Reconnaissance automatique de réseaux viaires urbains plausibles via un algorithme d'optimisation par colonies de fourmis

Xavier Marsault

Reconnaissance automatique de réseaux viaires urbains plausibles *via* 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. *Nous présentons un modèle empirique développé au laboratoire MAP-ARIA, initialement pour des recherches menées en modélisation automatique de scènes urbaines, et amélioré par la suite pour la reconnaissance automatique de la voirie en milieu urbain (thème peu traité par la communauté scientifique). En partant d'une description simplifiée du bâti, on vise à identifier des critères typo-morphologiques influant sur une organisation de voirie urbaine. La mise en action du modèle convoque une méthode générative innovante pour en identifier les meilleurs paramètres et calculer automatiquement un éventail de solutions plausibles par l'application d'un algorithme d'optimisation par colonies de fourmis. Les avantages et les limites de cette approche sont ensuite analysés, et l'on suggère comment ce type de modèle peut être étendu à d'autres applications urbaines.*

ABSTRACT. *We focus on an empirical model initially developed at MAP-ARIA for research in automatic modeling of urban scenes, and subsequently improved for automatic recognition of roads in urban areas (subject little discussed by the scientific community). Starting from simplified descriptions of the buildings, it aims to identify morphological criteria affecting an organization of urban roads. The actuation of the model uses an innovative generative method to identify its best parameters and automatically compute a range of plausible solutions by applying an ant colony optimization algorithm. Advantages and limitations of this approach are then discussed, and we suggest how this type of model can be extended to deal with other urban applications.*

MOTS-CLÉS : *réseaux de voirie, plus court chemin, optimisation par colonies de fourmis (ACO - SSP), classification automatique, conception.*

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

1. Introduction

La reconnaissance automatique d'éléments structurants du paysage urbain – comme par exemple le bâti ou les axes de voirie – est un sujet de recherche vaste et passionnant. Elle est d'abord l'objet d'intérêt de l'analyse d'images aériennes ou satellitaires, pour des applications en cartographie notamment, *via* le traitement puis l'exploitation systématique 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 une technologie moderne qui donne accès directement à l'implantation du bâti, ce qui n'est pas le cas de la voirie.

Historiquement, la structure viaire, souvent pré-existante, a contraint fortement le découpage parcellaire, le positionnement du bâti et l'îlotage. De fait, il existe de fortes corrélations entre le réseau de voirie et l'implantation du bâti. Cet article fait la synthèse et dresse le bilan d'une récente étude de notre laboratoire portant sur le couple voirie/bâti en milieu urbain, dont un objectif était d'extraire des connaissances morphologiques sur les tissus urbains, et ce faisant, de mieux comprendre certaines dynamiques urbaines.

A la suite de travaux de recherche en modélisation de scènes urbaines (Decoret *et al.*, 2002), notre hypothèse était qu'on devait pouvoir calculer automatiquement des plans de voiries plausibles à partir de la donnée d'implantation au sol du bâti (en préférence à la donnée des parcelles, information de plus haut niveau dans la chaîne de traitement), en construisant un modèle empirique descriptif de l'organisation implicite de la voirie. Notre questionnement a d'abord porté sur la nature et l'identification de paramètres typo-morphologiques susceptibles de qualifier des structures viaires urbaines, et permettre de reconnaître le mieux possible les avenues, les boulevards, les rues. On s'est ensuite intéressé à la manière de combiner judicieusement ces éléments afin d'obtenir des plans optimisés d'organisation de voirie.

Nous montrons dans cet article comment cette voie de recherche a donné des résultats satisfaisants, et permis, pour un tissu urbain réel, d'approcher le réseau viaire existant et d'ouvrir des perspectives pour traiter des problèmes de planification ou d'ingénierie urbaine, où les plans numérisés du bâti et des parcelles sont des éléments importants des études.

2. Travaux antérieurs

Dans la foulée de recherches menées en modélisation automatique de villes virtuelles 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 des empreintes au sol et de la hauteur des bâtiments (provenant 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 des contours vectorisés des parcelles ou du bâti (Decoret *et al.*, 2002). Mais son approche, purement géométrique, ne reposait sur aucune analyse typologique ou morphologique des éléments de voirie. Ce thème n'est pas abordé dans (Larive *et al.*, 2004) et (Chen *et al.*, 2008), qui sont pourtant des articles de référence.

2.1. Graphe de voirie potentielle

A l'heure actuelle, la classification automatique ou semi-automatique des tissus urbains est rarement effectuée à partir de données vectorielles, mais essentiellement à partir d'images satellites et/ou de photographies aériennes. Ainsi, l'extraction de caractéristiques quantitatives et qualitatives au niveau des îlots et des parcelles est bien souvent une étape préalable à l'analyse et au suivi de l'évolution des tissus urbains dans le temps. Elle passe la plupart du temps par une classification et une analyse morphologique qui peuvent être assimilées à un problème de fouille de données (Puissant *et al.*, 2010).

