le cnam

MPRO

Recherche Opérationnelle dans les Réseaux de Transports

rort

Optimisation du placement de fonctions de service dans les réseaux télécoms

---

Antoine Desjardins
Victor Spitzer
groupe 2

Table des matières

1 Modélisation en PL binaire

On remarque un **objectif**, à savoir limiter les coûts d'installation et d'ouverture, ainsi que 5 **contraintes** :

— Contrainte de **latence**
— contrainte de **multi-flot**
— contrainte de **capacité** sur les fonctions
— contrainte de **capacité** sur les sommets
— contrainte de **passage** par toutes les fonctions nécessaires pour les données de chaque commodité
— contrainte sur l'**ordre** de visite des fonctions pour les données de chaque commodité

Constantes Précisons ici les constantes utilisées :

— On travaille sur un graphe $G = (V, E)$
— On dispose d'une liste de fonctions F, et chaque fonction $f \in F$ a une capacité maximale α_f.
— Chaque sommet $i \in V$ a la capacité d'accueillir β_i fonctions
— On note $f_{kt} \in F_k \subset F$ la t-ième fonction à appliquer à la commodité $\mathcal{C}_k$. On pose également $n_k = |F_k|$ le nombre de fonctions f_{kt} à appliquer.
— $\mathcal{C}_k = (b_k, s_k, p_k, F_k, \mathcal{E}_k)$ représente la k-ième commodité, de quantité de données $b_k \in \mathbb{N}$, de source et puit s_k et p_k et de fonctions à appliquer sur les données dans l'ordre indiquées par la liste ordonnée $F_k = \{f_{k,0}, ..., f_{k,n_k-1}\} \subset F$, avec des contraintes d'exclusion $(v, w) \in \mathcal{E}_k \subset F_k^2$ qui impose que les deux fonctions ne peuvent pas traiter la commodité sur le même sommet. On note K le nombre total de commodités.
— $L_k \in \mathbb{N}$ représente la latence maximales autorisée pour la commodité k.
— $l_{ij} \in \mathbb{N}$ désigne la latence associée à l'arc $(i, j) \in E$.
— c_{fu} désigne le coût d'installation d'une fonction $f \in F$ au noeud $u \in V$
— c_0 désigne le coût d'ouverture d'un noeud, il est fixé à 1.

Variables Soit une commodité $\mathcal{C}_k$ qui doit être traitée dans l'ordre par les fonctions $(f_{kt})_{t\in[|0;n_k-1|]}$, on pose les variables suivantes :

— $x_i \in \{0, 1\}$: égale à 1 si au moins une fonction est installée en $i \in V$.
— $x_{fi} \in \mathbb{N}$: combien de fois la fonction $f \in F$ est installée sur le sommet i.
— $x_{ikf_{k,t}} \in \{0, 1\}$: égale à 1 si la fonction $f_{k,t}$ traite la commodité $\mathcal{C}_k$ au sommet i, et 0 sinon.
— $e_{ijkf_{k,t}} \in \{0, 1\}$: égale à 1 si l'arc (i, j) est parcouru entre le sommet où $f_{k,(t-1)}$ traite $\mathcal{C}_k$ (ou alors la source si $t = 0$) et celui où $f_{k,t}$ traite $\mathcal{C}_k$ (ou le puit si $t = n_k$), et 0 sinon.

L'objectif du problème est obtenu en sommant le coût d'installation des fonctions et en ajoutant un coût d'ouverture c_0 par sommet avec une fonction installée. On formule donc l'objectif du problème ainsi :

$$\min_{x,\,e} \sum_{i\in V} [x_i.c_0 + \sum_{f\in F} x_{fi}.c_{fi}]$$

Les variables x doivent également respecter certaines contraintes :

— Contrainte d'ouverture d'un sommet : $x_i \geq x_{ikf}, \quad \forall i \in V, \forall k \in \mathcal{C}_k, \forall f \in F$
— Contrainte de capacité sur les fonctions : $x_{fi}.\alpha_f \geq \sum_k x_{ikf}.b_k, \quad \forall i \in V, \forall f \in F$
— Contrainte de capacité sur les sommets : $\sum_{f\in F} x_{fi} \leq \beta_i, \quad \forall i$
— Contrainte d'exclusion : $x_{ikw} + x_{ikv} \leq 1, \quad \forall k \in \mathcal{C}_k, \forall i \in V, \forall (v, w) \in \mathcal{E}_k$

