Available online at www.sciencedirect.com
ScienceDirect
Computers, Environment and Urban Systems
31 (2007) 572–596
Computers, Environment and Urban Systems
www.elsevier.com/locate/compenvurbsys

An algebra for multicriteria spatial modeling

Salem Chakhar *, Vincent Mousseau

LAMSADE, University of Paris Dauphine, Place du Maréchal de Lattre de Tassigny, 75775 Paris Cedex 16, France

Accepted in revised form 31 July 2007

Abstract

The objective of this paper is to propose a new algebra, called decision map algebra (DMA), especially devoted to multicriteria spatial modeling. DMA is a generic and context-independent modeling language inspired from map algebra and other similar languages. It supports the conventional geometric objects and cartographic maps and adds a rich set of new operands representing different spatial multicriteria concepts. Moreover, DMA supports different conventional map algebra-like operations and adds several new ones corresponding to different spatial multicriteria evaluation functions. The proposed algebra is the first step towards the development of a generic spatial multicriteria modeling language.

Keywords: GIS; Spatial multicriteria modeling; Map algebra; Formal specification

1. Introduction

The geographical information system (GIS) is a powerful tool to collect, store, manage and analyze spatial data. However, GIS has several limitations in spatial decision making which are due in great part to the lack of sophisticated analytical and spatial modeling tools. The solution generally adopted to enhance the GIS in spatial decision making consists in coupling it with other computing and operational research tools. Among these tools, the multicriteria analysis is naturally the most appropriate one. Multicriteria analysis (MCA) is broadly defined as "a decision-aid and a mathematical tool allowing the comparison of different alternatives or scenarios according to many criteria, often conflict-

* Corresponding author. Tel.: +33 1 44 05 41 20; fax: +33 1 44 05 41 11.
E-mail addresses: salem.chakhar@dauphine.fr (S. Chakhar), mousseau@lamsade.dauphine.fr (V. Mousseau).

0198-9715/$ - see front matter
doi:10.1016/j.compenvurbsys.2007.08.007

ing, in order to guide the decision maker towards a judicious choice" (Roy, 1996). The major argument in favor of GIS-MCA integration is that spatial decision problems are inherently of multicriteria nature (Carver, 1991; Chakhar & Martel, 2003; Laaribi, Chevallier, & Martel, 1996; Malczewski, 1999).

A great deal of papers concerning GIS-MCA integration has been published since 1990 (see Malczewski (2006) for a recent survey of the literature). Unfortunately, the lack of adequate capabilities for supporting an effective multicriteria spatial modeling prevent these works from going beyond the academic contexts. Indeed, in most of the previous works attention is essentially oriented towards the physical integration of the two tools. Currently, research on GIS and MCA integration is matured enough and we think that more attention should be addressed to enhance GIS-based multicriteria analysis systems by adding multicriteria spatial modeling capabilities.

There is a large body of literature for adding spatial modeling capabilities in GIS. One possible solution is to develop generic, context-independent and script-like languages such as map algebra or other similar tools. Map algebra, introduced in early 1980s by Tomlin (Tomlin, 1983) and then extended and refined in a book (Tomlin, 1990), uses single-factor map layers as operands on which spatial operators are applied to generate new map layers as the result. Map algebra has played a useful function in the development of raster GIS, but it exists somewhat in isolation from other more general approaches in the broader literature such as image algebra developed by Ritter (Ritter, Wilson, & Davidson, 1990), the work of Serra on mathematical morphology (Serra, 1982), and the work of van Deursen and the PCRaster group on dynamic modeling (Karssenberg, Burrough, Sluiter, & de Jong, 2001; Wesseling, Karssenberg, van Deursen, & Burrough, 1996).

All these tools have played an important role in the development of spatial modeling capabilities in the GIS. However, all of them and their different extensions are not able to support multicriteria spatial modeling in GIS. This is due to the absence of convenient operators permitting to implement the different multicriteria evaluation functions. The objective of this paper is to propose a new algebra, called decision map algebra (or DMA), especially devoted to multicriteria spatial modeling. DMA contains, in addition to conventional geometric objects and cartographic maps, a rich set of new operands representing different spatial multicriteria concepts. Moreover, DMA supports different conventional map algebra-like operations and adds several new ones corresponding to spatial multicriteria evaluation functions.

The development of DMA has been influenced by several proposals. The Map Analysis Package (MAP) of Tomlin (1983, 1990) is the first comprehensive collection of analytical and spatial operations on the basis of regular tessellations. It has been incorporated in a large number of commercial GIS and remote sensing image processing packages, and it has been extended in area ranging from cellular automata (Takeyama & Couclelis, 1997), to environmental modeling (Pullar, 2001), to topographic analysis (Caldwell, 2000), to spatio-temporal analysis (Mennis, Viger, & Tomlin, 2005). Tomlin's map algebra has influenced the development of DMA at least in two aspects. First, a large number of its operators are required to define the ones of DMA. Second, the notion of "procedure" introduced by Tomlin is very important in DMA since most of multicriteria operations are complex ones needing the use of procedures for their formalizing.

An important limitation of Tomlin's map algebra is related to the fact that it describes map overlay operations textually, without applying the mathematical rigor necessary to analyze the behavior of the operations. Among the first rigorous formalization of map algebra is the one provided by Chan and White (1987), where the authors describe formally MAP operations in the C++ programming language. This formalization permits especially to avoid the ambiguous interpretations of the semantic of different operations. In particular, the authors associate two implicit axioms to each operation. The first one states that a function returns "error" if any of its arguments does not belong to the domain of the function. The second axiom states that a function returns "error" if any of its arguments is itself "error". Although these axioms are not included explicitly in the specification of DMA, they still fully apply to it with the same semantics. One limitation of Chan and White (1987)'s work is the lack of the definitions of the corresponding observe operations (i.e. operations to observe some of the properties of the datatype) so that no axioms about the behavior of the operations can be formulated (Dorenbeck & Egenhofer, 1991).

Another proposal that have inspired DMA is the one of Dorenbeck and Egenhofer (1991). A major finding of this paper is the use of decision table concept, which is adopted in our DMA. First, the authors propose a generic definition of the overlay operation basing on a "generalized value type" permitting to characterize all non-spatial properties of each element (raster) of the map-layer. The overlay operation is then defined in similar way to conventional arithmetic operations. Then, they propose and discuss two strategies for the overly operation implying more than two map layers. Both strategies are based on the use of decision table concept. Compared to the above cited papers, this one lacks geometric operations on cells, e.g., those which exploit the neighborhood relationship between cells.

A second important problem of Tomlin's map algebra is the strong link between geographical datatypes and data structures. To avoid this problem, Câmara, De Freitas, and Cordeiro (1994) propose the use of a general data model, which formally defines the various types of geographical data and integrates them in a unified environment. Basing on this data model, the authors introduce a field-oriented algebra where a map layer is defined as function f between a set of localization L and a set of values V. According to the nature of set V, three types of maps are defined: thematic maps, digital terrain models and images. The major addition of this paper in respect to previous ones is the distinction between digital terrain models and images. Although the data format for digital terrain models and images is the same, this distinction is practically useful since it permits to apply/extend a large number of image algebra operators to spatial modeling as in Su, Li, Lodwick, and Müller (1997).

The major shortcoming of above cited works is the grid cell data structure considered in all of them. More formal approaches are based on the representation of spatial data in terms of points, lines and areas are very useful in a large number of spatial problems. Lin (1998), for instance, uses the *many stored algebra* theory to define a family of models organized into five groups: "point", "line", "polygon", "network" and "raster". In addition to the fact that it permits to handle multiple datatypes, the main addition of this work is the introduction of "network" datatype which is useful for a large of body of applications such as petroleum or public (e.g., services electricity and heat) distribution problems.

