Rapport - RORT

Romain Barrault et Cyril Nederveen

April 14, 2023

1 Introduction

Dans cette étude, nous nous intéressons au problème d'optimisation de taxes en certains endroit d'un réseau de transport où transitent plusieurs types de commodités. On se place du point de vue du gestionnaire du réseau et il s'agira d'adopter une stratégie de taxes qui maximise le revenu de ce dernier. Pour modéliser ce problème nous nous appuyons sur les données suivantes :

$G = (V, A)$	Graphe orienté
A_1	Ensemble des arcs sujets au péage
A_2	Ensemble des arcs non sujets au péage
c_a pour $a \in A$	Coût de l'arc $a \in A$
K	Nombre de commodités
(o_k, d_k) pour $k = 1, ..., K$	Paire origine-destination de la commodité k
n_k pour $k = 1, ..., K$	Nombre d'usagers sur la commodité k

On suppose que pour chaque commodité k, le trajet effectué correspond au plus court chemin entre o_k et d_k où le coût dépend de l'arc emprunté : c_a pour $a \in A_2$ et $c_a + T_a$ pour $a \in A_1$ avec T_a le montant de la taxe décidé par le gestionnaire du réseau.

On comprend ainsi que le choix des taxes T_a détermine la stratégie de chaque commodité et dans un cas de figure où le gestionnaire déciderait naïvement de fixer, pour un certain arc $a \in A_1$, un montant T_a trop élevé, alors l'arc ne sera sûrement pas emprunté et aucun revenu ne sera généré sur ce dernier. Le choix optimal d'une stratégie n'a a priori rien d'évident et il convient de formuler cela sous la forme d'un problème d'optimisation.

On se retrouve donc avec les variables de décision suivantes :
Variables

$x_{a,k} \geq 0$ pour $a \in A$ et $k = 1, ..., K$	Flux d'usagers sur l'arc a pour la commodité k
$T_a \geq 0$ pour $a \in A_1$	Péage apposé aux arcs de A_1

Le problème décrit ci-dessus prend alors la forme :

$$\max_{T,x} \sum_{a \in A_1} \sum_{k \in 1,...,K} T_a x_{a,k}$$

$$\text{s.c.} \quad T_a \geq 0 \qquad \forall k \in A_1 \qquad (1)$$

$$x \in arg\min_{\overline{x} \in F} \sum_{a \in A_1} \sum_{k \in \{1,...,K\}} (c_a + T_a)\overline{x}_{a,k} + \sum_{a \in A_2} \sum_{k \in \{1,...,K\}} c_a \overline{x}_{a,k} \qquad (2)$$

Où F est l'ensemble des vecteurs de flots des commodités de leur origine à leur destination.

Ce problème est à deux niveaux et non linéaire. Nous allons mettre en place, dans la section suivante, une reformulation en PLNE de ce précédent problème.

2 Reformulation en PLNE

2.1 Passage d'un bi-niveau à mono-niveau

Dans un premier temps, nous allons procéder à une manipulation de la contrainte (2) afin de garantir l'optimalité du vecteur x sans avoir recours à un $arg\min$. Cela est possible en utilisant le théorème de dualité forte au sous-problème.

Dans la suite, on suppose que les $x_{a,k}$ sont binaires avec la valeur 1 indiquant si l'arc a est emprunté pour la commodité k. De cette manière on contraint chaque commodité à choisir un unique chemin dans le cas où plusieurs plus court chemin sont disponibles. Par ailleurs, l'ensemble F peut être décrit par des contraintes de flots :

$$\sum_{a\in\delta^+(o_k)} x_{a,k} - \sum_{a\in\delta^-(o_k)} x_{a,k} = 1 \qquad k = 1,...K \tag{3}$$

$$\sum_{a\in\delta^+(d_k)} x_{a,k} - \sum_{a\in\delta^-(d_k)} x_{a,k} = -1 \qquad k = 1,...K \tag{4}$$

$$\sum_{a\in\delta^+(v)} x_{a,k} - \sum_{a\in\delta^-(v)} x_{a,k} = 0 \qquad k = 1,...K, v \in V - \{o_k, d_k\} \tag{5}$$

Si on ajoute à ces contraintes, la fonction objectif $\sum_{a\in A_1}(c_a + T_a)n_k x_{a,k} + \sum_{a\in A_2} c_a n_k x_{a,k}$ à minimiser et que l'on considère à cette étape que les T_a sont fixés, alors nous obtenons un PLNE que nous noterons (SP).

