ICA Workshop on Geospatial Analysis and Modeling

8 July 2006, Vienna, Austria

DMA: An algebra for multicriteria spatial modeling

Salem Chakhar and Vincent Mousseau
Lamsade – University of Paris Dauphine,
Place du Maréchal de Lattre de Tassigny, 75775 Paris Cedex 16, Paris, France.
Phone: +33 (1)-44-05-49-18. Fax: +33 (1)-44-05-41-11.
E-Mail: {chakhar, mousseau}@lamsade.dauphine.fr

Absract. Map algebra is a powerful tool to implement cartographic modeling. Map algebra is now well recognized and most of its functionalities are supported by most of major geographical information systems products. Several extensions have been proposed to the original map algebra through the incorporation of new operators, the support of complex mathematical expressing and formulation, the support of other data types, the support of spatial dynamics modeling, the support of temporal dimension, and the support of visual modeling. On the other hand, spatial problems are inherently of multicriteria nature. However, neither the original map algebra nor its different extensions are able to support multicriteria aspects of spatial problems. This is due essentially to the absence of convenient operators permitting to support multicriteria modeling. The objective of this paper is to propose a new algebra especially devoted to multicriteria spatial modeling.

Keywords: Map algebra, Cartographic modeling, Multicriteria spatial modeling, GIS, Decision map.

1. Introduction

Map algebra is a powerful tool to implement cartographic modeling. It is introduced in the early 1980s by Tomlin (Tomlin, 1983) and then extended and refined in a book (Tomlin, 1990). The operands of map algebra are single-factor map layers. It uses spatial operators to generate new map layers as the result. Map algebra is originally raster data-oriented framework. However, several extensions have been proposed to the original map algebra. They concern the addition of new operators, the support of complex mathematical expressing and formulation, the support of other data types, the support of spatial dynamics modeling, the support of temporal dimension, and the support of visual modeling. Tomlin's map algebra is now well recognized and most of its functionalities are now supported by most of major geographical information systems (e.g. Microstation raster module, Arc/Info-GRID, GRASS mapcalc, the the GRID Analyst of Integraph's Modular GIS Environment, Idrisi, etc.).
On the other hand, spatial decision problems are inherently of multicriteria nature, where several, often conflicting, evaluation criteria should be taken into account for evaluating different potential alternatives. However, neither the original map algebra nor its different extensions are able to support multicriteria aspects of spatial problems. This is due essentially to the absence of convenient operators permitting to support multicriteria modeling. The objective of this paper is to propose a new algebra, called decision map algebra (or DMA), especially devoted to multicriteria spatial modeling.
This paper goes as follows. Section 2 briefly introduces Tomlin's map algebra. Section 3 deals with multicriteria analysis and modeling. Section 4 introduces the concept of decision map, the central ingredient of our algebra. Section 5 details the proposed algebra. Section 6 presents the ongoing implementation of DMA. Section 7 concludes the paper.

2. Map algebra

Map algebra is convenient framework for spatial analysis and modeling. The structure of map algebra consists of a set of map layers, primitives operations on and between map layers, and sequences of these operations (Berry, 1993). A map layer is an element of a cartographic model, defined as a

collection of map layers each of which represents the spatial distribution of a particular attribute over a common study area. The elementary unit of a map layer is the location, usually represented as a cell in a grid space. Each location on a map layer is associated with a numerical value representing the value of the corresponding attribute at that location.
The map algebra operations are used to transform map layers, location by location, into new map layers in order to extract information useful to the user (Takeyama and Couclelis, 1997). Any complex manipulation of map layers is then represented as an algebraic composition of such operations. Thus, the structure of map algebra is similar to that traditional algebra in the sense that map layer are treated as if they where numbers in an equation (Berry, 1986).
The operations are classified into (Tomlin, 1990; Takeyama and Couclelis, 1997): (i) *Local operations* that compute a new value for every location as a function of the values of one or more existing values with that location; (ii) *Zonal operations* that compute a new value for each location as function of the values from a specified layer that are associated not just with that location itself but with all locations that occur within its zone on another specified layer; (iii) *Incremental operations* that characterize each location as an increment of one-, two- or three-dimensional cartographic form. The size and shape of these increments are inferred from the value(s) of each location relative to those of its adjacent neighbors on one or more specified layers; and (iv) *Focal operations* that compute each location's new value as a function of the existing values, distances, and/or directions of neighboring (but not necessarily adjacent) locations on a specified map layer.
Recent implementations of map algebra in GIS include global (per-layer) operations (Menon et *al.*, 1992). These global operations are used, among others, for the generation of Euclidean distance and weighted cost distances maps, shortest path maps, nearest neighbor allocation maps, for the grouping of zones into connected, etc. In these extensions as in the original map algebra, statements and operations are normally expressed in an English-like syntax stimulating. Consider for instance the `LocalRating` operator that is used to characterize locations in terms of values from two or more layers. The following example is used to generate a new layer entitled *OpenDevelopment* on which `Vegetation` zones one, two and three (*HardWoods*, *SoftWoods*, and *MixedWoods*) are set to a value of six , while each location in zone zero (*Open-Land*) is set to that location's value on the exiting *Development* layer (Tomlin 1990, p. 73):

```
OpenDevelopment = LocaRating  of Vegetation with Devlopment for 0 with 6 for 1…3
```

Enhanced spatial modeling often needs a sequence of basic map algebra operations. This sequence of operations is called a *procedure*.
While map algebra is a well organized and simple framework for cartographic modeling, it has several limitations (e.g. essentially raster-oriented framework, not adequate for dynamic modeling, does not support multicriteria spatial modeling). To overcome these limitations, several extensions have been proposed. They concern (i) the addition of new operators (e.g. Caldwell 2000); (ii) the support of complex mathematical expressing and formulation (e.g. Takeyama and Couclelis, 1997); (iii) the support of other data types (e.g. Armstrong and Densham, 1996; Lin, 1998; Corripio, 2003); (iv) the support of spatial dynamics modeling (e.g. Takeyama and Couclelis, 1997); (v) the support of temporal dimension (e.g. Mennis and Viger, 2005a, 2005b); (vi) the support of visual modeling (e.g. Egnhofer and Bruns, 1994; Pullar, 2004). However, all these extensions are not sufficient to support multicriteria spatial modeling.

3. Multicriteria analysis and modeling

Multicriteria analysis (see, for e.g., Roy, 1996; Belton and Stewart, 2002), is a family of OR/MS tools that have experienced very successful applications in different domains since the 1960s. It has been advised in spatial context to overcome the limitations of GIS in spatial analysis and modeling. Indeed, GIS is a powerful tool of acquisition, management and analysis of spatially-referenced data but it is a limited tool in analysis and modeling as it is remarked by several authors (e.g. Carver, 1991; Laaribi, 2000; Malczewski, 1999; Chakhar and Martel, 2003, 2004). This is due essentially to its lack in more powerful analytical tools enabling it to deal with spatial problems, where several parties, often with conflicting objectives, are involved in the decision-making process and different, often contradictory, evaluation criteria should be considered. Naturally, multicriteria analysis is the most adequate tool to fill this gap.

It is generally assumed in multicriteria analysis that the decision maker (DM) has to choose among several possibilities, called *alternatives*. The *set of alternatives*, denoted A, is the collection of all alternatives. Selecting an alternative among this set depends on many characteristics, often contradictory, called *criteria*. Accordingly, the decision maker will generally have to be content with a *compromising solution*.

The multicriteria problems are commonly categorized as *continuous* or *discrete*, depending on the domain of alternatives (Zanakis et *al*., 1998). Hwang and Yoon (1981) classify them as (*i*) multiple attribute decision-making (MADM), and (*ii*) multiple objective decision-making (MODM). According to Zanakis et al. (1998), the former deals with discrete, usually limited, number of pre-specified alternatives. The latter deals with variable decision values to be determined in a continuous or integer domain of infinite or large number of choices. In the rest of this paper, we focalize only on MADM methods.

The general schema of MADM methods is shown in Figure 1. The first requirement of nearly all MADM techniques is a *performance table* containing the *evaluations* or *criteria scores* of a *set of alternatives* on the basis of a *set of criteria*.

Criteria are factors on which alternatives are evaluated and compared. More formally, a criterion is a function g_j, defined on A, taking its values in a totally ordered set, and representing the DM's preferences according to some points of view (Vincke, 1992). The evaluation of an alternative a according to criterion g_j is written $g_j(a)$.

Typically, in most of multicriteria problems, the DM considers that one criterion is more important than another. This relative importance is usually expressed in the form of *weights*, denoted w_j, which are assigned to different criteria. In addition to *weights*, criteria are often associated with different thresholds:

- The *indifference threshold*, generally denoted q, represents the largest difference preserving an indifference between two alternatives a and b in respect to a criterion g_j.
- The *preference threshold*, generally denoted p, represents the smallest difference compatible with a preference in favor of a in respect to criterion g_j.