Dès lors, dans l'optique d'une reconnaissance automatique de réseau viaire, on aurait pu d'abord rechercher les îlots, car ils délimitent généralement assez bien la voirie principale, et dans ce cas, son extraction et sa classification sont très nettement simplifiées (cf. section 6). Mais, cela suppose des connaissances particulières et n'est pas exempt de difficultés (Puissant *et al.*, 2010). De plus, on n'est pas dispensé d'une étape de classification et de regroupement d'éléments structurants, ce que la technique que nous avons mise au point ces dernières années résout très bien.

Il s'agit « d'une approche par l'image », cohérente, évitant la phase de vectorisation, et où les niveaux de gris peuvent coder différentes grandeurs comme l'empreinte, la hauteur ou la densité d'habitation. Notre approche exploite la topologie et la morphologie de l'espace inter-bâti, qui sont des données de bas niveau supposées suffisantes (c'est notre hypothèse), sans connaissance de la nature du sol ni de celle du bâti (objet d'études ultérieures). La correspondance pixel↔mètre est fixée par l'utilisateur pour chaque image, afin de tenir compte de l'échelle réelle des tissus analysés. Grâce à l'utilisation d'opérateurs issus de la morphologie mathématique (Graffigne, 1995) et de la théorie des graphes (Lacomme, 2001), nous avons travaillé sur l'extraction de données caractéristiques ou structurantes du tissu urbain (Denis, 2004) :

– la « carte des rayons » : qui évalue, grâce à un opérateur morphologique, la largeur maximale disponible pour la voirie en tout point de l'espace inter-bâti ;

– une périphérie paramétrable (figure 1a) qui favorise l'émergence d'une ceinture (optionnelle) de boulevards et d'avenues en l'absence de liaisons connues avec d'autres villes ;

– le calcul d'un graphe de voirie *G* par « squelettisation homotopique par zones d'influence » (Graffigne, 1995) de l'espace inter-bâti (figure 1b), support de toute configuration possible de réseau viaire médian totalement connecté.

Ce graphe 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. Nous invitons le lecteur à consulter (Marsault, 2009) pour de plus amples détails.

Figure 1. a) Tissu urbain et sa périphérie, b) squelette morphologique homotopique

2.2. Détermination automatique de chemins

Entre deux bâtiments proches, il existe toujours une jonction du graphe. Cette propriété du squelette morphologique (figure 1b) 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, ramenées à une même largeur. L'ensemble des jonctions formant le chemin *c(i,j)* adopte 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 locale » pour déterminer des chemins plausibles entre deux nœuds quelconques de *G,* basée sur une variante *H* de la fonction de valuation des jonctions de l'algorithme de plus court chemin de Dijkstra (Cormen, 2004). En prenant en compte la largeur maximale d'un chemin, sa longueur et sa rectitude, cette nouvelle valuation favorise (selon la valeur du paramètre arbitraire α) les passages les moins tortueux possibles par les endroits où il y a le plus d'espace (figure 2). Cela donne, pour une jonction *j* (longueur l_j, largeur r_j, rectitude $rect_j$) la valuation H_j (1) (rendue strictement positive en mettant « +1 »), dépendant d'un paramètre α arbitraire :

$$H_j = l_j \,.\, rect_j \,.\, (max\ \{\ r_i\ \} + 1 - r_j\ )^{\alpha} \text{ avec } 0 < \alpha \leq 2 \qquad (1)$$

Notons quelques particularités liées à *H*. Pour que la sélection d'une ceinture de boulevards, d'avenues ou de rocades soit activée, il suffit de choisir la distance qui sépare les jonctions périphériques du bâti suffisamment élevée (figure 1b) pour privilégier leur utilisation par l'heuristique (1). *H* peut aussi être adaptée pour prendre en compte des contraintes particulières, comme le relief du sol, et suivre des tronçons de voirie de faible pente.

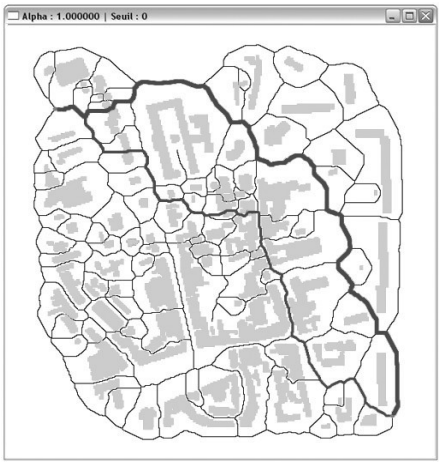


Figure 2. Chemins du réseau viaire potentiel (fin = Dijkstra ; épais = H_j)

On a ensuite cherché à explorer judicieusement le graphe *G* pour effectuer la reconnaissance automatique d'un réseau viaire plausible, en évitant notamment d'avoir à spécifier des critères géométriques locaux pour la détermination et le choix des intersections (problème plutôt délicat). Une solution « émergente » 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.

3. Méthode proposée

Le problème consiste à trouver un ensemble C de chemins de G constituant un plan de voirie optimisé – 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.

On va d'abord exploiter une classification typologique des chemins – dont la fiabilité est très bonne en général – pour identifier au départ tous les chemins admissibles du graphe et leur type, connaissant leurs caractéristiques morphologiques.

