# diviz: a tool for modeling, processing and sharing algorithmic workflows in MCDA

Patrick Meyer, Sébastien Bigaret
Institut Télécom, Télécom Bretagne
UMR CNRS 3192 Lab-STICC
Technopôle Brest-Iroise CS 83818, F-29238 Brest Cedex 3
Université européenne de Bretagne
{patrick.meyer,sebastien.bigaret}@telecom-bretagne.eu

**Abstract**

The diviz software is a tool for modeling, processing and sharing Multiple Criteria Decision Aid (MCDA) techniques. The target user community of diviz reaches from researchers to teachers and students via practitioners. One of the principal features of diviz is to ease the comparison of results produced from different methods and algorithms on the same decision problem instance.

**Keywords**: MCDA, algorithmic workflows, methods comparison, XMCDA

## 1 Introduction

Research activities in and around the field of Multiple Criteria Decision Aid (MCDA) have developed quite rapidly over the past years, and they have resulted in various streams of thought and methodological formulations for the resolution of complex decision problems. In particular, many so-called MCDA *methods* have been proposed in the literature and are very often available as software programs.

Unfortunately, at least four major difficulties arise when it comes to using these programs in practice:

1. different techniques are generally implemented in separate software products, with *heterogeneous* user interfaces;
2. testing multiple MCDA algorithms on one problem instance is not easy, because of the various input *data formats* required by the software applications;
3. a lot of MCDA algorithms which are presented and published in scientific articles are not easily *available* and consequently often only used by their authors;
4. several MCDA software products are not *free* (neither from a financial, nor from an open-source point of view), which can be considered as a weakness for their large dissemination.

In other scientific research fields, as, e.g., statistics or data mining, there exist software platforms which allow to easily compare different analysis methods and to test them on a given dataset inside a common framework. Among the most famous ones, one can cite platforms such as the GNU R statistical system by the R Development Core Team (2005) or the Weka suite of machine learning software by Hall et al. (2009). Both of these suites are open-source and OS independent, which has certainly contributed to their large dissemination and acceptance among many researchers and users.

In order to overcome the earlier mentioned difficulties linked to the software situation in the field of MCDA, a group of researchers has got together to create the Decision Deck project (Decision Deck Consortium, 2009). Its objective is to collaboratively develop open-source software tools implementing MCDA techniques. As such, its purpose is to provide effective tools for at least three types of users:

- practitioners who use MCDA tools to support actual decision makers involved in real world decision problems;
- teachers who present MCDA algorithms in courses;
- researchers who want to test, share and compare algorithms or to develop new ones.

In this article we focus on diviz, one of the software initiatives of the Decision Deck project, which eases the use of algorithmic resources from the field of MCDA. The diviz tool is an open-source software which allows to design complex *workflows* of MCDA algorithms in a very intuitive way, and to execute and share them.

The diviz tool uses extensively two other outcomes of the Decision Deck project, which we also present shortly in the sequel:

- `XMCDA`: a standardized `XML` recommendation to represent objects and data structures coming from the field of MCDA. Its main objective is to allow different MCDA algorithms to *interact* and to analyze a problem instance stored in `XMCDA` by various MCDA algorithms;
- `XMCDA` web-services: distributed open-source computational MCDA resources.

The article is structured as follows. In Section 2 we present diviz, its properties and its benefits in various practical contexts. Then, in Section 3 we detail its use on a classical MCDA problem from the literature, before concluding in Section 4.

## 2 Description and features of diviz

In this section we focus on the diviz software and the resources on which it relies. We first start by describing the usage and the main features of the tool. We also outline the original work methodologies which arise from the use of this tool, before mentioning the external resources on which diviz relies.

## 2.1 General use of diviz

The diviz tool is an easy to use software to build, execute and share complex workflows of MCDA algorithms. In the literature, such workflows are often called *methods* (consider, e.g., the ELECTRE method by Roy (1968), the UTA method by Jacquet-Lagrèze and Siskos (1982), etc). One of the main features of diviz is that it facilitates the construction of these classical MCDA *methods*, as well as derivatives and original ones, by combining various elementary calculation components[1].

This vision leads to a new work methodologies with MCDA softwares :

- Remove the *black box* effect of certain softwares;
- Better understand the heart of the methods and their similarities / dissimilarities;
- Easily test variations of methods by replacing one of its algorithmic components by another one.

For short, diviz enables to conveniently combine programs implementing MCDA algorithms, which can originate from various methodological schools, researchers and programmers. Furthermore, it can be used as a research and dissemination tool for methods, experiments and real-world analysis cases. We detail all these features hereafter.

### Workflow design

The design of the MCDA workflows is performed via an intuitive graphical user interface, where each algorithm is represented by a box which can be linked to data files or supplementary calculation elements by using connectors (see Figure 1 of Section 3 for an example). Thus, the design of complex algorithmic workflows does not require any programming skills, but only necessitates to understand the functioning of each calculation module (each of which is documented in details on the website of diviz[2]).