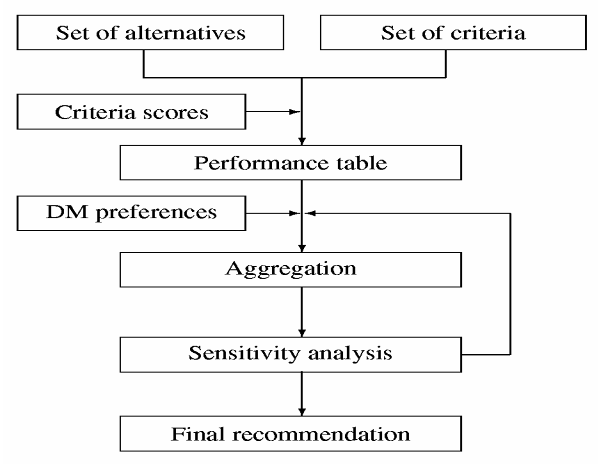


Figure 1. The MADM general model

We remark that *weights* and *thresholds* are often called *preference parameters*. The next step consists in the *aggregation* of the different criteria scores using a specific *aggregation procedure* and taking into account the decision maker *preferences.* The aggregation of criteria scores permits the decision maker to make comparison between the different alternatives on the basis of these scores. Aggregation procedures are somehow the identities of the multicriteria methods. In MADM, they are usually categorized into two great families: (*i*) utility function-based family, and (*ii*) outranking relation-based family (see Vincke, 1992).

The first family is essentially of Anglo-Saxon inspiration. Its basic principle is that the DM looks to maximize an utility function $U(x)=U(g_1(x),g_2(x),\ldots,g_m(x))$ that aggregate the partial evaluations (i.e. in respect to each criterion) of each alternative into a global one. It is important to mention that this family does not recognize the incomparability situations (i.e. the DM can compare any two alternatives) and that the indifference is transitive (i.e. if an alternative a is equivalent to another alternative b and b is equivalent to a third alternative c; than alternative a is necessarily equivalent to c). The weighted sum is an example of this family. The second family, which recognizes the incomparability situations and does not exclude the intransitivity of preference relations, is usually considered as of European inspiration. In contrast with the first family, here the aggregation methods are said to be partial. Indeed, criteria are aggregated into partial binary relation S, such that aSb means that "*a is at least as good as b*". The binary relation S is called outranking relation. The most known method in this family is ELECTRE (see Roy, 1996).

The uncertainty and the fuzziness generally associated with any decision situation require a *sensitivity analysis* enabling the decision maker to test the consistency of a given decision or its variation in response to any modification in the input data and/or in the decision maker preferences.

The aim of any decision model is to help the decision maker take decisions. The *final recommendation* in multicriteria analysis may take different forms, according to the manner in which a problem is stated. Roy (1996) identifies three types of results corresponding to three main ways for stating a problem: (*i*) *choice*: selecting a restricted set of alternatives; (*ii*) *sorting*: assigning alternatives to different pre-defined categories; and (*iii*) *ranking*: classifying alternatives from best to worst with eventually equal positions.

Finally, we mention that in spatial context, alternatives and evaluation criteria are associated with geographical entities and relationships between entities and therefore can be represented in the form of maps. Alternatives and criteria maps are generated using standard map algebra operations. In GIS-based applications, criteria generation process is often modeled in terms of flowcharts, which are intuitive and simple modeling environment, especially for users with limited knowledge.

4. Concept of decision map

In this section, we introduce the concept of *decision map*, which is the central ingredient of our algebra. First, we note that a full description of the concept of decision map and its generation process and uses are detailed in Chakhar et al (2005). Physically, a decision-map is a special kind of the map layer were the decision space is treated as a discrete surface composed of a finite number of polygonal homogenous spatial units obtained by applying a multicriteria sorting method. More formally, a decision map M is defined as the set $\{(u,\ f(u)):u\in U\}$, where U is a set of homogenous spatial units and f is a function defined as follows:

$$\begin{aligned} f:\ U &\to E \\ u &\to f(u)=\Phi[g_1(u),\ldots g_m(u)], \end{aligned}$$

where E is an ordinal (or cardinal) scale, Φ is a multicriteria sorting model and $g_j(u)$ is the performance of spatial unit u in respect to criterion g_j. Accordingly, a decision map summarizes the preferential information of the decision maker relatively to a set of conflicting evaluation criteria into an ordinal or cardinal information.

The first step for the construction of the decision map consists in the generation of criteria maps. Each criterion map represents a specific theme and is composed of set of homogenous spatial units; each of which is associated with one evaluation, $g_j(u)$. Then, these criteria maps need to be superposed to obtain an intermediate map, which is composed of a new set of spatial units resulting from the intersection of the boundaries of the spatial units of the different criteria maps. Each spatial unit is characterized with a vector of m evaluation relative to the m criteria maps. Formally, to each spatial unit u, we associate the vector $[g_1(u),g_2(u),\ldots,g_m(u)]$.
Description of territory as a set of units is not new in land management (Joerin and Musy, 2000). In classical cartographic modeling, these units are often defined through census tracts or administrative and political boundaries. In this paper as in Joerin and Musy (2001), the initial subdivision of territory into homogenous units (Joerin uses the term zone instead of unit) should respect the spatial natural (forest, body of water) and human (highways, parks, buildings) boundaries.
To generate a final decision map, we should first use an aggregation mechanism to aggregate the vector associated with each spatial unit u in the intermediate map into one global evaluation. Mathematically, we write: $g(u)=\Phi[g_j(u)]_{j\in F}$. F is the criteria family and Φ is defined as follows:

$$\begin{aligned}\Phi:\ & E^m \rightarrow E\\ & [g_1(u),g_2(u),\ldots,g_m(u)] \rightarrow g(u)\end{aligned}$$

The aggregation mechanism Φ permits to assign each unit to one or several predefined categories on the scale E. In this paper, the multicriteria sorting model used is ELECTRE TRI (see Appendix B for an overview and Yu (1992) for a complete description of ELECTRE TRI method).
The major merit of a decision map is that it permits to use outranking-based aggregation models in spatial modeling problems. In fact, most of multicriteria analysis based-GIS use methods based on utility function-like aggregation methods (e.g. Carver, 1991; Jankowski, 1995). Theses aggregation models still dominate today and only few works (e.g. Martin *et al.*, 2000; Joerin and Musy, 2000, 2001) use outranking-based aggregation models (see Roy, 1996). However, these last ones are more suitable in spatial context since they permit "to consider both objective and subjective criteria and require fewer amount of information from the decision maker" (Malczewski, 1999). In addition, they do not impose the transitivity of indifference and tolerate the incomparability situations. But the major (technical) drawback of outranking methods is that they are not suitable for problems implying a great or infinite number of alternatives since they require pairewise comparison across all criteria (Chakhar et *al.*, 2005).
Subdividing the study area into homogenous spatial units permits to reduce significantly the number of potential alternatives to be evaluated and leads to "a manageable set of alternatives" (Hall et al, 1992; Wang, 1994; Joerin and Musy, 2001) and outranking methods can be applied quite easily. Indeed, in spatial multicriteria modeling, potential alternatives are represented through one of three atomic spatial entities, namely point, line or polygon (Malczewski, 1999; Chakhar and Mousseau 2004). Therefore, in a facility location problem, potential alternatives take the form of points representing different potential sites; in a linear infrastructure planning problem (e.g. highway construction), potential alternatives take the form of lines representing different possible routes; and in the problem of identification and planning of a new industrial zone, potential alternatives are assimilated to a set of polygons representing different candidate zones. In raster-based GIS, these three alternatives are defined as a single pixel, collection of linearly adjacent pixels, and a set of contiguous pixels, respectively.
By using the decision map concept, punctual, linear and polygonal decision alternatives can be defined, respectively, as an individual homogenous spatial unit, a collection of linear adjacent homogenous spatial units, and a collection of contiguous homogenous spatial units, respectively. These ways of modeling are illustrated in Table 1.
It is important to remark that in many real-world applications, one may be called to represent alternatives with a combination of two or more atomic entities (Malczewski, 1999; Chakhar and

Mousseau 2003, 2004). In schools partitioning problem, for instance, decision alternatives can be assimilated to a combination of points and polygons where points represent schools and polygons represent zones to serve. A set of "point-point" composed alternatives may represent potential paths in a shortest path identification problem. In this paper, we consider only atomic decision alternatives. We just mention that in Chakhar and Mousseau (2004) we have proposed several other examples of spatial problems implying composed alternatives. We have also introduced a solution that may used to handle some of these problems.

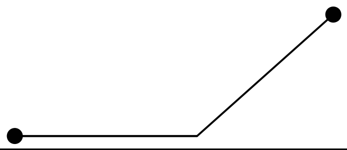
Problem	Representation in GIS		Proposed modeling
Location problems	Point	●	one spatial unit
Linear infrastructure problems (e.g. highways)	Line		a series of linearly adjacent spatial units
Land used and management problems	Polygon		one or several contiguous spatial units

Table 1. Modeling spatial decision alternatives

To conclude this section, we compare our decision map concept to maps generated through conventional cartographic modeling:

- Cartographic modeling is essentially an automatic procedure with no or less (generally a priori) interaction with the decision maker. Our decision map generation process (see Chakhar et al 2005) is largely controlled by the decision maker;
- Maps produced through cartographic modeling are essentially presentation-oriented ones and they are roughly used for an effective spatial decision-aid activity. Our decision maps are decision-oriented ones. In addition, decision map is a generic tool that may serve to the generation of potential alternatives and can be extended to support collaborative and communicative spatial decision making (as it is detailed in Chakhar et al., 2005);
- Decision map permits to explicitly represent spatial preferences of the decision maker. In classic cartography modeling, preferences of the decision maker are often reduced to a tabular representation, often with no explicit relation to their spatial locations;
- Aggregation is performed in early steps in the cartographic modeling process, which may lead to a substantially lost of the preferential information. In our approach, aggregation is performed in latter steps in the generation process;
- Aggregation in our approach is based on outranking relations, which we think more appropriate to deal with spatial decision making. Especially, this permits the integration of both qualitative and qualitative criteria in problem modeling.

