# Comparison of different approaches to combine road generalisation algorithms: GALBE, AGENT and CartoLearn

**Sébastien Mustière and Cécile Duchêne**

Institut Géographique National
COGIT Laboratory
2-4 av. Pasteur
94165 Saint-Mandé CEDEX
FRANCE
Tel : (33)-1-43-98-80-03 / Fax : (33)-1-43-98-81-71
sebastien.mustiere@ign.fr, cecile.duchene@ign.fr

## Abstract

In this article we discuss three different approaches to chain generalisation algorithms, in the special case of road generalisation. The three approaches consider that generalisation is done by means of applying different algorithms on different parts of the road. That is they consider generalisation as a step by step process, where a pertinent working space must be found to apply algorithms. They also use the same basic tools to transform and split a line. They differ in the way to choose what to do at a time of the process and in the way the process steps are chained until reaching a final state. The first approach, GALBE, is a predefined sequence of algorithms only guided by a coalescence criterion. The second one, AGENT applied to roads, is based on multi-agent and constraints principles. The third one, CartoLearn applied to roads, is a knowledge based system where knowledge bases have been automatically learnt from examples. We theoretically compare the three different processes in terms of the way objects and knowledge are represented, the way decisions are taken and the way actions are chained.
The AGENT process concentrates more on the engine guiding the whole process and the CartoLearn process concentrates on the knowledge used to choose an algorithm at a time of the process. As the empirical results obtained by these three processes acknowledge the global step by step approach, we conclude by proposing directions to merge the different processes in order to combine their advantages.

**Keywords** : Road generalisation, multi-agent, machine learning, knowledge based systems.

## 1. Introduction

A lot of research has been done to develop generalisation algorithms. If there is still a need for improving existing algorithms and developing new ones, in this work we focus on the question of the combination of these algorithms. This has been a central question in many generalisation models [Brassel and Weibel 88 ; McMaster and Shea 92 ; Ruas 99 ; Lamy et al. 99]. In this article we discuss three different approaches to chain algorithms, in the special case of road generalisation.

The three approaches have been developed at COGIT laboratory and have not been independently developed, which explains that some parts of them are so similar. They rely on the previous works on road generalisation done in the COGIT during the 90's [Ruas & Plazanet 96 ; Plazanet 96 ; Fritsch 97 ; Lecordix, Plazanet & Lagrange 97]. These works led to a set of measures, transformation algorithms and a platform dedicated to road generalisation. They also led to a first formalisation of cartographic rules, and some guidelines to know how to chain these basic tools.

In 1997, a test asking to cartographers to interactively use algorithms on different platforms in order to perform generalisation has been done by the OEEPE working group on map generalisation. Its careful analysis enables to learn a lot about how the performed generalisation [Ruas 98]. It shows that generalisation can be considered as a step by step process where several operations must be applied on each object, and where operations are performed at different levels of analysis : an individual object of a database, a group of spatially organised objects or even a part of an object (e.g. one bend of a road). It also shows that there is a correlation between the (visually described) kind of conflict handled at a time and the chosen algorithm.
On these bases, research has been performed to automate the choice of actions in a generalisation process. How to combine basic algorithms ? Which algorithms to apply ? In which order ? When to stop ?

As a result, the GALBE algorithm [Mustière 98] is a first combination of these basic tools to create a fully automated process for independent road generalisation. The efficacy of this process, used in an IGN-France production line, confirms the idea that the general approach is promising. Lessons learnt from GALBE and from the work on strategies done by [Ruas 99] pushed to develop two research directions :

- AGENT-Road [Duchêne 2001]: this work has been done during the European AGENT project (ESPRIT/LTR/24939, cf. [Lamy et al 99]), it uses multi-agent principles and a constraint based approach to represent user needs as proposed in [Ruas 99]. Relatively to GALBE, it reformulates GALBE rules in a constraint formalism and it looks for developing a more powerful and flexible engine guiding the algorithms combination.
- CartoLearn-Road [Mustière 2001] : this work explores the interest of using machine learning techniques to acquire knowledge necessary to guide generalisation. Relatively to GALBE, it looks for developing more powerful rules guiding each step of the process (i.e. rules to choose which algorithm to apply) but it does not look for improving the engine combining the steps.

For the sake of simplicity we will not explain here the algorithms parameter choice, even if this problem is handled in the described processes.

## 2. Basic algorithms used

The three processes hereafter described use the same basic algorithms, even if they differently chain them. Six basic algorithms are used to transform (smooth or caricature) either a part of a line or the whole line, one algorithm is used to split the line according to the coalescence, and another one is used to propagate side effects due to local algorithms transformations. These algorithms are illustrated in Figure 1. We very briefly describe hereafter the effect of each algorithm :

- *Plaster* algorithm [Fritsch 97] enlarges sharp bends and smoothes slight bends of any line.
- The *Gaussian smoothing* algorithm smoothes all the bends of a given line.
- *Accordion* [Plazanet 96] stretches bend series to make each bend distinct from the others.
- *Schematisation* [Lecordix, Plazanet & Lagrange, 97] removes two bends from bend series, in order to provide space for other bends.
- *Maximal Break* [Mustière 98] enlarges one bend and keeps its shape as much as possible.
- *Minimal Break* [Mustière 98] minimally enlarges one bend in order to make it legible.
- *Coalescence Based Splitting* [Mustière 98] splits a given line into parts with either no coalescence, or coalescence on one side, or coalescence on both sides.

- *Propagation* algorithm propagates the displacement of some parts of the line to the whole line. The strength of the displacement slightly decreases along the line.