3.1. Caractéristiques qualitatives et quantitatives des chemins

L'analyse morphologique de la voirie est issue de mesures sur un échantillon représentatif de villes européennes de tailles variables (Mangin *et al.*, 1999). Elle s'appuie sur quatre caractéristiques (longueur l, largeur r, rectitude r*ect*, et allongement l / r), et se traduit (tableau 1) par des seuils d'admissibilité par type d'élément (avenue, rocade/boulevard, cours, rue, ruelle, cours d'eau). Par exemple, l / r doit être supérieur à un seuil (plus une voie est large, plus elle est longue).

Tableau 1. Paramètres typologiques de la voirie issus de mesures sur les grandes capitales européennes

type d'élément	min(l)	min(r)	max(r)	min(l / r)	max(r*ect*)
ruelle	70 m	3,5 m	5,5 m	20	2
rue	110 m	5,5 m	16 m	20	2
avenue, cours	600 m	16 m	80 m	37,5	1,4
boulevard, rocade	900 m	24 m	80 m	37,5	2
fleuve	∅ (ville)	80 m	400 m	/	/

Cette typologie de voirie permet de classer chaque élément potentiel (tableau 1). En effet, à partir de la technique de cheminement exposée en section 2.2, il est simple de calculer pour un chemin donné ses caractéristiques morphologiques et typologiques en fonction de sa longueur, de sa largeur et de sa rectitude. 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. Ainsi, on adapte les seuils par type de chemin.

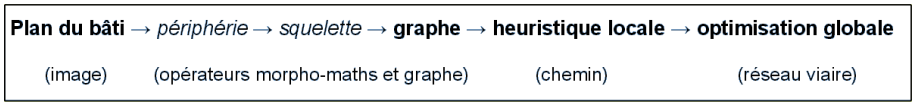
3.2. Obtention de plans de voirie optimisés

L'analyse des plans de voirie montre que les chemins tendent à utiliser au mieux l'espace inter-bâti, tout en étant le plus droit possible. On va donc chercher « en moyenne » des chemins parmi les plus longs, les plus larges, 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 – cf. section 6). La combinatoire du problème est réduite drastiquement en traitant deux niveaux d'optimisation séparés :

– on utilise l'heuristique locale H (1) qui simplifie le choix de chemins entre deux nœuds quelconques du graphe,

– étant donné qu'on ne dispose pas de bons critères locaux pour positionner les extrémités des chemins du plan de voirie, on va se tourner vers l'optimisation combinatoire. L'originalité de notre méthode consiste à mettre en compétition un grand nombre de chemins potentiels présélectionnés pour faire émerger un ou plusieurs plans de voirie définis par l'optimisation globale de critères (dont on teste la pertinence) et la satisfaction de contraintes. Et ce, sans se soucier de savoir où commencent et où finissent les éléments de la voirie.

Ainsi, notre problème n'est plus de calculer des chemins mais d'en sélectionner un sous-ensemble parmi un grand nombre de possibles, pour former un réseau d'axes possédant une typologie, optimisant des critères, et validant des contraintes. On peut alors résumer notre méthode par le diagramme des traitements suivant :



D'emblée, soulignons que les algorithmes évolutionnaires semblent peu adaptés ici, car la construction d'une solution, très contrainte, ne supporte pas de multiples divisions arbitraires (Renders, 2002 ; Deb *et al.*, 2002). En effet, avec les opérateurs de croisement et de fusion propres aux algorithmes génétiques, on perd nécessairement le caractère de non-superposition des chemins, mais aussi, parfois, leur connexité (cf. section 5.2). De telles contraintes justifient le recours à un procédé de construction incrémental, et l'algorithme *Ant-SSP* (section 4.3 et

algorithme 2) fournit une solution robuste et élégante à ce problème. Et nous pensons qu'il est difficile de trouver mieux actuellement.

4. Optimisation par colonies de fourmis

4.1. Optimisation combinatoire

De nombreux problèmes de conception ou de décision – généralement multi-acteurs et multicritères – 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 suffisamment faible de solutions optimales 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 (Talbi, 2009), 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 (algorithme 1), simulant sur plusieurs « lancés » ou « runs » 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 de la colonie. 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 arcs qui favorisent les meilleures solutions.

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 demande à chaque fourmi d'assembler incrémentalement des chemins (c'est l'étape de construction – algorithme 1) 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 des contraintes de consistance (section 5).

Algorithme 1 : ACO générique

Choisir un nombre N_{run} de lancés (ou runs) et n fourmis.
Pour chaque run :

  Initialisation des traces de phéromones

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

Retenir et analyser les meilleures solutions trouvées

4.4. Résolution de problèmes multicritères avec ACO

En présence de m objectifs partiels f_i qui ne sont pas complètement 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 critères à maximiser et d'autres à minimiser).

$$objectif\_mixte = \Sigma\ (\alpha_i \,.\, f_i)\ \ \textit{(combinaison linéaire)} \qquad (2)$$

Mais la plupart du temps, on ne 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-objectif. La différence 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 (Deb *et al.*, 2002). Dans ce cas,

on calcule souvent 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, des techniques de classement de second ordre, comme « la distance de peuplement ». 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 à chaque objectif (ainsi qu'une fonction heuristique), 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 œuvre d'*Ant-SSP*

5.1. Présélection de chemins admissibles

Lors d'une étape de présélection, on calcule et on stocke (en mémoire et dans un fichier XML) pour chaque chemin admissible de *G* : sa longueur, sa largeur, sa rectitude, son type, la liste de ses nœuds *N* et celle de ses jonctions *J.* Cela permet d'en éliminer d'emblée un grand nombre par l'application de seuils (tableau 1), et de diminuer considérablement le nombre de chemins admissibles pour être traités par *SSP*, ce qui permet d'accélérer le processus de recherche de solutions.

