HAL open science

Identification automatique de réseaux de voirie urbaine avec un algorithme d'optimisation par colonies de fourmis

Xavier Marsault

To cite this version:

Xavier Marsault. Identification automatique de réseaux de voirie urbaine avec un algorithme d'optimisation par colonies de fourmis. SAGEO 2010, Nov 2010, TOULOUSE, France. hal-04728475

HAL Id: hal-04728475

https://hal.science/hal-04728475v1

Submitted on 9 Oct 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Identification automatique de réseaux de voirie urbaine avec un algorithme d'optimisation par colonies de fourmis

Xavier Marsault*

** Laboratoire MAP-ARIA, FRE MAP 3315 du CNRS*
ENSA de Lyon
3, rue Maurice Audin, F-69512 Vaulx en Velin cedex
xavier.marsault@aria.archi.fr

RÉSUMÉ. *Le laboratoire MAP-ARIA développe – pour les phases de conception architecturale, urbaine et plus largement en aménagement du territoire – des dispositifs génératifs capables d'assister le concepteur ou le décideur dans l'élaboration de solutions formelles et fonctionnelles. Dans ce cadre, nous présentons ici une méthode générative innovante pour identifier des paramètres qualitatifs influant de manière générale sur une organisation de voirie, et calculer automatiquement des solutions optimisées. En partant de la seule donnée du plan du bâti urbain, nous parvenons à obtenir une hiérarchie émergente d'axes de voirie pour ces paramètres, qualitative (classification typologique: rocades, boulevards, avenues, rues et ruelles) et quantitative (caractéristiques du réseau). Nos recherches montrent qu'on peut se ramener à un problème de sélection de chemins, et un algorithme d'optimisation par colonies de fourmis permet alors de trouver de nombreuses solutions optimisées, cohérentes et pertinentes avec le choix des contraintes et des critères imposés.*

ABSTRACT. *At MAP-ARIA lab, we develop methods and tools for the generative design and assistance in the decision process in architecture and urban planning, that can assist the designer or decision maker in the development of formal and functional solutions. In this context, we present a new generative method for identifying qualitative parameters generally affecting a street network organization and automatically calculate optimized solutions. Only using urban building footprint data, we obtain emerging hierarchies of roads for these parameters, with qualitative (typology: ring roads / boulevards, avenues, streets and lanes) and quantitative (the network) aspects. Our research shows that we can reduce to a selecting paths problem, and an ant colony optimization algorithm can then find many optimized solutions, consistent and relevant to the selection criteria and constraints.*

MOTS-CLÉS : *réseaux de voirie, plus court chemin, optimisation par colonie de fourmis (ACO - SSP), classification et conception automatiques*

KEYWORDS: *street network, shortest path, ant colony optimization (ACO - SSP), automatic classification and design*

1. Positionnement

L'analyse spatiale d'images aériennes ou satellitaires, à des fins de reconnaissance automatique d'éléments structurants du paysage urbain - comme par exemple le bâti et les axes de voirie - est un sujet de recherche vaste et passionnant. Ses applications sont multiples et variées en géographie, en urbanisme, en géomatique, pour le paysage, et ses enjeux plus actuels à l'heure de l'exploitation systématique à visée « tout automatique » de banques de données massives.

Or, il est souvent plus aisé de détecter dans ces images le bâti que la voirie (moins visible, partiellement occultée, aux contours hachés, et aux teintes moins tranchées). De plus, la thermographie urbaine est un procédé moderne qui donne accès directement à l'implantation du bâti, ce qui n'est pas le cas de la voirie. Par ailleurs, la plupart des problèmes d'ingénierie et de développement urbains utilisent des systèmes d'informations géographiques dans lesquels les plans numérisés (contours ou empreintes) du bâti sont un des éléments les plus significatifs. Enfin, même si pour des raisons historiques, la structure des rues a souvent imposé le découpage en îlots et contraint fortement le positionnement du bâti, des travaux de recherche récents en modélisation de villes virtuelles (Decoret *et al.*, 2002), (Marsault, 2005) se sont intéressés au problème inverse : comment retrouver des organisations plausibles de réseaux viaires en partant de la seule connaissance de l'implantation du bâti ?

Dans ce contexte caractérisé par deux logiques imbriquées, nous proposons dans cet article de montrer comment cette dernière voie de recherche a donné de bons résultats, et permis, pour un tissu urbain réel, d'approcher le réseau viaire existant et d'ouvrir des perspectives pour traiter d'autres problèmes de planification.

2. Précédents travaux

Dans la foulée de recherches menées en modélisation automatique de scènes urbaines avec les *Iterated Function Systems* (Marsault, 2005), nous nous sommes intéressés à un problème peu traité par la communauté scientifique : la génération automatique de voirie, de places et d'éléments de décor urbain, à partir d'images 2D½ représentant uniquement les empreintes au sol et la hauteur des bâtiments (en provenance du monde réel ou de calculs). Decoret est le premier chercheur à avoir proposé une méthode robuste permettant de calculer un réseau de rues à partir de contours du bâti vectorisés (Decoret *et al.*, 2002). Mais son approche, à visée purement géométrique, ne reposait sur aucune analyse typologique ou morphologique. Le sujet n'est même pas abordé dans (Larive *et al.*, 2004) et (Chen *et al.*, 2008), qui sont pourtant des articles de référence.

Nous avons proposé dès 2004 une approche par l'image, où les niveaux de gris peuvent coder différentes grandeurs comme l'empreinte, la hauteur ou la densité d'habitation. La correspondance pixel↔mètre, fixée par l'utilisateur pour chaque image, permet de tenir compte de l'échelle réelle des tissus analysés. Grâce à l'utilisation d'opérateurs provenant de la morphologie mathématique (Graffigne, 1995) et de la théorie des graphes (Lacomme, 2001), nous avons travaillé sur

l'extraction de caractéristiques structurantes du tissu urbain (Denis, 2004).