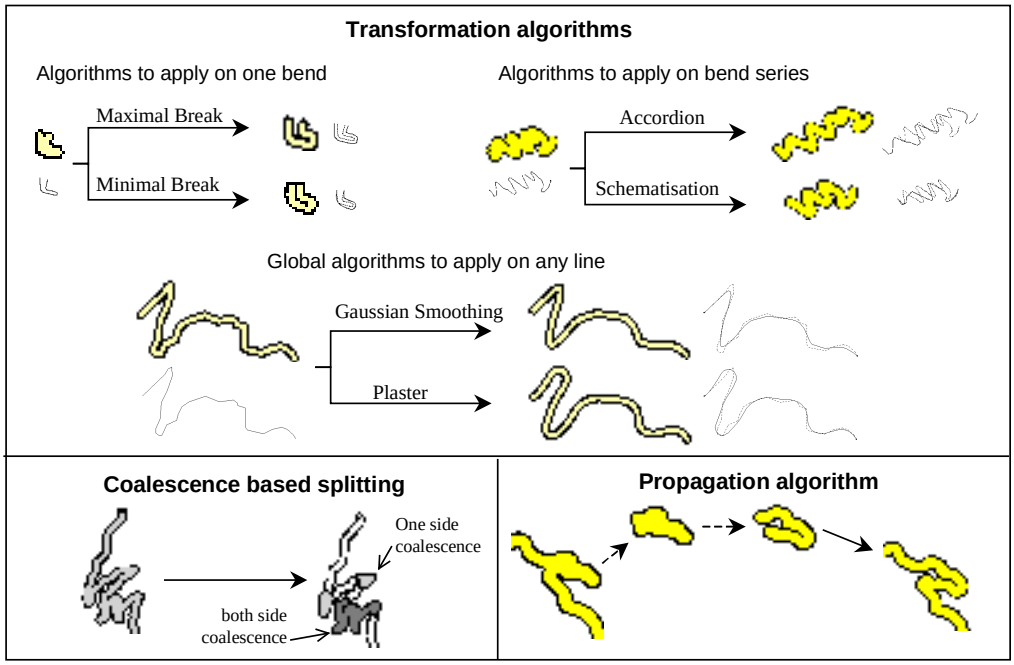


Figure 1. Basic transformation algorithms used

# 3. Description of the three approaches

## *3.1. GALBE [Mustière 98]*

GALBE is a first attempt to fully translate our knowledge about generalisation rules, measures and algorithms in terms of a process guiding algorithms application. It is a predefined combination of algorithms choice guided by tools based on coalescence analysis (GALBE is a French acronym that stands for Adaptive Linear Generalisation Based on Coalescence). It alternatively applies splitting, caricature and smoothing algorithms. After each algorithm application on one part of the line, side effects due to the displacement of its extremities are propagated to the whole line. The process has been empirically developed and is summarised in Figure 2.

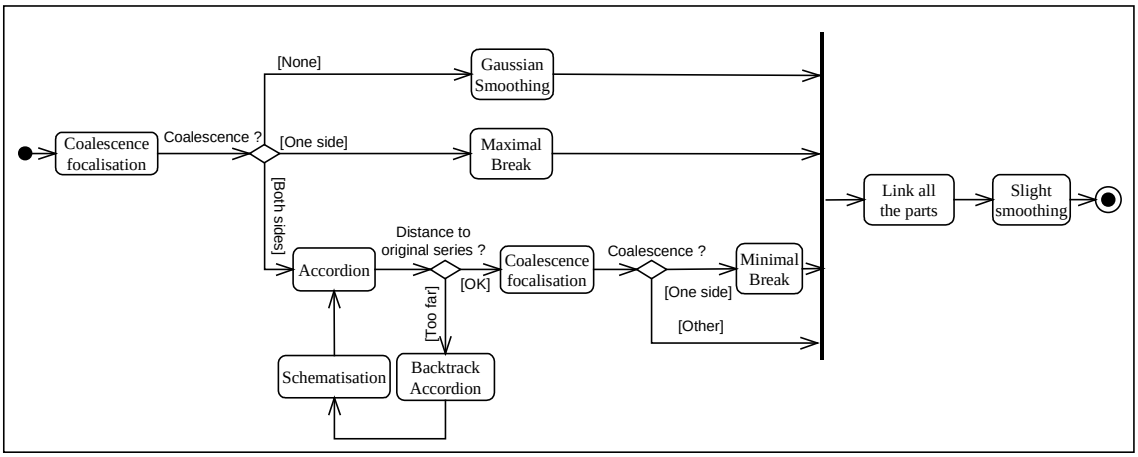


Figure 2. GALBE

### *3.2. AGENT-Road [Duchêne 2001]*

The principles of the generalisation prototype set up by the AGENT project have been described more in detail in [Ruas 99, Lamy et al 99]. We just give a brief description of them applied to roads. The described process is globally generic to any kind of geographic objects but, as research is still ongoing, some small differences exist between the global AGENT model and the model presented here for the roads.

#### ***Agents and constraints model***

Geographical entities are designed as agents, i.e. objects that have a goal and a certain autonomy to reach this goal. Each geographical agent is guided by a set of "constraints" objects that act as advisers. The constraints are used to represent the user needs. Each of them is watching on a particular character of the object and proposing possible plans (i.e. generalisation algorithms) to solve the constraint violation (plans are weighted but we do not detail that). The goal of the agent is to satisfy as well as possible all its constraints. For that it has the capacity of choosing one plan amongst those proposed by the constraints, applying this plan, and evaluating the improvement. To choose the *a priori* best plan, the agent first chooses a constraint to solve and then a plan proposed by this constraint. The *a posteriori* evaluation is a key of the process and controls the evolution of the objects, backtracking when an unacceptable degradation occurs. Moreover a "hill climbing" mechanism, which enables to backtrack to any state and try other plans, ensures that the system reaches the best possible solution according to its evaluation criteria [Regnauld 01]. Figure 3 shows the generic life-cycle of an agent when it is activated.

