Méthodes de calcul et d'optimisation de tournées d'un transport à la demande en convergence

Sommaire

6.1 Le cas de la convergence simple 118
6.1.1 Un algorithme d'énumération exhaustive 118
6.1.2 Un algorithme de parcours en largeur 122
6.1.3 Un algorithme génétique 125
6.2 Le cas du TAD en multiconvergence 130
6.2.1 Utilisation de NSGA-II 130
6.2.2 La moyenne généralisée dans la minimisation du temps 132
6.2.3 La norme L_p dans la minimisation des retards 133

Introduction

Les méthodes présentées dans ce chapitre fournissent chacune une solution à l'un des problèmes de TAD posés et formalisés dans le chapitre 4.

Leur conception s'inscrit dans la continuité d'un projet plus ancien, « Evolis-Gare », dévolu à la Communauté d'Agglomération du Grand Besançon et développé par D. Josselin, C. Lang et P. Chatonnay (Canalda *et al.*, 2004). Ainsi notre première étape de réflexion reprend le principe d'énumération exhaustive des ACT, qui constitue une première solution aux problèmes opérationnels de desserte, de tailles modestes (moins de 20 demandes).

Face à des problèmes de TAD de plus grande taille, l'énumération exhaustive des ACT s'essouffle rapidement. Ainsi le deuxième algorithme proposé est un parcours en largeur (méthode exacte), qui nous a permis d'aborder des problèmes de plus grande taille. Toutefois, pour traiter des problèmes encore plus complexes et repousser les limites de l'optimisation, il était nécessaire de développer une méthode heuristique. L'algorithmique génétique nous a paru particulièrement indiquée. En effet, son efficacité a déjà été prouvée pour de nombreux problèmes et elle est réputée pour être particulièrement adaptée à la résolution de problèmes multi-objectifs (Deb, 2001; Coello *et al.*, 2002). Elle permet en outre d'obtenir plusieurs solutions alternatives à soumettre au décideur, ce qui répond bien aux attentes dans le cadre du TAD. Par conséquent, un algorithme génétique dont l'encodage d'un ACT est directement calqué sur celui du parcours en largeur a été également développé.

Cette première version d'algorithme génétique nous conduit finalement à généraliser une méthode d'optimisation territoriale de TAD multi-objectifs basée sur une approche Pareto. Nous introduisons notamment la notion de normes dans l'évaluation du temps nécessaire aux différentes courses.

6.1 Le cas de la convergence simple

Dans cette section, nous étudions trois algorithmes de calcul des ACT (Chevrier *et al.*, 2006a). Les deux premières méthodes sont exactes et la troisième utilise les métaheuristiques. Dans le détail, la première méthode est un algorithme d'énumération exhaustive des ACT sur un DAG aux arcs non pondérés. La deuxième méthode, développée consécutivement, est un algorithme de parcours en largeur calculant directement les solutions optimales sur un DAG pondéré.

La troisième méthode est un algorithme génétique mono-objectif qui préfigure la méthode généralisée que nous étudions en deuxième section.

6.1.1 Un algorithme d'énumération exhaustive

L'algorithme d'énumération exhaustive (AEE), réalisé en étroite collaboration avec P. Canalda, est une construction itérative des branches de l'ACT en démarrant des nœuds minimaux du DAG (Chevrier *et al.*, 2006c). Un sommet est dit « minimal » lorsque celui-ci n'a pas de prédécesseurs (pas d'arcs incidents). Les nœuds minimaux sont nécessairement des feuilles des ACT à construire. *A contrario* le nœud terminal du DAG,

qui n'a pas de successeurs (i.e. pas d'arcs sortants), constitue la racine de l'arbre (qui correspond en fait à l'*antiracine* vu l'orientation des arcs).

D'une manière générale, la racine de l'arbre correspond au point de convergence de la desserte, tandis que les sommets minimaux sont des points de départ des véhicules. Ainsi la première étape de l'AEE consiste à identifier les nœuds minimaux du DAG qui sont les points de départ des ACT. Si nous appliquons l'AEE sur le DAG $D(V, E)$ de la figure 6.1, les sommets minimaux sont les points 0 et 1. Ensuite pour chaque sommet $v \in V$ ni minimal, ni terminal, on crée une nouvelle branche b avec le sommet v comme point de départ, que l'on ajoute à chaque ACT. Ces nouveaux ACT sont ajoutés à une liste intermédiaire à fusionner à la liste courante des ACT à la fin de chaque itération. Autrement, pour chaque ACT de la liste courante, nous cherchons si l'on peut fusionner l'arc incident comprenant le sommet v avec une des branches de l'ACT.

De cette manière, nous étendons progressivement et incrémentalement l'ensemble des ACT, d'une part en créant de nouvelles branches et d'autre part en fusionnant éventuellement des arcs incidents aux branches des ACT partiels déjà constitués. L'algorithme ci-après décrit formellement la méthode que nous venons d'énoncer.

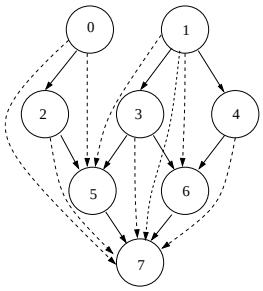


FIG. *6.1:* *DAG $D(V, E)$ à huit sommets.*

Algorithme d'énumération exhaustive des ACT

- v un sommet de V et $V_{min} \subset V$ l'ensemble des sommets minimaux du DAG ;
- act un ACT ;
- act' un ACT temporaire ;
- b une branche de l'ACT ;
- L_{act} la liste des ACT à construire ;
- L'_{act} la liste intermédiaire des ACT ;
- *racine* le nœud terminal du DAG ;
- *ajout* : un booléen.

