NETHERLANDS GEODETIC COMMISSION

PUBLICATIONS ON GEODESY NEW SERIES

ISSN 0165 1706 NUMBER 43

METHODS FOR THE GENERALIZATION OF GEO-DATABASES

EDITED BY

MARTIEN MOLENAAR

1996

NEDERLANDSE COMMISSIE VOOR GEODESIE, THIJSSEWEG 11, 2629 JA DELFT, THE NETHERLANDS
TEL. (31)-(0)15-2782819, FAX (31)-(0)15-2782745

Contents

Foreword

Multi-scale approaches form at present one of the focal points of the GIS research community. This is due to the rising awareness that many processes on the earth surface can only be monitored and managed if they are understood in their geographical context. Part of this context is defined by the scale range at which these processes work. If we consider for instance the development of land use in a district, then we see that this is driven by actors at a lower aggregation level such as farmers, residents and companies. Their activities are constrained however by the socio-economic conditions and the infrastructure at regional level and by the macro economic planning at national and supra national level. An other example is the development of the natural vegetation cover of certain regions. The actual state of such a vegetation cover is defined by the co-occurrence of species that form vegetation types, which are part of eco-systems. Their development will be constrained by climatic conditions, the geologic and soil condition of the region and its hydrology. Here too we find hierarchical levels of organization.

The monitoring and management of such processes requires information at different scale levels. The research problem for the GIS community in this context is:
- to decide for each type of process which information should be handled at each scale level,
- to develop methods for transferring information between the different scale levels so that duplication of expensive data acquisition can be avoided as much as possible, and so that the consistency between data at the different scale levels can be maintained.

This second item is strongly related to the long standing research problem of map generalization, that is why it is often seen from that perspective. Researchers in this field become more and more aware of the fact, however, that multi-scale approaches in a GIS environment can be dealt with by data base generalization operations. These allow approaches that are quite different form the procedures applied in map generalization and they are more flexible.

This new research field derives it terminology and many of its concepts from both cartography and the object oriented approaches of computer science. This mixture of the idiom of different disciplines leads often to confusion so that the concepts that are covered by the terminology become fuzzy. This confusion may send researchers in the wrong direction when they want to solve multi-scale problems. The subcommission on GIS of the Netherlands Geodetic Commission organized a workshop on the 7th December 1995 to discuss these problems. Five authors were invited to present papers in which generalization problems were treated from several perspectives:
- Ferjan Ormeling presented the cartographic perspective, where the visualization of spatial data at different scale levels plays a central role,
- Martien Molenaar presented a data base perspective for generalization, emphasising the role of topologic and semantic (hierarchical) data models,
- Peter van Oosterom et. al. explained how multi-scale storage of spatial data could be organised by means of three hierarchical datastructures,

- Arnold Bregt et. al. used the example of a soils database to explain three spatial aggregation strategies and their effect on the quality of the derived databases,
- Arthur van Beurden discussed the difference between the processing of aggregated data and the aggregation of processed data, the production of maps for environmental data served as an example.

These papers form the contents of this bundle. We hope that the different approaches they present will help the reader to understand the different concepts that play a role in the generalization of spatial data and that the different strategies that are presented are of use for solving multi-scale problems.

The organizers of the workshop would like to thank the Netherlands Geodetic Commission for the financial support for the workshop and for the publication of these proceedings. Special thanks are due to Frans Schröder, secretary of this commission, who was responsible for a major part of the preparations of the workshop and for this publication.

Martien Molenaar
Chairman of NCG subcommission on GIS

Aggregation objectives and related decision functions

Ferjan Ormeling

Cartography Department
Faculty of Geographical Sciences
Utrecht University, The Netherlands
E-mail: ormeling@frw.ruu.nl

Definitions

As this publication is aimed at both geodesists and cartographers, it seems sensible to first define what is understood by the terminology used hereafter: Generalisation is understood as the framework within which Aggregation is performed. Generalisation is defined, as the Swiss cartographer Knöpfli (1990) did, i.e. as an abstraction of those aspects of reality that are relevant to a specific query. In this process it is essential to portray optimally the terrain characteristics selected. The point is, 'to render the suitability of an area in terms of a specific objective' (Knöpfli 1990). So in order to be able to perform specific tasks (e.g. to read off specific information required) spatial data are being generalised.

Traditionally, in cartography graphical and conceptual generalisation are being differentiated between (see figure 1). **Graphic generalisation** refers to those conversions in which the type of symbolisation does not change; though it may be enhanced, or even exaggerated in order to still be legible, especially when reduced. The objective of graphical generalisation is to diminish the spatial resolution of the image. But one should not focus too much on this scale reduction: generalisation can also be performed on behalf of derived products on the same scale. Usually the action of graphical generalisation is performed on an analogue map (on an overlay placed over it) or, when performed on a digital map, on the monitor screen.

From the viewpoint of analogue cartography, **Conceptual generalisation** refers to those conversions in which either the manner of rendering, i.e. symbolisation, changes or where the relation between the objects rendered changes. So it is the relationships between the map topic and the symbol which represents it, that change. This is called a semantic relationship, and that is why one can also refer to this type as to semantic generalisation. So in this action either the symbol changes, or the meaning of the symbol. From the viewpoint of digital cartography this also engenders a change in the relationships defined within the database. Usually, classes or categories are being joined together (this will lead to a decrease in semantic resolution) or being reclassified. But semantic generalisation might also lead to consciously modifying existing relationships in order to structure the results.

This division between graphical and conceptual generalisation can also be applied to the aggregation part: so one may distinguish between graphical and conceptual aggregation. For the sake of this survey even a third type of aggregation was distinguished, as it could not be

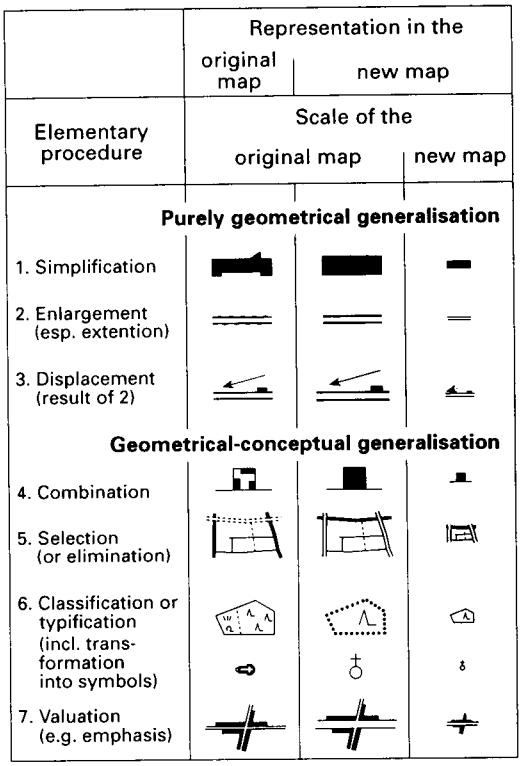


Figure 1
Graphical and conceptual generalisation
(Hake and Grünreich 1994)

grouped with either of the preceding ones: geometrical aggregation. So this paper will be distinguishing graphical, conceptual and geometrical aggregation.

Graphical aggregation refers to the joining together of map objects, on an analogue map or on a monitor screen, on the basis of their proximity, contiguity or their occurrence at the same location. The objective of this joining together generally is a reduction of the resolution. This reduction is necessary because, even if individual symbols may remain legible optically, too large an accumulation of symbols might not allow for perceiving any structure in the image.

Conceptual aggregation refers to the joining together, in the digital line model (DLM), of object classes to larger classes ('super classes'), having a higher level of abstraction. Here the resolution is reduced as well, even if this reduction takes place at another level. In principle, the individual lines or countours will not change, but their number may.

In **Geometrical aggregation** contiguous objects may be joined as well: enumeration areas may be joined and the class or attribute values for these enlarged areas be computed anew. Here, both the number of classes (because of averaging) and the number of map units (by joining together contiguous units or objects) will be reduced. It is realised that this can also be regarded as a form of conceptual aggregation, but the point of departure is different: areas are joined together first instead of combining attribute values. As the term 'semantic' refers (for the author) too much to the attribute contents, this third type is called geometrical aggregation.

Aggregation objectives

Each of the aggregation types discerned here has its own objectives. In all three cases a reduction of the resolution is aimed at. According to Knöpfli (1990), 'A good cartographic generali-

sation can be distinguished from a bad one in the appropriate data reduction: in a good generalisation the cartographer has understood to render the appropriate characteristics in a suitable number'. In conceptual aggregation, the direct objective (i.e. a reduction of the semantic resolution) is attained by a controlled reduction of the data resolution. The indirect objective here would be to enable or ease the data analysis. The direct objective of graphical aggregation would be a reduction of spatial resolution in order to generate images that would answer the communication objective (i.e. visualising relationships). The direct objective of geometrical aggregation would be strongly related to the former: the reduction of both spatial and semantic resolution. Its indirect objective being to provide an overview of larger areas.

Despite this tripartition it is realised that these three types of aggregation will often be difficult to disentangle. In the representation of inner cities on topographic maps, for example, not only buildings, but the backyards or gardens belonging to them as well will be joined in a new legend class, 'urban settlement'. This is a combination of conceptual and graphical aggregation. These various aggregation objectives have been defined from a conceptual viewpoint. Aggregation objectives that result in data reduction and reduction of the resolution can also be derived from a more general communication viewpoint. Then one might discern:

1. Obtaining an overview (by a reduction of the number of signatures to be discerned on the map)
2. The possibility for comparison with other datasets
3. Representation on a scale level for which the accuracy or representativity would be higher
4. For reasons of secrecy or privacy

1. That a better overview should be an indirect objective of aggregation is self-evident. The lesser the number of signatures or classes in the model or on the map, the better the chance for a good overview. Discovering patterns can also be arrived at by the use of an aggregation slide bar, as by changing the level of aggregation interactively the possibility for exploration will be optimalised. But the possibility to confirm patterns discerned will aslo be strengthened in this way. In Dutch physical planning, the 'Randstad' refers to the well-known pattern of a contiguous ring of highly urbanized areas around a central agricultural area ('Green Hart') in the western part of the country. But perceiving this pattern will depend on the level of aggregation: it will only stand out when the map base consists of municipalities. When smaller enumeration areas are used, the ring will not be contiguous, when larger areas are used, the central agricultural area will not stand out any more!

2. An example of the possibility of comparison with other data sets is provided by a GIS in which the Ammonia (NH_3) emission in the Netherlands per 5x5 km grid square is compared to a digitized analogue soil map 1:50 000. Combination of both datasets (in order to find out whether a relationship exists between specific soil types and emission values) using the overlay-technique would lead to an apparent accuracy that should be rejected completely. It is obvious the soil data should be processed first in the same way as the Ammonia data have been subjected to, and be aggregated to the same grid squares. Possible conclusions will make a lot more sense then.

3. The situation could arise that information for the enumeration area level selected is not accurate enough, for instance because the national samples were not large enough. By enlarging the enumeration areas, for instance by turning from municipalities to districts or cantons, the same sample could have an adequate representativity. On the other hand, aggregation can also lead to a perceived higher inaccuracy, by combining areas that belong to the same legend class: Because the contours between areas in the same class would be obliterated, enumeration areas will look much larger, so the resulting values will be perceived as being the

result of much more averaging, and therefore likely to be less reliable than the original map. See figure 2.

4. Finally, enumeration areas can be aggregated because otherwise, in an unit area with 3 companies, two of the three, by combining their resources, together could work out the characteristics of the third company. So because of privacy regulations and in order to guarantee individual data secrecy either areas are being aggregated or random numbers are added to the data at the lowest aggregation level. Both techniques are used, for instance, at the British Central Statistical office.

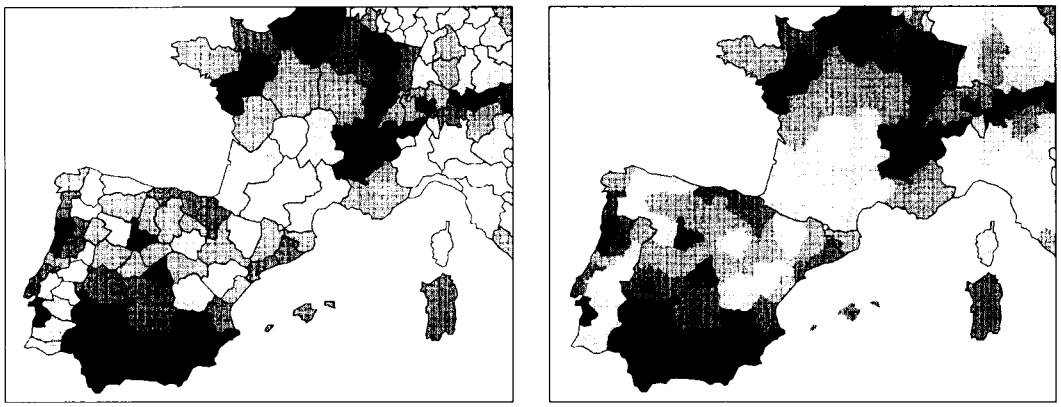


Figure 2
Comparing territorial units. Left with, right without enumeration area contours between areas that fall in the same class. (from: Quick, 1994)

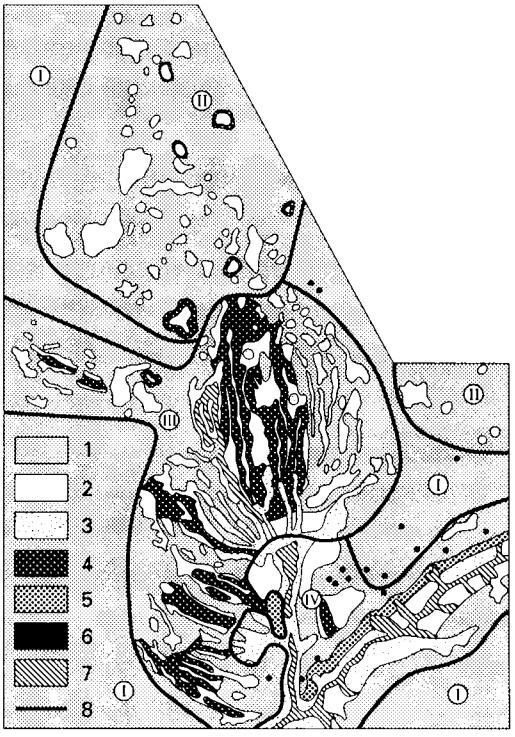
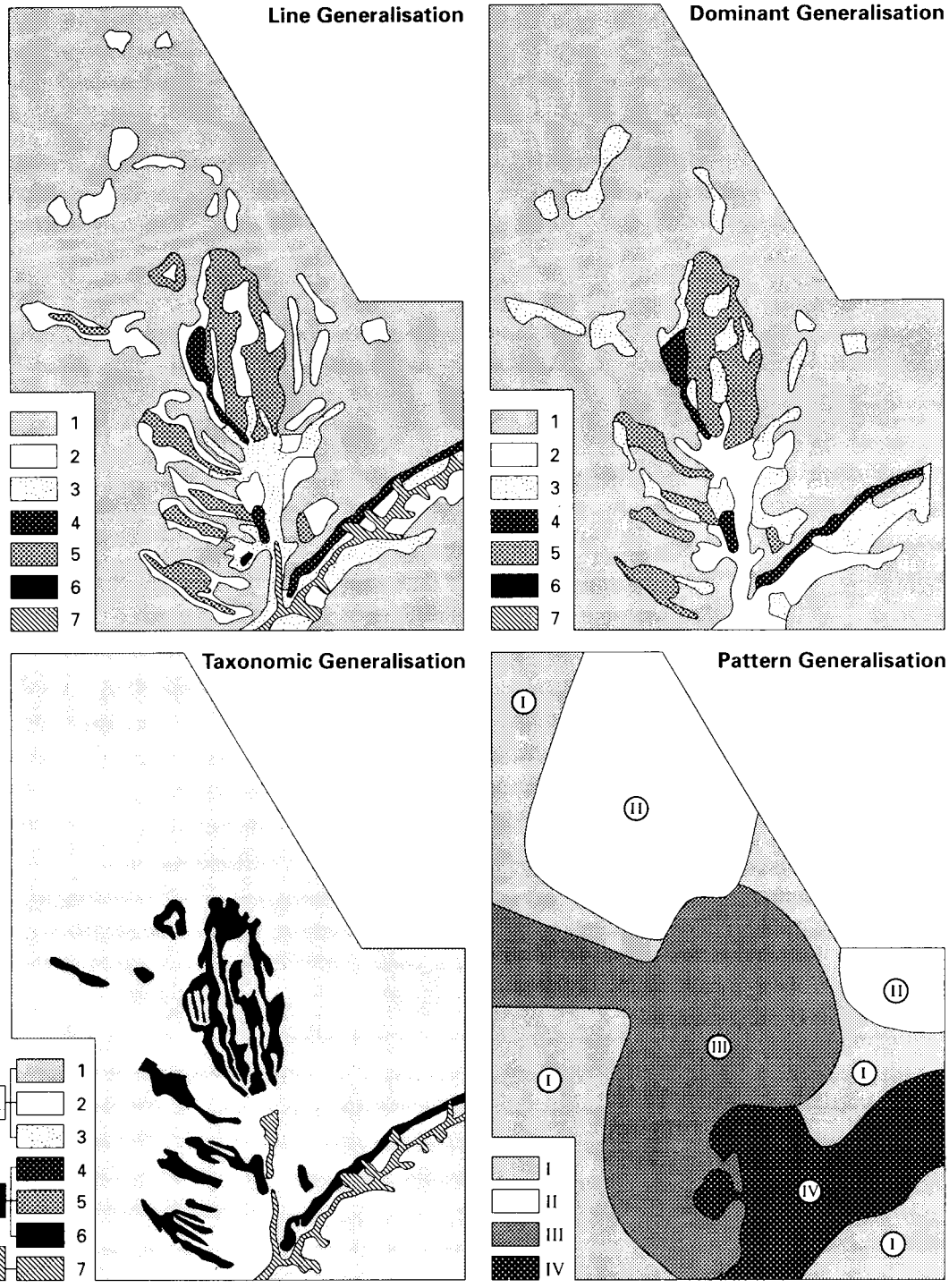


Figure 3
Different types of aggregation.
3a. original;
3b. line generalisation;
3c. dominant generalisation;
3d. taxonomic generalisation;
3e. pattern generalisation.
Based on Fridland (1976)



Notation of soils:

1) deep rich typical chernozems;
2) deep rich weakly leached chernozems;
3) deep rich strongly leached chernozems;
4) deep rich calcareous chernozems;
5) moderately deep rich calcareous chernozems;
6) burrowed calcareous chernozems;
7) chernozem-meadow soils.

Components:

I) patchy ESA
II) background closed leaching patchiness
III) background semiopen leaching patchiness
IV) simple open combine

Examples of aggregation applications

In order to apply the terminology defined above, it is proposed to look at a soil map generalisation example.This example is derived from Fridland (1976). The small area concerned, somewhere in the former Soviet Union, is characterised by a number of different soil types, occurring in elongated contiguous bands (see figure 3a). This area will be subjected to various types of generalisation. The scale of the examples is about 1:10 000. Fridland discerns between line generalisation, dominant generalisation, taxonomic generalisation and pattern-generalisation.

a. **line generalisation** (figure 3b). This type has little to do with aggregation, the only similarity being the attempt to reduce the resolution. In order to realise that in some places graphical aggregation has been performed. Locally contours will be simplified, while trying to characterize existing patterns. It is a form of graphical generalisation.

b. **dominant generalisation** (figure 3c). Here the issue is the assignment of areas below a given surface threshold value in the image to one of the larger units around them. Because of this type of generalisation the number of map units has been reduced substantially; during this process the remaining contours have been streamlined as well. On the Dutch soil map 1:50 000 it is customary not to render areas smaller than 4 hectare, if from the resulting unit they will be assigned to they will not make up more than 30%. So this is an aggregation process, but this type of aggregation only depends on considerations related to the geometry of the image, not to semantic ones. So this type of aggregation still belongs to graphical aggregation.

c. **taxonomic generalisation** (figure 3d). With this type the reduction of resolution is effectuated by intervening in the data model; the number of occurring soil types will be reduced to a smaller number of different types. This joining together of the soil types is effectuated on the basis of taxonomy. Generally this operation consists of going back a step in the process of subdividing. This will lead to a smaller number of legend-units, and thereby to a smaller number of map units as well. The remaining contours or boundaries will be as erratic as before. It will be plain that this type of aggregation belongs to the conceptual class. Vegetation maps will subdivide into species, genera, families, orders and classes. So an example of taxonomic generalisation can be a subdivision into genera instead of in species.

d. **pattern generalisation** (figure 3e). This type of aggregation strives at maintaining functional or topological relationships. The larger part of the sandy Eastern Netherlands used to be characterised by a land use pattern consisting of tiny areas of arable land (marks), fertilized by manure from the sheep that were held on the moors, while on the elongated meadows along the brooks cattle was held for dairy products. When one would generalise this kind of pattern, one would end up only with moors, as the tiny marks and the narrow elongated meadows would easily be suppressed in both line and dominant generalisation. But if one would recognise this complex of moors, marks and meadows as a functional unit, that is as a specific pattern, not to be subdivided, creating a moor/mark/meadows superclass would be a perfect solution for generalisation.

Fridland works along the same lines: he discerns patterns, and his method consists of naming these different patterns by developing a distinct terminology, on the basis of combinations of shapes. The definitions of the legend units in his method tend to be unwieldy, but mapwise it results in a much better overview.

Another of Fridland's examples works on the basis of groundwater level variations. Because of these, one and the same sedimentary deposit can result in different soil types, because of its

site being either on a hill top, on a slope, or in a valley. Generalisation of such a pattern would result in the same kind of map as a generalised contour pattern. But by aggregation of these soil types that only differ on the basis of the ground water level, larger complexes can be discerned. For instance: 'composition of brown forest soils on the hills, with podsolised brunizems on low slopes and pseudo gley soils in badly drained valleys'. This is a rather long denomination, but quite as relevant, if not more, than a simplification of the contourline pattern or an omission of all the distinct areas below a threshold value. Such an aggregation is clearly meant to provide an overview. If one would want to zoom in on the area, one could determine, on the basis of the contour map or the slope map, which kind of brown forest soil would be expected at a specific location. So here as well it is a reduction of the resolution which one arrives at, based on both graphical and conceptual aggregation.

A last example for applying the terminology is a socio-economic one: election results. In the Netherlands, about 20 different parties would participate, and the results would be polled at about 9000 different stations. In order to be able to visualise the results in a single image some substantial aggregation must be effectuated. Here we have conceptual aggregation by combining individual parties on the basis of similarities in their programmes: a tripartition into socialist, liberal and christian democrats can be arrived at. Another form of conceptual aggregation can be arrived at by joining together all those parties that have less than 5% each of the total number of votes. Apart from that, geometrical aggregation can be applied, by joining polling districts together, and combine them into municipalities. Within the municipalities the percentages for each party will have to be computed anew. Though there is no graphical aggregation in this example, there is some graphical generalisation, as generally, contours will be simplified.

Quite another type of pattern aggregation can be found in the kind of transformations that will be effectuated for example in trend surface analysis, or in determining residuals from regression. An example of this last case is the forest extent in Portugal. The necessary information can be obtained from aerial photographs or from land cover maps. When a grid is overlayed over such an inventory image, it would be possible to estimate the forest percentage for each grid cell; by this procedure the values have been generalised as they were related to larger enumeration areas. Possible derived products would be either grid-choropleths or isopleth maps of the forest distribution. These maps would allow one to compare the forest distribution with other parameters, such as precipitation or relief.

Aggregation techniques

What aggregation techniques have been used in these applications?

1. Adding small units to larger dominant units. This procedure is similar to resampling with a larger pixel size. It can be considered a type of graphical aggregation.
2. Taxonomical aggregation. This is a form of conceptual aggregation.
3. Pattern aggregation. This is a form of conceptual aggregation.
4. Functional aggregation (see the Eastern Netherlands landuse example). This also is a form of conceptual aggregation.
5. Clustering on the basis of enumeration units (geometrical aggregation).

Consequences of applying aggregation techniques

Before endeavouring aggregation, one should be aware of its possible effects. This is the more important when the aggregated products are to be analysed later on. In a digital environment one should not worry about the effects of graphical aggregation, as the processing will be

done within the digital line model (DLM) anyway. Graphical aggregation will only influence analysis results when analogue imagery will be analysed.

There are both numerical consequences as well as those relating to the perception of patterns in graphical aggregation. Amongst the numerical consequences are changes that will occur in numbers, surface areas as well as circumferences of aggregated objects. Relationships between the classes concerned can thereby change. When the representation of a forest is being generalised on the basis of the dominant generalisation method, whereby smaller outlieing forest areas are assigned to the surrounding arable lands, the share of the forest in the mapped area will diminish. By adhering to a threshold value for the surface area to be represented, patterns may change. When the clearings in a forest are determined by specific soil types, this aggregation procedure will lead to the disappearance of this pattern and of the possibility to correlate it with a soil map. German and Austrian cartographers have developed an extensive casuistry related to the various forms of aggregation and to the attempts to let the aggregated image continue to represent a correct image of both the shares, the patterns and structures.

In conceptual aggregation joining together classes will lead to reduction of the database used for computation. It will make a lot of difference whther one has to do computations for 20 parties and 9000 polling districts or with three political denominations and 600 municipalities. Of course this geometrical aggregation will lose one some local differentiation, and the taxonomic aggregation will reduce the information contents.