On introduit alors les variables duales $\lambda_{k,v}$ pour chaque commodité k et sommet $v \in V$. Le sous-problème dual noté (SD) s'écrit :

$$\max_{T,x} \quad n_k(\lambda_{k,d_k} - \lambda_{k,o_k})$$

$$\text{s.c.} \quad \lambda_{k,v} - \lambda_{k,u} \le c_a + n_k T_a \qquad \forall a = (u,v) \in A_1, k = 1,...,K \tag{6}$$

$$\lambda_{k,v} - \lambda_{k,u} \le c_a \qquad \forall a = (u,v) \in A_2, k = 1,...K \tag{7}$$

Le théorème de dualité forte nous permet d'affirmer que pour tout couple (x, λ) qui soit solution réalisable à la fois du sous-problème primal (SP) et dual (SD), et si de plus elle vérifie les inégalités :

$$\sum_{a\in A_1}(c_a + T_a)n_k x_{a,k} + \sum_{a\in A_2} c_a n_k x_{a,k} \le n_k(\lambda_{k,d_k} - \lambda_{k,o_k}) \quad , \quad k = 1,...K$$

Alors (x, λ) est une solution optimale à la fois pour le primal (SP) et le dual (SD). Ce constat permet alors de garantir l'optimalité de x dans le problème initial à deux niveaux y intégrant l'ensemble des contraintes de (SP) et (SD), et aussi la contrainte de dualité forte précédente :

$$\max_{x,T} \quad \sum_{a\in A_1}\sum_{k\in 1,\dots,K} T_a n_k x_{a,k}$$

s.c.

$$T_a \geq 0 \qquad \forall k \in A_1 \tag{8}$$

$$\sum_{a\in\delta^+(o_k)} x_{a,k} - \sum_{a\in\delta^-(o_k)} x_{a,k} = 1 \qquad k = 1,\dots K \tag{9}$$

$$\sum_{a\in\delta^+(d_k)} x_{a,k} - \sum_{a\in\delta^-(d_k)} x_{a,k} = -1 \qquad k = 1,\dots K \tag{10}$$

$$\sum_{a\in\delta^+(v)} x_{a,k} - \sum_{a\in\delta^-(v)} x_{a,k} = 0 \qquad k = 1,\dots K, v \in V \setminus \{o_k, d_k\} \tag{11}$$

$$\lambda_{k,v} - \lambda_{k,u} \leq c_a + n_k T_a \qquad \forall a = (u,v) \in A_1, k = 1,\dots,K \tag{12}$$

$$\lambda_{k,v} - \lambda_{k,u} \leq c_a \qquad \forall a = (u,v) \in A_2, k = 1,\dots,K \tag{13}$$

$$\sum_{a\in A_1} (c_a + T_a) n_k x_{a,k} + \sum_{a\in A_2} c_a n_k x_{a,k} \leq n_k(\lambda_{k,d_k} - \lambda_{k,o_k}) \quad k = 1,\dots,K \tag{14}$$

$$x_{a,k} \in \{0,1\}, \lambda_{k,v} \geq 0 \qquad \forall a \in A, v \in V, k = 1,\dots,K \tag{15}$$

Le programme précédent présente l'avantage d'avoir un seul niveau, en revanche on remarque des termes non-linéaires dans la fonction objectif et la contrainte de forte dualité.

2.2 Linéarisation et formulation définitive

Dans cette section, nous finalisons la formulation du problème initial en un PLNE en éliminant les termes non-linéaires. Ces derniers sont les termes en $T_a x_{a,k}$ pour tout $a \in A_1$ et $k = 1,\dots,K$. Nous introduisons alors les variables positives $z_{a,k}$ pour représenter $T_a x_{a,k}$. L'objectif aura la forme suivante :

$$\sum_{a\in A_1}\sum_{k\in 1,\dots,K} z_{a,k}$$

Étant donné qu'il s'agit d'un problème de maximisation, nous nous contentons de majorer les variables $z_{a,k}$. Dans tous les cas nous avons la majoration $z_{a,k} \leq n_k T_a$ pour $a \in A_1$. De plus, lorsque $x_{a,k} = 0$ nous devons avoir également $z_{a,k} = 0$, donc nous ajoutons à la contrainte de type "big M" : $z_{a,k} \leq M x_{a,k}$ où M est une constante relativement grande (dans notre cas nous considérons $M = \max\{n_k\} \sum_{a\in A} c_a$ qui constitue effectivement une majoration de $z_{a,k}$).