The inputs and outputs of these elementary components can be manifold and can correspond to various MCDA concepts or data elements. To illustrate this, consider the following example.

**Example** *diviz allows to use a component called* `weightedSum`*. This element calculates the weighted sum of alternatives' performances with respect to a set of weights associated with a list of criteria. Consequently,* `weightedSum` *requires four inputs: the description of the criteria, the description of the alternatives, the performance table containing the numerical evaluation of each alternative on each of the criteria, and the numerical weights associated with the criteria. The main output of this component are the overall values of the input alternatives via the weighted sum aggregation operator.*

To construct a new MCDA workflow, the user chooses one or more modules from a list of available calculation elements which he can drag and drop in a dedicated workspace. Then he adds data files to the workspace and connects

[1] Note that in the sequel we will unthinkingly use the words *component*, *program* or *module* to describe the algorithms which can be combined in diviz.

[2] http://www.decision-deck.org/diviz

them appropriately to the inputs of the elements. Finally he connects the inputs and outputs of the components by connectors to define the structure of the workflow.

### Execution and results

Once the design of the MCDA workflow is finished, the user can execute it in order to obtain the possibly multiple outputs of the algorithms. These calculations are performed on high performance computing servers through the use of the `XMCDA` web-services (see Section 2.2 describing the external resources used by diviz). As a consequence, diviz does not *physically* contain any calculation modules, but requires a connection to the Internet to access these resources.

After the execution of the workflow, the outputs of each of the components can be viewed and analyzed by the user. Some of these outputs might represent results of intermediate calculation steps of the workflow. Consequently, the user is given a detailed vision of the algorithm and he can quite easily tune the parameters of the algorithms. We show these features in the following example.

**Example** *Consider the following workflow (typically a UTA-like disaggregation method): a first module determines piecewise linear value functions on basis of a ranking of alternatives provided by the user; a second module transforms a performance table by applying these value functions on the performances of the alternatives; a third module calculates the sum of these performances for each of the alternatives; a fourth module draws a ranking of the alternatives on basis of the overall values previously computed. The intermediary results are the value functions, the transformed performance table and the overall values of the alternatives. As each of these elements is explicitly available for the user, first he can gain a deeper understanding of the decision aid method which he has constructed, and second the fine-tuning of the input parameters (here, the number of segments of the value functions to be constructed, the ranking provided by the user, etc.) is facilitated.*

In diviz the history of the past executions is kept in the software and can at any moment be viewed by the user. More precisely, if a workflow is modified, the former executions' results and their associated workflows are still available. This also contributes to the good understanding of the constructed chain of algorithms and helps calibrating the parameters of the workflow's elementary components.

### Available algorithmic components

As already mentioned earlier, the algorithmic elements available in diviz are web-services proposed by the Decision Deck project. At the time of writing, about sixty such components can be used, which can be divided into four main categories:

1. *calculation components* containing aggregation operators, disaggregation techniques, post-analysis elements, etc.;
2. *methods* containing full MCDA methods;

3. *visualization components* containing modules allowing to represent graphically certain input and output data elements;

4. *reporting components* containing techniques to create aggregated reports of multiple output data pieces.

Each of the available calculation components is documented on diviz's website and details are given on the requirements for the inputs.

These programs allow to reconstruct classical MCDA methods like the ELECTRE series by Roy (1968), the PROMETHEE series by Brans and Vincke (1985) and UTA-like techniques by Jacquet-Lagrèze and Siskos (1982). Next to that, more recent techniques linked to the elicitation of capacities in Choquet integral-based MCDA can also be used (see Grabisch et al., 2008), as well as some inverse analysis techniques by Bisdorff et al. (2009).

#### Comparison of "methods"

Next to designing and executing MCDA workflows, diviz can also be a convenient tool to compare the outputs of various methods and algorithms on the same input data.

Up to recently, such a task has been far from easy, as no unified software platform for MCDA techniques existed. However, with diviz and its possibility to construct complex workflows, it is easy to connect a dataset linked to a specific decision problem to various workflows in a single workspace, each of them representing a different MCDA method, and to compare their outputs. This is clearly a very simple way to check the robustness of the output recommendation of an analysis with respect to the choice of the decision aid technique. We present this innovative feature in further details in Section 3 via an application.

#### Workflow sharing and dissemination

The diviz software enables to export any workflow, with or without the data, as an archive. The latter can then be shared with any other diviz user, who can then import it (by loading the archive) into his software and continue the development of the workflow or execute it on the original data.

Consequently, diviz can be used as a convenient dissemination tool: first, in combination with a research article, the authors of a new MCDA technique or an experiment could propose the corresponding diviz workflow together with an appropriate data set as supplementary electronic material with their article. Second, in a practical context, MCDA analysts could also be willing to share the algorithmic treatment they have performed with the various stakeholders of the process. This dissemination feature will certainly contribute to a larger dissemination of new algorithms and facilitate their acceptance among many researchers and users.

