Application à la cartographie d'un réseau

Benoît Legeard et Richard Forest

L'objectif de cette communication est de présenter un programme de "généralisation cartographique vectorielle", développé en amont d'une application de SIG existante, dédiée à la cartographie et la gestion d'un réseau. Ce programme crée automatiquement après généralisation des cartes lisibles de ce réseau, à une échelle quelconque comprise entre le 1/25 000 et le 1/1 000 000, à partir de données digitalisées et renseignées au 1/25 000. Les données traitées proviennent de l'application dédiée et y sont visualisées après le processus de généralisation, puis imprimées. Ce travail s'est appuyé dans un premier temps sur l'analyse des règles de saisie en vigueur chez le producteur des données cartographiques. Dans un deuxième temps, le développement (en C++) a nécessité la mise en place d'algorithmes complexes, avec comme contrainte majeure de ne jamais supprimer d'objets géographiques ni de données attributaires... Ce projet a été conduit et réalisé par Pacte Novation, en collaboration avec ESRI France.

Dans le domaine de la cartographie, et malgré l'évolution des logiciels de SIG, certaines tâches restent du domaine du sensoriel. La connaissance du spécialiste est toujours nécessaire pour apprécier la lisibilité d'une carte, ou pour écrire les règles permettant d'obtenir cette lisibilité. Pour des besoins spécifiques à petite échelle, la précision du positionnement des objets géographiques revêt parfois moins d'importance que le contenu de l'information qu'ils portent. La précision est sacrifiée dans une limite acceptable, au bénéfice de la lisibilité du document final. C'est dans ce but que sont utilisées les techniques de la généralisation cartographique : modifier les données géométriques et symbologiques des éléments représentés à une échelle pour en améliorer la lisibilité cartographique à une échelle inférieure, en leur conservant une précision suffisante pour les positionner dans un système de coordonnées. Si, dans son sens habituel, la généralisation nécessite la suppression de données lorsqu'on réduit l'échelle, dans notre cas précis, aucune information ne doit disparaître d'une échelle à l'autre, et toute l'information doit rester lisible.

Moyennant cette contrainte majeure, nous présenterons ici un outil de généralisation automatique (développé en C++) dédié à la cartographie d'un réseau et réalisé en complément d'une application de SIG développée sous ARC/INFO 7.0 (ESRI).

Présentation du contexte

Un SIG existant permet de représenter et de stocker les informations géographiques d'un réseau. Celui-ci manipule les données graphiques et attributaires saisies au 1/25 000e, génère des cartes papier du réseau à cette échelle et doit également éditer des cartes à des échelles plus petites.

Ce mode d'utilisation pose un problème de lisibilité des données saisies au 1/25 000e et cartographiées à des échelles inférieures. En effet, on ne peut se contenter d'effectuer une réduction d'échelle sur les données de base très détaillées pour éditer une carte générale du réseau, au risque d'obtenir un document représentant des arcs tellement proches qu'ils sembleraient se superposer, masquant des points, coupés par des labels... bref, un document illisible et inexploitable.

Afin d'obtenir des échelles inférieures au 1/25 000e, un gros travail de généralisation manuelle est effectué par les cartographes. Il consiste en la correction, à l'écran et au curseur, du tracé des arcs, des nœuds, des points et des labels, sans jamais supprimer aucun élément. Ces modifications sont apportées selon des règles strictes de sémiologie (consignées par écrit) propre à l'usage des utilisateurs des cartes, conduisant à déplacer ou écarter ces différents éléments les uns des autres.

Par exemple, le déplacement d'un nœud de 40 mm sur la carte initiale au 1/25 000e induit une erreur de positionnement de 1 km, ce qui est beaucoup pour une carte à usage "terrain". Mais ce même déplacement de 1 km ne représente plus qu'un déplacement de 1 mm sur une carte papier au 1 000 000e, ce qui, pour une carte destinée à une vision générale, est acceptable.

On obtient donc après traitement une couverture supplémentaire à l'échelle inférieure voulue, ce travail de généralisation manuelle, long et fastidieux devant être répété autant de fois qu'il y a d'échelles usuelles inférieures au

1/25 000e pour pouvoir éditer des cartes lisibles. La saisie de données nouvelles au 1/25 000e nécessite donc la mise à jour de toutes les cartes généralisées existantes...

Le projet "Généralisation"

Le projet aboutit à la création d'une application informatique, aujourd'hui opérationnelle, permettant de créer des cartes du réseau à différentes échelles (du 1/25 000e au 1 000 000e), à partir d'une Base de Données Géographique de référence (au 1/25 000e), en ayant soin :

- de ne pas supprimer d'objets géographiques ;
- de conserver les valeurs attributaires attachées aux différents objets ;
- de conserver les rapports de distances entre les objets géographiques, malgré une perte de précision du fait du processus simplificateur de généralisation ;
- de permettre une bonne lisibilité de l'information aux petites échelles.