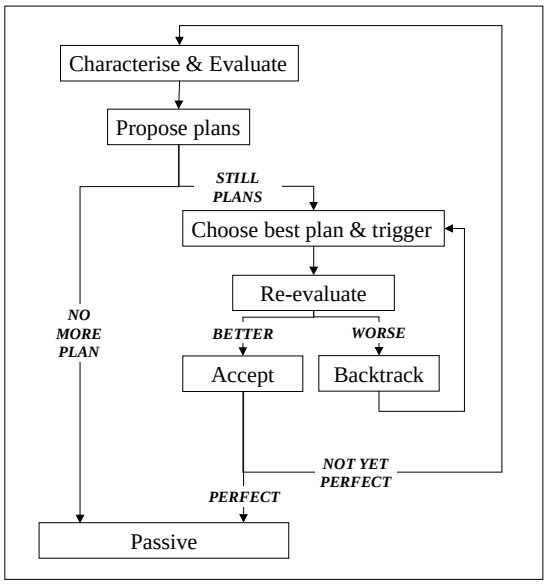


Figure 3 - Generic life-cycle of an agent

Three kinds of constraints have been identified for roads:

1. The coalescence constraint triggers the generalisation. According to the side and strength of coalescence detected it proposes:
   - Heterogeneous coalescence: generalise by parts (splitting and supervised generalisation as explained below), plaster
   - both-sides coalescence: accordion, schematisation, plaster
   - One-side coalescence: maximum break, minimal break, plaster

2. Three constraints aim to preserve characters: they do not propose any action but can lead to refuse a transformation during the re-evaluation stage (constraints for positional accuracy, internal topology, absence of “holes” in the symbol).
3. One constraint advises the road against triggering certain algorithms: a constraint describing the available space around the object, that advises against the algorithms that need too much space. This kind of constraint is not taken into account for the re-evaluation.

Other constraints could be added, such as a “noise constraint” that would trigger a smoothing. The only pre-requisite to introduce a constraint is to have a reliable and interpretable measure describing the related character.

***Multiple levels of analysis***

In the AGENT approach several levels of geographical analysis have been distinguished, in order to take into account the significance of spatially organised groups of objects. The lowest level of analysis is called the “micro” level. The upper level (level of the groups of organised features) is called the “meso” level. Specific engines are available for micro and meso levels.
For the roads, the micro level initially contains the single road arcs, as they are defined in the database. Any micro road can either generalise itself with a global algorithm, or split itself into homogeneous parts so that the coalescence conflicts can be handled separately thanks to specific algorithms (GALBE philosophy). Translated into AGENT principles, a split road becomes a group of homogeneous parts of the road, i.e. a “meso” object, and the parts become the new micro level. The temporary “meso-road” agent manages the generalisation of its parts, ensures the preservation of their connectivity and then recomposes itself back into a micro-road. This process is recursive: a micro-road stemming from a decomposition can decompose itself again if it becomes heterogeneous. More, a road can be decomposed and re-composed several times, according to the evolution of its heterogeneity, until reaching a final state.

### *3.3. CartoLearn-Road [Mustière 2001]*

Contrary to the AGENT-Road process, CartoLearn-Road does not propose a sophisticated engine. Instead, it tries to improve the rules used in the engine. In GALBE the rules, encoded as the branches in the process, are very basic. CartoLearn-Road is issued from the idea proposed by [Weibel, Keller and Reichenbacher 1995] to automatically learn generalisation rules from examples. These rules have been determined with the RIPPER learning algorithm [Cohen 95], from a set of 120 examples of roads. Each example contains a set of automatically computed measures to describe the road, an interactively determined abstract description of the road (e.g. "long", "sinuous", "slightly coalesced"…) and the interactive choice of which algorithms are applicable and which algorithm is the best to apply.

The detail of the learning process and the learnt rules is out of the scope of this paper. To shortly describe the rules, the system contains four distinct learnt knowledge bases (KB). The first KB links the measures to the abstract description of the road (rule example : "if length > 300m then Size = long"). The second KB links the abstract description and the previous operation done to the operation to be done ("if coalescence is high then caricature"). The third KB determines the applicable algorithms ("if Size = long then Plaster is applicable"). The last KB determines the algorithm to apply ("if Coalescence = Few and Plaster is Applicable then use Plaster").
In this process, stopping the generalisation or splitting the road is considered as an operation in itself like any transformation operation. The knowledge bases are thus used to determine when to stop or when to split the road as well as to determine which algorithm to apply. A

relatively close learning process used to learn rules to guide building generalisation has been presented in [Mustière, Zucker and Saitta 2000].
The CartoLearn-Road process is summarised in Figure 4. It must be noticed that, out of the "Choose algorithm" step, the engine steps are very simple : the "Select object from list" chooses the first one, and the "Add parts to list" step just puts them at the end of the list.