1. **Pour chaque** $v \in V_{min}$ **faire**
   (a) $b = (v)$ # Création d'une branche b avec v pour point de départ
   (b) $act = act + b$ # Ajout de la branche b au premier ACT
   (c) $L_{act} = L_{act} + act$ # Ajout de act à la liste des ACT

   Fait
2. **Pour chaque** $v \in V \setminus \{V_{min} + racine\}$ **faire**
   Pour chaque $act \in L_{act}$ **faire**
   (a) $b = (v)$ # Création d'une branche b avec v pour point de départ
   (b) $act' = act + b$ # ajout de la branche à l'ACT
   (c) $L'_{act} = L'_{act} + act'$ # ajout de l'ACT intermédiaire à la liste intermédiaire des ACT
   (d) $ajout = faux$
   (e) **Pour chaque** $b \subset act$ et $ajout = faux$ **faire**
      i. $e = dernier(b)$ # Obtention du dernier élément de b
      ii. **Si** $\exists \langle ev \rangle \in E$ **alors**
         $b = b + v$ # On étend la branche b avec v
         # v est le dernier élément de b
         $ajout = vrai$ # pour sortir de la boucle...

      Fait
   (f) $L_{act} = L_{act} \cup L'_{act}$ # Ajout de la liste intermédiaire à la liste finale

   Fait

   Fait

Si l'on déroule l'AEE sur le DAG D, nous pouvons suivre pas à pas la construction incrémentale de la liste exhaustive des ACT. Les figures 6.2 indiquent les trois premières étapes de la construction des ACT avec l'AEE. Les détails de la construction complète sont indiqués dans le tableau A.1 en annexe.

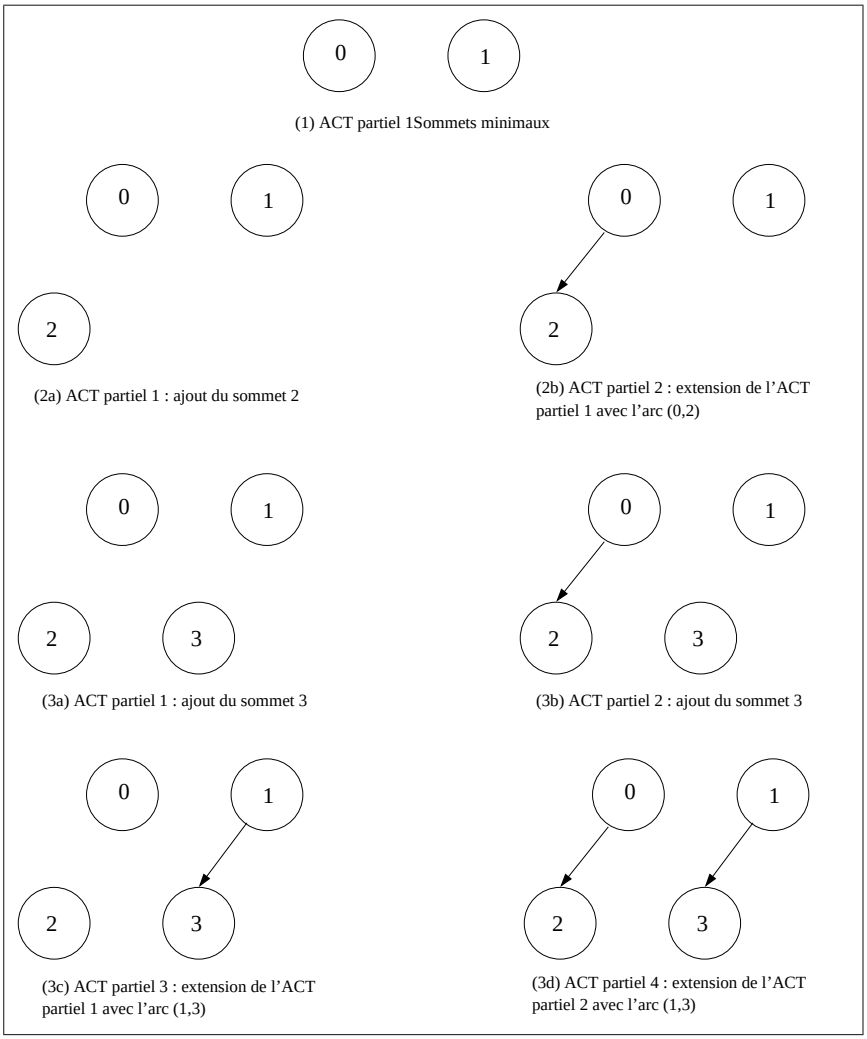


FIG. 6.2: *Les trois premières étapes de l'AEE appliqué au DAG de la figure 6.1*

6.1.2 Un algorithme de parcours en largeur

L'algorithme de parcours en largeur, proposé ici, cherche récursivement la solution optimale (Chevrier *et al.*, 2006a). Contrairement au précédent algorithme qui énumérait exhaustivement les ACT du DAG proposé, nous nous contentons dans ce cas de mémoriser le meilleur ACT trouvé, compte tenu d'un ensemble de critères $K = \{k_i\}$ évalués dans une fonction de coût C.

En détails, les éléments intervenant dans l'algorithme explicités ci-après sont les suivants :

- $G(V, E)$: le DAG à parcourir avec $V = \{v_i\}$ l'ensemble des sommets v, E l'ensemble des arcs et $V_{min} \subset V$ l'ensemble des sommets minimaux ;
- $K = \{k_i\}$: l'ensemble des critères k intervenant dans l'évaluation d'un ACT ;
- α : l'ACT en cours à évaluer. α_i indique le numéro du véhicule desservant le sommet de rang i ;
- A : le meilleur ACT trouvé, initialisé avec $\forall i < |V| - 1, A_i \leftarrow i$ (i.e. un véhicule par demande), par défaut le meilleur ACT attribue un véhicule par demande ;
- $\mathcal{A}$: l'ensemble des meilleurs ACT s'il y en a plusieurs égaux ;
- $\Lambda = \{\lambda_i\}$: l'ensemble des véhicules disponibles ;
- $C : \alpha \times K \rightarrow c$: la fonction évaluant le coût c d'une solution α compte tenu de l'ensemble des critères K ;
- $g : \alpha \rightarrow \{0|1\}$: une fonction vérifiant que la solution α est cohérente (valeur 1, sinon 0).