On fait l'hypothèse sur F_k que chacune des fonctions à appliquer est de capacité suffisante pour supporter la quantité de données b_k, et celles-ci sont toutes différentes deux à deux. On cherche la formulation PLNE pour un chemin admissible de s_k à p_k passant par des sommets traitant dans l'ordre les données. Cela peut être représenté comme une suite de problème de flot de valeur 1 avec :

— Un premier problème de flot de la source vers un sommet qui puisse appliquer la première fonction :

$$\sum_{j\in V} (e_{s_kjkf_{k,0}} - e_{js_kkf_{k,0}}) - x_{s_kkf_{k,0}} = 1, \quad \forall k \in \mathcal{C}_k$$

$$\sum_{j\in V} (e_{ijkf_{k,0}} - e_{jikf_{k,0}}) + x_{ikf_{k,0}} = 0, \quad \forall k \in \mathcal{C}_k, \forall i \neq s_k$$

— Un problème de flot du sommet qui applique la précédente fonction vers celui qui puisse appliquer la suivante :

$$\sum_{j\in V}(e_{ijkf_{k,t}} - e_{jikf_{k,t}}) - x_{ikf_{k,(t-1)}} + x_{ikf_{k,t}} = 0, \quad \forall k \in \mathcal{C}_k, \forall i, \forall t < |F_k|$$

— Un dernier problème de flot du sommet qui applique la dernière fonction vers le puit :

$$\sum_{j\in V} e_{jp_kkf_{k,n_k}} + x_{p_kkf_{k,n_k-1}} = 1, \quad \forall k \in \mathcal{C}_k$$
$$\sum_{j\in V}(e_{ijkf_{k,n_k}} - e_{jikf_{k,n_k}}) - x_{ikf_{k,n_k-1}} = 0, \quad \forall k \in \mathcal{C}_k, \forall i \neq p_k, i \in V$$

On obtient le chemin recherché en concaténant les solutions de ces problèmes de flot. Pour avoir la garantie que cette concaténation forme un chemin connexe, il est nécessaire d'imposer pour tout t qu'un seul sommet accueille la fonction f_{kt}. On pose donc la contrainte :

$$\sum_{i\in V} x_{ikf} = 1, \quad \forall i \in V, \forall k \in \mathcal{C}_k, \forall f \in F_k$$

Il faut également vérifier que ce chemin respecte la contrainte de coût sur les arcs :

$$\sum_{f\in F_k\cup\{f_l\}}\sum_{(i,j)\in A} e_{ijkf}.l_{ij} \leq L_k, \quad \forall k \in \mathcal{C}_k$$

On propose ainsi la formulation PLNE suivante, de complexité $\mathcal{O}(|C|.[F|.|A|)$:

$$\min_{x,\,e}\sum_{i\in V}[x_i.c_0 + \sum_{f\in F} x_{fi}.c_{fi}] \tag{1}$$
$$x_i \geq x_{ikf}, \quad \forall i \in V, \forall k \in \mathcal{C}_k, \forall f \in F \tag{2}$$
$$x_{fi}.\alpha_f \geq \sum_k x_{ikf}.b_k, \quad \forall i \in V, \forall f \in F \tag{3}$$
$$\sum_{f\in F} x_{fi} \leq \beta_i, \quad \forall i \in V \tag{4}$$
$$x_{ikw} + x_{ikv} \leq 1, \quad \forall k \in \mathcal{C}_k, \forall i \in V, \forall (v,w) \in \mathcal{E}_k \tag{5}$$
$$\sum_{j\in V}(e_{s_kjkf_{k,0}} - e_{js_kkf_{k,0}}) - x_{s_kkf_{k,0}} = 1, \quad \forall k \in \mathcal{C}_k \tag{6}$$
$$\sum_{j\in V}(e_{ijkf_{k,0}} - e_{jikf_{k,0}}) + x_{ikf_{k,0}} = 0, \quad \forall k \in \mathcal{C}_k, \forall i \neq s_k \tag{7}$$
$$\sum_{j\in V}(e_{ijkf_{k,t}} - e_{jikf_{k,t}}) - x_{ikf_{k,(t-1)}} + x_{ikf_{k,t}} = 0, \quad \forall k \in \mathcal{C}_k, \forall i, \forall t < |F_k| \tag{8}$$
$$\sum_{j\in V} e_{jp_kkf_{k,n_k}} + x_{p_kkf_{k,n_k-1}} = 1, \quad \forall k \in \mathcal{C}_k \tag{9}$$
$$\sum_{j\in V}(e_{ijkf_{k,n_k}} - e_{jikf_{k,n_k}}) - x_{ikf_{k,n_k-1}} = 0, \quad \forall k \in \mathcal{C}_k, \forall i \neq p_k, i \in V \tag{10}$$
$$\sum_{i\in V} x_{ikf} = 1, \quad \forall i \in V, \forall k \in \mathcal{C}_k, \forall f \in F_k \tag{11}$$
$$\sum_{f\in F_k\cup\{f_l\}}\sum_{(i,j)\in A} e_{ijkf}.l_{ij} \leq L_k, \quad \forall k \in \mathcal{C}_k \tag{12}$$
$$x_i \in \{0,1\} \quad \forall i \in V \tag{13}$$
$$x_{fi} \in \mathbb{N} \quad \forall f \in F, \forall i \in V \tag{14}$$
$$x_{ikf} \in \{0,1\} \quad \forall i \in V, \forall k \in \mathcal{C}_k, \forall f \in F \tag{15}$$
$$e_{ijkf} \in \{0,1\} \quad \forall i,j \in E, \forall k \in \mathcal{C}_k, \forall f \in F \tag{16}$$

2 Travail complémentaire

2.1 Décomposition de Dantzig-Wolfe

Le problème étudié est caractérisé pour chaque commodité par la résolution indépendante de problèmes de flot successifs. Cette résolution, pour la commodité k, concerne les variables x_{ikt} et e_{ijkt}. Elle est représentée par les contraintes (5 - 12, 15 - 16).

Les variables x_i et x_{fi} sont des variables dites "de sommets" qui regroupent l'information des différents sous-problèmes pour déterminer les sommets avec un coût d'ouverture et les fonctions à placer sur chacun d'eux. Cela est représenté par les contraintes (2 - 4, 13 - 14).

On se retrouve donc dans une situation où on doit résoudre plusieurs sous-problèmes de manière indépendante pour résoudre le problème principal. On note $Y^0 = (y_0)$ ainsi que $Y^k = (y_k)$ les ensembles de solutions du sous-problème sommet et des sous-problèmes de flot sur chaque commodité $k \in \mathcal{C}_k$ et on pose $Y = Y^0 \times \Pi_{k \in \mathcal{C}_k} Y^k$. Une formulation équivalente du problème est alors :

$$\min_{\lambda} c(\sum_{\chi_i \in Y} \lambda_i \chi_i) \tag{17}$$

$$\text{s.c.} \sum_{\chi_i \in Y} \lambda_i = 1, \tag{18}$$

$$\sum_{\chi_i \in Y} \lambda_i x^i_j \geq \sum_{\chi_i \in Y} \lambda_i x^i_{jkf}, \quad \forall j \in V, \forall k \in \mathcal{C}_k, \forall f \in F \tag{19}$$

$$\sum_{\chi_i \in Y} \lambda_i (x^i_{fj}.\alpha_f) \geq \sum_{\chi_i \in Y} \lambda_i \left(\sum_k x^i_{jkf}.b_k \right), \quad \forall j \in V, \forall f \tag{20}$$

$$\lambda_i \geq 0, \quad \forall \chi_i \in Y \tag{21}$$

Soit A la matrice des contraintes (19) et (20), $\mu^2 \geq 0$ et $\mu^3 \geq 0$ les variables duales de ces contraintes et η celle de la contrainte (20). Le Lagrangien du problème est alors de la forme :

$$\begin{aligned}
\mathcal{L}(\lambda, \mu, \eta) &= c(\sum_{\chi_i \in Y} \lambda_i \chi_i) - \mu(\sum_{\chi_i \in Y} \lambda_i (A\chi_i)) - \eta(\sum_{\chi_i \in Y} \lambda_i - 1) \\
&= \sum_{\chi_i \in Y} \lambda_i (c\chi_i - \mu A \chi_i - \eta) + \eta \\
&= \sum_{\chi_i \in Y} \lambda_i (cx_i - \sum_k \mu A^k y^k_i - \eta) + \eta
\end{aligned}$$

