in methodology of multi-criteria analysis to more concise works in this field. More comprehensive discussion on the methodological issues, and suggestions for further reading can be found in e.g., in [1], [4], [7], [8], and [9].

^{*}Integrated Modeling Environment Project, IIASA.

^{**}National Institute of Telecommunications, Warsaw, Poland.

^{***}Warsaw University of Technology, Warsaw, Poland.

2 Overview of MCA

MCA supports multiple criteria analysis of a large number of alternatives each characterized by a possibly large number of attributes. However, MCA also effectively supports analysis of problems having a small number of attributes and/or alternatives. Moreover, although the current version of MCA uses of simple way for preference specification, it is designed to be extended for using more advanced approaches to preference specification.

Each user has a private data space, in which all his/her data is kept. Upon the registration to MCA several problems are uploaded to this space, and the user can analyze those pre-loaded problems. The user may also want to create own problems, either interactively or by uploading them from csv files. The user data are stored on the MCA server, which allows a continuation of analysis at a later time.

The MCA has been designed to support each user in analysis of several (possibly also many) problems, some of which can be developed by modifications of a selected problem. For each analyzed problem it is possible to develop several problem instances, each composed of selected attributes that are used for defining criteria. For each instance one can create several analyses, each composed of a number of iterations; each iteration is defined by a specific set of preferences.

Such a structure of the MCA functionality supports an easy specification of instances of problems to be analyzed, and also structuring multiple criteria analysis by more advanced users.

In the remaining part of this Section we outline functionality of each of the MCA basic elements.

2.1 Access to the MCA

MCA is one of the currently two available IME Web-based applications that can be accessed through a link available from the Web site: <http://www.iiasa.ac.at/~marek>. MCA is free to use for research and educational purposes;¹ a prior registration is however required. The details on the registration and login procedures are provided in Section 4.3.

Registration to all IME applications needs to be done once; it is available from the login page, and it is automatized therefore it typically takes few minutes (for security reasons activation code for each registration is emailed to the new user, who is expected to activate his/her registration by simply clicking on the link included in the email.

2.2 Specification of multicriteria problem analysis

In the MCA we distinguish three types of entities: problems, instances of a selected problem, and analyses of a selected instance. These three-layer structure is illustrated in Fig. 1. Such an approach helps in organizing also a more comprehensive process of the problem analysis, as well as supports reusing the data (that can be shared between various analyses). The overhead for following this structure (as compared with a one-layer structure for simple analysis of simple problems) has been made negligible.

¹The conditions for the MCA use are available upon the registration.

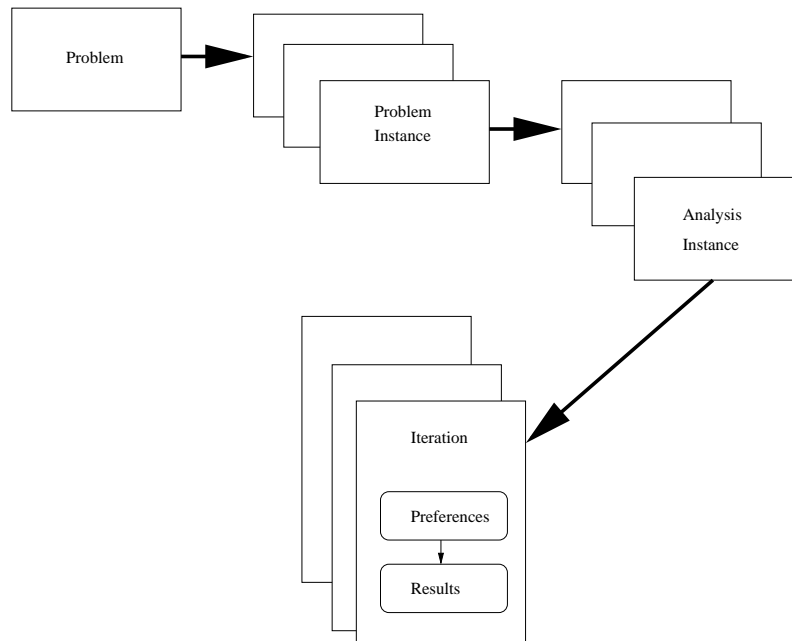


Figure 1: The components of problem analysis.

Therefore, the specification of an analysis is split into the corresponding three stages. Each of them consists of either a specification of a new, or a selection of a previously specified entity, namely:

1. a problem (Sec. 2.2.1),
2. an instance of a selected problem (Sec. 2.2.2),
3. an analysis of a selected instance composed of iterations (Sec. 2.2.3).

Specification of problems and their instances can be either done interactively or uploaded from a prepared simple text file. Specification of an analysis is done through a simple interactive form.

After an analysis is either newly generated or selected from the list of previously prepared analyses, the user can start the iterative analysis process composed of a sequence of iterations that forms an analysis instance. In each iteration the user specifies preferences; then the selected solver finds a Pareto alternative that fits best (in terms of the method implemented for the selected solver) the user preferences. The user is supported in analyzing the solutions in the criteria space as well as in the alternative space.

2.2.1 Problem specification

Multicriteria problem specification is composed of three parts:

- alternative names, indexed by $j = 1, \dots, m$ (also $j \in J$),
- attribute², names indexed by $i = 1, \dots, n$ (also denoted $i \in I$).
- attribute values denoted by q_{ij} specified for each pair $\{i, j\}$

Thus the attribute values are organized in a matrix Q composed of elements q_{ij} . For convenience j -th column of Q (composed of values of all criteria for j -th alternative) is

²Attributes are also commonly called indicators or outcomes, or criteria.

denoted by q_j , and i -th row of Q (composed of values of i -th criterion for all alternatives) is denoted by q_i .

2.2.2 Problem instance specification

An instance of a selected multicriteria problem is composed of:

- Set of alternatives (selected from the problem specification).
- Set of criteria defined using a selected attribute and one of the predefined types (maximized, minimized, target³).
- Optionally, a hierarchy of criteria can be defined. Users wishing to deal with criteria hierarchy are strongly advised to become familiar with the underlying methodology.

2.2.3 Analysis specification

From the user of point of view, a MC analysis of a selected problem instance is just a container for a set of iterations done by the user for a selected problem instance. Each iteration has a single parent (a selected iteration that provides initial values of preferences). The iterations are ordered in the sequence in which they were created, and organized in a tree. This approach provides a structure for possibly large sets of iterations, and also to assure privacy (each MCA user has access only to iterations they he/she has created).

Specification of an analysis is done interactively through a simple form. A user may specify several analysis for each problem instance. This is typical for advanced users who first make several analyses for learning about the properties of the instance, and then exploit such experience in a series of second-stage analyses.

3 Methodology implemented in MCA

3.1 Preliminaries

Multi-criteria analysis is typically an iterative process, in the MCA composed of iterations done for a selected analysis instance. The first iteration is generated automatically. Then the user can select any of the iterations (from the tree of iterations belonging to his/her analysis composed of the initial iteration, and the iterations he/she has made) as a basis for creating a new iteration. Upon analysis of criteria values for the selected parent iteration the user modifies her/his preferences, and calls the solver, which provides a Pareto-efficient alternative corresponding best to the specified preferences.

This typical MCA procedure contains the main challenge for both developers and users of any multicriteria analysis method:

how to find a Pareto solution that matches best the user preferences.

This challenge appears to be even bigger for the MCA methods designed and implemented for users having no experience in mathematical modeling. However, it has been observed that also users with extensive experience in MCA have problems with a consistent specification of preferences, if such a specification involves a process with several interlinked steps.

³Interface to the target-type has not been implemented yet.

For all methods implemented in the current MCA version, the specification of preferences is done in the probably easiest way: by selecting a relative importance of each criterion. The relative importance is expressed in qualitative terms, and then the relative importance are mapped into the user defined weights. Such weights are used for aggregating the preferences, and different methods use different aggregations.⁴

Thus the user forms a pattern of relative criteria importance, and expects to get a Pareto solution (alternative) having the corresponding pattern of criteria values, e.g., possibly best values of all very important criteria, good values for all important criteria, etc. However, usually there is no alternative having criteria values that correspond well to the user expectations. To illustrate this point, let us examine the alternatives summarized in Table 1.

	C1	C2	C3	C4
A1	0.5	0.4	0.7	0.7
A2	1	0	1	0
A3	0	1	0	1
A4	0.3	0.5	1	1

Table 1: A simple example of Pareto alternatives.

Consider a situation when the user analyzes alternative *A1*, and has the following preferences:

- would like to improve criteria *C1* and *C2*, and
- is ready to compromise the values of criteria *C3* and *C4*.

If only four alternatives (*A1* through *A4*) are available, then improving the values of both *C1* and *C2* is impossible.⁵ However, after learning that improving values of these two criteria simultaneously is not possible, the user is likely to examine other trade-offs in order to find the alternative that fits her/his preferences best. For example, upon analysis of the trade-offs for alternative *A4* the user may want to reconsider his/her preferences, and find the alternative *A4* more attractive than *A1*, i.e., he/she may agree to compromise the criterion *C1* (that he/she originally wanted to improve) for improving values of all other criteria.

The MCA is actually a learning process during which the user modifies his/her preferences in order to find an alternative with trade-offs between criteria values that fits best the user preferences that are modified upon analysis of attainable trade-offs between criteria values. In other words, a specification of preferences is a tool for finding a preferred alternative. The preferred means that it has the best (in a subjective opinion of the user) trade-offs of the criteria values amongst all alternatives; this should not be confused with the specified trade-offs which are often not attainable (i.e., there exists no alternative having the specified pattern of criteria values). Moreover, it is not really important which pattern of criteria importance led to finding the preferred alternative; actually, each alternative can be found for many specified preferences, most of which may differ substantially in pairwise comparisons.

⁴Basic information on the methods implemented in MCA, and the corresponding solvers are available in Section 3.7.

⁵This can easily be seen by looking at Table 1, but for problems with more criteria and/or alternatives one needs a MCA tool that assists the user in MC analysis.

The MCA supports two types of analysis:

- Choosing - to select one alternative (e.g., a car for buying).

The solution of the choice problem is non-unique. Its solution depends on preferences of a person making analysis in terms of trade-offs between the values of attributes; e.g., fuel consumption versus price or safety. In other words, different alternatives (each defined by a given set of attributes' values) correspond to different trade-offs. The primary role of the MCA is to support analysis of such trade-off in order to find a Pareto solution that fits best the personal preferences.

- Ranking - to order the set of alternatives.

In MCA ranking is a *derivative* of the Pareto-alternative selection which shows *the next best* alternatives. In other words, ranking is based on iterative procedure, in which the chosen Pareto alternative is removed from the set of alternatives, and the next best (for the same preferences) Pareto solution is found in such a smaller set.

Further on we focus on the choice problem, i.e., selection of a Pareto solution (non-dominated alternative)⁶ that fits best the user preferences.

3.2 Main elements of multiple criteria analysis in MCA

For each analysis instance (later on called *analysis*) initial iteration is generated automatically with equal preferences for all criteria selected for the corresponding model instance. The method called *Objective Choice* (described as *Objective selection* in [8]) is used for selecting the Pareto solution for such preferences.

Each user-created iteration starts from a previously done iteration that is selected by the user. During analysis and specification of new preferences such an iteration is called the *current iteration*. After the preferences are specified the current iteration is considered to be the *parent iteration* (because it provides initial values of the preferences to specified for a new iteration), and the newly created iteration takes the role of the current iteration. Of course, the user may at any time select any iteration to be the current one.

In other words, creation of a new iteration is composed of the following steps:

- Selection of a parent iteration.
- Analysis of the criteria values for the parent iteration.
- Optionally,⁷ selection of a specific MCA method to be used by the MC solver for computing a Pareto solution.
- Specification of preferences to be applied in the new iteration.
- Finding the corresponding Pareto solution.

We comment on these five elements in the following Sections.

3.3 Selection of a parent iteration

Typically, the user analyzes the result of the newest iteration, and modifies the corresponding preferences in order to find a solution with trade-offs between criteria that she/he considers to be better. However, often solution of the newest iteration is considered worse

⁶A summary of basic concepts of Pareto-efficiency can be found in [3], which is accessible also on-line from the MCA.

⁷The last selected method is used, if the user does not select another method.

than that of one of the previously found solutions. Therefore, MCA supports an easy access to any iteration from the tree of iterations composed of the initial iteration, and the iterations the user has made. A selected iteration is often also referred to as the current iteration.

The preferences of a current iteration can be modified or another solver can be chosen for finding Pareto solution for the same preferences. A child iteration is created for a selected iteration only if a solver is called. Therefore each iteration may have any number of children (including none), and each iteration (except of the initial one, which is automatically generated and assumes equal preferences for all criteria) has exactly one parent.

3.4 Analysis of results of a current iteration

The user can select any iteration to be the current one, analyze the solution for the corresponding preferences, and use them as a basis for a child analysis. Analysis of the solution of a current iteration is usually done in:

- the criteria space, and (optionally)
- the alternative space.

Such analysis is a key element of most of the multiple criteria analyses, therefore we now comment on these two types of analysis.