Recent works are essentially devoted to develop script-like programming languages (e.g., MapScript of Pullar (2001)), to support spatial–temporal analysis (e.g., Mennis et al., 2005), visual spatial modeling (e.g., Murray, Breslin, Ormsby, & Miller, 2000) and to the development of web-based map algebra-like frameworks (e.g., Grunberg et al., 2004). Although the relatively large number of algebra that have been proposed in the literature, to our knowledge there is not any one devoted to multicriteria spatial modeling. We think that the developing of DMA permits to facilitate multicriteria spatial modeling and enhance GIS-based multicriteria based spatial decision making (Malczewski, 2006).

This paper goes as follows. Section 2 briefly introduces multicriteria spatial modeling. Section 3 introduces the concept of decision map, which is a basic ingredient of our algebra. Section 4 presents some basic primitives required for the specification of the proposed algebra. Section 5 details the formal specification of this algebra. Section 6 concludes the paper.

2. Multicriteria spatial modeling

2.1. Multicriteria analysis

Multicriteria analysis is a family of OR/MS tools that have experienced very successful applications in different domains since the 1960s. Different multicriteria analysis methods are available in the literature (see, e.g., Belton & Stewart, 2002; Figueira, Greco, & Ehrgott, 2005a; Roy, 1996). Multicriteria methods are commonly categorized as *discrete* or *continuous*, depending on the domain of alternatives. 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 this paper we focus on the first category. Fig. 1 gives the general schema of discrete multicriteria methods. It illustrates how the different elements of the decision problem are linked to each other.

The first requirement of nearly all discrete methods is a *performance table* containing the evaluations or *criteria scores* of a set of alternatives on the basis of a set of criteria. The *alternatives* are decision objects from which the decision maker should choose one or several ones to implement. Let $A = \{a_1, a_2, \ldots, a_n\}$ denotes a set of n alternatives. The *evaluation criteria* are factors on which alternatives are evaluated and compared. Formally, a criterion is a function g, defined on A, taking its values in an ordered set, and representing the decision maker's preferences according to some points of view. The

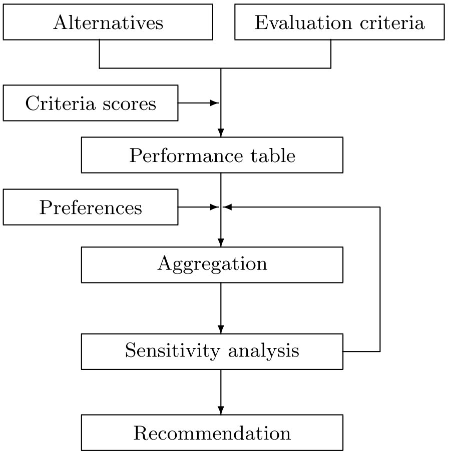

Fig. 1. General schema of multicriteria discrete methods.

evaluation of an alternative $a \in A$ according to criterion g is written $g(a)$. Let $G = \{g_1, g_2, \ldots, g_m\}$ a set of m evaluation criteria and $F = \{1, 2, \ldots, m\}$ the set of criteria indices.

The next step consists in the aggregation of the different criteria scores using a specific *decision rule* (or *aggregation procedure*) and taking into account the decision maker's *preferences*, generally represented in terms of weights that are assigned to different criteria. The term "aggregation" is used both in GIS and MCA literatures. In GIS community, this term implies spatial aggregation (e.g., aggregating spatial units). It is a combination function that "aggregate" different map layers. In MCA community, this term implies functional aggregation of partial evaluations of alternatives (using, for e.g., weighted sum technique) into a unique and global evaluation. The aggregation of criteria scores permits the decision maker to make comparison between the different alternatives on the basis of these scores. Decision rules are somehow the identities of the multicriteria methods. More details on decision rules within discrete methods are given in the next sub-section.

The uncertainty and the fuzziness generally associated with any decision situation require a *sensitivity analysis* enabling the decision maker(s) 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 final recommendation in multicriteria analysis may take different forms, according to the manner in which a problem is stated. Roy (1996) identifies four types of results corresponding to four ways for stating a problem: (i) *choice*: selecting a restricted set of alternatives, (ii) *sorting*: assigning alternatives to a set of predefined categories, (iii) *ranking*: classifying alternatives from best to worst with eventually equal positions, and (iv) *description*: describing the alternatives and their follow-up results. The currently available multicriteria methods permit to deal with the choice, sorting and ranking cases only.

2.2. Multicriteria decision rules

There are two great families of decision rules within multicriteria discrete methods: (i) *utility function-based decision rules*, and (ii) *outranking relation-based decision rules*. The basic principle of the first family is that the decision maker looks to maximize an utility function $U(a) = U(g_1(a), g_2(a), \ldots, g_m(a))$ aggregating the partial evaluations of each alternative into a global one. The most simple and also most used utility function is the additive form: $U(a) = \sum_j u_j(g_j(a))$; where u_j $(j = 1, \ldots, m)$ are the partial utility functions. Within this form, the preference P and indifference I binary relations are defined for two alternatives a and b as follows:

$$aPb \Longleftrightarrow U(a) > U(b) \quad \text{and} \quad aIb \Longleftrightarrow U(a) = U(b)$$

Decision rules of the first family differ mainly on the way the partial utility functions u_j $(j = 1, \ldots, m)$ are constructed and on the aggregation form used.

With the second family, criteria are aggregated into a 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 (Figueira, Mousseau, & Roy, 2005b). To construct the outranking relation S, we compute for each pair of alternatives (a, b) a *concordance index* $C(a, b) \in [0, 1]$ measuring the power of criteria that are in favor of the assertion aSb and a *discordance index* $ND(a, b) \in [0, 1]$ measuring the power of criteria that oppose to aSb. Then, the relation S is defined as follows:

$$C(a,b) \geqslant c \quad \text{and} \quad ND(a,b) \leqslant d$$

where c and d are the *concordance* and *discordance* thresholds, respectively. Often an exploitation phase is needed to "extract", form S, information on how alternatives compare to each other. At this phase, the concordance $C(a,b)$ and discordance $ND(a,b)$ indices are used to construct an index $\sigma(a,b) \in [0,1]$ representing the *credibility* of the proposition aSb, $\forall(a,b) \in A\text{x } A$. The proposition aSb holds if $\sigma(a,b)$ is greater or equal to a given *cutting level*, $\lambda \in [0.5,1]$.

Outranking relation-based decision rules differ mainly on the way the outranking relation S is constructed and on the manner the credibility index is computed.

2.3. Multicriteria spatial decision making

Multicriteria spatial decision making refers to the application of multicriteria analysis in spatial context where alternatives, criteria and other elements of the decision problem have an explicit spatial dimension. In the rest of this sub-section, we will focus only on the spatial dimension of decision alternatives and evaluation criteria since they constitute the basic ingredients of any multicriteria decision problem. More details concerning multicriteria spatial decision making are provided in Chakhar and Mousseau (2007).

A spatial decision alternative consists of at least two elements (Malczewski, 1999): *action* (what to do?) and *location* (where to do it?). The cardinality of the set of spatial decision alternatives is an important characteristic permitting to distinguish if this set is discrete or continuous. In the first case, the problem involves a discrete set of predefined decision alternatives. Spatial alternatives are then modeled through the basic spatial primitives, namely point, line, or polygon. 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 identifying a new industrial zone, potential alternatives are assimilated to a set of polygons representing different candidate zones. In many real-world applications, one may be called to represent decision alternatives by combining two or more atomic entities (Chakhar & Mousseau, 2004; Malczewski, 1999). In the rest of this paper we consider only simple and atomic decision alternatives.

The second case corresponds to a high or infinite number of decision alternatives, often defined in terms of constraints. For practical reasons, the set of decision alternatives in this case is often represented in a discrete form where each raster represents an alternative. Alternatives may also be constructed as a collection of rasters.

As for decision alternatives, spatial evaluation criteria are associated with geographical entities and relationships between entities and therefore can be represented in the form of maps (Malczewski, 1999). A *criterion map* is composed of a collection of spatial units; each of which is characterized with one value relative to the concept modeled. Mathematically, a criterion map $\mathbf{c}_j$ is the set $\{(s, g_j(s)) : s \in S_j\}$ where S_j is a set of spatial mapping units and g_j is a mono-valued criterion function defined as follows (E is a measurement scale):

$$\begin{aligned} g_j :& S_j \rightarrow E \\ & s \rightarrow g_j(s) \end{aligned}$$

To conclude this section, it is important to mention that multicriteria analysis has been used, since its emergence, to spatial decision problems without the use of GIS.

