INSTITUT POLYTECHNIQUE DE PARIS

ENSTA

IP PARIS

le cnam

INSTITUT POLYTECHNIQUE DE PARIS

PROJET DE ECMA

MAJOR : MASTER2 MPRO

ACADEMIC YEAR : 2021-2022

Optimisation robuste sur le problème le plus court chemin

Auteurs:
Yue ZHANG

Enseignants:
Zacharie ALÈS
Daniel PORUMBEL

Contents

1 Introduction du problème

1.1 Modélisation du problème statique

Données :

- un graphe orienté $G = (V, A)$;
- un sommet origine $s \in V$ et un sommet destination $t \in V$;
- une durée de trajet d_{ij} associée à chaque arc $ij \in A$;
- un poids p_i associé à chaque sommet $i \in V$.

Le problème statique consiste à trouver un plus court chemin de s à t dont le poids des sommets est inférieure ou égale à un entier S.

Variables : on propose deux variables binaires :
$x_{ij} = \begin{cases} 1, \text{si arc } (i,j) \text{ appartient plus court chemin} \\ 0, \text{sinon} \end{cases}$, et $y_i = \begin{cases} 1, \text{si sommet } i \text{ se trouve sur le plus court chemin} \\ 0, \text{sinon} \end{cases}$.

Objectif : L'objectif est donc minimiser le plus court chemin au sens la durée (1).

Contraintes :

- Contrainte du poids : la contrainte (2) indique la sommes des poids du sommet dans le plus court chemin ne dépasse pas la limite S.
- Contraintes du chemin de s à t : les contraintes (3)-(8) s'expriment un chemin de s à t. Pour cela, il exisiste exactement un arc sortant de s (3) et exactement un arc entrant à t (4). Les sommets s et t sont les extrémités du chemin (8). Pour tous les sommets intermédiaires (i.e. $\forall v \in V \backslash \{s,t\}$), au plus un arc peut rentrer à v (7), de même au plus un arc peut sortir de v (6). Dernièrement, la contrainte conservation (5): pour tous les sommets intermédiaires, le nombre de arcs sortant de v égale au nombre de arcs entrant à v. Evidemment, s'il exists un arc passant au sommet intermédiaire, alors ce sommet est bien-sûr dans le chemin de s à t.

Compacité : on peut dire que notre modèle mathématique est compacte. Le nombre de variables est $|V| + |A|$, le nombre de contraintes est $3(|V| - 2) + 4$.

Modèle :

$$\text{(P)} \quad \min_{x_{ij}} \quad \sum_{ij \in A} d_{ij} x_{ij} \tag{1}$$

$$\text{s.c.} \quad \sum_{i \in V} p_i y_i \leqslant S \tag{2}$$

$$\sum_{sj \in A} x_{sj} = 1 \tag{3}$$

$$\sum_{it \in A} x_{it} = 1 \tag{4}$$

$$\sum_{vi \in A} x_{vj} - \sum_{iv \in A} x_{iv} = 0 \, , \forall v \in V \backslash \{s,t\} \tag{5}$$

$$\sum_{vi \in A} x_{vj} = y_v \, , \forall v \in V \backslash \{s,t\} \tag{6}$$

$$\sum_{iv \in A} x_{iv} = y_v \, , \forall v \in V \backslash \{s,t\} \tag{7}$$

$$y_s = y_t = 1 \tag{8}$$

$$y_i \in \{0,1\} \, , i \in V \tag{9}$$

$$x_{ij} \in \{0,1\} \, , ij \in A \tag{10}$$

1.2 Modélisation du problème robuste

Le problème robuste considère les incertitudes sur les durées et sur les poids.

Données :

- un graphe orienté $G = (V, A)$;
- un sommet origine $s \in V$ et un sommet destination $t \in V$;
- une durée de trajet d^1_{ij} associée à chaque arc $ij \in A$;
- un poids p^2_i associé à chaque sommet $i \in V$.

Incertitudes :

- Incertitude sur les durées : $\mathcal{U}^1 = \left\{ \{ d^1_{ij} = d_{ij}(1 + \delta^1_{ij}) \}_{ij \in A} \text{ t.q. } \sum_{ij \in A} \delta^1_{ij} \leqslant d_1, \delta^1_{ij} \in [0, D_{ij}] \forall ij \in A \right\}$.
- Incertitude sur les poids : $\mathcal{U}^2 = \left\{ \{ p^2_i = p_i + \delta^2_i \hat{p}_i \}_{i \in V} \text{ t.q } \sum_{i \in V} \delta^2_i \leqslant d_2, \delta^2_i \in [0,2] \forall i \in V \right\}$.