In this context, the example which is presented in Section 3 is available as a downloadable archive from the diviz website, and can be tested by any interested reader.

### 2.2 Resources used by diviz

In the Introduction we made mention of the fact that diviz relies on two further outcomes of the Decision Deck project, namely the `XMCDA` data standard and

the XMCDA web-services. In this section we briefly detail these two initiatives and present how they contribute to the diviz tool.

### The XMCDA data standard

One of the major difficulties linked to the use of the tools from the MCDA research domain is the heterogeneity of the available software programs and their input and output data formats. Unfortunately, until recently, this lack of standardization of the data did not allow to combine the existing tools and to create treatment chains which would involve multiple software pieces.

Consequently, in such a situation, the resolution of a complex decision problem comes generally down to testing only one algorithm and using various post-processing tools to analyze the results of the resolution. This can be frustrating for a lot of MCDA analysts and practitioners who might like to test various methods on a given problem, without having to recode the instance in various data formats.

In order to overcome these difficulties, and in particular to allow running a problem instance through multiple techniques or methods and to allow the chaining of various MCDA algorithms, researchers of the Decision Deck project have defined a data standard, called XMCDA (see Decision Deck Consortium, 2010), which can be adopted by various programs to make them interoperable.

The XMCDA markup language is written in XML[3], a general-purpose syntax for defining markup languages. XML's purpose is to aid information systems in sharing structured data, especially via the Internet and to encode documents. XMCDA is defined via an XML Schema[4], a set of syntax rules (together with a set of constraints) which define its structure. The XML Schema of the latest version of XMCDA is available via the XMCDA website. At the time of writing, the official version approved by the Decision Deck Consortium is 2.1.0.

The root tag of XMCDA is named XMCDA and contains several sub-tags, each of them describing data related to a decision aid problem. To summarize, these tags can be put in five general categories:

- description of the current decision aid project or description of the XMCDA file;
- description of the MCDA concepts attributes, criteria, alternatives and categories;
- the performance table;
- preferences related to criteria, alternatives, attributes or categories (either provided as input by a decision maker or produced as the output of an algorithm);
- output messages from methods or algorithms (log or error messages) and input information for methods or algorithms (parameters).

Note that an XMCDA file does not require that all of these categories are present to be considered as valid with respect to the schema. A valid XMCDA file may contain only one tag under the root element.

[3] http://www.w3.org/XML/
[4] http://www.w3.org/XML/Schema

In Section 3, which focuses on the use of diviz for an application, we present an instance of an XMCDA file.

### XMCDA web-services

The XMCDA web-services are the initiative of the Decision Deck project which enables an easy access to MCDA algorithms. From a general point of view, a web-service is an application which can be accessed via the Internet and is executed on a remote system. One of the great advantages of such online programs is their availability to anyone at any time and any place and on any computer which is connected to the Internet.

XMCDA web-services have the following properties:

- they "speak" XMCDA: their inputs and outputs are formatted using this standard. This guarantees that all web-services are able to inter-operate;
- they are asynchronous: each of them exposes a method for submitting a problem and an other one for retrieving the results. Consequently, one can submit long-running tasks (hours, or even days) and retrieve them afterwards, without having to stay connected in-between;
- they can be made of any programming language (with the current limitation that it should be runnable on a Linux machine): everyone can participate to the web-services effort using their favorite language;

The XMCDA web-services are moreover released under an open-source license.

So-called MCDA methods are very often sequences of various more elementary algorithms. In most of the implemented MCDA tools, this is generally poorly visible. In order to clear things up, the XMCDA web-services propose such elementary modules which are available as separate software pieces, which, if properly chained, can rebuild the original methods.

Moreover, the general approach behind XMCDA web-services is also aimed at avoiding repeated and unnecessary reimplementations of the same algorithms: by gathering individual algoritmic components, it fulfills the need of capitalization and software re-use in the MCDA domain.

At the time of writing, about 60 XMCDA web-services are available. Some of them are built on functions from the kappalab library by Grabisch et al. (2008) which can be used for Choquet integral-based MCDA. Others contain disaggregation techniques based on outranking relations (PyXMCDA library, by Veneziano (2010)) and additive value functions (using the UTAR library by Leistedt (2010)). Others propose outranking relations-based techniques (based on the J-MCDA library by Cailloux (2010) and on the digraph module by Bisdorff (2007)). Various data manipulation and visualization elements are available via the ws-RXMCDA library by Bigaret and Meyer (2010).

For an exhaustive list of existing XMCDA web-services, please refer to http://www.decision-deck.org/ws. On this website, each of the available calculation components is documented and details are given on the requirements for the inputs.