Nevertheless, multicriteria analysis alone is not able to take into account explicitly the spatial dimension of spatial decision problems. Koo and O'Connell (2006), for instance, remark that with conventional spatial multicriteria analysis (i.e. without the use of GIS), the performances of decision alternatives are either intrinsically a spatial or aggregated into a unique value. In other words, these approaches suppose that the study area is *spatially homogeneous*, which is unrealistic. Furthermore, most of multicriteria analysis softwares are of little utility in spatial decision making since they lack functionalities required (i) to collect, store, manage and analysis spatial data, and (ii) to represent the spatial dimension of decision problems. These two problems are well handled by GIS. Thus, GIS and multicriteria analysis are two complementary tools, each of which has advantages and some limitations in spatial decision making; their integration permits to avoid these limitations.

3. Concept of decision map

The concept of decision map was introduced in Chakhar, Mousseau, Pusceddu, and Roy (2005) and Chakhar (2006) to facilitate the use of outranking relation-based decision rules in the GIS. In fact, the major part of GIS and multicriteria analysis integration works use utility function-based decision rules. However, outranking relation-based decision rules are generally more appropriate to deal with ordinal aspects of spatial decision problems. The natural explication to this situation is that these decision rules have computational limitations in respect to the number of alternatives (Marinoni, 2006). Indeed, methods based on outranking relation, except those devoted to multicriteria sorting problems, require pairwise comparison across all alternatives.

One possible solution to facilitate the use of outranking relation-based decision rules in GIS is to reduce the number of decision alternatives. The idea generally used consists in subdividing the study area into a set of homogenous spatial units (or zones) which are then used as decision alternatives (Joerin, Thériault, & Musy, 2001; Marinoni, 2006) or as a basis for constructing these alternatives (Chakhar & Mousseau, 2006). This permits to reduce considerably the number of decision alternatives and make possible the use of outranking relation-based decision rules.

A *decision map* is a planar subdivision of the study area represented as a set of non-overlapping polygonal spatial units that are assigned, using a multicriteria sorting model Γ_w, into an ordered set of categories. More formally, a decision map $\mathbf{M}$ is defined as $\mathbf{M} = \{(u, \Gamma_w), u \in U\}$, where U is a set of homogenous spatial units and Γ_w is defined as follows:

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

where (i) $E : [e_1, e_2, \ldots, e_k]$: (with $e_i > e_j$, $\forall i > j$) is an ordinal (or cardinal) measurement scale defined such that e_i, for $i = 1, \ldots, k$, represents the evaluation of category C_i; (ii) $g_j(u)$ ($j = 1, \ldots, m$): is the performance of spatial unit u in respect to criterion g_j associated with criterion map $\mathbf{c}_j$; and (iii) w: is a set of preference parameters required to apply Γ.

To be useful, a decision map should be composed of non-overlapping spatial units. In addition, we generally suppose that the evaluations of adjacent (i.e. share at least one segment) spatial units are distinct, that is $\Gamma_w(u_i) \neq \Gamma_w(u_j)$, for all adjacent spatial units u_i and u_j.

The construction of a decision map needs the superposition of a set of criteria maps. The result is an intermediate map composed of a new set of spatial units that result from the intersection of the boundaries of the features in the criteria maps:

$$\begin{aligned} \oplus :& \mathbf{G} \rightarrow U \\ & \mathbf{c}_1 x \ldots x \mathbf{c}_m \rightarrow \vee_{i=1\ldots m} S_i \end{aligned}$$

where $\oplus$ is the union variant of GIS overlay operation that yields a new map by combining all involved features in the input criteria maps $\mathbf{G} = \{\mathbf{c}_1, \ldots, \mathbf{c}_m\}$; and $\vee$ returns the set of spatial units resulting from the intersection of the boundaries of the spatial objects contained in $S_1, S_2, \ldots, S_m$. Recall that S_j $(j = 1, \ldots, m)$ is the set of spatial units composing criterion map $\mathbf{c}_j$ (cf. Section 2.3). At the end of this step, each spatial unit u is characterized with a vector $\mathbf{g}(u) = (g_1(u), \ldots, g_m(u))$ of m evaluations.

The first version of $\mathbf{M}$ is then obtained by applying the multicriteria sorting model Γ_w to associate each spatial unit u in U to a category in E:

$$\begin{aligned} \mathbf{M} :& U \rightarrow E \\ & u \rightarrow \Gamma_w(u) \end{aligned}$$

The obtained decision map $\mathbf{M}$ may contain adjacent spatial units having the same evaluation. These spatial units need to be "merged" to obtain a final decision map.

Finally, we mention that criteria maps must represent the same territory and must be defined according to the same spatial scale and the same coordinate system. Note also that the overlay operation may generate silver polygons which should be eliminated. In addition, we mention that criteria maps must be polygonal ones. However, input datasets may be sample data points, raster maps, contour lines, etc. We need to transform all non-polygonal input datasets into polygonal ones. For example, a set of sample data points may be transformed into a TIN by computing the triangulation having vertices at data points or contour lines may be transformed into a polygonal map by a *constrained Delaunay triangulation* (see, e.g., de Floriani, Magillo, & Puppo, 1999; Chrisman, 2002).

4. Primitives and definitions

DMA contains several data types that may be classified into several groups: (i) geographic maps, (ii) geographic objects, (iii) multicriteria operands, (iv) collection of geographic maps/objects, and (v) several other operands required for the specification of DMA. The relationships among these data types are depicted in Fig. 2 using UML formalism.

The basic data type in DMA is map-layer. A `map-layer` is the most elementary data that contains the *map image* and other documentary items 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 raster `map-layer` is called gPixel. The map image of a vector `map-layer` is a collection of three types of geometric objects:

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

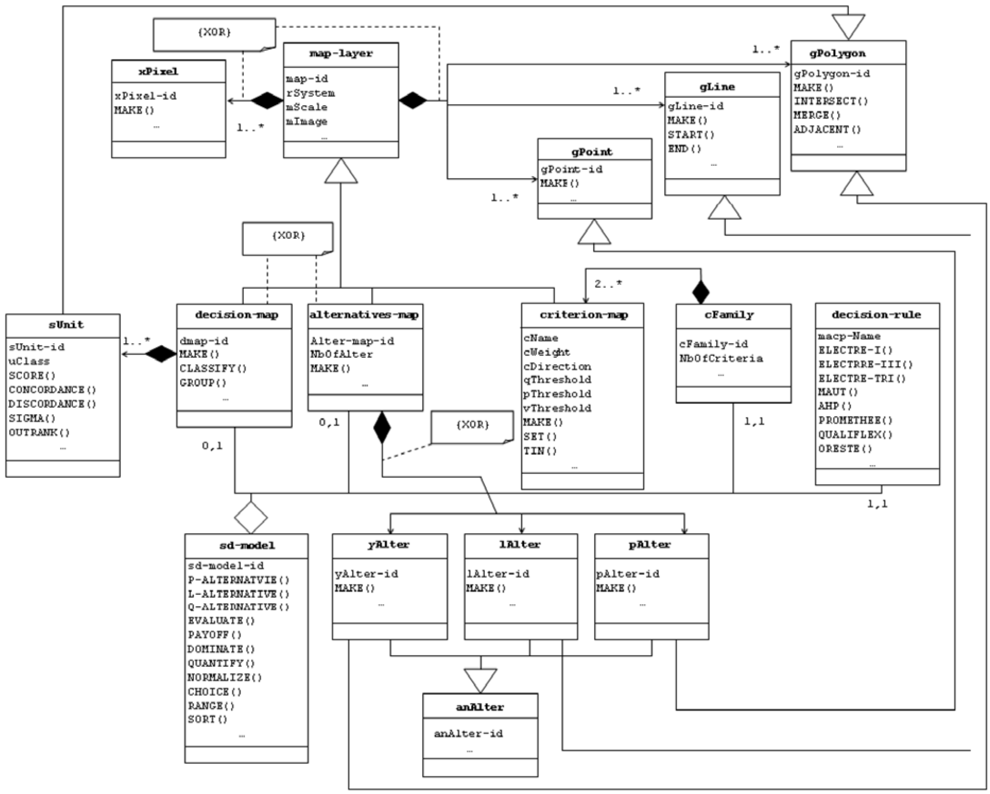

Fig. 2. DMA data types relationships.