3.4.1 Analysis in the criteria space

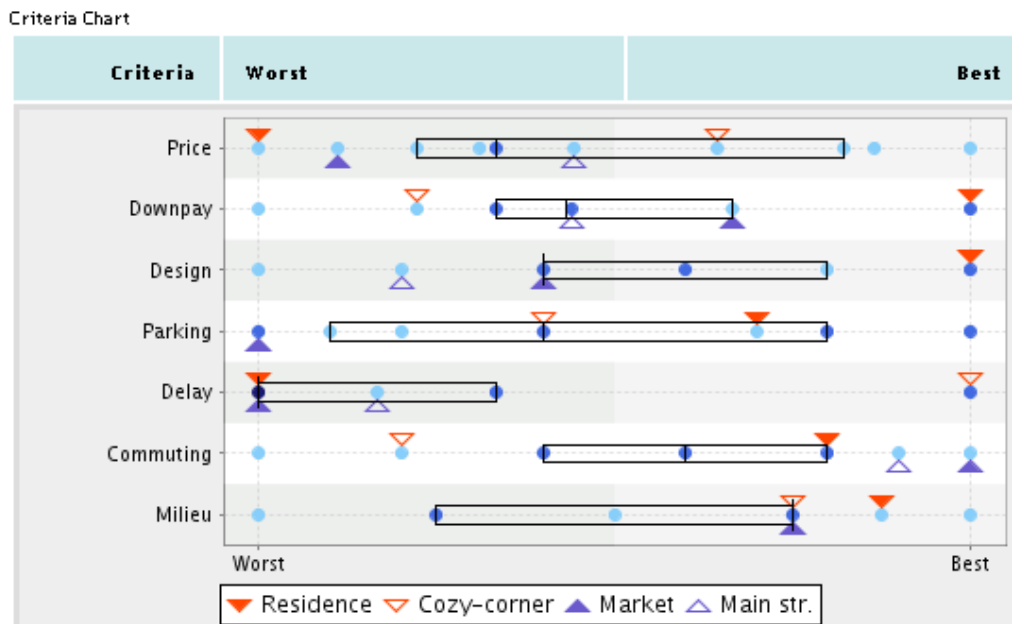


Figure 2: Criteria and selected solutions chart.

The main source of information about the criteria and selected solutions is available on the *Criteria chart* shown in Fig. 2. Composition of this chart extends the parallel coordinates concept (i.e., displaying each criterion in a line) by providing also information about the

distributions of criteria values, and marking selected solutions. Therefore the chart is composed of lines; each line corresponds to a criterion and its marked by the criterion name.⁸

Each line contains the following elements characterizing the corresponding criterion (some of the elements are optional, and can be controlled by the user, as described below).

- Blue dots mark the criterion values for the corresponding alternative. The values are normalized, therefore the worst and best are located always at the left and right end of the line, respectively. The intensity of the color indicates:
 - ★ light - only one alternative has the corresponding value,
 - ★ medium-dark - at least two (but less than 1/3 of all) alternatives has the corresponding value,
 - ★ dark - more than two (but at least 1/3 of all) alternatives has the corresponding value.
- By pointing the mouse over a dot, one can display a hint (not illustrated in the Figure) which contains:
- ★ the name of the corresponding alternative (or alternatives, if more than one has the corresponding value),
 - ★ the percentage of the normalized range of the criterion values (i.e., 0% and 100% denotes the worst and the best value, respectively),
 - ★ original value of the criterion.

Note that dots marking values that differ by less than 2% of the corresponding criterion values range are clustered. Therefore, the medium-dark and dark dots represent alternatives with the criterion values that either are the same, or differ by less than 0.02.

- Boxplot containing the two middle quantiles of the criteria values distribution; the bar in the boxplot indicates median (if the median overlaps with one of the boxplot ends, then it is slightly longer). In other words: 25% of alternatives with worst and the other 25% best are located outside of the boxplot, on the left and right side respectively.
- Solid red triangle indicating the criterion value for the alternative that corresponds to the solution of the current iteration (the name of the alternative is displayed in the caption of the chart).
- Red triangle border indicates the criterion value for the alternative that was the best one for preferences specified in the parent iteration (therefore, if the value of this alternative is the same as the currently best, then the marks overlaps, and the border is not seen).
- Blue solid triangle and its border marking values of up to two alternatives optionally selected by the user (see below).

The following options of the criteria chart can be selected by the user by clicking on the marker on the right side of the *Chart option*:

- Selecting up to two alternatives values of which will be marked on the chart by the blue triangles.
- Hiding the boxplot.
- Connecting the values of the selected alternatives by lines.
- Hiding the inactive criteria.

Selected chart options have to be confirmed by clicking on the *Redisplay* button. The

⁸Almost all Figures in this guide use screen dumps from the analysis of the tutorial example outlined in Section 6.2. Generally, if a criteria hierarchy is defined for the corresponding problem instance (which is not the case for the analyses discussed in this guide), then only the leaf-criteria are displayed in the criteria chart. Higher level criteria cannot be included in the chart because criteria values are available only for attributes (i.e., lowest-level criteria).

selected options remain valid until the user resets them.

3.4.2 Auxiliary analysis in the alternative space

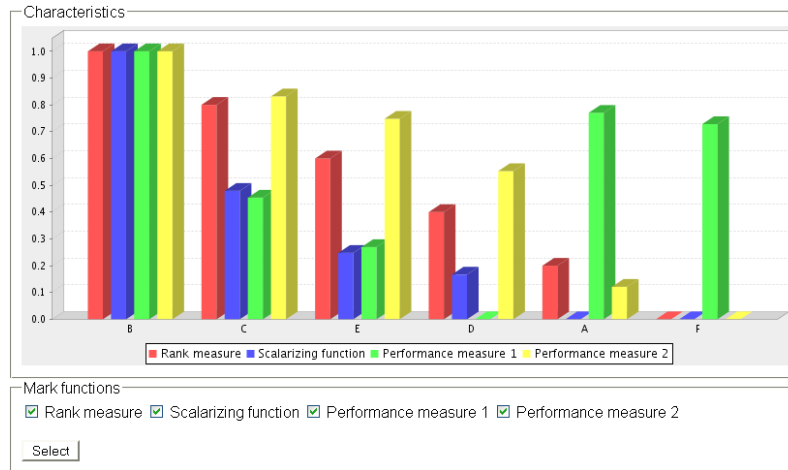


Figure 3: Characteristics of alternatives (for problems with less than 11 criteria).

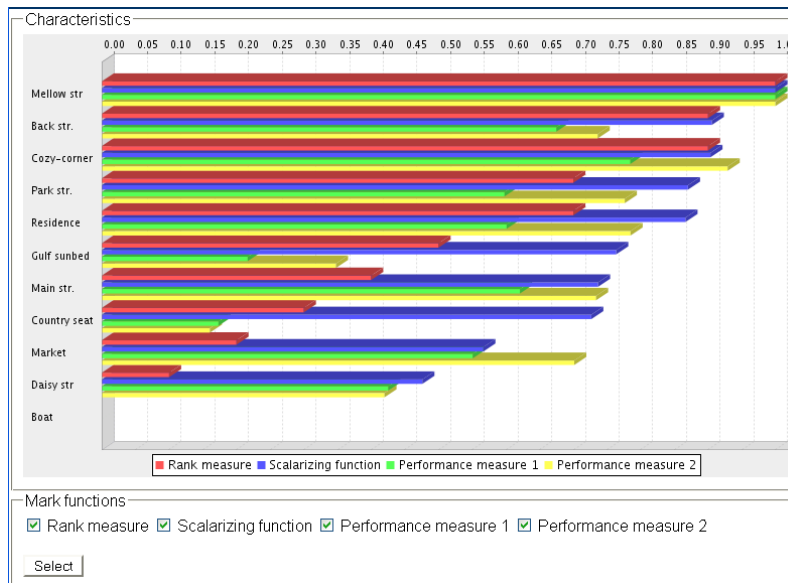


Figure 4: Characteristics of alternatives (for problems with more than 10 criteria).

Two charts and one table characterizing the solution in alternative space are available by clicking on the corresponding button (the buttons are inactive for iterations for which the solver reported errors):

- *Characteristics*, see Fig. 3 and 4,
- *Compare by Criteria*, see Fig. 5,
- *Ranking* provides a table with ranking of alternatives.

The following characteristics of alternatives are provided in charts shown in Fig. 3 and Fig. 4:⁹

- Rank measure.
- Values of the scalarizing functions.
- Optionally, two performance measures (which indicate the degree of the corresponding similarity measure between the best (displayed as left-most) and each other criterion).

One can hide selected measures by deselecting the corresponding box on the mark panel and confirming the choice by the *Select* button. To facilitate readability values for all bars are linearly mapped to the range [0, 1], where the value of 1 always corresponds to the best value/position. Pointing the mouse on a bar displays the hint with the name of the alternative, the name of the measure and the relative (as the fraction of the best value assumed to be equal to one) value.

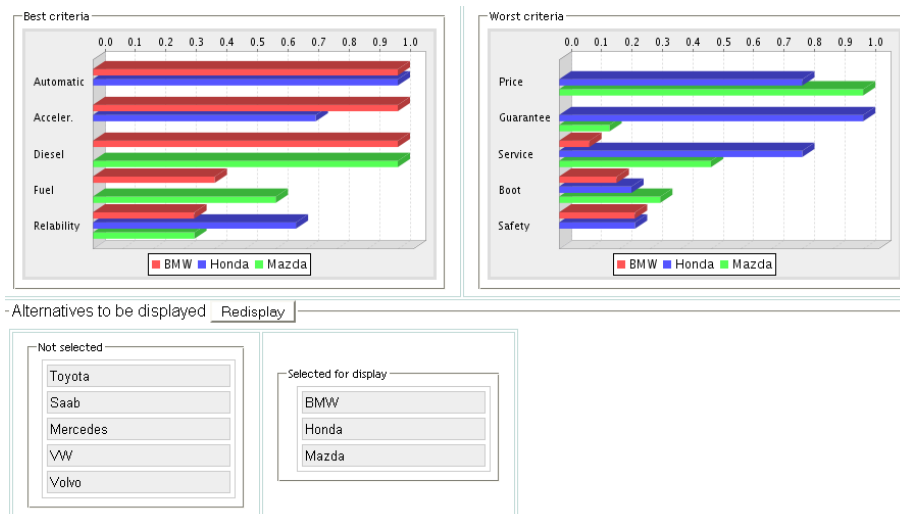


Figure 5: Alternatives and their strengths/weaknesses.

One can also analyze strengths and weaknesses (in respect to criteria values) of selected alternatives by the interactive chart shown in Fig. 3. First, one selects alternatives to compare by dragging them from the left to right box (or in the reverse direction to remove from the analysis) shown in the panel below the two displayed charts. The master alternative (see below) has to be dragged to the top of the right box. The selection has to be confirmed by clicking on the *Set* button.

For the selected master alternative the criteria are sorted according to their values for this alternative. Values for the upper and lower half of criteria are displayed in the left and right chart, respectively. Therefore the left panel contains the criteria on which the master alternative is strong, and displays the corresponding values of the other alternatives. The right panel contains the criteria on which the master alternative is weak (the weakest criterion at the top). Pointing the mouse on a bar displays the hint with the names of alternative and the criterion, and the relative (as the fraction of the best value assumed to be one) criterion value. This is useful, when many alternatives are selected for a comparison.

⁹The charts differ by one predefined chart option: vertical bars are used for smaller problems, and horizontal bars for larger problems (for which charts with vertical bars would not fit to a screen).

Finally, the alternative ranking is also available as a table (by selecting the *Ranking* button on the control panel).

3.5 Selection of a solver

The IME team has developed and tested over 30 methods and the corresponding solvers for multiple criteria analysis of discrete alternatives. Most of these methods are described in [3] and [5]. However, for a typical user an informed selection of a solver from such a large set is cumbersome and actually hardly needed. Therefore, in the currently available version of the MCA only four solvers are available:

- POA-NF and OPOA-Inv-NF described in [3], and
- RFP-Nadir and RFP-Pareto described in [5].

The first three solvers require specification of only relative criteria importance. The RFP-Pareto solver additionally requires preferences on the criteria improvement.

The solver used in the parent iteration is the default selection in the choice list of solvers. Usually, there is no need to change the solver, however in some situations it is rational to do so. Therefore we briefly comment on this issue.

The solution time is practically the same for all available solvers. However the Pareto alternative corresponding to a given set of preferences may be different, therefore it might be rational to try different solvers for the same preferences. Moreover, it is often rational to use the RFP-Pareto solver at the advance stages of analysis. More comments on this issue are provided in Section 3.7.

A change of a solver, if desired, should be done before the preferences are specified because the type of preference information is different for the solvers. However, in most cases the specified preferences will be retained after the solver selection is changed.

3.6 Specification of preferences

As long as the user is not satisfied with any solution found during the analysis she/he typically wants to modify her/his preferences in order to find another solution having the desired trade-offs between the criteria values. This can be done by creating a new iteration using the currently selected one as the parent iteration. This process is composed of the following steps:

- Optional selection of another MC method to be used by the solver for finding the Pareto solution fitting best the selected preferences. The method can be selected from the choice list labeled *Method*. The method selection should be done before actual specification of preferences because some methods use a different way of preference specification than the method used for the parent iteration. If this is the case, then the composition of the preference specification panels can be changed.
- Selection of preferences for each criterion by clicking on the corresponding button. The buttons are organized into panels, each composed of buttons representing a given type of preferences. The initial positions of the buttons are inherited from the parent iteration, therefore the choice of the method should be preferably made before starting the specification of preferences (should the preferences be already specified for a new iteration, the choices made will be reset). The choices made in the parent iteration are marked by gray shadow attached to the corresponding buttons. Moreover, the choices

made in the grand-parent iteration are marked by the gray dotted-line boarder (which overlap with the markers for choices made in the parent iteration, if they were the same). Therefore, it is easy to recognize how the preferences differ between the current, the parent, and the grand-parent iterations.

The preferences (and also a selected solver) are not stored until they are not confirmed by clicking on the *Solve* button. Confirmation of the preferences causes creation of a new iteration with the confirmed preferences.

For all currently implemented methods, the preferences are specified as relative criteria importance. For some methods the user additionally specifies for each criterion whether it should be improved, stabilized, or compromised. These two types of preferences are discussed in Sections 3.6.1 and 3.6.3, respectively. Comments specific to specification of preferences for instances with hierarchical criteria structure are summarized in Section 3.6.2.