Techniquement, l'application est réalisée sous une approche orientée objet, donc avec une modélisation forte par composants. Elle est ensuite développée en C++ et alimentée en amont par un script d'Importation/Exportation des données géométriques et attributaires d'ARC/INFO.

Les règles cartographiques de généralisation

3.1 Extraction des règles

En amont de la conception et du développement, il est nécessaire de connaître les règles utiles aux cartographes lors de la généralisation manuelle. En effet ceux-ci, dans un soucis d'homogénéité des cartes produites, ont identifié un certain nombre de règles à appliquer afin d'écarter ou rapprocher les éléments graphiques, en fonction de la symbologie utilisée à telle ou telle échelle, et du contenu de l'information attributaire de ces objets.

Les règles issues des documents techniques des cartographes sont mises en forme pour l'écriture des algorithmes. Elles sont nombreuses, mais la méthodologie reste la même pour tous les types d'objets : en fonction de la géométrie des objets, de variables attributaires des éléments géographiques et des symboles utilisés, les valeurs des paramètres de déplacement sont affectées.

Une règle simple, extraite après analyse du discours et des documents des cartographes est traduite ainsi en vue de la programmation :

```
If
(A).ARC# = (B).ARC#
AB =< E.0,0028
VARS (A) >= 150
VARS (B) >= 150
STAT(A) = " ACTIF "
STAT(B) = " INACTIF "

(Then)
P= 0,0028
Z= 1/3
```

Avec :

- (A).ARC# , (B).ARC# l'identifiant de deux arcs
- VARS, STAT des variables " métier " (fictives) de l'utilisateur du réseau extraite du discours
- AB une distance minimale à respecter entre deux objets graphiques
- P et Z deux coefficients permettant de calculer dans l'algorithme les nouvelles positions des objets à déplacer

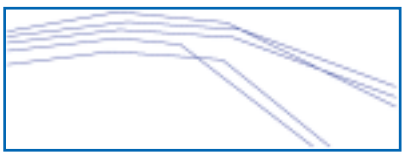
3.2 Le traitement des objets graphiques

Les traitements les plus simples sont appliqués à l'écartement des nœuds et points (voir ci dessus). La taille papier des symboles étant constante d'une échelle à l'autre, l'écartement des nœuds est nécessaire pour que ces symboles ne se superposent pas. En fonction de la taille des symboles utilisés, de la distance d'éloignement entre les points et des valeurs attributaires des nœuds, ceux-ci sont déplacés. Vingt règles de cette forme ont été identifiée pour le déplacement des nœuds les uns par rapport aux autres.

Il en existe de nombreuses autres pour d'autres types de traitements beaucoup plus complexes (nœuds/arcs, labels, zones denses...) et particulièrement en ce qui concerne l'écartement des arcs regroupés en faisceaux **(Figure 1)**. Non seulement, certaines règles doivent s'appliquer au calcul des distances entre les arcs, mais on doit également conserver leur ordre afin de ne pas générer de croisements qui ne figureraient pas dans les données initiales **(figure 2)**. Ceci implique que chaque arc connaisse son ordre dans le faisceau (de droite à gauche ou de haut en bas) en tout point de son tracé, qu'il détecte également ses voisins (afin d'en apprécier la proximité et de connaître leurs données attributaires), que le programme sache quels sont les arcs qui aboutissent au même nœud, etc.

Devant la complexité de la tâche, il faut reconstituer toute la topologie des données.

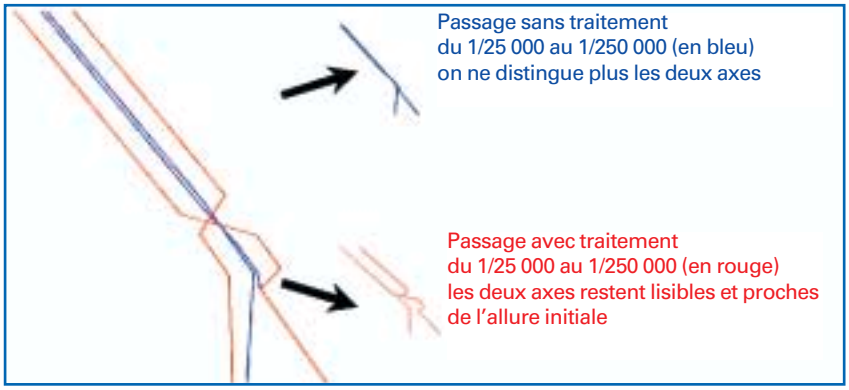
Figure 1 : exemple d'un faisceau d'arcs sur la carte initiale



3.3 Principe du traitement pour le placement des labels

L'objectif est de déplacer le label sur un emplacement voisin, en supprimant les conflits avec les autres éléments graphiques présents **(figure 3)**.