A collection of `gPixel`, `gPoint`, `gLine` and `gPolygon` are denoted `xSet`, `pSet`, `lSet` and `ySet`, respectively. The `gPixel` elements of a raster `map-layer` are fully identified with their XY (or XYZ) 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: `decision-map`, `alternatives-map` and `criterion-map` (see Fig. 2). As defined earlier, a `decision-map` is a planar subdivision of the study area represented as a set of non-overlapping polygonal spatial units that are assigned, using a multicriteria sorting model Γ_w, into a set of ordered categories representing different evaluation levels (cf. Section 3). Each element of a `decision-map` is of type `sUnit` (for spatial unit). A collection of spatial units is denoted by `sSet`. We denote by `uClass` the category to which a spatial unit is assigned by Γ_w.

An `alternatives-map` is a special kind of `map-layer` which contains a collection of specific objects called *alternatives*. It is important to mention that alternatives are normally generated through `decision-maps`. The `alternatives-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 (cf. Section 2.3). A collection of `pAlter`, `lAlter` and `yAlter` are denoted `pAlters`, `lAlters`, and `yAlters`, respectively. The generic terms anAlter and `sAlt-`

ers denote an alternative and a set of alternatives, regardless to their nature. It is important to mention that for a given spatial decision problem, each alternatives-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 collection of evaluation criterion-maps is called cFamily (for *criteria family*). In multicriteria analysis, each criterion is often associated with a weight, a direction of optimization and several preference parameters. 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 comparison 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. For a pair of alternatives a and b, and a criterion g_j, the indifference threshold represents the largest difference $g_j(a) - g_j(b)$ preserving an indifference between a and b on criterion g_j. The preference threshold represents the smallest difference $g_j(a) - g_j(b)$ compatible with a preference in favor of a on criterion g_j.
- vThreshold: represents the smallest difference between the performance of two alternatives incompatible with the outranking assertion.

We introduce two additional complex data types to support multicriteria modeling. The first is the decision-rule data type devoted to implement the different decision rules (cf. Section 2.2). The second is sd-model (for spatial decision model) which is an aggregation of one decision-map (or an alternatives-map), at least two criterion-maps and a decision-rule. Several other data types are needed to formalize our algebra are not included in Fig. 2 but they will be introduced progressively hereafter.

5. Formal specification of DMA

To specify DMA, we adopt the algebraic specification method of Guttag (1977) (see also Guttag & 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 (Dorenbeck & Egenhofer, 1991; Guttag & Horning, 1978): (i) a *set of sorts* (or *operands*) 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 input sorts and the sort of the result; and (iii) a *set of axioms* (or *equations*) that describe the behavior of the operations. It is important to mention that details about the basic geometric data types such as point, line or polygon, and basic methods like get, set, etc., are not included in this paper.

5.1. *Specification of* sUnit *data type*

The formal specification of sUnit data type in given in Fig. 3. As shown in Fig. 2, sUnit data type is a subclass of gPolygon data type. This means that sUnit inherits

```
| Type:  sUnit
| set:  sUnit, gPolygon, aCriterion, cFamily, aProfile, real, value, boolean

syntax:
| ADJACENT      sUnit × sUnit → boolean
| ASSIGN        sUnit × aCriterion × value → sUnit
| SCORE         sUnit × aCriterion → real
| CONCORDANCE   sUnit × aProfile × cFamily → real
| DISCORDANCE   sUnit × aProfile × cFamily → real
| SIGMA         sUnit × aProfile × cFamily → real
| OUTRANK       sUnit × aProfile × cFamily × value → boolean

axioms:
| u:  sUnit; h:  aProfile; f:  cFamily; g:  aCriterion; λ:  value
|
| SIGMA(u, h, f)
| = CONCORDANCE(u, h, f) * DISCORDANCE(u, h, f)
|
| SIGMA(u, h, f)
| = CONCORDANCE(u, h, f)
|
| OUTRANK(u, h, f, λ)
| = if SIGMA(u, h, f) ≥ λ then uSh
```

Fig. 3. Formal specification of sUnit data type.

all the basic operations associated with gPolygon such as the operator ADJACENT which tests if two spatial units are adjacent or not. All the other operators are devoted to spatial multicriteria modeling.

The operator ASSIGN permits to set the performance of a sUnit in respect to a given criterion provided as parameter. The ASSIGN is a *hidden function* (Chan & White, 1987), i.e., it is not part of the algebra and serves its purpose in the specification only. The SCORE operator returns the performance of a spatial unit in respect to a given criterion. These two operators are quite straightforward and their specifications are not detailed in Fig. 3.

Data type sUnit is associated with a set of operators devoted to implement the multicriteria model Γ_w. The CONCORDANCE (resp. DISCORDANCE) operator computes a real value in $[0,1]$ corresponding to the global concordance (resp. discordance) index (cf. Section 2.2). The specifications of CONCORDANCE and DISCORDANCE are not included in Fig. 3 because they vary according to the multicriteria sorting model Γ_w.

The SIGMA operator permits to compute the *credibility index* as explained in Section 2.2. Fig. 3 contains two different specifications for SIGMA operator which permit to cover most of multicriteria sorting models. The operator OUTRANK permits to test if the spatial unit u outranks the profile h (profiles are limits of categories on the measurement scale E; see Section 3). This holds if and only if SIGMA (u,h,f) is greater or equal to the cutting level, λ.

5.2. Specification of decision-map data type

The decision-map data type corresponds to the concept of decision map introduced in Section 3. Fig. 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 operator CLASSIFY is the assignment procedure associated with the multicriteria sorting model Γ_w. It permits to assign each spatial unit in the decision-map to a predefined set of categories defined in terms of their profiles. CLASSIFY compares each spatial unit *u* to all the profiles starting from the best to the worst. The spatial unit *u* is assigned to the first category for which *u* outranks its lower limit.

The MERGE operator 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 operator aOperator, the evaluations associated with the initial 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 given in Appendix A. The aOperator must provide operation to combine a series of values. Since a large set of aggregation operators are available, the specification of its CREATE operator is *differed* (Meyer, 1988; Dorenbeck & Egenhofer, 1991), because it depends upon the particular aggregation operator.

The GROUP operator 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 category.

5.3. Specification of criterion-map data type

The specification of a criterion-map is shown in Fig. 5. The operator SET permits to set the preference parameters associated with criterion-map data type (cf. Section 5.1). The operator MAKE takes as input a predefined map algebra *procedure* (Tomlin, 1990), that is, a sequence of map algebra operations; and generates a criterion map. The following example permits to better illustrate the notion of procedure. The example looks to the construction of a criterion map to measure the "the level of underground pollution" in a given study area. First, we introduce the concept of decision table.

```
Type:  decision-map
set:  map-layer, criterion-map, sUnit, cFamily, aOperator, aProfile, sProfiles, value

syntax:
  MAKE       criterion-map × ··· × criterion-map → decision-map
  CLASSIFY   decision-map × cFamily × sProfiles → decision-map
  MERGE      decision-map × sUnit × ··· × sUnit × cFamily × aOperator → sUnit
  GROUP      decision-map × cFamily × aOperator → decision-map

axioms:
  d: decision-map; u,u1,u2: sUnit; c1,···,cm: criterion-map; g: aCriterion; f: cFamily;
  h: aProfile; b: sProfiles; op: aOperator; λ: value

  MAKE(c1,···,cm)
  = INTERSECT(c1,···,cm)

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

  MERGE(d, u1, u2, f, op)
  = u.make(d, [BOUNDARIES(u1) ∪ BOUNDARIES(u2)]\[BOUNDARIES(INTERSECTION(u1, 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)]
```

Fig. 4. Formal specification of decision-map data type.