3.6.1 Relative importance of criteria

Due to the requirements explained in [6] the current version of MCA uses a very simple way of preference specification that is suitable also for users without analytical skills. Since the inter-criteria preferences need to be specified therefore we refrained from specification of preferences within each criterion. In order to rationally deal with criteria types (maximized or minimized) and very diverse orders of criteria value magnitudes, all criteria values are linearly mapped into the $[0, 1]$ interval, where 0 and 1 correspond to the worst and best value, respectively.

User preferences

Criteria names	Relative importance
Price	<input type="radio"/> <input checked="" type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>
Fuel	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input checked="" type="radio"/> <input type="radio"/> <input type="radio"/>
Guarantee	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input checked="" type="radio"/> <input type="radio"/>

Figure 6: Specification of relative importance of criteria.

The preferences for the currently implemented methods are specified as relative criteria importance. The specification is done by selecting (for each criterion) one of the buttons in the panel shown in Fig. 6. The values are as follows (from left to right):

- ignore,
- vastly less important than average,
- much less important than average,
- less important than average,
- average importance,
- more important than average,
- much more important than average,
- vastly more important than average.

The value attached of each button is displayed as a hint whenever the mouse points on the button. The gray button surrounding shows the button value selected at the parent iteration.

The mapping of the selected values into scaling/weighting coefficients is presented in [3]. Note that the weights are normalized, therefore specifying an equal importance for all criteria has the same effect irrespectively of the selected importance level.

3.6.2 Relative importance in hierarchy of criteria

User preferences








Criteria names	Relative importance
Economy	
Price	
Fuel	
Guarantee	
Performance	
Safety	
Reliability	

Figure 7: Specification of relative importance of criteria defined within a hierarchical structure.

Specification of relative importance of criteria defined within a hierarchical structure requires understanding of:

- The way in which the weights are normalized in the subsets of siblings having a common parent.
- Multiplicative effects of weights at different hierarchy levels. In particular one should note that criteria "below" an ignored criteria (i.e., children, grandchildren, etc.) will also be ignored.

The MCA interface allows for specification of any preferences that are formally correct. However, a conscious specification¹⁰ of relative importance in hierarchical structure is a challenging task even for experienced analysts, especially for problem instances for which the criteria form an unbalanced hierarchy tree, and for problems with a large number of criteria organized in structures having more than two criteria levels.

A formal specification of the weight normalization in the criteria hierarchy can be found in [3].

A sample of the button panel for specification of relative criteria importance in in-

¹⁰Which involves a good understanding of the consequences in terms of the resulting weights that are computed for the leaf-criteria.

stances with criteria hierarchy is illustrated¹¹ in Fig 7. Note that some criteria may have the corresponding button row empty. This is the case in situations when (because of the normalization) any selection of relative importance will result in the same value of the corresponding weight, namely equal to 1.

3.6.3 Selecting criteria for improvement and compromising

User preferences		
Criteria names	Relative importance	Improvement
Price	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input checked="" type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>	<input type="radio"/> <input checked="" type="radio"/> <input type="radio"/> <input type="radio"/>
Amenities	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input checked="" type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>	<input type="radio"/> <input checked="" type="radio"/> <input type="radio"/> <input type="radio"/>
Parking	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input checked="" type="radio"/> <input type="radio"/> <input type="radio"/>	<input type="radio"/> <input checked="" type="radio"/> <input type="radio"/> <input type="radio"/>
Commuting	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input checked="" type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>	<input type="radio"/> <input checked="" type="radio"/> <input type="radio"/> <input type="radio"/>
Area	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input checked="" type="radio"/> <input type="radio"/> <input type="radio"/>	<input type="radio"/> <input type="radio"/> <input checked="" type="radio"/> <input type="radio"/>
Milieu	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input checked="" type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>	<input type="radio"/> <input checked="" type="radio"/> <input type="radio"/> <input type="radio"/>

Figure 8: Specification of preferences: relative importance combined with selecting the criteria to be improved, relaxed, stabilized, and freed.

For some multiple criteria methods the MCA user, in addition to the relative criteria importance, specifies the second set of preferences. Namely, he/she indicates criteria to be improved, relaxed, stabilized, and freed. This is done by selecting (for each criterion) one the buttons in the second panel shown in Fig. 8. The meanings of the buttons (from left to right) are as follows:

- relax (worsen), i.e., allow to compromise the criterion value;
- free (i.e., direction and amount of the criterion value change is not important),
- button: stabilize the criterion value (preference for keeping worsening of the criterion value small);
- improve the criterion value.

The meaning of each button is displayed as a hint whenever the mouse points on the button. The gray area surrounding one button in each row shows the button value selected at the parent iteration (if this panel was used, otherwise the *free* values are marked).

The mapping of the selected buttons into parameters of the RFP-Pareto method is presented in [5].

This second set of preferences is used by the RFP-Pareto method together with the specified relative criteria importance, and the reference alternative, which is the Pareto solution found for the parent iteration.

While using this type of preferences, one should keep in mind the characteristics of the reference alternative. It is always possible to find another alternative which has a better value of one selected criterion unless the reference alternative has already the best

¹¹This example is taken from the analysis of future energy technologies, available from the MCA-Needs web-site (linked to the MCA documentation site).

possible value for this criterion. However, attempting to improve values of more than one criterion may not be possible, see Section 3.1. Moreover, an improvement of a criterion (or criteria) can only be achieved (if at all possible) by worsening values of at least one other criterion.

3.7 Pareto alternative for the specified preferences

By clicking on the *Solve* button the user actually creates a new iteration which is composed of the specified preferences and the corresponding Pareto alternative. The Pareto solution is selected as the result of solving the underlying parametric¹² optimization problem done by the solver the user selected as described in Section 3.5. Note that MCA is implemented in the client-server architecture, where the client is a browser through which the user accesses the MCA, and the server resides on one of the IIASA Solaris workstations. Thus the data representing the user preferences are stored in a DBMS on the server, then a selected solver reads the needed data, solves the optimization problem, stores the solution in the DBMS, and makes the solution available to the client. In this way a new iteration is completed, and all iterations made by the user can be accessed from any browser at any time.

As already discussed, in the current version of the MCA the specification of user preferences is done in a probably most simplest way, i.e., by setting relative criteria importance. However, there is a sort of price for such a simple specification of preferences. Namely, the lack of specification by the user of intra-criterion preferences. The latter is to some extent compensated by the pairwise outperformance measures and the reference-point methods (for details see [3] and [5], respectively) implemented in the solvers. Additionally, for one of the methods available in MCA (namely, the RFP-Pareto¹³), the user selects a criterion (or several criteria) that he/she wants to improve. Such an improvement can only be achieved (if at all possible) by worsening values of some other criteria.

Finally, we note that different solvers may find different Pareto alternatives for the same preferences. This is either a rare or an often situation, depending on the problem and the set of used solvers.¹⁴ Therefore, in some situations it is rational to try different solvers for the same preferences. Moreover, it is usually helpful to use in the advance stages of analysis the RFP-Pareto solver, for which preferences on criteria improvement/compromising are specified.

4 MCA basic structures and functions

This Section provides a step-by-step tutorial to the MCA. We first provide an overview of the MCA structure and functions, and then walk the readers through the whole process which starts with registration and login to the MCA, followed by specification of three types of entities, namely multiple criteria problems, their instances, and analyses of instances. After an analysis instance is prepared, the actual analysis process can start, and such a process is discussed in more detail using a tutorial example.

¹²Parameters are derived from the preferences specified by the user.

¹³RFP stands for reference point.

¹⁴Results of extensive tests can be found in [3] and [5].

The multiple criteria analysis process in the MCA is therefore composed of the following stages:

- Login to the MCA (see Section 4.3).
- Problem specification or selection (see Section 5.1).
- Specification or selection of an instance of a selected problem (see Section 5.2).
- Definition or selection of an analysis (see Section 5.3).

Before illustrating each of these stages by tutorial examples we summarize:

- Information on documentation and contacts with the developers in Section 4.1.
- Overview of the data structure, and the pre-loaded problems in Section 4.2.
- Navigation in Section 4.4.

In order to help in exploration of the MCA several problems are pre-loaded for each user. Each of these problems has an automatically created instance, which in turn has also automatically created analysis. Moreover, there are also tutorial analyses (with the preferences set and commented on by the authors) aimed at helping in becoming familiar with the MCA. Thus the users who prefer to explore the MCA through one of the predefined problems and/or tutorial analyses may want use the prepared data and preferences, and may therefore want to skip the details of Sections 5.1 through 5.3.

4.1 Contact, help and documentation

The MCA has been developed and is maintained by a very small team. Therefore we ask the user for understanding of implications of the limited resources and for help in improving the MCA.

The users are kindly asked to contact the MCA team through the *Contact* form (available in each MCA window, see Section 4.4). The form is composed of four simple to use elements:

- choice list of issue type (improvement, malfunction, question);
- summary: a concise title-like characteristics (e.g., editing loaded instance, or layout of the criteria chart);
- choice list of the issue priority (from the user perspective);
- description: actual description of the reported issue.

Information about the user,¹⁵ the id of the screen from which the form was launched, as well as other details (id of the problem, instance, analysis, iteration, type of the used browser) are passed to the data-base automatically, therefore it should not be included in the issue description. This makes using the *Contact* utility easy for the users and yet provide the developers with the information necessary for either replicating and solving the reported malfunction, or for answering questions, or for replicating the situation, if necessary for improving the MCA functionality.

The content of each completed contact-form is stored in a database. The developers will be notified and will try their best to address the issue as soon as possible. The provided and collected data will be used exclusively for addressing the reported issue.

The documentation of the MCA is organized into a simple Web-site linked to the MCA through the *Documentation* button (available in each MCA window, see Section 4.4).

¹⁵For the Contact used from the login screen this information is not automatically available, therefore if the user expects a feedback from the developers, then he/she should also fill-in the field labeled *Your email*.

4.2 Overview of the data structure

Each user can easily manage the data in her/his data space. The data is organized in the three-layer hierarchical structure, each layer being a container for one of the following three types of data entities:

- **Problems:** each problem is composed of alternatives and attributes (often also called criteria or indicators); a problem can be presented as a matrix having:
 - ★ rows corresponding to alternatives,
 - ★ columns representing to attributes,
 - ★ elements defining values of attributes for each alternative.
- **Instances of a problem:** each instance is specified by defining a set of criteria; each criterion is defined by attaching a criterion type (maximized, minimized, target) to a selected attribute.
- **Analyses of an instance:** each analysis is a container of iterations; each iteration is composed of the preferences specified by the user and the corresponding solution consists of:
 - ★ Pareto alternative that fits best (according to the selected analysis method) to the specified preferences,
 - ★ optional ranking of alternatives,
 - ★ some characteristics in the criteria and alternative spaces.

The user may enter the process at any stage by selecting a particular problem, one of instances of a selected problem, and one of analysis of a selected instance.

At any of these stages the user may create a new entity (i.e., a problem, an instance, and an analysis). New problems and instances can be created in two ways:

- by modifications of a selected problem or instance, respectively;
- by a new specification of all data required for the corresponding entity.

The user may also continue with an analysis that either he/she has done earlier or has been pre-loaded for him/her upon creation of his/her data space.

Therefore, the MCA data structure and the corresponding functionality allows for an easy and effective management of the data space, and helps in structuring the analysis process by users who use the MCA intensively. It is also easy to be used by those who want to either analyze only one particular problem, or even only explore the MCA.

4.2.1 Pre-loaded problems

The following tutorial problems are pre-loaded for each user for an easy start with the MCA, and safe exploring of both the MCA functionality and the multiple criteria analysis.

The problems (and the corresponding instances and analyses) pre-loaded into the general-purpose MCA are toy examples and do not correspond to actual problems or objects. The values of attributes have been set in order to either illustrate various features of multiple criteria analysis, or to test different analysis methods. Currently, the following problems are pre-loaded:

- *Home*, a comprehensive choice problem, two versions (called *home*, and *home-realistic*, respectively) are defined for basic and advanced analysis discussed in detail in Sections 6.4 and 6.5, respectively;
- *Car*, an extension of the classical example of a car choice; two instances (called *simple* and *hierarchical*) are predefined for the one-level and a simple hierarchical criteria

structures, respectively;

- *Success Ltd*, a very simple problem of a company management (described in [8]) for exploring basic concepts and features;
- *Spectrum*, the spectrum management problem described in [8].

Moreover, four problems loaded into the dedicated version called MCA-Needs are copies of the actual analysis of future energy technologies developed within the Needs project [6] for four European countries. These problems use a rather complex hierarchical structure of criteria.

4.3 Login to MCA and MCA-Needs

Figure 9: The MCA login form.

Access to any of the IME applications is available for each registered user (registration procedure is summarized in Section 4.3.1) through the same pair of {Nick-name, PIN-code}. Access to the MCA Web-site is described in Section 2.1, and the login form is shown in Figure 9. The MCA login page contains also hints on the login procedure, and information about the requirements for Web-browsers for which the MCA design and implementation has been optimized.

A session is created for each user after a successful login. This takes some computing resources, therefore the users are kindly asked to *logout*¹⁶ whenever they do not actively use the MCA; in this way more resources are available for other users. Additionally, there is a timeout procedure implemented; it invalidates sessions for which there was no activities during some period of time.

4.3.1 Registration of new users

New users of the IME applications are kindly asked to complete the registration procedure which is available through the *New user of the IME applications* link available in the login form shown in Figure 9.

The prospective users are kindly asked to review the conditions (available in the registration form) for using the IME applications, and proceed with the registration only if they agree with these conditions.

The first field of the registration form is the *Nick-name*. Each new user may select any nick-name of her/his choice provided that it conforms to the conditions specified in the information box of the registration form. The other fields on the registration form are self-explanatory.

¹⁶See Section 4.4.

The registration procedure is automatized, but for security reasons composed of several stages, two of which require the user action:

1. Completion of the registration form.
2. Activation of the registration (activation code and information how to use it are emailed to the user after the registration form is successfully submitted).