Dans le cas d'une décomposition de colonne, pour des variables duales fixées, le problème de Pricing sera :

$$\begin{aligned}
\min_i cx_i - \sum_k \mu A^k y^k_i - \eta &= \sum_{k=0}^{n_k} \left(\min_{y^k \in Y^k} c^k y^k - \mu A^k y^k \right) - \eta \\
&= \min_{y^0 \in Y^0} (cy^0 - \mu A^0 y^0) - \sum_{k=1}^{n_k} \left(\min_{y^k \in Y^k} \mu A^k y^k \right) - \eta
\end{aligned}$$

La matrice A représente les contraintes (18 - 19), soit :

$$\begin{aligned}
&x_i \geq x_{ikf}, \quad \forall i \in V, \forall k \in \mathcal{C}_k, \forall f \in F \\
&x_{fi}.\alpha_f \geq \sum_k x_{ikf}.b_k, \quad \forall i \in V, \forall f
\end{aligned}$$

Pour $k \geq 1$, la matrice A^k représente ces contraintes sur les variables x_{ikt} et e_{ijkt} pour tout i, j, t. Soit μ^2 et μ^3 les variables duales respectives de ces deux contraintes. Alors :

$$\mu A^k = \left(-(b_k.\mu^3_{i,f})_{i \in V,\, f \in F} - (\mu^2_{i,k,f})_{i \in V,\, f \in F}, \quad 0, \quad ..., \quad 0 \right)$$

Pour $k = 0$, la matrice A^0 représente les contraintes (2 - 3) sur les variables x_i et x_{fi}. On a ainsi :

$$\mu A^0 = \left((\textstyle\sum_{f,\,k} \mu^2_{ikf})_{i \in V}, \quad (\mu^3_{if}.\alpha_f)_{i \in V,\, f \in F} \right)$$

Ainsi, le sous-problème de la variable de sommet est :

$$\begin{aligned}
\min_{x} \quad & \sum_{i \in V} [x_i.(c_0 - \sum_{f,\,k} \mu^2_{ikf}) + \sum_{f \in F} x_{fi}.(c_{fi} - \alpha_f \mu^3_{if})] \\
\text{s.c} \quad & \sum_{f \in F} x_{fi} \leq \beta_i, \quad \forall i \in V \\
& x_i \in [0,1] \quad \forall i \in V \\
& x_{fi} \geq 0 \quad \forall f \in F, \forall i \in V
\end{aligned}$$

Ce sous-problème rend les valeur (x^*_i) et (x^*_{fi}) de la coupe à ajouter à la prochaine itération.

Et les sous-problèmes des variables de flot pour chaque commodité $k \in \mathcal{C}_k$ sont :

$$\begin{aligned}
\min_{x,\,e} \quad & \sum_{i,\,k,\,f} x_{if}(b_k \mu^3_{if} + \mu^2_{ikf}) \\
\text{s.c} \quad & x_{iw} + x_{iv} \leq 1, \quad \forall i \in V, \forall (v,w) \in \mathcal{E}_k \\
& \sum_{j \in V} (e_{s_k j f_{k,0}} - e_{j s_k f_{k,0}}) - x_{s_k f_{k,0}} = 1 \\
& \sum_{j \in V} (e_{ij f_{k,0}} - e_{ji f_{k,0}}) + x_{i f_{k,0}} = 0, \quad \forall i \neq s_k \\
& \sum_{j \in V} (e_{ij f_{k,t}} - e_{ji f_{k,t}}) - x_{i f_{k,(t-1)}} + x_{i f_{k,t}} = 0, \quad \forall i, \forall t < |F_k| \\
& \sum_{j \in V} e_{j p_k f_{k,n_k}} + x_{p_k f_{k,n_k - 1}} = 1 \\
& \sum_{j \in V} (e_{ij f_{k,n_k}} - e_{ji f_{k,n_k}}) - x_{i f_{k,n_k - 1}} = 0, \quad \forall i \neq p_k, i \in V \\
& \sum_{i \in V} x_{if} = 1, \quad \forall i \in V, \forall f \in F_k \\
& \sum_{f \in F_k \cup \{f_l\}} \sum_{(i,j) \in A} e_{ijf}.l_{ij} \leq L_k \\
& x_{if} \in [0,1] \quad \forall i \in V, \forall f \in F \\
& e_{ijf} \in [0,1] \quad \forall i,j \in E, \forall f \in F
\end{aligned}$$