L'algorithme $f(\alpha, i)$ ci-après consiste à attribuer le minimum de véhicules à l'ensemble des requêtes correspondant aux différents points du graphe (V). La contrainte dans cette attribution consiste à vérifier la validité des chemins empruntés par les véhicules à l'aide de la fonction g. Le coût d'une solution, à minimiser, est donné par la fonction C. Cet algorithme est mono-objectif.

Algorithme $f(\alpha, i)$

Si $i > |V| - 1$ **alors**

Si $C(\alpha, K, |V| - 1) < C(A, K, |V| - 1)$ **alors** # le coût de α est plus faible que celui de A

$A \leftarrow \alpha$ # α devient le nouvel ACT optimal

$\mathcal{A} = \{A\}$ # l'ensemble des ACT optimaux est réinitialisé à A

Si $C(\alpha, K, |V| - 1) = C(A, K, |V| - 1)$ **alors** # α et A ont des coûts équivalents

$A \leftarrow \alpha$ # α devient le nouvel ACT optimal

$\mathcal{A} = \mathcal{A} \cup \{A\}$ # on ajoute A à l'ensemble des ACT optimaux

Sinon

Pour λ de 1 à $|\Lambda|$ **faire** # on itère sur les véhicules disponibles

$\alpha_i \leftarrow \lambda$ # le sommet i de α est desservi par le véhicule λ

Si $g(\alpha, i) = 1$ **alors** # l'ACT α est-il cohérent ?

Si $C(\alpha, K, i) \leq C(A, K, i)$ **alors** $f(\alpha, i + 1)$ # si α est meilleur que A jusqu'au sommet i, on lance la récursion sur sur α à partir du sommet $i + 1$

Fait

L'application de l'algorithme au DAG D est décrite par l'exécution pas à pas dans le tableau 6.1. La fonction C indique dans ce cas le nombre de branches de l'ACT α (égal au nombre de véhicules requis par la solution), de telle sorte que l'algorithme minimise le nombre de véhicules nécessaires pour assurer une desserte optimale :

$$C : \alpha \times K \times i \rightarrow |\alpha|$$

Préalablement, certaines variables sont initialisées. Ainsi l'ensemble Λ comprend autant d'éléments qu'il y a de nœuds minimaux du graphe :

$$\forall v_{min} \in V_{min}, V_{min} \subset V, \Lambda = \Lambda \cup \{\lambda = v_{min}\}$$
$$\text{donc initialement } \Lambda = \{0, 1\}$$

L'ACT optimal A initial correspond à l'affectation d'un véhicule par requête :

$$A = ((0,7); (1,7); (2,7); (3,7); (4,7); (5,7); (6,7))$$

Le premier sommet traité correspond au premier nœud non-minimal : $i = 2$

λ	i	Λ	α	$C(\alpha,i) \leq C(A,i)$
0	2	0,1	((0,2,7) ;(1,7) ;(3,7) ;(4,7) ;(5,7) ;(6,7))	oui : $f(\alpha, i+1)$
0	3	0,1	faux	
1	3	0,1	((0,2,7) ;(1,3,7) ;(4,7) ;(5,7) ;(6,7))	oui : $f(\alpha, i+1)$
0	4	0,1	faux	
1	4	0,1	faux	
		0,1,**2**	Ajout d'un véhicule : $\Lambda = \Lambda \cup \{2\}$	oui : $f(\alpha, i+1)$
2	4	0,1,2	((0,2,7) ;(1,3,7) ;(4,7) ;(5,7) ;(6,7))	oui : $f(\alpha, i+1)$
0	5	0,1,2	((0,2,5,7) ;(1,3,7) ;(4,7) ;(6,7))	oui : $f(\alpha, i+1)$
0	6	0,1,2	faux	
1	6	0,1,2	((0,2,5,7) ;(1,3,6,7) ;(4,7))	oui : $f(\alpha, i+1)$ $C(\alpha,i) < C(A,i) \Rightarrow A \leftarrow \alpha, \mathcal{A} = \{A\}$
2	6	0,1,2	((0,2,5,7) ;(1,3,7) ;(4,6,7))	oui : $f(\alpha, i+1)$ $C(\alpha,i) < C(A,i) \Rightarrow A \leftarrow \alpha, \mathcal{A} = \mathcal{A} \cup \{A\}$
1	5	0,1,2	((0,2,7) ;(1,3,5,7) ;(4,6,7))	oui : $f(\alpha, i+1)$
0	6	0,1,2	faux	
1	6	0,1,2	faux	
2	6	0,1,2	((0,2,5,7) ;(1,3,7) ;(4,6,7))	oui : $f(\alpha, i+1)$ $C(\alpha,i) < C(A,i) \Rightarrow A \leftarrow \alpha, \mathcal{A} = \mathcal{A} \cup \{A\}$
2	5	0,1,2	faux	
2	3	0,1,2	faux	
2	2	0,1,2	faux	
				Terminé : $\mathcal{A}$ ={((0,2,5,7) ;(1,3,6,7) ;(4,7)) ; ((0,2,5,7) ;(1,3,7) ;(4,6,7)) ; ((0,2,5,7) ;(1,3,7) ;(4,6,7)) }

TAB. *6.1:* *Application de l'algorithme de parcours en largeur au DAG de la figure 6.1*

6.1.3 Un algorithme génétique

Cette partie de la thèse a été réalisée avec le support de P. Chatonnay, Maître de Conférences à l'Université de Franche-Comté.