4.3.2 Managing PIN-codes

Each registered user is identified by the IME applications through his/her email. The first PIN-code associated with each registered email is randomly generated and emailed automatically.

Should the user forget his/her PIN-code, he/she use the link *Re-make my PIN* available from the login form (see Figure 9), and fill-in the form requesting a new randomly generated PIN-code.

The user may also want to use the *Change my PIN* link to the request form through which it is possible to define a PIN-code. A user-defined PIN has to conform to the basic security requirements specified in the information box in the request form.

4.4 Navigation through the MCA

The MCA is a rather complex application yet its control has been made easy also for less experienced users. Navigation through the MCA is done through selecting a button that invokes the corresponding action. The buttons controlling similar type of actions are organized into sets and the corresponding panels discussed below. Additionally, whenever desired, relevant information is displayed.

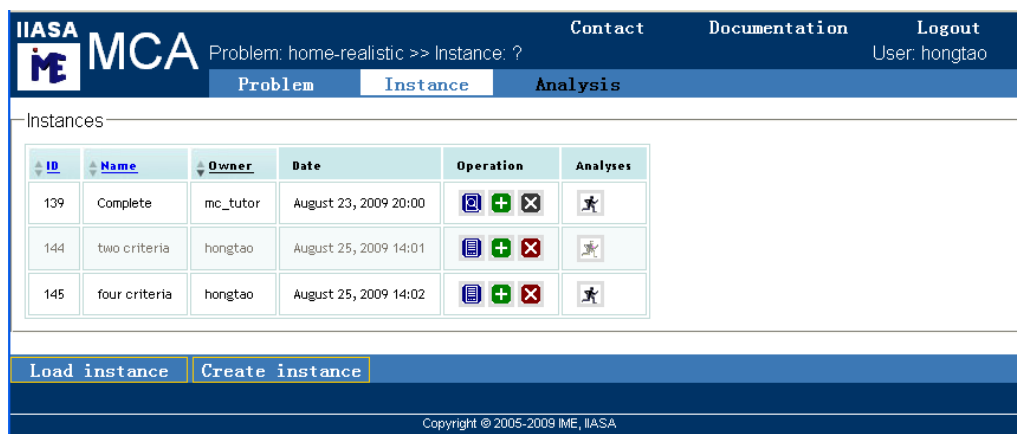


Figure 10: Navigation through MCA: instance specification screen.

We summarize now the basics of the navigation using the example of the instance specification screen illustrated in Figure 10.

Most of the MCA controls are organized into three parts:

- the upper control panel,
- the bottom control panel,
- working area between these control panels.

The working areas differ according to their function, although similar functions are implemented in a similar way; therefore the organization of the working areas is discussed separately for each stage of the MCA, see Sections 2.2.1 through 6. We summarize below the organization of the upper control panel (which is the same for almost all MCA screens). The functionality of the bottom control panels and the working area are discussed in Section 5.

4.4.1 Upper control panel

The upper control panel shown in Figure 10 is composed of two one-line menus and the status line between them.

The top menu-line is composed of the following controls with the following functionality:

- Contact: launch the contact form described in Section 4.1.
- Documentation: link to the MCA documentation site, see Section 4.1.
- Logout: finish working in the MCA, see Section 4.3.

The lower menu-line of the upper control panel is composed of three controls labeled:

- Problem
- Instance
- Analysis

Each of the latter controls provides access to the list of entities of the corresponding stage of the specification. Note that the stages of specification have hierarchical structure, therefore the available controls are located on the left side of the currently active stage. The currently active stage is marked by the white background, and one also can select this button; non-selectable controls are inactive and displayed in black color. Figure 10 illustrates the situation when one can select two of the three stages (the currently selected middle button has the white background). Figures in the following Sections illustrate other situations.

The status line contains information about the selected problem, instance, analysis, and iteration. Not all these elements are always available; whenever this is the case, the status line is composed of the available elements, and the first one not yet available is marked by the question mark. In Figure 10 only information on the selected problem is displayed. Figures in the following Sections illustrate situations when either partial or full information is available.

5 Problems, instances, and analyses

5.1 Problem specification

The interface to the specification and/or selection of a problem has two-layer hierarchical structure. The top layer is composed of the working area (Section 5.1.1) and the bottom control panel composed of controls for the bottom layer (Section 5.1.2); The bottom layer serves specification of new problems; its functions are described in Section 5.1.3.



Figure 11: Problem specification and selection screen.



Figure 12: Icons used for controlling the actions.

5.1.1 The working area

The working area of the main screen for problem specification and selection (located between top and bottom control panels shown in Figure 11) contains the list of previously specified problems that are available for the user. This list is composed of six columns.

The first four columns contain for each problem its id, name, owner and date of creation. Three of these columns have clickable headers, which allows sorting the whole list according to the selected header.

The two other columns are composed of two types of controls:

- Operation: selection of one of the four possible icons initiates one of the following actions on the corresponding problem:
 - ★ edit (and optionally commit) the not yet committed problem,
 - ★ view the already committed problem,
 - ★ add: create a new problem by modification of the selected problem,
 - ★ delete the problem.

Note that the first two actions are mutually exclusive, therefore the operation icons are organized into three columns. The icons corresponding to the *view*, *add*, *delete* actions are shown in the left box of Fig. 12.

- Instances: clicking on the only one icon in this column provides access to the instances of the corresponding problem. The instances can only be created for committed problems, therefore the *Run* icon (shown in the right box of Fig. 12) is grayed (i.e., inactive) as long as the corresponding problem is not committed.

The readers not interested in creating new problems may want to move now to Section 5.2.

5.1.2 Bottom control panel

The bottom control panel is composed of two controls for specification of a new problem:

- Load a problem. Selection of this control will initialize loading problem specification from a csv-type file. The user will be asked to browse files on her/his computer, and select a file with the problem specification. Loading problems (typically combined with specification of an instance) is recommended for large problems only. The structure of the csv-file with problem and instance specification is defined in Appendix A.
- Create a problem. Clicking on this button initializes interactive specification of a new problem; this is explained in next Section.

5.1.3 Specification of a new problem

Attributes	Price	Fuel
Alternatives		
Toyota	22.0	8.0
Honda	19.0	11.0
Volvo	24.0	7.0

Figure 13: Specification of a new problem.

The interface to specification of a new problem is illustrated in Figure 13. The user accesses such a screen if one of the following actions (described above) is selected:

- Create a problem.
- Add a problem.
- Edit a problem; note that this action is also available for problems loaded from csv-files.

The only difference between the first and the other two situations is that a problem creation starts with empty lists of attributes and alternatives while in the other two ways of problem specification these lists (and also the attribute values) are inherited (from the problem that has been selected or uploaded, respectively). Figure 13 shows an example with several attributes and alternatives already defined.

The problems are specified and modified through several linked forms. There are three buttons in the bottom control panel with the same functionality on each MCA form. These buttons and the corresponding actions are as follows:

- Save: save the modifications made to the form since the previous save.
- Cancel: ignore the modification.
- Commit: first save the modifications, and then flag the problem as defined. The latter implies that no further modifications of the problem specifications (except of its name,

and the note) are allowed. Note that only committed problems can be used for specification of instances.¹⁷

Clicking on one of these buttons causes the execution of the corresponding action, followed by returning to the screen from which the form was initiated.

Each problem is in MCA specified in the form of matrix having named columns and rows corresponding to attributes and alternatives, respectively. The elements of the matrix represent the corresponding values of attributes. Additionally, each problem has a unique name, and an optional note.

New attributes and/or alternatives can be added as long as the problem is not committed. This is done by one of the first two button in the bottom control panel:

- Add attribute opens a simple form for specification of four elements of a new attribute:
 - ★ short name,
 - ★ name,
 - ★ description,
 - ★ unit.

The fields that must be filled-in are marked by the red asterisks. There is also a simple choice list for adding attribute to the list of already defined attributes.

The new attribute form has two buttons (Save and Cancel) in the bottom control panel. Selecting one of them:

- ★ either saves (i.e., actually creates alternative with the defined data) or ignores the data entered into the form, and then
- ★ returns the control to the screen from which the form was initiated.
- Add alternative has the same functionality for alternatives as the above described button for attributes. The only difference is that alternatives have no measurement units.

Values of attributes can be modified directly in the displayed matrix. The values in the matrix are initialized to those inherited from either the problem for which the *Add* function was used or the uploaded csv-format file. For each newly created attribute (or alternative) a new column (or row) is generated with values of all alternatives set to zero.

The already defined alternatives, attributes, and their values can be modified as long as the problem is not committed. Modifications of alternatives and attributes can be done through clicking on the corresponding name. This generates the corresponding form with the functionality similar to the forms handling definition of new attributes and alternatives. The forms for modification differ from the corresponding forms for specification by the following two elements:

- The *Remove* button is added to the bottom control panel to allow for removing the selected alternative or attribute. If this button is selected then the corresponding row (for alternative) or column (for attribute) is removed from the matrix defining the problem.
- The form is filled by the previously specified data.

The problem name and the note can be modified also after the problem is committed.

5.2 Instance specification and selection

An instance is always defined for a selected problem, therefore the only way to access the instance specification and selection screen is to click on the *Run* icon of the correspond-

¹⁷Therefore the *run* icon (in the problem specification/selection screen shown in Fig. 11) remains grayed (inactive) until the corresponding problem is committed.

ing problem, see Section 5.1.1. From the corresponding screen the user gets access to the already defined instances for the selected problem, and can also specify for it a new instance.

Attribute	Short name	long name	Type
<input checked="" type="checkbox"/> Price [Euro]	Price	Price	<input checked="" type="radio"/> Min <input type="radio"/> Max
<input checked="" type="checkbox"/> Fuel [l/100km]	Fuel	Fuel	<input checked="" type="radio"/> Min <input type="radio"/> Max

Figure 14: Instance specification form.

The locations and functions of controls in the instance screen are exactly the same as for the problem screen therefore we refrain from repeating the corresponding descriptions. There is however one obvious substantial difference between instance and problem specification, namely creation and editing of the corresponding entity. Therefore we shall now summarize creation and editing of an instance. The corresponding form is shown in Figure 14. As already discussed, an instance is defined by selecting attributes that are used as criteria, and selecting for each criterion its type. Selection of attributes are done by checking the corresponding check-boxes. Optionally, one can define criteria names (the names are initialized to the corresponding attribute names). Selection of the criteria types are done by clicking on the corresponding radio buttons.

Similarly to the specification of a problem, the values in the form are initialized, if the specification or editing is initialized by either the *Add* button or by definition uploaded from a csv-format file.

Currently, specification of instances with the hierarchical criteria structure is possible only through a corresponding csv-format file. Instances inherited from those with hierarchical structure are generated in such a way that only the leaf-level criteria are retained, and the higher level criteria are ignored. In other words, inherited instances have a non-hierarchical (flat) structure of only those criteria for which the corresponding attributes have been defined.

5.3 Analysis specification and selection

An analysis is always associated with a selected instance, therefore the only way to access the analysis specification and selection screen is to click on the *Run* icon of the corresponding instance, see Section 5.2. In this way, the user gets access to the already defined analyses for the selected instance, and can also specify for it a new analysis.

The locations and functions of controls in the analysis screen are exactly the same as discussed above for the problem and instance screens. However, a specification of a new analysis is much simpler. One needs to only specify a unique name for the analysis, and an

optional note. By saving these data a new analysis is created. These data can be modified at any time, therefore there is no *Commit* function for the specification of analysis.

6 Analysis process

In this Section we illustrate the multiple criteria analysis methodology presented in Section 3 by a step-by-step example of analysis of a fictitious but yet realistic problem. We suggest the readers to become familiar with the methodology discussed in Section 3 before making any actual analysis, and to refresh the methodological background while proceeding with the tutorial examples.

6.1 Controlling the analysis process

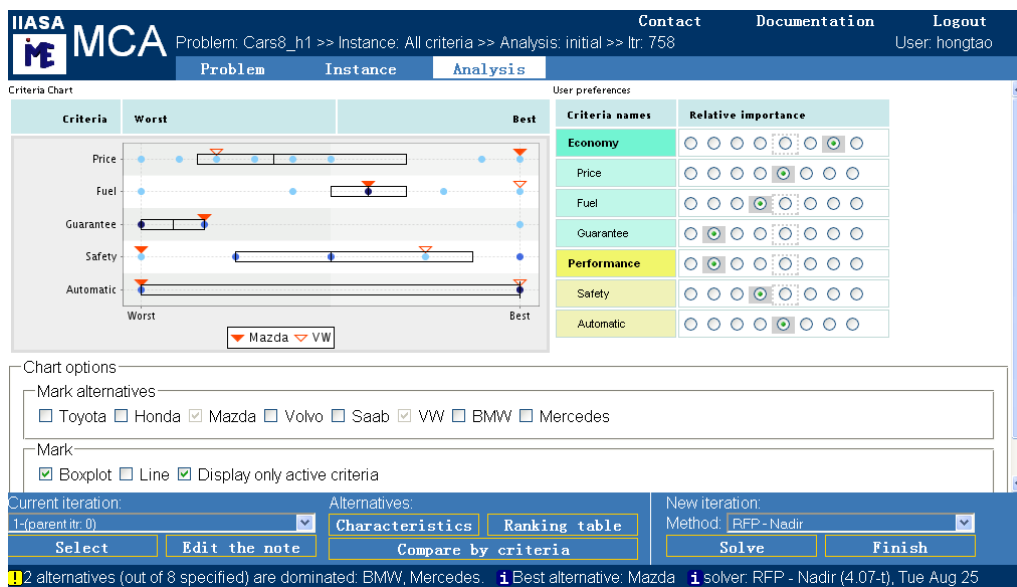


Figure 15: The main window of the analysis.