By joining enumeration areas (administrative areas, grid cells), geometrical aggregation will lead to a strong fluctuation in discernable patterns in areas with a non-homogeneous population density. It will also lead to a disappearance of locally or regionally occurring anomalies. So resulting images will be scale-dependant.

Decision moments

Because of a number of aggregation consequences, that can be expected a priori, one has to decide about the aggregation levels or -thresholds one intends to adhere to before effectuating one of these types of aggregation.

There is the **temporal effect** first: in the clustering of areas that occurs in geometrical aggregation, it is important whether derived data are computed prior to or after aggregation, as this might lead to different values. The mode and the median value for the Labour party percentage on the basis of 9000 polling districts will be different from that for 600 municipalities or 12 provinces. The more units there will be taken into account, the more extreme relative values will be found.

In **graphical aggregation** a specific minimum surface for individually rendered map objects carries a potential pollution of larger map units with deviating units. Setting specific threshold values will have computable consequences for the level of pollution, and thereby also for the accuracy or quality of the map. When a specific accuracy of the attribute information is required, this could be effectuated by setting these threshold values for graphical aggregation at specific values.

In **conceptual aggregation** the reduction of the number of categories will have a predictable influence on the database size and thereby on the processing time for analytical operations. It is more difficult to establish quality criteria here for setting the aggregation level so that they will answer the objectives.

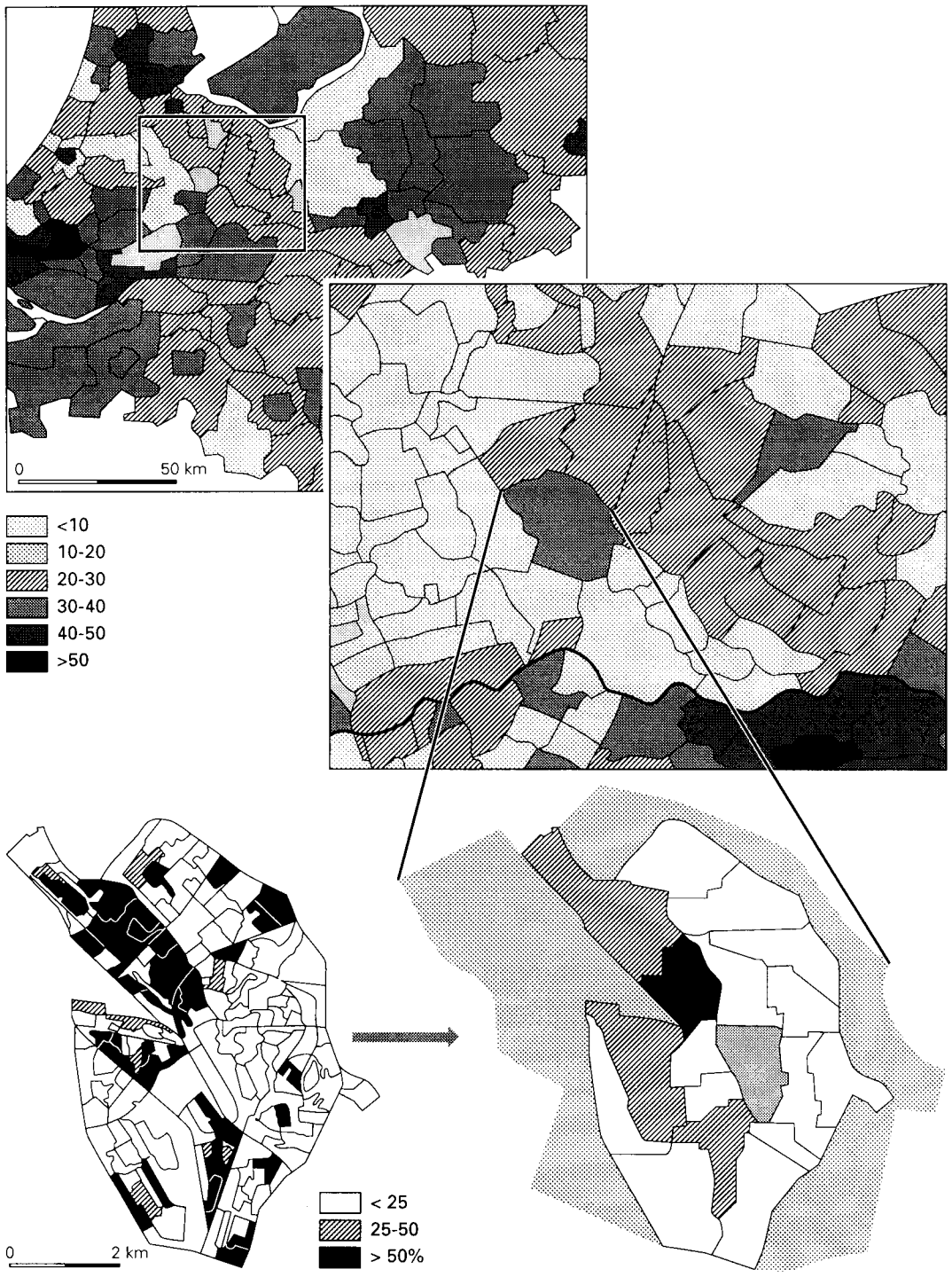


Figure 4
Changes in aggregation level in the presentation of election results for the labour party in the Central Netherlands, 1984. Representation at enumeration district level, ward level, municipality level, and economic-geographical region level.

In **geometrical aggregation** the level of the enumeration units used will determine the spatial level on which one would be able to draw conclusions from statistical data. Selecting a specific aggregation level means at the same time the selection of a specific interpretation level. Rendering election data on a polling district level will show extreme local variations that will mask regional trends. Labour party results at polling district level for Utrecht municipality (see figure 4) will show high scores for polling districts with counsel housing projects. Rendering the data at a municipality level will primarily show the differences between urban and rural areas. Again, these differences between urban and rural areas would be more outspoken than the national trends which would be masked by them, and only be visible on a provincial level.

So the selection of a specific aggregation level (local, regional, national) restricts the possibilities for interpretation of the resulting image at the same time to that level. On the basis of a rendering on a local level, no decisions with regional or national implications can be based.

It should be clear as well that the higher the level of geometrical aggregation, the smaller the differentiation in the attribute data will be. The number of persons refered to will increasingly approach a normal distribution, and the extreme values for the smallest areas will draw closer and closer to the average values.

Selecting a specific type of geometrical aggregation is another decision moment: municipal data can be aggregated in the Netherlands to larger agrarian regions ('landbouweconomische gebieden', groupings of contiguous municipalities based on common agricultural practices), to economic-geographical regions (based on similarities in the municipal economic structure) or service regions ('COROP' areas, i.e. groupings of municipalities serviced by the same large town). Selection of each of these geometrical aggregation types will result in a different regional pattern. The emergence of these various patterns generally can be attributed to an urban-rural opposition. On the basis of the agricultural groupings (which can be retraced to soil differences) the regions discerned can be grouped to larger, physically-determined groups. It would be the physical subdivision of the Netherlands which then would determine the framework within which patterns are visualised.

On the basis of economic-geographical regions (combinations of municipalities with a similar socio-economic structure, socio-economically homogeneous regions are taken as a starting point. This will boost regional differentiation, be it again within boundaries set beforehand. The aggregation into COROP regions (combinations of municipalities oriented on the same service-town) actually masks these local socio-economical differences, because urban and rural territory are united in one new region.

In aggregating grid cells the selection of the aggregation unit (into either four, nine or sixteen etc. cells), in relation to the dimensions of the topic studied , will a priori make predictable the consequences for the rendering of the topic studied.

The final aggregation aspect to be refered to here, is overlap. In order to account for the random influence of the overlayed grid or the enumeration-area subdivision (which will be seldom relevant to the topic being studied), one can apply, when aggregating, a moving average procedure which can lead to a further dimming of the extreme values that occured in the original cells or enumeration units. This can contribute to the elimination of the random influence of the orientation and the origin of the system of grid cells on the representation of the topic. As a form of iterative aggregation, it is one of the many aggregation aspects whose influence on data quality and representativity is not known yet.

references:

A.U.C.J. van Beurden and P.Padding - Areal units for environmental decision support. Proceedings Third egisconference, Paris 1994, pp 352-362.

A.U.C.J. van Beurden and A.A. van der Veer - Areal units for environmental decision support revisited. pp 292-297 in: Proceedings vol 1, First European Conference and exhibition on Geographical Information, The Hague, Netherlands, 1995.

P.A.Burrough and A.Frank - Concepts and paradigms in spatial information. International Journal of gis 1995, vol 9-2, pp 101-116.

J.P.Fonteyn - Vlakaggregatie in GISsen. MSc thesis Tilburg: Katholieke Universiteit Brabant, 1994.

M.Fridland - Patterns of the soil cover. Moscow 1972/Jerusalem 1976.

G.Hake and D.Grünreich - Kartographie. 7e Auflage. Berlin: De Gruyter 1994.

R.Knöpfli - Die Bedeutung der Ästhetik für die Übertragung von Information. International Yearbook for Cartography 1990. pp 71-81.

M.Monmonier - How to lie with maps. Chicago: Chicago University Press 1992.

M.Quick - Die Vergleichbarkeit territorialer Einheiten in der komparativen Europaforschung. pp 20-29 in: Europa Regional 2 (1994), 3.

The role of topologic and hierarchical spatial object models in database generalization

Martien Molenaar

Centre for Geo-Information Processing (CGI)
Wageningen Agricultural University, The Netherlands
E-mail: martien.molenaar@wetensch.lmk.wau.nl
Tel: +31 8370 82910 Fax: +31 8370 84643

ABSTRACT

Topological object relationships in combination with object classification hierarchies appear to be fundamental in the definition of the aggregation rules for spatial objects. Such rules are essential building blocks for the construction of generalization procedures in spatial databases. A model for spatial database generalization can be formulated based on the syntax of the Formal Data Structure (FDS) for single valued vector structured maps, as proposed in (Molenaar 1989). The syntax of the FDS will be formalised first, then database generalization procedures will be formulated with this syntax. Four strategies will be explained for generalization, these are: class driven generalization, geometry driven generalization, functional generalization and structural generalization.

An example of the latter strategy, using a drainage network, will show how database generalization procedures can be used for the representation of complex spatial landscape structures at several scale levels. The transition from a larger scale level to a smaller level can then be formalized as a set of information abstraction operations (Martinez Casasnovas 1994):
1) Selection of drainage elements and the subcatchments to be aggregated.
2) Elimination of drainage elements that are not representable at the target scale.
3) Aggregation of subcatchments that should not be represented individually any more.
4) Reclassification of the hierarchical order of the remaining drainage network.

The main structure of the drainage system is left in tact by the generalization process. Some of its semantic aspects in the context of erosion modelling will be effected however. Generalization should be seen as a database transformation, which, like any transformation, requires a specification which aspects of the information content of the database should remain invariant; other aspects of the information contents of the data base might then be affected by the transformation.

1. Introduction

A spatial database contains data that represent in principle elementary statements about some spatial situation. These elementary statements often refer to the relationships between objects and geometric data and thematic data etc. Query operations are applied to derive other statements that contain more relevant information for the user, e.g. about the state of the objects and about their mutual relationships. The semantics of the derived statement is generally of a higher level of complexity than the stored data. They should help the user to understand the structure of the mapped area,

therefore they often refer to spatial relationships between the mapped objects. If the area structure should be understood at a higher abstraction level though, these derived statements could also refer to relationships among aggregated objects. The understanding of the structure of an area at several abstraction levels is strongly related to the problem of spatial generalization and multi scale representations.

Aggregation hierarchies for spatial objects can serve as basic tools for multiple representations of geo-data within the context of conceptual generalization (information abstraction) processes. These aggregation hierarchies can be based on the formal data structure (FDS) for single valued vectormaps (Molenaar 1989), which combines aspects of object-oriented and topologic datamodels. Point-, line- and area objects are represented with their geometric and thematic aspects. Their geometric representation supports the analysis of topologic object relationships, whereas their thematic description is structured in object classes that fora generalization hierarchies. These class hierarchies together with the topologic object relationships support the definition of aggregation hierarchies of objects. The classification- and aggregation hierarchies play an important role in linking the definition of spatial objects at several scale levels. Accordingly, these structures are fundamental in the definition of rules for modelling generalization of spatial information at different resolution levels. The capacity of Geographical Information Systems (GIS) to register and handle topological information in combination with object hierarchies makes them very useful tools for the automation of conceptual generalization of spatial data.

In a cartographic context, generalization can be defined as the process of abstracting the representation of geographic information when the scale of a map is changed. It is a complex process involving abstraction of thematic as well as geometric data of objects. The process usually involves two phases:

a) a conceptual generalization phase, which implies the determination of the content of a representation in the generalized situation (information abstraction), and the definition of rules how the generalized objects can be derived from the objects at lower generalization level
b) a graphical generalization phase (cartographic generalization), which implies the application of algorithms for geometric simplification of shapes and for symbolization to assure map legibility.

Information abstraction in these subprocesses is mainly determined by expert knowledge and can be usually expressed as logical rules. These rules are susceptible to be translated as database management procedures in a GIS environment (Martinez Casasnovas 1994, Richardson 1993). Regarding information abstraction, several processes are recognized: classification, association, (class) generalization and aggregation. Class generalization and aggregation are directly related to changes in the level of definition of objects when the mapping scale changes. Aggregation is the combination of elementary objects to build composite objects and will be based on two types of rules:

a rules specifying the classes of elementary objects building a composite object and
b rules specifying the geometric characteristics (such as minimum size) and topological relationships of these elementary objects (i.e. adjacency, connectivity, proximity, etc.).

The syntactic structure of a data model for handling topologic and hierarchical relationships between spatial objects will be explained in this article. Processes for database generalization will be formulated with this data model.

A hydrologic example will be used to demonstrate how a complex terrain situation with several types of objects and their mutual topologic relationships can be handled with this grammar and how multi scale representations can be formulated.

2. Topologic Structures for the Representation of Spatial Objects

Entity Types For Spatial Data
The spatial structure of an area can be expressed in terms of point-, line- and area objects. Their spatial extend and their topologic relationships will be expressed by means of a set of geometric elements. (Frank ea 1986) showed that the geometric structure of a vector map can be described by means of cell complexes. For a two dimensional map these consist of 0-cells, 1-cells and 2-cells. The 0-cells and 1-cells play similar roles as respectively the nodes and edges when the geometry of the map is interpreted as a planar graph, the 2-cells can then be compared to the faces related to the planar graph through Eulers formula (Gersting 1993). The terminology of the planar graph interpretation will be used here, but their relationships will be formulated as those for cells according to the concepts presented in (Molenaar 1994). This formulation is then based on

- three geometric types: *nodes, edges and faces* represented by the symbols *n, e and F*
- three geometric object types: *point objects, line objects, area objects.*

The further developments will only use line- and area objects which will be represented by the symbol O_l *and* O_a. Instances of these entity types will be indicated by suffixes.

The reader will recognise that the formalization explained in this paper is to a large extent isomorphic with topologic data structures defined for GIS such as ATKIS/DLM, DGF, TIGER and DIME etc., see (Hesse 1992, Walter 1994, Marx 1990, USBUREAU 1990). This formalization will be based on the FDS described in (Molenaar 1989).

Relationships Between Nodes, Edges and Faces
The following relationships can be defined between the geometric elements of a planar graph:

- *Edge* e_i *has node* n_j *as the begin node* $\rightarrow Begin[e_i, n_j] = 1$ *otherwise* $= 0$
- *Edge* e_i *has node* n_k *as the end node* $\rightarrow End[e_i, n_k] = 1$ *otherwise* $= 0$

We will consider edges as straight line segments. Each edge will always have one face at its left hand side and on at its right hand side. These relationships will be expressed by the following functions:

- Edge e_i has face f_a at its left-hand side $\rightarrow Le[e_i, f_a] = 1$
  For any $f_b \neq f_a$ we get then $Le[e_i, f_b] = 0$
- Edge e_j has face f_a at its right-hand side $\rightarrow Ri[e_j, f_a] = 1$
  and again for $f_b \neq f_a$ we get then $Ri[e_j, f_b] = 0$

If an edge e_i has face f_r at the right hand side and face f_l at the other side then these faces are adjacent at this edge, which will be expressed by the function

$$ADJACENT[f_r, f_l \mid e_i] = 1 \text{ (and } = 0 \text{ otherwise)}$$

the fact that there is some edge where the faces are adjacent can then be expressed by

$$ADJACENT[f_r, f_l] = 1 \text{ (and } = 0 \text{ otherwise)}$$

Line Objects
The geometry of a simple line object is represented by a chain of edges as in figure 1a. The fact that an edge e_p is part of the object can be established by the function $Part_{11}[e_p, O_l]$. The notation $Part_{uv}[\ ]$ means that an entity with spatial dimension u is a part of an entity with dimension v. If the edge is part of the object then $Part_{11}[\ ] = 1$, else it has a value $= 0$.

A line object will have a begin node

n_b = $BEG(O_l)$ and an end node
n_e = $END(O_l)$. These can be found through the edges of O_l, the direction of the object can then be specified by $Dir[O_l] = \{ n_b, n_e \}$

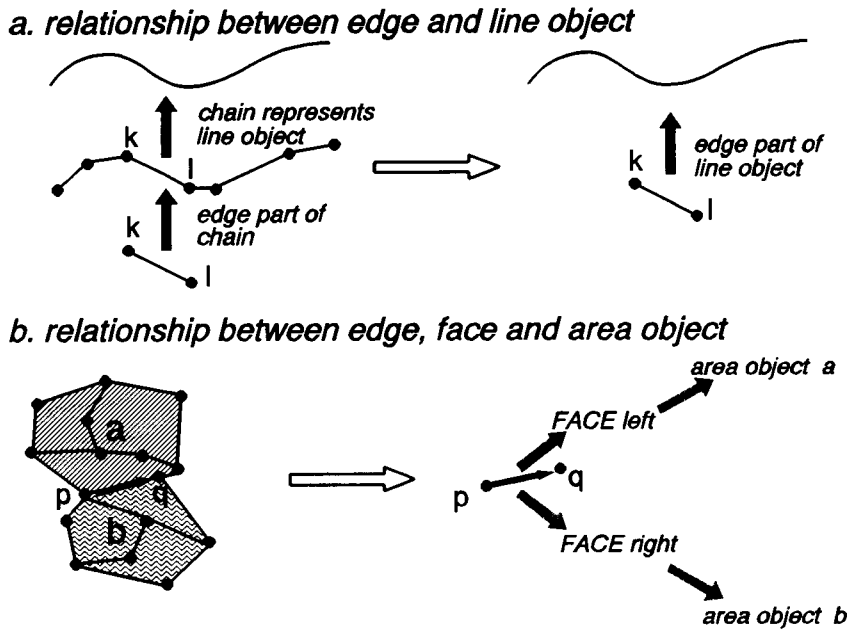


fig. 1: Relationships between edges and objects.

Area Objects
The geometry of an simple area object is represented by one or more adjacent faces as in figure 1b. If a face f_v is part of an area object O_a this will be represented by $Part_{22}[f_v, O_a] = 1$. The set of all objects from which a face f is a part is $OA(f) = \{ O \mid Part_{22}[f, O] = 1 \}$

This function relates two two-dimensional entities. If there are overlapping area objects then each face might be part of several objects, but each object will also consist of one or more faces. Therefore this is a many-to-many relationship. Overlapping objects can be found through their common faces.

Now it is possible to check whether edge e_i is related through face f_v to an area object O_a. There is at most one face for which both $Le[e_i, f_v] = 1$ and $Part_{22}[f_v, O_a] = 1$. If such a face exists then the function relating the edge to the object will get the value = 1, in all other cases it will be = 0. Hence if edge e_i has area object O_a at its left-hand side then $Le[e_i, O_a] = 1$ else = 0.

Similarly if edge e_i has area object O_a at its right-hand side then $Ri[e_i, O_a] = 1$ else = 0.
The combination of these two functions gives for edge e_i:

$$B[e_i, O_a] = Le[e_i, O_a] + Ri[e_i, O_a]$$

If an edge e_i is part of the boundary of O_a then only one of the functions Ri and Le is equal to 1 but not both, so for such an edge we find $B[e_i, O_a] = 1$. If e_i has O_a both at its left-hand side and at its right-hand side then $B[e_i, O_a] = 2$, in that case it is running through O_a. If $B[e_i, O_a] = 0$ there is no direct relationship between e_i and O_a.

Adjacent Area Objects
When an edge has an object O_a at its left hand side and not at its right hand side and object O_b at its right hand side and not at its left hand side then these objects are adjacent at this edge. If the objects overlap not at all, i.e. if they have no common faces and they are adjacent at least one edge, then they are adjacent which is expressed by the function $ADJACENT[O_a, O_b] = 1$ (and = 0 otherwise).

Line- And Area Objects
Several important relationships between a line object O_l and an area object O_a can be found by checking for each edge that is part of the line object how it is related to the area object. This will be expressed by the functions

$$Le[O_l, O_a \mid e_i] = MIN(Le[e_i, O_a], Part_{ll}[e_i, O_l])$$
$$Ri[O_l, O_a \mid e_i] = MIN(Ri[e_i, O_a], Part_{ll}[e_i, O_l])$$

For the relationship between a line object O_l and an area object O_a we can write

$$B[O_l, O_a \mid e_i] = Le[O_l, O_a \mid e_i] + Ri[O_l, O_a \mid e_i]$$

If this function has the value = 2 then the line object runs through the area object at edge e_i if the value = 1 then it is at the border and if it is = 0 then there is no relationship. The relationship between the two objects might be different at different edges.

3. A Hydrologic Example

For modelling hydrological systems three types of elementary objects will be defined according to (Martinez Casasnovas 1994), these are the water course lines, the drainage elements and their catchments, see figure 2. The drainage elements are gullies, each element has a catchment area from which it receives overland flow of water. Each element also receives water from upstream elements (if there are any) and it empties into a downstream element. The water flow through each element is represented by a water course line.

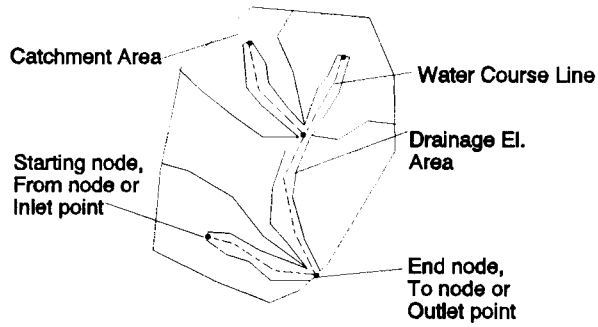


fig. 2:Elementary objects in a drainage system.

The relationship between these objects is one to one in the sense that each drainage element D_i contains exactly one water course line W_i and is embedded in exactly one subcatchment area C_i. A subcatchment area may be dissected by its drainage element, as can be seen in figure 2, but it is still considered as one subcatchment. The topologic relationships between these objects can be expressed by functions of section 2:

for water course line W_i and drainage element D_i is
$(\forall e_k \mid Part_{ll}[e_k, W_i] = 1) \Rightarrow B[W_i, D_i \mid e_k] = 2$
this will be written shortly as $B[W_i, D_i] = 2$
if $j \neq i$ *then* $B[W_i, D_j] = 0$

This means that W_i runs through D_i so that it has D_i at both sides and it is not related to any other

drainage element. This is a topologic restriction due to a semantic constraint valid in the context of this hydrologic model. Another semantic constraint is

for drainage element D_i and catchment C_i is
$ADJACENT[D_i, C_i] = 1$
if $j \neq i$ then $ADJACENT[D_i, C_j] = 0$

so that D_i is only adjacent to C_i and to no other catchment.

Each drainage element is also connected to a downstream element and, depending on its position in the network, to one or more upstream elements. The relationship between the drainage elements can also be found through the watercourse elements. These should be directed according to the direction of the water flow, for each W_i we can find the upstream element W_h through the rule $END(W_h) = BEG(W_i)$. This relation between these water course lines will be expressed by $Upstr[W_i, W_h] = 1$, this function will have the value = 0 otherwise.

Due to the 1 to 1 relationships between W, D and C the upstream relationship can be transferred as follows

$$Upstr[W_j, W_i] = Upstr[D_j, D_i] = Upstr[C_j, C_i]$$

so that the order relationships between the water course lines can be translated into order relationships between the areas in which they are contained. We will assume here that the stream network structure is defined so that for each W_j with a Strahler number > 1 there are two or more upstream water lines W_i, but for each W_i there is only one downstream water line W_j.

4. Object Classes and Class Hierarchies

Terrain objects refer to features that appear on the surface of the earth and are interpreted in a systems environment with a thematic and geometric description. In most applications the terrain objects will be grouped in several distinct classes and a list of attributes will be connected to each class. Let Ci be a class, and let the list of its attributes be $LIST(Ci) = \{A_1, A_2, \ldots, A_n\}$ then

$$LIST(Ci) \neq LIST(Cj) \text{ for } i \neq j$$

i.e. these attribute lists will be different for different classes. Terrain objects inherit the attribute structure from their class, i.e. each object has a list containing a value for each class attribute, thus for member e of class C:

$LIST(e) = \{a_1, a_2, \ldots, a_n\}$
where: $a_k = A_k(e)$ *is value of* A_k *for object e*
$e \in C$
$A_k \in LIST(C)$

When two or more classes have attributes in common, then a superclass can be defined with a list containing these common attributes as "superclass-attributes" (Molenaar 1993). The original classes are subordinated to these super classes, for example, the class 'forest' is a superclass containing subclasses such as "deciduous", "evergreen", and "mixed forest". The terrain objects are then assigned to these classes.