Suite au développement d'algorithmes exacts, nous nous sommes orientés vers une résolution bâtie sur un AG. Notre choix est motivé par l'existence de plusieurs avantages tels que :

- l'aspect multi-objectifs, c'est-à-dire la capacité de l'AG à optimiser plusieurs objectifs éventuellement antagonistes (dans le cadre de la monoconvergence, l'AG presenté agrège un ensemble d'objectifs) ;
- la génération de plusieurs solutions différentes équivalentes, offrant ainsi une plus grande marge de manœuvre aux AOT pour choisir les meilleures tournées ;
- l'efficacité des AG, qui se traduit notamment par leur capacité à produire de bonnes solutions dans un contexte de forte montée en charge de la demande.

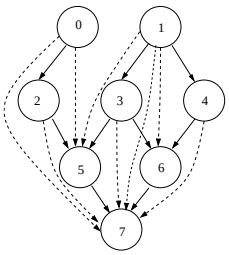
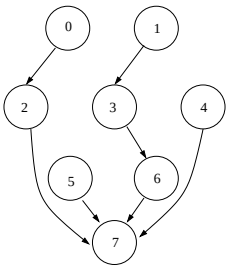
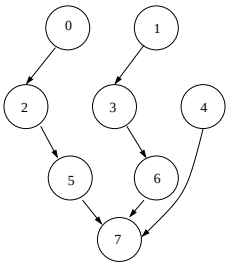
Représentation de la solution et initialisation de la population

Le chromosome représentant l'ACT se traduit par un tableau unidimensionnel où chaque cellule (gène) correspond à un point de ramassage (« *pickup* », pkp). La valeur du gène indique quant à elle le numéro du véhicule desservant le point. Les points de passage étant ordonnés selon leurs horaires, la séquence du chromosome (de gauche à droite) indique l'ordre de passage des véhicules aux différents arrêts.

Comme nous l'avons souligné précédemment, grâce au DAG dressé avec la convergence des flux, nous pouvons d'ores et déjà introduire les premières optimisations structurelles dans la représentation chromosomique. Comme un nœud minimal est nécessairement un point de départ d'une tournée, le numéro de ce point est identique au numéro du véhicule. Le tableau 6.2 reprend le problème de desserte que nous avons représenté auparavant, ainsi que deux ACT optimaux, pour lesquels nous indiquons leurs représentations chromosomiques. Les nœuds minimaux 0, 1 du graphe $G(V, E)$ de la figure 6.2 sont desservis par les véhicules 0, 1. Comme il y a, au plus, autant de véhicules que de points de ramassage, nous affectons aux derniers gènes des valeurs aléatoires comprises entre 0 et $n = |V| : 0 \leq \omega \leq n$

Ainsi, avec n points de ramassage et n_{min} points minimaux ($n_{min} < n$),

$$\forall i \leq n, \text{gène}_i = \begin{cases} i & \text{si } i \leq n_{min} \\ \omega \in [0; n] & \text{si } i > n_{min} \end{cases}$$

	$0 : 2 \rightarrow 5 \rightarrow 7 \rightarrow /$ $1 : 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow /$ $2 : 5 \rightarrow 7 \rightarrow /$ $3 : 5 \rightarrow 6 \rightarrow 7 \rightarrow /$ $4 : 6 \rightarrow 7 \rightarrow /$ $5 : 7 \rightarrow /$ $6 : 7 \rightarrow /$ $7 : /$	g_0 g_1 g_0 g_2 g_1 g_0 g_3
	((0,2,7) ;(1,3,6,7) ;(4,7) ;(5,7))	**0** **1** 0 1 2 3 1
	((0,2,5,7) ;(1,3,6,7) ;(4,7))	**0** **1** 0 1 2 0 1

TAB. 6.2: *Un DAG à 8 nœuds et deux ACT : le DAG est défini par ses listes d'adjacences et on lui associe une représentation chromosomique*

De plus, la convergence étant intrinsèquement connue, il n'est pas nécessaire d'indiquer le nœud 8 sur le chromosome. Nous pouvons donc faire l'économie de ce gène sur la représentation. Cette seconde optimisation structurelle due à un caractère géographique contribue à accélérer l'optimisation. Cela est d'autant plus vrai dans le cas particulier de monoconvergence à plusieurs dessertes, où nous pouvons dans ce cas économiser plusieurs gènes sur la représentation.

Évaluation

Avant de sélectionner les individus qui vont être croisés, ceux-ci sont dans un premier temps évalués à l'aide d'une fonction « objectif » Φ (*fitness function*) qui attribue un score φ_i à chaque individu i. C'est à ce niveau que les capacités k_λ des véhicules $\lambda \in \Lambda$ sont prises en compte.

Bien que le problème soit multi-objectifs, nous considérons le problème comme un agrégat d'objectifs, en sommant les différents scores obtenus sur chacun des objectifs, que nous maximisons. Ainsi le chromosome ζ a pour fitness :

$$\varphi_\zeta = \Phi(\zeta) = \varphi_1 + \varphi_2$$

Cette approche a été retenue dans ces premiers développements. Elle est complétée par la suite par une approche multi-critères avec front de Pareto.

Les objectifs φ_1, φ_2 sont définis ainsi (par ordre d'importance) :

1. φ_1 : minimiser le nombre de véhicules $\lambda \in \Lambda$ (ou indirectement maximiser les taux de remplissage ou encore maximiser le nombre de sommets traversés par le véhicule λ) : $\max \dfrac{1}{|\Lambda|}$.
2. φ_2 : minimiser les temps des courses t_j (*maximiser l'inverse*) : $\forall j \leq |\Lambda|, \max \dfrac{1}{\sum t_j}$;

Comme les objectifs sont agrégés, nous sommons les deux scores précédents pour affecter un *fitness* φ_ζ à l'individu ζ. Néanmoins, ces deux scores ne sont pas comparables, d'où l'importance de se réorienter ensuite vers une approche multi-objectifs.