After an analysis has been selected the main window for the interactive analysis shown in Figure 15 is displayed. As all MCA screens it is composed of upper and bottom control panels and the working area between them.

The functions of the upper control panel are described in Section 4.4.1 therefore here we only mention that:

- The status line displays names of the currently analyzed problem, its instance, and selected analysis of the instance, as well as the id of the current iteration.
- The controls of this panel can be used for getting access to the problem and instance data, as well as for switching to another analysis (of the current instance), or another instance (of the current problem), or another problem.

The methodology corresponding to the elements located in the working area, as well as those available in pop-up windows is presented in detail in Section 3. In the following Sections we show how this methodology works by illustrating it by walking through a

typical process of analysis. This tutorial is split into initial/basic and advanced stages of analysis. However, in order to help the user in following such a process we need to first explain how to control the process through the bottom control panel.

Let us recall that the multiple analysis process is in the MCA composed of iterations, and each new iteration is composed of:

- Selection of a parent iteration.
- Analysis of the criteria values for the parent iteration.
- Optionally,¹⁸ selection of a specific MCA method to be used by the MC solver for computing a Pareto solution.
- Specification of preferences to be applied in the new iteration.
- Finding the corresponding Pareto solution.



Figure 16: Control panel of the analysis main window.

These elements are discussed in detail in Section 3. Now we show how the analysis process can be controlled by the user. This is done through the controls organized in the bottom control panel shown in Figure 16. It is composed of three sets of controls:

- Current iteration.
- Alternatives (of the current iteration).
- New iteration.

6.1.1 Selection and analysis of the current iteration

Let us recall that for each analysis instance (usually called *analysis*) the initial iteration is generated automatically (with equal preferences for all criteria selected for the corresponding model instance). Each user-created iteration starts from a previously done iteration that is selected by the user to be the current one. If preferences of a current iteration are modified and a new iteration is created, then the current iteration is considered to be also the *parent iteration* of the new iteration because it provides initial values of the preferences to be specified for the new one.

After an analysis instance (either newly created or an old one) is started, the newest (the last defined) iteration is automatically selected to be the current iteration. However, the user can easily select another iteration to be the current one.

Managing the iterations is supported by a group of buttons located in the left part of the control panel shown in Figure 16 that support the following actions:

- Selecting a current iteration. By default the newest iteration is displayed. The user can change it by selecting another iteration from the choice list (labeled *Current iteration*). The items on the list are composed of:
 - ★ a sequence number (with the iterations made for the current analysis),
 - ★ initial part of the note (see below) that is attached to the number in order to help the user to identify the iterations easier.

¹⁸The last selected method is used, if the user does not select another method.

A choice of another (than currently displayed) iteration has to be confirmed by the *Select* button.

- Edit the note attached to the currently selected iteration. The user may replace the default text (set to the id of the parent iteration) by any comment. The modification is done in a pop-up window, and has to be confirmed by the *Save* button available in this window. Each new note overwrites the previous one.

Note that the solver used for the current iteration is displayed as the default solver choice for the new iteration.

The analysis of the current iteration is a crucial element of multiple criteria analysis, and is therefore discussed in detail in Section 3.4. Here we only mention that the main analysis focus is typically on the criteria trade-offs which are represented in the chart displayed in the main working area. Analysis of this chart are discussed also in the tutorial examples that follow.

Access to the auxiliary analysis in the alternative space (discussed in Section 3.4.2) is provided by the three controls in the middle group of the control panel labeled:

- Characteristics.
- Ranking table.
- Compare by criteria.

By clicking on any of these buttons one can display the corresponding chart in a pop-up window. The windows contain information about the corresponding iteration, therefore more experienced users can easily compare characteristics of solutions corresponding to several iterations.

We sum-up this Section with an obvious but important observation. A selection and analysis of a current iteration may provide a good basis creating a better (in the sense of the criteria trade-offs fit to the user preferences) child iteration. However, there are no rules that help to judge whether or not an iteration is a good candidate for a parent iteration. Thus, selection of the current iteration requires a combination of three elements: understanding of the multiple criteria analysis methodology, experience, and intuition.

6.1.2 Creating a new iteration

The actual challenge in creating a new iteration that can be a valuable element of the analysis process is in specifying the preferences that will result in selecting a Pareto solution having criteria trade-offs that fit well the user preferences. Actually, specification of preferences requires the same combination of key elements as mentioned above, and should be made in accordance with the selection of the parent iteration.

Technically, creation of a new iteration is very easy, and is composed of three steps:

- Decide, if the solver should be changed. To help in this one may recall the discussion in Section 3.5. One of the available solvers can be selected from the choice list in the right group of controls of the bottom control panel.
- Specify the preferences keeping in mind the methodological background summarized in Section 3.6.
- Click on the *Solve* button located below the solver choice list.

The new iteration is actually created by clicking on the *Solve* button, and is composed of the specified preferences and the corresponding Pareto alternative. During creation of the new iteration and execution of the solver the MCA main screen is inactive. Depending on the size of the problem and the speed of the connection between the server

and the user Web-browser creation of a new iteration (including the solution time) takes between few seconds (for small problems and fast connections) up to a fraction of one minute (for problems with over 1000 alternatives).

6.1.3 Finishing the analysis

The analysis process of the current analysis instance can be finished by:

- clicking on the *Finish* button located next to the *Solve* button; in such a case the user will next see the window with analysis instances;
- selecting one of the two buttons in the upper control panel, namely *Problem* or *Instance*; this is typically done for changing the problem or instance, respectively;
- *Logout* from the MCA;
- The session time-out (see Section 4.3).

Note that finishing the analysis process does not create a new iteration. Therefore any modifications of the preferences made to the current iteration will be lost. Such preferences will also be lost, if another iteration will be selected as the current one. In other words, in order to store preferences one needs to create the corresponding new iteration by clicking on the *Solve* button.¹⁹

6.1.4 Messages from the solver and the user interface

The area below the control panel is used for displaying messages generated by the solver and by the user interface. Each message is prepended by an icon, color of which indicates the type of message:

- blue - information; typically three information messages are displayed:
 - ★ name of the best alternative selected by the solver,
 - ★ solver version, and the date of solution;
- yellow - warning;
- red - error (either in the specification of preferences, or in the solution process); should the error(s) be not self-explanatory, then please use the *Contact* button (in the upper control panel) to clarify the problem with the MCA developers.

6.2 Tutorial example

Before walking through the tutorial analysis we summarize the illustrative example that is used in this tutorial. Let us assume that we consider buying a new apartment or house, and we consider the basic characteristics of a new property by the attributes given in Table 2.

The considered alternatives are summarized in Table 3. The values of attributes for these alternatives are available in the on-line tutorial (in the *Problem* specification of the *Home* problem), as well as in the *home.csv* file available from the MCA documentation Web-site.

¹⁹This software design decision was made to assure the consistency, i.e., that each specified set of preferences has the corresponding solution.

Short name	Description	Units
Price	Total price (incl. renovation)	kEuro
Downpay	Downpayment (incl. renovation)	kEuro
OMC	Yearly operational and maintenance (excl. credit) costs	kEuro
Impression	Family/individual overall look-and-feel impression	grade
Size	Size of the apartment/house	sq. mtr
Design	Rooms, kitchen, bathroom(s), balcony, garden, cellar, etc.	grade
Amenities	Heating/air-cond., convenience (lift, cellar, garbage)	grade
Parking	Garage/parking (0-extremally difficult, ..., 4-easy, 5-private, 7-10 garage)	grade
Delay	Availability (incl. renovation)	months
House	House (1) or apartment(0)	binary
Ground	Size of the property associated ground	sq. mtr

Table 2: Attributes of candidates for our new home.

Nickname	Short description
Market	Market square, middle of the center
Main str.	Central district, busy street
Back str.	Central district, quiet street
Park str.	Close to center, quiet place
Residence	City-residential area, apartment houses
Daisy str	Sub-urban apartments
Mellow str	Sub-urban apartments
Cozy-corner	Remote sub-urban apartments
Chateau blue	City-residential area, exclusive houses
Noble hut	City-residential area, exclusive houses
Green hills	Sub-urban area, sparsely populated
Forest nest	Remote location, close to natural park
Country seat	Renovated old farmer's house
Gulf sunbed	Remote location, recreational area
Boat	Floating home
Mobile	Luxurious camping car
Adventure	Simple camping car
Tent	Simple tent (just a joke)

Table 3: Nicknames and the corresponding descriptions of alternatives for our new home.

6.3 Accessing the tutorial analysis

The problem specification summarized in Section 6.2 has been prepared in the file named `home.csv` available in the MCA documentation Web-site. For this tutorial the file has been uploaded (using the *Load problem* from the *Problem* screen), the specification contained in the file was committed without any changes, then icon *run* corresponding to the problem named *home* selected. The latter action moved the flow to the *Instance* screen on which the instance named *Basic* was created interactively (using the *Create instance* button). Next, the *run* icon was clicked to create the analysis named *Tutorial*.

The reader can simply use (after successfully login to the MCA, as described in Section 4.3) the prepared analysis by clicking the corresponding *run* icons on the sequence of screens corresponding to *Problem*, *Instance*, *Analysis*. If the reader will follow this sequence then she/he will see the last iteration made in the tutorial analysis. However, it is easy to switch to the screen shown in Figure 17 by selecting the 0-th iteration to be the

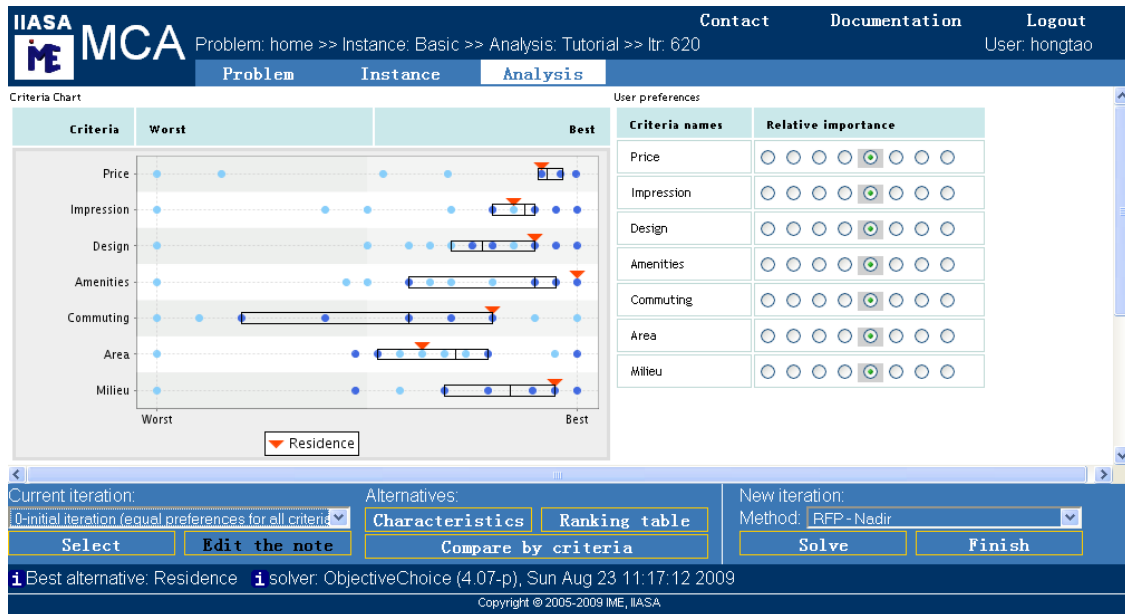


Figure 17: Initial iteration of the basic analysis.

current one.

One can also have a look at the form for creating a new instance from a copy of the *Basic* instance; this form can be displayed by clicking on the corresponding *add* icon in the *Instance* screen. Alternatively, the users may want to copy the csv-file from the MCA Web-site to their computer, modify it, and then use it for experimenting with the MCA.

6.4 Basic analysis

Let us comment step-by-step on each tutorial iteration defined in the basic analysis. Therefore we start with the initial iteration illustrated in Figure 17.

6.4.1 Setting the stage

For the basic analysis a subset of seven criteria was selected. The names of the criteria are shown in the *User preferences* panel,²⁰ and their descriptions can be seen (one by one) as hints displayed whenever the users points the mouse to the corresponding criterion name. As already noticed, the preferences for the initial iteration are set automatically to be equal for all criteria, and the *Objective Choice* method is used for finding the corresponding Pareto alternative.

The main source of information about the criteria and the selected solution is the *Criteria chart*; a good understanding of elements of this chart is essential therefore we suggest to refresh the its detailed description contained in Section 3.4.

Especially the initial iteration requires some time for reflecting on the characteristics of the problem. One usually starts with analyzing the distributions of values for each

²⁰Right part of the screen shown in Figure 17.

criterion. It is easy to notice that the two criteria shown at the top of the chart (*Price, Impression*) have very narrow range (up to 10%) range of the two middle quantiles (which means that values of the corresponding criterion for half of the alternatives are within the 10% of the range of all values. Moreover, the lowest quantiles cover 80% and 90% of the corresponding range of values, respectively; this suggests that 75% of alternatives are very good values of these criteria. The other four criteria also appear to have rather favorable distributions of values. Only the *Commuting* criterion has rather typical distribution of values, i.e., the lowest and highest quantiles cover 20% of worst and best values, respectively; thus the middle two quantiles cover 60% of middle values.

The Pareto alternative of the initial iteration is usually called a compromise solution. On the criteria chart the solution for the current iteration is always marked by the solid red triangle, and in our example, the corresponding alternative called *Residence* has also rather good values of all criteria (the relatively worst criterion *Area* has the 63% of the corresponding best value).

Considering the trade-offs in such a simple way one may conclude that the first choice is rather good, and combining such a hypothesis with the analysis of value distributions summarized above one believe that a rational choice of our new home will be pretty easy, and the solution will likely have pretty good values of all criteria. In the course of actual analysis we will see that neither the hypothesis is necessarily true nor a more informed choice will be easy.