### Consequences on diviz

The use of these resources has three direct consequences on diviz :

- first, as the XMCDA web-services can interoperate via the XMCDA data standard, all components available in diviz also can. Typically, the output of one algorithm can be injected into other elementary modules without requiring data transformations;
- second, the inputs and the outputs of the elementary components in diviz are typed with respect to the different data types defined in XMCDA. This facilitates the creation of complex combinations of components;
- third, diviz takes advantage of the powerful feature of XSLT transformations to convert XMCDA documents into HTML pages for the visualization of their contents in a web browser integrated in diviz.

### 2.3 A quick look at diviz's architecture

Technically speaking, diviz is a classical 3-tier application made of: the client which has been described in this section, a component accessing the XMCDA web-services, and a server. The diviz server's main task consists in planning and controlling the execution of the submitted workflows, making it possible for different components in the workflow to be executed in parallel, when appropriate.

The users of diviz download the client only; the latter connects to the server which takes care of distributing the computations to the dedicated web-services; it gathers all intermediary and final results which are ultimately sent back to the user.

## 3 Use of diviz in practice

In this section we present the use of diviz on a classical MCDA problem which has been widely discussed in the literature, namely the choice of a sports car (see Bouyssou et al. (2000), chapter 6). We first present the context of the example and outline its XMCDA encoding. Then we detail a fictitious decision aid procedure which would use two separate MCDA techniques for the resolution of the problem.

### 3.1 The data and its XMCDA coding

Let us briefly recall the main characteristics of this example and the underlying data.

In 1993, Thierry, a student aged 21, is passionate about sports cars and wishes to buy a middle range 4 years old car with a powerful engine. He selects five viewpoints related to cost (criterion *g1*), performance of the engine (criteria *g2* and *g3*) and safety (criteria *g4* and *g5*). The list of alternatives and their evaluations on these five criteria is presented in Table 1. The "cost" criterion (€) and the performance criteria "acceleration" (seconds) and "pick up" (seconds) have to be minimized, whereas the safety criteria "brakes" and "road-hold" have to be maximized. Note that the values of the latter two criteria are average evaluations obtained from multiple qualitative evaluations which have been recoded as integers between 0 and 4. Further details on these data can be found in Bouyssou et al. (2000).

| car ID | car name | cost ($g1$, €) | accel. ($g2$, s) | pick up ($g3$, s) | brakes ($g4$) | road-hold ($g5$) |
|---|---|---|---|---|---|---|
| a01 | Tipo | 18342 | 30.7 | 37.2 | 2.33 | 3 |
| a02 | Alfa | 15335 | 30.2 | 41.6 | 2 | 2.5 |
| a03 | Sunny | 16973 | 29 | 34.9 | 2.66 | 2.5 |
| a04 | Mazda | 15460 | 30.4 | 35.8 | 1.66 | 1.5 |
| a05 | Colt | 15131 | 29.7 | 35.6 | 1.66 | 1.75 |
| a06 | Corolla | 13841 | 30.8 | 36.5 | 1.33 | 2 |
| a07 | Civic | 18971 | 28 | 35.6 | 2.33 | 2 |
| a08 | Astra | 18319 | 28.9 | 35.3 | 1.66 | 2 |
| a09 | Escort | 19800 | 29.4 | 34.7 | 2 | 1.75 |
| a10 | R19 | 16966 | 30 | 37.7 | 2.33 | 3.25 |
| a11 | P309-16 | 17537 | 28.3 | 34.8 | 2.33 | 2.75 |
| a12 | P309 | 15980 | 29.6 | 35.3 | 2.33 | 2.75 |
| a13 | Galant | 17219 | 30.2 | 36.9 | 1.66 | 1.25 |
| a14 | R21t | 21334 | 28.9 | 36.7 | 2 | 2.25 |

Table 1: Data for Thierry's car selection problem

Let us now show some excerpts from the `XMCDA` coding of this problem. First of all, the alternatives are defined as follows:

```
<alternatives>
	<alternative id="a12" name="P309">
		<description>
			<comment>Peugeot 309</comment>
		</description>
	</alternative>
	[...]
	<alternative id="a14" name="R21t">
		<description>
			<comment>Renault 21</comment>
		</description>
	</alternative>
</alternatives>
```

Then, the criteria are defined by the following piece of code:

```
<criteria>
	<criterion name="Cost" id="g1"/>
	[...]
	<criterion name="Road-hold" id="g5"/>
</criteria>
```

The evaluations of the cars on the criteria are stored in the following performance table:

```
<performanceTable>
	<alternativePerformances>
		<alternativeID>a11</alternativeID>
		<performance>
			<criterionID>g1</criterionID>
			<value><real>17537</real></value>
		</performance>
		[...]
		<performance>
			<criterionID>g5</criterionID>
```

```
                <value><real>2.75</real></value>
            </performance>
        </alternativePerformances>
        [...]
        <alternativePerformances>
            <alternativeID>a14</alternativeID>
            <performance>
                <criterionID>g1</criterionID>
                <value><real>21334</real></value>
            </performance>
            [...]
            <performance>
                <criterionID>g5</criterionID>
                <value><real>2.25</real></value>
            </performance>
        </alternativePerformances>
</performanceTable>
```

In the following section we present various analyses of this example via the diviz software and XMCDA web-services, both relying on XMCDA files.