5. Proposed decision map algebra

To support multicriteria modeling, we introduce in this section a new algebra that we call decision map algebra (or DMA). It is important to note that DMA does not substitute Tomlin's map algebra:

DMA is an extension of Tomlin's map algebra that is especially devoted to multicriteria spatial modeling. Further, we mention that in practice we need to combine both of them for a complete and efficient modeling. Moreover, several operators of DMA use the ones of map algebra.
DMA contains three sorts of operands: (i) geographic objects; (ii) geographic maps; and (iii) some other operands required to multicriteria modeling. In addition to the standard operators of map algebra, DMA contains several new ones devoted to multicriteria modeling.

5.1 Primitives and definitions

This section defines the terms used in the specification of DMA. DMA contains several data types. The relationships among these data types are depicted in Figure 2. In this figure, the symbols '`s`' and '`agg`' signify the specialization/generalization and the aggregation relationships, often used in object oriented modeling; the double brakes symbol '`{}`' means '*a collection of*' ; the '`xor`' is the XOR binary operator. Geographic objects and maps are represented by oval or rectangular shapes with a solid line. The other shapes (with a dashed line) represent descriptive attributes and multicriteria concepts that are introduced for the purpose of DMA.
The basic data type in DMA (and also in most of map algebra-like languages) is `map-layer`. A `map-layer` is the most elementary data that contains the map image and other documentary items for the `map-layer` including the global geographic reference system, `rSystem`; the map scale, `mScale`; etc. There are two types of `map-layer` data type: *raster* and *vector*. The map image of a raster `map-layer` is composed of pixels. Each element of a aster `map-layer` is called `gPixel`. The map image of a vector `map-layer` is a collection of three types of geographical objects:

- `gPoint`: is an individual, one-dimensional repressing a punctual entity in real-world.
- `gLine`: is a two-dimensional object representing a linear entity in real-world.
- `gPolygon`: is a three-dimensional object representing a polygonal entity in real-world.

In DMA, a `map-layer` is either of raster or vector nature. This is ensured with the '`xor`' symbol in Figure 2. For the purpose of DMA, a collection of `gPixel`, `gPoint`, `gLine` and `gPolygon` are denoted `xSet`, `pSet`, `lSet` and `ySet`, respectively. A `gPixel` can be associated with one value which may be any arbitrary data type representing a natural or an artificial attribute. A `gPoint`, `gLine` or `gPolygon` can be associated with several values representing a set of descriptive attributes that apply to these objects as a whole.
The `gPixel` elements of a raster `map-layer` are fully identified with their XY coordinates on the map image. The `gPoint`, `gLine` and `gPolygon` associated with a vector-based `map-layer` need to be uniquely identified.
We define three new subclasses of `map-layer` data type: `alternative-map`, `criterion-map` and `decision-map` (see Figure 2). An `alternative-map` is a special kind of `map-layer` which, in addition to the general information associated with any `map-layer`, contains a collection of specific objects called *alternatives*. It is important to mention that alternatives are normally generated through `decision-maps` as explained in section 4. But the `alternative-map` data type is included in DMA to deal with spatial problems for which the use of a `decision-map` is difficult or not possible.
There are basically three types of alternatives: `pAlter`, `lAlter` or `yAlter` representing punctual, linear or polygonal decision alternatives, respectively. A collection of `pAlter`, `lAlter` and `yAlter` are denoted `pAlters`, `lAlters`, and `yAlters`, respectively. When it is necessary, the generic terms `anAlter` and `sAlters` will be used to denote an alternative and a set of alternatives, regardless to their nature. Each alternative is characterized with a vector of values relative to a set of criteria or attributes and applies to these objects as a whole. It is important to mention that for a given spatial decision problem, each `alternative-map` contains only one kind of alternatives; each of which is uniquely identified.

A criterion-map is a specific, mono-valued, map-layer where each spatial object (i.e. gPixel, gPoint, gLine or gPolygon) is characterized by one value representing the evaluation of this element in respect to a given criterion. A criterion-map is different from a simple map-layer in the sense that it models some subjective information. That is, two persons may give different values for the same spatial object, along with their study perspectives.

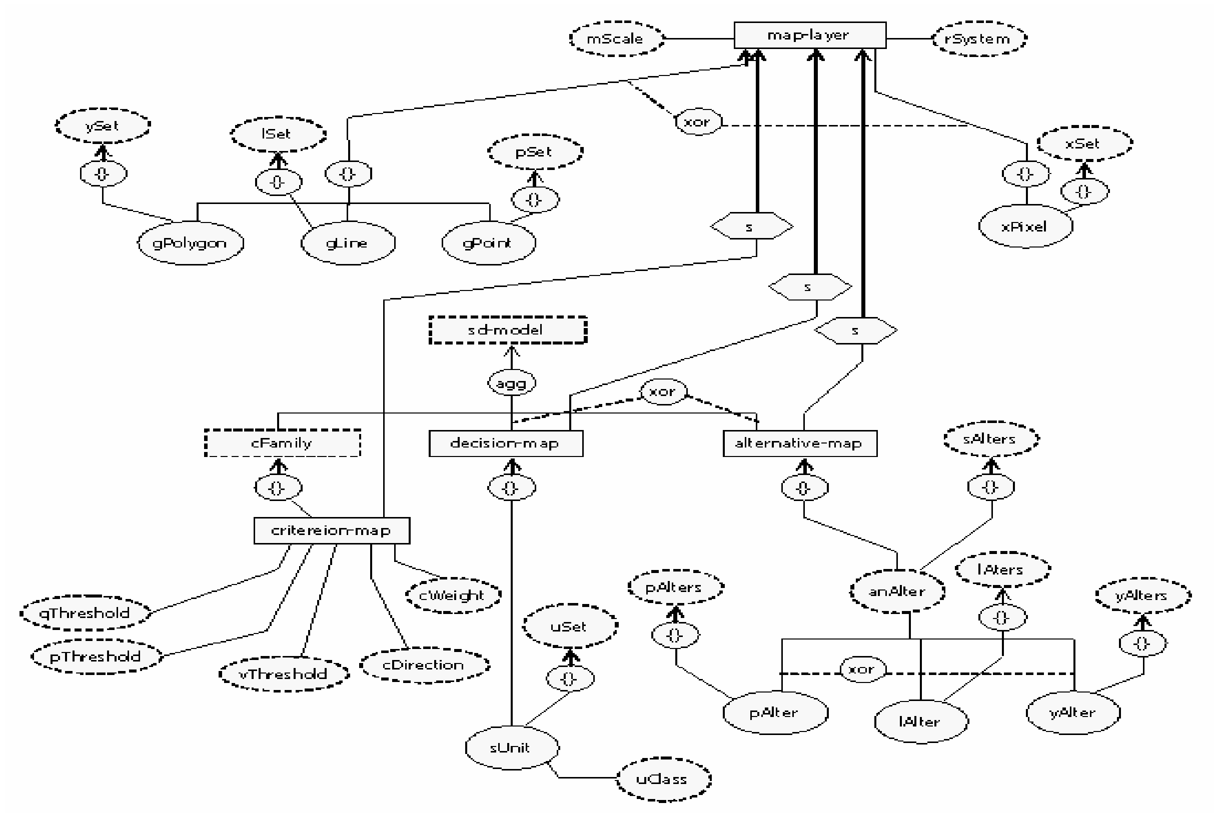


Figure 2. Data types relationships

In multicriteria analysis, each criterion is often associated with a weight, a direction of optimization and several preference parameters (see section 3). To take into account these information, we associate to each criterion-map the following data types:

- cWeight: An importance degree reflecting the power of the criterion during the comparaison of alternatives.
- cDirection: The optimization direction of the criterion which may be maximization or minimization.
- qThreshold and pThreshold: corresponding to the indifference and preference thresholds introduced in section 3;
- vThreshold: represents the smallest difference between the performance of two alternatives incompatible with the outranking assertion as explained in Appendix B.

For coherence reasons, we need to have: $0 \leq$ qThreshold $\leq$ pThreshold $\leq$ vThreshold. A collection of evaluation criterion-maps are called cFamily (for criteria family) in DMA.
As mentioned earlier, a decision-map is a special kind of map-layer where the decision space is treated as a discrete surface composed of a finite number of polygonal homogenous spatial units. Each

element of a decision-map is of type `sUnit` (for spatial unit). A collection of spatial units is denoted by `sSet`.
As explained earlier, to operationalize a `decision-map`, we to need to assign all of its spatial units to a set of predefined categories through the multicriteria sorting method ELECTRE TRI. We denote by `uClass` the class to which a spatial unit is assigned.
To support multicriteria modeling, we add a new data type, called `sd-model` (for spatial decision model). The `sd-model` is an aggregation of one `decision-maps` (or an `alternative-map`) and at least two `criterion-maps` data types.
Finally, we mention that several other data types (as decision table, aggregation operator, preference structure and choice function) are needed to formalize our algebra. These additional data types are not included in Figure 2 but they will be introduced progressively hereafter.

5.2 Formal specification of DMA

To specify our DMA, we adopt the algebraic specification method of Guttag and Horning (1978). The algebraic specification methods are mainly used in software engineering to describe the behavior of complex systems. An algebraic specification consists of three parts (Guttag and Horning; 1978; Dorenbeck and Egenhofer, 1991): (i) a set of *sets* including the data type to define and the types needed to define its properties; (ii) a set of *operations* defined on the operands. Each operation is defined by its name, the Cartesian product of the inputs sorts and the sort of the result; and (iii) a set of *axioms* (or *equations*) that describe the behavior of operations.
In the following, we discuss the formal specification of some data types. The specifications of the other data types are provided in Appendix A.