5.2. Critères à optimiser

L'objectif de maximiser la longueur globale *L* de tous les chemins 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 :

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

– leur largeur moyenne :

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

– leur rectitude moyenne :

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

– l'écart moyen *E(C)* des largeurs aux intersections – dont on souhaite évaluer le rôle et la pertinence – mesure la variation typologique moyenne aux intersections. Ce paramètre à maximiser est calculé à partir de chaque nœud d'intersection de chemins et de chaque nœud terminal libre de la solution (cf. section 5.4) :

$$E(C)=\sum_{(i,j)\in SkxSk\,|\,ci\cap cj} |r(ci)-r(cj)| \tag{6}$$

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 (algorithme 2). Ce sont les filtres initiaux (tableau 1) qui aident à contrôler 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. Pour l'instant, nous avons laissé cela de côté.

Tableau 2. Quarante critères testés pour l'optimisation (le signe « - » désigne une minimisation). Les zones surlignées correspondent aux meilleurs critères (section 6)

C	-C	L	-L	E	-E
R	-R	-Rect	Σl	-Σl	Σr^{α}l
-Σr^{α}l	**Σ(r^{α}l/rect)**	-Σ(r^{α}l/rect)	Σ(r^{α}l)/E	Σ(r^{α}l).E	**Σ(r^{α}l)/Rect**
L+R-Rect	**L.R^{α}/Rect**	**R/Rect**	Σ(r^{α}l/rect)/E	Σ(r^{α}l/rect).E	**Σ(r^{α}l/rect)/Σl**
E.R/Rect	L.R^{α}.E/ Rect	R.E	Σ(r^{α}l).E/Rect	L.E	L+R-Rect+E
L.R^{α}	**L.R^{α}.E**	L+R	L+R+E	**Σr^{α}l / C**	L.R^{α}/Rect+E
Pareto(L,R)	Pareto(L, R, Rect)	Pareto(L, R, Rect, E)	Pareto(L.R^{α}/Rect, E.R/Rect)		

Quarante critères d'optimisation (tableau 2) ont été conçus à partir de ces paramètres (approche mixte comme L.E qui maximise le produit de L par E, ou multicritère au sens de Pareto comme (L,R,Rect) qui cherche les compromis non dominés sur L, R et Rect), en cohérence avec l'heuristique (1) pour certains. On notera 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. La méta-heuristique d'optimisation par colonies de fourmis *Ant-SSP* permet de calculer les meilleures solutions optimisées qu'ils produisent.

5.3. Construction d'une solution

La méta-heuristique d'optimisation par colonies de fourmis *Ant-SSP* permet de trouver des solutions optimisées pour chaque critère du tableau 2. A chaque cycle d'un run (algorithme 1), n fourmis sont mises en concurrence pour construire incrémentalement un plan de voirie par l'ajout successif de chemins. La morphologie urbaine habituelle nous a déjà fait introduire trois contraintes d'admission pour ces chemins, sous forme de filtres (tableau 1) sur la longueur, la largeur et le rapport l / r. 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'extrémité (connexité faible) ou un nœud 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 rehaussés en moyenne de 20 %,

– 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 conserver 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 nœuds déjà utilisés. Nous proposons de ne plus la considérer comme une contrainte, mais comme un critère d'optimisation propre à la construction (section 5.6).

5.4. 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 œuvre de *Ant-SSP* pour résoudre progressivement l'objectif du minimum d'impasses T_c. Comme il faut placer 2 nœuds *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 nœuds 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 $\delta T(k)$ 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 décroître T_c jusqu'à atteindre le minimum. Chaque choix influe de manière différente sur *T(k),* mais seuls quatre d'entre eux sont potentiellement à réguler, car avec respectivement (*EE, IE, TI, TT),* δT_c = (+2, +1, -1, -2). Et la probabilité d'utiliser (*TT*) devrait être plus faible que celle d'utiliser (*TI)*, car ce choix n'explore pas de nouveaux nœuds. Pourtant, en pratique, on observe que la diminution rapide de T_c se fait en augmentant cette probabilité. De plus, l'usage de (*IE*) et de (*EE*) doit bien sûr être limité, car il s'oppose à la décroissance de *T*. Ainsi, nos expérimentations ont conduit à construire pour chaque cas les fonctions heuristiques $h_c(7)$ avec d'excellents résultats.

$$h_C(IE) = C(k) / (C(k).T_c(k)) \text{ et } h_C(EE) = C(k) / (C(k).T_c(k)) \qquad (7)$$
$$h_C(TE) = h_C(II) = h_C(TT) = h_C(TI) = 1$$