De plus les objectifs sont sujets aux contraintes suivantes :
– le trajet $x^* \rightarrow y^*$ est possible si et seulement si l'arc $\langle x^* y^* \rangle$ existe ;
– les trajets $x^* \rightarrow y^*$ sont cohérents en temps : $h_{x^*} + t_{x^* \rightarrow y^*} \leq h_{y^*}$;
– les temps de parcours t_j (un parcours par véhicule λ) sont inférieurs à une durée t_{max} : $\forall j \leq |\Lambda|, t_j \leq t_{max}$;
– les véhicules sont en nombre limité et ont des capacités fixes en termes de places assises.

Sélection

Nous optons ici pour la méthode avec « roulette des scores » (*the score wheel method*) qui choisit aléatoirement un chromosome parmi la population. Cependant, un chromosome a une probabilité d'être choisi relative à son score, plus celui-ci est élevé, plus sa chance d'apparition lors de la sélection est importante.

Croisement

Le croisement est une application de : $C \times C \rightarrow C \times C$ permettant de brasser la population. Il s'effectue sur des paires de chromosomes choisis durant la phase antérieure

de sélection et consiste à reproduire les chromosomes des parents et à échanger des séquences des chromosomes, produisant ainsi de nouveaux individus au patrimoine génétique brassé.

Parmi les très nombreuses méthodes de croisement, nous optons pour le croisement à deux points. La méthode consiste à choisir deux points de croisement p, q aléatoirement sur des chromosomes de taille l, $p < q < l$, et les séquences (pq) sont échangées pour créer les enfants de taille l (cf. fig 6.3).

De plus, comme les premières séquences des chromosomes sont identiques (i.e. les nœuds minimaux), il n'est pas utile d'opérer un croisement à partir de ces sous-séquences communes. Notons g_c le premier gène à partir duquel l'on peut opérer un croisement (après la sous-séquence commune). Nous échangeons donc des séquences de taille aléatoire entre g_c et le dernier gène des chromosomes : $g_c \leq p$.

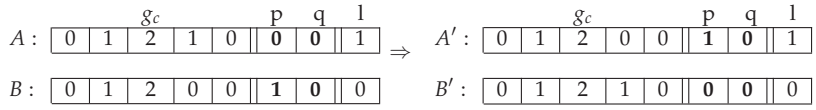


FIG. 6.3: *Croisement à deux points*

Mutation

La mutation est une application de : $C \rightarrow C$, visant à introduire de la diversité dans le patrimoine génétique d'une population, et éventuellement à dégager de nouvelles solutions qui auraient été laissées de côté lors de la recherche itérative.

Bien que le croisement à deux points soit suffisant pour faire converger une population vers un ensemble de solutions sub-optimales voire optimales, nous pouvons utiliser la mutation pour accélérer la recherche de bonnes solutions et faire converger la population vers les puits locaux de solutions optimisées.

L'opérateur de mutation orienté convergent (Chevrier *et al.*, 2006b) procède de la manière suivante. Comme nous souhaitons minimiser le nombre de véhicules présents dans la solution (n_v), il suffit d'exploiter l'opérateur « *modulo* » (donnant le reste d'une division euclidienne) pour maintenir ou mieux, réduire le nombre de véhicules. Un gène est choisi aléatoirement et sa valeur est sommée à une valeur entière aléatoire ω, on applique *modulo* n_v sur cette nouvelle valeur (g'_i) pour la diminuer (ou la maintenir

le cas échéant) :

$$\forall i, g_c \leq i \leq |C|, g'_i = (g_i + \omega) \; modulo \; n_v$$

Dans l'exemple de la figure 6.4, le gène en position 3 est choisi au hasard, et on lui ajoute une valeur aléatoire $\omega = 5$. Sur ce chromosome, nous dénotons la présence de quatre véhicules nécessaires ($n_v = 4$). Donc la mutation agit en produisant le gène muté g'_3 :

$$g'_3 = (g_3 + \omega) \; modulo \; n_v \tag{6.1}$$

$$g'_3 = (2 + 5) \; modulo \; 4 \tag{6.2}$$

$$g'_3 = 3 \tag{6.3}$$

Suite à cette mutation, il n'y a plus que trois véhicules (allèles 0,1,3). Si l'on minimise les allèles de telle sorte que les valeurs des gènes constituent un ensemble « continu » (ici l'allèle 3 devient 2 pour qu'il n'y ait plus que les allèles contigus 0,1,2), le chromosome muté est équivalent à celui correspondant à l'ACT_2, qui est minimal (fig. 6.5).

0	1	0	**2**	1	0	3	⇒	0	1	0	**3**	1	0	3

FIG. 6.4: *Exemple de mutation : passage de 4 à 3 véhicules*

0	1	0	**3**	1	0	**3**	⇔	0	1	0	**2**	1	0	**2**

FIG. 6.5: *Minimisation des allèles*

Génération de la nouvelle population

Une fois la mutation effectuée, nous sommes presqu'au terme de l'itération i de l'algorithme génétique canonique, il reste maintenant à générer la population $\mathcal{P}_{i+1}$ à partir de la population $\mathcal{P}$ augmentée de sa descendance $\mathcal{E}$, ses enfants.

Diverses politiques de renouvellement de la population existent, simple ou élitiste. Dans le cas d'un renouvellement de population simple, seuls les enfants générés constituent la nouvelle population. Cependant, dans le cas d'un renouvellement élitiste comme celui appliqué dans notre méthode, les chromosomes enfants sont ajoutés à la population temporaire elle-même déjà constituée des parents. Tous ces invidus sont évalués

et classés selon leur score. Seuls les n meilleurs chromosomes sont conservés, pour revenir à la taille de population initiale à n individus.