With these observations we find the class hierarchical structure of figure 3. In literature on semantic modelling (Brodie 1984, Brodie e.a. 1984, Egenhofer e.a. 1989, Oxborow e.a. 1989) the upward links of the classification hierarchy are labelled respectively as "ISA" links. These links relate each

particular object to a class and to super classes.

It is possible to add more hierarchical levels to the structure of figure 3. At each level the classes inherit the attribute structure of their superclass at the next higher level and propagate it normally with an extension to the next lower level. At the lowest level in the hierarchy are the terrain objects, at this level the attribute structure is not extended any more, but here the inherited attributes are evaluated. In this case we find for e:

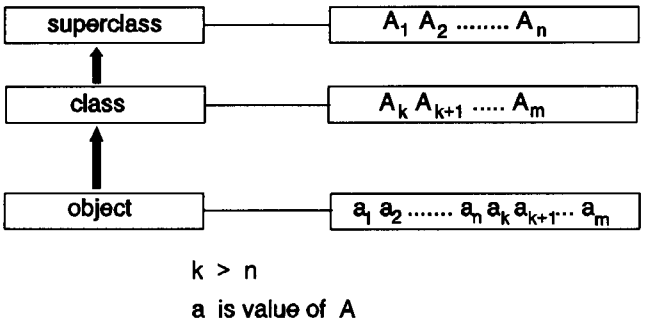


fig 3. The hierarchical relationships between objects and classes and their attributes.

LIST(e) = $\{a_1, a_2, \ldots, a_n\}$
where:
$a_l = A_l(e)$ *is value of* A_l
$A_l \in$ *LIST(C) U LIST(SC) U .....*

thus A_l is an attribute of the class *or superclass(es)* of e. If the classes at each level are disjoint so that the hierarchy has a tree structure then the terrain objects will get their attribute structure only through one inheritance line in the hierarchy, i.e. they have a unique thematic description. We will work under this assumption in this paper.

The terrain objects occur at the lowest level in the classification hierarchy. They can be seen as the elementary objects within the thematic field represented by the classification system. This implies that the decision, whether certain terrain objects should be considered as elementary or not, should always be made within the frame work of a thematic field. Objects that are considered as elementary in one thematic field are, however, not necessarily elementary in another thematic field.

5. Merge Operations

By means of merge operations new elements are composed of existing elements, after the operation the original elements cease to exist. This situation can occur both for geometric elements and for objects.

The merge operation at the level of the geometric elements will be explained by means of an example for faces. If redundancy is to be avoided in a database then it should only contain geometric elements that carry semantic information, i.e. information about the geometry of spatial objects. This means that edges should carry information about the adjacency of area objects and/or about the geometry of line objects. The adjacency information that an edge e_i carries for area objects can be found through the Le[ ] and Ri[ ] relationships for this edge. Suppose that if Le[ ] relates it to a face f_l and Ri[ ] relates it to face f_r then the edge carries adjacency information if the faces are related to different area objects so that $OA(f_l) \neq OA(f_r)$. If these sets are equal then the edge has the same objects at both sides and carries no adjacency information about area objects; in that case it might be that it relates the area objects of $OA(f_l) = OA(f_r)$ to one or more line objects through the $Part_{11}[\ ]$ relation ship, i.e. there are line objects for which this function = 1 for this edge. If however $OA(f_l) = OA(f_r)$ and there is no line object related to the edge, i.e. $\forall O_p \Rightarrow Part_{11}[e, O_p] = 0$, then the edge carries no semantic information. In that case one could decide to eliminate the edge

from the database. That implies that the two faces f_r and f_l that are adjacent at the edge so that $ADJACENT[f_r, f_l \mid e_i] = 1$, should be merged to form a new face f_m. After the merge process the original faces f_r and f_l have been eliminated. The new face f_m replaces the original faces in all relationships in which they occurred and we have $OA(f_m) = OA(f_r) = OA(f_l)$.

Objects can be merged too in the sense that several smaller objects are merged to form a larger object. An example of such a situation is when two cadastral parcels are combined to form a larger one. The original parcels cease to exist in that case and a new larger parcel is generated. This case is comparable to what has been explained for the merging of faces. The new object will replace the original objects in all relationships in which they occurred. As a consequence there may be faces of the original objects that can be merged in the geometric description of the new object.

The terminology of this section defines merging as a process where entities are put together to build a composite entity, after the process the original entities cease to exist. This quite distinct from the process in the next section where objects are aggregated to form a composite object. The semantics of the original objects is then transferred to the new object, but the original objects do not cease to exist.

6. Object Aggregation

Objects can be aggregated to build composite objects at several levels of complexity. These may form aggregation hierarchies which are quite distinct from classification hierarchies. An aggregation hierarchy shows how composite objects can be built from elementary objects and how these composite objects can be put together to build more complex objects and so on. In literature on semantic modelling (Brodie 1984; Brodie e.a.1984; Egenhofer e.a.1989; Oxborrow e.a.1989) the upward relationships of an aggregation hierarchy are called "PARTOF" links. These links relate a particular set of objects to a specific composite object and on to a specific more complex object and so on. For example, 'James Park is PARTOF Westminster is PARTOF London.'

For composite spatial objects the PARTOF links might be based on two types of rules involving the thematic and the geometric aspects of the elementary objects. Consequently the generic definition of a type of an aggregation should consist of the following rules (Molenaar 1993):

- *rules specifying the classes of the elementary objects building an aggregated object of this type,*
- *rules specifying the geometric and topologic relationships among these elementary objects.*

Suppose that aggregated objects of a type T should be formed. To do that we should first identify the objects O_i that could be part of such aggregates. These objects should fulfil certain criteria, which according to the two sets of rules given earlier will often be based on the thematic data of the objects. Let these criteria be expressed by a decision function

$D(O_i, T)$ = *1 if the object fulfils the criteria*
= *0 otherwise*

Regions can now be formed by applying two rules:

> *all objects in the region satisfy the decision function for T*
$(\forall O_i \mid O_i \in R_r) \Rightarrow D(O_i, T) = 1$

> *All objects that satisfy the decision function for T and that are adjacent to objects of the region belong to the region*
$(\forall O_i \mid D(O_i, T) = 1)(\exists O_j \in R_r \mid ADJACENT[O_i, O_j] = 1) \Rightarrow (O_i \in R_r)$

The second rule implies that a region can be formed when at least one object has been identified that fulfils the first rule. This object is then the seed around which the region can grow by identification

of the other objects that fulfil both rules.

A region R_r can be expressed as a set of objects, i.e.:

$$R_r = \{...., O_i, ...\}$$

The objects of the region can be aggregated to form an aggregated or composite object O_{ar}, the suffixes express that the object is of aggregation type a and r is its identification number. The operation will be expressed by

$$O_{ar} = AGGR(R_r) = AGGR(\{...., O_i, ...\})$$

The fact that O_i is part of O_{ar} is expressed by

$$Part_{kl}[O_i, O_{ar}] = 1$$

The reverse relation expresses that the object O_{ar} consists of the region R_r, i.e. the function identifies the object that are the components of O_{ar} :

$$COMP(O_{ar}) = R_r = \{...., O_i, ...\} = \{ O_i \mid Part_{kl}[O_i, O_{ar}] = 1\}$$

The geometry of the aggregates can be found through the geometry of the original objects, for each geometric element we can check whether it will be part of an aggregated object of type T_a. This should be done in two steps, which will be explained for the faces of an area object O_i in relation to an aggregated area object O_{ar}. The first step evaluates the function:

$$Part_{22}[f_j, O_{ar} \mid O_i] = MIN(Part_{22}[f_j, O_i], Part_{22}[O_i, O_{ar}])$$

this function expresses whether the face is related to an aggregate through object O_i. If that is true then both functions in the expression at the right hand side of the equation will have the value = 1, and this value is assigned to the function at the left hand of the equation. If it is not the case then at least one of the functions at the left hand side will have the value = 0, so that also the function at the left hand side will get the value = 0. The second step is the evaluation of

$$Part_{22}[f_j, O_{ar}] = MAX_{Oi}(Part_{22}[f_j, O_{ar} \mid O_i])$$

If there is any object through which the face will be part of an aggregate then this function will have the value = 1, otherwise it will be = 0. If this function has been evaluated for all faces of the map then the geometry of the object O_{ar} can be found through their adjacency graph. For the edges e_i of these faces the function $B[e_i, O_a]$ can be evaluated and with this function the boundary edges can be found (i.e. $B[e,O] = 1$) and through these the topologic relationships with the other objects.

The geometry of the aggregated area object O_a can sometimes be simplified by a reduction of the number of faces. Therefor the edges e_i should be identified for which $B[e_i, O_a] = 2$, that are the interior edges. If these edges are not part of some line object so that $LO(e_i) = \varnothing$ then they do not carry any semantic information at this aggregation level and could therefor be eliminated.

The example refers to the situation where a face is related through an area object to an aggregated are object, so that all involved elements are of dimension 2. Other combinations of dimensions might occur as well, this could be the case when for example an edge is related through a line object to

an aggregated area object, e.g. it is related through a river to a country.

Two cases can be distinguished with respect to the thematic aspects of the aggregated objects:
- in the first case the aggregates are defined within the same classification hierarchy and sometimes even within the same class as the original objects, this will be explained in subsection 6.1,
- in the second case the aggregates are new objects that require a completely new thematic description, i.e. the require the definition of new classes or even a new classification hierarchy, this will be explained in subsection 6.2.

6.1. Object aggregation within a classification hierarchy

Suppose that a database contains the situation A of figure 4, this is a detailed description of a terrain situation with agricultural fields, forrest areas and natural grasslands. This description might be too detailed for a structural analysis which should give information about the areas covered by the

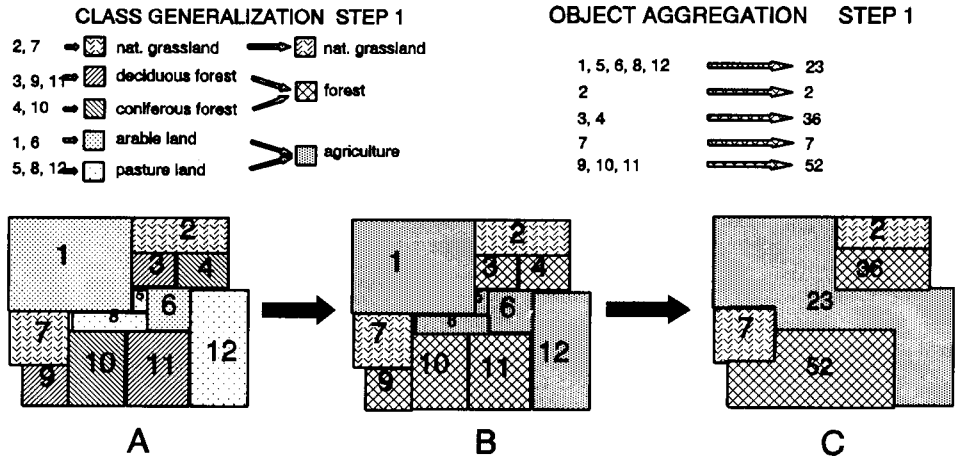


fig. 4: Object aggregation through the formation of regions per class.

different major types of land use and their spatial distribution. A less detailed spatial description can then be obtained, if the original objects area aggregated to form larger spatial regions per major land use class.

Figures 4 and 5 show that this less detailed description can be obtained in two steps:
- first the objects are assigned to more general classes representing the major land use types this results in situation B of figure 4,
- then mutually adjacent objects are combined per class to form regions, this results in situation C of figure 4.

These final regions can be considered as aggregated objects. The functions $D(O,T)$ express then that objects should be aggregated per (super)class, i.e. if aggregated objects should be formed for agriculture then $D(O,Agriculture) = 1$ if $O \in$ Agriculture else $D(O,Agriculture) = 0$.

The output of the aggregation process are regions in the sense of the previous section. Each region is an aggregate of objects that belong to one land use class, so if R_a is an agricultural region then:

- *for all objects* $O_i \in R_a$ *is* $D(O_i,Agriculture) = 1$
- *if* $O_i \in R_a$ *and* $ADJACENT[O_j, O_i] = 1$ *and* $D(O_j,Agriculture) = 1$

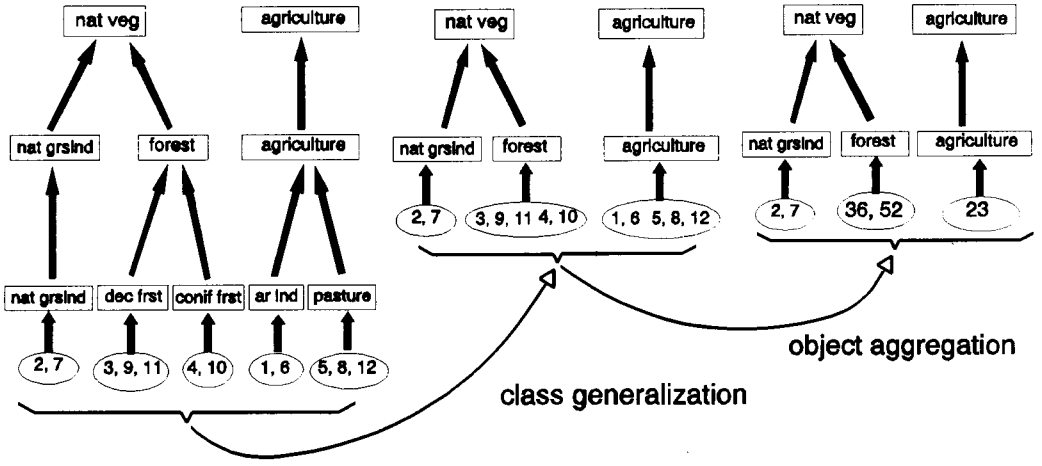


fig. 5: A diagram representing the generalization and aggregation steps of the object generalization process of figure 4.

then $O_j \in R_a$

A consequence of this rule is that after the aggregation process there can be no two adjacent regions that are of the same type, i.e. that represent the same land use class.

The output of this process could be used as the input for a following aggregation step. This has been illustrated in figure 6 where the regions of situation C are assigned to more general classes in situation D and the aggregated to form the larger regions of situation E.

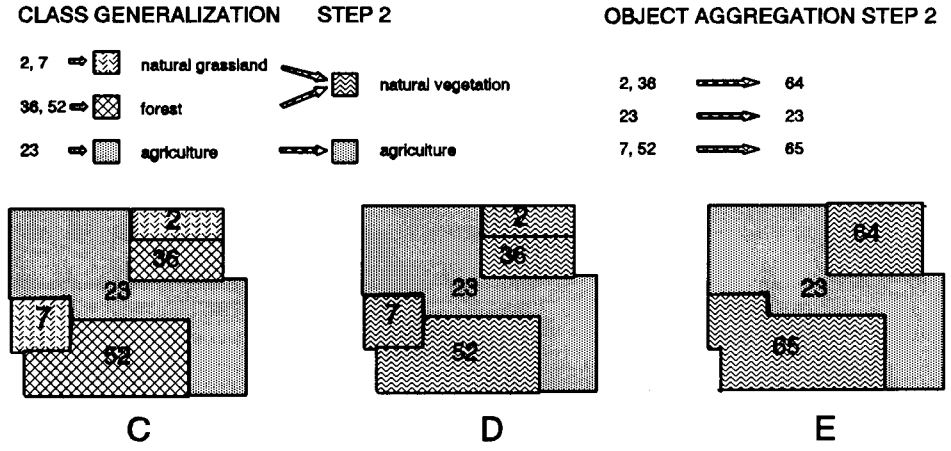


fig. 6: The second aggregation step for the objects of figure 4.C.

Thematic and geometric resolution

The examples represented in the figures 4 thru 6 show situations where the thematic aspects of the newly aggregated objects can still be handled within the original class hierarchy. It might be that the same classes can be used as for the original objects, but the example shows a situation where it is quite clear that with each database generalization step the class hierarchy is adjusted; per step the occurring lowest level of classes is removed, only the more general classes remain, see also

figures 5 and 10. That means that the thematic resolution is adjusted to the geometric resolution of the terrain description.

There might be situations where it is not necessary to jump to more general classes with each aggregation step. In those cases the new objects can be assigned to the original classes with consequence that they have the same attributes as the objects from which they have been composed. This is in fact the case if we consider the step from B to C in figure 3 in isolation. There the objects 1,5,6,8 and 12 all belong to the class "agriculture". Therefore they have the same attribute structure. They are distinct because they had different attribute values. Within this class they are aggregated to form the composite object 23, i.e.

$O_{23} = AGGR(O_1, O_5, O_6, O_8, O_{12})$.

This new object still belongs to the same class "agriculture" and has attributes in common with original objects. The attribute values of the original objects will then be transferred to the new object as in figure 7.

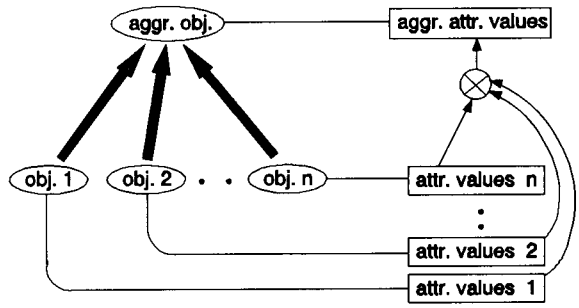


fig. 7: The aggregation of attribute values.

That will always have the effect that the spatial variability of the attribute values will be reduced, because after each aggregation step the attribute values that were assigned per object will then be merged into one value for a larger object. That means that the relationship between spatial and thematic resolution is not only expressed through the link between class level and aggregation level, it also expressed by the spatial variability of the attribute values.

When the attribute values are of the ratio scale type then the aggregated value can often be obtained by summation or by taken the average value over the objects that compose the new object, e.g.:

$A[O_a] = \Sigma_{Oi \in COMP(Oi)} A[O_i]$

Examples are attributes like wood volume and crop yield and population. For other attributes like vegetation cover or population density it might be that (weighted) averages should be computed.

6.2. Object aggregation with change of object classes

It is certainly not always so that object aggregation can be done within the framework on one class hierarchy. In many cases object aggregation will imply a completely different thematic description of the objects, so that new classes should be defined. This is illustrated in figure 8 where farm yards and fields have been aggregated into farms and these in their turn into farm districts. The aggregation hierarchy has a bottom up character in the sense that starting from the elementary objects composite objects of increasing complexity are constructed in an upward direction (in figure 8 from left to right). In figure 8 the farm districts should only consist of farms and the farms should be mutually adjacent so that the adjacency graph (see section 7) of the farms that belong to one district is connected.

It is possible to define aggregation types by means of their construction rules. (Note that these types must not be confused with the object classes of classification hierarchies.) If elementary objects

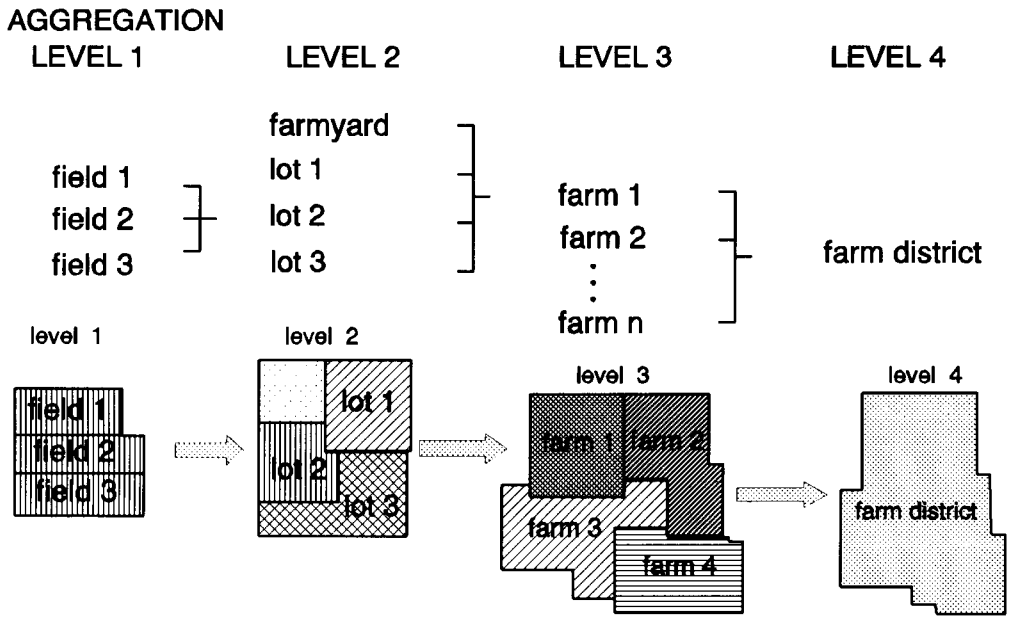


fig. 8: An example of the links between objects at different aggregation levels.

are combined to form a compound object, their attribute values are often aggregated as well as in figure 7. Farm yield is the sum of the yields per field, and the yield per district is the sum of farm yields. The desaggregation of such values is usually quite difficult because it can only be done if information is added to the system. An aggregation hierarchy has therefore a bottom-up character, in the sense that the elementary objects from the lowest level are combined to compose increasingly complex objects as one ascends in the hierarchy. The compound objects inherit the attribute values from the objects by which they are composed.

The PARTOF relations connect groups of objects with a certain aggregate and possibly on a higher level with another even more complex aggregate, and so on. That means that an aggregation hierarchy expresses the relationship between a specific aggregated object and its constituent parts at different levels. This is different from class hierarchies where classes at several generalization levels can be defined with their attribute structured and their intentions, but where the objects can be assigned to these classes in a later stage of a mapping process.

The aggregation steps in figure 8 show how the fields are considered as elementary objects at level 1. They are defined per growing season as spatial units under one crop. For the farmer they are management units, because his management operations are planned and performed per field. They are aggregated to lots which are elementary objects at level 2, i.e. these objects belong to the extensions of classes such as "arable-lot" and "grass land". These are management units at a higher level; the farmer will maintain a drainage system per lot and he will decide per growing season how to partition each lot into fields. These lots might both belong to a superclass "farm lots" in a land use data base and these again might belong to an even higher superclass "lot" which also contains the classes "forest lot" and "residence lot". The aggregation step from level 1 to 2 and the next steps to the levels 3 and 4 where we have the farms and farm districts show that after each step new objects are created. At farm level the farmer will decide whether he will be a cattle farmer or whether he will grow arable crops, in the latter case he has to decide on a rotation scheme. At district level the infrastructure and irrigation schemes will be developed. The objects at each level have their

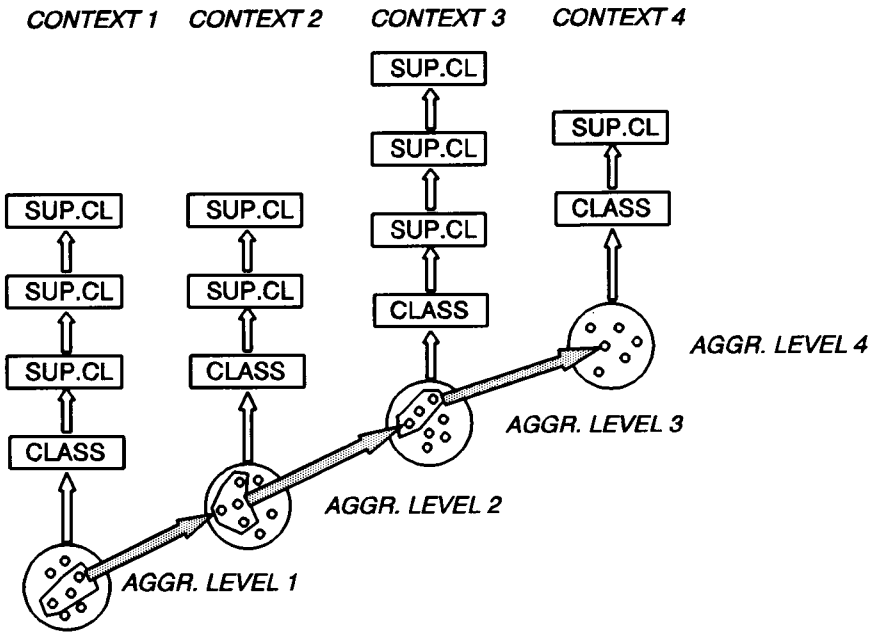


fig. 9: Classification hierarchies related to aggregation levels.

own thematic description expressed in an attribute structure that should be defined in a class hierarchy according to section 2. In this example each aggregation level requires its own classification hierarchy. This should be structured so that the generated attribute structures provide the information to support the management operations defined at the aggregation levels of the objects. The diagram of figure 9 represents the fact that a classification hierarchy should be defined per aggregation level.

This is an example of a more general situation where objects at each aggregation level are functional units with respect to some process. In this case these were farm management processes, but we could also take examples like ecologic development, or demography and many more. Each aggregation level within such a hierarchy will have its own (sub) context within a thematic field, expressed through a class hierarchy with related attribute structures. The different (sub) contexts are related by the fact that sets of objects at one level can be aggregated to form the objects at the next higher level. There are often also relationships between various classes of the different class hierarchies related to the aggregation levels as was the case for the cover classes for the farm lots, the farm types and land use types of the farm districts at the levels 2,3 and 4 of figure 8.

There are bottom-up relationships between the objects at different levels in the sense that the state information of the lowest level objects, as contained in the attribute data, can be transferred through a process like figure 7, to give state information about the objects at higher levels. There are top-down relationships in the sense that the behaviour of lower level objects will be constrained by the information contained in the higher level objects.