5.5. Mise en œuvre d'Ant-SSP – Construction des solutions

Les chemins favorisant les meilleurs plans sont récompensés grâce à des traces de phéromones virtuelles, et ont une probabilité plus forte d'être sélectionnés au cycle suivant. Deux modélisations phéromonales (vertex, clique) sont généralement proposées (Solnon, 2008) : soit pour associer à chaque composant *i* une trace qui représente l'expérience passée de la colonie concernant l'intérêt de sélectionner l'objet *i*, soit pour associer à chaque paire de composants *(i,j)* une trace représentant l'intérêt de sélectionner *i* et *j* dans le même ensemble. Nous n'avons utilisé que la structure de type vertex, car il ne s'agit pas d'un problème d'ordonnancement, et

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, ce qui est le cas (cf. section 6).

Algorithme 2 : Un « run » d'Ant-SSP mono-objectif (pour simplifier)

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

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:

$$p(Oi) = \frac{h(Oi)^{\alpha} . t(Oi)^{b}}{\sum_{i} h(Oi)^{\alpha} . t(Oi)^{b}} \quad (8)$$

ajouter O_i à S_k
enlever O_i de candidats
enlever de candidats tout O_j non consistant 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 meilleures fourmis du 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}

L'évaluation de chaque objectif lors de la construction est incrémentale 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*, 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).

Pour de nombreux problèmes de graphe, on cale souvent le nombre *n* de fourmis par cycle sur le nombre N_d de nœuds 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 130 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* (algorithme 2) et d'obtenir plus de combinaisons optimales par laps de temps. Les autres réglages optimaux des paramètres de l'algorithme *Ant-SSP* affectés à notre problème 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), $n = 100 \pm 30\%$, $N_c \in [1000 ; 30000]$, $N_{run} = 12$

5.6. Algorithme Ant-SSP incrémental mono-objectif

On note *S* l'ensemble des chemins admissibles de G, ($f_1,...,f_n$) *n* fourmis qui construisent respectivement les solutions ($S_1,..., S_n$) pour chaque cycle d'optimisation, S_{best} la meilleure solution globale et F_{best} sa note d'objectif. L'algorithme 2 décrit le déroulement de *SSP* mono-objectif. A noter qu'on ne peut améliorer les solutions par recherche locale (algorithme 1), qui consiste, à partir d'une solution consistante, à appliquer des modificateurs de voisinage (ex : déplacements de nœuds) respectant les contraintes. En effet, les chemins étant prédéfinis, 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 (tableau 2), et cela fonctionne très bien.

6. Tests et résultats

Afin d'illustrer les précédents exposés théoriques, et analyser quelles solutions émergent naturellement du processus d'optimisation – dans des temps de calculs raisonnables, pour les critères/contraintes retenus – on a choisi un exemple simple (figure 3 – portion de la ville de Vénissieux, avec 284 nœuds et 18 712 chemins admissibles issus du filtrage de 40 470 éléments potentiels), et un tissu plus dense (figure 4 – portion de la ville de Décines, avec 2 580 nœuds et 194 584 chemins admissibles parmi 3 329 490 éléments). Les images et les plans utilisés proviennent de la « base de données géographiques de référence » de la Communauté urbaine de Lyon (Grand Lyon). On a testé l'ensemble des 40 paramètres morphologiques du tableau 2, mais il n'est pas possible de montrer ici tous les résultats, y compris toutes les comparaisons avec le tissu réel.

6.1. Analyse des résultats

La démarche de consolidation pour trouver les meilleurs critères a nécessité, pour chacun d'eux :

– le calcul des meilleures solutions sur un très grand nombre de cycles Ant-SSP (plusieurs dizaines de milliers parfois),

– leur observation fine : caractéristiques visibles, manques, mauvaises détections,

– la comparaison avec le tissu urbain réel (qualitative, peu quantitative).

On mesure d'abord la qualité d'une solution (par rapport à l'existant), par sa capacité à bien reconnaître le plus grand nombre possible d'artères principales (avenues, boulevards, rues importantes), et la localisation de leurs extrémités. La position, la densité et la répartition des rues capillaires sont plus délicates à juger, mais on peut observer la préservation de morphologies réalistes ou plausibles (par exemple un réseau régulier dans un lotissement, avec les rues principales parallèles et les capillaires orthogonaux). Ensuite, on vérifie la préservation des impasses naturelles et des intersections importantes.

La plupart du temps, il faut filtrer les rues capillaires d'une solution calculée, surabondantes et peu en rapport avec le tissu réel, car l'algorithme ajoute toujours des petites rues, tant qu'elles sont admissibles. Un seuillage des éléments de voirie permettrait peut-être d'extraire les axes principaux (non nécessairement classifiés) et de retrouver grossièrement le positionnement et l'allure des îlots. Mais il est important de noter que l'on ne peut pas calculer hiérarchiquement le réseau en traitant séparément chaque type d'élément de voirie, et ce, pour deux raisons. D'abord, parce que la contrainte de connexité peut ne pas être validée dans ce cas.