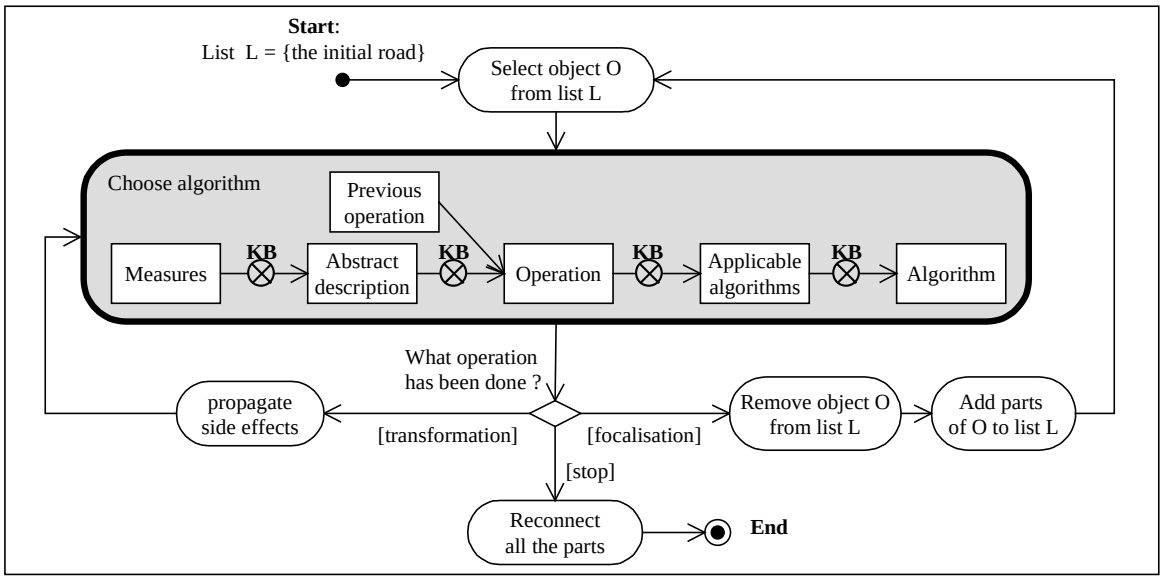


Figure 4. CartoLearn-Road

# 4. Theoretical comparison of the methods

## *4.1. Objects and knowledge representation*

### ***Knowledge representation***

Knowledge used to link measures describing an object and algorithms to apply are complex to acquire, to manage and to maintain. The three processes use different approaches to handle this complexity.
GALBE makes the choice of simplicity : it uses a minimal number of measures to be able to manipulate them easily.
AGENT makes the choice of organisation : it uses templates (the constraints) to guide the measures representation and use. Each constraint contains one measure to describe the character it is considering, one goal value, and some other attributes like the priority or the flexibility of the constraint.
CartoLearn makes the choice of automation : knowledge is automatically built from examples. Any measure can be used to describe an object, the machine learning process is automatically dealing with all these measures to determine rules to link measures to algorithms. However, the learnt knowledge is also organised in order to be understandable: different types of knowledge are located in different rule bases.

### ***Characters and constraints***

GALBE only uses a coalescence measure to describe objects conflicts and the hausdorff distance to describe the planimetric displacement of an object. It thus uses two characters (the coalescence and the position), and the constraints on these characters (removing coalescence

and keeping accuracy) are represented by the means on tests on these measures during the process.
In AGENT constraints are explicitly represented and the considered characters are only the one related to constraints that must be satisfied by the object. Constraints can possibly represent comparisons of the object to its initial state.
In CartoLearn any character describing the object can be considered, be it related to a constraint or not (e.g. the length), but no comparison to the initial object is used.

***Generic and task-specific knowledge***

The three process do not represent knowledge specific to one generalisation task (here generalising roads) the same way. In GALBE knowledge are mainly located in the process itself. For AGENT and CartoLearn the global engines are as much as possible independent from the specific task. In AGENT most parts of specific knowledge are located in the constraints, but the constraints organisation and the engine chaining actions are generic. In CartoLearn specific knowledge are located in the learnt rule bases, but the engine and the rule bases organisation are generic. The choice to concentrate knowledge in predefined parts of the system is sometimes a limitation but is a usual choice in knowledge based systems to ensure a good maintainability of systems.

***User's needs representation***

Amongst the three methods, AGENT is the only one where the user's needs are explicitly represented, in terms of goal values to reach for the constraints. As the constraints are explicit and separated from the engine, the AGENT model enables to take different user's needs into account by changing the goal values of the constraints (or their relative importance), without modifying the engine.
On the contrary to AGENT, in both GALBE and CartoLearn the elements that constrain the generalisation have not been isolated from the engine. In GALBE only the thresholds on the tests in the process can be tuned in order to obtain different results. But there is no means in the GALBE model of preserving first the positional accuracy or the shape. In CartoLearn, the user needs are not explicit either. They are taken into account by means of the examples used to learn the rules: those examples have been generalised to satisfy given user needs, and the rules learnt from those examples enable to generalise other features to make them fit to the same needs. To adapt the CartoLearn process to other user needs, either new examples fitting to these needs should be set up and the associated knowledge bases should be re-computed, or the knowledge bases should be manually modified.
The three processes are anyway relatively adaptable to different scales as the used measures are related to the used symbol width.

### *4.2. Choosing the next object to generalise*

The three methods use a line segmentation algorithm and thus have to handle several parts of road after splitting. Moreover, several nested splittings can occur, i.e. one part of road stemming from a splitting can be split again. Several questions arise:

- In case of nested splittings, do we model several nested levels, or two (the level of the whole line and the level of the parts, whatever their origin), or only one (the lower parts level) ?
- Within a given level, in which order to handle the parts?
- In case of nested splittings, when splitting a second time, do we first handle the parts that have just been split (depth first) or do we first handle the parts of same level and then the newly split parts (breadth first)?