7. Object Hierarchies and Database Generalization

Chapter 6. explained the concept of spatial object aggregation and its relationship to class hierarchies. In the process of object aggregation the information of lower level objects is aggregated to higher level objects, but in principle the original detailed information is maintained so that it is possible to access the detailed information of the lower level objects through the aggregated objects. The

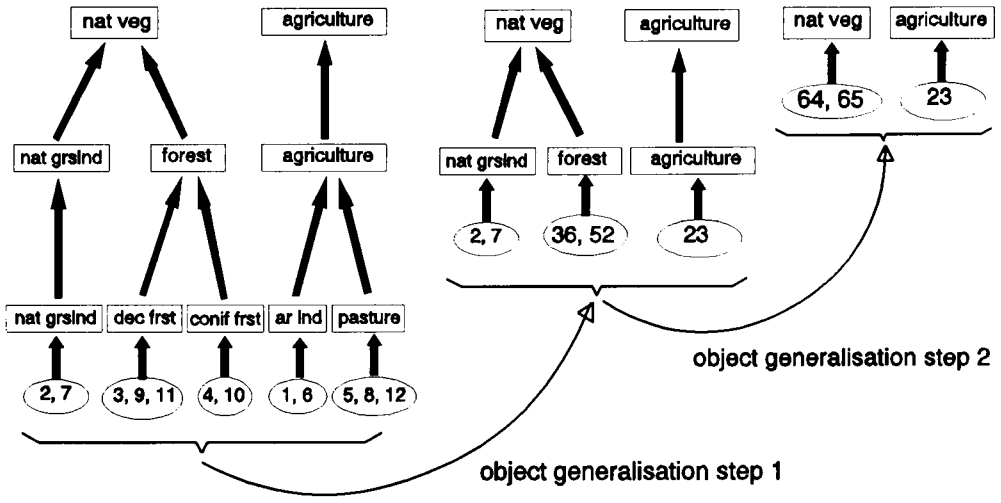


fig. 10: A diagram representing the object aggregation steps of figures 4 and 6.

result of such an aggregation process is a less detailed terrain description that may be compared to the result of a map generalization. The class generalization and object aggregation steps of the approach of figures 4 and 6 have been represented in a different way in figure 10. This figure combines per database generalization step two steps like those of figure 4. In the first step of figure 10 the objects of the different classes are first assigned to the super classes at the next higher level in the hierarchy (compare the class generalization step of fig 4), then in the same step the objects that form a region per super class are aggregated to form a larger object (compare the object aggregation step of fig 4). This procedure is repeated in the second step of figure 10.

This figure shows that the two steps of the example of section 6 reduce the number of objects, that is why we rather talk of database generalization because the process generated objects with a lower spatial and thematic resolution then the original objects. Due to the fact that the original objects formed a geometric partition of the mapped area and due to the fact that generalization process made use of the topologic and hierarchical structures in which the objects had been modelled, this process resulted in a new set of objects that also formed a geometric partition of the mapped space. But the result was a terrain description of a reduced spatial complexity as is shown in the stepwise reduction of the complexity of the adjacency graphs of figure 11. Each object is represented by a node in these graphs and the adjacency between two objects is represented by an arc. This figure gives the adjacency graphs related to each stage of a process that starts from the situation B of figure 4 where the original objects have been assigned to their super classes. If we follow the steps of fig. 10 then we see that:

- in step a the regions per class have been identified,
- in step b the objects in each region are aggregated to form a composite object that is represented by one node,
- these regions are after step c assigned to more general classes,
- in step d regions at this higher class level have been identified, these are composed of the objects obtained after step b,
- then finally after step e each of these regions have been aggregated again to form the objects

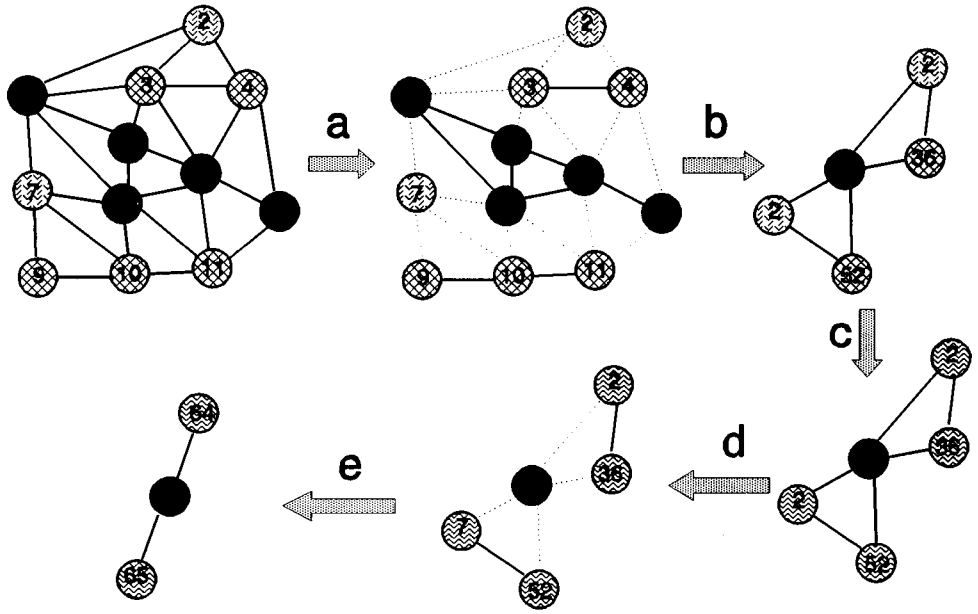


fig. 11: *The adjacency graphs related to different stages of the generalization processes of figures 4 and 6.*

at the higher aggregation level which is then represented by one node, this is the adjacency graph of situation E of figure 6.

The reduction of spatial complexity is one of the important aspects of generalization processes as they are known in mapping disciplines. This process has traditionally been applied in the form of map generalization to reduce the information content of a map so that a mapped area could be represented at a smaller map-scale. This process has two steps, the conceptual generalization and the graphic generalization. The conceptual generalization results in a redefinition of the mapped spatial features or objects to reduce their number for the terrain description at the smaller scale. The graphical generalization is in fact a simplification of the graphical representation of these features or objects, including such aspects as geometric simplification, object displacement, resymbolization etc.

When we deal with spatial database generalization in a GIS environment then this might include the graphical representation as well, but that is not necessarily so. The main aim will be a simplified terrain description, i.e. a lower spatial complexity to emphasize spatial patterns and relationships that might be difficult to find in a more detailed terrain description. That means that this process is very much related to the conceptual generalization step mentioned before, the main aim of this step is to obtain a data reduction. We will see now that it can to a large extend be defined in the form of database operations for databases that are implementations of the formal data structure (FDS) that has been explained in the previous chapters. Such databases will be called shortly FDS-databases.

The process of section 6.1 merely aggregated objects per land use region, all objects in a region were aggregated into a larger object. In many applications it is required though that some of the original objects are maintained in the generalised terrain description, a reduction of spatial complexity requires the selection of objects that are to be maintained and the elimination of those that have not been selected. Therefore some rank order of the objects must be generated so that the most

significant objects can be selected, i.e. most significant with respect to some criterion defined by the process.

8. Rank Order Selection of Objects

Figure 12.a shows a situation where there are areas with arable and pasture land, furthermore there are deciduous or coniferous forests and natural grasslands. These are subclasses of the super classes "agriculture" and "forest". For a representation of this area at a smaller scale the data could be reduced when only the super classes are represented. This step is shown in the situation of figure 12.b. where all objects have been assigned directly to their super classes.
A second step in data reduction for representation at a lower resolution or scale level could be the removal of objects that are not sufficiently significant, i.e. objects that are too small should not be represented at a smaller scale level. For those objects there are several options. One possibility is that they are ignored completely, this could be done for isolated small objects. For clusters of mutually adjacent small objects we could decide to aggregate them to form a larger object. The result of such a process is given in figure 12.b, there we observe three types of situations:

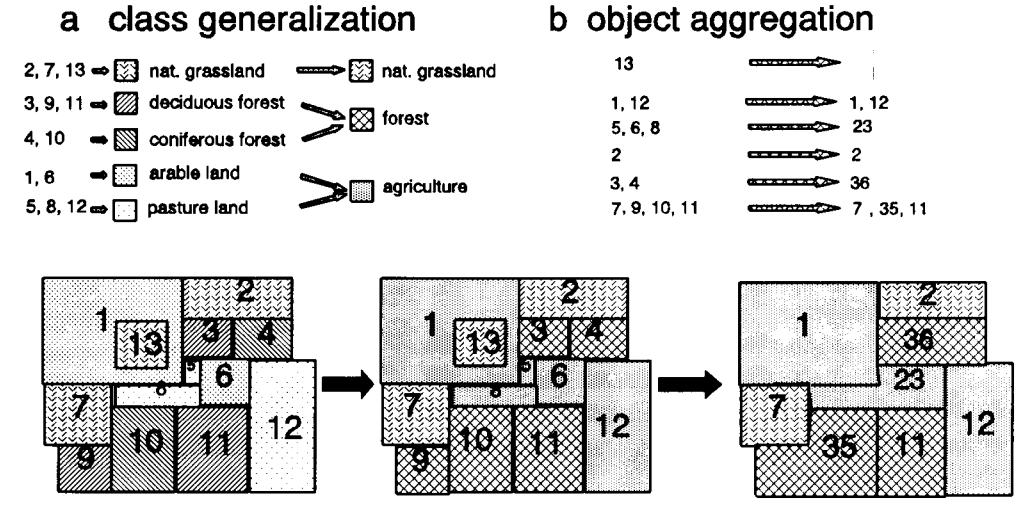


fig. 12: Two steps for the generalization of spatial objects after rankorder selection by size.

- objects 5, 6 and 8 are aggregated into object 23 and the objects 3 and 4 into object 36,
- object 9 was too small to be represented after generalization and has therefor been combined with object which was large enough, together they form object 35,
- object 13 was too small to be maintained, it was an island inside object 1 and could therefor not be combined with other objects of its class. It has been dissolved with the effect that the new object 1 is now polluted.

It has been stated earlier that two types of rules can be used to define aggregates, i.e. rules referring to the thematic content of the aggregated object and rules referring to the geometric or topologic relationships among the constituting elementary objects. In this case the thematic rules expressed in the function $D(O,T)$ specify the (sub) classes and the size of the objects that should be aggregated. The topologic rules for the aggregation operations specify that the elementary objects building the aggregate should be a connected set, i.e. the adjacency graph of these objects should be connected.

The selection criterion in this example was simply formulated by means of a cut off value for one of the object attributes, in this case the size of the objects. It is also possible that a certain percentage

of the objects should be selected and that the selection should be based on the importance of the objects expressed through the values of one of their attributes. This importance could refer to their size, but also to the biomass of vegetated areas or the number of inhabitants of populated places etc. For such a selection some rank order should be defined for the objects.

Suppose that a selection of area objects is required for the generalization of a map M. If there is a hierarchical classification structure, a rank order can be defined for all area objects by means of the values of an attribute A defined superclass SC_h at the highest level of the hierarchy (Richardson 1993). The selection of the objects can be done according to the following process:

RANK ORDER SELECTION PROCESS 1 (ROSP1):
>Select $A_p \in LIST(SC_h)$
>Assign a rank order number **a** *to the area objects of* SC_h *so that*
$(\forall O_a \in SC_h) \Rightarrow A_p[O_a] > A_p[O_{a+1}]$
>set threshold N (with $0 \leq N \leq 1$*)*
>select the objects O_a *for which*
$a \leq n$ *where* $(n/a_{max}) = N$

The threshold N specifies the fraction of the objects to be selected for representation at a smaller scale, in % this fraction is Nx100%. The objects which are not selected will consequently not be represented at the smaller scale. The parameters of the rank order selection procedure are the (super-) class, the attribute and the cut-off value that are used to specify the objects to be selected. The output of this process is

$$ROSP1(SC_h, A_p, N) = \{O_a \mid a \in \{1,\ldots,n\}\}$$

The ROSP1 selects a certain percentage of the objects belonging to a superclass. This process can be amended so that the first t objects are selected for which the cumulative value of the attribute A_p meets a criterion T, e.g.:

ROSP1a:
>Select $A_p \in LIST(SC_h)$
>Assign a rank order number **a** *to the area objects of* SC_h *so that*
$(\forall O_a \in SC_h) \Rightarrow A_p[O_a] > A_p[O_{a+1}]$
>set threshold T
>select the objects O_a *for which* $1 \leq a \leq t$ *so that*
$\Sigma_{a \in \{1,\ldots,t\}} A[O_a] \leq T$ *and* $\Sigma_{a \in \{1,\ldots,t+1\}} A[O_a] > T$

The result of this process is

$$ROSP1a(SC_h, A_p, T) = \{O_a \mid a \in \{1,\ldots,t\}\}$$

For the definition of ROSP1 and ROSP1a we assumed that SC_h was the superclass at the top of the hierarchy so that all area objects of U_M belong to this class. Quite often this is not the case, then we have to deal with parallel classes which form a thematic partition of the set of mapped objects. Rank order selection of objects can then be made in two steps (Richardson 1993). In this process the relative importance of each class is observed and depending on the nature of the data, given a priority or weighting, i.e. a rank order number i is assigned to the classes of the partition P under consideration. A rank order number j is then assigned to the objects per class according to the value of some super class attribute. A process of this type will be as follows:

RANK ORDER SELECTION PROCESS 2:
>Define a measure of class importance I[C]
>Assign a rank order number i to the classes so that
$(\forall C_i \in P) \Rightarrow I[C_i] < I[C_{i+1}]$
>Set a threshold n_p *and*
>select the classes C_i *with* $i > n_p$
>for each $C_{i \leq np}$ *do ROSP1*(C_i, A_p, N) *or ROSP1a*(C_i, A_p, T)

In this process the importance of the classes increases according to their rank order. The classes with a sufficient high rank order are selected for representation in the reduced data set, i.e. all objects of these classes will be kept in the database. For the classes that have not been selected to be represented completely, one could decide to'test their objects individually by a ROSP1 or a ROSP1a, so that only the objects of these classes that are sufficiently significant will be kept. But one could also decide the reject the complete class.

If a thematic partition consists of e.g. five classes, perhaps only three will be of significance to the user and the remainder can be eliminated. In a wetland classification structure for example, five classes occur in the Canadian scheme; these are bog, fen, swamp, marsh, and shallow open water. If these classes are required in the context of fuel analysis, classes may be ranked with fens as the most potential for production and bogs as the least. For generalized representations, the class order provides immediate access according to the context. Similarly if populated places in a superclass are organized by classes, such as cities, towns and then villages, this provides a general view. However, if the application were on a representation of unemployment statistics, this abstraction may not be appropriate since it abstracts first for example, major centres, such as city, town and village, and then orders unemployment statistics according to the class or major centre in which they occur. In this instance, superclass abstraction would be more appropriate since it would provide those centres with highest unemployment figures of their class.

9. An Example: The Generalization of Catchment Areas

Our example will be based on a database where the spatial description of a drainage system has been structured according the FDS as in section 3, see (Martinez Casasnovas 1994). Besides the geometric information the database also contains thematic data as described for the object classes. Let the attribute ORDER contain the Strahler order number of each drainage element. These numbers in combination with the function Upstr[ ] make it possible to analyze the stream network built by the drainage elements. Through this network aggregation steps can be defined for the catchment areas. The methodology for these aggregation steps will follow to a great extend procedures defined by (Richardson 1993 and 1994).

The database of our example has been designed for a terrain representation at scale = 1:50.000 and should be reduced to a target scale = 1:100.000. The process starts from the identification of the drainage elements that are not mappable at the target scale, those are the elements with Strahler number = 1 with an average width $aw < Thr(eshold)$. The minimum mapping width of the drainage elements will be put at 0.75mm, that gives a threshold $Thr = 0.75mm/scale$ at terrain scale, hence $Thr = 75m$ in this case. The average width for an drainage element D_i can be computed from the $AREA_i$ of the element and the $LENGTH_i$ of its water line W_i hence
$aw_i = AREA_i / LENGTH_i$

The ROSP2 applied to the drainage elements is then
> *select the drainage elements* D_h *with* $ORDER_h > 1$
> *select from the class with* $D_h = 1$ *the elements* D_i *with* $aw_i \geq Thr$

The set of elements that should be eliminated is then
$S = \{D_i \mid ORDER_i = 1,\ aw_i < Thr\}$,
their catchments should be combined with adjacent catchments to form aggregates.

The elimination of the drainage elements $D_i \in S$ should consist of the following steps

> *eliminate* W_i
> $AGGR(D_i, C_i) = C_{DHi}$
> *find* C_h *for which* $Upstr[C_h, C_i] = 1$
> $AGGR(C_{DHi}, C_h) = C_{h;i}$

where the notation C_{DHi} means that the area of Di has been merged into the area of its subcatchment, the notation $C_{h;i}$ means that the area of C_{DHi} has been merged into the area of C_i. When water line W_i joins the outlet point $END(W_i)$ with only one water line W_j of a drainage element that will not be eliminated then the catchment of W_j should be merged with $C_{h;i}$

> *find* C_j *for which* $Upstr[C_h, C_j] = 1$
if $D_j \notin S$
and *there is no* $D_{k \neq j} \notin S$ *with* $Upstr[C_h, C_k] = 1$
then
> $AGGR(D_j, D_h) = D_{jh}$
> $AGGR(C_j, C_{h;i}) = C_{jh;i}$

The notation D_{jh} means that D_j and D_h have been merged and $C_{jh;i}$ means that C_j and $C_{h;i}$ have been merged. When these steps have been done for each element $D_i \in S$, then new Strahler numbers can be assigned to the remaining elements according to their new position in the network. Then the selection procedure can be repeated and so on until no more elements are eliminated.

A Test Case
The drainage system represented in figure 13.a will be used as an example to demonstrate the generalization process. This figure is a schematic representation of the Romani Drainage system in the Anoia-Penedes Area in NE-Spain (Martinez Casasnovas 1994). The total area of this drainage system is 28.53 km^2, at a 1:50.000 scale it has 37 mappable drainage elements. The figure only shows the water lines W_i with their catchments, the drainage element D_i are not shown here.

The transition from the original scale to the scale 1:100.000 will be done through the generalization procedure explained before. So first the drainage elements with Strahler number = 1 and width < 75m are eliminated, their catchments are aggregated with their downstream catchments. Then the remaining drainage elements are reclassified and the procedure is repeated until no more elements are eliminated.

The result of this procedure is shown in figure 13.b. The drainage system represented at 1:100.000 scale has only nine mappable drainage elements. Their catchments are aggregates of the catchments shown at the 1:50.000 scale. The fact that they are considered to be aggregated catchments implies that the information carried by the original catchments is now transferred to these aggregates.

The Romani drainage system has been mapped for an erosion survey. Erosion classes are estimated per catchment from the information contained in the attributes of the drainage elements. These are used to compute per catchment the drainage density in km/km^2 and the crenellation ratio in km/km^2 These data combined with the depth and the activity class of the drainage elements determine the

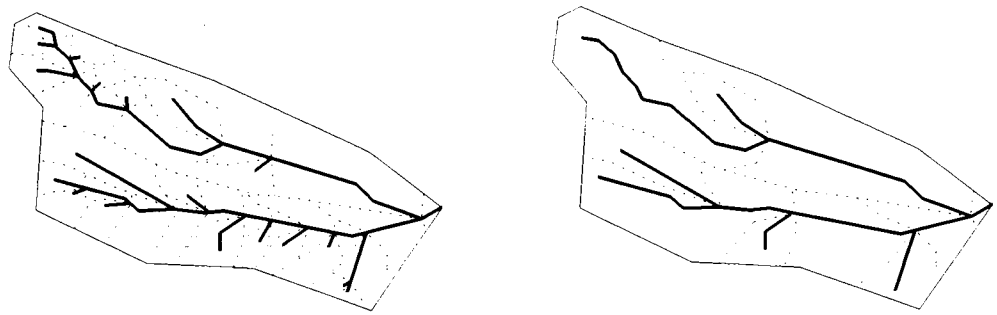


fig. 13: *a) The drainage system at 1:50.000 scale representation*
b) The drainage sustem at 1:100.000 scale representation

erosion class of each catchment. When the area of the catchments are summed per erosion class we find in the original situation at 1:50.000 scale that 68% of the area is slightly eroded, 30% is moderately eroded and about 2% shows severe erosion, see figure 14.a.

After generalization aggregated catchments are formed for which the erosion classes have to be estimated. Although a large number of drainage elements have not been represented any more at the reduced scale, the information they carry has been transferred to the aggregated catchments.

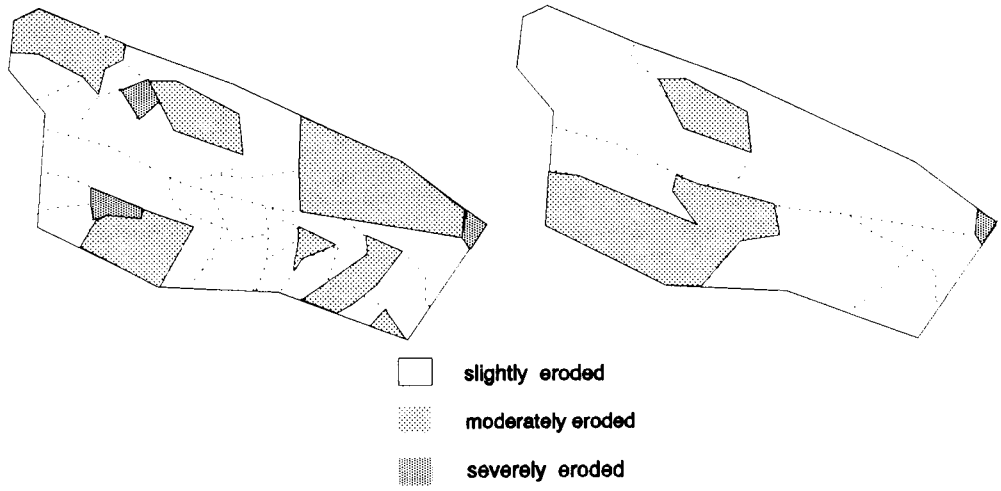


fig. 14: *a) Erosion classes estimated in 1:50.000 scale representation*
b) Erosion classes estimated in 1:100.000 scale representation

With these data we find now for the erosion classes that 79,5% of the area is slightly eroded, 20% is moderately eroded and 0.5% is severely eroded, see figure 14.b. These numbers deviate significantly from the original values, furthermore the spatial distribution of the occurrence of the erosion classes is quite different from the original distribution. The case is even worse if we had completely ignored the information carried by the eliminated drainage elements. Then the values would be respectively 99.5%, 0% and 0.5%.

10. Conclusion

A first observation to be made in this concluding part refers to the role of the developed formalism. The formalism helps us to express the structure of spatial datasets. This can be done in an abstracted sense, i.e. without any reference to the logic model of any implemented spatial data base. Processes applied to such datasets could also be expressed through this formalism. We took a generalization process as an example. The four basic operations used in the generalization process were:

- the *selection* of objects to be represented at the reduced scale, this selection was based on the attribute data of the objects,
- the *elimination* from the data base of objects that should not be represented,
- the *aggregation* of area objects such as drainage elements and subcatchments,
- the *reclassification* of the remaining drainage elements.

For these latter three operations information about the spatial structure of the mapped area was required. Firstly to check which relationships the eliminated objects had with their environment and what the effect of their elimination would be on the spatial structure of that environment. Secondly this information was required to formulate aggregation rules for the objects that were to be merged. Once the process has been formulated one can choose how to implement it in any environment that is suitable. The hydrologic example of this paper has been implemented in an Arc\Info environment, but other students of the author have made implementations of similar applications in an Oracle data base, and exercises with Prolog have been made as well.

A second observation concerns the strategies for data base generalization. Three strategies have been shown in this article, a fourth can be identified also. These strategies are:

- **class driven generalization**: this is the strategy explained in section 6.1. where regions were identified, consisting of mutually adjacent objects belonging to the same class. These objects were then aggregated to form larger spatial units with uniform thematic characteristics. The generalization is then driven by the thematic information of the spatial data.
- **geometry driven generalization**: this is the counterpart of the first strategy, it has not been dealt with in this article. In this strategy it is the geometric information that drives the aggregation process. A clear example of this case is when the geometry of the spatial data has a raster structure. If it is then decided that the resolution of the raster will be decreased, i.e. when the cell size increases, then the original, smaller, cells are merged into new larger cells. The thematic information carried by the original cells should then be transferred to the new cell according to a process like figure 7.
- **functional generalization**: the process described in section 6.2 is an example of this case. Spatial objects at a low aggregation level are aggregated to form new objects at a higher level. The objects are functional units with respect to some process defined at their aggregation level, the processes at the different aggregation levels are related.
- **structural generalization**: the generalization process of the drainage network in section 9 is an example of this case. The main aim of the process was to simplify the description of the network, while keeping the overal structure intact. This to the fact that after generalization the total functioning of the same system can be understood at a less detailed level. Other examples can be found for transport networks, etc.

Each strategy has its own range of applications. Data base users should be well aware of why they are generalizing spatial data, so that they can chose which strategy is to be followed.

This brings us to a third observation, which concerns the semantic effects of the generalization process. The data base in our example contained information about the hydrologic structure of a drainage system, let us call that the primary interpretation of the data base. The database contained also

information about the area covered by certain erosion classes, let us call that the secondary interpretation of the data base. The generalization process to derive the 1:100.000 representation from the 1:50.000 representation emphasized the primary interpretation. This is in fact what we want to do when generalizing spatial data bases, we reduce complexity to stress spatial structure. Here the spatial structure referred to the network structure of the drainage system in relation to the subcatchments. The generalization process kept the area of the aggregated subcatchments and the network structure of the system invariant so that the computation of overland water flows per node in the network will not be effected significantly.