Ensuite, parce que la dynamique de construction d'une solution par *Ant-SSP* est nécessairement modifiée si l'on construit le réseau par hiérarchies indépendantes.

Sur les 40 critères testés, 7 critères mixtes (14, 18, 20, 24, 25, 32, 35, surlignés en tableau 2) donnent des réseaux assez bien structurés et connectés aux endroits où dans la réalité il existe des connections jugées pertinentes, et évitant en général des groupements de chemins aléatoires. Ils permettent de retrouver une bonne proportion des axes principaux de la voirie existante. On peut remarquer que 5 critères sur 7 sont en concordance avec l'expression algébrique de l'heuristique *H.* On notera encore que l'objectif de minimisation de la rectitude est essentiel dans 5 des 7 meilleurs critères, même si l'heuristique la favorise déjà (ex : critère 20 comparé à 31). On observe enfin sur la ville de Décines un découpage en îlots plausibles (pas réels) – mais avec des perturbations – signe de l'émergence morphologique d'un niveau supérieur à celui des rues. Pour confirmer cette tendance, il serait utile de poursuivre l'étude en filtrant les plans de voirie par axes de largeurs décroissantes.

Enfin, comparées aux meilleures solutions des mono-objectifs mixtes, les solutions du front de Pareto sont plus diversifiées, pas trop nombreuses (quelques dizaines tout au plus), mais assez médiocres. Ce qui nous incite à penser (peut être hâtivement) que la reconnaissance de structure viaire telle qu'on la définit n'est pas un problème multicritère.

6.2. Pertinence des paramètres

Bien que n'ayant pas programmé la minimisation du nombre de chemins (sauf avec le critère 35), les critères (20, 32, 35) favorisant les solutions avec une faible valeur de C témoignent de l'efficacité du regroupement des jonctions en chemins les plus longs, larges et connectés possibles. Le fait de maximiser le produit L.R par exemple, ainsi que les critères (19, 20, 31, 33, 35), diminuent systématiquement C, tandis que (18, 36) l'augmentent mais préservent visiblement mieux les alignements ou les parallélismes naturels des voies des lotissements, par exemple. Les critères (7, 36) apparaissent ici comme un compromis entre ces deux tendances, mais dont la qualité est plus discutable.

Notre hypothèse était que l'introduction du paramètre E dans les critères devait globalement favoriser les raccordements entre chemins de types différents. Or, sa minimisation privilégie systématiquement les solutions avec des chemins peu larges et peu longs, au détriment des autres paramètres (6, 16, 22). Et sa maximisation, qui favorise le raccordement de chemins de largeurs très différentes, agit certes comme un modérateur de longueur (26, 30, 32, 35, 36), mais a tendance à casser la régularité des chemins. E n'est finalement présent que dans deux des bons critères (25, 32).