***Number of considered nested levels***

In GALBE only two nested splittings are possible, when a bend series is isolated and stretched by accordion and then split into bends. The two levels are distinguished and handled separately. In CartoLearn only one level is modelled, even if the number of nested splittings is unlimited: all the parts are at the same level. When a part is split, it is destroyed and replaced by all the "sub-parts". In AGENT all the nested levels are recursively modelled: when a part is split again, it is turned into a meso-road, which is linked to the (micro-road) sub-parts.

***Intra-level objects order***

Concerning the order in which the parts of a same level are handled, in GALBE it depends on the coalescence type (two sides-coalescence then one side-coalescence then the others). In CartoLearn no order is computed, the objects are put in a list with no particular order at the splitting time and then handled in the order of the list. In AGENT, for a given level at any time the next best candidate is dynamically computed on coalescence type and strength criteria. The possible candidates for the next step are the parts which are not yet in a perfect state, and which have not just been handled without reaching a perfect state. Moreover, a same part can be handled several times: in the case where it is damaged by a propagation of another part, it becomes again a possible candidate for next step. Thus a same part can be split and merged back differently several times.

***Inter-level order***

When splitting a second time, in GALBE the resulting parts are handled immediately (depth first), before the other parts of level one. All the parts are merged back only at the end. In CartoLearn, when a part is split again the resulting "sub-parts" are pushed at the end of the list, so that they will be handled after the other parts of the same splitting level (breadth-first, on the contrary to GALBE). As in GALBE, all the parts are merged back only at the end.
In AGENT, in case of splitting of a part the "sub-parts" are handled immediately (depth first) as in GALBE, but the parts are immediately merged back after treatment contrary to GALBE and CartoLearn.

### *4.3. Choosing the next algorithm to apply on an object*

***From which information ?***

In GALBE any state is a node of a predefined process, so that the choice of the next action depends on all the previous ones. In AGENT it only depends on the current state. In CartoLearn it depends on the current state plus the last performed operation (AGENT and CartoLearn are markovian processes).

***Why choosing an action ?***

The description of the object is used to choose the algorithms to trigger. In AGENT, an action is explicitly chosen in order to solve a given conflict of the object, whereas in GALBE and CartoLearn an action is chosen in given conditions (these conditions are the characters values and the last triggered action). Thus, in GALBE and CartoLearn a character which is not constrained can be used to guide the choice of the next action, which is theoretically not foreseen by the AGENT model. To bypass this, a "special" type of constraint was created in AGENT-Road as explained in paragraph 3.2 (constraint describing the available space around the object). But it is on the fringe of the model since it does not respect the genericity of the constraints behaviours.
Another consequence of the AGENT constraints philosophy is that an algorithm can only be triggered if (1) it aims to solve a precise conflict and (2) a measure exists to detect this

conflict. Thus in AGENT, no smoothing operation is triggered for the time being: the smoothing aims to solve granularity conflicts, and no robust measure of granularity has been introduced in the process at this time. On the contrary, in GALBE and CartoLearn, the smoothing can be triggered after re-connection of several parts of road, or when there is no coalescence.

***How to choose an action ?***

Choices made by GALBE are based on simple tests on the type of coalescence. AGENT and CartoLearn both use the intermediate notion of possible algorithms before choosing the *a priori* best one. In AGENT the computation is done in two stages: first the unsatisfied constraints propose possible algorithms to improve their current state. This stage uses knowledge specific to each constraint. Then the agent uses a generic choice process that chooses first the constraint to handle, then the algorithm. CartoLearn first determines the type of operation to perform, then the possible algorithms to do it according to the object, then the best one, by means of learnt knowledge bases.
It must be noticed than in all the processes splitting is an action chosen like any other action, even if the management of the effects of this particular action is special.

### 4.4. Validating and continuing the action

Several question arise about the validation : how and when to validate an action ? what to do when it is invalidated ? when it is validated shall we stop the process or continue ? when it is validated, how to manage side effects ?

***The validation stage***

Amongst the three methods, AGENT is the only one that uses a continuous control of the operations. A validation stage is systematically performed after each triggered algorithm, in order to control the improvement of the handled constraint and avoid irreversible degradations. In GALBE the only operation which is thrown back into question is the accordion, when the distance to the initial geometry is too big. In CartoLearn the absence of validation is part of the background: one aim of CartoLearn was to set up knowledge bases that find the a priori best algorithm, in a context where no validation process is available (however as a pre-requisite for GALBE and CartoLearn the algorithms are supposed not to be bugged, so simple tests exist that detect bugged results and backtrack them).

***Backtracking***

The possibility of backtracking an algorithm is closely linked to the existence of a validation process, even if they are two independent components of the model. Except the case of accordion for GALBE, only AGENT enables to backtrack operations. Two kinds of backtracks are made. The first one is made when the result is evaluated as “worse than the previous state” by the validation stage. The second kind of backtrack is part of the “hill climbing” process [Regnauld 2001] which tries a lot of algorithms combinations until reaching a perfect one or having no more combination to try (and then keeps the best found).

***Stopping criteria***

The question of stopping the generalisation process covers two issues: when to take the decision of stopping and how to ensure the convergence of the system?
In GALBE the sequence is pre-defined and has a finite number of steps, so the process stops at the end of the sequence and there is no risk of non-convergence. On the contrary, both CartoLearn and AGENT are markovian processes for which the two questions are important.