But the emphasis at the primary interpretation was at the cost of the secondary interpretation, i.e. the generalized network could not be used to formulate reliable statements about the erosion classes of the areas in the system. Reliable statements about erosion at different scale levels require another generalization process where we have to specify what statements about erosion should be invariant after transformation. A class driven or a geometry driven strategy might have been more useful in this case. This third observation can be interpreted in the more general sense. If we consider generalization operations as a type of transformation of a spatial data base, then we should make explicit decisions about which aspects of the original data bases are to remain invariant, so we should decide what is to be considered as the primary interpretation of the data base. This choice will be made within some users context of the data base, i.e. the user will be interested in the correct representation of some spatial characteristics, while others may be deformed by the transformation.

REFERENCES

Brodie, M.L. (1984): *On the Development of Data Models*. in: On Conceptual Modelling (eds. Brodie, Mylopoulus, Schmidt). Springer Verlag, New York.

Brodie, M.L. & D. Ridjanovis (1984): *On the Design and Specification of Database Transactions*. in: On Conceptual Modelling (eds. Brodie, Mylopoulos, Schmidt), Springer-Verlag, New York.

Egenhofer, M.J. & A.U. Frank (1989): *Object-Oriented Modelling in GIS: Inheritance and propagation*. Auto-Carto 9, p.588.

Frank, A.U. and W. Kuhn (1986): *Cell Graphs: A Provable Correct Method for the Storage of Geometry*. Proceedings of the 2nd International Symposium on Spatial Data Handling, Seatle.

Gersting, J. L. (1992): *Mathematical structures for computer science*. Computer Science Press, New York, Third edition.

Hesse, W. and F.J. Leahy (1992): *Authorative Topographic-Cartographic Information System ATKIS*. Landes Vermessungamt Nordrhein-Wetsfalen, Bonn, 22pp.

Marx, R.W. (1990): *The TIGER system: automating the geographic structure of the United States census*. In: Introductory readings in GIS, (Peuquet, D.J. and D.F. Marble eds.), Taylor and Francis, London, pp120-141.

Martinez Casasnovas, J.A. (1994): *Hydrgraphic Information Abstraction for Erosion Modelling at Regional Level*. MSc. thesis, Dept of Landsurveying and Remote Sensing, Wageningen Agricultural University, Wageningen, 92pp.

Molenaar, M. (1989): *Single valued vector maps - a concept in GIS*. Geo-Informationssysteme, vol.2 no.1, pp18-26.

Molenaar, M. (1993): *Object Hierarchies and Uncertainty in GIS or Why is Standardization so Difficult*. Geo-Informations-Systeme, vol.6, No.3.

Molenaar, M. (1994): *A syntax for the representation of fuzzy spatial objects*. In: Advanced Geographic Data Modelling, Molenaar, M. and S. de Hoop eds., Netherlands Geodetic Commission, New Series, Nr.40, Delft, pp155-169.

Oxborrow, E. & Z. KEMP (1989): *An Object-Oriented Approach to the Management of Geographical data*. Conference on Managing Geographical Data and Databases, Lancaster.

Richardson, D.E. (1993): *Automatic Spatial and Thematic Generalization Using a Context Transformation Model*. (Doctoral Dissertation, Wageningen Agricultural University) R&B Publications, Ottawa, Canada.

Richardson, D.E. (1994): *Contextual Transformations and Genralizations of Remotely Sensed Imagery for Map Generation*. In: Advanced Geographic Data Modelling, Molenaar, M. and S. de Hoop eds., Netherlands Geodetic Commission, New Series, Nr. 40, Delft, pp170-178.

U.S.BUREAU OF THE CENSUS (1990): *Technical description of the DIME System*. In: Introductory readings in GIS, (Peuquet, D.J. and D.F. Marble eds.), Taylor and Francis, London, pp100-111.

Applying reactive data structures in an interactive multi-scale GIS

Peter van Oosterom and Vincent Schenkelaars

Cadastre Netherlands, Company Staff
P.O. Box 9046, 7300 GH Apledoorn
The Netherlands
E-mail: oosterom@kadaster.nl
Tel: +31 55 528 5163, Fax: +31 55 355 7931

TNO Physics and Electronics Laboratory
P.O. Box 96864, 2509 JG The Hague
The Netherlands
E-mail: schenkelaar@fel.tno.nl

Abstract

This paper presents the development of the first Geographic Information System that may be used to manipulate a single dataset at a *very large range of scales* (different detail levels). The design of this multi-scale GIS is fully integrated in the *open DBMS Postgres* and the *open GIS GEO++*. Besides the system design, this paper will also give details of the implementation in the Postgres DBMS environment of three generalization tools: 1. the *BLG-tree* for line and area simplification, 2. the *Reactive-tree* for selection based on importance and location, and 3. the *GAP-tree* for solving problems when using the other two structures for an area partitioning. Together with the geographic frontend, the DBMS forms the flexible basis for the realization of powerful GIS applications. The implementation has been successfully benchmarked with two large datasets: *World Databank II* and *DLMS DFAD*. The response times improve with one to two orders of magnitude.

1 Introduction

The advent of Geographic Information Systems (GISs) has changed the way people use maps or geographic data. The modern information systems enable browsing through geographic data sets by selecting (types of) objects and displaying them at different scales. However, just enlarging the objects when the user zooms in, will result in a poor map. Not only should the objects be enlarged, but they should also be displayed with more detail (because of the higher resolution), and also less significant objects should be displayed.

A simple solution is to store the map at different scales (or level of details). This would introduce redundancy with all related drawbacks: possible inconsistency and increased memory usage. Therefore, the geographic data should be stored in an integrated manner without redundancy, and if required, supported by a special data structure.

Detail levels are closely related to cartographic map generalization techniques. The concept of *on-the-fly* map generalization is very different from the implementation approaches described in the paper of Müller et al.: *batch* and *interactive* generalization (Muller, Weibel, Lagrange, & Salge, 1993). The term batch generalization is used for the process in which a computer gets an input data set and returns an output data set using algorithms, rules, or

constraints (Lagrange, Ruas, & Bender, 1993) without the intervention of humans. This in contrast to interactive generalization ("amplified intelligence") in which the user interacts with the computer to produce a generalized map.

On-the-fly map generalization does not produce a second data set, as this would introduce *redundant* data. It tries to create a temporary generalization, e.g., to be displayed on the screen, from one detailed geographic database. The quick responses, required by the interactive users of a GIS, demand the application of specific data structures, because otherwise the generalization would be too slow for reasonable data sets. Besides being suited for map generalization, these data structures must also provide spatial properties; e.g., it must be possible to find all objects within a specified region efficiently. The name of these types of data structures is *reactive data structures* (van Oosterom, 1989, 1991, 1994).

The emphasis is put on the generalization techniques *simplification* and *selection*. The core of the reactive data structure is the "Reactive-tree" (van Oosterom, 1991), a spatial index structure, that also takes care of the selection part of the generalization. The simplification part of the generalization process is supported by the "Binary Line Generalization-tree" (van Oosterom & van den Bos, 1989). When using the Reactive-tree and the BLG-tree for the generalization of an area partitioning, some problems are encountered: gaps may be introduced by omitting small features and mismatches may occur as a result of independent simplification of common boundaries. These problems can be solved by additionally using the "GAP-tree".

The Postgres DBMS (Stonebraker, Rowe, & Hirohama, 1990) is extended with these reactive data structures. Postgres, the successor of Ingres, is a research project directed by Michael Stonebraker at the University of California, Berkeley. The characteristic new concepts in Postgres are: support for complex objects, inheritance, user extensibility (with new data types, operators and access methods), versions of relations, and support for rules. In particular, the extensibility of Postgres is used for the implementation of the reactive data structures. The Postgres reference manual (Postgres Research Group, 1991) contains all the information required to use the system.

Short descriptions of the principles of the BLG-tree and the Reactive-tree are given in Section 2. The GAP-tree is described in Section 3. Postgres related implementation aspects are described in Section 4. The implementation is tested and evaluated with two large data sets. The performance improvements achieved by using the new reactive data structures can be found in Section 5. In the next section, an enhanced version of the Reactive-tree is presented: the *self-adjusting Reactive-tree*. This new version is better suited to deal with all kinds of object distributions among the detail levels. Finally, conclusions are given in Section 7 along with some indications of future work.

2 Reactive Data Structures

The term reactive data structure was introduced in the IDECAP project (Projectgroep, 1982; van den Bos, van Naelten, & Teunissen, 1984). A reactive data structure is defined as a *geometric* data structure with *detail levels*, intended to support the sessions of a user working in an interactive mode. It enables the information system to *react* promptly to user actions. This section gives a short description of two reactive data structures: the *BLG-tree* and the *Reactive-tree*. More details can be found in (van Oosterom, 1991; van Oosterom & van den Bos, 1989). These structures are used for the cartographic generalization techniques simplification and selecting. There is still no support for the other generalization techniques: combination, symbolization, exaggeration, and displacement (Shea & McMaster, 1989).

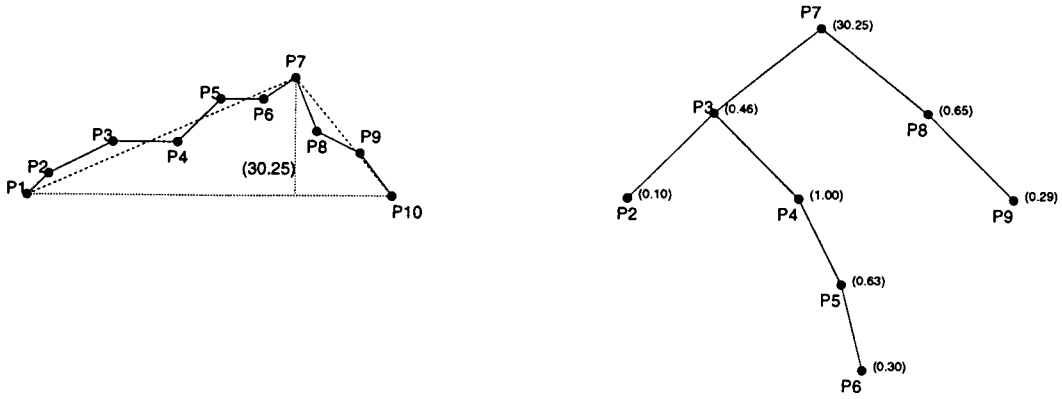


Figure 1: Binary Line Generalization tree

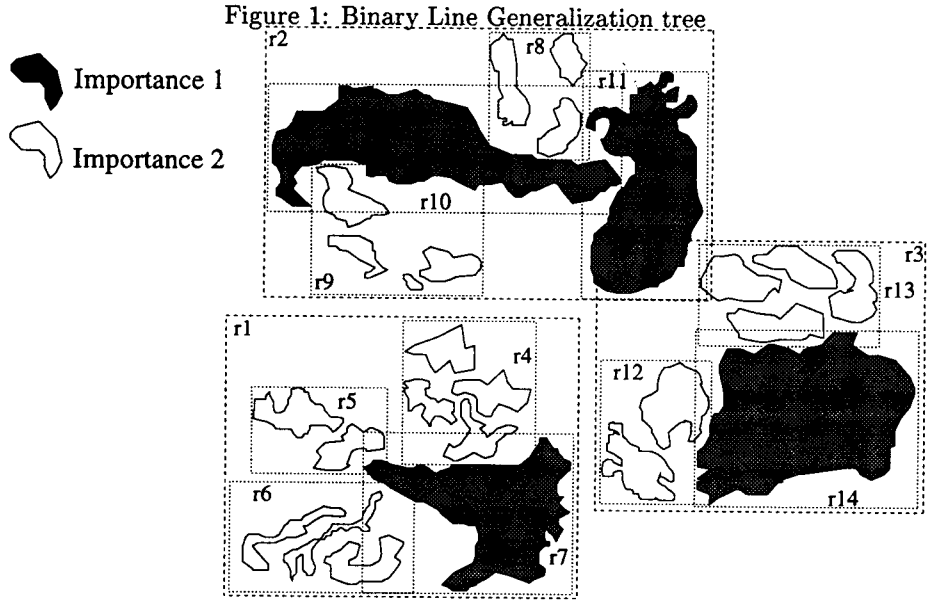


Figure 2: An example of Reactive-tree rectangles

2.1 BLG-tree

The Binary Line Generalization tree (BLG-tree) is a data structure used for line generalization. Only important objects (represented by polylines) need to be displayed on a small scale map, but without a generalization structure, these polylines are drawn with too much detail. A few well-known structures for supporting line generalization are the *Strip tree* (Ballard, 1981), the *Arc tree* (Günther, 1988), and the *Multi-Scale Line Tree* (Jones & Abraham, 1987). We use another data structure, the BLG-tree, because it is suited for polylines, continuous in detail level, and can be implemented with a simple binary tree.

The algorithm to create a BLG-tree is based on the Douglas-Peucker algorithm (Douglas & Peucker, 1973). The first approximation for a polyline is a line between the first point P_1 and the last point P_n. These points are always necessary, otherwise polygons which are composed of several polylines may not be closed when the end-points are omitted. To acquire the next approximation the point P_k with the greatest distance with respect to the line segment (P_1, P_n) is selected. When the original polyline is represented with these

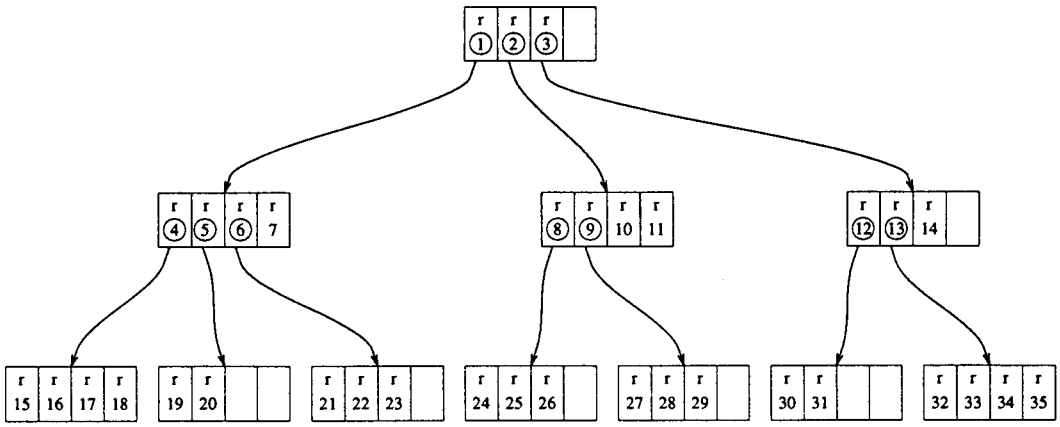


Figure 3: The Reactive-tree

three points P_1, P_k, and P_n, a better approximation is normally found. The point P_k is used to split the polyline into two parts. The same procedure is recursively applied to the polyline parts from P_1 to P_k, and from P_k to P_n. Figure 1 illustrates the BLG-tree creation procedure and the resulting binary tree. This tree is stored for each polyline in order to avoid the expensive execution of the Douglas-Peucker algorithm, each time a line generalization is needed.

2.2 Reactive-tree

The Reactive-tree is based on the R-tree (Guttman, 1984), and has therefore similar properties. The main differences with the R-tree are that the internal nodes of a Reactive-tree can contain both *tree entries* and *object entries*, and the leaf nodes can occur at higher tree levels. An object entry looks like: *(MBR, imp-value, object-id)* where *MBR* is the minimal axes-parallel bounding rectangle, *imp-value* is a positive integer indicating the importance level, and *object-id* contains a reference to the object. The type of an object can be any geometric type; e.g. point, polyline, or polygon.

A basic notion is the *importance* of an object o, which is a function of its type (or role), thematic attributes, and geometric size: $I(o) = f(type, attributes, size)$. For example, the size of an area object could be measured by its area $A(o)$ or perimeter $P(o)$. The weight of the type of the object depends on the application. For example, a city area may be more important than a grassland area in a given application. This could be described with the following importance function (without using the thematic attributes): $I(o) = A(o) * WeightFactor(o)$.

A tree entry looks like *(MBR, imp-value, child-pointer)* where *child-pointer* contains a reference to a subtree, *MBR* is the minimal axes-parallel bounding rectangle of the whole subtree, while *imp-value* ($I(o)$) contains the importance of the child nodes decremented by one. The Reactive-tree has the following properties which must be maintained during insertion and deletion:

1. Each node is a physical disk page which contains between $M/2$ and M entries unless it has no siblings (a *pseudo-root*).
2. All nodes on the same level contain entries with the same importance value. More important entries are stored higher.

3. The root contains at least two entries unless it is also a leaf.

In nodes with both tree entries and object entries, the importance level of the object entries is equal to the importance of the tree entries. The Insert and Delete algorithms which maintain the properties of the Reactive-tree are described in (van Oosterom, 1991). Figure 2 shows a scene with objects and MBRs. Figure 3 shows the resulting Reactive-tree. Tree entries are marked with a circle.

The further one zooms in, the more tree levels must be addressed. Roughly stated, during map generation based on a selection from the Reactive-tree, one should try to choose the required importance value such that a constant number of objects will be selected. This means that if the required region is large only the more important objects should be selected and if the required region is small, then the less important objects must also be selected. The recursive *Search* algorithm starts by accessing the root of the tree. All the object entries in the node are checked for overlap with the search area, if there is overlap, the object is returned. If the importance of the root is larger than the required importance, all the tree entries are checked for overlap. In case overlap exist, the corresponding subtree is descended. This recursive process is continued for all subtrees, until an importance level equal to the required level is reached. Since all objects of the same importance are on the same level, no further search is needed in this case.

3 The GAP-tree

This section first describes an area partitioning as a map basis, and the problems encountered when generalizing this type of map. In Subsection 3.2 the key to the solution is introduced: the area partitioning hierarchy. The creation of the structure that supports this hierarchy, the GAP-tree, is outlined in Subsection 3.3. Operations on the GAP-tree are described in the last subsection.

3.1 Problems with an Area Partitioning

An area partitioning[1] is a very common structure used as a basis for many maps; e.g. choropleths. Two problems occur when using the techniques described in the previous section:

1. *Simplification:* Applying line generalization to the boundaries of the area features might result in ugly maps because two neighboring area features may now have overlaps and/or gaps. A solution for this problem is to use a topological data structure and apply line generalization to the common edges.
2. *Selection:* Leaving out an area will produce a map with a hole which is of course unacceptable. No obvious solution exists for this problem.

A solution for the second (and also for the first) problem is presented. It is based on a *novel* generalization approach called the *Area Partitioning Hierarchy*, which can be implemented efficiently in a tree structure.

3.2 Area Partitioning Hierarchy

The gap introduced by leaving out one area feature must be filled again. The best results will be obtained by filling the gap with neighboring features. This can be easy in case of

[1] In an area partitioning each point in the 2D domain belongs to exactly one of the areas (polygons). That is, there are no overlaps or gaps.

so called "islands": the gap will be filled with the surrounding area, but it may be more difficult in other situations.

By taking both the spatial relationships and the importance (see Subsection 2.2) of an area feature into account, an *area partitioning hierarchy* is created. This hierarchy is used to decide which area is removed and also which other area will fill the gap of the removed feature.

3.3 Building the GAP-tree

The polygonal area partitioning is usually stored in a topological data structure with nodes, edges, and faces. The process, described below, for producing the area partitioning hierarchy assumes such a topological data structure (Boudriault, 1987; DGIWG, 1992; Molenaar, 1989; Peucker & Chrisman, 1975); see Figure 4. The topological elements have the following attributes and relationships:

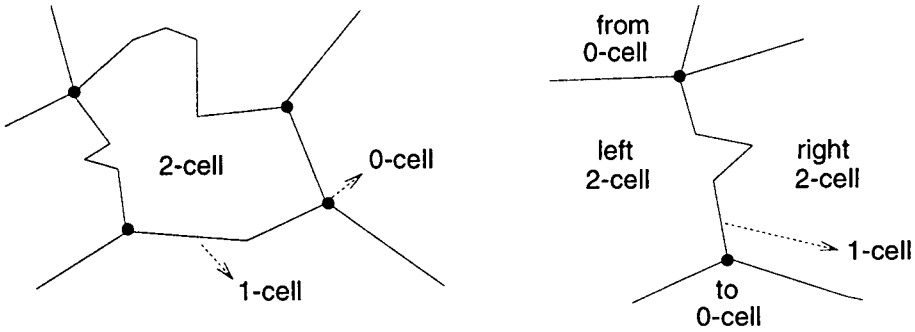


Figure 4: The topological data structure

- a *node* (or 0-cell) contains its point and a list of references to edges sorted on the angle;
- an *edge* (or 1-cell) contains its polyline, length and references to the left and to the right face;
- a *face* (or 2-cell) contains its *WeightFactor*, area, and a list of sorted and signed references to edges forming the outer boundary and possibly inner boundaries;

Note that this topological data structure contains some redundant information because this enables more efficient processing later on. After the topological data structure has been created, the following steps will produce a structure, called the *Generalized Area Partitioning* (GAP)-tree, which stores the required hierarchy:

1. For each face in the topological data structure an "unconnected empty node in the GAP-tree" is created;
2. Remove the least important area feature a, i.e., with the lowest importance $I(a)$, from the topological data structure;
3. Use a topological data structure to find the neighbors of a and and determine for every neighbor b the length of the common boundary $L(a, b)$
4. Fill the gap by selecting the neighbor b with the highest value of the collapse function: $\text{Collapse}(a, b) = f(L(a, b), \text{CompatibleTypes}(a, b), \text{WeightFactor}(b))$. The function $\text{CompatibleTypes}(a, b)$ determines how close the two feature types of a and b are in the feature classification hierarchy associated with the data set (AdV, 1988;

DGIWG, 1992; DMA, 1986). For example, the feature types "tundra" and "trees" are closer together than feature types "tundra" and "industry" in DLMS DFAD (DGIWG, 1992);

5. Store the polygon and other attributes of face a in its node in the GAP-tree and make a link in the tree from parent b to child a;
6. Adjust the topological data structure, importance value $I(b)$, and the length of common boundaries $L(b, c)$ for every neighbor c of the adjusted face b to the new collapsed situation.

Repeat the described steps 2–6 until all features in the topological data structure are at the required importance level (for a certain display operation). This procedure is quite expensive and probably too slow for large data sets to be performed on-the-fly. Therefore, the hierarchy is pre-computed and stored in the GAP-tree. The 2–6 steps are now repeated until only one huge area feature is left, because we can not know what the required importance level will be during the interactive use in a GIS. The last area feature will form the root of the GAP-tree. Further, a priority queue may be used to find out efficiently which face a has the lowest importance value $I(a)$ in step 2 of the procedure.

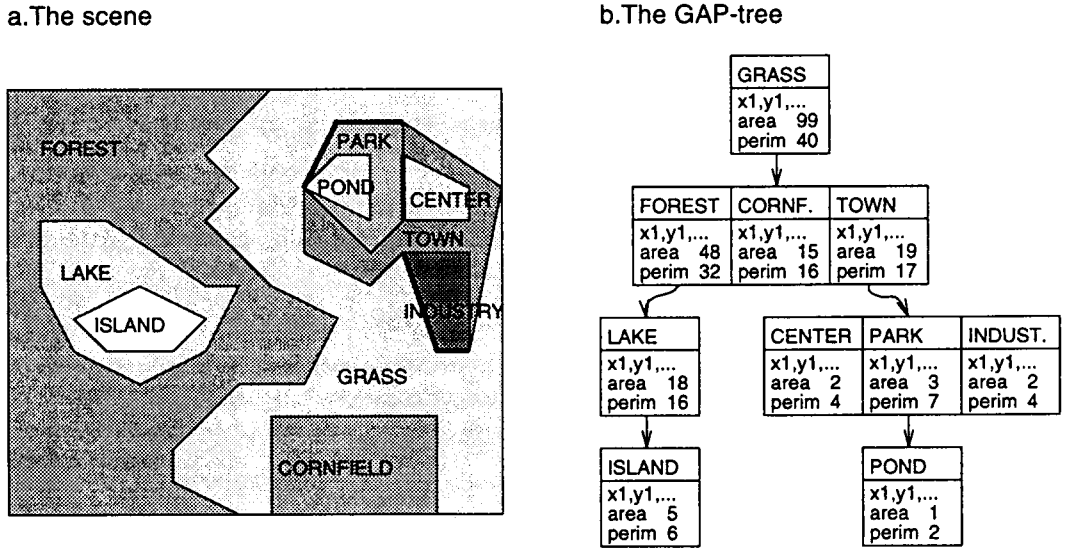


Figure 5: The scene and the associated GAP-tree

Figure 5.a shows a scene with a land-use map in the form of an area partitioning. In Figure 5.b the GAP-tree, as computed by the procedure described above, is displayed. Note that a few attributes are shown in Figure 5.b: polygon, area, and perimeter of the final feature in the GAP-tree. The polygon is a real self-contained polygon with coordinates, and it is not a list of references. It is important to realize that this data structure is not redundant with respect to storing the common boundaries between area features. The only exception to this is the situation where a child has a common edge with another child or its parent; see the thick edges in Figure 5.a.