Les étapes fondamentales de ce travail étaient : la construction automatique d'enceintes plausibles d'un tissu urbain avec ou sans raccordement vers l'extérieur (systèmes de villes), et le calcul d'un graphe de voirie potentielle *G* (Figure 1) par squelettisation homotopique de l'espace inter-bâti (le sol) - qui est à l'image ce que le diagramme de Voronoï est à l'approche vectorielle (Decoret *et al.*, 2002). Nous avons aussi montré comment repérer automatiquement des cours d'eau d'une certaine importance qui traversent une ville de part en part. Mais notre plus forte contribution a été la détermination automatique des chemins et de leur type (résumée ci-après). Nous invitons le lecteur à consulter (Marsault, 2009) pour de plus amples détails.

Figure 1. *a) tissu urbain test, b) enceinte, c) squelette morphologique homotopique.*

Entre deux bâtiments proches, il existe toujours une jonction du graphe. Cette propriété du squelette morphologique (Fig. 1c) permet de cheminer dans ce graphe totalement connecté, entre deux points quelconques. Dorénavant, on appelle chemin *c(i,j)* un groupe de jonctions consécutives entre deux nœuds *i* et *j* quelconques, de même largeur. L'ensemble des jonctions formant le chemin *c(i,j)* adoptent ainsi pour largeur la plus faible largeur des composants du chemin, Ce nivellement au minimum a un sens physique fort issu du caractère homogène d'un tracé.

Nous avons mis au point une heuristique efficace pour déterminer des chemins plausibles, basée sur l'algorithme de plus court chemin de Dijkstra (Cormen, 2004), dont on a modifié la fonction de valuation « piéton » des jonctions. En prenant en compte la largeur maximale d'un chemin, sa longueur et sa rectitude, cette nouvelle valuation favorise le passage par les endroits où il y a le plus d'espace, avec le moins de méandres possibles (Fig. 2). Cela donne, pour une jonction *j* (longeur l_j, largeur r_j, rectitude $rect_j$) la valuation V_j [1], dépendant d'un paramètre α arbitraire :

$$V_j = l_j \,.\, rect_j \,.\, (max\,\{\, r_i \,\} + 1 - r_j\,)^{\alpha} \text{ avec } 0 \leq \alpha \leq 3 \qquad [1]$$

Pour favoriser l'émergence d'une ceinture de boulevards et d'avenues dans le plan de voirie recherché, il suffit de choisir la distance qui sépare les jonctions d'enceinte du bâti suffisamment élevée pour priviégier leur utilisation dans l'heuristique [1].

On a alors recherché une méthode s'appuyant sur la robustesse de cette technique et l'exploration judicieuse du graphe *G* pour effectuer la reconnaissance automatique d'un réseau hiérarchisé de voirie. Une solution est apparue en découvrant l'intérêt de

l'algorithme d'optimisation par colonies de fourmis *Ant-SSP* (section 4). Avant d'entrer dans les détails (section 5), nous présentons la méthode mise au point.

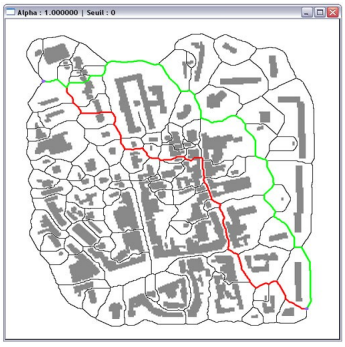

Figure 2. *Chemins dans le réseau viaire potentiel (rouge = Dijkstra ; vert = V_j).*

3. Méthode

Le problème consiste à trouver un ensemble *C* de chemins de *G* constituant un plan de voirie optimal - c'est-à-dire maximisant un certain nombre de critères tout en satisfaisant un certain nombre de contraintes - et de les hiérarchiser en les classant par type de voie. L'analyse morphologique de la voirie urbaine (Mangin *et al.*, 1999) permet de dégager 4 critères caractéristiques, que l'on va combiner au sein d'un processus d'optimisation pour trouver « en moyenne » des chemins parmi les plus longs, les plus larges, les plus variés aux intersections et les moins tortueux possibles. On souhaite aussi que leur nombre soit plutôt faible, sans pour autant à avoir à le minimiser (ce qui n'aurait pas de sens physique).

Une typologie de voirie (rocade/boulevard, avenue, rue, ruelle) basée sur ces critères permet de classer chaque élément potentiel (Figure 3). En effet, à partir de la technique de cheminement exposée en section 2, il est simple de calculer pour un chemin donné ses caractéristiques morphologiques et typologiques en fonction de sa longeur *L* [3], sa largeur *R* [4] et sa rectitude *Rect* [5]. La donnée d'une table de seuils d'admissibilité (Mangin *et al.*, 1999) permet d'obtenir à partir de ces valeurs la classification souhaitée. Pour certains éléments dont le rapport *L/R* est inférieur au seuil typologique fixé par la largeur *R*, on va ajuster *R* pour coller au type inférieur ou égal. Cette correction est essentielle pour ne pas perdre d'éléments indispensables. Enfin, bien que privilégiant les axes les moins tortueux possibles, définir un seuil de rectitude n'est pas souhaitable pour les boulevards, les rocades et les fleuves, naturellement courbes. Ainsi, on adapte les seuils par type de chemin.