Chacun des sous-problèmes rend les valeur (x^*_{ikf}) et (e^*_{ijkf}) de la coupe à ajouter à la prochaine itération.

Ces sous-problèmes doivent être résolus à chaque itération, avec les variables duales courantes, pour déterminer quelle variable λ_i ajouter ensuite par génération de colonne au problème principal. Si le minimum parmi les solution aux sous-problèmes est positif, alors la solution au problème principal est optimale.

3 Étude Numérique

Nous avons choisi d'implémenter notre modèle en `Julia` en utilisant `JuMP` et `CPLEX`.

3.1 Relaxation

Nous obtenons une borne inférieure du problème via la résolution du problème relaxé. Les contraintes modifiées apparaissent en rouge.

$$\min_{x,\,e} \sum_{i \in V} [x_i.c_0 + \sum_{f \in F} x_{fi}.c_{fi}]$$

$$x_i \geq x_{ikf}, \quad \forall i \in V, \forall k \in \mathcal{C}_k, \forall f \in F$$

$$x_{fi}.\alpha_f \geq \sum_k x_{ikf}.b_k, \quad \forall i \in V, \forall f \in F$$

$$\sum_{f \in F} x_{fi} \leq \beta_i, \quad \forall i \in V$$

$$x_{ikw} + x_{ikv} \leq 1, \quad \forall k \in \mathcal{C}_k, \forall i \in V, \forall (v,w) \in \mathcal{E}_k$$

$$\sum_{j \in V} (e_{s_k j k f_{k,0}} - e_{j s_k k f_{k,0}}) - x_{s_k k f_{k,0}} = 1, \quad \forall k \in \mathcal{C}_k$$

$$\sum_{j \in V} (e_{ijkf_{k,0}} - e_{jikf_{k,0}}) + x_{ikf_{k,0}} = 0, \quad \forall k \in \mathcal{C}_k, \forall i \neq s_k$$

$$\sum_{j \in V} (e_{ijkf_{k,t}} - e_{jikf_{k,t}}) - x_{ikf_{k,(t-1)}} + x_{ikf_{k,t}} = 0, \quad \forall k \in \mathcal{C}_k, \forall i, \forall t < |F_k|$$

$$\sum_{j \in V} e_{jp_k k f_{k,n_k}} + x_{p_k k f_{k,n_k - 1}} = 1, \quad \forall k \in \mathcal{C}_k$$

$$\sum_{j \in V} (e_{ijkf_{k,n_k}} - e_{jikf_{k,n_k}}) - x_{ikf_{k,n_k - 1}} = 0, \quad \forall k \in \mathcal{C}_k, \forall i \neq p_k, i \in V$$

$$\sum_{i \in V} x_{ikf} = 1, \quad \forall i \in V, \forall k \in \mathcal{C}_k, \forall f \in F_k$$

$$\sum_{f \in F_k \cup \{f_l\}} \sum_{(i,j) \in A} e_{ijkf}.l_{ij} \leq L_k, \quad \forall k \in \mathcal{C}_k$$

$$x_i \in [0,1] \quad \forall i \in V$$

$$x_{fi} \geq 0 \quad \forall f \in F, \forall i \in V$$

$$x_{ikf} \in [0,1] \quad \forall i \in V, \forall k \in \mathcal{C}_k, \forall f \in F$$

$$e_{ijkf} \in\in [0,1] \quad \forall i,j \in E, \forall k \in \mathcal{C}_k, \forall f \in F$$

3.2 Heuristique

Afin d'obtenir une borne supérieure pour le problème, nous implémentons une heuristique. Dans cette section, nous décrivons son fonctionnement.