Concerning the decision to stop, in AGENT, at any splitting level the generalisation of one part stops either when a perfect state is reached (which is decided by the evaluation stage), or when all the possible algorithms have been tried. Thus, in the case where no perfect solution exists, all the solutions will be tried to determine the best one, because the process does not know *a priori* when no better result can be reached. In CartoLearn, stopping generalisation is considered as an operation, exactly in the same way as smoothing or caricature. So during the learning of when to apply which operation, the system has also learnt when to stop. It means that on the contrary to AGENT, CartoLearn is supposed to know *a priori* when it is impossible to improve the current state.
Concerning the convergence, both AGENT and CartoLearn present risks of non-convergence: AGENT because a same part of road can be triggered several times, CartoLearn because nothing ensures that a rule ordering to stop will be triggered. Both systems have a basic mechanism to prevent this risk of divergence based on a maximum number of action to perform on an object. But in practice this maximum number is rarely reached.

***Side-effects management***

Some of the used algorithms displace the extremities of the road, disconnecting it from its neighbours. The propagation algorithm presented in part 2 is used to reconnect the neighbour parts to the moved extremities and propagate the displacement. The arising question is to choose if we propagate side effects just after the modification of any part, or if we separately handle each part and globally deal with side effects at the end. The three methods propagate the side-effect after each triggered algorithm.

### *4.5. Computation complexity*

***Computation time***

Comparing the rough computation time would not be very meaningful since the three systems do not run on the same platforms. Instead we can compare the number of operations triggered to reach a solution. In GALBE, few backtracks are allowed and only two splitting levels are possible. So less than three algorithms are triggered on most parts of the line. CartoLearn allows no backtrack at all, but neither the number of splitting levels nor the number of algorithms triggered on one part are limited. Thus it is slower than GALBE. In AGENT the number of splitting levels is also unlimited, and the “hill climbing” mechanism can perform many backtracks. AGENT is slower than CartoLearn and GALBE. AGENT and CartoLearn also need several measures to describe an object at any state which is time consuming, particularly for CartoLearn which uses many measures.

## 5. Empirical comparison

Let’s first present images of good results of the three processes to show that they are globally efficient, which make us definitely believe in the efficiency of the general approach (cf. Figure 5). But of course all the results are not so good and the problems stand in the particular cases.

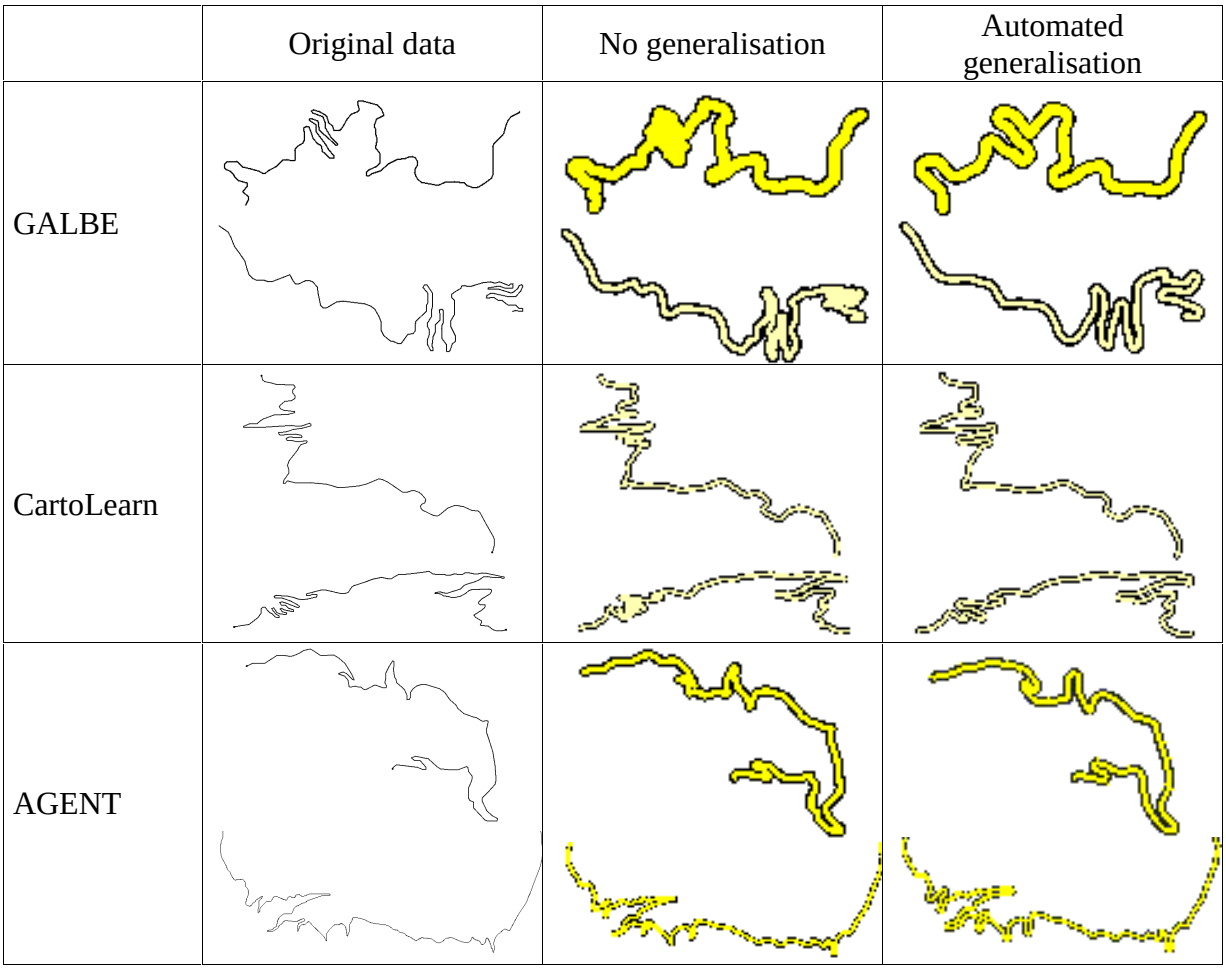


Figure 5. Some results of the three processes

The three processes have been intensively empirically tested in flat and mountain areas, but we did not have enough time yet to make an intensive empirical comparison or the three processes on some big data sets. However Figure 6 shows the results obtained by the three methods on one road.