Désormais, le problème n'est plus de calculer des chemins (section 2), mais d'en sélectionner un sous-ensemble parmi tous les possibles, pour former un réseau d'axes possédant une typologie, optimisant des critères, et obéissant à des contraintes (section 5). Mais quels chemins choisir, sachant qu'on ne dispose pas de bons critères locaux pour positionner leurs extrémités ? Nous voilà ramenés à un problème d'optimisation fortement combinatoire. D'emblée, soulignons que les

algorithmes génétiques (Renders, 2002) semblent peu adaptés ici, car la construction d'une solution, très contrainte, ne supporte pas de multiples divisions arbitraires. On aura donc recours à un procédé de construction incrémental, et l'algorithme *Ant-SSP* (section 4.3 et Figure 6) fournit une solution robuste et élégante à ce problème.

Type d'élément	min(*L*)	min(*R*)	max(*R*)	min(*L/R*)	Max(*Rect*)
ruelle	70 m	3,5 m	5,5 m	20	3
rue	110 m	5,5 m	20 m	20	2,5
avenue	600 m	20 m	80 m	30	1,4
boulevard/rocade	720 m	24 m	80 m	30	/
fleuve	diamètre (ville)	80 m	400 m	/	/

Figure 3. *Paramètres typologiques de la voirie issus de mesures sur des villes.*

Lors d'une étape de présélection, on calcule et l'on stocke (en mémoire et dans un fichier XML) pour chaque chemin admissible de G : sa longueur, sa largeur, sa rectitude, son type et la liste de ses noeuds N et jonctions J. Cela permet d'en éliminer d'emblée un grand nombre par l'application des seuils (Figure 3), et de diminuer considérablement le nombre de chemins admissibles pour *SSP*, afin d'accélérer le processus de recherche de solutions.

4. Optimisation par colonies de fourmis (ACO)

4.1. *Optimisation combinatoire*

De nombreux problèmes de conception ou de décision - généralement multi-acteurs et multi-critères - sont traduits sous forme de programmes mêlant des phases de génération et d'optimisation combinatoire, notamment en synthèse de formes (Kicinger, 2006). Ce couplage, destiné à réduire drastiquement l'espace des solutions explorables, doit fournir un nombre de solutions optimales suffisamment faible pour être étudiées en pratique. Or, les techniques de l'optimisation exacte s'appliquent peu dans le champ de l'ingénierie conceptuelle ou décisionnelle, et on doit recourir à des heuristiques, voire des méta-heuristiques, qui calculent des solutions optimisées (approchant l'optimum) en temps raisonnable. Comme elles sont généralement assez efficaces, on se contente d'en parfaire le fonctionnement pour le problème traité. Inspirés par des systèmes naturels, les algorithmes génétiques (biologie de l'évolution), et les algorithmes de colonies de fourmis (éthologie) sont de plus en plus performants pour traiter des problèmes dans de nombreux domaines fort éloignés de la nature, comme le rappelle (Baquiast, 2006) : « une fois que la science commence à comprendre la façon dont la nature a sélectionné certaines formes et pas d'autres, aussi bien dans le monde physique que dans le monde biologique, elle s'applique à transposer les mécanismes correspondants en vue de résoudre des problèmes intéressant la fabrication d'artefacts, outils ou objets finaux ».

4.2. *La méta-heuristique ACO (Ant Colony Optimization)*

Dans la nature, les fourmis se déplacent et se dirigent en déposant des phéromones, substances olfactives volatiles qui jouent le rôle de mémoire collective. Ce concept forme la base de la méta-heuristique ACO (Dorigo *et al.*, 2004) utilisée en optimisation combinatoire (Figure 4), simulant sur plusieurs lancés (N_{run}) la coopération stigmergique entre insectes sociaux (fourmis, termites, guêpes). La stigmergie est une méthode de communication indirecte dans un environnement émergent auto-organisé, où le travail individuel est stimulé par l'activité antérieure de l'ensemble. Elle s'applique parfaitement à la recherche de cheminement ou d'ordonnancement dans un graphe, en utilisant des traces de phéromones virtuelles pour marquer les arêtes qui favorisent les meilleures solutions.

Choisir un nombre de runs *Nrun* et *n* fourmis. Pour chaque run :
- **initialiser** les traces de phéromones

- Jusqu'à atteindre un **critère d'arrêt** (**nombre de cycles** N_c) :
  - **construction** de solutions par les *n* fourmis,
  - **amélioration** des solutions par recherche locale (optionnelle),
  - **sauvegarde** des meilleures solutions,
  - **mise à jour** des traces de phéromones pour ces solutions.

Analyser les meilleures solutions trouvées

Figure 4. *Algorithme ACO générique.*

4.3. *L'algorithme Ant-SSP*

D'autres problèmes se ramènent plutôt à des processus de sélection, et non d'ordonnancement de composants. Le but est alors de trouver un sous ensemble d'objets dans un lot prédéfini, satisfaisant certaines contraintes et optimisant une ou plusieurs fonctions objectifs (ou critères). On appelle ces problèmes *SSP* (Subset Selection Problems), et le plus connu est le remplissage optimal du sac à dos (Cormen, 2004). Une adaptation d'*ACO* générique, *Ant-SSP*, décrit dans (Solnon, 2008), semble bien convenir à notre cas puisqu'il permet d'assembler incrémentalement des chemins dont l'ordre d'introduction dans la solution n'a a priori pas d'importance. Cette caractéristique permet aussi d'adapter le choix des chemins de manière à satisfaire les contraintes de consistance (section 5).