5.2.1 Specification of `sUnit` data type

The most elementary data type is the spatial unit, denoted `sUnit`. The formal specification of `sUnit` data type in shown in Figure 3. The first operator associated with `sUnit` is `ASSIGN` which permits to set the performance of a `sUnit` in respect to a given criterion provided as parameter. The `ASSIGN` is a *hidden* function (Chan and White, 1987), i.e. it is not part of the algebra and serves its purpose in the specification only. The operator `ADJACENT` tests if two spatial units are adjacent or not. `GET-EDGES` and `GET-VERTICES` permit to extract, respectively, the set of edges and vertices for a given spatial unit. The specifications of these three spatial manipulation operators are similar to the ones associated with `gPolygon` data type provided in Appendix A. The `SCORE` operator returns the performance of a spatial unit in respect to a given criterion. This operator is quite straightforward and its specification is not detailed in Figure 3.
Data type `sUnit` is associated with a set of operators devoted to implement ELECTRE TRI model. The operators `PCONCORDANCE`, `CONCORDANCE`, `PDISCORDANCE` and `DISCORDANCE` implement Equations 1, 2, 3 and 4 in Appendix B, respectively. The `PCONCORDANCE` and `PDISCORDANCE` operators take in input two values and a criterion and generate a real that indicates the partial concordance and discordance indices, respectively. The two values correspond to the scores of the spatial unit and the profile to be compared in respective to the criterion under consideration.
To implement the `PCONCORDANCE` and `PDISCORDANCE` operators, we need to define the concept of *decision table*, denoted `dTable`. A decision table is a method to specify formally the behavior of operations, particularly those which can be described by a series of rules. It consists of two parts (Dorenbeck and Egenhofer, 1991): (i) a set of *conditions* which have to be satisfied simultaneously; and (ii) the corresponding *actions* to be taken upon conditions. Decision tables are most naturally presented in the form of a table with the set of conditions being put into the upper half of the table and the corresponding set of actions underneath. The specification of `dTable` is provided in Appendix A. Each `dTable` has two operators: `CRETATE` and `ACTION`. The specification of the `CRETATE` operator is

DEFFERED (Meyer, 1988; Dorenbeck and Egenhofer, 1991), because it depends upon the particular context. The general structure of decision tables associated with PCONCORDANCE and PDISCORDANCE are shown in Table 2 and Table 3, respectively.

v1- v2	≥ p	≤ q	< p ∧ > q
ACTION	0	1	[p - v1 + v2]/[p - q]

Table 2. Decision table associated with PCONCORDANCE operator

v2- v1	≥ p	≤ v	< v ∧ > p
ACTION	0	1	[v - v1 + v2]/[v - p]

Table 3. Decision table associated with PDISCORDANCE operator

```
Type: sUnit
set: map-layer, aCriterion, cFamily, sUnit, aProfile, pSet, lSet, dTable, real, value,
     Boolean

Syntax:
ASSIGN       sUnit x aCriterion x value → sUnit
ADJACENT     sUnit x sUnit → boolean
GET-VERTECES sUnit → pSet
GET-EDGES    sUnit → lSet
SCORE        sUnit x aCriterion → real
PCONCORDANCE value x value x aCriterion → real
CONCORDANCE  sUnit x aProfile x cFamily → real
PDISCORDANCE value x value x aCriterion → real
DISCORDANCE  sUnit x aProfile x cFamily → real
SIGMA        sUnit x aProfile x cFamily → real
OUTRANK      sUnit x aProfile x cFamily x value → boolean


Axioms:
u: sUnit; h:aProfile; f:cFamily; g:aCriterion; t1,t2,t3:dTable; v:value

PCONCORDANCE(u,h,g)
  =t1.action(SCORE(u,g),SCORE(h,g),g)

CONCORDANCE(u,h,f)
  =   [Σg ∈ f PDISCORDANCE(u,h,g) * g.cWeight] /
      [Σg ∈ f g.cWeight]

PDISCORDANCE(u,h,g)
  =t2.action(SCORE(u,g), SCORE(h,g),g)

DISCORDANCE(u,h,f)
  = Πg∈f∧(PDISCORDANCE(u,h,g)>CONORDANCE(u,h,g))
      [1-PDISCORDANCE(u,h,g)]/[1-CONORDANCE(u,h,g)]

SIGMA (u,h,f)
 = CONCORDANCE(u,h,f)* DISCORDANCE(u,h,f)

OUTRANK(u,h,f,v)
  =t3.action(u,h,SIGMA(u,h,f),SIGMA(h,u,f),v)
```

Figure 3. Formal specification of sUnit data type

In Tables 2 and 3, the parameters q, p and v should be mapped to qThreshold, pThreshold, and vThreshold attributes; and v1 and v2 correspond to SCORE(h,g) and SCORE(u,g), respectively. All of them are provided as parameters for ACTION operator.

The CONCORDANCE and DISCORDANCE operators take in input a spatial unit, a profile and a family of criteria; and generates a real value in [0,1] corresponding to the global concordance and discordance indices. The cWeight attribute used in the specification of CONCORDANCE operator correspond to criterion weight (see Figure 2). The SIGMA operator permits to compute the credibility indices as in Equation 5 in Appendix B.
The operator OUTRANK permits to get the preference situation. As mentioned in Appendix B, there are four disjunctive possible situations that hold when comparing a spatial unit u to a profile h: aIh, aPh, hPa or aRh. The operator OUTRANK uses the concept of decision table. The decision table associated with OUTRANK operator is shown in Table 4. Note that u and h in Table 4 correspond to the spatial unit and the profile to be compared, respectively; and the values v1 and v2 in Table 4 correspond to SIGMA(u,h,f) and SIGMA(h,u,f), respectively. It is easy to see the four preference situations mentioned in Appendix B in the underneath part of Table 4. In addition to u, h, SIGMA(u,h,f) and SIGMA(u,h), the ACTION operator takes the value of the cutting level λ (see Appendix B), and returns the corresponding decision.

v1	≥ λ	≥ λ	< λ	< λ
v2	≥ λ	< λ	≥ λ	< λ
ACTION	uIh	uPh	hPu	uRh

Table 4. Decision table associated with OUTRANK operator

5.2.2 Specification of decision-map data type

Figure 4 specifies the decision-map data type. As it is shown in this figure, the syntax part of decision-map data type contains four operators. The MAKE operator creates a decision map as the intersection of a set of criterion-maps. The result of this intersection is an initial decision-map. Each spatial unit of this initial decision-map is associated with a set of values relative to different criteria.

```
Type: decision-map
set: map-layer, criterion-map, sUnit, cFamily, aOperator, aProfile, sProfiles
Syntax:
MAKE       criterion-map x…x criterion-map → decision-map
CLASSIFY   desicion-map x cFamily x sProfiles → desicion-map
GROUP      decision-map x cFamily x aOperator → decision-map
MERGE      decision-map x cFamily x sUnit x…x sUnit x aOperator → decision-map

Axioms:
d: decision-map; c1,…,cm,g: criterion-map; f;cFamily; v: value; h:aProfile; b:sProfiles

MAKE(c1,…,cm)
  =INTERSECT(c1,…,cm)

CLASSIFY(d,b,f)
 = ∀(u) (u∈d)
   [∀(h) (u∈b)
    if OUTRANK(u,h,f,cLevel) then u.uClass ← h+1
      ]

MERGE(d,u1,u2,op,f)
  =u.make(d,[GET-VERTECES(u1) ∪ GET-VERTECES(u2)] \
            [GET-VERTECES(u1) ∩ GET-VERTECES(u2)])
   ∀(g)(g∈f)[ASSIGN(u,g,op.combine(SCORE(u1,g),SCORE(u2,g)))]

GROUP(d,op)
  = ∀(u1)∀(u2)(u1 ∈ d)(u2 ∈ d) ∧ (u1 <> u2)
    [if ADJACENT(u1,u2) ∧ u1.uClass = u2.uClass then MERGE(d,u1,u2,op,f)]
```

Figure 4. Formal specification of decision-map data type

The operator CLASSIFY is the implementation of the pessimistic assignment procedure of ELECTRE TRI (see Appendix B). It permits to assign each spatial unit to a predefined set of categories defined in terms of their profiles. As it is shown in Appendix B, the pessimistic assignment procedure with ELECTRE TRI compares each alternative u (here alternatives are the spatial units) to all profiles starting from the best to the worst. The spatial unit u is assigned to first class for which u outranks its lower limit.
The MERGE operator simply groups two or more adjacent spatial units. It uses the MAKE operator, inherited from gPolygon data type, to create a new spatial unit. The evaluations of the new spatial unit in respect to all criteria are obtained by aggregating, using the aOperator, the initial evaluations of original spatial units. The specification of the MERGE operator is shown for two spatial units. The generalization to more than two spatial units is straightforward.
The specification of the aOperator is defined in Appendix A. A aOperator must provide operation to combine a series of values. Since a large set of aggregation operators are available, the specification of its CRETATE operator is DEFFERED (Meyer, 1988; Dorenbeck and Egenhofer, 1991), because it depends upon the particular aggregation operator.
The GROUP function takes a decision-map and an aggregation operator, aOperator; and generates a new decision-map by merging all adjacent spatial units that are assigned to the same class. It is simply the generalization of the MERGE operator to the entire decision-map.