Le problème robuste consiste à trouver le minimum le plus long chemin au sens durée de s à t dont le poids des sommets est inférieure ou égale à un entier S en considérant tous les sénarios.

Objectif : L'objectif est donc minimiser le plus long chemin au sens de la durée (11) en considérant tous les sénarios.

Modèle :

$$\text{(PR)} \quad \min_{x_{ij}} \max_{d^1_{ij} \in \mathcal{U}^1} \sum_{ij \in A} d^1_{ij} x_{ij} \tag{11}$$

$$\text{s.c.} \quad \sum_{i \in V} p^2_i y_i \leqslant S \ , \ p^2_i \in \mathcal{U}^2 \tag{12}$$

contraintes du chemin s à t : $(3) - (8)$

variables : $(9) - (10)$

2 Résolution par plans coupants

2.1 Reformuler l'objectif

On modifie le problème afin que la robustesse n'apparaisse plus dans l'expression de l'objectif mais dans les contraintes.

$$(\overline{\text{PR}}) \quad \min_{x_{ij}} \min_{z} z \tag{13}$$

$$\text{s.c.} \quad \sum_{ij \in A} d^1_{ij} x_{ij} \leqslant z \, , \, d^1_{ij} \in \mathcal{U}^1 \tag{14}$$

contrainte du poids : (12)

contraintes du chemin s à t : (3) − (8)

variables : (9) − (10)

2.2 Définir le problème maître

On définit le problème maître ci-dessous :

$$(\text{MP}) \quad \min_{z} z \tag{15}$$

$$\text{s.c.} \quad \sum_{ij \in A} d^1_{ij} x_{ij} \leqslant z \, , \, d^1_{ij} \in \mathcal{U}^{1*} \tag{16}$$

$$\sum_{i \in V} p^2_i y_i \leqslant S \ , \ p^2_i \in \mathcal{U}^{2*} \tag{17}$$

contraintes du chemin s à t : (3) − (8)

variables : (9) − (10)

d'où on propose trois différentes définitions possibles de $\mathcal{U}^{1*}$ et $\mathcal{U}^{2*}$:

- Il n'y a pas d'augmentation pour la durée d_{ij} de chaque arc $ij \in A$ (i.e. $\delta^1_{ij} = 0, \forall ij \in A$). De même, Il n'y a pas d'augmentation du poids pour chaque sommet (i.e. $\delta^2_i = 0 \, , \forall i \in V$).

$$\mathcal{U}^{1*} = \left\{ \{ d^1_{ij} = d_{ij}(1 + \delta^1_{ij}) = d_{ij} \}_{ij \in A} \right\}$$

$$\mathcal{U}^{2*} = \left\{ \{ p^2_i = p_i + \delta^2_i \hat{p}_i = p_i \}_{i \in V} \right\}$$

- Toutes les augmentations de durée δ^1_{ij} sont uniformément la moyenne de la limite d_1 (i.e. $\delta^1_{ij} = \min\left(\frac{d_1}{|A|}, D_{ij}\right) \, , \forall ij \in A$). De la même manière, les augmentations de poids sont le produit de $\hat{p}_i$ et

la moyenne de la limite d_2 (i.e. $\delta_i^2 = \min\left(\frac{d_2}{|V|}, 2\right), \forall i \in V$).

$$\mathcal{U}^{1*} = \left\{\{d_{ij}^1 = d_{ij}(1+\delta_{ij}^1)\}_{ij\in A} \text{ t.q. } \delta_{ij}^1 = \min\left(\frac{d_1}{|A|}, D_{ij}\right), \forall ij \in A\right\}$$

$$\mathcal{U}^{2*} = \left\{\{p_i^2 = p_i + \delta_i^2 \hat{p}_i\}_{i\in V} \text{ t.q. } \delta_i^2 = \min\left(\frac{d_2}{|V|}, 2\right), \forall i \in V\right\}$$

- On prend aléatoirement un sous-ensemble des arcs $A' \subset A$, tel que tous les arc hors de A' n'a pas d'augmentation de durée et tous les arcs dans A' ont une pourcentage d'augmentation de $\frac{d_1}{|A'|}$ (i.e. $\delta_{ij}^1 = \begin{cases} \min\left(\frac{d_1}{|A'|}, D_{ij}\right), \text{ si l'arc } ij \in A' \\ 0, \text{ sinon} \end{cases}, \forall ij \in A$). De même façon, on prend aléatoirement un sous-ensemble des sommets $V' \subset V$, tel que tous les sommets hors de V' n'a pas d'augmentation de poids et tous les sommets dans V' ont une augmentation du produit de $\hat{p}_i$ et $\frac{d_2}{|V'|}$ (i.e. $\delta_i^2 = \begin{cases} \min\left(\frac{d_2}{|V'|}, 2\right), \text{ si le sommet } i \in V' \\ 0, \text{ sinon} \end{cases}, \forall i \in V$).