4.4. *Résolution de problèmes multi-critères avec ACO*

En présence de *m* objectifs partiels f_i qui ne sont pas complétement orthogonaux (indépendants), on peut définir un mono-objectif mixte par combinaison algébrique des f_i, souvent par pondération linéaire [2] avec des coefficients α_i positifs ou négatifs (car il y a des crières à maximiser et d'autres à minimiser).

$$objectif_mixte = \Sigma\,(\alpha_i \,.\, f_i)\ \ (combinaison\ linéaire) \qquad [2]$$

Mais la plupart du temps, on se sait pas quantifier les poids α_i de chaque objectif, car ils sont de nature différente. On peut l'éviter en considérant une vraie

optimisation multi-objectifs. La difficulté est qu'il n'y a plus une « meilleure » solution correspondant à un seul objectif chiffré, mais un ensemble de solutions avec *m* objectifs chiffrés, qu'il s'agit de comparer (Agrawal *et al.*, 2000). Dans ce cas, on va calculer le « front de Pareto », qui est l'ensemble des solutions non dominées, c'est-à-dire au moins aussi bonnes que toutes les autres sur l'ensemble des objectifs, et meilleures sur un objectif au moins. Ces solutions, par définition, ne sont pas comparables entre elles : c'est à l'utilisateur, au final, de choisir, en faisant intervenir des critères non quantifiables. La mise en œuvre avec *ACO* ou *SSP* est décrite dans (Alaya *et al.*, 2007) et (Angus *et al.*, 2009). Elle associe en général sa propre trace de phéromones (ainsi qu'une fonction heuristique) à chaque objectif, pour tenter de l'optimiser indépendamment des autres. Pour cela, à chaque étape de la construction d'une solution, on ajoute un élément qui favorise aléatoirement l'un des *m* objectifs.

5. Détails de mise en oeuvre d'*Ant-SSP*

5.1. *Critères à optimiser*

L'objectif de maximiser la longueur globale *L* de tous les chemins *c* en même temps que leur largeur *R* et celui de minimiser leur rectitude sont déjà atteints au départ avec l'ensemble des jonctions-chemins de *G*. C'est donc en moyenne que l'on va calculer ces objectifs, pour faire émerger des solutions intéressantes, en pondérant chaque terme par la longueur du chemin, afin de tenir compte de l'étendue spatiale. *C* étant l'ensemble des chemins d'une solution et *l(k)* la longueur d'une jonction *k*, on définit quatre critères simples :

- la longueur moyenne des chemins :

$$L(C)=\frac{\sum_{c\in C} l(c)}{[C]} \quad \text{avec} \quad l(c)=\sum_{k\in C} l(k) \qquad [3]$$

- leur largeur moyenne :

$$R(C)=\frac{\sum_{c\in C} r(c).l(c)}{\sum_{c\in C} l(c)} \quad \text{avec} \quad r(c)=\min_{k\in c} r(k) \qquad [4]$$

- leur rectitude moyenne :

$$Rect(C)=\frac{\sum_{c\in C} rect(c).l(c)}{\sum_{c\in C} l(c)} \quad \text{avec} \quad rect(c(i,j))=\frac{l(c)}{dist(i,j)} \qquad [5]$$

- l'écart moyen *E(C)* des largeurs aux intersections : ce paramètre à maximiser est calculé à partir de chaque noeud d'intersection de chemins et de chaque noeud terminal libre de la solution (cf. §5.4) :

$$E(C)=\sum_{(i,j)\in Sk\times Sk / ci\cap cj} |r(ci)-r(cj)| \qquad [6]$$

La nature même de l'algorithme *Ant-SSP* évite d'introduire un critère de densité de voirie, car les solutions qui émergent optimisent naturellement le nombre de chemins pour un critère donné. En effet, tant que le réservoir de chemins admissibles n'est pas vide, on en choisit un élément via un critère probabiliste (Fig 6). Ce sont les

filtres initiaux (Fig 3) qui aident à contôler la densité du réseau. On pourrait aussi introduire des objectifs optionnels visant à satisfaire par exemple un ratio de répartition des rues, une loi d'échelle. Nous avons laissé cela pour l'instant de côté.

On a conçu et testé 36 critères variés (Figure 5) à partir de ces 4 critères partiels, et en cohérence avec l'heuristique [1] pour certains : produits globaux, sommes de critères simples (avec normalisation, car on additionne des valeurs de nature différente) ou somme de produits de critères sur les éléments de la solution. A noter la nature purement géométrique et morphologique de ces choix, justifiée dans une première approche où seule l'empreinte au sol du bâti est vue comme signifiante.

5.2. *Contraintes à satisfaire*

La morphologie urbaine habituelle nous a déjà fait introduire trois contraintes d'admission, sous forme de filtres (Figure 3) sur la longueur, la largeur et le rapport *L/R* qui doit être supérieur à un seuil (plus une voie est large, plus elle est longue). La construction incrémentale d'une solution par *SSP* ajoute trois contraintes supplémentaires, dites « de consistance » :

- aucune jonction de la solution n'est commune à plusieurs chemins (principe de non superposition, même partielle).

- toute solution forme un sous-graphe connexe de *G (*possibilité de se déplacer n'importe où dans la ville). Pour ce faire, il suffit que tout chemin ajouté à chaque étape ait au moins un nœud d'extremité (connexité faible) ou un noeud quelconque (connexité forte) en commun avec la solution en cours S_k. On obtient de meilleures solutions avec la seconde option, mais avec des temps de calculs réhaussés de 10%.