More statistical information has to be obtained on how frequent this situation occurs in real data sets. Islands may be relatively uncommon for many area partitionings, particularly statistical reporting zones and political boundaries. A solution for the "common edges" problem is to store references to these edges, similar to a topological data structure. Care

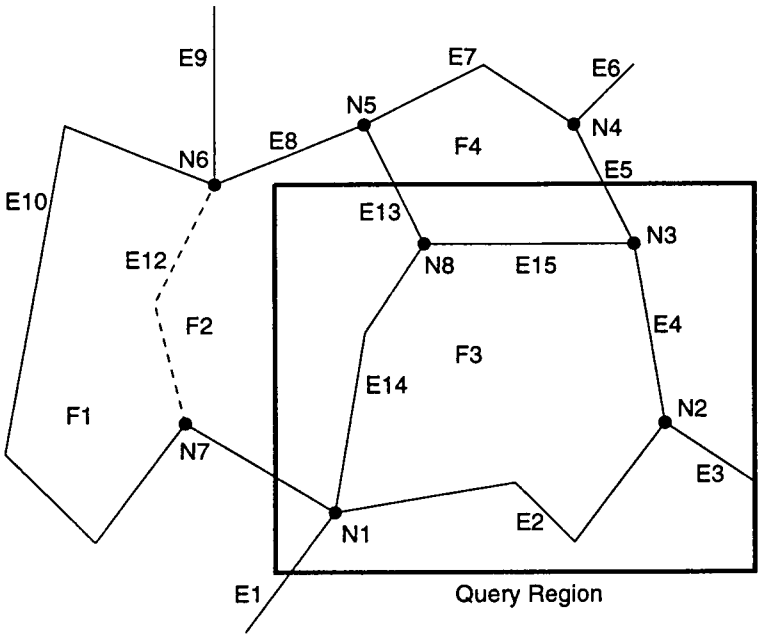


Figure 6: Missing edges with the query region overlap selection

must be taken in order to achieve that these edges are also in the right place in the GAP-tree. Because the tree is traversed (and displayed) in a breadth-first order, the best location for a shared edge is together with the area feature that is on the highest level in the GAP-tree. The child that shares this edge, contains a reference to it. The objects retrieved from the database are buffered (in main memory) and displayed, so references can be traced efficiently. In the case of a common edge between two objects at the same level, the edge is stored in the first one and the second one contains the reference to this edge. A final problem is illustrated in Figure 6 where the neighbor of feature F2 is feature F1. Feature F1, which contains edge E12, falls outside the query region. Therefore, the common edge E12 is not in the buffer. However, the missing edge is not really necessary as it is also completely outside the query region. The area feature F2 can be clipped against the query region.

As can be seen in Figure 5.b, the GAP-tree is a multi-way tree and not a binary tree. Some visual results of the on-the-fly generalization techniques with real data are displayed in Section 5. The additional operations using the GAP-tree, not necessarily related to visualization, are described in the next subsection.

3.4 Operations on the GAP-tree

As a feature in the GAP-tree contains the total generalized area, the actual area A of the polygon has to be corrected for the area of its children with the following formula:

$$A(actual) = A(parent) - \sum_{child \in childeren} A(child)$$

It is important to realize that only one level down the tree has to be visited for this operation and not the whole subtree below the parent node. In a similar way the perimeter P of a polygon can be computed, with the only difference that the perimeter of the children have

to be added to the perimeter of the parent. This results in the formula:

$$P(actual) = P(parent) + \sum_{child \in childeren} P(child)$$

This formula for perimeter only works if the children have no edges in common with each other or with the parent. Often the boundaries of the areas in the GAP-tree are indeed non-redundant. This also enables the use of the BLG-tree for simplification of important area features at small-scale maps without producing overlaps or gaps between features. The use of the BLG-tree has a very positive effect on the response times of small-scale maps; see Section 5.

4 Postgres Implementation

In this section we indicate how the implementations of the reactive data structures are linked to the Postgres DBMS. There is a big implementation difference between the BLG-tree and the Reactive-tree, because the BLG-tree is implemented as a part of an abstract data type, while the Reactive-tree is a new access method. The latter is far more difficult to add, because an access method interacts with many (undocumented) parts of the Postgres DBMS. The reader will be spared from the C-code details of the implementation, which can be found in (Schenkelaars, 1992). The implementations of the BLG-tree, Reactive-tree, and GAP-tree are described in Subsections 4.1, 4.2, and 4.3 respectively. An example using these structures is given in Subsection 4.4.

4.1 BLG-tree

The BLG-tree is integrated into the `POLYLINE2` data type (Vijlbrief & van Oosterom, 1992). In Postgres, each data type should at least be provided with two functions: an input function which converts an external ASCII representation into an internal representation, and an output function which translates the internal representation into an ASCII string. The `POLYLINE2` input function is modified to create a BLG-tree and to store this tree along with the defining points of the original line.

The process of retrieving a generalized polyline from a BLG-tree is less complex than the creation process. The output function `Blg2Pln` is provided to retrieve a generalized polyline. It is called with two parameters: the original polyline, and the maximum distance (error) between the original polyline and the generalized polyline. Note that in this manner, the database does not return unnecessary points to the application, which can save a substantial amount of data transfer time. For polygons a similar implementation is created: the `POLYGON2` data type with the output function `Blg2Pgn`.

4.2 Reactive-tree

In order to use the Reactive-tree as a new access method, some meta information must be inserted in the following Postgres system tables: `pg_am`, `pg_proc`, `pg_operator`, `pg_opclass`, `pg_amop`, and `pg_amproc`. An access method has to be registered in the table `pg_am`. This is done with the following query:

```
append pg_am(
  amname = "ReactiveTree",          amowner = "6",
  amstrategies = 8,                 amsupport = 6,
  amgettuple = "reactgettuple",     aminsert = "reactinsert",
```

```
amdelete = "reactdelete",        ambeginscan = "reactbeginscan",
amrescan = "reactrescan",        amendscan = "reactendscan",
ammarkpos = "reactmarkpos",      amrestrpos = "reactrestrpos",
ambuild = "reactbuild")
```

The first attribute contains the name of the new access method (`ReactiveTree`). The second attribute indicates the user-id of the owner of the access method; in this case user-id number 6. In Postgres an access method is based on:

1. A set of (boolean) *access method user functions* to indicate the relative position of two objects; in case of a 2D spatial index this set contains 8 (`amstrategies`) elements. For the Reactive-tree these are called: `Left2ReaRea`, `Overleft2ReaRea`, `Overlap2ReaRea`, `Right2ReaRea`, `Overrgt2ReaRea`, `Equal2ReaRea`, `Contain2ReaRea`, and `ContBy2ReaRea`. For each of these functions an operator symbol has to be defined (in the `pg_operator` table), which can be manipulated by the query optimizer. These will be called the *access method user operators* and are numbered according to the following sequence <<, &<, &&, &>, >>, ~=, ~, and @.
2. A set of *access method procedures*; in case of the Reactive-tree 6 (`amsupport`) functions are needed and numbered in the order: `Union2ReaRea`, `Inter2ReaRea`, `Size2Rea`, `IsObject2Rea`, `MakeTreeObj2Rea`, `React2Imp`.
3. A fixed set of *access method manipulation functions*:
   - `amgettuple` contains the name of the function which performs an index search: `reactgettuple`;
   - `aminsert` contains the function which inserts a tuple in the index: `reactinsert`;
   - `amdelete` contains a function which deletes a tuple from the index: `reactdelete`;
   - `ambeginscan` contains a function which initializes an index scan: `reactbeginscan`;
   - `amrescan` contains a function which rescans a previous started scan: `reactrescan`;
   - `amendscan` contains a function which ends an index scan: `reactendscan`;
   - `ammarkpos` contains a function which marks a position in a scan: `reactmarkpos`;
   - `amrestrpos` contains a function which releases a marked position in the scan: `reactrestrpos`;
   - `ambuild` contains a function which initializes the index and add all tuples that are in the relation: `reactbuild`.

All these functions (access method user functions, access method procedures, and access method manipulation functions) must be registered in the `pg_proc` table. This is done, for example for the `reactinsert`, as follows:[2]

```
define function reactbuild (language = "c", returntype = int4)
  as "$REACTHOME/reactree.o"
```

For each data type the user wants to access with the defined access method, an "operator class" has to be defined (in the `pg_opclass` table). An operator class collects both access method procedures (registered in the `pg_amproc` table) and access method operators (registered in `pg_amop` table).

[2]One problem arises: the functions often need arguments of types that do not exist as standard Postgres types. For example `reactinsert` needs a `Relation` type. Therefore no arguments, and an int4 as return value, are defined in the function definition. Postgres does no type checking in this case. Only the number of arguments the function expects, has to be provided:
`define function reactinsert (language = "c", returntype = int4) as "$REACTHOME/reacttree.o"`
`replace pg_proc (pronargs = 2) where pg_proc.proname = "reactinsert"`

In order to make sensible use of the Reactive-tree, a data type is required that possesses both a geometric attribute and an importance level. For this purpose a new type was created: REACTIVE2, which contains a bounding box and an importance level.

The following sequence of Postquel queries shows the registration of the operator class Reactive2_op in the pg_opclass table, and examples of adding a operator respectively a procedure to the pg_amop table respectively to the pg_amproc table.

```
append pg_opclass (opcname = "Reactive2_ops")

append pg_amop(
  amopid = am.oid, amopclaid = opc.oid,
  amopopr = opr.oid, amopstrategy = "1"::int2, ...)
  from opc in pg_opclass, opr in pg_operator,
    opt in pg_type, am in pg_am
  where opr.oprname = "<<" and opt.typname = "REACTIVE2" and
    opt.oid = opr.oprright and opt.oid = opr.oprleft and
    am.amname = "ReactiveTree" and opc.opcname = "Reactive2_ops"

append pg_amproc(
  amid = am.oid, amopclaid = opc.oid,
  amproc = proc.oid, amprocnum = "1"::int2)
  from opc in pg_opclass, proc in pg_proc,
    am in pg_am
  where am.amname = "ReactiveTree" and
    opc.opcname = "Reactive2_ops" and proc.proname = "Union2ReaRea"
```

4.3 GAP-tree

A possible implementation difficulty of the GAP-tree is the fact that it is a multi-way tree and not a binary tree. Therefore, a simple linear version has been derived from the GAP-tree by putting the features in a list based on their level in the tree. The top level feature in the tree will be the first element of this list, the second level features will follow, and so on. For example, the linear list for the scene in Figure 5 is: GRASS, FOREST, CORNFIELD, TOWN, LAKE, CENTER, PARK, INDUSTRY, ISLAND, POND. When the polygons are displayed in this order, a good map can be produced without the GAP-tree. However, it is very difficult to compute the actual area without the GAP-tree.

As mentioned before, the on-the-fly generalization has been developed within the Postgres DBMS environment. Postgres is an extensible relational system. A relation does not guarantee any order among its elements. A good display can be obtained by sorting on the sequence number in the list or on the area of the feature.

4.4 Example

In this subsection an example of the use of the Reactive-tree, BLG-tree, and GAP-tree will be given. The Reactive-tree access method can be defined before any tuples are added to the relation, or one can first insert all tuples and decide later that a Reactive-tree is needed. The following two Postquel queries show the definition of the user table AreaFeature and the definition of a Reactive-tree index on this table using the Reactive2_ops operator class.

```
create AreaFeature (Height=int2, Idcode=int2, Tree=int2,
   Roof=int2, shape=POLYGON2, reactive = REACTIVE2)\g
```

```
define index af_index on AreaFeature
   using ReactiveTree (reactive Reactive2_ops)\g
```

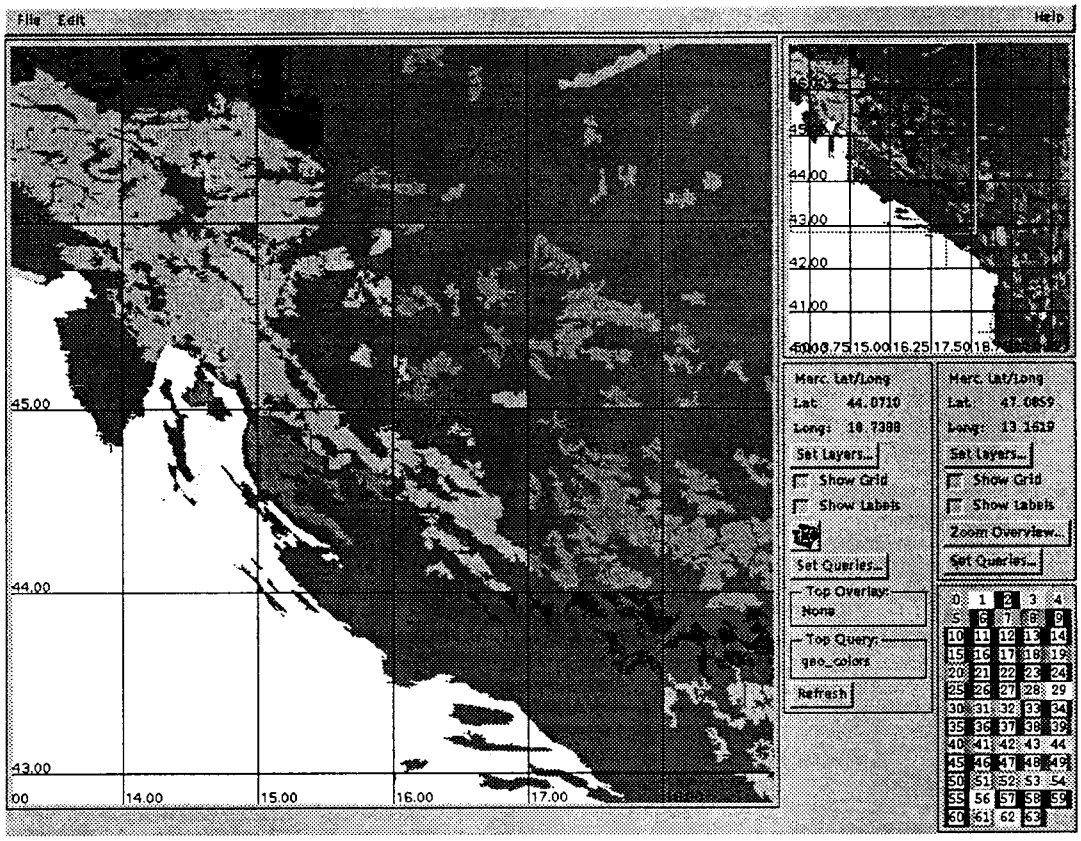


Figure 7: Using the GAP/Reactive-tree and BLG-tree (DLMS DFAD; coarse map)

Figures 7 and 8 show the DLMS DFAD data set in the "interactive use" of the Reactive-tree, the BLG-tree, and the GAP-tree in the Postgres GIS frontend GEO++ (Vijlbrief & van Oosterom, 1992). Note that no visual loss of information occurs at the top map which is created from the same geographic data set. The GEO++ system automatically generates Postquel queries with the proper values for the BLG-tree and the Reactive-tree depending on the current scale. For the map in Figure 7 the following query is generated:

```
retrieve (blg_pgn2= Blg2Pgn(AreaFeature.shape,"0.01"::float4))
  where AreaFeature.reactive && "(13,40,23,47,2)"::REACTIVE2
  sort by AreaFeature.oid
```

The Postgres query optimizer automatically selects the Reactive-tree access method when evaluating this query. The BLG-tree is used by specifying the function `Blg2Pgn` in the target list of the query. The linearized GAP-tree is reflected by the *sort by* clause of the query.

5 Performance Results

In this section the benchmarks of the reactive data structures are presented. In subsection 5.1 is The World Databank II (WDB II; see Figure 9) (Gorny & Carter, 1987) test described.

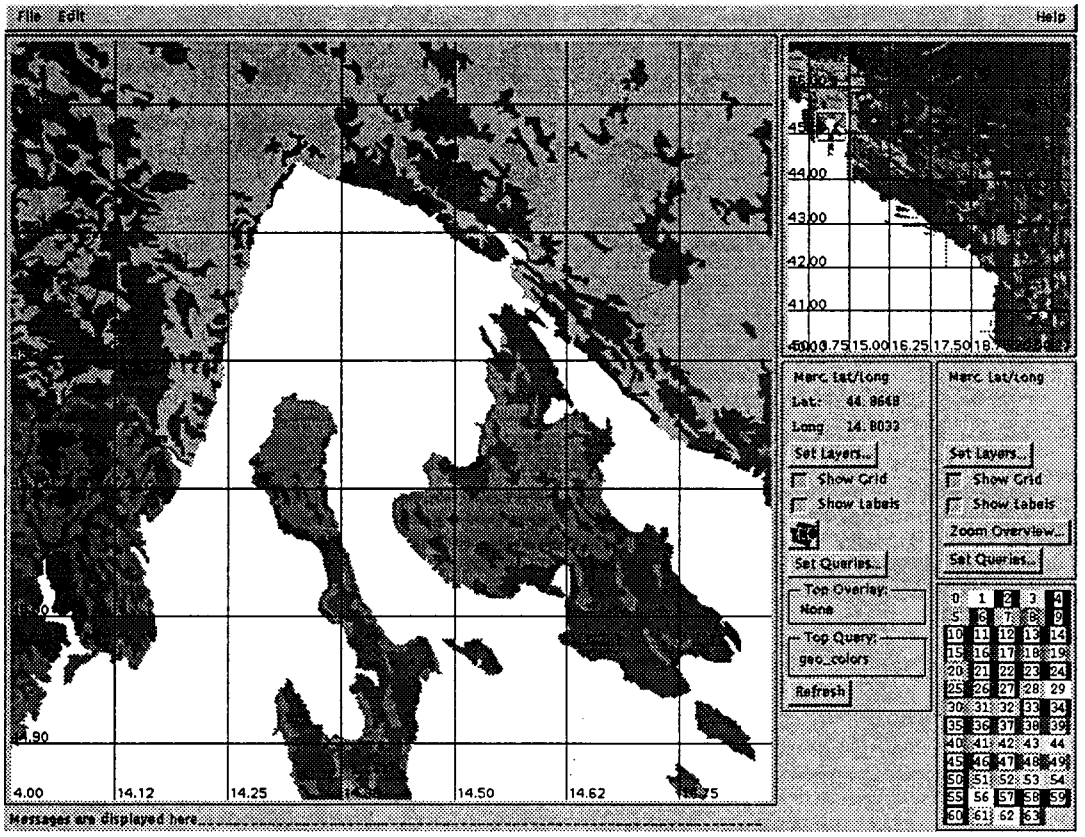


Figure 8: Using the GAP/Reactive-tree and BLG-tree (DLMS DFAD; detail map)

In subsection 5.2 are the results shown of the tests with the DLMS DFAD[3] data (DMA, 1986) of the former Republic of Yugoslavia presented.

The tests were performed on a Sun SPARCstation II (32Mb main memory) under SunOs 4.1.2. The data was retrieved over a network file system by Postgres. The special test program is a simple Postgres frontend application. The response time was measured with the Unix `time` command. The test program needs a parameter which indicates the size of the search area. This size is used to calculate which importance values are retrieved: thus less important features are retrieved when the area is smaller. Of course, the user could overule the level of importance, as generated by the frontend, in a specific query.

The R-tree has no mechanism to select on importance level, that is why the R-tree retrieves much more objects in the larger areas. The following cases were tested: no index structure, an R-tree index, an R-tree index and a BLG-tree, a Reactive-tree index and a BLG-tree. The test on the DLMS DFAD data set uses the GAP-tree too in the last case. In all test, ten random area queries are executed. The presented response time is in seconds and is the average time over the ten randomly generated queries. The Reactive-tree uses importance levels to reduce the number of selected objects at the more coarse level. Actually, the information density (i.e., the number of displayed features) should remain equal under the varying scales.

[3]DLMS (Digital LandMass) DFAD (Digital Feature Analyses Data) Level 1 has a data density which can be compared to 1:200,000 scale map

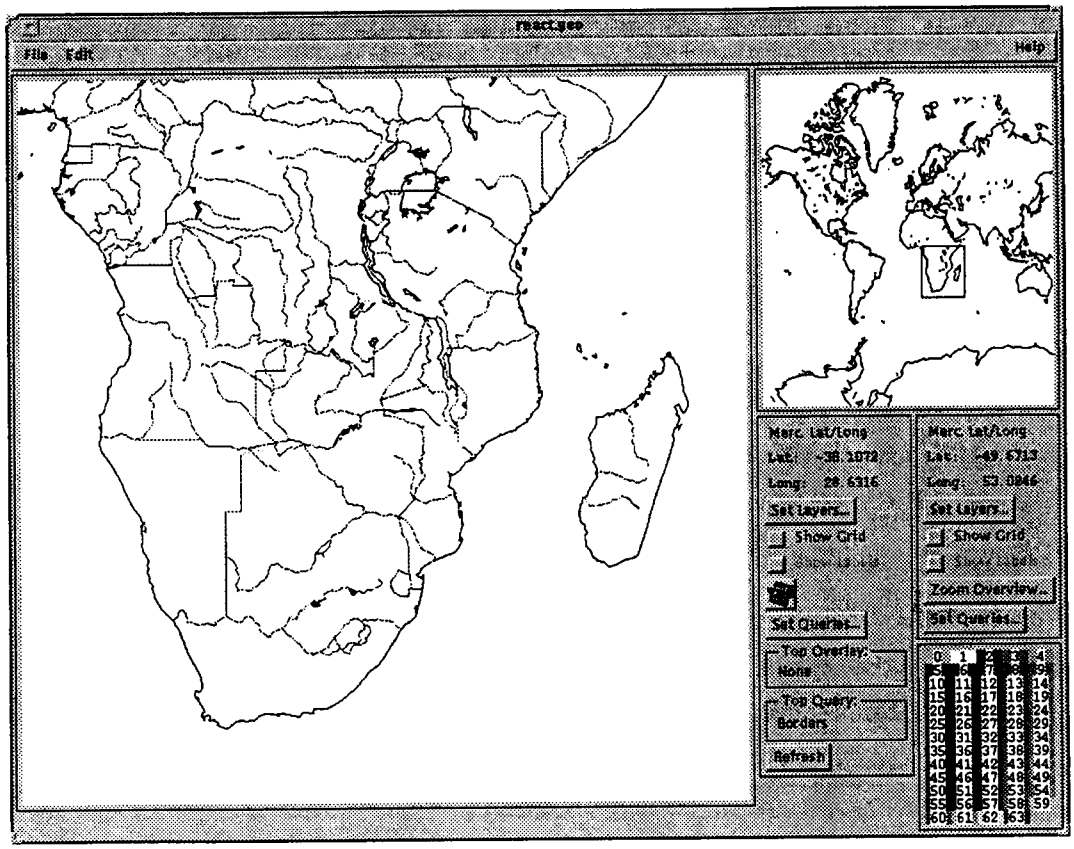


Figure 9: Using the Reactive-tree and BLG-tree (WDB II)

5.1 The World Data Bank II Test

The WDB II set is not ideal for the Reactive-tree because the number of objects does not increase enough when a lower importance level is reached. But because of the very large polylines in the dataset, the BLG-tree resulted in a substantial performance improvement. The total number of features in this data set is 38.096.

First the raw data needed to be translated into Postgres tuples. This resulted in a database of approximate 120Mb (116Mb user data, 3Mb index, 1Mb Postgres meta-data).

Queries which selected tuples at four different area sizes were executed. The area sizes varied from 32 × 32 degrees to 4 × 4 degrees latitude/longitude. These area sizes were selected because in each area decrement, one level of detail more is shown. The Reactive-tree in the WDB II test used importance levels form 1 to 4.

5.2 The DLMS DFAD Test

In this section the results of the performance tests of the combined use of the GAP-tree, the Reactive-tree, and the BLG-tree are presented. The DLMS DFAD data of the former Republic of Yugoslavia is used. Only the area features are used in order to evaluate the effectiveness of the GAP-tree.

The visual results of the on-the-fly generalization techniques can be seen in Figures 7 and 8. All maps, including the overview in the upper right, are generated by the same query

with only different sized retrieved regions. DLMS DFAD can be regarded as land-use data with over 100 different area classifications, such as: lake, water, trees, sand, swamp, tundra, snow/ice, industry, commercial, recreational, residential, etc. A few notes with respect to the visualization:

1. Many colors on the screen are lost in the gray scales of the printer.
2. As the emphasis is on the GAP-tree, the line and points features have been omitted, resulting in an incomplete map.
3. In the upper right corner of each figure, an overview of the region is shown without the mainland of Italy.
4. Though the DLMS DFAD data is stored in a seamless database, it has been digitized on a map sheet base. During this process similar features have been classified differently; see Figure 7 near the 44 degrees meridian.

The DLMS DFAD test database contains 70.272 area features (requiring about 60Mb) which are given an importance value ranging from 1 to 5, at each level the more detailed level contains about one order of magnitude more features. Queries which selected tuples at five different area sizes were executed. The area sizes varied from 1.6×1.6 degrees to 0.1×0.1

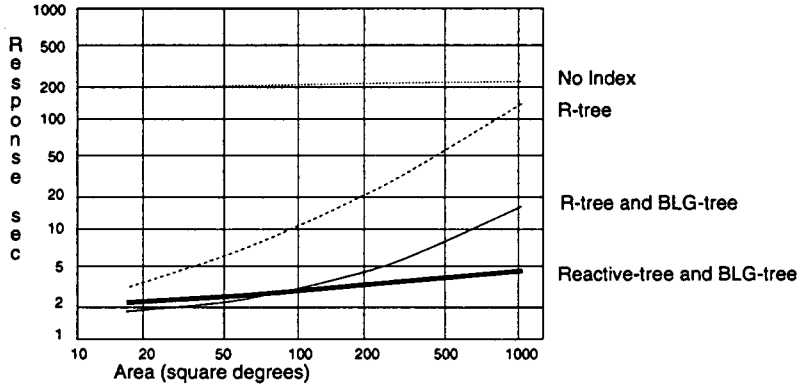


Figure 10: The average response time of the index structures in WDB II

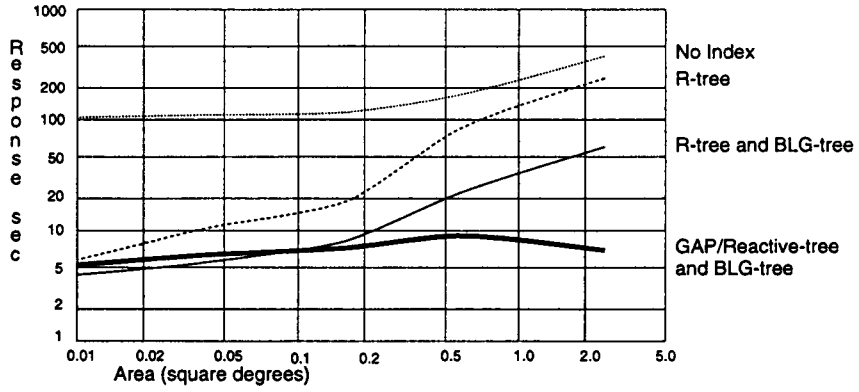


Figure 11: The average response time of the index structures in DLMS DFAD

Table 1: Area sizes and the average number of returned objects in WDB II

Area (in square degrees)	16	64	256	1024
No index	16.2	45.9	244.2	1380.4
R-tree	16.2	45.9	244.2	1380.4
R-tree & BLG-tree	16.2	45.9	244.2	1380.4
Reactive-tree & BLG-tree	10.7	23.8	67.0	257.6

degrees latitude/longitude. The use of the GAP-tree will make sure that the map does not contain gaps when omitting the less important area features.