6.4.2 Simple trade-offs in criteria space

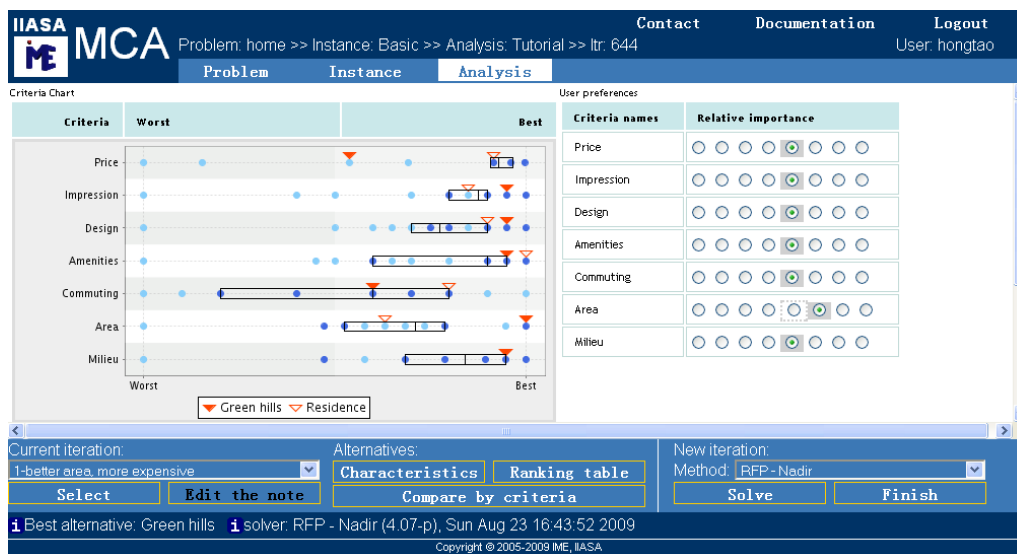


Figure 18: Initial iteration of the basic analysis.

Let us assume that after analysis of the initial iteration we want to improve the worst-performing criterion, called *Area* (see the hint for its definition). To see how this was attempted please select as the current iteration the 1-st iteration. After this will be done you will see the criteria chart and the user-preference panel shown in Figure 18. By

considering that shadow-markers of the user-preference panel (described in Section 3.4) it is easy to notice that the importance of the *Area* criterion was increased by one level.

It is easy to compare the attributes (criteria values) of the current and previous solution as both are marked on the chart. Therefore we notice that the new choice *Green hills* has the best possible value of *Area* but considerably worse values of two criteria, namely *Price* and *Commuting*. We also notice that the value of the price is in the middle of the lowest quantile, and that we have better chances to improve one criterion than two at a time, therefore let's try increase our preferences to the *Commuting* criterion only.

However, before moving to a next iteration, it good to note observations about each iteration by using the *Edit the note* button. The authors have done while preparing this tutorial, therefore the 1-st iteration has an informative label. You may also notice that the initial iteration has id equal to 620, while the first iteration has id = 644. This is because there was a break in writing this tutorial, and meanwhile another user made over 20 iterations. The iteration id is not important from the user point of view, it is needed only for tracing and possible problems. From the user point of view only the iteration sequence numbers and the corresponding (user-defined) comments are important.

To see the results of attempting to find a place with better commuting please select as the current the iteration number 2. Now we see that we got a much cheaper solution called *Boat*, but the commuting attribute did not improve, and all but price attributes are worse than for the previously found *Green hills*. So let's try to improve the *Area* again but setting a yet higher importance to it. The results can be seen in iteration 3. Namely the same alternative was selected. At latest now we shall realize that there is no cheap alternative with very good values of most alternatives, so let us explore more expensive solutions by slightly relaxing importance of the price criterion. The results can be seen by selecting iteration 4 to be the current one.

The newly found solution *Chateau blue* has the best possible values of four criteria, and one criterion with the second best (95% of the best) value. Also the six-th criterion (commuting) is has 70% of the best value. The "only" problem is that this solution is the second most expensive.

Let us now comment on the other elements of the analysis before making preliminary conclusions from the experience gained so far.

6.4.3 Using the chart options and notes

The results of the fourth iteration (discussed in Section 6.4.5) show the advantage of marking up to two selected alternatives. Therefore we now summarize the available options of the criteria chart controlled by corresponding selections in the *Chart option* panel (see Figure 19):

- Selecting up to two alternatives values of which will be marked on the chart by the blue triangles. Note that it is not rational to use such markers for alternative(s) already marked by the red triangle(s) therefore such alternative(s) cannot be selected for marking. Moreover, if more than two alternatives are selected then only the first two (on the list) are marked in the graph.
- Hiding the boxplot.
- Connecting the values of the selected alternatives by lines.
- Hiding the inactive criteria.

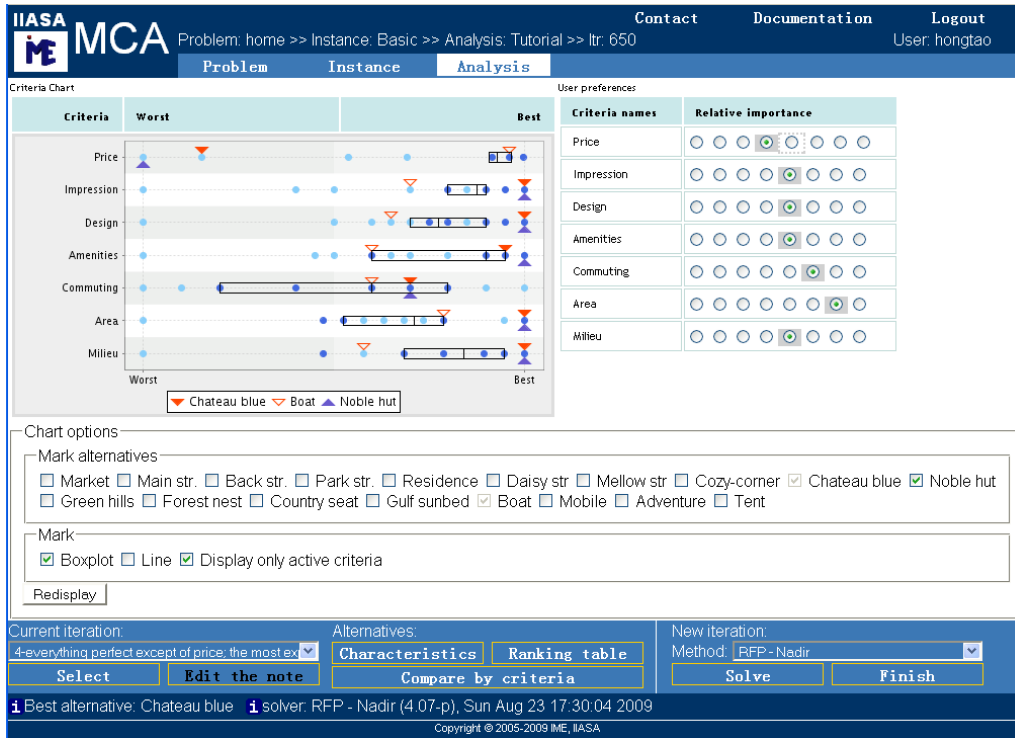


Figure 19: Criteria chart and preferences for the fourth iteration.

Selected chart options have to be confirmed by clicking on the *Redisplay* button. The selected options remain valid for the session unless the user resets them.

Notice also that the note for each iteration in this tutorial has been written by the tutorial authors. Such notes are very useful in finding an appropriate iteration in the set of iterations belonging to each analysis. Therefore the authors strongly recommend the users to use the note corresponding each iteration for at least helping to identify the iteration.

6.4.4 Analysis in the alternative space



Figure 20: Comparison of expensive alternatives.

We illustrate now the role of the auxiliary analysis in the alternative space (discussed in Section 3.4.2) by commenting on the comparison of the four most expensive houses shown in Figure 20. The user may see the same Figure on her/his screen by:

- selecting the iteration 4 as the current iteration;
- clicking on the *Compare by criteria* button;
- dragging the *Forest nest* alternative from the *Not selected* window to that labeled *Selected for display*;
- arranging (by dragging) the alternatives in the *Selected* window into the order shown in Figure 20.

On the left side of the graph one can see the performance of the three best (for the reference alternative, for which the *Chateau blue* was selected) criteria, and in the right-side panel the performance of the worst four criteria is shown. The results of the comparison of the four most expensive alternative is consistent with the intuition. However, the user may want to continue this analysis by choosing different sets of alternatives (and their order in the *Selected* window).

6.4.5 Final steps of the basic analysis

Coming back to the analysis of iteration 4 we can observe that by marking the most expensive alternative (*Noble hut*) it is easy to conclude that there is clear trade-off: for additional 1 mln€ we can get a slight improvement of only one criterion (*Amenities*). However, we shall remember that the trade-off is more complex, if the full set of criteria is considered. Assuming that most readers might be interested in exploring moderately priced options let us now change our preferences by increasing the importance of the price criterion by two levels (from the less important to the more important than average), and decrease by one level the relative importance of the *Area* criterion. Generally, it is recommended to attempt to control the new trade-offs by specifying which criterion we want to compromise in order to improve the selected criterion (or criteria).

The results of this change of preferences can be seen by selecting iteration 5 to become the current iteration. We may now recall that we have seen the *Boat* solution, and nothing interesting can be learned from this iteration. Therefore after editing the note the authors of the tutorial went back to the previous iteration, and made a stronger increase (to the second highest) of the importance of the price criterion, as well as lowering the importance of the *Area* criterion.

The results of these preferences can be seen by selecting iteration 6. We got the almost cheapest solution but, given the values of other criteria, it is unlikely that our family would actually like to seriously consider it. However, just for an analytical exercise let us try to get the really cheapest solution. This can be achieved by the preferences set for iteration 7.

As the final step of the basic analysis let us illustrate the analytically obvious property of the relative criteria importance. Most of the users at some point try to set all (or almost all) criteria to be the most important. Such an experiment is illustrated by iteration 8. It is easy to recall and check, that the solution is the same as for the initial iteration for which also equal relative importance was set. Readers with analytical skills will most likely notice that this however cannot be guaranteed. There is one more condition that need to be fulfilled for guaranteeing that the same Pareto alternative is selected for all importance set to a same level (irrespective of this level). Solution of this little puzzle is however left to the readers.

6.4.6 Towards preliminary conclusions

We stop making more iterations for the tutorial basic analysis now, and leave it to the reader whether to continue it, or two switch to the advanced analysis.

The motivation for considering now another analysis of the same problem comes from the observation that continuation of the analysis of all alternatives using the selected subset of criteria is not very promising. In actual situations one also makes at least two stage analysis. First, a preliminary analysis of a possibly wide set of alternatives using several key criteria, followed by the second stage with a smaller number of alternatives by a wider set of criteria.

In our case we will continue of analysis of reasonably priced alternatives, where *reasonably* stands for excluding the four qualitatively most expensive properties, as well as the three alternatives that are very cheap but not really acceptable. We will also include to the analysis all criteria for which the corresponding attribute values are available.

6.5 Advanced analysis

6.5.1 Preparation of the analysis

The problem (called *home-realistic*) specification according to the comments summarized above has been prepared by using the *add* option of the *Problem* specification screen. We briefly comment now on using this option, but the reader can quickly start the advanced analysis (from any iteration of the basic analysis by following the sequence:

- click on the *Problem* button (in the top menu),
- click on the *run* icon corresponding to the *home-realistic* problem,
- click on the *run* icon corresponding to the *Complete* instance,
- click on the *run* icon corresponding to the *Tutorial* analysis.

The *home-realistic* problem was created from the original *home* problem by using the *plus* icon, and then editing the copied problem by removing (one by one) the seven alternatives we decided to discontinue to consider. Then the name of the problem was changed, and the modified problem was committed. Next a new instance and a new analysis have been created for the newly created problem and instance, respectively; the way for doing this have already been discussed.

If the reader will follow the sequence of clicks described above, then she/he will see the last iteration made in the tutorial advanced analysis. However, also here it is easy to switch to the screen shown in Figure 21 by selecting the 0-th iteration to be the current one.

6.5.2 Preliminary considerations

The criteria chart shown in Figure 21 illustrates the distributions of criteria value that considerably differ from the corresponding distributions of the basic analysis shown in Figure 17. The distributions of the currently considered problem are more common (than those of the basic analysis) for advance analysis which often concentrates on analysis of a subset of "more realistic" alternatives, i.e., the alternatives which have ranges of criteria values (e.g., prices in our case) that are considered to worth to consider. However, even for such a set of alternatives, some criteria may have bi-modal (or multi-modal) distributions

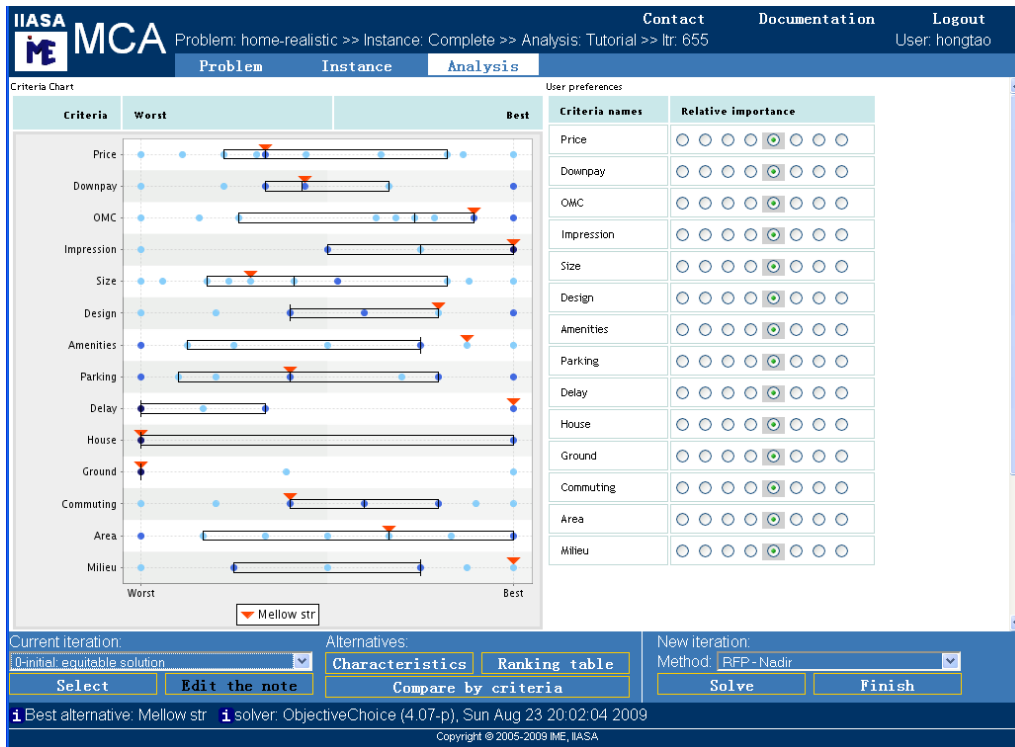


Figure 21: Initial iteration of the advanced analysis.

of values. In our case such a situation is illustrated by two criteria, namely *House* and *Ground*.