- minimisation du nombre d'impasses T_c (chemins libres à une seule extrémité, en partie générés par le processus de construction), tout en s'efforçant de garder les impasses naturelles du tissu urbain. Mais la satisfaction de cette contrainte à chaque étape est impossible car tout chemin ajouté à la solution ne peut pas toujours s'appuyer sur 2 noeuds déjà utilisés. Nous proposons de ne plus la considérer comme une contrainte, mais comme un critère d'optimisation propre à la construction

C	-C	L	-L	E	-E
R	-R	-Rect	Σl	$-\Sigma l$	$\Sigma r^{\alpha} l$
$-\Sigma r^{\alpha} l$	$\Sigma(r^{\alpha} l/rect)$	$-\Sigma(r^{\alpha} l/rect)$	$\Sigma(r^{\alpha} l)/E$	$\Sigma(r^{\alpha} l).E$	$\Sigma(r^{\alpha} l)/Rect$
L+R-Rect	$LR^{\alpha}/Rect$	R/Rect	$\Sigma(r^{\alpha} l/rect)/E$	$\Sigma(r^{\alpha} l/rect).E$	$L.R^{\alpha}.Rect$
(R/Rect).E	$L.R^{\alpha}.E/$ Rect	R.E	-Rect.E	L.E	L+R-Rect+E
$L.R^{\alpha}$	$L.R^{\alpha}.E$	L+R	L+R+E	$\Sigma(r^{\alpha} l).E/Rect$	$L.R^{\alpha}/Rect+E$

Figure 5. *36 critères testés pour l'optimisation avec Ant-SSP.*

5.3. *Heuristiques locales*

Le but des facteurs heuristiques *h* est d'évaluer l'intérêt de retenir des chemins,

connaissant la solution partielle S_k déjà construite. Ils concernent à la fois l'amélioration de la construction de la solution (h_C) et sa qualité (h_Q). Ainsi, l'heuristique finale $h = h_Q . h_C$ est utilisée conjointement à une trace phéromonale pour définir la probabilité de sélectionner un chemin [8]. De plus, si les heuristiques statiques sont préstockées avec chaque chemin pour plus d'efficacité, l'évaluation des heuristiques dynamiques doit être faite le plus rapidement possible, puisqu'elle intervient des millions de fois durant la recherche. La définition de ces facteurs n'est pas toujours aisée, et demande d'effectuer de nombreux tests. Par exemple : $h_Q(L) = L$; $h_Q(R) = R$; $h_Q(E) = L / R$; $h_Q(LR, Rect) = LR / Rect,$ sont des heuristiques de qualité statiques et performantes.

Une heuristique de construction h_C est utilisée lors de la mise en oeuvre de *Ant-SSP* pour résoudre progressivement l'objectif du minimum d'impasses T_c. Comme il faut placer 2 noeuds i et j et que l'un d'eux appartient à S_k (connexité), il y a 6 possibilités d'ajout d'un nouveau chemin à chaque étape, selon qu'on distingue les noeuds de terminaison libre (T, valence 1), interne à un chemin ou terminaison non libre (I, valence 2) ou externe à S_k (E, valence 0) : (TT), (TI), (TE), (II), (IE) et (EE). Le nombre total de chemins admissibles $C(k)$ se décompose, respectivement à ces 6 choix, en : $C_{TT}(k)$, $C_{TI}(k)$, $C_{TE}(k)$, $C_{II}(k)$, $C_{IE}(k)$ et $C_{EE}(k)$. On désigne par $\delta T_c(k)$ la variation de T_c à la fin de l'étape k. L'algorithme *Ant-SSP* s'exécute tant que $C(k)$ est non nul, et l'arbitrage dynamique de ces 6 possibilités a pour but de faire décroitre T_c jusqu'à atteindre le minimum. Chaque choix influe de manière différente sur $T_c(k)$, mais seuls quatre d'entre eux réclament une régulation, car avec (EE) $\delta T_c = +2$, avec (IE) $\delta T_c = +1$, avec (TI) $\delta T_c = -1$ et avec (TT) $\delta T_c = -2$. La probabilité d'utiliser (TT) doit être plus faible que celle d'utiliser (TI), car ce choix n'explore pas de nouveaux noeuds. L'usage de (IE) ou (EE) doit bien sûr être limité, car il s'oppose à la décroissance de T_c. Nos expérimentations ont conduit à construire pour chaque cas les fonctions heuristiques h_c [7] avec d'excellents résultats.

$$\begin{array}{llll} h_C(TT) = C(k) / C_{TT}(k) & \text{et} & h_C(TI) = C(k) / C_{TE}(k) & \\ h_C(IE) = C_{IE}(k) / (C(k).T_c(k)) & \text{et} & h_C(EE) = C_{EE}(k) / (C(k).T_c(k)) & [7] \\ h_C(autre) = 1 & & & \end{array}$$

5.4. *Mise en oeuvre d'Ant-SSP – Construction des solutions*

Deux modélisations phéromonales (vertex, clique) sont généralement proposées (Solnon, 2008) : soit associer à chaque composant i une trace t_i qui représente l'expérience passée de la colonie concernant l'intérêt de sélectionner l'objet i, soit associer à chaque paire de composants (i,j) une trace t_{ij} représentant l'intérêt de sélectionner i et j dans le même ensemble. Nous n'avons utilisé que la structure de type vertex, car l'ordre d'inclusion des chemins dans la solution n'a a priori aucune importance. De plus, la structure de type clique, plus consommatrice en temps de calcul, n'est pas plus efficace si les temps de calcul doivent être limités.

L'évaluation de chaque objectif lors de la construction est incrémental et très rapide : par exemple, quand on choisit un chemin candidat c, on vérifie rapidement qu'aucune jonction de c n'est déjà utilisée dans S_k, en maintenant une liste de marqueurs. Il en est de même pour la vérification des contraintes.