5.3 Results Analysis

The difference in response time between the Reactive-tree and the R-tree is decreasing when moving to smaller areas, since the difference in the number of returned objects is narrowing. Tables 1 and 2 show the average number of returned objects for each method. In a similar way, the BLG-tree is more effective when used in a large area search. In that case, a lot of polyline or polygon points can be omitted.

Table 2: Area sizes and the average number of returned objects in DLMS DFAD

Area (in square degrees)	0.01	0.04	0.16	0.64	2.56
No index	25.9	64.7	198.0	723.6	2262.3
R-tree	25.9	64.7	198.0	723.6	2262.3
R-tree & BLG-tree	25.9	64.7	198.0	723.6	2262.3
GAP/Reactive-tree & BLG-tree	25.9	52.8	43.6	55.8	48.7

All in all, the Reactive-tree in combination with the BLG-tree gives very good improvements. The process of selection and generalization results in a better map without too much detail. The selection could be done with the R-tree, but in that case a user should change the queries on every scale, in order to select only the desired objects. Also, the important objects would then be located at the leaf level in the tree just as the other objects, which would slow down the small-scale queries. Figures 10 and 11 show the response times of the different methods, while Tables 3 and 4 show the same information in tabular form.

Using no index at all, results in extremely long response times for data sets of the presented size even for relatively small query regions: 205 seconds for WDB II (16 square degrees) and 113 seconds for DLMS DFAD (0.01 square degrees); see Tables 3 and 4. This is due to the fact that a sequential scan over the whole data set has to be performed. Using a spatial index structure enhances the performance for small query regions dramatically as can be seen in Figures 10 and 11.

For very small query regions, using the R-tree and BLG-tree, is even slightly more efficient

Table 3: Area sizes and the average response time in WDB II

Area (in square degrees)	16	64	256	1024
No index	205.5	211.2	230.8	348.8
R-tree	3.4	7.0	27.0	143.5
R-tree & BLG-tree	1.8	2.2	5.0	15.3
Reactive-tree & BLG-tree	2.3	2.6	3.2	4.9

Table 4: Area sizes and the average response time in DLMS DFAD

Area (in square degrees)	0.01	0.04	0.16	0.64	2.56
No index	113.8	117.7	127.1	183.4	378.7
R-tree	6.6	10.2	19.0	80.5	285.8
R-tree & BLG-tree	4.3	5.7	8.1	21.2	61.2
GAP/Reactive-tree & BLG-tree	5.4	6.9	7.5	9.7	7.3

than using the Reactive-tree and the BLG-tree. The reason for this is the small overhead in using a user-defined access method (Reactive-tree) instead of a Postgres build-in access method (R-tree). The differences are very small. However, the R-tree/BLG-tree response times increase a lot when the query size regions increases. This is caused by the large number of objects which do not need to be retrieved when using the Reactive-tree/BLG-tree. This makes it possible to achieve more or less constant response times irrespective of the size of the query region: always less than 5 seconds for WDB II and always less than 10 seconds for the DLMS DFAD.

If geometrically close objects are guaranteed stored close together on disk, some extra speed may be gained. The Reactive-tree does not give such a guarantee, it depends on the inserting order, whether two geometrically close objects are also stored close on disk. *Clustering* can be used to improve this storage aspect. In Postgres, a simple kind of clustering can be easily simulated by retrieving the objects using a rectangle that contains all the objects. In fact, an access method scan is performed. When the objects are stored in a new relation in the retrieved order, geometrically close objects are stored close together on disk. This reduces the number of disk pages to be fetched for spatial queries and therfore the results will be returned faster.

6 Self-Adjusting Reactive-tree

In this section another type of Reactive-tree is presented: the *self-adjusting Reactive-tree.* A drawback of the normal Reactive-tree, as presented in 2.2, is that there is a one-to-one correspondence between the levels in the tree and the importance values. This is no problem if there is a hierarchical distribution of the data with respect to the importance values and when these values are properly numbered. However, in case the distribution is different, one would still want the Reactive-tree to behave in an efficient manner. The same is true if the user would decide only to use "strangely" numbered importance values; e.g. 1, 10, 15, 50, and 900. The pseudo-roots may cause the tree to be underfull and badly balanced.

The one-to-one correspondence between tree levels and importance values probably will have to be abandoned. It is then possible that objects of different importance are stored in the same node (at the *same* tree level), as long as less important objects are never stored in nodes *above* more important objects. It might also be convenient to store objects of the same importance at different tree levels if this helps to get a well-filled and balanced tree. There are two ways in which this new object ordering may be carried out:

- In a *global* manner: nowhere in the tree is it allowed that a less important object is stored on a higher tree level than a more important object.
- In a *local* manner: if a node contains an object of a certain importance than the sub-tree below this node may not contain more important objects.

It is not difficult to see that if the global ordering has to be satisfied, it is then impossible to

always get a well-filled (and balanced) tree. For example, in the case that there is a strange data distribution, such as: a large number of important objects to the left-hand side of the scene and a number of less important objects located to the far right-hand side. Objects which are spatially far apart should not be stored in the same sub-tree. However, if they are stored in different subtrees, then the less important objects would be stored too high unless there is a path of (nearly empty) pseudo-roots.

Therefore, we abandoned the global ordering and tried to design a Reactive-tree which satisfies the less strict local object ordering. In subsequent research we will carry out practical testing with both the normal Reactive-tree and the self-adjusting Reactive-tree for different data sets; both well and badly distributed.

The self-adjusting Reactive-tree satisfies the following defining properties:

1. For each object entry *(MBR, imp-value, object-id)*, *MBR* is the smallest axes-parallel rectangle that geometrically contains the represented object of importance *imp-value*. The most important objects have the lowest importance values.
2. For each tree entry *(MBR, imp-value, child-pointer)*, *MBR* is the smallest axes-parallel rectangle that geometrically contains all rectangles in the child node and *imp-value* is the importance of the child-node adequately decremented.
3. All the nodes on the same level are in the same importance value range. This implies that it is not possible that a subtree contains object entries that are more important (lower importance value) than the object entries in the parent of this subtree.
4. Every node contains between $m(\leq M/2)$ and M object entries and/or tree entries.
5. The importance of the tree entries is less or equal to the importance value of the least important (highest importance value) object entry in every node.
6. The root contains at least 2 entries unless it is a leaf.

The fact that the empty tree satisfies these properties and that the Insert and Delete algorithms (Schenkelaars, 1992) do not destroy them, guarantees that a self-adjusting Reactive-tree always exists.

7 Conclusion

After the Reactive-tree and BLG-tree, the GAP-tree forms a new important step towards the realization of an interactive multi-scale GIS. It is now possible to interactively browse through large geographic data sets. In both the DLMS DFAD database (70.000 features, 60 Mb) and the WDB II database (38.000, 120 Mb) it is now possible to get map displays at any required scale in about five seconds. In the near feature, more tests with datasets are planned; e.g., with the Topographic base map of The Netherlands. An interesting open question is how to assign importance values automatically to features when building the Reactive-tree for a new data set.

A new reactive data structure is presented: the self-adjusting Reactive-tree, which should be even more effective. Future implementation and testing will have to confirm this expectation. Another question which still has to be answered is: When is it better to store the BLG-tree and when is computing a generalized line on the fly preferred? This is a matter of balancing CPU and disk speed.

Further research is also required to determine how the GAP-tree can be maintained efficiently under edit operations. Additional further research topics are: the use of (dynamic) clustering

techniques, and the design and implementation of other generalization techniques to support: combination, symbolization, and displacement.

Acknowledgments

We would like to thank the Postgres Research Group (University of California at Berkeley) for making their system available. Special thanks to Tom Vijlbrief for helping us with the GEO++ issues. Many valuable comments and suggestions on a preliminary version of this paper were made by Marcel van Hekken and Paul Strooper.

References

AdV (1988). Amtliches Topographic-Kartographisches Informationssystem (ATKIS). Arbeitsgemeinschaft der Vermessungsverwaltungen der Länder der Bundesrepublik Deutschland (AdV). (in German).

Ballard, D. H. (1981). Strip Trees: A Hierarchical Representation for Curves. *Communications of the ACM, 24(5)*, 310–321.

Boudriault, G. (1987). Topology in the TIGER File. in *Auto-Carto 8*, pp. 258–269.

DGIWG (1992). DIGEST – Digital Geographic Information – Exchange Standards – Edition 1.1. Defence Mapping Agency, USA, Digital Geographic Information Working Group.

DMA (1986). Product Specifications for Digital Feature Analysis Data (DFAD): Level 1 and Level 2. Defense Mapping Agency, Aerospace Center, St Louis, Mo.

Douglas, D. H., & Peucker, T. K. (1973). Algorithms for the Reduction of Points Required to Represent a Digitized Line or its Caricature. *Canadian Cartographer, 10*, 112–122.

Gorny, A. J., & Carter, R. (1987). World Data Bank II: General Users Guide. US Central Intelligence Agency.

Günther, O. (1988). *Efficient Structures for Geometric Data Management.* No. 337 in Lecture Notes in Computer Science. Springer-Verlag, Berlin.

Guttman, A. (1984). R-Trees: A Dynamic Index Structure for Spatial Searching. *ACM SIGMOD, 13*, 47–57.

Jones, C. B., & Abraham, I. M. (1987). Line Generalization in a Global Cartographic Database. *Cartographica, 24(3)*, 32–45.

Lagrange, J. P., Ruas, A., & Bender, L. (1993). Survey on Generalization. IGN.

Molenaar, M. (1989). Single Valued Vector Maps: A Concept in Geographic Information Systems. *Geo-Informationssysteme, 2(1)*, 18–26.

Muller, J. C., Weibel, R., Lagrange, J. P., & Salge, F. (1993). Generalization: state of the art and issues. European Science Foundation, GISDATA Task Force on Generalization.

Peucker, T. K., & Chrisman, N. (1975). Cartographic Data Structures. *American Cartographer, 2(1)*, 55–69.

Postgres Research Group (1991). The Postgres Reference Manual, Version 3.1. Tech. rep. Memorandum, Electronics Research Laboratory, College of Engineering.

Projectgroep (1982). IDECAP Interactief Pictorieel Informatiesysteem voor Demografische en Planologische Toepasingen: Een verkennend en vergelijkend onderzoek. Tech. rep.

Publicatiereeks 1982/2, Stichting Studiecentrum voor Vastgoedinformatie te Delft.

Schenkelaars, V. F. (1992). Master's thesis: Implementation of Reactive data structures for Postgres. Tech. rep. FEL-92-S343, FEL-TNO Divisie 2.

Shea, K. S., & McMaster, R. B. (1989). Cartographic Generalization in a Digital Environment: When and How to Generalize. in *Auto-Carto 9*, pp. 56–67.

Stonebraker, M., Rowe, L. A., & Hirohama, M. (1990). The Implementation of Postgres. *IEEE Transactions on Knowledge and Data Engineering, 2(1)*, 125–142.

van den Bos, J., van Naelten, M., & Teunissen, W. (1984). IDECAP Interactive Pictorial Information System for Demographic and Environmental Planning Applications. *Computer Graphics Forum, 3*, 91–102.

van Oosterom, P., & Schenkelaars, V. (1993). Design and Implementation of a Multi-Scale GIS. in *Proceedings EGIS'93: Fourth European Conference on Geographical Information Systems*, pp. 712–722. EGIS Foundation.

van Oosterom, P., & van den Bos, J. (1989). An Object-Oriented Approach to the Design of Geographic Information Systems. *Computers & Graphics, 13(4)*, 409–418.

van Oosterom, P. (1989). A Reactive Data Structure for Geographic Information Systems. in *Auto-Carto 9*, pp. 665–674.

van Oosterom, P. (1991). The Reactive-Tree: A Storage Structure for a Seamless, Scaleless Geographic Database. in *Auto-Carto 10*, pp. 393–407.

van Oosterom, P. (1994). *Reactive Data Structures for Geographic Information Systems.* Oxford University Press, Oxford.

Vijlbrief, T., & van Oosterom, P. (1992). The GEO++ System: An Extensible GIS. in *Proceedings of the 5th International Symposium on Spatial Data Handling, Charleston, South Carolina*, pp. 40–50 Columbus, OH. International Geographical Union IGU.

Application-oriented generalization of area objects

Arnold Bregt and Jandirk Bulens

DLO Winand Staring Centre for
Integrated Land, Soil and Water Research (SC-DLO)
P.O. Box 125, 6700 AC Wageningen, The Netherlands
Tel: +31 317 474458 Fax: +31 317 424812
E-mail: a.k.bregt@sc.dlo.nl
E-mail: j.d.bulens@sc.dlo.nl

Abstract

Generalization of spatial data is frequently applied in geographical information processing. At the DLO Winand Staring Centre this is particular the case in generating input data for simulation models to be applied on a regional or national level. Until now the generalization process has been mainly manual, which is a time-consuming and subjective procedure. In order to overcome these problems research was done on practical automated methods for generalization of area objects. Three methods for generalizing area objects were compared. The first method is based on the geometry of objects, the second method on the attributes of objects, and the third method uses a combination of geometry and attributes. The last two methods incorporate procedures for application specific generalization. To evaluate the effects of the different methods various effect measures were defined. These effect measures can be used by the user to select the most appropriate generalization method and procedure for his application.

Introduction

Automated generalization of spatial data continues to attract the attention of researchers, software producers and users in the GIS community. Recently, a book on the subject has been published by Müller et al. (1995a), describing the results of a specialist meeting in France. Most authors in this book argue that automated generalization is of major importance to spatial-data handling. When going through the different papers in the book, one can, however, not suppress the feeling that the generalization issue is far from solved. More questions are raised than answers given.
Some of the problems mentioned are (Müller et al., 1995b):
- the subjective character of most of the manual generalization procedures, which makes full automation extremely difficult;
- the lack of quantification of the effects of generalization;
- the lack of application-oriented generalization procedures.

This paper discusses the results of a research project on automated generalization of area objects. In this project special attention was paid to the problems mentioned above, such as

reproducibility, application orientation and generalization effects. The study was focused only on so-called model generalization. The attribute types are limited to data on a nominal or ordinal measurement scale.

Generalization

Generalization of spatial data can be defined as the process of creating an application-oriented abstraction of spatial data. Keywords in the above definition are *application orientation* and *abstraction of spatial data.* Abstraction of spatial data involves reduction of the detail of the data. This can be done for both the geometry and the attributes of spatial objects. For instance, when area objects are generalized, small areas can be joined with neighbouring larger areas or the detail of attributes can be reduced by classification. In most cases generalization of attributes also leads to a generalization in geometry.
A second keyword in the definition is application orientation. The generalization process needs to be done with a certain application in mind. The requirements of the user or users group must control the generalization process. This means a flexible generalization procedure in order to meet the different requirements of the user.
The process of generalizing spatial data has been applied for ages. Until recently, the only form has been cartographic generalization for visual presentation. The main object of cartographic generalization is to change a map on a larger scale to a smaller scale. Typical actions are removing (small) area objects and simplifying area geometry. Cartographic generalization has been a manual procedure for a long time, but is more and more supported by automated procedures (Brassel & Weibel, 1988; Lee, 1995).
Due to the increasing use of geographical information systems another form of generalization is increasingly being applied: model generalization. In the literature (e.g. Van Smaalen, 1995) this form of generalization is also called information abstraction or database generalization. We prefer the term database generalization. However, in most of the literature the term model generalization is used and therefore we shall use this term here. In model generalization the main object is not to produce a readable map, but to achieve data reduction. In practice, we quite often see a combination of model generalization and cartographic generalization. First model generalization is carried out to achieve data reduction, followed by cartographic generalization to produce a readable map.
The DLO Winand Staring Centre mainly carries out model generalization, because the main object is generating input data for simulation models which are applied on regional and national levels.

Three generalization methods for area objects

In this study the following three methods for generalization were compared:
- surface method;
- attribute class method;
- similarity method.

Below they will be discussed in more detail. In Table 1 some characteristics of the three methods are presented.

Table 1. Characteristics of the generalization methods

Method	Generalization on basis of	Application-oriented	Application-oriented control by	Expert knowledge
surface method	geometry	no	-	no
attribute class method	attributes	yes	importance factor	yes
similarity method	geometry and attributes	yes	similarity factor	yes

Surface method

According to the surface method generalization is performed on the basis of the geometry of spatial objects. The surface of an area object is used as generalization criterion. Small areas are joined with the adjacent largest area. The intensity of generalization can be controlled by indicating the reduction in the number of areas. An advantage of this method is its simplicity. A disadvantage is that the thematic aspects of the areas are not taken into account. The method can be characterized as application-independent.

Attribute class method

According to the attribute class method generalization is performed on the basis of the classification hierarchy of attributes of spatial objects (Molenaar, 1989; Richardson, 1993). Attributes of area objects can be presented at different levels in the classification hierarchy. For instance, an area object of the type soil has different abstraction levels (Fig. 1).

At the most detailed level (Level 1) an area can be described as heavy clay. At a higher level (Level 2) in the classification hierarchy the same area can be classified as clay. Generalization can be performed by selecting a higher level in the classification hierarchy and joining areas with the same attribute description. In this way no real application-oriented generalization is achieved.

For some applications, however, it may be useful to keep some of the area objects on the most detailed level, whereas for others a presentation at a higher level is sufficient. In the implemented method the application dependency is achieved by adding an importance factor to the attribute classes at the lowest level.

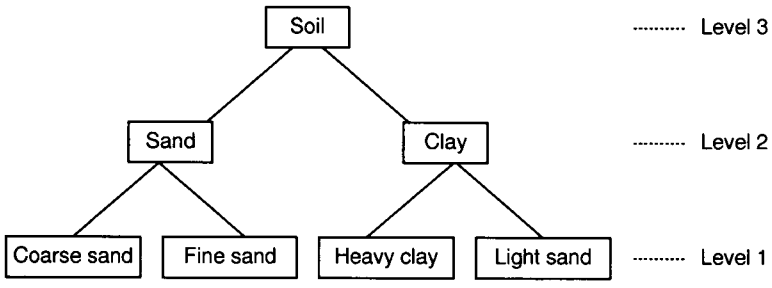


Fig 1. Classification hierarchy of soil data

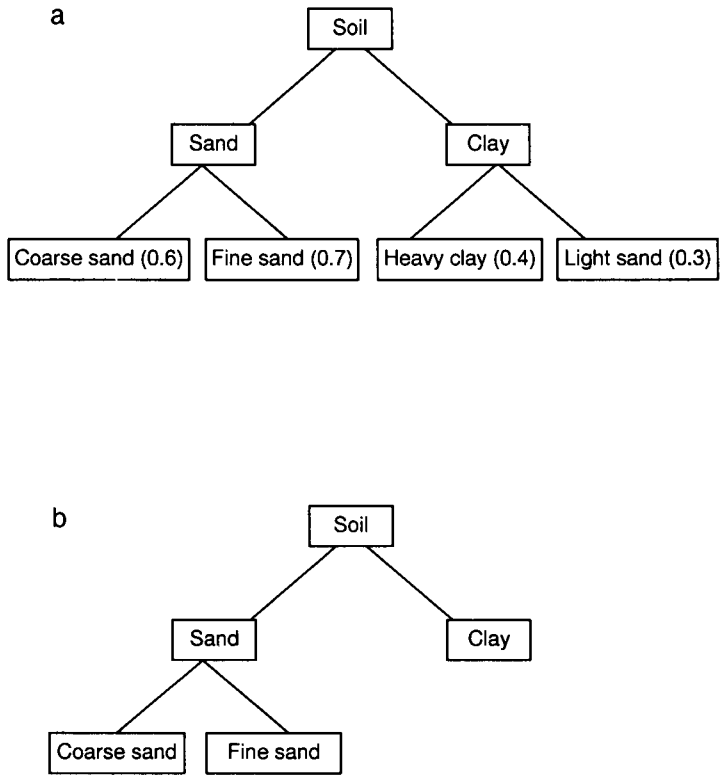


Fig 2. Classification hierarchy with importance factors
a) before generalization
b) after generalization (threashold value of importance factor = 0.5)

The importance factor varies between 0 and 1. 0 indicates that the attribute class at the lowest level is of no importance to the specific application; 1 indicates a high importance to the application.
The user can control the generalization process by specifying the threashold value of the importance factor. For instance, with a threasholdvalue of the importance factor of 0.5 all the area objects with higher factors will be displayed at the most detailed level, whereas area objects with lower values will be displayed at a higher hierarchical level (Fig. 2).
An advantage of the attribute class method is its application orientation. It also uses the natural classification hierarchy, which is present in most spatial data. A disadvantage of the method is the use of importance factors, the estimation of which requires quite a lot of expert knowledge of the data and the requirements of the application field.

Similarity method
According to the similarity method generalization is done on the basis of the geometry and attributes of area objects. Generalization is performed by evaluation of the generalization error resulting from joining two area objects. Areas with the smallest generalization errors will be joined first. The generalization error between area objects is calculated according to:

generalization $error_{a,b}$ = (1 - similarity $factor_{a,b}$) * $surface_a$

where the similarity factor is a value between 0 and 1, indicating the similarity between attribute values of areas a and b. A factor of 1 indicates no difference for a certain application, whereas a value of 0 indicates a large difference. Similarity factor values are given with a certain application in mind. This is done in practice by filling in a similarity matrix showing all the possible combinations of attribute values. In the calculation of the generalization error the surface of the area is also taken into account, which means that smaller areas have a greater chances of being joined with other areas than larger ones. Advantages of the similarity method are its application orientation and the direct insight into the generalization error. A disadvantage is the similarity matrix, which needs to be filled in for each application. This requires quite a lot of knowledge of the data set and the requirements of the application.

Quantifying the effect of generalization

In generalizing it is important to know the effects of the generalization process (Joao,1995). Only then can the user make a good decision on the generalization method and the reduction in the number of area objects.
Effect measures can be used to evaluate the effect of the generalization process. These measures allow the user to compare different strategies. Various measures can be defined. The measures used in our study are described below.

Area reduction index
The area reduction index is a measure for the reduction of the number of area objects due to

the generalization process, and is defined as:

$$\text{area reduction index} = \frac{\text{number of area objects after generalization}}{\text{number of area objects before generalization}}$$

The area reduction index is a simple measure and can be calculated for all three methods.

Surface change index
The surface change index is a measure for the change in the geometry of the area objects due to the generalization process, and is defined as:

$$\text{surface change index} = \frac{\text{sum of surfaces of removed area objects}}{\text{total surface}}$$

The surface change index can be calculated for all three methods. It is one measure for the whole data set. It is, however, also possible to calculate the changes at the attribute level. This can be done by the attribute change table.

Attribute change table
The attribute change table gives the change in surface per attribute due to the generalization. This table allows the user to evaluate the effect of generalization for the different units. Using this table the user can judge whether, due to the generalization process, there has been a strong increase or decrease in a certain type of area object. The overall change in attributes can also be presented in one measure: the attribute change index.

Attribute change index
The attribute change index gives an impression of the change in attribute descriptions of the area objects, and is defined as:

$$\text{attribute change index} = \frac{\text{sum of absolute surface differences per attribute}}{\text{total surface}}$$

If the attribute change index has a value of 0, no surface changes in the different attributes have occurred. The attribute change index can be calculated from the attribute change table.

Error index
The error index is a measure for the generalization error and is defined as:

$$\text{error index} = \frac{\text{sum of generalization errors}}{\text{total surface}}$$

The error index is based on the generalization errors, calculated for the similarity method. The error index can only be calculated if similarity factors have been appointed to the attributes. This must be done for the similarity method, but is not necessary for the other methods.
The effect measures described above characterize different aspects of the generalization process. It is the user who must decide which method and generalization intensity are most suitable for his application. The effect measures allow the user to make a sound decision on basis of quantitative information.

An application

The three generalization methods described have been implemented in Arc/Info and were applied to a fragment of the digital soil map of the Netherlands on a scale 1 : 50 000. Map Sheet 39 East was used as test area. This sheet contains the towns of Wageningen, Ede and Rhenen. The soil data have a classification hierarchy in two levels. At the highest level there are seven classes (peat, sand, clay, water, etc.). At the second level these seven classes are divided into forty subclasses. For instance, sand is divided into coarse and fine sand. The forty subclasses were mapped in the area as four hundred area objects. Similarity factors and importance factors were assigned to all forty units on subclass level by an expert in soil information. These factors were given with the application *transport of water in the soil profile* in mind. For this application the texture of the soil plays an important role. The number of area objects was strongly reduced by generalization. The number of area objects

of the 1 : 250 000 soil map of the same area was used as focal point. The database of this map contains 118 area objects.
The preliminary results of the application show that the effect measures provide good possibilities for the user of controlling and evaluating the generalization process.