## 3.2 Use of diviz for Thierry's choice problem

Thierry's goal is to buy one of the cars from Table 1. We therefore will determine a ranking on the available cars, to help him to select the one which seems the most appropriate in view of his preferences.

However, in order to show the potential of diviz, we will extend the classical example as follows:

- In a first step, Thierry considers that he knows well some of the cars, and expresses some preferences their ranking as follows (as done in Bouyssou et al., 2006):

$$\text{P309-16} \succ \text{Sunny} \succ \text{Galant} \succ \text{Escort} \succ \text{R21t};$$

- Later, in a second step, after a discussion with an MCDA analyst, he changes his mind, and is no longer confident in the ranking he provided. He rather prefers to indicate preferences on the importance of the criteria and on discrimination thresholds (indifference and preference). This information is summarized in Table 2. Note that Thierry is aware that these preferences might not be compatible with the ranking that he provided earlier, but he wishes to compare both recommendations.

| | cost ($g1$, €) | accel. ($g2$, s) | pick up ($g3$, s) | brakes ($g4$) | road-hold ($g5$) |
|---|---|---|---|---|---|
| weight (%) | 40 | 20 | 20 | 10 | 10 |
| indifference | 500 | 1 | 0.5 | 1 | 1 |
| preference | 2000 | 1.5 | 1 | 2 | 2 |

Table 2: Thierry's intra- and intercriteria preferences

In order to determine the two recommendations and compare them, the following workflow is constructed in diviz:

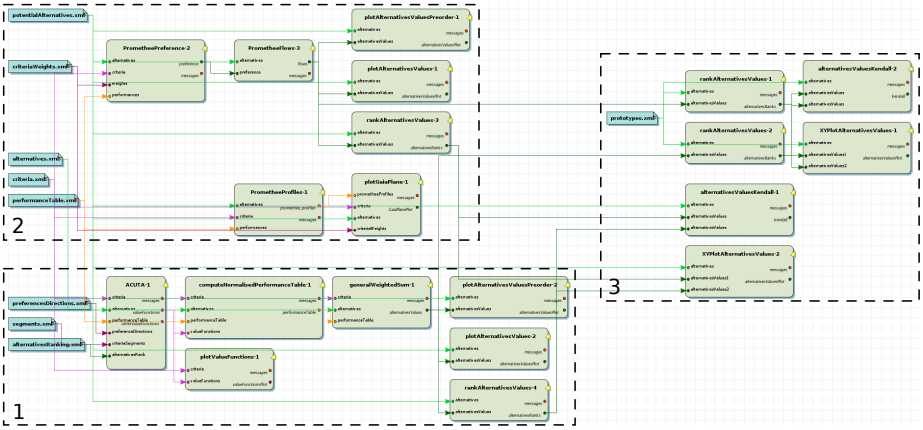


Figure 1: The workflow for the car selection problem

- A first part of the workflow represents a variant of the UTA method (Jacquet-Lagrèze and Siskos, 1982) named ACUTA (Bous et al., 2010) in order to determine a ranking of the alternatives on basis of Thierry's initial ranking (dashed box 1 on Figure 1);
- A second part of the workflow represents the PROMETHEE method (Brans and Vincke (1985)) in order to determine a ranking of the alternatives on basis of the intra- and intercriteria preferences (dashed box 2 on Figure 1);
- Finally, a third part of the workflow is used to compare the outputs of the two methods (dashed box 3 on Figure 1).

Let us now present each of these sub-workflows in further details.

### Part 1: ACUTA

A zoom on the ACUTA part is shown on Figure 2. This workflow contains 7 elementary components, including the one implementing the ACUTA method. For short, ACUTA determines piecewise linear partial value functions on basis of an input ranking of a subset of alternatives, compatible with an additive value model with piecewise linear value functions. Consequently, the `ACUTA`