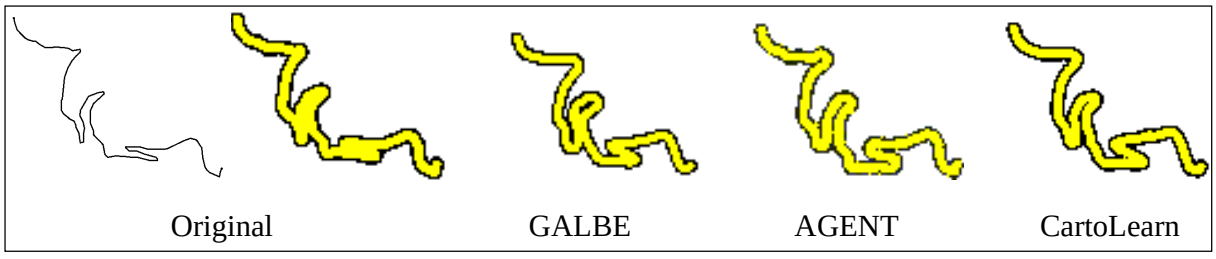


Figure 6. Results of the three methods on one road

For the time being we can say that GALBE as the advantage of speed over the two other processes. It globally provides good results but some particular roads are not sufficiently well generalised. This is due to the lack of flexibility of the model and the absence of validation stage: the GALBE result of Figure 6 presents a hole in the symbol, which is cartographically unacceptable but has not been detected since not validation has been performed.
AGENT also provides good results, it is dealing well with more particular cases than GALBE because it can try different solutions on one road. But it is clearly missing the smoothing operations, as shown by Figure 6.

CartoLearn is also dealing well with more particular cases than GALBE, because it is using stronger rules to guide the process. But it is clearly missing the opportunity to try and compare different solutions like in AGENT. It clearly appears on Figure 6: the solution found by CartoLearn is acceptable but AGENT has found a better one from the coalescence point of view, because it has tried and compared several solutions. As a result, AGENT and CartoLearn are not generalising perfectly the same particular cases.

## 6. Toward the unification of the approaches

Both AGENT and CartoLearn have been derived from GALBE with different focus issues, so that they do not have the same strengths and weaknesses. The idea now would be to unify them in a new process keeping the best of each approach.
The main strength of CartoLearn lies in the power of its knowledge bases to choose an algorithm to apply at a time of the process, including the choice of when to split and when to stop. Then it is not constrained in term of the number and type of measures to use. It also can take advantage of the information about the previous information made. Another strength of CartoLearn is the way the process is built from examples, but in this paper we concentrate on the results of the processes and not on their construction.
The first strength of AGENT lies on the generic management of constraints by means of a powerfull engine including a dynamic validation stage, a dynamic choice of the next object to consider and a management of backtracks. Another strength lies on the explicit representation of user needs.

They key idea leading to a unification of the approaches would be to keep the model and engine of AGENT and to add in it extracts of the CartoLearn knowledge bases to optimise some of the choices made by AGENT. Some of these addings are quite straightforward, others would ask for modifications in the AGENT model to efficiently use the knowledge extracts.

### ***Validation and stopping criterion***

Because measures describing an object are for the time being not efficient enough to perfectly choose when to stop the process, it would be interesting to use the parts of CartoLearn knowledge bases deciding when to stop. Actually, these knowledge make a great use of the information about the last performed action. This is particularly the case when a smoothing is done. In fact, because we do not know if the granularity of a part of the line is too high or not, rules of CartoLearn consider that when a smoothing has been done and no coalescence remains it is time to stop the process on the considered part. This could easily be added in AGENT if the last performed action is kept in memory.
Another possible improvement would be not to start the hill climbing process when CartoLearn rules consider it is time to stop. If CartoLearn thinks the hill summit is impossible to reach, do not try to climb it.

### ***Choosing an algorithm***

CartoLearn uses characters which are not constraints to choose the algorithm to apply on a part of the line. For example it does not advise to use the Plaster algorithm when the line is small. This could be added to the AGENT process but this asks for modifications on the model, which does not allow for the time being characters to be explicitly represented on the object outside a constraint. But this is not a mistake of the model, this is a choice. Several models have been envisaged in order to use characters to choose an algorithm in AGENT. But the genericity and maintainability pushed in the direction of the "constraint-only" choice. A

detailed analysis of CartoLearn rules could help to understand more which and when information is useful to choose an algorithm. Such an analysis could help to efficiently refine the AGENT model to introduce more characters inside or outside of the constraints, while keeping a good compromise between, on the one hand, the capacity to handle complex rules and, on the other hand, the genericity and maintainability of the model.
A very simple (but efficient ?) solution would be to introduce in AGENT the knowledge bases as they are and make the agent step "choice of algorithm" take into account the advice made from the knowledge bases, as well as the advice made by the constraints.