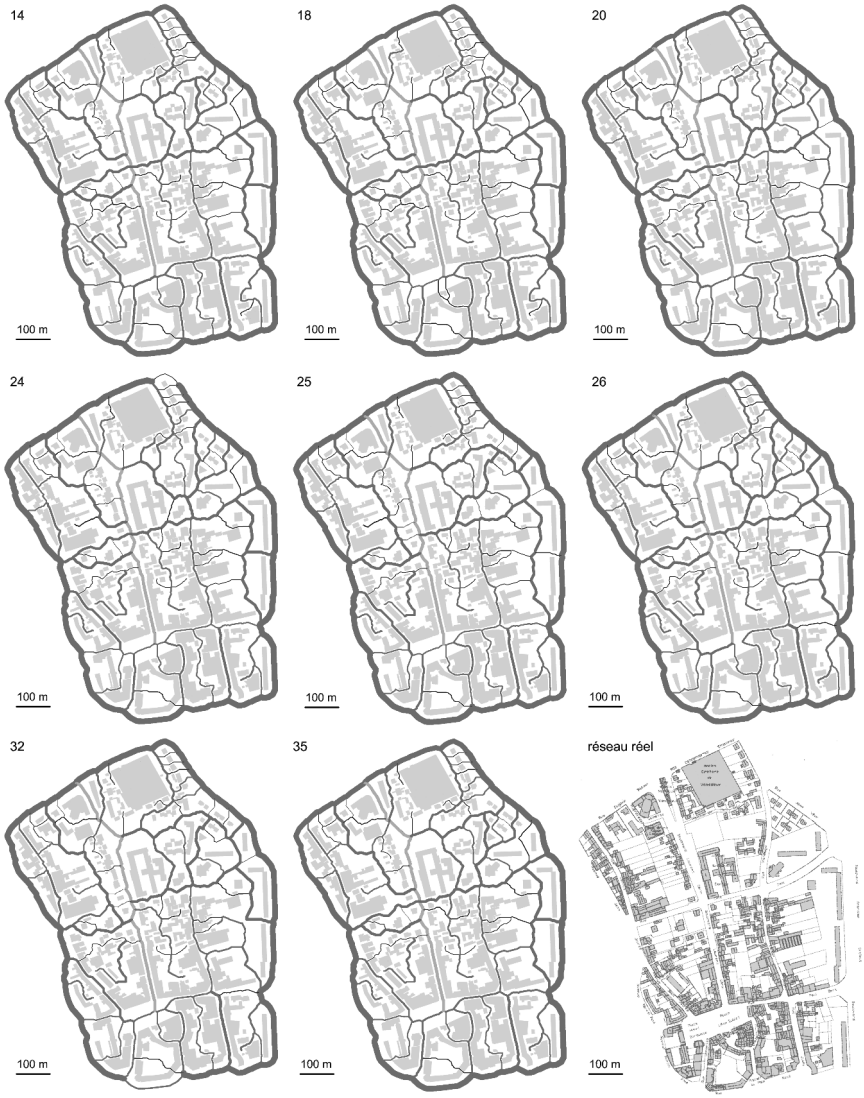


Figure 3. Optimisation des critères (14, 18, 20, 24, 25, 26, 32, 35) sur un tissu test, portion de la ville de Vénissieux. Meilleures solutions sur 5 000 cycles de calcul – Légende : gris clair (bâti existant), gris foncé (boulevards ou rocades), gris moyen (avenues), noir (rues/ruelles) – diamètres proportionnels aux largeurs des axes

Figure 4. Optimisation des critères (14, 24, 25, 35, 18-parcelle) sur une portion de la ville de Décines. Meilleures solutions sur 1 000 cycles de calcul. Légende : gris clair

(bâti existant), gris foncé (boulevards ou rocades), gris moyen (avenues), noir (rues/ruelles) – diamètres proportionnels aux largeurs des axes

Analysons enfin comment le paramètre α de l'heuristique *H* agit sur les solutions produites. Tout d'abord, faire croître α augmente vite le nombre de chemins admissibles et le nombre de chemins sélectionnés *C*, et donc le temps de calcul, sans amélioration notable de la qualité du réseau viaire construit. On va donc limiter α pour minimiser *C*. Cependant, on s'aperçoit qu'il ne faut pas trop diminuer α car on perd l'intérêt de l'heuristique *H*. Au final, les meilleures solutions (au sens des critères d'évaluation précédents) sont obtenues avec des valeurs de α dans [0.9 ; 1.1].

6.3. Complexité de l'algorithme, convergence et temps de calcul

L'optimisation poussée des codes C++ et un réglage fin des paramètres de l'algorithme *SSP* ont permis d'éviter des calculs superflus et contribué à diminuer massivement les temps de recherche. Notamment un bon compromis entre les nombres de cycles et de fourmis par cycle a été trouvé, et les solutions sont déjà très satisfaisantes à partir de 2000 cycles de calcul (figure 5).

Néanmoins, la complexité actuelle de notre algorithme en fonction du nombre de chemins *c* et du nombre de nœuds *n* demeure en O($c.n^{½}$), ce qui – sauf à paralléliser les calculs – conduit à des temps de calculs fastidieux pour de vastes tissus urbains. Certes, nous avons mis au point une version nettement plus efficace de l'algorithme, dont la complexité optimale est en O(*c*).

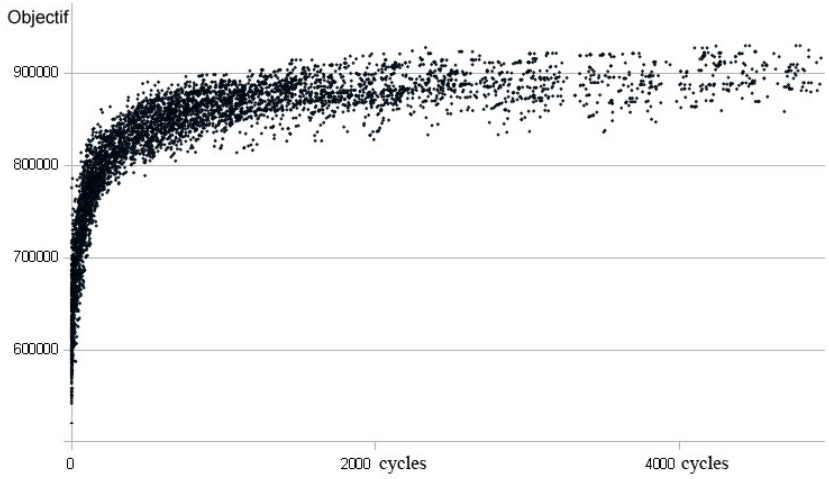


Figure 5. Exemple de graphe de convergence : objectif en fonction du cycle, sur 40 runs (ce qui est nécessaire pour atteindre les meilleures solutions)

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, même pour des tissus urbains de moyenne taille. La seule possibilité actuelle d'amélioration consiste à diminuer c le plus possible, en réalisant des filtrages efficaces. Par ailleurs, le calcul actuel des solutions n'est pas dynamique : une modification du bâti entraîne le recalcul complet. Sur un Intel X5650 (processeur récent), 5 000 cycles de calcul prennent 57 minutes pour les résultats de la figure 3, alors que pour ceux de la figure 4, le calcul de 1 000 cycles requiert 8 heures.