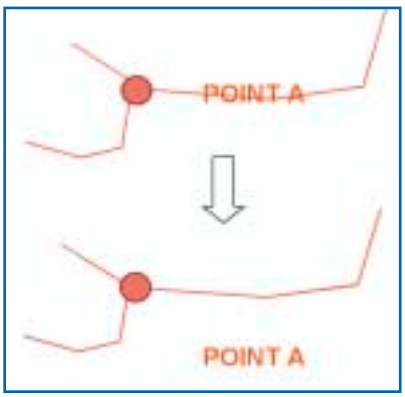
Figure 2 : exemple de traitement des arcs dans un faisceau



Les labels sont dotés d'une table des collisions pour connaître les problèmes de localisation rencontrés. On teste successivement les différentes positions possibles du (ou des) label(s) du point selon un ordre défini.

Le label ne doit jamais intercepter un arc, un nœud ou un point (objets immuables). Si la seule solution est d'intercepter un autre label, c'est le label antérieur qui doit trouver un autre emplacement en fonction de sa table de collisions. Si aucune place n'est disponible (uniquement des objets immuables), on place le label sur un emplacement par défaut, et l'opérateur intervient manuellement pour trouver un emplacement satisfaisant.

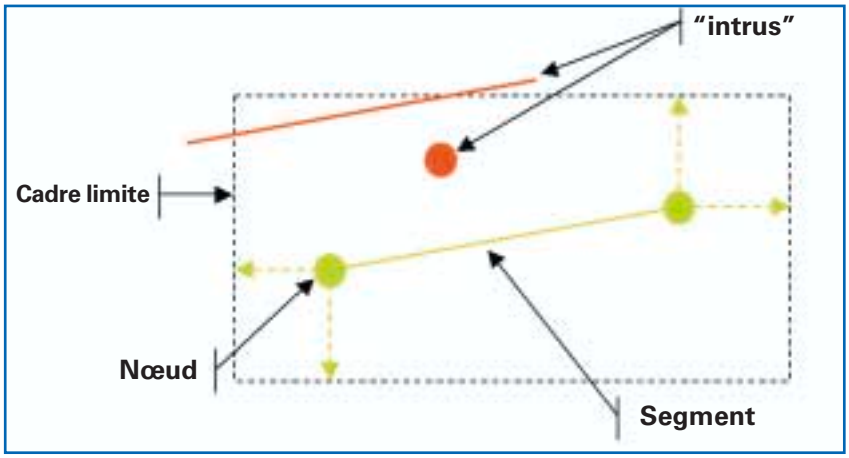
Figure 3 : déplacement automatique d'un label



Topologie

Grâce aux règles définies en 3.1 on sait désormais quels traitements appliquer aux objets pour qu'ils restent lisibles sur la carte généralisée. Il faut également identifier les objets à traiter et la manière de détecter leur proximité et les relations qu'ils entretiennent. Cette phase du projet a nécessité la mise en place d'une architecture orientée objet complexe, dans laquelle certaines classes sont dédiées spécifiquement aux traitements géométriques des objets et au renseignement de leurs environnements immédiats respectifs.

Figure 4 : principe d'un cadre limite autour d'un segment d'arc



Le principe a été de créer un carré autour des points ou un rectangle autour des segments, appelé "cadre limite" permettant de détecter les objets situés au moins en partie à l'intérieur de ce périmètre **(figure 4)** Dans le cas où un objet en détecte un autre, il s'informe des caractéristiques de ce voisin (sa nature, ses références dans son arc s'il s'agit d'un segment, ses données attributaires, ses coordonnées géographiques, etc).

Ainsi renseigné sur cet "intrus", l'objet va pouvoir s'en éloigner, en fonction de ses propres caractéristiques et des règles cartographiques désormais programmées.

Les algorithmes

Après avoir identifié les règles et reconstitué la topologie de la carte, il convient d'écrire ou de modifier les différents algorithmes nécessaires au déplacement des objets graphiques. Certains algorithmes s'appliquent à l'écartement de plusieurs nœuds trop proches, d'autres à l'écartement des arcs (entre eux, avec des nœuds...), d'autre encore à la gestion des arcs et au placement des labels, etc. Un algorithme est également spécialement dédié au lissage des arcs **(figure 5),** qui supprime les vertices devenus inutiles à certaines échelles moins précises. Il conserve l'allure générale de l'arc tout en allégeant son tracé.

Figure 5 : exemple de lissage d'un arc

Ces algorithmes sont fondés sur une série de calculs vectoriels et trigonométriques dans le plan. Ils s'appliquent successivement sur les objets dans un ordre défini, l'ensemble des traitements étant itératif pendant un nombre prédéfini de cycles, afin d'éviter que le programme ne boucle sur des situations impossibles à résoudre.