## 7. Conclusion and research directions

The results provided by the three processes encourage us in thinking as proposed by [Ruas 99] that generalisation can be seen as a markovian step by step process, where the search for the right working space is of first importance. It also encourages us in thinking that the coalescence criteria is well adapted to guide the road generalisation.
As we explained, several approaches can be derived from these ideas. Both are making a compromise between the efficiency and the complexity of the process. Both are also dealing differently with the missing tools. Missing knowledge about algorithms condition of use is bypassed by AGENT by testing several algorithms combinations. Missing measures to know when to stop the process is bypassed by CartoLearn by using information about the last performed action.
The CartoLearn approach is a good way to automatically learn rules from examples when we do not know how to create these rules. But it has been possible because the engine used by it is not too complex. Introducing learnt rules in a more powerful engine is not always straightforward. At least the study of the content of the learnt rules can help us to understand the generalisation process and then improve generalisation models like the AGENT one.

We limited our study to three relatively close processes using relatively close ideas and tools. We should make some comparison of our processes to other ones using a completely different approach to learn more about the efficiency and lacks of our approach. For example Harrie and Sarjakoski [2000] propose to generalise roads by using least square adjustment methods that could be compared, empirically and theoretically, to our processes.

The different tests of the processes led us to identify several important research directions.
Whatever the process, a good description of the objects is needed. If we want to deal with a particular aspect of an object, we must be able to describe it by the means of measures. In our case we mainly missed measures to describe lines granularity and shape, to be able, for examples, to give priority to the shape over the position. These concepts are not easy to define and even less easy to compute and manipulate. Describing objects (including shape and granularity) by means of measures has been the purpose of many works, this allowed us to make these processes. But research on this area must continue.
Another problem providing us many difficulties in these processes was the stopping choice. It is of first importance to know, given any object, if it is well generalised or not. For the validation also, we particularly missed measures to describe the shape modification of an object. We think that much efforts on this question must be done to develop more powerful generalisation processes.

## References

**Brassel K. et Weibel R. 1988**. A review and conceptual framework of automated map generalization. *International Journal of Geographical Information Systems.* 1988, Vol. 2, no. 3, pp.229-244.

**Cohen W. 1995**. Fast Effective Rule Induction.. In *Proceedings of 12th International Conference on Machine Learning*, pp.115-123.

**Duchêne C. 2001**. Road generalisation using agents. In *Proc. of 9th Annual Conference on GIS Research in United Kingdom*, Glamorgan, 2001.

**Fritsch E. 1997**. *Représentations de la Géométrie et des Contraintes Cartographiques pour la Généralisation du Linéaire Routier*. PhD Thesis, université de Marne-la-Vallée, France.

**Harrie L. et Sarjakoski T. 2000**. Generalisation of Vector Data Sets by Simultaneous Least Squares Adjustement. Dans *International Archives of Photogrammetry and Remote Sensing*, Vol.XXXIII, Part B4, Amsterdam, pp.348-355.

**Lamy S., Ruas A., Demazeau Y., Jackson M., Mackaness W.A. and Weibel R. 1999**, The Application of Agents in Automated Map Generalisation. In *Proceedings of 19th International Cartographic Conference*, Ottawa.,vol.2, pp.1225-1234.

**Lecordix F., Plazanet C. and Lagrange J.-P. 1997**. A Platform for Research in Generalization: Application to Caricature. *GeoInformatica*, 1997, Vol. 1:2, pp.161-182.

**McMaster R.B. et Shea K.S. 1992**. *Generalization in Digital Cartography*. Association of American Geographers, Washington.

**Mustière S. 1998**. GALBE: Adaptive Generalisation. The need for an Adaptive Process for Automated Generalisation, an Example on Roads. In *Proceedings of 1st GIS'PlaNet conference*, Lisbonne.

**Mustière S., Zucker J.-D. and Saitta L. 2000**. An Abstraction-Based Machine Learning Approach to Cartographic Generalisation. In *Proceedings of 9th International Symposium on Spatial Data Handling*, Pekin, sec.1a, pp.50-63.

**Mustière S. 2001**. Apprentissage Supervisé pour la Generalisation Cartogrpahique. Ph.D. Thesis. University of Paris VI-Jussieu. June 2001.

**Plazanet C. 1996.** *Enrichissement des bases de données géographiques : analyse de la géométrie des objets linéaires pour la généralisation cartographique (application au routes).* Thèse de doctorat, université de Marne-la-Vallée.

**Regnauld N. 2001**. Constraint based mechanism to achieve automatic generalisation using agent modelling. In *Proc. of 9th Annual Conference on GIS Research in United Kingdom*, Glamorgan, 2001.

**Ruas A. et Plazanet C. 1996**. Strategies for Automated Generalization. In *Proceedings of the 7th International Symposium on Spatial Data Handling*, Delft, pp.319-336.

**Ruas A. 1998**. First results on the OEEPE test on generalisation. *OEEPE Newsletter*, 1998, vol.1.

**Ruas A. 1999**. *Modèles de généralisation de données géographiques à base de contraintes et d'autonomie*. PhD Thesis, université de Marne-la-Vallée, France.

**Weibel R., Keller S. and Reichenbacher T. 1995**. Overcoming the Knowledge Acquisition Bottleneck in Map Generalization : the Role of Interactive Systems and Computionnal Intelligence. In *Proceedings of 2nd International Conference On Spatial Information Theory (COSIT 95)*, pp.139-156.