Critiques de la méthode

Restreignant l'évaluation d'une solution à la maximisation d'un agrégat d'objectifs, nous pouvons craindre une convergence des solutions vers des puits locaux de solutions et que l'AG occulte des solutions optimales. La seule alternative pour en sortir consiste à augmenter la probabilité de mutation (Goldberg, 1989) comme nous le montrons dans le chapitre 7, lors des simulations réalisées.

6.2 Le cas du TAD en multiconvergence

Pour assurer un service de TAD en convergence multiple, nous ne développons pas d'approches exactes, trop lentes face à la complexité et à la montée en charge des demandes. Toutefois, nous poursuivons dans la lignée des approches métaheuristiques et à cette fin nous faisons évoluer notre AG d'une version élitiste à agrégat d'objectifs vers un AG élitiste avec approche Pareto, de type NSGA-II, et ce, pour plusieurs raisons :

- dans un contexte opérationnel, nous avons besoin de bonnes solutions rapidement et NSGA-II est pourvu d'une méthode de tri particulièrement rapide (d'une complexité totale de $O(mN^2)$) ;
- l'approche Pareto permet d'explorer complètement l'ensemble des solutions et d'éviter le cantonnement à un puits local de solutions.

6.2.1 Utilisation de NSGA-II

La fonction d'évaluation Φ de notre algorithme tend à minimiser quatre objectifs, respectivement quatre valeurs $\varphi_1, \varphi_2, \varphi_3, \varphi_4$ (plus le score est faible, meilleur il est) :

- maximiser le nombre de points desservis n_d (sous contrainte de la validité des chemins) : $\varphi_1 = -\dfrac{n_d}{n_p}$ (n_p est le nombre de points à desservir), la meilleure valeur est $\varphi_1 = -1$, indiquant que tous les points sont desservis. Si cette valeur n'est pas atteinte, la solution est fausse. Cet objectif est également une contrainte (obligation de service) ;
- minimiser le nombre de véhicules utilisés (ensemble Λ) : $\varphi_2 = |\Lambda|$;

– minimiser les temps de parcours t_λ de chaque véhicule λ. La valuation de cet objectif φ_3 est réalisée à l'aide de la moyenne généralisée de Hölder étudiée dans la section 6.2.2 ;
– minimiser les retards D_λ de chaque véhicule λ éventuellement accumulés à chaque requête. Pour ce faire, nous intégrons la distance de Minkowski (section 6.2.3) dans la réduction de ces retards.

Nous utilisons une flotte hétérogène de véhicules légers (5 et 8 places) comme dans le cadre d'un véritable TAD. La taille des véhicules a certes un lien direct avec le nombre de véhicules à utiliser, mais aussi avec les temps de parcours. En effet, la maximisation des taux de remplissage est liée à la minimisation du nombre de véhicules à utiliser et impacte fortement les temps de parcours.

Ainsi les objectifs φ énoncés sont sujets aux contraintes $\Psi = (\psi_1, \psi_2, \psi_3)$ suivantes :

– ψ_1 : le nombre de véhicules $n_\lambda, \lambda \in \Lambda$ est limité : $n_\lambda = |\Lambda|$;
– ψ_2 : les capacités k_λ des véhicules λ, $k_\lambda \in K$;
– ψ_3 : le nombre de places restantes (i.e. cela permet de maximiser les taux de remplissage).

Quatre objectifs impliquent l'utilisation de quatre axes pour représenter l'espace de solutions $\mathcal{E}$. Néanmoins, une solution juste implique nécessairement $\varphi_1 = -1$, la solution étant fausse le cas échéant. Si l'on restreint l'analyse et la comparaison des solutions à celles étant valides, on peut réduire à trois dimensions l'espace de solutions $\mathcal{E}$, caractérisé par les objectifs $\varphi_2, \varphi_3, \varphi_4$ (respectivement les véhicules, les temps de parcours et les retards).

Les solutions ainsi obtenues (et valides) se répartissent sur un front de Pareto s'étirant sur trois axes (fig. 6.6 véhicules, temps, retards).

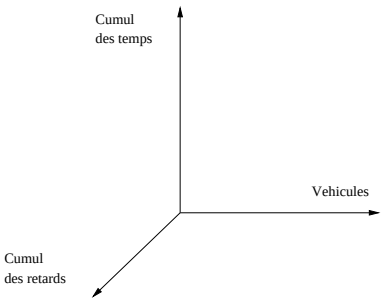


FIG. 6.6: *Mise en concurrence des objectifs d'optimisation : véhicules/temps/retards.*

6.2.2 Intégration de la moyenne généralisée dans la minimisation du temps

La moyenne généralisée de Hölder (*power mean* ou *generalized mean*) généralise les calculs de moyenne (Bullen, 2003) et établit pour un ensemble de valeurs $X = \{x_1, ..., x_n\}$ la moyenne $\overline{X}_q$:

$$\overline{X}_q = \left(\frac{1}{n} \sum_{i=1}^{n} x_i^q \right)^{\frac{1}{q}}, i, n \in \mathbb{N}, q \in \mathbb{R}$$

À travers cette abstraction l'on retrouve les cas particuliers de moyennes définies ainsi :

$$\begin{array}{lcll}
\text{moyenne arithmétique} & : & \overline{X}_{q=1} & = \dfrac{1}{n} \sum_{i=1}^{n} x_i \\
\text{moyenne quadratique} & : & \overline{X}_{q=2} & = \left(\dfrac{1}{n} \sum_{i=1}^{n} x_i^2 \right)^{\frac{1}{2}} \\
\text{moyenne harmonique} & : & \overline{X}_{q=-1} & = \dfrac{n}{\sum_{i=1}^{n} \frac{1}{x_i}} \\
\text{moyenne géométrique} & : & \overline{X}_{q \to 0} & = \left(\prod_{i=1}^{n} x_i \right)^{\frac{1}{n}} \\
\text{maximum} & : & \overline{X}_{q \to \infty} & = \max(x_i) \\
\text{minimum} & : & \overline{X}_{q \to -\infty} & = \min(x_i)
\end{array}$$

La moyenne géométrique était déjà connue et étudiée des Grecs durant l'Antiquité. La notion *géométrique* de la moyenne éponyme vient de la volonté des mathématiciens grecs à résoudre le problème suivant : trouver un carré (côté c) dont la surface est égale à celle d'un rectangle (côtés a, b) (Bullen, 2003). Dans ce cas, $c = \sqrt{ab}$ est la moyenne géométrique de a et b.