As we have seen, including in the analysis alternatives with extreme values of criteria (in our case, very expensive and very cheap alternatives) make the analysis very difficult. To illustrate this point one may want to compare the distributions of the *Price* criterion values shown in Figure 17 and Figure 21. It is clear from the analytical point of view that analysis of the trade-offs between values that differ very little is very difficult, and, for really small differences, impossible. We also note that although we have excluded from the current analysis extremally priced (both expensive and cheap) alternatives, we still consider a wide range of prices (the ratio of the most expensive to the cheapest is 4).

We have included in the analysis 11 alternatives and 14 criteria. While a large number of alternatives does not make the analysis much more difficult (comparing to a smaller number, which is still large enough to make pairwise comparison not practicable), a large number of criteria is difficult to analyze in a comprehensive way (assuming that aggregation of criteria is not acceptable). There is a commonly accepted observation that most humans can analyze 5-9 things at a time, therefore most analysts and practitioners concentrate on analysis of less (often substantially less) than 10 criteria at a time. We have included in our advanced analysis 14 criteria but for the coming iterations we will ignore four of them. Including all 14 criteria makes it easy for the users to experiment with analysis of either different sets of criteria, or with more criteria.

The reasons for temporally ignoring the selected criteria is as follows:

- *Downpayment*: this criterion may be really important in situation if buying a property is either considered as investment of available funds (then the criterion is maximized)

or the buyer has limited cash and prefers to take a mortgage (then the criterion is minimized). Thus this attribute was included to illustrate the situation when the same criterion is either maximized or minimized by different users.

- *Delay*: This criterion is important only in specific situations, and usually has a secondary importance (unless the delay is the property availability is really long).
- *House*: This is example of a binary criterion (a property is either an apartment or a house). Some users may have strong preference for either type of the property (then the criterion is either maximized, or minimized, respectively), other users are indifferent here. Thus this criterion is introduced for users who may want to make a follow-up analysis according to preferences in terms of this criterion.
- *Ground*: This is example of a bi-modal distribution of the criterion value that is highly correlated the *House* criterion; interesting for the users who prefer houses.

6.5.3 Initial analysis of realistic alternatives

Let us start with looking at the compromise solution (for equal importance of all active criteria) for the 10 criteria that remain active available as iteration 1. One may also note that the Pareto alternative *Park* has the possible value of the ignored criterion *Downpayment*, which is analytically correct, and illustrates the fact the ignoring a criterion does not imply that its value in a selected solution is bad. From now on, we suggest to use the chart-option *Display only active criteria* which helps to concentrate on analysis of the active criteria.

As in the basic analysis, we now try to improve the values of criteria that perform poorly and are important for us. We also follow the practice of first doing so by a fine control of relative importance. So let's try to improve the *Area* by increasing its importance. The results are stored in iteration 2.

Comparing the new solution *Cozy-corner* with the previously found *Park* we see that the new choice is better (or much better) on five criteria, the values of three criteria remain the same (and the *Impression* criterion cannot be improved because it is already the best possible). However, the worsening of the two criteria (*Commuting* and *Parking*) may be too high compromise for improving the five criteria.

If commuting quality is important for us we shall consider to slightly relax preferences for price and increase the preferences for commuting while keeping the preferences for the *Area* unchanged. The solution of iteration 3 (compared by marking the *Park* alternative) is likely to convince us to not compromise on the *Price* criterion. Therefore, we change the importance for this criterion back to the average, and keep the other two unchanged.

The *Back str* solution of iteration 4 has all criteria values in the two middle-quantiles, so it might be preferred by those who like compromise solutions and have preferences for those criteria which have values better than 50% of the best value. However, this alternative is really strong on only commuting and OMC criteria, so it will unlikely be accepted by many. A closer look at the *Area* and *Commuting* criteria shows that it is unrealistic to select a solution that have good values for both criteria, so one should not attempt to achieve this. So it is rational to consider now which of these two attributes is more important for us and compromise on the other one. Let us assume that we prefer a nice surrounding over convenient commuting, and therefore relax the preference for the latter one to average. In other words, we aim at compromising on commuting for finding a nicer home having also a more convenient parking. Let us try to explore this by using

the current Pareto solution as a reference point (RFP) for improving the *Parking* criterion while compromising the *Commuting* criterion.

6.5.4 Using a Pareto solution as the RFP

The solution of iteration 5 meets most of the expectations corresponding to the latest modification of preferences. Actually all but the *Commuting* criteria (which is now the only one with the value within the lowest quantile) have been improved, most of them substantially. We may now want to explore, if there is a good alternative having better parking possibilities while not compromising much on the four criteria for which we selected importance over the average. This type of exploration is a typical application of methods which use a Pareto solution as the RFP.

To illustrate such an analysis let us switch to the RFP-Pareto method, for which – in addition to preferences for the relative criteria importance used so far – the user specifies his/her preferences on subsets of criteria that should be either improved, or compromised, or stabilized. Let us keep the relative importance of the criteria unchanged, and set the following improvement preferences:

- improve the *Parking* criterion,
- stabilize *Impression, Design, Area* criteria,
- compromise the *Commuting* criterion,
- set free the other five criteria.

The results are seen by selecting iteration 6 as the current iteration. The Pareto alternative *Gulf sunbed* is of a different character than the previously best *Cozy-corner*. Let us reflect on the criteria whose values differ substantially between these two alternatives. There is a substantial worsening of the *OMC* criterion but it is compensated by the *Price*. We may however want to look for alternatives with better values of *Amenities* and *Milieu*. For attempting this let us let these two criteria to be improved, and change the *Parking* (which achieved its best value) to be stabilized. The results of this attempt are stored in iteration 7, whose solution is again the *Cozy-corner*.

6.5.5 Analysis in the alternative space

Reflecting on the experience gathered so far is likely to trigger us to also explore the alternative space. While looking at the pop-up windows²¹ by clicking from iteration 7 on the *Ranking table* and the *Characteristics* buttons we will notice that four alternatives with the highest rank have also clearly highest values of the scalarizing function and the performance measures. Therefore it is worth to compare them in the pop-up window generated by the *Compare by criteria* button. The three top ranked alternatives are already preselected for the display, therefore we shall drag there only the fourth one, namely *Residence*, and then click the *Redisplay* button.

In the left window of Figure 22 we see the performance of these four alternatives on the criteria sorted by the strength of the reference alternative (the top-listed in the *Selected* window, i.e., the *Cozy-corner*). We notice that also the values of the criteria we set to be disregarded are displayed which gives us a more complete picture than the one using the subset of active criteria. In the right window we see the criteria on which the performance

²¹ Assuming you have set the appropriate options in your browser as advised by the information displayed in the login screen.

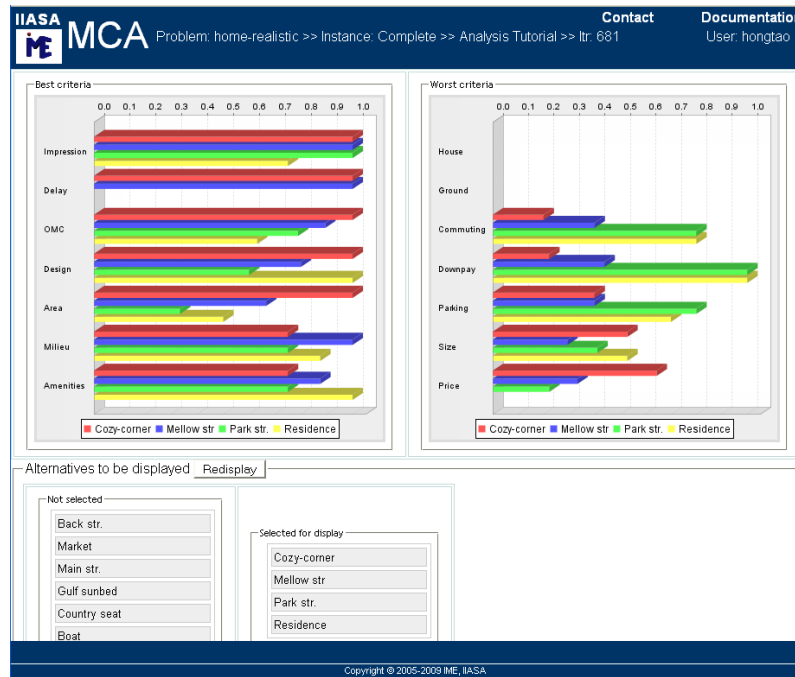


Figure 22: Comparing the four top alternatives from iteration 7.

of reference alternative is week (the worst one at the top). All four selected alternatives have the worst possible performance on the *House and Ground* criteria, which is obvious because all of them are apartments. Considering the other weak (from the perspective of the *Cozy-corner*) criteria we notice that for *Commuting* and *Downpayment* each of the other three performs substantially better.

The latter observation may motivate us to look closer at trade-offs between improving commuting and compromising other criteria. This can be done also by selecting as the reference alternative one of those four already displayed that performs well on commuting. Therefore we consider two for this purpose, namely *Park* and *Residence*. Noting additionally that *Park* is substantially cheaper we drag it to the top position in the *Selected* window.²² The results are shown in Figure 23. We now see the criteria in a different order than in the previous window; this is because the criteria are ordered according to their performance for the reference alternative. By looking at Figures 22 and 23 it is easy to compare differences between the corresponding reference alternatives. For example, one can easily recognize there subsets of criteria for which:

- both alternatives perform rather well,
- both alternatives perform poorly,
- one alternative performs substantially better than the other one.

By considering the second subset we can decide if either of the alternatives is acceptable or not, while through the analysis on the third criteria subset one can evaluate which of these two alternatives is better for us than the other alternative.

Finally, some users may want to analyze the graphs with the following values com-

²²Note, that one can open another pop-window with *Compare by criteria* and see two windows side-by-side, or on a second monitor. This routine substantially helps in the analysis.

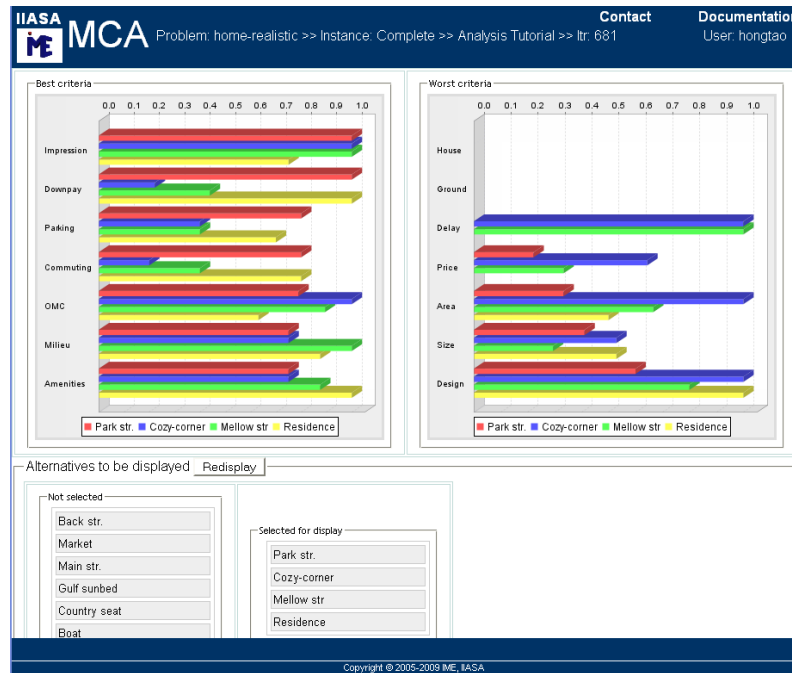


Figure 23: Same comparison as in Figure 22 but with the *Park* as the reference alternative.

puted for each alternative:

- rank measure,
- scalarizing function,
- two performance measures.

Such graphs are available by clicking on the *Characteristics* button at the bottom control panel. The values are normalized to 1, and the alternatives are ordered according to their ranking (with the Pareto alternative at the top).