Conclusions

In the last five years there has been a strong increase in the use of spatial data due to the increasing use of geographical information systems. The availability of digital spatial data offers the possibility of generating more information on demand. Generalization is one of the processing activities which can bridge the gap between user demand and data availability. To do so, the generalization process must be fast, flexible and application-oriented. Model-based generalization offers good possibilities when information is requested rather than a map. Practical implementations of model-based generalization are, however, still limited.
This paper presents three methods for model-based generalization of area objects. The application-sensitive methods, such as the attribute class method and the similarity method, are preferable to the application-independent surface method. Application of the attribute class method and the similarity method, however, requires quite a lot of knowledge of the data and the application field. Estimating the importance and similarity factors proved to be not easy in practice.
Determining and evaluating effect measures is essential to change the generalization process from a magic exercise to a clear processing activity.

References

Brassel, K.E. & R. Weibel, 1988. A review and conceptual framework of automated map generalization. International Journal of Geographical Information Systems 2:229-244.

Lee, D., 1995. Experiment on formalizing the generalization process. In: J.C. Müller, J.P. Lagrange & R. Weibel, GIS and Generalization: Methodology and Practice, Taylor & Francis, London.

Joao, E.M. ,1995. The Importance of quantifying the effects of generalization. In: J.C. Müller, J.P. Lagrange & R. Weibel, GIS and Generalization: Methodology and Practice, Taylor & Francis, London.

Molenaar, M., 1989. Single valued vector maps - a concept in GIS. Geo-Informationssysteme 2 (1):18-26.

Müller, J.C., J.P. Lagrange & R. Weibel, 1995a. GIS and Generalization: Methodology and Practice, Taylor & Francis, London.

Müller, J.C., R. Weibel, J.P. Lagrange & F. Salge, 1995b. Generalization: state of art and issues. In: J.C. Müller, J.P. Lagrange & R. Weibel, GIS and Generalization: Methodology and Practice, Taylor & Francis, London

Richardson, D.E, 1993. Automated spatial and thematic generalization using a context transformation model. PhD thesis. Landbouwuniversiteit, Wageningen.

Van Smaalen, J.W.N., 1995. Automated generalization of geographic Information. In: Workshop advanced geographic data modelling, Merrichville, Ontario, Canada, October 4-6, 1995.

Applying aggregations in policy support research at RIVM

Arthur van Beurden

Dutch National Institute of Public Health and Environment (RIVM)
P.O. Box 1, 3720 BA Bilthoven
The Netherlands
E-mail: arthur.van.beurden@rivm.nl

Abstract

The Dutch National Institute of Public Health and Environment (RIVM) deals with scientific research, on which national, regional and sometimes international policy is based. Supporting policy decision making requires certain settings for scientific research and certain ways to present results to decision makers at various levels.
This paper describes the role and position of aggregation in that process. The basic division into three research stages (pre-processing, processing and post-processing) helps to identify specific matters in answering decision questions. Aggregation issues need to be addressed in both pre- and post-processing. The huge databases are aimed at assisting the task of RIVM, but since aggregations are involved in both stages it is not always certain whether the data are really applicable to the decision question. Some examples are shown to highlight the specific geographic problems involved in aggregation for policy support.

Introduction

Environmental policy in the Netherlands is often based on maps (RIVM 1995). These maps may show locations of high or low emissions, transportation or deposition of compounds, regional concentrations and where people or organisms may be affected by such compounds. This not only goes for chemical compounds, but for associated factors as well, such as radiation, noise or virtually any risk.
To assess possible benefits from measures and legislation in the domain of environmental policy, decision makers have been used to set out generic measures, mainly aimed at emission reduction in general (RIVM 1991). Such generic policy is guided by the idea that the situation is bad and will improve by setting national or international standards and reference values. These are easily tested and exceedances will show whether or not to activate additional policy.
In recent years some main risk factors for environment have been reduced in that manner. It now becomes more important to focus on a region, assess possible risks for the environment and tackle the most serious risks. It allows the occurrence of for instance concentrations above a reference value in a certain region, because the overall long-term load may be 'safe'. This trade-off principle is seen to support sustainable development within regions.

It is therefore important for the decision maker to receive information about the state of the environment in a presentation that is tailored to a specific decision question. Presentation of maps is crucial: recent reports on environment and environmental policy have indicated the importance of maps in those reports (e.g. RIVM 1995).
This paper discusses the use at RIVM of aggregation in policy supporting mapping. The next section will dissect the process leading to policy supporting maps, identifying the role of players involved. Then, the *a priori* considerations will be addressed. The overview ends with the stage in which aggregations are usually applied: just before presentation.

Policy supporting research for regions

Policy questions
To make environmental policy, especially the policy aimed at regions, the decision makers involved pose typical questions, like the following ones.

1. Where are values exceeding certain reference values or standards? Those are the regions where policy will focus its attention.
2. Given certain planned functions in a region (like recreation, nature, urban development), are the local situations sufficient to support those functions for long periods of time (sustainability)?
3. Given a certain budget, where can 'policy' best spend its money to enhance environmental quality for a specific function?
4. Given a certain budget and set of measures, where can function best be planned to ensure sustainable development of those functions within a region?

Clearly, it is not sufficient to present maps depicting regional concentration differences, it is important to know relations between measures, what they cost and what they yield in terms of environmental quality, and the places where such measures can be implemented and the effect the have. An effect is composed of many aspects, for instance the effect on nature, on people, or even on the regional economic situation. Environmental models usually only consider socio-economic elements to a very limited extent.

Three stages in research
Goodchild (1993) identifies three important stages in any research process. These stages are present in regional policy supporting research as well.

1. In *pre-processing* the model is devised to solve a specific research problem or answer a policy question. Algorithms and levels of resolution in both time and space are chosen so that scientifically sound results can be achieved.
2. During *processing* this planned result is actually obtained.
3. In *post-processing* the scientific result is transformed to precisely match the decision maker's needs. The final map may contain a simpler spatial pattern than the original result. Aggregations play a role in this final process (see the section on post-processing).

As simple as this division seems to be, actual research practice is less simple. The main reason is that a separate research process builds on results from previous or other research. Therefore, elements of pre-processing and post-processing may affect all three stages, as will be discussed in the sections on this themes. It is especially important in the process of modelling.

Modelling
Environmental modelling can be defined as the identification of relevant objects and their relations and behaviour. Those objects need not be sharply bounded in space, relations can be uncertain or fuzzy to some extent (Zimmerman 1991). Most spatial characteristics of those

objects and relations are stored in spatial data if they are sufficiently static: location, position, topology and attributes. More dynamic elements of relations and behaviour are expressed in equations and algorithms: processes in time and space. (Beerling *et al.* 1970)
Hence, one field of expertise is to compose equations and algorithms for value computation that can be checked with observations. Since this part of modelling contains simplifications about behaviour and relations, the computed result is often not exactly the same as (subsequent) observations. The group of scientists involved almost exclusively consider their algorithms.
A quite different domain is occupied by people who focus on the more static part of the model: the representation of the objects themselves. This is stored as data and in the case of spatial characteristics as digital maps. These also contain simplifications and reality may be more heterogeneous or variable than the map suggests.
Integrating the 'products' of these two groups is a difficult task, since data need not exactly match the algorithm, or the combination need not yield the precise answer to a decision question. Geographic Information Systems (GIS) have been found to assist in the integration of those two domains (Goodchild *et al.* 1993).

Players in the research process
Burrough (1995) has discussed the traditional division between the two groups mentioned above, and also shows the benefits of GIS. Van Beurden & Douven *(in prep.)* add the decision question, since all parties should let the decision matter guide the modelling effort. This means that the decision question should define which model (both algorithm and data) is to be considered for answering that decision question and what (post-) processing is required to present results in the appropriate way.

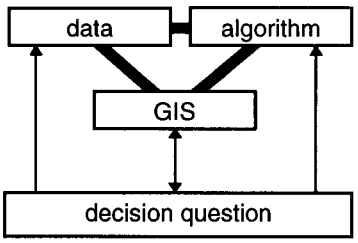


Figure 1: Relations involved in the decision supporting process (extension of Burrough 1995).

Pre-processing considerations

Data model
A model is always a simplification of reality: elements are left out and the seemingly relevant characteristics are described for the objects and their relations. However, any data will be either analysed or used with algorithms to yield some result. When data are to be used with algorithms, for example to compare policy scenarios, the data should match the requirements of the algorithm. And, *vice versa*, an algorithm should not be used on ill-fitting data.
The research procedures at RIVM contain four logical steps for data modelling.

1. First describe the process involved in the detail required for answering the decision question.
2. Next choose and define a model: it either contains spatial relations or it doesn't, and it is either static or dynamic.

3. This model is transformed into algorithms to enable computation.
4. The model characteristics also define the areal units that can be chosen. It is found that non-spatial, static models can use virtually any area of any size and shape; limitations grow when the model is spatial; there are very strict limits to size and shape when the model is both spatially intricate and dynamic. The data model is based on these limits.

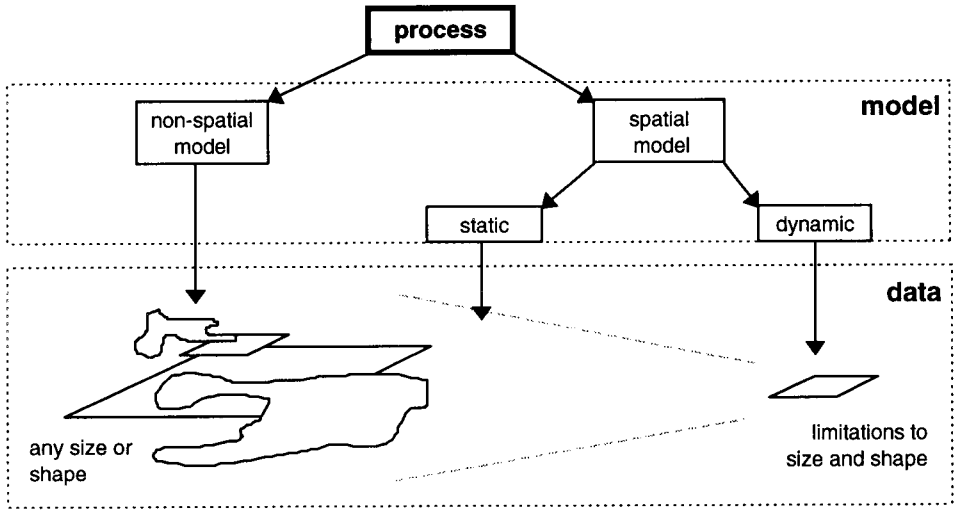


Figure 2: Choosing areal units for the data model, based on the process characterisation.

The limits are set by the model itself: a model can only be valid between size and shape limits. For example, an empirical model about chemical behaviour in the soil, tested in a soil column with a diameter of 25 cm, need not be valid for computation on soil units of a size up to 10 km^2. It is however very tempting to do just that, since storing such detailed soil data and computing them takes a lot of time. Obviously, data storage space and run-time play a role in the final choice of the areal unit.
Thus, in some cases the model (data and algorithm) that is coincidentally present is used to try and answer the decision question. Obviously, this may yield inappropriate maps. However, the GIS is capable of transforming maps (especially the result maps) to the degree where they actually make sense for the decision maker.

Aggregated input data
In some cases the data used as input for a 'model run' are themselves the result of a research process. One of the finest examples is the soil map. Observations on point locations exhibit variations over short distances. However, for the specific purposes (often agriculture) of the soil map, these observations are classified and presented as map units. A map unit contains variations, but these are not shown in the map. Soil map unit inclusions can no longer be located in the map unit.
For some environmental modelling, e.g. the leaching of compounds like nitrogen, pesticides or heavy metals, the texture variations are crucial. The soil map unit shows some form of a mean texture class, with known variability. Though these inclusions and variations may be estimated (Fisher 1993, Bierkens *et al.* 1994), it adds a touch of uncertainty to any result based on that soil map, originally compiled for a different purpose.

Post-processing

Aim of aggregation

The model result can be used to support a decision. This at least is the purpose of policy supporting maps produced by RIVM. It is of the utmost importance to present only the relevant information to a decision maker, and not include anything that might make the map more difficult to interpret. This is especially important since ministers, members of parliament and ministerial civil servants have little time for such interpretation.

The main aim of aggregation in this post-processing stage for presentation is to present the right message, the answer to the decision question. The map should therefore be *simple*, without confusing detail, and retain the *right* information so that it is still scientifically justifiable. Van Beurden & Van der Veen (1995) have demonstrated the inherently opposed targets of these two principles. However, cartographic rules can be used to find the optimal solution. Aggregation is one of the generalisation methods to obtain that optimal situation.

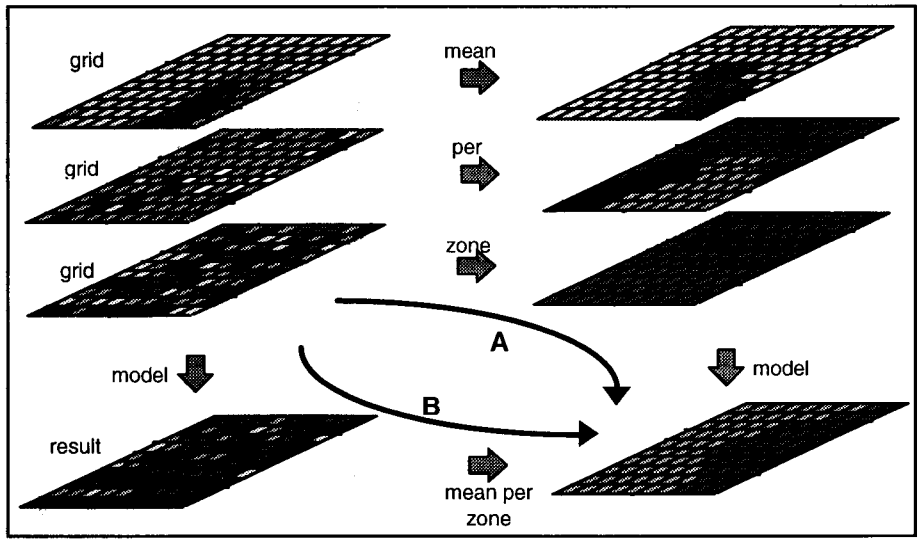


Figure 3: Approach A: first taking a mean per zone and then modelling; approach B: first computing the model and then taking the mean per zone.

Two approaches

There are basically two approaches, as shown in figure 3, to yield an aggregated result map.

1. Approach A means that all required input maps area aggregated to the new (decision) level. Subsequently the algorithms are applied and the model is 'run'. The result is a map, which shows the aggregated map units.
2. Approach B reverses that order. First, the model is 'run' to yield a result map, subsequently only this result is aggregated.

When a model is relatively simple (for instance a first order equation), and the aggregation method is simple (as is the mean in figure 3), both approaches should yield the same aggregated result. But when one or both of them are more complex, both approaches can yield different results. It then becomes necessary to examine the exact cause of the difference and evaluate what result best retains the scientific message to get across.

The main method is to compare the two results. Differences are easily spotted or might be computed with for instance a GIS. The overall image and pattern structure may be compared using the overall cross-correlation.

Figure 4 gives an example of a leaching process. For this compound (atrazine) the organic matter content of the top soil and the occurrence of the target crop are the main input maps. The result is an estimate of the groundwater concentration. The several maps show both approaches. The map codes of figure 4 are shown in table 1, which gives the comparison between maps.

Table 1: Comparison between maps: likeness.

	pec 25 nn	pec 10 a1	pec 10 b1	pec pr a1	pec pr b1
pec 10 a1	0.37	1			
pec 10 b1	0.41	0.61	1		
pec pr a1	0.28	0.42		1	
pec pr b1	0.36		0.54	0.89	1

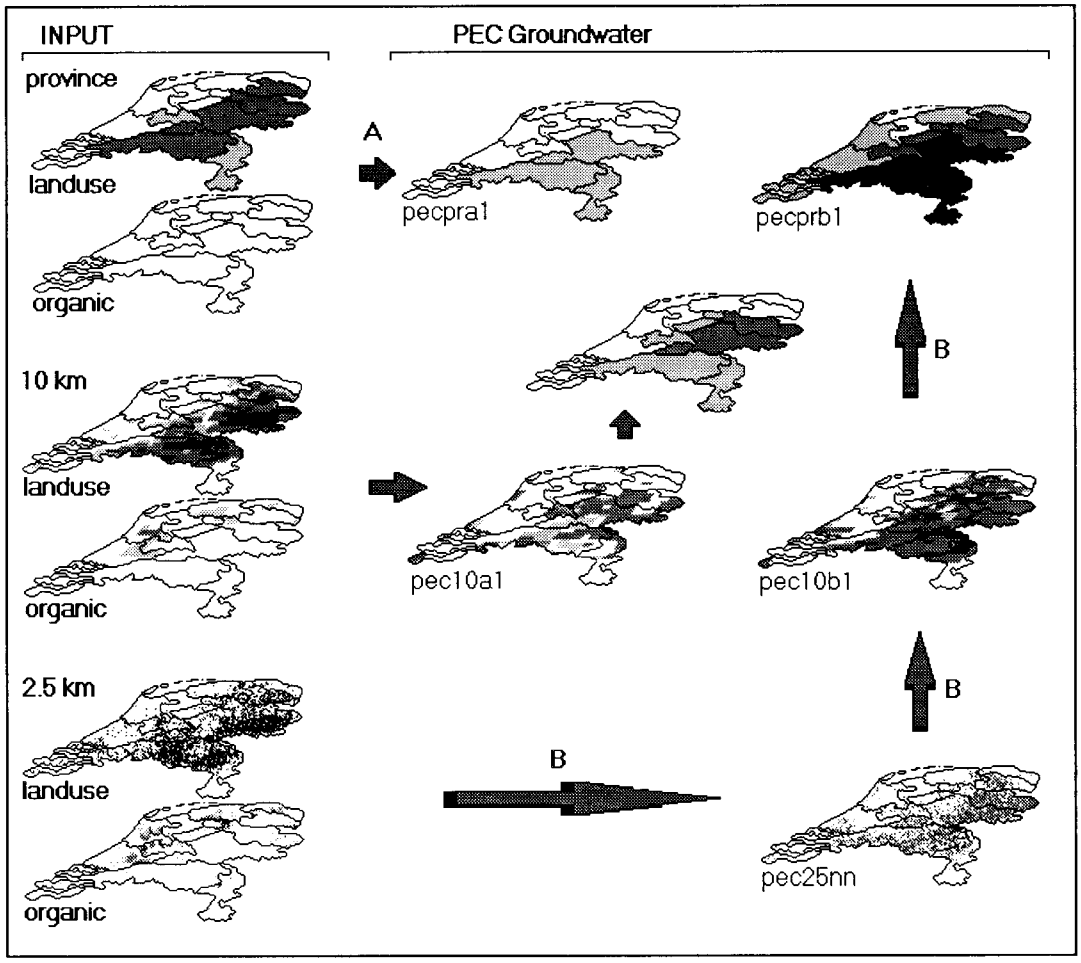


Figure 4: Aggregation results (see text above), maps are shown perspectively.

Spatial pattern
Spatial patterns may be important to the decision maker. This may be based on the trade-off principle, it may also be based on legal considerations. Not only the amount of 'spots' of high concentration in a region may give cause for action, any cluster may be interpreted as increased risk.
Consider the two maps in figure 5, depicting hypothetical extraction zones for drinking water. All groundwater within the zone will arrive at the extraction zone within 30 years. The mean per zone is equal, but the interpretation is that map 5b has a higher risk than map 5a.

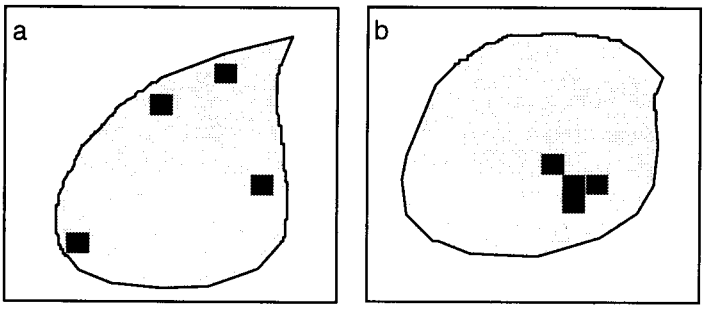


Figure 5: Two extraction zones with the same mean concentration, 5b having a higher risk than 5a.

Patterns can be quantitatively indicated per region. Every indicator technique uses specific assumptions, and therefore they can be used for specific decision support rules only. Such typical spatial data can be used in decision making (Jankowski 1995).

Multi-level consistency
For legislative purposes the requirement of comparability of decisions is becoming more important. This is usually true for admission of compounds, planning or exemptions. Suppose for instance that the European Union allows a certain pesticide on the market. A national government can still decide to ban the pesticide in that country. However, if the compound is banned on a European level, it cannot be allowed on the national level (EU directive 91/414/EEC). The importance of the differences in result maps from different aggregation approaches then becomes clear.

Concluding remarks

Aggregations are useful for presenting scientific results to decision makers, in some cases perhaps even indispensable. It can help to clarify the scientists' message. The examples presented in this paper are but a small portion of the procedures and maps used in RIVM's decision supporting processes (more is exemplified in other RIVM publications, two of which are mentioned in the references); they nevertheless show that aggregations should be used with caution. It is therefore difficult to set strict criteria to choose aggregation techniques, it may even make more sense to simply stress the alterations of the map's message when aggregation is performed on the original data. The two approaches for aggregation can be guided by strict rules, bounded however by data availability.
Aggregation touches on the subjects of choosing the appropriate areal tessellation for both computation and presentation. It is simply one of the oldest geographic problems, sometimes known as the Modifiable Areal Unit Problem (Openshaw 1981). There is just no solution for it, it

requires geographic and common sense to find the right representation. Storing such knowledge in some kind of database may assist researchers operating in an intelligent GIS environment (Burrough 1992).
The use and application of aggregation will be enhanced (both quantitatively and qualitatively) when underpinning techniques are operational in GIS. For instance, pattern indicators are as yet not too sophisticated and might be approved according to presently known theories (e.g. Cressie 1993, Isaaks & Srivastava 1989). This is but one example of techniques which are ready to be improved.

References

Beerling, R.F., S.L. Kwee, J.J.A. Mooij & C.A. van Peursen, 1970. *Inleiding tot de wetenschapsleer.* Bijleveld, Utrecht.

Beurden, A.U.C.J. van & A.A. van der Veen, 1995. *Areal units for environmental decision support, revisited.* In: Proceedings, Volume 1, Joint European Conference and Exhibition on Geographic Information, The Hague, the Netherlands, March 26-31, 1995. AKM Messen AG, Basel. pp. 292-297.

Beurden, A.U.C.J. van & W.J. Douven, *in prep. Aggregation issues of spatial information in environmental research.* Sent for acceptation.

Bierkens, M.F.P., H.T.J. Weerts & P.A. Burrough, 1994. *Geostatistical characterization of a complex semi-permeable layer.* Engineering Geology of Quaternary Sediments (Rengers, ed.). Balkema, Rotterdam. pp. 73-92.

Burrough, P.A. 1992. *The development of intelligent geographic information systems.* Int.J. GIS, Vol.6, no.1, Taylor & Francis, Washington. pp. 1-11.

Burrough, P.A., 1995. *Opportunities and limitations of GIS-based modeling of solute transport at the regional scale.* Proceedings *'Applications of GIS to the modeling of non-point source pollutants in the Vadose Zone'*, ASA-CSSA-SSSA Bouyoucos Conference, Riverside CA, May 1-3, 1995. 19p.

Cressie, N.A.C., 1993. *Statistics for spatial data, revised edition.* Wiley Series in Probability and Mathematical Statistics, Applied Probability and Statistics. John Wiley & Sons, New York.

EU Directive 91/414/EEC, 1991. Council directive concerning the placing of plant protection products on the market. Official Journal of the European Communities, L239.

Goodchild, M.F., 1993. *The state of GIS for environmental problem-solving.* In: M.F. Goodchild, B.O. Parks & L.T. Steyaert (eds.), *Environmental modelling with GIS*, Oxford University Press, New York.

Goodchild, M.F., B.O. Parks & L.T. Steyaert (eds.), 1993. *Environmental modelling with GIS*, Oxford University Press, New York.

Isaaks, E.H. & R.M. Srivastava, 1989. *Applied Geostatistics.* Oxford University Press, New York.

Jankowski, P., 1995. *Integrating geographical information systems and multiple criteria decision making methods.* Int.J.GIS, Vol.9, no.3. Taylor & Francis, Washington. pp. 251-273.

Openshaw, S., 1981. *The modifiable areal unit problem.* Catmog Series, no.38. Geo Books, Norwich.

RIVM, 1991. *Nationale Milieuverkenning 1990-2010.* (National Environmental Outlook 1990-2010.) Samsom, Alphen a/d Rijn.

RIVM, 1995. *Milieubalans 95. Het Nederlandse milieu verklaard.* (Environmental Balance 95. Explaining the Dutch environment.) Samsom, Alphen a/d Rijn.

Zimmerman, H.-J., 1991. *Fuzzy set theory and its applications.* 2nd edition. Kluwer Academic Publishers, Dordrecht.