A *decision table*, denoted dTable, 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 & Egenhofer, 1991): (i) a *set of conditions* which have to be satisfied simultaneously; and (ii) the corresponding *actions* to be taken upon these 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: `CREATE` and `ACTION`. The specification of the `CREATE` operator is deferred.

Next, "the level of underground pollution" criterion map can be generated (coarsely) by using the following procedure:

```
SLICE((WEIGHTING(sol-map,wTable) *0.3+(1/SLOPE(dtm-map) *0.7)),
sTable)
```

This procedure uses three operators:

- `WEIGHTING`: permits to transform a *soils map* into a *weighted soils map*. The input for this operator is the map layer `sol-map` and the decision table `wTable` shown in Table 1.
- `SLOPE`: permits to generate a map layer representing the slope basing on the digital terrain model, `dtm-map`.
- `SLICE`: permits to combine the two previous map layers—obtained by the application of `WEIGHTING` and `SLOPE` operators—to obtain a suitability map where suitability varies from 0.0 to 1.0 according to the data in input. `SLICE` takes in input the decision table `sTable` shown in Table 2. The value of "weight-slope" parameter associated with this decision table is given by

```
(WEIGHTING(sol-map, wTable) * 0.3) + (1/SLOPE(dtm-map) * 0.7)
```

Within the multicriteria methods that rely on utility function-based decision rules, the term criterion is a generic one that includes objectives and attributes. In this case, a criterion map represents the spatial distribution of an "attribute" measuring the satisfaction of

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

syntax:
  MAKE               procedure → criterion-map
  SET                criterion-map × aParameter × value → criterion-map
  ATTRIBUTE          map-layer × ··· × map-layer → attribute-map
  OBJECTIVE          attribute-map × ··· × attribute-map → criterion-map
  TIN                sample-data-points → map-layer
  C-TIN              contours → map-layer
  THEMATIC-SLICING   dtm → map-layer
  WEIGHTING          thematic-map × dTable → dtm

axioms:
  m: map-layer; d: dTable
  WEIGHTING(m, d)
    = d.action(m)
```

Fig. 5. Formal specification `criterion-map` of data type.

an "objective". 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 operators ATTRIBUTE and OBJECTIVE are destined to model the concepts of objective and attribute.

As we have mentioned in Section 3, the construction of a decision map require that all criteria maps are polygonal ones. In Fig. 5 we have included a series of map transformation operations permitting to obtain polygonal maps. The TIN operator permits to generate a triangulation from a sample data points. The C-TIN operator represents the *constrained Delaunay triangulation* operation permitting to transform a contours map into a TIN. The THEMATIC-SLICING operator transforms a digital terrain model into a thematic map. The WEIGHTING operator is introduced in the example above.

5.4. Specification of pstructure data type

A preference structure, pStructure, permits the decision maker to articulate his/her preferences when comparing two alternatives. In multicriteria analysis, a pStructure is often operationalized through a criterion function. Thus, a pStructure may be associated with several thresholds. Here, we adopt the most general case 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 preference situations when comparing two alternatives *a* and *b*:

$$\mathtt{aPb} \Longleftrightarrow \mathtt{g(a)} > \mathtt{g(b)} + \mathtt{pThreshold}$$
$$\mathtt{aQb} \Longleftrightarrow \mathtt{g(b)} + \mathtt{pThreshold} \geqslant \mathtt{g(a)]} > \mathtt{g(g)} + \mathtt{qThreshold}$$
$$\mathtt{aIb} \Longleftrightarrow \mathtt{g(b)} + \mathtt{Threshold} \geqslant \mathtt{g(a)andg(a)} + \mathtt{qThreshold} \geqslant \mathtt{g(b)}$$

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* (Vincke, 1992). For example, we may associate to the preference structure above the following characteristic function:

$$a\mathrm{S}b \Longleftrightarrow a\mathrm{P}b \vee a\mathrm{Q}b \vee a\mathtt{I}b$$

Table 1
Decision table wTable associated with WEIGHTING operator

type-of-sol	=Lg	=Aq	=Le
ACTION	0.2	0.3	0.7

Table 2
Decision table sTable associated with SLICE operator

weight-slope	⩾0.0 ∧ ⩽0.5	>0.5 ∧ ⩽0.8	>0.8 ∧ ⩽ 1.0
ACTION	unsuitable	possible	Recommended

The formal specification of `pStructure` data type is provided in Fig. 6. It includes three operators, `P`, `Q` and `I`, implementing the three preference situations mentioned above; and an operator, `S`, implementing the characteristic function given above.

5.5. Formal specification of `decision-rule` data type

This data type is devoted to implement the different decision rules. Each decision rule is modeled as a map algebra procedure. Fig. 7 contains the formal specification of decision-rule data type. It includes some known `decision rules`. The result of a decision rule may be either a real representing the global utility of the alternative in the input (this applies for utility function-based decision rules) or a credibility index for the pair of alternatives in the input (this applies for outranking relation-based decision rules).

For illustration, the *axioms* part of Fig. 7 contains the specification of the multicriteria sorting model ELECTRE TRI (Figueira et al., 2005b). ELECTRE TRI decision rule is defined in terms of the following procedure:

$$\mathtt{CONCORDANCE}(\mathtt{a},\mathtt{h},\mathtt{f}) * \mathtt{DISCORDANCE}(\mathtt{a},\mathtt{h},\mathtt{f})$$

The `CONCORDANCE` and `DISCORDANCE` operators permit to compute the global concordance and discordance indices, respectively (cf. Section 2.2). They are defined (for ELECTRE TRI method) as follows:

$$\left|\begin{array}{l}\mathtt{CONCORDANCE}(u,h,f) = \left[\sum_{g\in f}\mathtt{PCONCORDANCE}(u,h,g)^{*}g,c\mathtt{Weight}\right]\Big/\left[\sum_{g\in f}g.c\mathrm{Weight}\right]\\ \mathtt{DISCORDANCE}(u,h,f) = \prod_{g\in f\wedge(\mathtt{PDISCORDANCE}(u,h,g)>\mathtt{CONCORDANCE}(u,h,g))}[1-\mathtt{PDISCORDANCE}(u,h,g)]/\\ \qquad [1-\mathtt{CONCORDANCE}(u,h,g)]\end{array}\right.$$

The `CONCORDANCE` and `DISCORDANCE` operators use two new operators: `PCONCORDANCE` and `PDISCORDANCE`. The formal specifications of these two operators are defined as follows:

$$\left|\begin{array}{l}\mathtt{PCONCORDANCE}(u,h,g) = t_{1.}\mathtt{action}(\mathtt{SCORE}(u,g),\mathtt{SCORE}(h,g),g)\\ \mathtt{PDISCORDANCE}(u,h,g) = t_{2.}\mathtt{action}(\mathtt{SCORE}(u,g),\mathtt{SCORE}(h,g),g)\end{array}\right.$$

These two operators are defined through the decision tables shown in Tables 3 and 4, respectively. In these tables, 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 the `ACTION}` operator of the associated decision table.

5.6. Specification of `sd-model` data type

The specification of `sd-model` data type is provided in Fig. 8. The three first operators permit to generate the decision alternatives. For instance, `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>`