Here we briefly comment on the graphs corresponding to iteration 7:

- The *Rank measure* values form a non-increasing sequence; note that the worst (for the preferences specified in iteration 7) three alternatives are ranked at the same position.
- The *Scalarizing function* values do not form a non-increasing sequence; by pointing a mouse on the corresponding bar one can see that its values for *Park str* and *Residence* alternatives are 0.945 and 0.948, respectively, although the *Park str* is ranked higher than the *Residence*.
- The two performance measures illustrate different similarity (of the corresponding alternative in respect to the Pareto alternative) measures. Interpretation of similarity measures in the multicriteria analysis context is difficult. We can however observe that the values of the two displayed measures are not only different (for most alternatives very different), and there is rather weak correlation between these values and the ranking position of the corresponding alternative.

6.6 Carrying forward

We stop here the illustrative analysis of the tutorial example, and invite the reader to experiment with own analysis of either this or other instance/problem. Please note that one

can select any iteration as the current one, look at the result, and use a selected analysis as a basis for a next iteration, for which other preferences and/or another method is specified. Therefore you can come back to any element of the tutorial, and use it as a starting point for your own branch of experiments. The only restriction you will face is the inactive *Edit the note* control button for the iterations created by the *mc_tutor* user.

Of course, the reader is also encouraged to use the tutorial problems and/or instances as a starting point for creating own problems and instances. Finally, the authors invite the readers to define own problems, either interactively or from uploaded files.

7 Tips for the MCA

Multicriteria analysis supports the users in finding an efficient alternative which fits best his/her preferences. How efficient is such an analysis process depends on the user's knowledge and experience. Many users have neither time nor incentives for studying the methodological background of multicriteria analysis. Fortunately, understanding of the mathematical background is not necessary for effective use of the MCA methods; however, understanding of the basics of the methodology is essential. One can illustrate this by the level of knowledge required for driving a car. Very few drivers understand the knowledge used for designing and manufacturing the car, and the resulting technical characteristics. However, although good drivers poses usually only a tiny subset of such knowledge, they combine it with experience and intuition, and this synergy enables them to drive a car almost equally well as the engineers who have a much more advance knowledge about cars.

Here we summarize several basic thoughts that aim at helping in an effective analysis. We start with recalling the essence of the multicriteria analysis: assuming we consider Pareto (efficient) alternatives only, one cannot improve performance of any criterion without worsening another one. However, the Pareto alternatives have different compositions of the criteria values, and different users have different preferences. Therefore usually each alternative will be selected by some users, i.e., those who have preferences corresponding to the alternatives criteria value composition.

In an actual analysis we suggest to consider the following guidelines in specification of preferences in the terms of relative importance when attempting to find an alternative with better values of the selected criteria:

- Remember that we specify *relative* importance for all criteria. Therefore increasing importance (especially to one of the two highest levels) of many criteria is usually not a good strategy.
- Increase the importance by one level at a time.
- Increasing importance of selected criteria should be "matched" by decreasing importance for other criteria. A good *rule of the thumb* is to keep the number of criteria with importance above the average to be smaller than the number of criteria having importance below the average.

Please note that one can select any iteration as the current one, look at the result, and use a selected analysis as a basis for a next iteration, for which other preferences and/or another method is specified. Each iteration provides a Pareto alternative, but for many iterations such an alternative will have criteria trade-offs that do not fit your preferences. This is a typical feature of the multicriteria analysis, and one of the reasons we use the

word *analysis* instead of *optimization*. Therefore, the users are encouraged to experiment, which usually results in many iterations. It is also useful to make short notes²³ about "interesting" iteration, e.g., criteria with (non-)satisfactory values. The notes are displayed on the iteration list this helping to identify an iteration that can be used as a parent iteration for further analysis.

Finally, we recommend to follow good practice of rational problem analysis and decision-making. In particular, to make breaks (preferably at least one night-break) in analysis to reflect on the results, and to experiment with different patterns of preferences.

We close the tips with pointing out that the MCA has a dedicated (and easy to use) function for contacting the authors, which can be activated by the *Contact* button available on the top control panel. We encourage the users to contact the MCA developers through this function for sharing suggestions, reporting problems, or asking questions. This is by far the most effective way to get assistance, and to help in improving the application.

References

- [1] GRANAT, J., AND MAKOWSKI, M. Interactive Specification and Analysis of Aspiration-Based Preferences. *European J. Oper. Res.* 122, 2 (2000), 469–485. available also as IIASA's RR-00-09.
- [2] GRANAT, J., AND MAKOWSKI, M. Multicriteria methodology for the NEEDS project. Interim Report IR-09-10, International Institute for Applied Systems Analysis, Laxenburg, Austria, 2009.
- [3] GRANAT, J., MAKOWSKI, M., AND OGRYCZAK, W. Multiple criteria analysis of discrete alternatives with a simple preference specification: Pairwise-outperformance approaches. Interim Report IR-09-23, International Institute for Applied Systems Analysis, Laxenburg, Austria, 2009.
- [4] MAKOWSKI, M. Management of attainable tradeoffs between conflicting goals. *Journal of Computers* 4, 10 (2009), 1033–1042. ISSN 1796-203X.
- [5] MAKOWSKI, M., GRANAT, J., AND OGRYCZAK, W. Overview of methods implemented in MCA: Multiple criteria analysis of discrete alternatives with a simple preference specification. Interim Report IR-09-24, International Institute for Applied Systems Analysis, Laxenburg, Austria, 2009.
- [6] MAKOWSKI, M., GRANAT, J., REN, H., SCHENLER, W., AND HIRSCHBERG, S. Requirement analysis and implementation of the multicriteria analysis in the NEEDS project. Interim Report IR-09-09, International Institute for Applied Systems Analysis, Laxenburg, Austria, 2009.
- [7] MAKOWSKI, M., AND WIERZBICKI, A. Modeling knowledge: Model-based decision support and soft computations. In *Applied Decision Support with Soft Computing*, X. Yu and J. Kacprzyk, Eds., vol. 124 of *Series: Studies in Fuzziness and Soft*

²³For this purpose the *Note* field is available, see Section 6.1.1.

Computing. Springer-Verlag, Berlin, New York, 2003, pp. 3–60. ISBN 3-540-02491-3, draft version available from <http://www.iiasa.ac.at/~marek/pubs/prepub.html>.

- [8] WIERZBICKI, A., GRANAT, J., AND MAKOWSKI, M. Discrete decision problems with large number of criteria. Interim Report IR-07-25, International Institute for Applied Systems Analysis, Laxenburg, Austria, 2007.
- [9] WIERZBICKI, A., MAKOWSKI, M., AND WESSELS, J., Eds. *Model-Based Decision Support Methodology with Environmental Applications*. Series: Mathematical Modeling and Applications. Kluwer Academic Publishers, Dordrecht, 2000. ISBN 0-7923-6327-2.

A Structure of the csv-file with problem and instance definition

It may be convenient to specify large problems and their instances in the form of the csv-format file than to input the data interactively as described in Sections 5.1 and 5.2. Therefore such an option is provided by the MCA. We summarize here the content of the csv-file that corresponds to the tutorial example on creating a csv-format file presented in Section A.1. The corresponding csv-format file (named *car8h.csv*) is available from the MCA Documentation site.

A.1 Car selection problem example

Before explaining the preparation of the csv-format file we summarize the illustrative example that is used for this purpose. Let us assume that you have the set of cars and characteristics of this cars by several attributes given in Table 4.

	Price	Fuel	Diesel	Safety	Reliab.	Guar.	Serv.	Boot	Accel.	Autom.
	kEuro	l/100km	binary	grade	grade	years	kkm	ltr	l/sec.	binary
Toyota	22	8	1	4	5	3	10	350	0.5	0
Honda	19	11	0	3	4	8	18	300	0.72	1
Mazda	17	8	1	2	3	3	15	320	0.42	0
Volvo	24	7	1	6	3	2	16	250	0.56	1
Saab	23	8	1	6	3	2	12	400	0.5	0
VW	25	6	1	5	2	3	20	450	0.63	1
BMW	27	9	1	3	3	2	11	290	0.83	1
Mercedes	26	8	1	4	2	2	12	460	0.45	1

Table 4: Illustrative example of fictitious cars' attributes.

The attribute values have been arbitrarily chosen to illustrate basic features of the multicriteria analysis, and especially to define an easy to follow collection of Pareto-efficient alternatives. They should therefore not be associated with real cars although the names used correspond to commonly know brands. We therefore stress that the names of cars are used here only to ease to use the fictitious example as a tutorial for the multiple criteria analysis.

A.2 Defining a new problem

This preparatory stage is composed of two steps:

- Preparation of text file with problem specifications.
- Loading the file.

The problem definition should be provided in a text file in the so-called csv-format (therefore it is advised to use the *csv* extension in the file name). An example of such a file (together with comments about it structure) is provided below.

The specification of the problem consist of the following sections:

1. Problem name specification.

It is composed of short (up to 16 characters) problem name, long name (up to 128

characters), and short description (up to 2000 characters).

```
<problem>
# Line structure (Fields in square braces are optional, numbers in brackets
# specify maximum number of characters of the corresponding field):
#
#   short name(16);[long name(128)];[description(2000)]
#
Cars_8;; Car (8 alternatives, 10 criteria) example
```

2. The attributes names

The attributes are defined by 4 elements: short name of the attribute (up to 16 characters), long name of the attribute (up to 128 characters), unit (up to 16 characters) that are obligatory, and description of the attribute (up to 2000 characters)

```
<attributes>
#   short name(16);[long name(128)];unit(16);[description(2000)]
#
Price;Price;kEuro;
Fuel;Fuel;l/100km;
Diesel;Diesel;(binary);
Safety;Safety;grade-scale;
Reliability;Reliability;grade-scale;
Guarantee;Guarantee;years;
Service;Service;km;
Boot;Boot;ltr;
Acceler.;Acceler.;sec.;
Automatic;Automatic;(binary);
```

3. The names of the alternatives.

The alternatives are defined by 4 elements: short name of the attribute (up to 16 characters), long name of the attribute (up to 128 characters), unit (up to 16 characters) that are obligatory, and description of the attribute (up to 2000 characters)

```
<alternatives>
#   short name(16);[long name(128)];[description(2000)]
#
Toyota;;
Honda;;
Mazda;;
Volvo;;
Saab;;
VW;;
BMW;;
Mercedes;;
```

4. Define the values of the attributes for each of defined alternative

```
<values>
# A line defines values of attributes (indicators) for an alternative.
# The number and order of the fields in each line must correspond to the
# number and order of attributes defined in the <attributes> section.
# The number and order of lines must correspond to the alternatives
# defined
# in the <alternatives> section.
# The (optional) decimal point character is the dot "."
#
22;8;1;4;5;3;10;350;0.5;0
19;11;0;3;4;8;18;300;0.72;1
17;8;1;2;3;3;15;320;0.42;0
24;7;1;6;3;2;16;250;0.56;1
23;8;1;6;3;2;12;400;0.5;0
25;6;1;5;2;3;20;450;0.63;1
27;9;1;3;3;2;11;290;0.83;1
26;8;1;4;2;2;12;460;0.45;1
```

A.3 Problem instance specification

For each problem one can define several problem instances. An instance specification consist of:

1. Definition of the name of the instance

Short name of the problem (up to 16 characters) and long name (up to 128 characters).

```
<instance>
# short name(16);[long name(128)]
#
# All criteria:
```

2. The definition of the criteria.

In order to start multicriteria analysis we have to define the criteria. We have to define the short name of the criterion (up to 16 characters), long name of the criterion (up to 128 characters), attribute that define the criterion, and the type of the criterion.

```
<criteria>
# short name(16);[long name(128)];[attribute(16)];[parent(16)];type(6);[target value(16)]
#
# Comments:
# - short name is used as identifier of the corresponding criterion, and
# also for forms/graphs, optional long criterion name may be used when
# more space will be available (e.g., in reports)
# - attribute (that defines a criterion on the lowest level) must correspond
# to the short name of a defined attribute (indicator)
# - empty attribute field implies a criterion above the lowest criteria
# level; such a criterion must be a parent of at least one lower-level
# criterion
# - the type has to be one of the keywords: {min max target}
# - the target value field is processed only for the target
# criterion-type.
#
Economy;;;max;
Price;;Price;Economy;min;
Fuel;;Fuel;Economy;min;
Guarantee;;Guarantee;Economy;max;
Service;;Service;Economy;max;
Diesel;;Diesel;Economy;max;
Performance;;;max;
Safety;;Safety;Performance;max;
Reliability;;Reliability;Performance;max;
Boot;;Boot;Performance;max;
Acceler.;;Acceler.;Performance;max;
Automatic;;Automatic;Performance;max;
```

In this example the criteria names are the same as the corresponding attribute names. This is usually convenient, but in some situations the user may prefer different names for these two entities. The criterion type is either max or min, denoting maximized and minimized criteria, respectively.

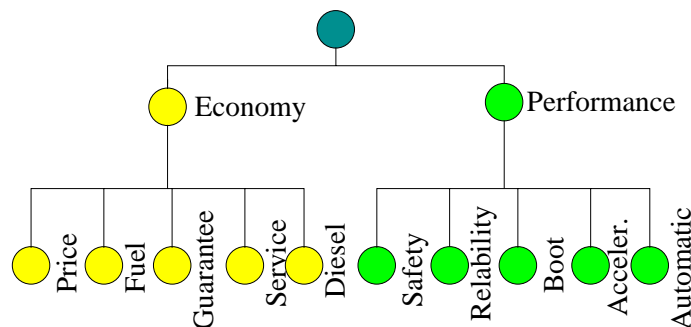


Figure 24: A simple example of criteria hierarchy,

3. Optional specification of criteria hierarchy.

For problems with a large number of criteria some users may prefer to organize

them into a hierarchical structure (a simple example is shown in Fig. 24). To illustrate how the corresponding specification is done in the csv-file, we have defined the hierarchy also for the small tutorial example discussed above.

For problem instances without criteria hierarchy one should have left the *parent* field empty, and refrain from defining upper-level criteria. The csv-format file for the car example with criteria defined in the traditional (i.e., without hierarchy) way is available from the MCA Documentation Web-site is called *car8.csv*.