Par ailleurs, la moyenne harmonique est également un héritage grec et plus particulièrement des Pythagoriciens, qui étaient en quête des proportions « idéales », notamment en musique (Dahan-Dalmedico et Peiffer, 1986) :

$\frac{c}{a} = \frac{h}{d}$ avec $a = \frac{c+d}{2}$, la moyenne arithmétique et $h = \frac{2cd}{c+d}$, la moyenne harmonique.

Des exemples d'application de ces moyennes sont donnés dans (Wilf, 1985), où l'auteur présente notamment l'intérêt des différentes moyennes à évaluer les complexités des algorithmes.

Généralement, la technique utilisée en recherche opérationnelle pour réduire les temps de parcours consiste à minimiser la somme des temps. Par analogie, cette somme correspond à la moyenne arithmétique à un facteur $\frac{1}{n}$ près, n correspondant au nombre de véhicules utilisés (cf. état de l'art sur le DARP de Cordeau et Laporte, 2007).

Dans le cadre de cette thèse, nous effectuons des comparaisons sur l'influence des moyennes arithmétique, géométrique et harmonique sur la minimisation des temps de parcours. Les résultats des simulations sont présentés dans le chapitre 8.

6.2.3 Intégration de la norme L_p dans la minimisation des retards

Dans l'optique de réduire les retards dus aux détours nombreux et aux fenêtres de temps offrant de la souplesse et la flexibilité dans la desserte, nous utilisons la norme L_p, également connue comme distance de Minkowski utilisée notamment dans les sciences statistiques ou en géographie pour localiser les centres optimaux (problèmes de « *location/p-allocation* ») (Labbé *et al.*, 1995; Thomas, 2002).

Norme L_p ou distance de Minkowski

À la base de la norme L_p se trouve l'inégalité triangulaire de Minkowski, qui établit que dans un espace vectoriel à p dimensions :

$$\|x + y\|^p \geq \|x\|^p + \|y\|^p$$

Cette inégalité est à la base du calcul de la distance de Minkowski ou norme L_p qui tend à minimiser les distances par rapport à une valeur centrale x_c.

Ainsi la norme $L_{p,p\in[1,+\infty[}$ appliquée à la distribution de points $X = (x_0, ..., x_n)$ se définit comme suit :

$$||X_p|| = \left(\sum_{i=0}^{n} |x_i - x_c|^p \right)^{\frac{1}{p}}, p \in [1, +\infty[$$

Dans le cas de la norme $p \to \infty$, il s'agit de borner la limite de la fonction norme :

$$||X_\infty|| = \max(|x_i - x_c|)$$

Rapportée à la géographie urbaine, la distance de Minkowski permet de choisir une norme adéquate (selon p) pour mesurer les plus courtes distances entre deux points dans des villes aux configurations différentes. Si l'on considère une ville nord-américaine, dont le maillage est une structure carroyée régulière (figure 6.7), une distance entre deux points ne peut être mesurée à vol d'oiseau mais correspond à la somme des côtés des carrés longés. C'est la distance de Manhattan qui correspond à la norme L_1. La distance AB $(A(x_A, y_A), B(x_B, y_B))$ dans le plan de Manhattan correspond à :

$$AB = |x_B - x_A| + |y_B - y_A|$$

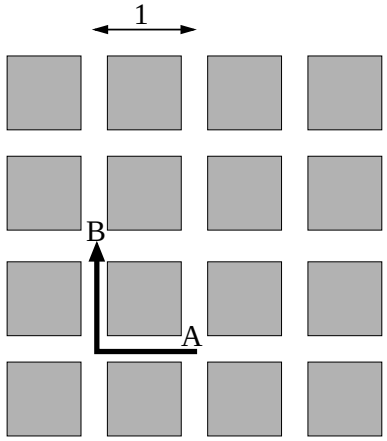


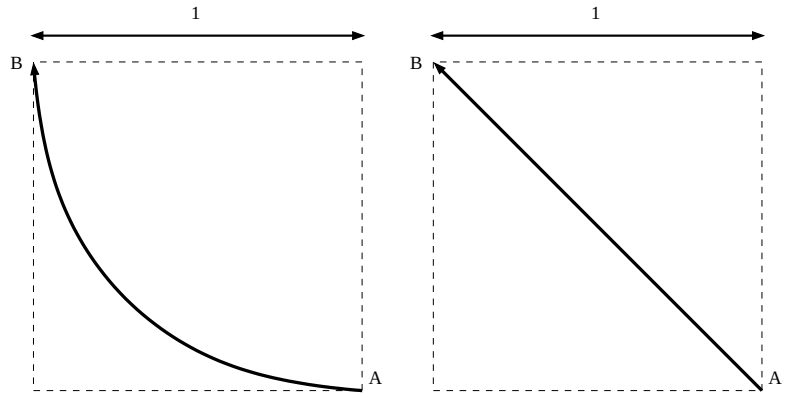
FIG. 6.7: *Norme L_1 ou distance de Manhattan : la distance AB à la norme L_1 correspond à la somme des côtés du carré, $AB_1 = 2$*

Par contre sur une surface plane sans obstacle, la plus courte distance AB est une droite correspondant à la distance euclienne (cf. fig. 6.8(b)), c'est-à-dire la norme L_2 :

$$AB_2 = \left((x_B - x_A)^2 + (y_B - y_A)^2\right)^{\frac{1}{2}}$$

Le choix de la norme a un impact direct sur la valeur centrale et trouver la norme la plus robuste a pour conséquence de « modifier » les distances. On peut par exemple envisager une norme $L_{1.5}$ (cf. fig. 6.8(a)), intermédiaire aux normes L_1 et L_2.