Le principe de sélection élitiste de *SSP* exige de faire concourir de nombreuses fourmis avant de mettre à jour les phéromones, seulement pour les meilleures fourmis du cycle, avec un dépôt inversement proportionnel à l'écart entre la combinaison construite et la meilleure combinaison obtenue depuis le départ [9]. Par rapport à une utilisation classique d'*ACO*, nous avons aussi constaté qu'il est préférable de récompenser à chaque cycle les meilleures solutions, avec une faible tolérance (ε=0,01) par rapport à l'optimal du cycle. La vitesse de convergence est légèrement plus lente, mais les objectifs sont nettement supérieurs, et comme en plus ils sont atteints plus tôt, on retiendra cette astucieuse méthode.

Initialiser les facteurs phéromonaux à tmax
$S_{best} = \varnothing$, $F_{best} = -\infty$

Tant que la condition d'arrêt n'est pas atteinte (nombre de cycles ou objectif seuil)
{
  // lancer un cycle d'optimisation (ici: maximisation)
  $S_{best_cycle} = \varnothing$, $F_{best_cycle} = -\infty$
  pour chaque fourmi $f_k \in (f_1 ... f_n)$, construire une solution admissible S_k:
  {
    $S_k = \varnothing$
    candidats = { $O_j \in S$ / O_j consistant dans S }

    tant que (candidats $\neq \varnothing$)
    {
      mise à jour des heuristiques dynamiques h pour tous les candidats
      sélectionner $O_i \in$ candidats avec la probabilité:

$$p(Oi) = \frac{h(Oi)^a . t(Oi)^b}{\sum_i h(Oi)^a . t(Oi)^b} \quad [8]$$

      ajouter O_i à S_k
      enlever O_i de candidats
      enlever de candidats tout O_j non consitant avec S_k
    }

    évaluer la fonction objectif $F(S_k)$
    si $F(S_k) \geq F_{best}$ faire $F_{best} = F(S_k)$ et $S_{best} = S_k$
  }

  // mettre à jour les phéromones
  appliquer la stratégie d'évaporation à tous les facteurs phéromonaux t_i:
    $t_i = t_i * (1 - \rho)$
  récompenser les N meilleures fourmis (tolérance ε par rapport à F_{best_cycle}):
    $t_i = t_i + 1 / (1 + F_{best} - F_{best_cycle})$ [9]
  appliquer la stratégie Min-Max sur les facteurs phéromonaux t_i :
    $clamp (t_i , t_{min} , t_{max})$

  si $F_{best_cycle} \geq F_{best}$ faire : $F_{best} = F_{best_cycle}$ et $S_{best} = S_{best_cycle}$
}
Retenir la solution S_{best} et son objectif F_{best}

Figure 6. *Un « run » de l'algorithme Ant-SSP mono-objectif*

Pour de nombreux problèmes de graphe, on cale souvent le nombre n de fourmis par cycle sur le nombre N_d de noeuds du graphe, lequel est souvent modeste. Dans notre cas, il peut y avoir des milliers de nœuds, donc des millions de chemins admissibles (N_{paths}). Nous avons constaté expérimentalement que le nombre de fourmis peut être fixé à $k\sqrt{N_{paths}}$ (largement plus faible que N_d, grâce au filtrage initial), sans pénaliser la qualité des solutions, tout en rendant le calcul plus rapide. Cependant, l'ajustement de k débouche en pratique sur un nombre de fourmis n peu dépendant de N_{paths} (entre 70 et 150 fourmis par cycle). On obtient ainsi un bon compromis entre le nombre de cycles N_c et le nombre de fourmis par cycle n, dont le produit est proportionnel au temps de calcul, ce qui permet de lancer plusieurs « runs » de *SSP* et d'obtenir plus de combinaisons optimales par laps de temps. Ainsi, les réglages optimaux des paramètres de l'algorithme *Ant-SSP* affectés à notre problème de reconnaissance de voirie ont été les suivants :

$t_{min} = 0.01$, $t_{max} = 5$ ($\in [4 ; 10]$), $\rho = 0.01$ (d'après [Solnon, 2008])
$a = 4$, $b = 3$ (ajustement optimal), $n = 100 \pm 10\%$, $N_c \in [1000 ; 30000]$, $N_{run} = 12$

5.5. *Algorithme Ant-SSP incrémental mono-objectif*

On note S l'ensemble des chemins admissibles de G, ($f_1 \ldots f_n$) n fourmis qui construisent respectivement les solutions ($S_1 \ldots S_n$) par cycle d'optimisation, S_{best} la meilleure solution et F_{best} sa note d'objectif. La figure 6 décrit le déroulement de *SSP* mono-objectif. A noter qu'on ne peut améliorer les solutions par recherche locale (Figure 4), qui consiste, à partir d'une solution consistante, à appliquer des modificateurs de voisinage (ex: déplacements de nœuds) repectant les contraintes. Or, les chemins étant pré-définis, il est impossible d'en modifier un élément. La prise en compte de l'objectif (T) se traduit simplement à la fin en soustrayant $T_c(k)/20$ au critère mixte évalué (figure 5), et cela fonctionne très bien.

6. Tests et résultats

Pour illustrer les précédents exposés théoriques, et analyser quelles solutions émergent naturellement dans des temps de calculs raisonnables, pour les critères/contraintes retenus, on a choisi un exemple simple avec 208 nœuds et 7190 chemins admissibles (issus du filtrage de 89253 éléments potentiels), et un tissu plus dense (portion de la ville de Décines, avec 2580 noeuds et 194584 chemins). On a testé les 36 paramètres morphologiques de la Figure 5, et le calcul du « front de Pareto » des critères 30 et 36 éclatés. La Figure 7 montre les meilleurs plans de voirie obtenus après 10000 cycles de calcul (il n'est pas possible de montrer ici tous les résultats, y compris des comparaisons avec le tissu réel). Les 8 critères suivants, la plupart mixtes, donnent de bons réseaux, bien structurés, bien connectés aux endroits où dans la réalité il existe des connections jugées pertinentes, et évitant des groupements de chemins aléatoires. Il s'agit des critères : R (7), $\Sigma r^{\alpha} l$ (12), $\Sigma(r^{\alpha} l) / Rect$ (18), $L.R^{\alpha} / Rect$ (20), $L.R^{\alpha}.E / Rect$ (26), $L+R-Rect+E$ (30), $\Sigma(r^{\alpha} l).E/Rect$ (35) et $LR^{\alpha} / Rect+E$ (36), visiblement en concordance avec l'heuristique [1], mais avec des nuances. Par exemple, les solutions qui émergent optimisent naturellement le nombre C de chemins pour un critère donné, mais (20,26)