Résolution itérée Le principe de fonctionnement de l'heuristique est de résoudre non pas le problème dans son ensemble, mais plutôt commodité par commodité. Chaque résolution se fait par PLNE binaire, tout comme la résolution exacte et naïve du problème maître. On résout tout d'abord pour la première commodité, comme si elle était seule. Puis, on fixe pour la résolution de la deuxième commodité les données du problème en conséquence de la solution optimale retenue pour la première commodité. En particulier, certains noeuds qui ont déjà été ouverts ont un coût d'ouverture nul. Leur capacité d'accueil est réduite (si l'on a implémenté 2 fonctions sur le noeud 1 de capacité 6 lors de la résolution de la commodité 1, la commodité 2 devra être résolue en considérant que le noeud 1 a une capacité de $6 - 2 = 4$). De plus, afin d'améliorer la qualité de l'heuristique, on peut retenir que certains noeuds ont de la capacité de fonction disponible (reprenons notre exemple : si la commodité 1 a laissé 30 de capacité de la fonction 3 sur le noeud 1, la commodité 2 résoudra un problème où ces 30 unités sont disponibles (gratuites) sur le noeud 1). Cette amélioration demande une grosse complexité au niveau code et nécessite de modifier en profondeur le PLNE, pour ce qui semble être un gain faible au vu de la qualité de l'heuristique de base, aussi nous ne l'avons pas réalisée. Nous obtenons ainsi une résolution itérée du problème, dont le résultat ne dépend que de l'ordre de traitement des commodités.

Atouts Cette heuristique implique de résoudre autant de PLNE que de commodités. Cependant, résoudre le problème pour une seule commodité est à priori beaucoup plus simple, d'une part par le nombre réduit de variables et de contraintes considérées, et d'autre part parce que le problème devient assez largement sous-contraint. De manière plus abstraite, on se sépare des contraintes liantes qui apparaissent

sous la forme de contraintes liées à l'instance (aux données) et donc plus aux variables. Il n'y a donc plus besoin de brancher sur ces variables (au lieu de se dire "je peux utiliser cette ressource, mais je dois me demander si d'autres en ont besoin", il suffit de se dire "je peux utiliser cette ressource").

Défauts L'heuristique ne trouve pas systématiquement de solution optimale. En effet, si le problème est très fortement contraint, il se peut qu'une commodité occupe un noeud qui était essentiel à une autre commodité qui ne peut pas se permettre de faire de détour. Cependant, ce genre de problèmes n'est pas tant un défaut de notre heuristique que de l'approche heuristique en soit : même les heuristiques qui peuvent trouver toutes les solutions (telles que le recuit simulé) mettraient dans ce genre de cas un temps extrêmement long à converger. A vrai dire, si le problème est fortement contraint, la Programmation Par Contrainte sera plus adaptée.

3.3 Implémentation de l'algorithme de Dantzig-Wolfe

On a présenté précédemment les modèles mathématiques à utiliser pour appliquer la décomposition de Dantzig-Wolfe. On initialise cet algorithme avec une première coupe valide obtenue par l'heuristique présentée précédemment.

Lors de l'implémentation de l'algorithme, il est simple de déclarer les modèles des sous-problèmes en amont : d'une itération à l'autre, seuls les coefficients de l'objectif représentés par les variables duales changeront. Ainsi on peut déclarer les modèles en amont et déclarer une nouvelle fois l'objectif à chaque itération.

En revanche, le problème maître nécessite d'être entièrement redéfini à chaque itération car toutes ses contraintes changent en acceptant une variable supplémentaire (une nouvelle colonne, donc un nouveau λ_i). Pour gagner en efficacité, il est préférable de résoudre son dual : ajouter une colonne sur le primal revient à ajouter une coupe sur le dual, soit à ajouter une seule et unique contrainte à chaque itération.