Les normes L_p interviennent dans la localisation des centres. La norme L_1 correspond à une approche médiane de type barycentrique, tandis que la norme L_2 corres-



(a) Norme $L_{1.5}$ (b) Norme L_2 ou distance euclidienne

FIG. 6.8: *Norme $L_{1.5}$ et norme L_2 : la distance AB à la norme $L_{1.5}$ vaut $AB_{1.5} \simeq 1.5874$, tandis que la distance AB à la norme L_2 correspond à la distance à vol d'oiseau, $AB_2 = \sqrt{2}$*

pond à une approche moyenne.

Notre analyse se limite aux normes L_1, L_2, L_∞. L'interprétation géographique et économique de ces trois normes souligne pour chacune des objectifs socio-économiques différents. La norme L_1 transcrit l'*efficacité*, la norme L_∞ traduit l'*équité* (Beguin, 1989). La norme L_2 traduit quant à elle une certaine forme d'*égalité* (Josselin, 2008).

Application au TAD

Dans notre application au TAD, nous utilisons une analogie à la norme L_p. En effet, le centre x_c n'est pas matérialisé par une valeur unique ou un centre géométrique ou géographique avec ses coordonnées dans un espace multidimensionnel. Ainsi, nous considérons l'écart au temps le plus court. Notons d le retard observé à un point de passage. Ce retard correspond à l'écart entre le temps théorique de parcours (t_{theo}) et le temps effectivement observé (t_{obs}) :

$$d = \begin{cases} t_{obs} - t_{theo} & \text{si } t_{obs} - t_{theo} > 0 \\ 0 & \text{sinon.} \end{cases}$$

Nous évaluons ici l'objectif φ_4 selon trois normes (avec n_λ le nombre de retards observés pour le véhicule λ) :

– $L_1 : \varphi_4 = \sum_{i=0}^{n_\lambda} d_i$;

- $L_2 : \varphi_4 = \sqrt{\sum_{i=0}^{n_\lambda} d_i^2}$;
- $L_\infty : \varphi_4 = max(d_i)$.

Il est intéressant de noter que les normes les plus utilisées dans les problèmes d'optimisation des transport sont les normes L_1 et L_∞ (Chevrier et Josselin, 2008). Elles proposent des visions opposées de l'objectif d'optimisation, respectivement l'efficacité et l'équité. La norme L_2, présentant une certaine forme d'égalité et correspondant géométriquement à une approche barycentrique de la position de la solution, reste moins usitée.

D'un point de vue pratique par rapport à notre recherche de l'optimum, notons que la valeur de p induit directement le poids respectif des valeurs dans la détermination de la solution. En effet, pour la norme L_1, ce sont les points proches de la valeur centrale (logique d'efficacité) qui vont peser le plus sur la solution. En norme L_∞, ce sont les points les plus éloignés (logique d'équité). La norme L_2 intermédiaire implique l'ensemble des points dans sa définition. Ces comportements ont une influence directe sur la forme des tournées, puisque les points deviennent des arrêts ou des adresses de demande de transport, et leur localisation par rapport aux tournées et aux autres requêtes va jouer sur la solution optimale, et donc, sur la répartition spatiale et l'organisation séquentielle des tournées des véhicules. En norme L_1, on favorisera la création de tournées regroupant les passagers proches du paquet de requête. En norme L_∞, on privilégiera les demandes éloignées, quitte à pénaliser les personnes proches des destinations ou les unes des autres. La norme L_2 prendra en compte toutes les demandes, mais avec des poids respectifs variables (dus au carré intervenant dans la distance euclidienne).

Conclusion

Au terme de ce chapitre, nous avons présenté les principales contributions algorithmiques de cette thèse. Ces algorithmes, au nombre de quatre, retracent l'évolution de notre pensée et du cheminement intellectuel qui nous a conduit à la méthode généralisée d'optimisation de TAD en multiconvergence.

D'abord centré sur la monoconvergence du TAD, nous développons un objet spécifique répondant à ce problème : l'arbre couvrant tentaculaire, dont les branches indiquent le nombre de véhicules requis ainsi que leurs chemins.

Cet objet donne lieu à trois algorithmes. Dans l'ordre, le premier est l'algorithme d'énumération exhaustive calculant l'ensemble exhaustif des ACT. Le deuxième algo-

rithme repousse les limites de calcul en s'intéressant uniquement à la recherche de solutions optimales avec un parcours en largeur. Suite à ces deux algorithmes exacts, nous développons un algorithme génétique qui optimise un agrégat d'objectifs et calcule des ensembles de bonnes solutions, éventuellement optimales.

Le développement de ces algorithmes conclut une première phase de recherche consacrée à la desserte d'un seul point ou zone (Chevrier *et al.*, 2008). La généralisation de la desserte à plusieurs points, la multiconvergence, donne lieu à un nouvel algorithme génétique basé sur NSGA-II pour optimiser concurremment plusieurs objectifs dans le cadre d'une approche Pareto.

L'évaluation des objectifs nous amène à nous questionner sur l'impact des différences d'optimisation sur les tournées produites. Autrement dit, si nous modifions l'évaluation d'un objectif, à quels changements dans l'exploration de l'espace des solutions allons-nous assister ? Le questionnement qui nous anime maintenant consiste à déterminer une relation entre l'optimisation et le territoire. Selon que l'espace est isotrope ou non, isomorphe ou non et que la répartition des demandes de transport est homogène ou hétérogène, nous nous demandons quel est l'impact de la moyenne généralisée de Hölder sur les temps de parcours, de la distance de Minkowski sur les retards.