Il faut ajouter une contrainte supplémentaire afin d'éviter les cas où T_a augmente indéfiniment par rapport à z_a : $n_k T_a \leq z_a + M(1 - x_{a,k})$

Ainsi nous obtenons la formulation finale du PLNE en remplaçant les termes en $T_a x_{a,k}$ par $z_{a,k}$ et en ajoutant les deux contraintes précédentes :

$$\max_{x,z,T} \quad \sum_{a\in A_1}\sum_{k\in 1,\dots,K} z_{a,k}$$

s.c.

$$T_a \geq 0 \qquad \forall k \in A_1 \tag{16}$$

$$\sum_{a\in\delta^+(o_k)} x_{a,k} - \sum_{a\in\delta^-(o_k)} x_{a,k} = 1 \qquad k = 1,\dots K \tag{17}$$

$$\sum_{a\in\delta^+(d_k)} x_{a,k} - \sum_{a\in\delta^-(d_k)} x_{a,k} = -1 \qquad k = 1,\dots K \tag{18}$$

$$\sum_{a\in\delta^+(v)} x_{a,k} - \sum_{a\in\delta^-(v)} x_{a,k} = 0 \qquad k = 1,\dots K, v \in V \setminus \{o_k, d_k\} \tag{19}$$

$$z_{a,k} \leq n_k T_a \qquad \forall a \in A_1, k = 1,\dots,K \tag{20}$$

$$z_{a,k} \leq M x_{a,k} \qquad \forall a \in A_1, k = 1,\dots,K \tag{21}$$

$$\lambda_{k,v} - \lambda_{k,u} \leq c_a + n_k T_a \qquad \forall a = (u,v) \in A_1, k = 1,\dots,K \tag{22}$$

$$\lambda_{k,v} - \lambda_{k,u} \leq c_a \qquad \forall a = (u,v) \in A_2, k = 1,\dots,K \tag{23}$$

$$\sum_{a\in A_1} (c_a n_k x_{a,k} + z_{a,k}) + \sum_{a\in A_2} c_a n_k x_{a,k} \leq n_k(\lambda_{k,d_k} - \lambda_{k,o_k}) \qquad k = 1,\dots,K \tag{24}$$

$$x_{a,k} \in \{0,1\}, \lambda_{k,v} \geq 0 \qquad \forall a \in A, v \in V, k = 1,\dots,K \tag{25}$$

Cette formulation est celle implémentée dans le fichier `PLNE_solving.jl` et on présentera ses résultats dans la dernière section de ce rapport.

3 Implémentation de la métaheuristique pour une résolution approchée

3.1 Principe général

La métaheuristique implémentée repose sur l'Optimisation par Essaims Particulaires (**PSO**). Le principe est le suivant :

- On se donne un nombre fixe de solutions initiales réalisables du problème du péage optimal dans un réseau routier. Ces solutions réalisables seront appelées **particules** tout au long de l'algorithme, et peuvent être uniquement caractérisées à l'aide des péages sur les arcs de $A1$ grâce à l'hypothèse optimiste ;
- On leur attribue arbitrairement une vitesse initiale dans l'espace des solutions admissibles. On confondra donc la valeur de l'instanciation des péages T_i avec la **position** de ces particules (où i indexe les particules) ;
- On résout un PLNE (décrit plus tard) afin d'obtenir les flux usagers minimisant dans un premier temps le coût total du flux d'usagers pour l'ensemble des commodités, et dans un second temps maximisant la somme récupérée par l'ensemble des péages (la valeur réelle de l'objectif correspondant aux valeurs des péages) ;
- On garde en mémoire la meilleure solution atteinte invividuellement par chaque particule, notée P_i ;
- On note P_g la position de la particule P_i dont la valeur de l'objectif est maximale ;
- On rentre dans une boucle avec un nombre d'itérations fixé:
  - Pour chaque particule i :

* On choisit aléatoirement $b_1 \in [0, \phi_1]$, $b2 \in [0, \phi_2]$ où ces deux paramètres sont fixés au préalable et, à l'aide d'un troisième paramètre ω, on effectue l'équation de mouvement suivante :
  $V_i = \omega V_i + b_1(P_i - T_i) + b_2(P_g - T_i)$
  $T_i = T_i + V_i$
* On résout alors le même PLNE que précédemment afin d'aboutir aux flux d'usagers correspondant aux valeurs des péages, et surtout à la valeur de l'objectif $max_{T,x} \sum_{a \in A_1} T_a X_a = curr\_obj_i$;
* Si $curr\_obj_i$ est strictement supérieur à l'objectif atteint en P_i, alors $P_i \leftarrow curr\_obj_i$

– Si à l'issue de la mise à jour des caractéristiques de toutes les particules l'une d'elle aboutit à un objectif meilleur que celui atteint en P_g, on instancie P_g à la position de cette particule.