```
| Type:  pStructure
| set:  pStructure, aAlter, aCriterion, boolean

syntax:
| P aAlter × aAlter × aCriterion → boolean
| I aAlter × aAlter × aCriterion → boolean
| Q aAlter × aAlter × aCriterion → boolean
| S aAlter × aAlter × aCriterion → boolean

axioms:
| a, b:  anAlter; g:  aCriterion; p:  pStructure
|
| 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)
```

Fig. 6. Formal specification of `pStructure` data type.

<bOperator> <value>; where aCriterion represents an evaluation criterion and bOperator is a binary operator.

The *constraints*, or *admissibility criteria*, dichotomize a set of alternatives under consideration into two categories: *acceptable* (or *feasible*) and *unacceptable* (or *unfeasible*) (Malczewski, 1999). The FEASIBLE operator associated with the sd-model permits to generate the set of acceptable alternatives. It takes in input a set of decision alternatives (aSet) and a constraint similar to the one presented in the previous paragraph. Two examples of constraints are: "slope < 0.3" or "altitude > 290". When several admissibility criteria are needed, we may model them as a series of FEASIBLE operations; each of which takes the output of the previous one as input.

The alternatives generated by L-ALTERNATIVE or Q-ALTERNATIVE operators are composed of a set of spatial units, each of which is associated with a set of partial evaluations. The EVALUATE takes a linear or polygonal decision alternative, a family of criteria and an aggregation operator, aOperator, and returns the new partial evaluations that apply to the alternative as a whole.

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, aList. The PAYOFF operator takes a set of alternatives and a family of criteria and returns the performance table.

The DOMINATE is an implementation of the dominance relation used in multicriteria modeling. The *dominance relation* is defined for two alternatives *a* and *b*; and a family of criteria *F* as follows:

$$a\Delta b \Longleftrightarrow g_j(a) \geqslant g_j(b); \quad j \in F,$$

```
Type:   decision-rule
set:  decision-rule, anAltere, cFamily, pParameters, aProfile, cIndex, real

syntax:
ELECTRE-III    anAlter × anAlter × cFamily × pParameters → cIndex
ELECTRE-TRI    anAlter × aProfile × cFamily × pParameters → cIndex
MACBETH        anAlter × anAlter × × cFamily × pParameters → cIndex
PROMETHEE      anAlter × anAlter × × cFamily × pParameters → cIndex
MUAT           anAlter × cFamily × pParameters → real
AHP            anAlter × cFamily × pParameters → real
QUALIFLEX      anAlter × anAlter × cFamily × pParameters → cIndex
ORESTE         anAlter × anAlter × × cFamily × pParameters → cIndex

axioms:
a, b:  anAlter;  f:  cFamily;  h:  aProfile

ELECTRE-TRI(a, b, f)
= CONCORDANCE(a, h, f) * DISCORDANCE(a, h, f)
```

Fig. 7. Formal specification of `decision-rule` data type.

Table 3
Decision table associated with PCONCORDANCE operator

vl-v2	⩾p	⩽q	<p ∧ > q
ACTION	0	1	[p − vl + v2]/[p − q]

Table 4
Decision table associated with PDISCORDANCE operator

v2–vl	⩾ p	⩽ v	<v ∧ > p
ACTION	0	1	[v − vl + v2]/[v − p]

with at least one strict inequality. 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.

The operator QUANTIFY takes in input a performance table and a measurement scale and returns a new performance table where all qualitative criteria are transformed into quantitative ones. This operator is particularly useful for methods based on utility function-based decision rules since they require quantitative evaluation criteria.

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, e.g., Malczewski, 1999), the CREATE operator of a nProcedure is a deferred.

The operator AGGREGATE applies a decision rule to all the alternatives. The CHOICE, SORT and RANGE operators correspond to the three types of recommendations in multicriteria analysis mentioned in the end of Section 2.1. 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 in Section 5.4 the following choice function:

```
Type: sd-model
set: map-layer, decision-map, criterion-map, sUnit, uSet, cFamily, aAlter, pAlters, lAlters, sAlters, aProcedure,
nProcedure, aOperator, pTable, aScale, value
syntax:
P-ALTERNATIVE     decision-map × aCriterion × bOperator × value → pAlters
L-ALTERNATIVE     desicion-map × sUnit × sUnit → lAlters
Q-ALTERNATIVE     desicion-map × aConstraint → aAlters
EVALUATE          aAlter × cFamily × aOperator → aAlter
SCORE             aAlter × aCriterion → real
P-VECTOR          aAlter × cFamily → aList
PAYOFF            aSet × cFamily → pTable
FEASIBLE          sAlters × aCriterion × bOperator × value → sAlters
DOMINATE          sAlters × cFamily → uSet
QUANTIFY          pTable × aScale → pTable
NORMALIZE         sAlters × cFamily × nProcedure → decision-map
AGGREGATE         aSet × cFamily × decision-rule → decision-map
CHOICE            aSet × pStructure × cFunction → aSet
SORT              aSet × sCatogries × sProfiles → aSet
RANGE             aSet × rdirection → aSet
axioms:
m, r: decision-map; u:sUnit; f: cFamily; a: aProcedure; n: nProcedure; s:aSet;b:pStructure; c:cFunction;
x: anAlter; y:sAlters; v, l, m:aList

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

FEASIBLE(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(u1,g),···,SCORE(ur, g))]

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))

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

RANGE(s, p, f)
= i ← 1
While s <> ∅
[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)]
```

Fig. 8. Formal specification of sd-model data type.

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

Note that other functions may also apply as for example: $\mathtt{C(B)} = \{\mathtt{a} \in \mathtt{B} : \neg\exists \mathtt{b} \in \mathtt{B} : \mathtt{bPa}\}$. As it is shown in Fig. 8, 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 operator establishes a partial pre-order on the set of alternatives. A pre-order consists of (i) a set of equivalence classes and (ii) an order relation upon these classes. Thus, it can be implemented as a series of CHOICE operators; each of which is of the following form:

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

In each step i, RANGE returns the most preferred alternatives from set $\mathbf{B}^i$; where $\mathtt{B}^i = \mathtt{B} \backslash \mathtt{B}^{i-1}$; and $\mathtt{B}^1 = \mathtt{B}$. 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 Fig. 8, 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 class (identified by its position i in the list).

Compared to CHOICE and RANG 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 the operator CLASSIFY associated with decision-map data type. The only difference is related to the fact that CLASSIFY uses the preference structure associated with the decision-map, while SORT is a more general one and can be used to implement other multicriteria sorting methods.

6. Concluding remarks

Coupling the geographical information system (GIS) and multicriteria analysis is an active research topic as illustrated by the relatively large number of papers published since early 1990s. Unfortunately, the lack of adequate capabilities for supporting an effective multicriteria spatial modeling prevents these works from going beyond the academic contexts. In this paper, we have proposed a new algebra called decision map algebra (DMA) especially devoted to multicriteria spatial modeling. We think that DMA is of interest not only for multicriteria analysis but also in terms of developing formal specifications for spatial modeling (Ding & Fotheringham, 1992) in general.

DMA supports the major part of multicriteria spatial evaluation functions and concepts introduced in Section 2. Table 5 shows the correspondence between these functions and concepts and DMA operators. DMA includes also several operators permitting to construct decision maps as OUTRANK, GROUPING and CLASSIFY. Specifically, the concept of decision map is very useful in the sense that it facilitates the use of outranking relation-based decision rules in GIS which, according to several authors (e.g., Joerin et al., 2001; Malczewski, 1999; Marinoni, 2006), are more appropriate in spatial decision making.

To show how DMA can be used in practice, we consider the following hypothetical problem. The objective is to select a corridor for some linear infrastructures. This example may apply in problems like the construction of highways, pipelines, etc. 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 (Chakhar & Mousseau, 2006). This problem may be "solved" using the following sequence of instructions:

```
m: map-layer
c1,c2,c3: criterion-map
s,e: sUnit
res: uSet
f: cFamily
b: sProfiles
c1 ← m.MAKE((population/ area) * 100)
c2 ← m.MAKE((active-population/ area) * 100)
```

Table 5
Multicriteria evaluation functions and the corresponding DMA operators

Multicriteria modeling functions	DMA operators
Generation of decision alternatives	`P-ALTERNATIVES`, `L-ALTERNATIVES`, `Q-ALTERNATIVES`
Construction of criteria maps	All the operators associated with data type `criterion-map`
Construction of attribute maps	`ATTRIBUTE`, `OBJECTIVE`
Generation of performance table	`SCORE`, `EVALUATE`, `P-VECTOR`, `PAYOFF`
Quantification	`QUANTIFY`
Normalization	`NORMALIZE`
Pre-analyse of dominance	`DOMINATE`
Generation of feasible decision alternatives	`FEASIBLE`
Preference modeling	`P`, `Q`, `I`, `S`, `C`
Evaluation criteria weighing	The operator SET associated with `criterion-map`
Aggregation	`AGGREGATE`
Final recommendation	`RANGE`, `CHOICE`, `SORT`