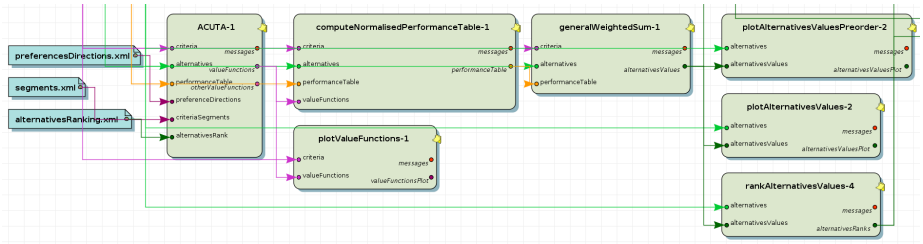


Figure 2: The ACUTA part of the workflow

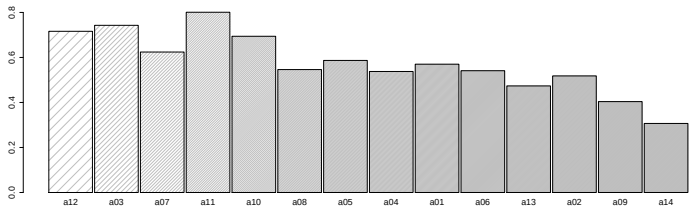


Figure 3: Bar chart representing the overall values of the alternatives

module requires as inputs the description of the alternatives and the criteria, a performance table evaluating each alternative on the criteria, a ranking of a subset of alternatives, the preference directions on the values of the criteria and the number of segments of each piecewise linear value function. One of the outputs of this component is a description of a set of value functions compatible with the input ranking.

These value functions are then used by `computeNormalisedPerformanceTable` to transform the original performance table into a version, where all the evaluations of the alternatives are in the real unit interval. The value functions are also plotted via the `plotValueFunctions` component. From these first computations we can observe that the cost criterion (*g1*) has the biggest influence in Thierry's choice and that he highly values cars which cost less than 18000 EUR.

The `generalWeightedSum` module is used to calculate the aggregated value of each of the alternatives of the transformed performance table by a simple sum. This output is then plotted in a bar chart via the `plotAlternativesValues` component (see Figure 3) and as a ranking via the `plotAlternativesValuesPreorder` module. Finally, the `rankAlternativesValues` calculation element is used to obtain the ranks of the alternatives according to their overall values.

We can easily observe that alternative a11 (P309-16) obtains the highest overall value and can as such be considered as the best car for Thierry via this model.

Let us now take advantage of the flexibility of diviz in order to compare these results with the output of the second MCDA technique, namely the PROMETHEE method.

**Part 2:** PROMETHEE

A zoom on the PROMETHEE workflow is presented on Figure 4. It is made of 7 components, among which 3 are very generic and are not related to this particular method. The `PrometheePreference` module calculates, on basis of a performance table, criteria, alternatives, discrimination thresholds and weights of the criteria a preference index between all pairs of alternatives (see Table 2 for details on the input parameters). This index is then used to calculate the net flow via the `PrometheeFlows` component, which is used to rank the alternatives. Similarly as for the ACUTA part, the preorder of the alternatives is computed via the `plotAlternativesPreorder` module, a bar chart of the net flows is generated via the `plotAlternativesValues` element (see Figure 5), and the ranks of the alternatives are obtained through the `rankAlternativesValues`

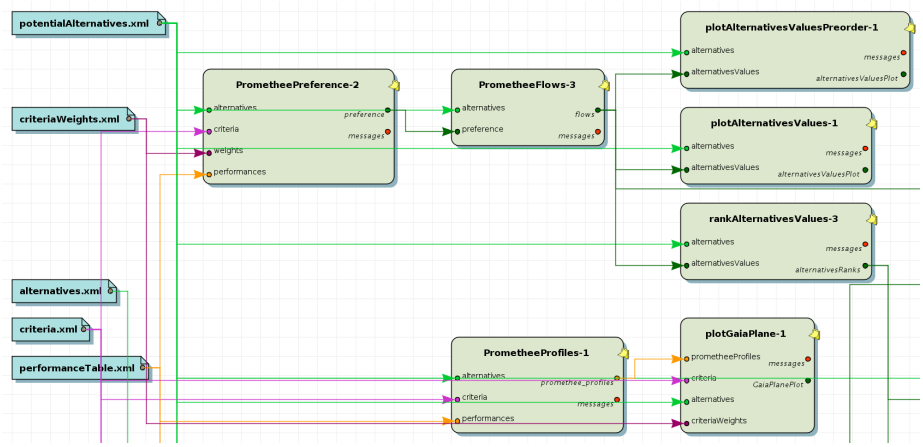


Figure 4: The PROMETHEE part of the workflow

component.

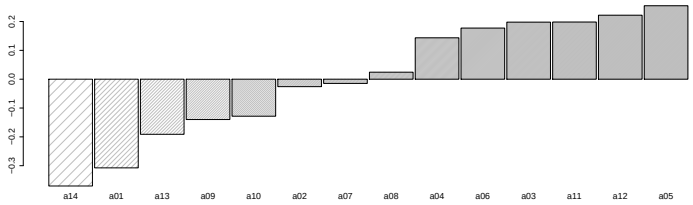


Figure 5: Bar plot of the net flows

According to the net flows obtained via the PROMETHEE method, car *a5* (Colt) is to be considered as the best one for Thierry via this model of his preferences.