7. Conclusions

7.1. Intérêt, limites de l'approche

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 reconnaissance de voirie urbaine en ne connaissant que la trace des bâtiments au sol. Elle fournit des résultats encourageants, et l'analyse d'un grand nombre de solutions optimisées a permis une certaine consolidation du choix des critères. Qualitativement, on observe nettement sur un large tissu, un découpage en îlots plausibles, mais de petites tailles (figure 4), 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 œuvre, et nous ne saurions conclure si l'optimisation de Pareto fournit réellement de meilleurs résultats que les bons 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 de l'algorithme.

Enfin, et c'est une limite majeure de notre travail, le mode actuel de calcul du squelette, médian et sans lissage, rend approximatif la bonne reconnaissance du contour des îlots, et le calcul des paramètres morphologiques. Par contre, la méthode montre toute son efficacité quand on travaille directement avec une base de données des empreintes parcellaires, même si ce n'était pas notre objectif premier ; et cela permet de la valider à nouveau (figure 4, critère 18-parcelle...) : la reconnaissance de la voirie est alors aisée, et les temps de calculs sont très nettement diminués. Certes, la reconnaissance automatique des parcelles ou des îlots est un problème difficile (Puissant *et al.*, 2010), et on s'est contenté ici d'utiliser la base de données du Grand Lyon. Remarquons cependant que n'importe quel graphe peut être utilisé comme support des calculs d'optimisation pour prendre en compte des

alignements plus réalistes de bâtiments. Il sera bon d'y réfléchir attentivement pour extraire de meilleurs graphes de voirie potentielle, plus proches de la réalité.

7.2. Perspectives

Un traitement des places, des carrefours et des vastes espaces non construits s'avère aussi nécessaire pour éliminer du graphe des éléments issus de la squelettisation et non liés à la voirie, mais qui sont interprétés comme tels, et souvent avec des largeurs importantes. 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 (la piste du calcul parallèle sur GPU sera explorée).

Notre démarche est transposable à d'autres problèmes basés sur l'exploitation d'un graphe spatial, notamment pour optimiser des qualités de la voirie en fonction de l'environnement naturel ou construit (en cours d'étude). Moyennant la donnée de transpositions, dont voici quelques exemples possibles :

– bâtiments → contrainte spatiale, unité de consommation énergétique,

– rue → liaison, élément de réseau,

– graphe → ensemble des relations spatiales et de leurs connexions,

– largeur → débit, flux, intensité, densité, consommation (eau, énergie...),

– heuristique H (chemins plausibles) → toute optimisation locale du problème,

– critères d'optimisation → propres à chaque problème,

et l'utilisation de la carte des empreintes pour transférer au graphe G multivalué les données nécessaires, ce type de modélisation peut être adapté à des problèmes d'ingénierie urbaine basés sur l'exploitation d'un graphe et de données : sociales (ex : densité de population, type de transport et flux associés), environnementales (ex : nature du sol) ou énergétiques (ex : flux, consommation, coût).

On peut ainsi envisager : d'améliorer un réseau viaire existant en étudiant des variantes, de résoudre un problème économique (ex : minimum de chemins déservant le maximum de lieux, asservis à la densité de population et à une fourniture énergétique...), ou encore de calculer des plans de circulation ou de faire de la simulation de trafic sur un réseau. Au laboratoire MAP-ARIA, ce travail s'intègre dans le développement d'un outil de simulation de réseaux urbains aériens ou souterrains (voirie, réseaux techniques...) s'appuyant sur l'optimisation multicritère.

Remerciements

A Christine Solnon, pour son accueil, sa gentillesse, ses conseils et sa patience. A Jean-Baptiste Denis pour ses codes C++ très structurés qui ont été une base solide et bien utile pour implémenter les algorithmes d'optimisation et de visualisation.

Bibliographie

Alaya I., Solnon C., Ghedira K. (oct 2007). Ant Colony Optimization for Multi-objective Optimization Problems, *19th IEEE International Conference on Tools with Artificial Intelligence*. (ICTAI), Patras, Greece.

Angus D., Woodward C. (2009). Multiple objective ant colony optimisation, *Swarm Intelligence*, Springer, vol. 3, p. 69-85.

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

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

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

Deb K., Pratap A., Agrawal S., Meyarivan T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II, *IEEE Transactions on Evolutionary Computation*, vol. 6, n° 2.

Decoret X., Sillion F. (2002). *Street Generation for City Modelling, Architectural and Urban Ambient Environment*, rapport interne INRIA.

Denis J.B. (2004). *Calcul automatique de voiries et de places d'une ville représentée par l'image des empreintes de ses bâtiments*, rapport de Magistère Informatique de Paris.

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

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

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

Lacomme P., Prins C., Sevaux M. (2001). *Algorithmes de graphes*, Eyrolles.

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

Mangin D., Panerai P. (1999). *Projet urbain*, Editions parenthèses.

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

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

Puissant A., Lachiche N., Skupinski G., Braud A., Perret J. (17-19 novembre 2010). Classification des tissus urbains à partir de données vectorielles – application à Strasbourg, *Conférence Internationale SAGEO (Spatial Analysis and GEOmatics)*, Toulouse, France.

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

Solnon C. (2008). *Optimisation par colonies de fourmis*, Hermès.

Talbi E., (2009). *Metaheuristics: From Design to Implementation*, Editions Wiley.