A l'issue de cet algorithme, on renvoie la valeur maximale de l'objectif à laquelle on a abouti, la position P_g correspondante ainsi que la durée nécessaire pour atteindre cette position par l'algorithme.

3.2 Présentation du PLNE pour déduire x à partir de T

L'idée de ce PLNE a émergé du fait qu'il est bien plus intuitif de se déplacer uniquement dans l'espace des T que de le faire additionnellement dans l'espace des x. Le problème est alors, à partir d'un instance de péages fixée T, de déduire l'ensemble des flux d'usagers sur l'ensemble du réseau, afin dans un premier temps de minimiser le coût total payé par l'ensemble des usagers sur le réseau, défini ainsi :

$$min_x \sum_{k \in K} (\sum_{a \in A_1} (c_a + T_a) x_{ak} + \sum_{a \in A_2} c_a x_{ak})$$

Dans un second temps, il faut instancier x de manière à maximiser l'objectif "réel" du problème, à savoir :

$$max_x \sum_{a \in A_1} T_a \sum_{k \in K} x_{ak}$$

(on rappelle que dans ce sous problème, T est une constante du problème et non pas une variable).
Le sous-problème peut alors être défini ainsi :

Variables : $\{(x_{ak})_{a \in A, k \in K}$: flux (entier) d'usager sur l'arc a pour la commodité k.

Contraintes :
$$\begin{cases} \forall a \in A, k \in K, D_k \geq x_{ak} \geq 0; \\ \sum_{a \in \delta^+(o_k)} x_{ak} - \sum_{a \in \delta^-(o_k)} x_{ak} = D_k; \\ \sum_{a \in \delta^+(d_k)} x_{ak} - \sum_{a \in \delta^-(d_k)} x_{ak} = -D_k; \\ \forall k \in K, v \in Vn\{o_k, d_k\}, \sum_{a \in \delta^+(o_k)} x_{ak} - \sum_{a \in \delta^-(o_k)} x_{ak} = 0; \end{cases}$$

Enfin, pour l'objectif, plutôt que d'effectuer du multiobjectif lexicographique, on introduit deux membres dans l'unique objectif (chacun correspondant aux deux objectifs mentionnés précédemment), en ajoutant une pondération à l'objectif le plus important, à savoir $\sum_{k \in K} (\sum_{a \in A_1} (c_a + T_a) x_{ak} + \sum_{a \in A_2} c_a x_{ak})$. Cette pondération, à l'aide d'une constante M suffisamment grande, doit être telle que la plus grande variation de l'autre objectif $\sum_{a \in A_1} T_a \sum_{k \in K} x_{ak}$ ne puisse pas compenser la plus petite variation (pondérée) du premier objectif. On peut aisément déterminer une constante M suffisante en remarquant que :

$\forall a \in A, k \in K, 0 \leq x_{ak} \leq D_k$ d'où $0 \leq \sum_{a \in A_1} T_a \sum_{k \in K} x_{ak} \leq (\sum_{a \in A_1} T_a)(\sum_{k \in K} D_k)$

Les T_a étant fixés, ce dernier membre est bien une constante, et si les T_a étaient entiers, on pourrait immédiatement poser M comme étant égal à cette constante. Or on a fait plutôt le choix de permettre des T fractionnaires, afin que le mouvement dans l'espace des solutions paraisse plus naturel qu'avec des T entiers uniquement ; cela peut donc induire des toutes petites variations du premier objectif. On a donc décidé de permettre au maximum 3 décimales pour chaque valeur de T_a ; la variation du premier objectif est donc minorée par 0.001, et donc celle de l'objectif pondéré par $0.001M$. On a donc posé

$$M = 1000(\sum_{a \in A_1} T_a)(\sum_{k \in K} D_k)$$

, d'où l'objectif du sous-problème :

Objectif : $\{min_x M \sum_{k \in K}(\sum_{a \in A_1}(c_a + T_a)x_{ak} + \sum_{a \in A_2} c_a x_{ak}) - \sum_{a \in A_1} T_a \sum_{k \in K} x_{ak}$.