Pour une implémentation efficace de ce dual, il est nécessaire d'imposer une norme supérieure sur les variables μ. Comme les coefficients des contraintes du dual sont positifs et entiers, il vient immédiatement qu'une borne supérieur des μ^2_{ikf} et μ^3_{if} est le minimum des valeur objectif de chacune des coupes. On choisit donc comme borne supérieure la valeur objectif de la coupe avec laquelle on initialise l'algorithme, donc la valeur objectif obtenu avec l'heuristique.

Ce problème dual se formule ainsi :

$$
\begin{aligned}
&\max_{\eta,\,\mu} \eta \\
&\text{s.c. } \sum_{j\in V}[x^i_j.(c_0 - \sum_{f,\,k}\mu^2_{jkf}) + \sum_{f\in F} x^i_{fj}.(c_{fj} - \alpha_f\mu^3_{jf})] - \sum_{j,\,k,\,f} x^i_{jkf}(b_k\mu^3_{jf} + \mu^2_{jkf}) - \eta \geq 0, \quad \forall \chi_i \in Y \\
&\quad \eta \in \mathbb{R} \\
&\quad \mu^2_{jkf} \leq \sum_{j\in V}[x^0_j.c_0 + \sum_{f\in F} x^0_{fj}.c_{fj}], \quad \forall j \in V, \forall k \in \mathcal{C}_k, \forall f \in F \\
&\quad \mu^3_{jf} \leq \sum_{j\in V}[x^0_j.c_0 + \sum_{f\in F} x^0_{fj}.c_{fj}], \quad \forall j \in V, \forall f \in F \\
&\quad \mu^2_{jkf} \geq 0, \quad \forall j \in V, \forall k \in \mathcal{C}_k, \forall f \in F \\
&\quad \mu^3_{jf} \geq 0, \quad \forall j \in V, \forall f \in F
\end{aligned}
$$

4 Résultats Expérimentaux

Dans cette section, nous comparons nos résultats sur quelques instances. Une limite de 5 minutes a été appliquée pour les instances trop grosses. Les instances n'apparaissant pas sont celles trop volumineuses.

Fichier	Relâché	PLNE	Heuristique	Dantzig-Wolfe	Score PLNE	Score Relâché
pdh 1	3.42	3.78	∅	∅	26197.0	18467.72
pdh 2	2.99	3.11	∅	∅	30553.0	23176.13
pdh 3	3.1	3.03	∅	∅	27211.0	19897.19
pdh 4	3.02	3.02	0.47	5.27	28335.0	20361.00
pdh 5	3.04	2.98	∅	∅	27962.0	18273.72
pdh 6	3.05	3.14	∅	∅	21744.0	14944.87
pdh 8	3.1	2.94	∅	∅	26994.0	20506.27
pdh 9	3.16	2.97	∅	∅	23702.0	16674.40
pdh 10	3.01	3.0	∅	∅	35208.0	27132.51
di-yuan 1	2.86	3.88	∅	∅	22551.0	17313.82
di-yuan 3	2.76	2.94	1.03	12.85	22219.0	17671.93
di-yuan 6	2.6	3.06	1.07	14.24	21428.0	15959.6
di-yuan 9	2.55	2.87	1.05	8.68	25124.0	19537.63

TABLE 1 – Compte-rendu des expériences réalisées. Les durées sont exprimées en secondes. ∅ désigne l'impossibilité de résoudre l'instance concernée par la méthode en question.

On peut remarquer que le score relâché vaut environ 60% du score exact, ce n'est donc pas une très bonne borne mais tout de même utile. L'heuristique ne fonctionne pas bien sur les instances fortement contraintes ou adversariales, mais est, comme prévu, bien plus rapide à exécuter (cela se verrai probablement d'autant plus sur de plus grosses instances). La résolution du problème relâché est à peine plus rapide que celle du PLNE. Notre implémentation de Dantzig-Wolfe demande le plus fort temps de calcul, ce qui est probablement dû à la difficulté de réaliser une implémentation très optimisée de cet algorithme comlpexe. Nous restons cependant dans des durées tout à fait raisonnables au vu de la taille des instances.