diminuent vraiment *C*, tandis que (12,18,35,36) l'augmentent mais préservent mieux les alignements ou des parallélismes naturels des axes des lotissements. (7,36) apparaissent ici comme un compromis entre ces deux tendances, mais dont la qualité est plus discutable. Nos expérimentations montrent aussi que l'objectif de minimisation de la rectitude *Rect* est essentiel dans 6 des 8 meilleurs critères, même si l' heuristique [1] la favorise déjà. De plus, la minimisation de *E* privilégie systématiquement les solutions avec des chemins peu larges et peu longs, au détriment des autres critères. Par contre, sa maximisation favorise le raccordement de chemins de largeurs très différentes, et agit comme un régulateur de longueur (26,30,35,36). On a donc 8 critères typo-morphologiques mixtes permettant de retrouver et de classifier des axes de voirie existants, avec une petite préférence pour (12,18,35,36), même s'ils diminuent la longueur moyenne des chemins en augmentant leur nombre, car les réseaux qu'ils donnent sont plus réguliers et plus proches de ceux observés dans la réalité.

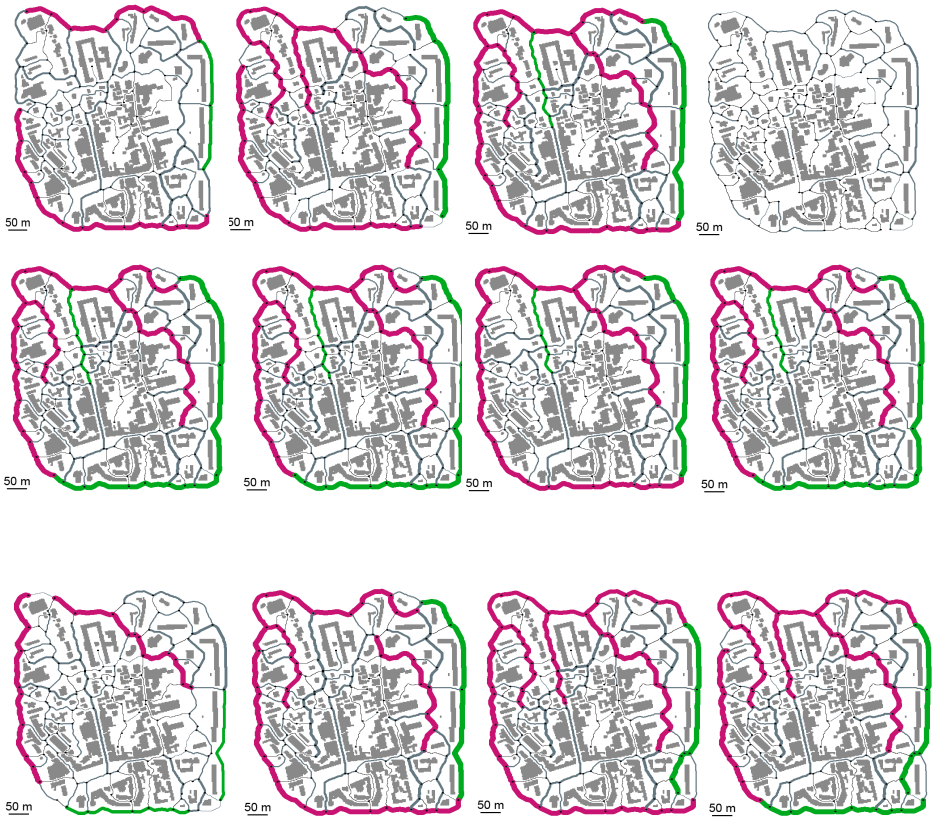

Figure 7. Optimisation des critères (2,5,7,9,12,18,20,28,26,30,35,36) sur un tissu test, portion de la ville de Saint-Genis. Meilleures solutions sur 10000 cycles de calcul – Légende: gris clair (bâti existant), rouge (boulevards ou rocades), vert (avenues), gris foncé (rues/ruelles) – rayons proportionnels aux largeurs des axes.

Figure 8. Optimisation des critères (18, 35, Pareto(30), Pareto(36)) sur une portion de la ville de Décines. Meilleures solutions sur 1000 cycles de calcul. Légende: gris clair (bâti existant), rouge (boulevards ou rocades), vert (avenues), gris foncé (rues/ruelles) – rayons proportionnels aux largeurs des axes.

7. Conclusions et perspectives

Nous avons mis au point et testé une méthode générative originale pour tenter d'identifier expérimentalement des paramètres qualitatifs qui influent sur un problème difficile de reconnaisance de voirie urbaine. Elle fournit de bons résultats, encourageants, mais pas nécessairement une optimisation globale (la consolidation du choix des critères est loin d'être achevée). Qualitativement, on observe nettement sur un large tissu un découpage en îlots plausibles (Figure 8), ce qui traduit l'émergence d'un niveau morphologique supérieur à celui des seuls axes.