```
c3 ← m.MAKE((total-area - used-area))
d ← INTERSECT(c1,c2,c3)
f ← {c1,c2,c3}
b ← {{3.5,0.2,0.3,2.5,0.2,0.3,0.25,0.2,0.3},{2,0.2,0.3,3.
5,0.2,0.3,4,0.2,0.3},
{20,2,3,15,2,3,10,2,3}}
d ← d.CLASSIFY(d,b,f)
res ← d.L-ALTERNATIVE(s,e)
```

First, we start by generating the criteria maps. In this example, three criteria are considered (`c1`, `c2` and `c3`). These criteria maps are generated by using the `MAKE` operator. Three map algebra procedures are used and applied on the initial map layer `m`. These procedures, provided as parameters for the `MAKE` operator, compute: (i) the demographic density, (ii) the employment level, and (iii) the available territory.

Then a decision map `d` is generated using `INTERSECT` operator to combine the criteria maps. Then, the `CLASSIFY` operator is applied on `d` to generate a final decision map. To apply this operator, we need to use a family of criteria, `f`, and a set of profile limits, `b`. In the example above, the set of profiles `b` is defined as a matrix. The elements of this matrix are the following:

(g_j)	$g_j(b_3)$	$q_3(g_j)$	$p_3(g_j)$	$g_j(b_2)$	$q_2(g_j)$	$p_2(g_j)$	$g_j(b_1)$	$q_1(g_j)$	$p_1(g_j)$
g_1	3.5	0.2	0.3	2.5	0.2	0.3	0.25	0.2	0.3
g_2	2	0.2	0.3	3.5	0.2	0.3	4	0.2	0.3
g_3	20	2	3	15	2	3	10	2	3

Then, the operator `L-ALTERNATIVE` is applied on d to generate the set of potential corridors relating the start (`s`) and the end (`e`) spatial units, which are provided as parameters.

Currently, the implementation of `DMA` is ongoing. `DMA` is being implemented on ArcGIS 9.1 of `ESRI`. In future time, we envisage the development a script-based version of `DMA`.

Acknowledgements

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions, which considerably improved the technical and the presentation of the paper. The authors would like to thank Dr. M. Manouvrier for her valuable comments on an earlier version of this paper and Dr. M.-J. Bellosta-Tourtier for her comments and suggestions on the UML model.

Appendix A. Formal specification of some elementary data types

See Figs. A.1–A.7.

```
 Type:  gPoint
 set:  gPoint, gPolygon, real, boolean
syntax:
 MAKE          real × real → gPoint
 X             gPoint → real
 Y             gPoint → real
 ISEQUAL       gPoint × gPoint → boolean
 DISTANCE-PP   gPoint × gPoint → real
 DISTANCE-PQ   gPoint × gPolygon → real
axioms:
 i,j,k,l:  real; p,q:  gPoint

 X(MAKE (i, j))
 = i

 Y(MAKE (i,j))
 = j

 DISTANCE-PP(MAKE (i, j), MAKE (k, l))
 = sqrt(((X(MAKE (i, j)) - X(MAKE (k, l))) * (X(MAKE (i, j)) - X(MAKE (k, l)))) +
   ((Y(MAKE (i, j)) - Y(MAKE (k, l))) * (Y(MAKE (i, j)) - Y(MAKE (k, l)))))
```

Fig. A.1. Formal specification gPoint.

```
 Type:  gLine
 set:  gPoint, gLine, gPolygon, real, boolean
syntax:
 MAKE             gPoint × gPoint → gPoint
 START            gLine → gPoint
 END              gLine → gPoint
 LENGTH           gLine → real
 POINT-IN-LINE    gLine × gPoint → boolean
 INTERSECT-LQ     gLine × gPolygon → boolean
axioms:
 p,q:  gPoint, l:  gLine

 START (MAKE (p, q))
 = p

 END (MAKE (p, q))
 = q

 LENGTH(l)
 = DISTANCE-PP(START(l),END(l))
```

Fig. A.2. Formal specification of gLine data type.

```
Type:  gPolygon
set:  gPoint, gLine, gPolygon, Rectangle, real, boolean
syntax:
  MAKE               gPoint × ··· × gPoint → gPolygon
  POINT-IN-POLYGON   gPolygon × gPoint → boolean
  INTERSECT          gPolygon × gPolygon → boolean
  INTERSECTION       gPolygon × gPolygon → gPolygon
  CLIPPING           gPolygon × Rectangle → gPolygon
  ADJACENT           gPolygon × gPolygon → boolean
  AREA               gPolygon → real
  CENTROID           gPolygon → gPoint
  BOUNDARIES         gPolygon → gLine
  MERGE              gPolygon × ··· × gPolygon → gPolygon
axioms:
  p1, ···, pn, r:  gPoint; l:  gLine;

  AREA(MAKE(p1, ···, pn))
  = _area

  CENTROID(MAKE(p1, ···, pn))
  = _gPoint

  POINT-IN-POLYGON(MAKE(p1, ···, pn), r)
  = if (∀ (p) in (p1, ···, pn) X(r) ≤ X(p) and Y(r) ≤Y(p)) then true

  INTERSECTS(MAKE (p1, ···, pn),l)
  = POINT-IN-POLYGON( MAKE(p1, ···, pn), SART(l)) or
     POINT-IN-POLYGON(MAKE(p1, ···, pn), END(l))
```

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

```
Type:  map-layer
set:  map-layer, gPoint, gLine, gPolygon, real
syntax:
  MAKE name → map-layer
  PUT-P map-layer × real × real → gPoint
  PUT-L map-layer × gPoint × gPoint → gLine
  PUT-Q map-layer × gPoint × ··· × gPoint → gPolygon
  INTERSECT map-layer × ··· × map-layer → map-layer
  UNION map-layer × ··· × map-layer → map-layer
axioms:
  m:  map-layer; p, p1, p2, q1, ···, qn:  gPoint; l:  gLine; q:  gPolygon; v1, v2:  real

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

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

  PUT-Q(m, q1, ···, qn)
  = q.MAKE(q1, ···, qn)
```

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

```
Type:  aOperator
set:  aOperator, value
syntax:
  CREATE DEFERRED → aOperator
  COMBINE value × value × ··· × value → value
axioms:
  x, y, z:  value

  COMBINE(x,y)
  = x+y

  COMBINE(x,COMBINE(y,z))
  = x + COMBINE(y,z)
```

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

```
| Type:  dTable
| set:  dTable, bCondition, value
syntax:
| CREATE bCcondition × ··· × bCcondition × value ···
|          bCcondition × ··· × bCcondition × value → dTable
| ACTION value × value × ··· × value → value
axioms:
| v_1, ···, v_m, a_1, ···, a_n:  value; c_1^1, ···, c_1^m, ···, c_n^1, ···, c_n^m:  bCondition
|
| CREATE
| = (((c_1^1, ···, c_1^m), a_1), ···, ((c_n^1, ···, c_n^m), a_n))
|
| ACTION(v_1, ···, v_m)
| = if c_1^1, ···, c_1^m then a_1
|   else
|   [···
|     [if c_{n-1}^1, ···, c_{n-1}^m then a_{n-1} else a_n]
|   ]
```

Fig. A.6. Formal specification of `dTable` data type.