5.2.3 Specification of `criterion-map` data type

Data type criterion-map specified in Figure 5 has two operators. The operator MAKE takes as input a predefined map algebra *procedure*, that is, a sequence of map algebra operations that takes in input one or more map-layers; and generates a criterion map.
The operator SET permits to set a preference parameter. The aParameter parameter may take the values of weight, direction, indifference, preference, or veto.
Within the multicriteria methods based on utility-function aggregation operator, the term criterion is a generic one that includes objectives and attributes. An objective is a statement about the desired future which is made operational by assigning to it one or more attributes describing a geographical entity or the relationship between geographical entities (Malczewski, 1999). The specification of criterion-map data type presented above of does not include these aspects of criteria modeling but they will be included in the future. This will enhance the applicability of DMA.

```
Type:  criterion-map
set:  map-layer, criterion-map, cWeight, cDirection, qThreshold, pThreshold, vThreshold,
      aParameter, value, procedure

Syntax:
MAKE  procedure → criterion-map
SET   criterion-map x aParameter x value → criterion-map

Axioms:
c: criterion-map;  a:aParameter; v: value

SET(c,a,v)
 = c.a ← v
```

Figure 5. Formal specification of criterion-map data type

5.2.4 Specification of `sd-model` data type

As mentioned earlier, the sd-model data type is an aggregation of a decision-map and at least two criterion-maps. It is important to reminder that the concept of decision-map may not apply for

some spatial problems. In this case, we may use an `alternative-map`. In the specification below, we suppose that a `decision-map` is in use. The specification with an `alternative-map` is not included in this paper.
The specification of `sd-model` data type is provided in Figure 6. It contains different multicriteria modeling operators. The `P-ALTERNATIVE` operator permits to generate a set of punctual alternatives. It takes a `decision-map` and returns a set of spatial units verifying the constraints ensured by the expression `<aCriterion><bOperator><value>`; where `aCriterion` represents an evaluation criterion and `bOperator` is a binary operator.
The `L-ALTERNATIVE` operator permits to generate a set of linear alternatives. It takes a `decision-map` and two spatial units representing the start and the end points. It generates a set of corridors relating start and end spatial units.
In multicriteria modeling, we may need to eliminate from consideration some alternatives that present some undesirable aspects. These restrictions imposed, by the nature or by human beings, on alternatives are called *constraints*. Constraints may also be called *admissibility criteria* since they represent criteria that must be fully verified by any alternative; true criteria represent conditions that need to be satisfied at maximum (Laaribi, 2000). The constraints dichotomize a set of alternatives under consideration into two categories: acceptable (or feasible) and unacceptable (or unfeasible). The `CONSTRAINT` operator associated with the `sd-model` implements the concept of constraint. It takes a set of alternatives and a constraint similar to the one used with `P-ALTERNATIVE` operator; and returns all the alternatives verifying the constrain. When several constraints are required, they may be modeled sequentially; each of which takes the result of the previous constraint as an input.
The alternatives generated by `L-ALTERNATIVE` operator are composed of a set of spatial units, each of which is associated with a set of partial evaluations. The `EVALUATE` takes a linear alternative, a family of criteria and an aggregation operator, `aOperator`, and returns the new partial evaluations that apply to the alternative as a whole. We note that the spatial units $u_1,\ldots,u_r$ in the specification of `EVALUATE` operator are the individual spatial units composing the alternative. Intuitively, the `EVALUATE` operator is not needed for punctual alternatives (generated by the `P-ALTERNATIVE` operator).
The `SCORE` operator is a hidden one. It takes an alternative and a criterion and returns the partial performance of that alternative in respect to the criterion. The `P-VECTOR` returns the performances of an alterative in respect to a family of criteria. The result is stored in a conventional ADT list. This function uses the `INSERT` operator associated with the ADT list (called `aList` in Appendix A).
The `PAYOFF` operator takes a set of alternatives and a family of criteria and returns the performance matrix. In the formal specification of this operator, the result matrix is defined as a list of performance vectors, i.e., each line is simply the performance vector of an alternative.
The `DOMINATE` is an implementation of the dominance relation Δ used in multicriteria modeling. The dominance relation is defined for two alternatives *a* and *b*; and a family of criteria `F` as follows:

$$a \Delta b \Leftrightarrow g_j(a) \geq g_j(b);\ j \in F,$$

with at least one strict inequality. Within DMA, `DOMINATE` operator takes a set of alternatives of the same type and returns all the non-dominated ones in respect to a family of criteria, `cFamily`.
In real-world problems, criteria scores can be quantitative or qualitative and can be expressed according to different measurement scales (ordinal, interval, and ratio). However, several multicriteria problems require that all criteria evaluations are expressed on the same scale. The `NORMALIZE` operator is introduced to re-scale, when it is necessary, the different criteria scores between 0 and 1. It takes in input a set of alternatives of the same type, a family of criteria and a normalization procedure, denoted `nProcedure`. The specification of a `nProcedure` (see Appendix A) is similar to that of `aOperator`. Since different methods of normalization are available (see Table 5 for some examples), the `CRETATE` operator of a `nProcedure` is a `DEFFERED` one.

#	**Rescaled value**
1	$g(a) / maxi_i\ g(a_i)$
2	$[g(a) - min_i\ g(a_i)] / [maxi_i\ g(a_i) - min_i\ g(a_i)]$
3	$g(a) / \sum_i g(a_i)$
4	$g(a) / \sqrt{[\sum_i g(a_i)^2)]}$

Table 5. Some normalization techniques (g(a) is the initial evaluation)

The operator AGGREGATE uses a multicriteria aggregation procedure to aggregate all the partial evaluations into a global one. The aggregation procedures are denoted aProcedure (see Appendix A for the specification of aProcedure data type).
The CHOICE, SORT and RANGE operators correspond to the three types of recommendations in multicriteria analysis that have mentioned in Section 3. The definition of these three operators needs the introduction of a new concept: *preference structure*. A preference structure pStructure permits to the decision maker to articulate his/her preferences when comparing two alternatives. In multicriteria analysis, a pStructure is often operationalized through a criteria function. Thus, a pStructure may be associated with several thresholds. Here, we adopt the most general way that considers the presence of two preference thresholds: an indifference threshold, called qThreshold, and a preference threshold, called pThreshold. A such pStructure permits to model three situations when comparing two alternatives a and b:

$$aPb \Leftrightarrow g(a) > g(b) + pThreshold$$
$$aQb \Leftrightarrow g(b) + pThreshold \geq g(a)] > g(g) + qThreshold$$
$$aIb \Leftrightarrow g(b) + Threshold \geq g(a) \text{ and } g(a) + qThreshold \geq g(b)$$

The P, Q and I symbols are the binary relations of strict preference, weak preference and indifference, often used in preference modeling. Any pStructure can be fully characterized in terms of its characteristic function. For example, we may associate to the preference structure above a characteristic function S defined as:

$$aSb \Leftrightarrow aPb \vee aQb \vee aIb$$

The formal specification of pStructure data type is provided in Appendix A. It includes three operators, P, Q, and I, implementing the three situations mentioned above; and an operator, S, implementing the characteristic function.
The CHOICE operator may be implemented in terms of a choice function defined on a pStructure. A choice function C is a function that associates to a set B a subset C(B) of B. For example, we may associate to pStructure defined above, the following choice function:

$$C(B)=\{ a \in B: aSb, \forall b \in B\}.$$

Note that other functions may also apply as for example: $C(B)=\{ a\in B: \neg\exists b \in B : bPa\}$. As it is shown in Figure 6, the CHOICE operator is defined in terms of the above choice function. It takes in input a set of alternatives, a preference structure and a family of criteria; and returns the alternatives that verify the choice function S associated with the preference structure.
The RANGE and SORT operators are defined by using the CHOICE operator. The RANGE operator establishes a partial pre-order on the set of alternatives A. A pre-order consists of (i) a set of

equivalence classes and (ii) an order relation upon these classes. Thus, it can be implanted as a series of CHOICE operators; each is of the following form:

$$C(B)=\{\ a\in B:\ aPb \vee aIb,\ \forall\ b\in B\}.$$

In each step i, RANGE returns the most preferred alternatives from the set B^i where $B^i=B\backslash B^{i-1}$; and $B^1=B$. In the first step, the choice function is applied to all the alternatives in the set B. The next steps use the set generated in the previous step minus the selected alternatives as input. As it is shown in Figure 6, the result of the RANGE operator is a list of ordered set of equivalence classes. The INSERT and GET operators are those of aList data type. They are used to insert and to get the alternatives for a given equivalence classe (identified by its position i in the list).
Compared to CHOICE and RANGE operators, the SORT operator has an important characteristic: the two first ones compare each alternative to all the other ones while the third one compares each alternative to a set of p profiles defining a set of p+1 predefined categories. The implementation of the SORT operator is similar to operator CLASSIFY associated with decision-map data type. The only difference is related to the fact that CLASSIFY uses the preference structure associated with ELETCTRE TRI method and works on decision-map, while SORT is a more general one and can be used to implement other multicriteria sorting methods.

```
Type: sd-model
set: map-layer, decision-map, criterion-map, sUnit, uSet, cFamily, aAlter, pAlters, lAlters,
sAlters, aProcedure,  nProcedure, aOperator

Syntax:
P-ALTERNATIVE   decision-map x aCriterion x bOperator x value → pAlters
L-ALTERNATIVE   desicion-map x sUnit x sUnit  → lAlters
EVALUATE        aAlter x cFamily x aOperator → aAlter
SCORE           aAlter x aCriterion → real
P-VECTOR        aAlter x cFamily → aList
PAYOFF          aSet x cFamily →  pTable
CONSTRAINT      sAlters x aCriterion x bOperator x value → sAlters
DOMINATE        sAlters x cFamily → uSet
NORMALIZE       sAlters x cFamily x nProcedure → decision-map
AGGREGATE       decision-map x cFamily x aProcedure → decision-map
CHOICE          aSet x pStructure x cFunction → aSet
SORT            aSet x sCatogries x sProfiles → aSet
RANGE           aSet x rdirection → aSet

Axioms:
m,r: decisision-map; u:sUnit;  f: cFamily; a: aProcedure; n: nProcedure; s:aSet; b:pStructure;
c:cFunction; x: aAlter; y:sAlters; v,l,m:aList

P-ALTERNATIVE(d, g, op, v)
  ={u : u ∈ d ∧ SCORES(u,g) op v }

L-ALTERNATIVE (d,s,e)
  ={u_i : u_i ∈ d ∧ u_1 = s ∧ u_n = e ∧ ADJACENT(u_i,u_i+1)}

CONSTRAINT(y, g, op, v)
  ={x : x ∈ y ∧ SCORES(x,g) op v }

EVALUATE(x,f, op)
  =∀(g)(g∈f)[ASSIGN(u,g,op.combine(SCORE(u_1,g),…,SCORE(u_r,g)))]

P-VECTOR(x,f)
  = i ←1
     ∀(g)(g∈f)
     [insert(v,SCORE(x,g),i)
      i ← i+1
     ]
```

Figure 6. Formal specification of sd-model data type (*continued in the next page*)

```
PAYOFF(s,f)
  =∀(a) (a∈s) m.insert(P-VECTOR(a,f))

DOMINATE(y,f)
  = {x:  x ∈ y ∧ ∀(x')(x'∈y)
         [∀(g)(g∈f) SCORE(x,g)  ≥  SCORE(x',g)] ∧
         [∃(g')(g'∈f) SCORE(x',g') > SCORE(x,g')]
    }

NORMALIZE(y,f,n)
  = ∀(x)∀(g) (x ∈ y) (g ∈ f)
      SCORE(x,g) ← n.combine(SCORE(x1,g),…,SCORE(xr,g))

AGGREGATE(x,f,o)
  = o.combine(P-VECTOR(x,f))

CHOICE(s,p,f)
  = {a ∈ s : p.S(a,b,g) ∀ b ∈ s ∀ g ∈ f}

RANGE(s,p,f)
 = i ←1
   While s<>ϕ
     [insert(l,CHOICE(s,p,f),i)
      i ← i+1
      s ← s\GET(i-1,l)
     ]

SORT(s,b,p,f)
 = ∀(x) (x∈s)
   [∀(h) (u∈b)
    if p.S(a,h,g) then insert(l,x,h+1)
   ]
```

Figure 6. Formal specification of `sd-model` data type (*continued*)

6. Implementing DMA

The implementation of DMA is ongoing. DMA is being implementing through C++ on ArcGIS 9.1. Each data type in DMA is defined as a class and the operations associated with it are defined as methods for these classes. Figure 7 illustrates the generic definition some data types. The other data types are defined in similar way.
Each of these classes contains, in addition to the methods corresponding to the operations of the data type that they implement, two specific methods representing the constructor and destructor of the class. These two methods are not shown.
Figure 7 shows also the implementation of `DISTANCE` and `SET` operators with `gPoint` and `criterion-map` data types.
The piece of code in Figure 8 shows a didactic example illustrating the use of some data types from DMA. The objective of this example is to select a corridor for some linear infrastructures. This example may apply in problems like the construction of highways, pipelines, etc. As mentioned earlier, within a `decision-map`, a corridor is modeled as a sequence of linearly adjacent spatial units linking two spatial units representing the start and end points. They are denoted s and e in Figure 8.
First, the example generates an initial `decision-map`, called `r`, by intersecting three `criterion-maps c1, c2 and c3`; which we suppose that they have been already created. Then, the example uses the `CLASSIFY` operator to generate a final `decision-map`. To apply this operator and for the purpose of this example, `cFamily` and `sProfiles` are simply defined as one-dimensional and two-dimensional array, respectively.

```
class gPoint{
//...
public:
    gPoint MAKE(double, double);
    boolean ISEQUAL(gPoint, gPoint);
    double DISTANCE(gPoint, gPoint);
    double X(gPoint);
    double Y(gPoint);
//...
};

double gPoint::DISTANCE(gPoint p, gPoint q){
 return sqrt(((p.X - q.X) * (p.X - q.X)) + ((p.Y - q.Y) * (p.Y - q.Y)));
};

class map_layer{
private:
     string rSystem;
     double mScale;
//…
public:
     map-layer MAKE(string name);
     gPoint PUT_P(map_layer, double,  double);
     gLine  PUT_L(map_layer, gPoint,  gPoint);
     gPolygon PUT_Y(map_layer, pSet);
     map_layer INTERSECT(map_layer, map_layer);
//…
}

class criterion_map{
private:
    float cWeight;
    float qThreshold;
    float pThreshold;
    float vThreshold;
//...
public:
    criterion_map SET(char, float);
//...
};

criterion_map criterion_map::SET(char type, float value){
   if (type='q') this->qThreshold=value;
   if (type='p') this->pThreshold=value;
   if (type='v') this->vThreshold=value;
};
```

Figure 7. Generic definitions of some data types in C++

```
//…
main()
{
 decision_map r;
 criterion_map c1, c2, c3;
 r.intersect(c1,c2,c3);
 string cFamily[3];
 cFamily =(c1,c2,c3};
 double sProfiles[4][3];
 //…
 r.classify(r,b,f);
 sUnit s, e;
 //…
 uSet corridors;
 corrirods=r.L-ALTERNATIVE(r,s,e);
}
```

Figure 8. A didactic example illustrating some operations from DMA

7. Conclusion

Map algebra is a powerful tool to implement cartographic modeling. Several extensions to the original map algebra are available in literature. However, neither the original map algebra nor its different extensions are able to support multicriteria aspects of spatial problems. This is due essentially to the absence of convenient operators permitting to support multicriteria modeling.
In this paper, we have proposed a new algebra, called DMA, especially devoted to multicriteria spatial modeling. As other map algebras, the proposed DMA has several merits: (i) it is a rigorous mathematical modeling framework, (ii) well adapted to object-oriented implementation, which is a natural way to deal with geographic objects and phenomena; and (iii) its independent from the way data are internally stored. Additionally, our DMA works for both vector and raster data representation (most of proposed algebra works only on raster data representation). More importantly, DMA is a powerful environment for multicriteria spatial modeling. In addition, using DMA needs a limited knowledge of multicriteria modeling.
Currently, the implementation of DMA is ongoing. DMA is being implemented on ArcGIS 9.1 of ESRI using Visual C++.
A part some common operators, the major part of DMA operators are especially devoted to outranking-based multicriteria modeling methods. The addition of other operators devoted to utility function-based multicriteria modeling methods is under study. In future time, we envisage the development of visual and script-based versions of DMA.

Acknowledgement

The first author would like to thank Rim Kalai for the constructive discussions concerning an earlier version of this paper.

References

M.P. Armstrong and P.J. Densham, "*Toward a network map algebra*", Proceedings Thirteenth International Symposium on Automated Cartography (Auto-Carto13), pp. 1 – 10, 1997.

V. Belton and T.J. Stewart, "*Multiple criteria decision analysis: An integrated approach*", Clawer Academic Publishers, Boston, 2002.

J.K. Berry, "*Cartographic modeling: The analytic capabilities of GIS*", In: Geographic Information Systems and Environmental Modeling, M.F. Goodchild, B.O. Parks and L.T. Steyaert (editors), Oxford University Press, Oxford, England, 1993.

J.K. Berry, "*A mathematical structure for analyzing maps*", **Environmental Management**, Vol. 11, No. 3, pp. 317-325, 1986.

P.H.A. Burrough and R.A. McDonnell, "*Principles of geographical information systems*", Oxford University Press, N.Y., 1998.

D.R. Caldwell, "*Extending map algebra with flag operators*", Proccedings of The Fifth International Conference on GeoComputation. The University of Greenwich's School of Earth and Environmental Sciences, pp. 23-25, August 2000.

S. Carver, "*Integrating multi-criteria evaluation with geographical information systems*", **International Journal of Geographical Information Systems**, Vol. 5, No. 3, pp. 321-339, 1991.

S. Chakhar, V. Mousseau, C. Pusceddu and B. Roy, "*Decision map for spatial decision making and urban planning*", The 9th International Computers in Urban Planning and Urban Management Conference (CUPUM'05), London, UK, 29 July-1 August, 2005.

S. Chakhar and J.-M. Martel, "*Towards a spatial decision support system: Multi-criteria evaluation functions inside geographical information system*", **Annales du LAMSADE**, No. 2, pp. 97-123, University of Paris Dauphine, Paris, France , 2004.

S. Chakhar and V. Mousseau, "*Towards a typology of spatial decision problems*", **Annales du LAMSADE**, No. 2, pp. 125-154, University of Paris Dauphine, Paris, France, 2004.

S. Chakhar and J.-M. Martel, "*Enhancing geographical information systems capabilities with multi-criteria evaluation functions*", **Journal of Geographic Information and Decision Analysis**, Vol. 7, No. 2, pp. 47-71, 2003.