Le paramétrage du calcul des solutions optimisées s'est révélé assez long à mettre en oeuvre, et nous ne saurions conclure encore si l'optimisation de Pareto donne de meilleurs résultats que les bonnes pondérations de critères retenus (section 6). Curieusement, la contrainte de connexité peut émerger de la seule optimisation de certains critères mixtes (en cours d'analyse), au bout de plusieurs centaines de milliers de cycles. Son relâchement permet plus de liberté pour l'intersection des chemins, mais il est préférable de la maintenir pour accélérer la convergence.

L'optimisation poussée des codes C++ et un réglage fin des paramètres de l'algorithme *SSP* ont permis d'éviter des calculs superflus et contibué à diminuer massivement les temps de recherche. Notamment un bon compromis entre les nombres de cycles et de fourmis par cycle a été trouvé. Néanmoins, la complexité actuelle de notre algorithme en fonction du nombre de chemins c et du nombre de noeuds n demeure en $O(c.n^{½})$, impliquant des temps de calculs non négligeables pour des tissus urbains dont les nœuds du graphe se comptent en milliers. Sur un Intel QX6800, 3.16 Ghz, 10000 cycles de calcul prennent 18 minutes de calcul pour les résultats de la figure 7, alors que pour ceux de la figure 8, 15 heures sont requises pour 1000 cycles. Nous avons mis au point une version de l'algorithme dont la complexité est en $O(c)$, donc très rapide. Hélas, l'énorme quantité de mémoire nécessaire à son fonctionnement est de l'ordre de $c(c+1)/16$ octets, handicap qui la rend pour l'instant impossible à utiliser pour des tissus urbains de moyenne taille.

Il nous reste à analyser comment le paramètre α de l'heuristique [1] agit sur les solutions produites ($\alpha = 0.8$ semble meilleure pour l'échantillon présenté), et à chercher s'il existe une corrélation avec une mesure de la fractalité du réseau de rues. On améliorera aussi le calcul et le lissage des axes médians du squelette afin de prendre en compte des alignements plus réalistes de bâtiments et moduler la distance de l'axe à ces bâtiments avant de lancer les calculs d'optimisation. On effectuera enfin des tests sur des tissus urbains plus larges, ce qui nécessite d'accéder à plus de mémoire et certainement de paralléliser les recherches de solutions par les fourmis.

Notre démarche peut être transposée à d'autres problèmes basés sur l'exploitation d'un graphe spatial. En adaptant l'heuristique [1], le choix des critères et en utilisant les niveaux de gris de la carte des empreintes pour coder des données urbaines, sociales (ex: densité de population) ou environnementales (ex: nature du sol), transférables au graphe *G,* on peut envisager : d'améliorer un réseau viaire existant, de résoudre un problème économique (minimum de chemins déservant le maximum de lieux, asservis à la densité de population et à une fourniture énergétique...), ou encore de faire de la simulation de trafic sur un réseau.

Remerciements

A Christine Solnon, pour son accueil, sa gentillesse, ses conseils et sa patience. A Jean-Baptiste Denis pour les premiers codes C++ très structurés écrits en 2004, qui m'ont bien aidé à implémenter les algorithmes d'optimisation et de visualisation.

8. Bibliographie

Agrawal S., Pratap A., Meyarivan T., A fast elitist non-dominated sorting genetic algorithm for multi-objective optimisation: NSGA-II, K.Deb, *Kangal report n° 200001*, 2000.

Alaya I., Solnon C., Ghedira K., Ant Colony Optimization for Multi-objective Optimization Problems, *19th IEEE Int.Conf. on Tools with Art. Intel. (ICTAI)*, Patras, Greece, oct 2007.

Angus D., Woodward C., Multiple objective ant colony optimisation, *Swarm Intelligence,* vol3, pp69-85, Springer, 2009.

Baquiast J.P., La morphogénèse, http://www.automatesintelligents.com/echanges/2004/jan/morphogenese.html, 2006.

Chen G, Esch G, Wonka P, Muller P, Zhang E, Interactive Procedural Street Modeling, *ACM Transactions on Graphics (TOG)*, Vol 27, Issue 3, Siggraph 2008.

Cormen T, Leiserson E, Rivest R, Stein C, Introduction à l'algorithmique, Ed. Dunod, 2004.

Decoret X, Sillion F, Street Generation for City Modelling, Architectural and Urban Ambient Environment, INRIA, 2002.

Denis J.B., Calcul automatique de voiries et de places d'une ville représentée par l'image des empreintes de ses bâtiments, Magistère Informatique de Paris, MAP-ARIA, avril-juillet 2004.

Dorigo M, Stützle T, Ant Colony Optimization, MIT Press, 2004.

Graffigne C., Zerubia J., Masson, Analyse d'images : filtrage et segmentation, 1995.

Kicinger R., Arciszewski T., Empirical analysis of memetic algorithms for conceptual design of steel structural systems in tall buildings, George Mason University, USA, 2006.

Lacomme P, Prins C, Sevaux M, Algorithmes de graphes, Editions Eyrolles, 2001.

Larive M, Gaildrat V, Génération Automatique de Zones Urbaines: Etat de l'Art, AFIG, 2004.

Mangin D, Panerai P, Projet urbain, Editions parenthèses, 1999.

Marsault X., Can Ants Build Urban Street Networks ?, *International Conference on Computers & Industrial Engineering (CIE'39)*, 6-8 juillet 2009, Troyes, France.

Marsault X., Generation of textures and geometric pseudo-urban models with the aid of IFS, "Chaos in Art and Architecture", *Int.Journal of Dynamical System Research*, vol I, n°3, 2005.

Renders J., Algorithmes génétiques et réseaux de neurones, Hermès, 2002.

Solnon C., Optimisation par colonies de fourmis, Hermès, 2008.