The GAIA plane (see Brans and Mareschal, 1994) allows to observe the alternatives and the criteria in a common plane, and to observe the influence of the criteria weights on the final ranking via the net flows. To draw this plane, the module `PrometheeProfiles` is used to compute a transformation of the performance table into flows decomposed for each criterion. This output is then connected to the input of the `plotGaiaPlane` module to obtain Figure 6.

The GAIA plane explains partly the ranking obtained by the PROMETHEE method, and shows that the alternatives *a05* and *a12* will probably remain in the first positions, unless the relative weights of *g1* and *g3* are dramatically lowered. We can also note that the information contained in criteria *g4* and *g5* is to some extent redundant for the PROMETHEE method.

Let us now proceed to the comparison of the outputs of both methods.

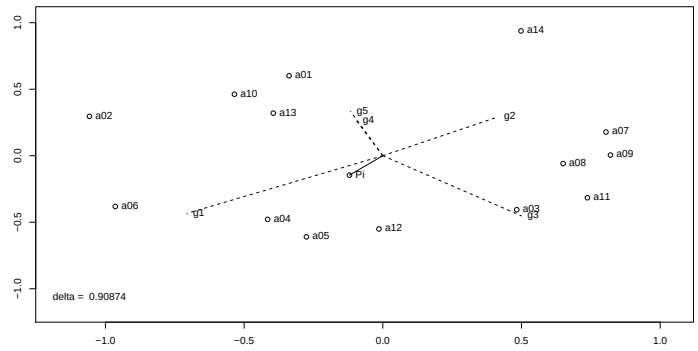


Figure 6: The GAIA plane

### Part 3: Comparison

A zoom on this third part is done on Figure 7. First of all Thierry is interested

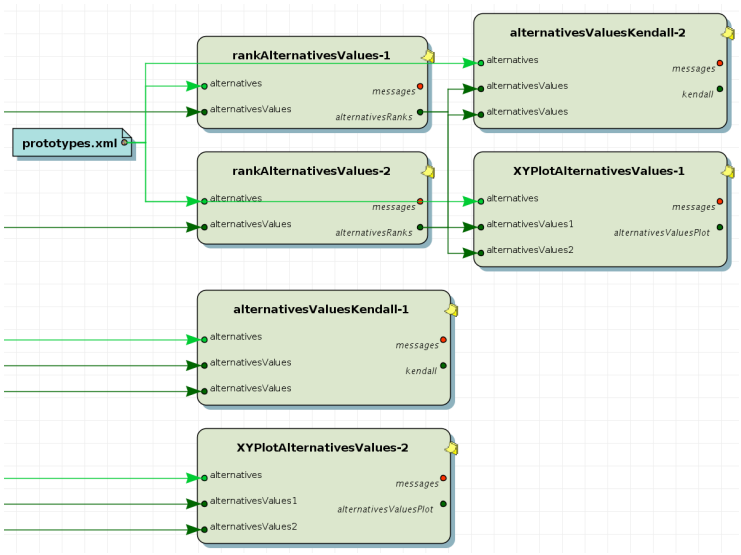


Figure 7: Comparison of the two methods

by the output of the PROMETHEE method on the 5 cars of his initial ranking that was used by the ACUTA method. Therefore we first start by restricting the rankings of the previous two sub-workflows to these 5 cars and compare these two new rankings.

This is done in the upper part of Figure 7. Two `rankAlternativesValues` components take a restricted list of those 5 alternatives as input (`prototypes.xml` file), as well as the two main outputs of the previous sub-workflows, to produce

two new rankings. These are then compared via the `alternativesValuesKendall` component through Kendall's tau (rank correlation coefficient) and a graphical comparison of both rankings via `XYPlotAlternativesValues`. Kendall's tau equals 0.8, which means that for those 5 alternatives, there exists one inversion in the ranking (in this case, *a09* is put before *a13* in the output of PROMETHEE, instead of the inverse requirement in Thierry's initial ranking).

In the lower part of Figure 7, the two output rankings of the PROMETHEE and ACUTA sub-workflows are compared via their Kendall's tau and again the `XYPlotAlternativesValues` module. The output of the latter one is given on Figure 8. Such a shape of the XY plot means that there are quite a lot of inversions between both rankings (equal rankings generate a monotonically increasing graph). Consequently, Kendall's Tau equals 0.47, which confirms that the rankings of the complete set of cars are quite different for the two chosen methods.