Nous sommes alors certains que le sous-problème minimisera le coût total payé sur le réseau, et uniquement dans un second temps, il maximisera l'argent ramassé par l'ensemble des péages. Il faudra alors récupérer le second membre de cet objectif pour caractériser la qualité de la position d'une particule.

3.3 Instanciations initiales des particules

Pour commencer l'algorithme, il faut attribuer à nos particules une position initiale (*i.e.*, un ensemble admissible de valeurs de péages T_a) ainsi qu'une vitesse initiale dans l'espace des solutions.

3.3.1 Instanciation de la position

On récupère dans un premier temps deux paramètres de l'instance : $M2 = max_{a \in A_2} c_a$ et $m1 = min_{a \in A_1} c_a$. Le principe est le suivant : de manière générale, on peut chercher à "augmenter" artificiellement (par le biais des péages) le coût d'emprunt d'une route de A_1 de façon à ne pas dépasser le coût d'une route de A_2, afin d'augmenter les bénéfices du péage sans pour autant rendre les routes de A_2 plus intéressantes à emprunter. Ce principe nous pousse à introduire le paramètre $\lambda = \frac{M2-m1}{2}$, décrivant un ordre de grandeur du péage que l'on peut introduire sur toute route de A_1.
Par la suite, les péages sur A_1 seront alors générés aléatoirement entre 0 et λ.

3.3.2 Instanciation de la vitesse

On a fait le choix de se restreindre à deux types de vecteurs vitesses initiales, l'un vers "l'intérieur", l'autre vers "l'extérieur". La vitesse initiale de toute particule sera alors **proportionnelle** à son vecteur position initiale, avec un coefficient de proportionnalité choisi aléatoirement entre $-\frac{1}{2}$ et $\frac{1}{2}$. Autrement dit, lors de l'initialisation des particules, on a :
$V_i = \pm\frac{1}{2}T_i$ Cette formulation a l'avantage d'être relativement simple à concevoir et implémenter, et le facteur hautement aléatoire des positions initiales des particules laisse présager une grande diversité de vecteurs vitesses initiales (quand bien même ceux-ci ne sont pas décorrélés des positions des particules).

Important : Lors de la définition de ces caractéristiques initiales, ainsi qu'à toute application des équations du mouvement des particules, on s'assure, conformément à la règle établie précédemment, que le nombre de décimales de tout T_a quelle que soit la particule considérée soit inférieur à 3.

3.4 Choix des paramètres de la PSO

Il reste enfin à choisir des valeurs pour les paramètres de la PSO :

- Le nombre de particules n_{part} ;
- Le nombre d'itérations n_{it} ;
- Le paramètre ω (appelé **inertie**), décrivant une forme de continuité du mouvement des particules ;
- Les paramètres ϕ_1 et ϕ_2, décrivant respectivement :
  - Le facteur maximum d'attirance vers la meilleure solution déterminée par l'essaim ;
  - Le facteur maximum d'attirance vers la meilleure solution déterminée par la particule.

Nous avons émis plusieurs remarques sur ces paramètres et leurs valeurs, a priori et a posteriori :

- Le choix d'un nombre d'itérations à la place d'un temps de calcul maximum permet de placer sur un pied d'égalité les petites et grandes instances, puisque ces dernières nécessiteront inévitablement plus de temps à chaque itération de mouvement de l'essaim ;
- Poser $\omega < 1$ conduit très souvent à une immobilisation généralisée de l'essaim, ce qui n'est pas ce que nous souhaitons afin de s'extirper des extrema locaux ;
- Plus on accroît le nombre de particules, plus on affine la recherche d'optimum et plus on augmente la diversité des directions explorées.

Également dans le but de garder un temps de calcul maximum raisonnable, et après comparaisons avec des instanciations semblables, voilà les valeurs de ces paramètres que nous avons retenues pour l'ensemble des tests:

- $n_{part} = 50$;
- $n_{it} = 300$;
- $\omega = 1$;
- $\phi_1 = \phi_2 = 0.5$.

4 Résultats sur les instances

Nous présentons dans cette section les résultats obtenus grâce à nos implémentations.