J.G. Corripio, "*Vectorial algebra algorithms for calculating terrain parameters from DEMs and solar radiation modelling in mountainous terrain*", **International Journal of Geographical Information Sciences**, Vol. 17, No. 1, pp. 1-23, 2003.

M.J. Egenhofer and H.T. Bruns, "*Visual map algebra: A direct manipulation user interface for GIS*", Proceedings of the Third IFIP 2.6 Working Conference on Visual Database Systems, Lausanne, Switzerland, S. Spaccapietra and R. Jain, (editors), pp. 235-253, Chapman & Hall, 1995.

G.B. Hall and F. Wang and Subaryono, "*Comparison of boolean and fuzzy classification methods in land suitability analysis by using geographical information systems*", **Environment and Planning A**, Vol. 24, pp. 497-516, 1992.

C.L. Hwang and K.L. Yoon, "*Multiple attribute decision making: Methods and applications*", Springer-Verlag, N.Y., 1981.

P. Jankowski, "*Integrating geographical information systems and multiple criteria decision-making methods*", **International Journal of Geographical Information Systems**, Vol. 9, No. 3, pp. 251-273, 1995.

F. Joerin and A. Musy, "*Land management with GIS and multicriteria analysis*", **International Transactions on Operational Research**, Vol. 7, pp. 67-87, 2000.

F. Joerin and A. Musy, "*Using GIS and outranking multicriteria analysis for land-use suitability assessment*", **International Journal of Geographical Information Science**, Vol. 15, pp. 153-174, 2001.

A. Laaribi, "*SIG et analyse multicière*", Hermès Sciences Publications, Paris, France, 2000.

F.-T. Lin, "*Many sorted algebraic data models for GIS*", **International Journal of Geographical Information Science**, Vol. 12, No. 8, pp. 765-788, 1998.

J. Malczewski, "*GIS and mutlicriteria decision analysis*", John Wiley & Sons, N.Y., 1999.
J. Mennis, R. Viger and C.D. Tomlin, "*Cubic map algebra functions for spatio-temporal analysis*", **Cartography and Geographical Information Science**, Vol. 32, No. 1, pp. 17-32, 2005.

N. Martin, B. St-Onge and J.-P. Waaub, "*An integrated decision aid system for the development of Saint-Charles River's alluvial plain, Quèbec, Canada*", **International Journal of Environmental Pollution**, Vol. 12, pp. 264-279,2000.

J. Mennis, J. Leong and R. Khanna, "*Multidimensional map algebra*", Proceedings The 8th International Conference on Geocomputation, Ann Arbor, MI, August 1-3, 2005.

S. Menon, P. Gao and C. Zhan, "*GRID: A data model and functional map algebra for raster geo-processing*", Proceedings of GIS/LIS'92, American Society for Photogrammetry and Remote Sensing, Bethesda, 1992.

V. Mousseau, "*A general framework for constructive learning preference elicitation in multiple criteria decision aid*", **Cahiers du LAMSADE**, No. 223, University of Paris Dauphine, Paris, France, 2005

V. Mousseau and R. Slowinski, "*Inferring an ELECTRE TRI model from assignment examples*", **Journal of Global Optimization**, Vol. 12, pp. 157-174, 1998.

D. Pullar, "*SimuMap: A computational system for spatial modelling*", **Environmental Modelling & Software**, Vol. 19, pp. 235-243, 2004.

B. Roy, "*Multicriteria methodology for decision aiding*", Dordrecht: Kluwer Academic Publishers.

M. Takeyama and H. Couclelis, "*Map dynamics: integrating cellular automata and GIS*", **International Journal of Geographical Information Science**, Vol. 11, No. 1, pp. 73-91, 1997.

C.D. Tomlin, "*Geographic information systems and cartographic modeling"*, Prentice Hall, Englewood Cliffs, New Jersey, USA, 1990.

C.D. Tomlin, "*A map algebra*", Proceedings of the 1983 Harvard Computer Graphics Conference, Vol. 2, pp. 127-150, Cambridge , Massachusetts, Harvard Laboratory for Graphics and Spatial Analysis, 1983.

C.D. Tomlin, "Cartographic modelling, "*Geographical information systems: Principles and Applications*", D.J. Maguire, M.F. Goodchild and D.W. Rhind, (editors) , pp. 361-374, Longman Scientific & Technical , Harlow, Essex, UK, 1991.

Ph. Vincke, " *Multicriteria decision-aid*", Chichester: John Wiley and Sons.

F. Wang , "*The use of artificial neural networks in geographical information system for agricultural land-suitability assessment*", **Environment and Planning A, Vol. 26,** 265-284, 1994.

W. Yu, ELECTRE TRI: Aspects méthodologiques et guide d'utilisation, **Document du LAMSADE**, No. 74, University of Paris Dauphine, Paris, France, 1992.

S.H. Zanakis, A. Solomon, N. Wishart, and S. Dublish, "*Multi-attribute decision making: A simulation comparison of select methods*", **European Journal of Operation Research**, Vol. 107, No. 3, pp. 507-529, 1998.

Appendix A: DMA data types specifications

In this Appendix we provide the description of DMA data types. To make the paper short, the axiom part is provided for only some operators.

```
Type: gPoint
set: gPoint, real, boolean

Syntax:
 MAKE       real x real → gPoint
 ISEQUAL    gPoint x gPoint → boolean
 DISTANCE   gPoint x gPoint → real
 X          gpoint → real
 Y          gpoint → real

Axioms:
  i,j: real; p, q: gPoint

   X(MAKE (i, j))
       = i
   Y(MAKE (i,j))
      = j
```

Figure A.1. Formal specification of `gPoint` data type

```
Type: gLine
set: gPoint, gLine, real

Syntax:
 MAKE     gPoint x gPoint → gPoint
 START    gLine →gPoint
 END      gLine →gPoint
 LENGTH   gLine →real

Axioms:
  p, q: gPoint

  START (MAKE (p, q))
    = p
  END (MAKE (p, q))
    = q
```

Figure A.2. Formal specification of `gLine` data type

```
Type: gPolygon
set: gPoint, gLine, gPolygon, real, boolean

Syntax:
 MAKE         gPoint x … x gPoint → gPolygon
 AREA         gPolygon  → real
 CENTROID     gPolygon  →  gPoint
 CONTAINS     gPolygon  x gPoint → boolean
 INTERSECTS   gPolygon  x gLine→ boolean
Axioms:
  p1,…, pn, x, r: gPoint; l:gLine; v: real

  AREA(MAKE(p1,…,pn))
     =  _area
  CENTROID(MAKE(p1,…,pn))
     = _gPoint
  CONTAINS(MAKE(p1,…,pn), r)
   =if (∀(p) in (p1,…, pn) X(r) ≤ X(p) and Y(r) ≤ Y(p)) then T

  INTERSECTS(MAKE (p1,…,pn),l)
    = CONTAINS( MAKE(p1,…,pn), SART(l))      or
        CONTAINS( MAKE(p1,…,pn), END(l))
```

Figure A.3. Formal specification of `gPolygon` data type

```
Type: map-layer
set: map-layer, gPoint, gLine, gPolygon, rSystem, mScale, real

Syntax:
MAKE      name → map-layer
PUT       map-layer x real x real → gPoint
PUT       map-layer x gPoint x gPoint → gLine
PUT       map-layer x gPoint x … x gPoint → gPolygon
INTERSECT map-layer x … x map-layer → map-layer

Axioms:
m: map-layer; p,p1,p2, q1,…,qn:gPoint; l: gLine; y: gPolygon; v1, v2: real

PUT(m,v1,v2)
  = p.MAKE(v1,v2)


PUT(m,p1,p2)
  = l.MAKE(p1,p2)

PUT(m,v1,v2)
  = p.MAKE(v1,v2)

PUT(m,q1,…,qn)
  = y.MAKE(q1,…,qn)
```

Figure A.4. Formal specification of `map-layer` data type

```
Type: aOperator
set: aOperator, value

Syntax:
CREATE DEFERRED → aOperator
COMBINE    value x value x···x value → value
```

Figure A.5. Formal specification of `aOperator` data type

```
Type: aProcedure
set: aProcedure, value

Syntax:
CREATE DEFERRED → aProcedure
COMBINE    value x value x···x value → value
```

Figure A.6. Formal specification of `aProcdure` data type

```
Type: nProcedure
set: nProcedure, value

Syntax:
CREATE DEFERRED → nProcedure
COMBINE    value x value x···x value → value
```

Figure A.7. Formal specification of nProcedure data type

```
Type: dTable
set: dTable, bRelation, value

Syntax:
CREATE DEFERRED → dTable
ACTION value x value x value → bRelation
```

Figure A.8. Formal specification of `dTable` data type

```
Type: aList
set: aList, value, aPosition

Syntax:
aList
INSERT   aList x value x aPosition → aList
LOCALIZE value x aList → aPosition
GET      aPosition x aList → value
FIRST    aList → aPosition
RAZ      aList  → aList
REMOVE   aPosition x aList → aList
NEXT     aPosition x aList → aPosition
BEFORE   aPosition x aList → aPosition
```