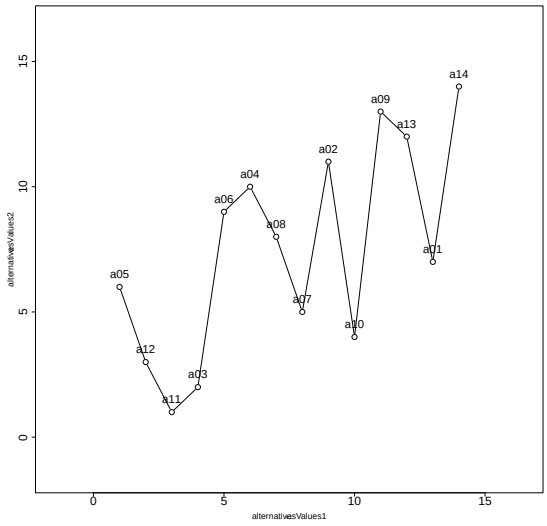


Figure 8: XY plot representing both rankings (abscissa: ranks by PROMETHEE, ordinate: ranks by ACUTA)

The reader may wonder what solution was finally chosen by Thierry. To finish this story with a happy ending, we will suppose here that Thierry likes the fact that PROMETHEE (nearly) confirms his initial ranking. As furthermore he is quite confident in the inter- and intracriteria preferences that he provided, he chooses to follow the recommendation given by the PROMETHEE method, and buys car a05 (Colt).

Note that the workflow described in this section can be downloaded from the website of the diviz initiative (http://www.decision-deck.org/diviz/workflow.URPDM2010SpecialIssue.html) and can easily be imported into any diviz client for testing.

# 4 Conclusion

In this article we have presented diviz which can be used to design and execute algorithmic MCDA workflows and to disseminate research results. The diviz software is being constantly improved, and the number of available components is quickly growing.

The example detailed in Section 3 underlines the big potential of the software for the targeted user community. Researchers can easily share and test MCDA workflows or experiments, explore robustness issues linked to the choice of the algorithm or the values of the input parameters and spread their own algorithms very conveniently. diviz is also an easy to use pedagogical tool for teachers who need to present and compare classical MCDA methods. Last but not least, diviz may also be used by practitioners who wish to solve real world decision problems with a given method.

All in all, diviz gives rise to an innovative work methodology in MCDA, which no longer considers the methods as static and immutable black boxes, but rather as dynamic workflows which can be changed and adapted for the current purpose.

# References


Bigaret, S. and Meyer, P. (2009-2010). ws-RXMCDA. http://github.com/paterijk/ws-RXMCDA. 7

Bisdorff, R. (2007). The Python digraphs module for Rubis. http://ernst-schroeder.uni.lu/Digraph/doc/. 7

Bisdorff, R., Meyer, P., and Veneziano, T. (2009). Inverse analysis from a condorcet robustness denotation of valued outranking relations. In Rossi, F. and Tsoukiás, A., editors, *Algorithmic Decision Theory*, LNAI, pages 180–191. Springer-Verlag Berlin Heidelberg. 5

Bous, G., Fortemps, P., Glineur, F., and Pirlot, M. (2010). Acuta: A novel method for eliciting additive value functions on the basis of holistic preference statements. *European Journal of Operational Research*, 206(2):435–444. 11

Bouyssou, D., Marchant, T., Pirlot, M., Perny, P., Tsoukias, A., and Vincke, P. (2000). *Evaluation and decision models, A critical Perspective*. Kluwer's International Series. Kluwer, Massachusetts. 8

Bouyssou, D., Marchant, T., Pirlot, M., Tsoukias, A., and Vincke, P. (2006). *Evaluation and decision models with multiple criteria, Stepping stones for the analyst*. Springer's International Series. Springer, New York. 10

Brans, J. and Mareschal, B. (1994). The PROMETHEE-GAIA decision support system for multicriteria investigations. *Investigation Operativa*, 4(2):102–117. 13

Brans, J.-P. and Vincke, P. (1985). A preference ranking organization method. *Management Science*, 31(6):647–656. 5, 11

Cailloux, O. (2010). J-MCDA. http://sourceforge.net/projects/j-mcda/. 7

Decision Deck Consortium (2009). Strategic manifesto of the Decision Deck project. http://www.decision-deck.org/manifesto.html. 2

Decision Deck Consortium (2010). The XMCDA data standard. http://www.decision-deck.org/xmcda. 6

Grabisch, M., Kojadinovic, I., and Meyer, P. (2008). A review of capacity identification methods for Choquet integral based multi-attribute utility theory; Applications of the Kappalab R package. *European Journal of Operational Research*, 186(2):766–785. 5, 7

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The weka data mining software: an update. *SIGKDD Explorations*, 11(1):10–18. 2

Jacquet-Lagrèze, E. and Siskos, Y. (1982). Assessing a set of additive utility functions for multicriteria decision making: the UTA method. *European Journal of Operational Research*, 10:151–164. 3, 5, 11

Leistedt, B. (2010). UTAR. http://github.com/ixkael/RMCDA. 7

R Development Core Team (2005). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-00-3. 2

Roy, B. (1968). Classement et choix en présence de points de vue multiples (la méthode ELECTRE). *Revue française d'informatique et de recherche operationelle (RIRO)*, 2:57–75. 3, 5

Veneziano, T. (2010). ws-PyXMCDA. http://github.com/quiewbee/ws-PyXMCDA. 7