4.1 Résultats de la reformulation du problème linéaire

	PLNE	
Graphe	optimum trouvé	tps CPU s.
grille_2x3	10	0,09
grille_3x4	9	0,02
grille_4x5	18	0,16
grille_5x6	38	0,34
grille_6x7	34	0,77
grille_6x8	37	0,97
grille_6x9	39	1,29
grille_6x10	39	1,18
grille_6x11	86	3,31
grille_7x8	42	0,92
grille_7x9	58	1,98
grille_7x10	77	0,9
grille_7x11	105	2,23
grille_8x9	30	4,62
grille_8x10	35	0,93
grille_8x11	79	87,53
grille_8x12	99	37,5
grille_8x13	64	4,63
grille_8x14	50	12,42
grille_9x10	67	3,88
grille_9x11	79	7,41
grille_9x12	140	9,76
grille_9x13	176	381,39
grille_10x10	97	40,96
grille_10x9	109	43,84
grille_10x8	69	1,54
grille_10x7	115	8,01
grille_10x6	90	4,2
grille_10x5	136	1,52
grille_10x4	469	0,3
grille_10x3	561	0,13

Figure 1: Résultats sur les instances pour la reformulation du PLNE

On observe donc que la reformulation du problème permet de résoudre l'ensemble des instances en un temps extrêmement restreint (moins d'une minute pour la plupart d'entre elles, pas plus de 7 pour toutes). Ses performances sont donc très satisfaisantes. Par ailleurs, vous trouverez dans le fichier `main.xlsx` une feuille comportant l'ensemble des solutions de ces instances.

4.2 Résultats de la métaheuristique

	Obj final	% amélioration	Temps de convergence(s)	Gap (%)	Obj réel
taxe_grille_2x3	9,952	76	56	0,48	10
taxe_grille_3x4	8,984	5	70	0,17777778	9
taxe_grille_4x5	17,947	10	97	0,29444444	18
taxe_grille_5x6	33,214	49	234	12,5947368	38
taxe_grille_6x7	30,33	90	341	10,7941176	34
taxe_grille_6x8	34,682	101	438	15,4097561	41
taxe_grille_6x9	30,29	56	461	22,3333333	39
taxe_grille_6x10	37,772	56	411	3,14871795	39
taxe_grille_6x11	73,93	110	539	14,0348837	86
taxe_grille_7x8	36,363	50	443	13,4214286	42
taxe_grille_7x9	43,2	67	407	25,5172414	58
taxe_grille_7x10	67,65	62	578	12,1428571	77
taxe_grille_7x11	75,24	60	270	28,3428571	105
taxe_grille_8x9	23,035	51	296	23,2166667	30
taxe_grille_8x10	24,75	57	768	29,2857143	35
taxe_grille_8x11	61,61	64	339	22,0126582	79
taxe_grille_8x12	75,188	67	900	24,0525253	99
taxe_grille_8x13	41,79	68	906	34,703125	64
taxe_grille_8x14	34,13	83	1042	31,74	50
taxe_grille_9x10	51,5	36	818	23,1343284	67
taxe_grille_9x11	59,7	60	564	24,4303797	79
taxe_grille_9x12	100,651	76	769	28,1064286	140
taxe_grille_9x13	136,147	65	723	22,64375	176
taxe_grille_10x3	527,752	193	260	5,92655971	561
taxe_grille_10x4	342,806	91	265	26,9070362	469
taxe_grille_10x5	110,223	131	313	18,9536765	136
taxe_grille_10x6	71,33	76	750	20,7444444	90
taxe_grille_10x7	86,304	83	424	24,9530435	115
taxe_grille_10x8	42,14	61	787	38,9275362	69
taxe_grille_10x9	66,074	67	945	39,3816514	109
taxe_grille_10x10	71,28	60	731	26,5154639	97

Figure 2: Résultats sur les instances pour la métaheuristique de PSO

On peut émettre plusieurs remarques :

- Cette démarche est très chronophage : la métaheuristique doit résoudre un PLNE à chaque itération du mouvement d'une particule.
- Le pourcentage d'amélioration, se référant à la meilleure valeur de l'objectif atteinte par l'essaim lors de l'initialisation, témoigne de l'impact des itérations sur cette dernière.
- Pour la plupart des instances, caractérisées par leur grande taille, la métaheuristique peine à se rapprocher de l'optimal (gap entre 20 et 30% en moyenne) ; cela peut tenir du fait que le nombre d'itérations est constant, alors qu'il peut être judicieux de l'accroître avec la taille du domaine réalisable (ce qui augmenterait d'autant plus le temps de calcul...).

5 Conclusion

L'efficacité de la reformulation du programme linéaire est très apparente, et le recours à une métaheuristique qui peine à résoudre des instances non petites permet de la mettre en évidence. En guise d'ouverture, on pourrait s'intéresser à une nouvelle méthode pour déterminer l'objectif correspondant aux valeurs des péages à travers de la programmation dynamique, au lieu de résoudre un programme linéaire.