D'une manière générale, le traitement dans le programme s'organise comme suit :

1. Conversion des données ARC/INFO au format "texte"
2. Reconstitution de la géométrie de l'ensemble des données (liste des segments et des vertices composant les arcs, liste des nœuds de début, de fin des arcs...)

Après avoir identifié les règles et reconstitué la topologie de la carte, il convient d'écrire ou de modifier les différents algorithmes nécessaires au déplacement des objets graphiques. Certains algorithmes s'appliquent à l'écartement de plusieurs nœuds trop proches, d'autres à l'écartement des arcs (entre eux, avec des nœuds...), d'autre encore à la gestion des arcs et au placement des labels, etc.

3. Reconstitution de la topologie des éléments géographiques
4. Application de l'algorithme de lissage
5. Simplification des zones de convergence des arcs
6. Application de l'algorithme de traitement des relations de nœud à nœud
7. Application de l'algorithme de traitement des relations d'arc à nœud
8. Traitement des faisceaux d'arcs
9. Placement des labels
10. Conversion des données "texte" au format ARC/INFO

6 Résultats

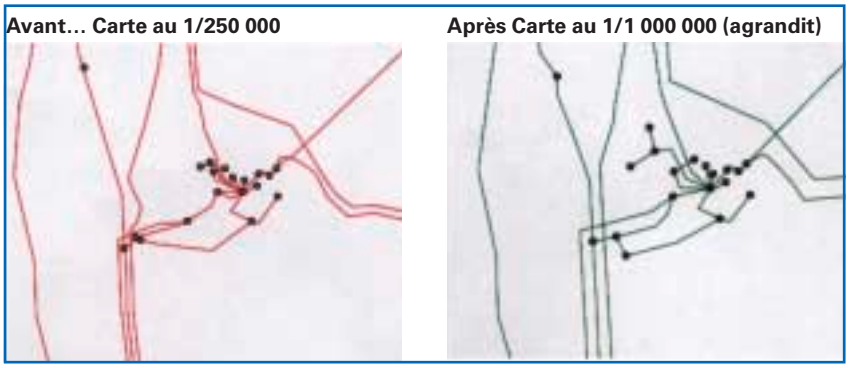
La figure 6 présente un exemple de traitement. On constate à droite que certains objets ont été déplacés de manière à rester visible à l'échelle réelle.

Le programme généralise de façon satisfaisant 80 % des objets géographiques (et approche les 90 % à certaines échelles). Les objets n'ayant pas pu être traités automatiquement de manière satisfaisante sont mis en évidence sur la carte afin de facilité leur positionnement manuel.

Conclusion

Ce travail sur la généralisation d'un réseau vectoriel est l'illustration d'une démarche opérationnelle dans une problématique qui, par ailleurs, appartient toujours au domaine de la recherche en cartographie et sciences géographiques. Cet outil, par sa conception et sa programmation orientée objet est susceptible d'évoluer en fonction des besoins des utilisateurs. Les réflexions et développements menés ici pourront ainsi être repris pour le traitement d'autres réseaux vectoriels, dans des thématiques variées. ●

Figure 6 : exemple de traitements du 1/250 000 au 1/1 000 000 (1/1 000 000 agrandi ici au 1/250 000 pour comparaison)



Orientations bibliographiques

BUTTENFIELD BP, McMASTER R.P.: Map generalization : making rules for knowledge representation -Longman scientific & technical, New-York, 1991.

GDR 1041 CNRS "MISS-CASSINI", Axe B : multi-échelle, rapport PSIG 1995 (programme national sur les systèmes d'information géographique).

HIRSH S.A. : An algorithm for automatic name around point data - The American Cartographer, 9(1) : 5-17.

MULLER J-C., LAGRANGE J-P ., WEIBEL R. : GIS & Generalization : methodological an practical issues. Taylor & Francis, Londres, 1995.

Contact :

PACTE NOVATION
2, Rue du Docteur Lombard
92441 Issy-Les-Moulineaux Cedex
Tel : 01 45 29 06 06 Fax : 01 45 29 25 00
Site : www.pactenovation.fr

Abstract

This paper deals with a program on " vectorial map generalization ", developped upstream of an existing GIS application (for Arc/Info 7.0). It is dedicated to the mapping and management of a network. This program produces automatically readable maps of this network after a generalization process, for any scale from 1/25000 to 1/000000. It uses scanned and filed data at a 1/25000 scale. Maps are edited and printed from the dedicated application. This work relies first on the analysis of rules given by the geographic data producer. Secondly, programming (in C++) asks for complex algorithms, without deleting neither geographic objects nor associated data. This project was managed and achieved by Pacte Novation, in collaboration with ESRI France.