Figure A.9. Formal specification of aList data type

```
Type: pStructure
set: pStructure, aAlter, sAlter, aCriterion, bRelation, value

syntax:
P  aAlter x aAlter x aCriterion → bRelation
I  aAlter x aAlter x aCriterion → bRelation
R  aAlter x aAlter x aCriterion → bRelation
S  aAlter x aAlter x aCriterion → bRelation

Axioms:
a,b: aAlter; g: aCriterion; p: pStructure, s: sAlter

P(a,b,g)
  = if SCORE(a,g) > SCORE(b,g) + g.pThreshold then aPb else ¬(aPb)

Q(a,b,g)
  = if [SCORE(b,g) + g.pThreshold ≥ SCORE(a,g)] ∧
       [SCORE(a,g) > SCORE(b,g) + g.qThreshold]  then aQb else ¬(aQb)

I(a,b,g)
  = if [SCORE(b,g) + g.qThreshold  ≥  SCORE(a,g)] ∧
       [SCORE(a,g) + g.qThreshold  ≥ SCORE(b,g)] then aIb else ¬(aIb)

S(a,b,g)
  = if (P(a,b,g) ∨ I(a,b,g)) then aSb else ¬(aSb)
```

Figure .10. Formal specification of `pStructure` data type

Appendix B: Overview of ELECTRE TRI method

We give here a very brief overview of the ELECTRE TRI method. First, we mention that this overview is reproduced from Mousseau (2005). A complete description can be found in Roy and Bouyssou (1993). ELECTRE TRI is a multicriteria sorting method used to assign alternatives to predefined ordered categories. The assignment of an alternative a results from the comparison of a with the profiles defining the limits of the categories. Let A denote the set of alternatives to be assigned and let $K=\{1,2,\ldots,n\}$ be the set of indices of the alternatives. Let F denote the set of the indices of the criteria $g_1,g_2,\ldots,g_m$ ($F=\{1,2,\ldots,m\}$), k_j the importance coefficient of the criterion g_j, B the set of indices of the profiles defining $p+1$ categories ($B=\{1,2,\ldots,p\}$), b_h being the upper limit of category C_h and the lower limit of category C_{h+1}, $h=1,2,\ldots,p$. Each profile b_h is characterized by its performances $g_j(b_h)$ and its thresholds $p_j(b_h)$ (preference thresholds), $q_j(b_h)$ (indifference thresholds) and $v_j(b_h)$ (veto thresholds). In what follows, we will assume, without any loss of generality, that preferences increase with the value on each criterion and that $\sum_{j\in F} k_j =1$.
Further on, we use $a\rightarrow C_h$ to denote that the alternative a is assigned to the category C_h. ELECTRE TRI builds a fuzzy outranking relation S whose meaning is "at least as good as". Preferences on each criterion are defined through pseudo-criteria (see Roy and Vincke (1984) for details on this double thresholds preference representation). The threshold $q_j(b_h)$ represents the largest difference $g_j(a)-g_j(b_h)$ preserving an indifference between a and b_h in respect to criterion g_j. The threshold $p_j(b_h)$ represents the smallest difference $g_j(a)-g_j(b_h)$ compatible with a preference in favor of a in respect to criterion g_j. Thus, the limits of categories are defined in terms of profiles b_h, $h\in B$; each one is delimited by two imprecision zones (See Figure B.1).

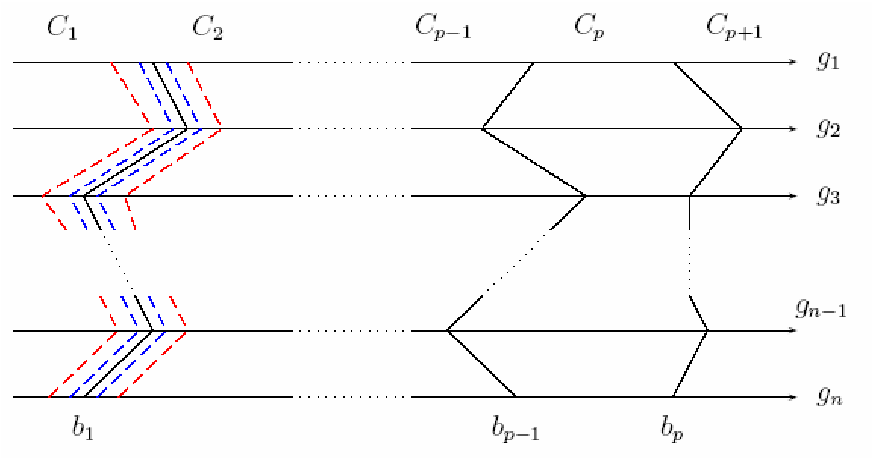


Figure B.1. Defining of categories in terms of profiles

To validate the proposition aSb_h (b_hSa, resp.), two conditions must hold:

i) *Concordance*: An outranking aSb_h (b_hSa, resp.) is accepted only if a "sufficient" majority of criteria are in favor of this proposition.

ii) *Non-discordance*: When the concordance holds, none of the minority of criteria shows an "important" opposition to aSb_h (b_hSa, resp.).

Beside the intra-criterion preferential information, represented by the indifference and preference thresholds, $q_j(b_h)$ and $p_j(b_h)$, the construction of S also makes use of two types of inter-criterion preferential information:

i) the set of weight-importance coefficients $(\{k_j,\ j\in F\})$ is used in the concordance test when computing the relative importance of the coalitions of criteria being in favor of the assertion aSb_h (b_hSa, resp.)

ii) the set of veto thresholds ($\{v_j(b_h),\ j\in F,\ h\in B\}$) is used in the discordance test; $v_j(b_h)$ represents the smallest difference $g_j(b_h)-g_j(a)$ incompatible with the assertion aSb_h (b_hSa, resp.).

As the assignment of alternatives to categories does not result directly from the relation S, an exploitation phase is necessary; it requires the relation S to be "defuzzyfied" using a so-called λ-cut: the assertion aSb_h (b_hSa, resp.) is considered to be valid if the credibility index of the fuzzy outranking relation is greater than a "cutting level" λ such that $\lambda \in [0.5,1]$. This λ-cut determines the preference situation between a and b_h.
ELECTRE TRI constructs an indices $\sigma(a,b_h) \in [0,1]$ ($\sigma(a,b_h)$, resp.) representing the credibility of the proposition aSb_h (b_hSa, respectively), $\forall\ a \in A$, $\forall\ h \in B$. The proposition aSb_h (b_hSa, resp.) holds if $\sigma(a,b_h) \geq \lambda$ ($\sigma(b,a) \geq \lambda$, resp.). The indices $\sigma(a,bh)$ is defined as follows (the values of $\sigma(a,b_h)$ is defined in similar way):

1. Compute partial concordance indices $S_j(a,b_h)$, $\forall j \in F$:

$$S_j(a,b_h) = \begin{cases} 0, & \text{if } g_j(b_h) - g_j(a) \geq p_j(b_h) \\ 1, & \text{if } g_j(b_h) - g_j(a) \leq q_j(b_h) \\ [p_j(b_h)-g_j(b_h)+g_j(a)]/[p_j(b_h)-q_j(b_h)], & \text{otherwise} \end{cases} \quad (1)$$

2. Compute global concordance indice $S(a,b_h)$:

$$S(a,b_h) = \sum_{j \in F} k_j\ S_j(a,b_h) \quad (2)$$

3. Compute partial discordance indices $ND_j(a,b_h)$, $\forall j \in F$:

$$ND_j(a,b_h) = \begin{cases} 0, & \text{if } g_j(a) \leq g_j(b_h) + p_j(b_h) \\ 1, & \text{if } g_j(a) > g_j(b_h) + v_j(b_h) \\ [v_j(b_h)-g_j(a)+g_j(b_h)]/[v_j(b_h)-p_j(b_h)], & \text{otherwise} \end{cases} \quad (3)$$

4. Compute the global discordance indice $ND(a,b_h)$:

$$ND(a,b_h) = \prod_{j \in F'} ([1-ND_j(a,b_h)]/[1-S(a,b_h)]) \quad (4)$$

   With $F'=\{j \in F : ND_j(a,b_h) > S(a,b_h)\}$
5. Compute credibility indice $\sigma(a,b_h)$:

$$\sigma(a,b_h)) = S(a,b_h) * ND(a,b_h) \quad (5)$$

The values of $\sigma(a,b_h)$, $\sigma(b_h,a)$ and λ determine the situation of preference concerning a and b_h:

- $\sigma(a,b_h) \geq \lambda$ and $\sigma(b_h,a) \geq \lambda \Rightarrow aSb_h$ and $b_hSa \Rightarrow aIb_h$
- $\sigma(a,b_h) \geq \lambda$ and $\sigma(b_h,a) < \lambda \Rightarrow aSb_h$ and $\neg(b_hSa) \Rightarrow aPb_h$
- $\sigma(a,b_h) < \lambda$ and $\sigma(b_h,a) \geq \lambda \Rightarrow \neg(aSb_h)$ and $b_hSa \Rightarrow b_hPa$
- $\sigma(a,b_h) < \lambda$ and $\sigma(b_h,a) < \lambda \Rightarrow \neg(aSb_h)$ and $\neg(b_hSa) \Rightarrow aRb_h$

Two assignment procedures are available: optimistic and pessimistic. Their role being to analyze the way in which an alternative a compares to the profiles so as to determine the category to which a should be assigned. The result of these two assignment procedures differs when the alternative a is incomparable with at least one profile b_h

i) Pessimistic procedure:
  a) Compare a successively to b_i; $i=p,p-1,\ldots 0$.
  b) Let b_h the first profile such that aSb_h, then assign a to category C_{h+1} ($a \rightarrow C_{h+1}$).

ii) Optimistic procedure:

  a) Compare a successively to b_i; $i=1,2,\ldots p$.
  b) Let b_h the first profile such that b_hSa, then assign a to category C_h ($a \rightarrow C_h$)