```
| Type:  aList
| set:  aList, value, aPosition
syntax:
| INSERT aList × value × aPosition → aList
| LOCALIZE value × aList → aPosition
| GET aPosition × aList → value
| FIRST aList → aPosition
| RAZ aList → aList
| REMOVE aPosition × aList → aList
| NEXT aPosition × aList → aPosition
| BEFORE aPosition × aList → aPosition
axioms:
| v_1, ···, v_n, v:  value; i:  integer
|
| INSERT((v_1, ···, v_{i-1}, v_i, ···, v_n),v,i)
| = (v_1, ···, v_{i-1}, v, v_i, ···, v_n)
|
| GET(i,(v_1, ···, v_i, ···, v_n))
| = v_i
```

Fig. A.7. Formal specification of `aList` data type.

References

Belton, V., & Stewart, T. J. (2002). *Multiple criteria decision analysis: An integrated approach*. Boston: Kluwer Academic Publishers.

Caldwell, D. R. (2000). Extending map algebra with flag operators. In *Proceedings of the fifth international conference on geocomputation*.

Câmara, G., De Freitas, U. M., & Cordeiro, J. P. C. (1994). Towards an algebra of geographical fields. In *5th Proceedings Brazilian symposium on computer graphics and image processing, Anais, Curitiba* (pp. 205–212).

Carver, S. (1991). Integrating multi-criteria evaluation with geographical information systems. *International Journal of Geographical Information Systems, 5*(3), 321–339.

Chakhar, S. (2006). *Multicriteria decisional cartography: Formalization and implementation*. Ph.D. thesis, University of Paris Dauphine, Paris, France.

Chakhar, S., & Mousseau, V. (2006). Generation of spatial decision alternatives based on a planar subdivision of the study area. In *Proceedings the 2006 ACM/IEEE international conference on signal-image technology and internet-based systems, Hammamet, Tunisia, December 17–21* (pp. 78–87).

Chakhar, S., Mousseau, V., Pusceddu, C., & Roy, B. (2005). Decision map for spatial decision making. In *The 9th international computers in urban planning and urban management conference, London, UK, 29 July–1 August*.

Chakhar, S., & Martel, J.-M. (2003). Enhancing geographical information systems capabilities with multi-criteria evaluation functions. *Journal of Geographic Information and Decision Analysis, 7*(2), 47–71.

Chakhar, S., & Mousseau, V. (2004). Towards a typology of spatial decision problems. *Annales du LAMSADE*(2), 125–154.

Chakhar, S., & Mousseau, V. (2007). Spatial multicriteria decision making. In S. Shehkar & H. Xiong (Eds.), *Encyclopedia of geographic information science*. New York: Springer-Verlag.

Chan, K., & White, D. (1987). Map algebra: An object-oriented implementation. In R. Angeebrug, & Y. Schiffman (Eds.), *International geographic information systems (IGID) symposium: The research agenda, Arlington, VA* (Vol. II, pp. 127–150).

Chrisman, N. (2002). *Exploring geographic information systems, chapter transformations* (2nd ed., pp. 217–242). New York: John Wiley & Sons.

de Floriani, L., Magillo, P., & Puppo, E. (1999). *Handbook of computational geometry chapter applications of computational geometry to geographic information systems* (pp. 333–386). Elsevier Science B.V.

Ding, Y., & Fotheringham, A. S. (1992). The integration of spatial analysis and GIS. *Computers, Environment and Urban Systems, 16*(1), 3–19.

Dorenbeck, C., & Egenhofer, M. (1991). Algebraic optimization of combined overlay operations. In D. Mark, & D. White (Eds.), *Auto-Carto 10: Technical papers of the 1991 ACSM-ASPRS annual convention, Baltimore, Maryland, USA, March, ACSM-ASPRS* (pp. 296–312).

Figueira, J., Greco, S., & Ehrgott, M. (Eds.). (2005a). *Multiple criteria decision analysis: State of the art surveys*. New York: Springer-Verlag.

Figueira, J., Mousseau, V., & Roy, B. (2005b). ELECTRE methods. In *Multiple criteria decision analysis: State of the art surveys* (pp. 133–162). New York: Springer-Verlag.

Grunberg, W., Dale, J. C., Haseltine, M., Lerman, N., Olsson, A., & Orr, B. J. (2004). Web-based map-algebra challenges: A polygon solution. In *Proceedings of the 24th annual ESRI international user conference, San Diego, California, August 9–10*.

Guttag, J. (1977). Abstract data types and the development of data structures. *Communications of the ACM, 20*(6), 396–404.

Guttag, J., & Horning, J. J. (1978). The algebraic specification of abstract data types. *Acta Informatica, 10*(1), 27–52.

Joerin, F., Thériault, M., & Musy, A. (2001). Using GIS and outranking multicriteria analysis for land-use suitability assessment. *International Journal of Geographical Information Science, 15*, 153–174.

Karssenberg, D., Burrough, P. A., Sluiter, R., & de Jong, K. (2001). The PCRaster software and course materials for teaching numerical modelling in the environmental sciences. *Transactions in GIS, 5*, 99–110.

Koo, B. K., & O'Connell, P. E. (2006). An integrated modelling and multicriteria analysis approach to managing nitrate diffuse pollution: 1. Framework and methodology. *International Journal of Science of the Total Environment, 359*, 1–16.

Laaribi, A., Chevallier, J. J., & Martel, J.-M. (1996). A spatial decision aid: A multicriterion evaluation approach. *Computers, Environment and Urban Systems, 20*(6), 351–366.

Lin, F.-T. (1998). Many sorted algebraic data models for GIS. *International Journal of Geographical Information Science, 12*(8), 765–788.

Malczewski, J. (1999). *GIS and multicriteria decision analysis*. New York: John Wiley & Sons.

Malczewski, J. (2006). A GIS-based multicriteria decision analysis: A survey of the literature. *International Journal of Geographical Information Science, 20*(7), 703–726.

Marinoni, O. (2006). A discussion on the computational limitations of outranking methods for land-use suitability assessment. *International Journal of Geographical Information Science, 20*(1), 69–87.

Mennis, J., Viger, R., & Tomlin, C. D. (2005). Cubic map algebra functions for spatio-temporal analysis. *Cartography and Geographical Information Science, 32*(1), 17–32.

Meyer, B. (1988). *Object-oriented software construction*. New York: Prentice-Hall.

Murray, S., Breslin, P., Ormsby, T., & Miller, B. (2000). A visual framework for spatial modeling. In: *Proceedings of the 4th international conference on integrating GIS and environmental modeling (GIS/EM4): Problems, prospects and research needs, Banff, Alberta, Canada*.

Pullar, D. (2001). MapScript: A map algebra programming language incorporating neighborhood analysis. *GeoInformatica, 5*(2), 145–163.

Ritter, G. X., Wilson, J., & Davidson, J. (1990). Image algebra: An overview. *Computers Vision, Graphics, and Image Processing, 49*, 297–331.

Roy, B. (1996). *Multicriteria methodology for decision aiding*. Dordrecht: Kluwer Academic Publishers.

Serra, J. P. (1982). *Image analysis and mathematical morphology*. London: Academic Press.
Su, B., Li, Z., Lodwick, Z. G., & Müller, J. C. (1997). Algebraic models for the aggregation of area features based upon morphological operators. *International Journal of Geographical Information, 11*(3), 233–246.
Takeyama, M., & Couclelis, H. (1997). Map dynamics: Integrating cellular automata and GIS through Geo-Algebra. *International Journal of Geographical Information Science, 11*(1), 73–91.
Tomlin, C. D. (1983). A map algebra. In *Proceedings of the 1983 Harvard computer graphics conference, Cambridge, Massachusetts* (vol. 2, pp. 127–150).
Tomlin, C. D. (1990). *Geographic information systems and cartographic modeling*. Englewood Cliffs, NJ, USA: Prentice-Hall.
Vincke, Ph. (1992). *Multicriteria decision-aid*. Chichester: John Wiley & Sons.
Wesseling, C. G., Karssenberg, D., van Deursen, W. P. A., & Burrough, P. A. (1996). Integrating dynamic environmental models in GIS: The development of a dynamic modelling language. *Transactions in GIS, 1*, 40–48.