$$\mathcal{U}^{1*} = \left\{\{d_{ij}^1 = d_{ij}(1+\delta_{ij}^1)\}_{ij\in A} \text{ t.q. } \delta_{ij}^1 = \begin{cases} \min\left(\frac{d_1}{|A'|}, D_{ij}\right), \text{ si l'arc } ij \in A' \\ 0, \text{ sinon} \end{cases}, \forall ij \in A\right\}$$

$$\mathcal{U}^{2*} = \left\{\{p_i^2 = p_i + \delta_i^2 \hat{p}_i\}_{i\in V} \text{ t.q. } \delta_i^2 = \begin{cases} \min\left(\frac{d_2}{|V'|}, 2\right), \text{ si le sommet } i \in V' \\ 0, \text{ sinon} \end{cases}, \forall i \in V\right\}$$

2.3 Définir les sous-problèmes

Soient (x_{ij}^*, y_i^*, z^*) est la solution du (MP).

Sous-problème lié à $\mathcal{U}^1$: on développe $\max_{d_{ij}^1 \in \mathcal{U}^1} \sum_{ij\in A} x_{ij}^* d_{ij}^1$:

$$(\text{SP}_1) \quad \max_{\delta_{ij}^1} \sum_{ij\in A} x_{ij}^* d_{ij}(1+\delta_{ij}^1) \tag{18}$$

$$\text{s.c.} \quad \sum_{ij\in A} \delta_{ij}^1 \leqslant d_1 \tag{19}$$

$$\delta_{ij}^1 \leqslant D_{ij}, \forall ij \in A \tag{20}$$

$$\delta_{ij}^1 \geqslant 0, \forall ij \in A \tag{21}$$

Sous-problème lié à $\mathcal{U}^2$: on sait que $\sum_{i\in V} y_i^* p_i^2 \leqslant S, \forall p_i^2 \in \mathcal{U}^2$ est équivalent au fait que la valeur objective de (22) est inférieure ou égale à S :

$$(\mathrm{SP}_2) \quad \max_{\delta_i^2} \sum_{i\in V} y_i^*(p_i + \delta_i^2 \hat{p}_i) \tag{22}$$

$$\text{s.c.} \quad \sum_{i\in V} \delta_i^2 \leqslant d_2 \tag{23}$$

$$\delta_i^2 \leqslant 2\ , \forall i \in V \tag{24}$$

$$\delta_i^2 \geqslant 0\ , \forall i \in V \tag{25}$$

2.4 Condition optimale

Une solution du problème maître (x_{ij}^*, y_i^*, z^*) est optimale, si $\sum_{ij\in A} d_{ij}^{1*} x_{ij}^* \leqslant z^*$ et $\sum_{i\in V} p_i^{2*} y_i^* \leqslant S$ avec $d_{ij}^{1*} = d_{ij}(1 + \delta_{ij}^{1*})$, et $p_i^{2*} = p_i + \delta_i^{2*}\hat{p}_i$ où δ_{ij}^{1*} est la solution donnée par la résolution (18) et δ_i^{2*} donnée par la résolution (22). Dans ce cas l'algorithme coupant s'arrête et la solution obtenue est optimale. Sinon, on ajoute les contraintes violées par les sénarios touvées aux ensembles de contraintes $\mathcal{U}^{1*}$ ou/et $\mathcal{U}^{2*}$ du problème maître (voir subsection 2.5).

2.5 Ajouter les coupes

Sous-problème lié à $\mathcal{U}^1$: Soit δ_{ij}^{1*} la solution du sous-problème (SP_1). Si $z^* < \sum_{ij\in A} x_{ij}^* d_{ij}(1 + \delta_{ij}^{1*})$, alors on ajoute la contrainte ci-dessous dans (MP) :

$$\sum_{ij\in A} x_{ij} d_{ij}(1 + \delta_{ij}^{1*}) \leqslant z$$

Sous-problème lié à $\mathcal{U}^2$: Soit δ_i^{2*} la solution du sous-problème (SP_2). Si $\sum_{i\in V} y_i^*(p_i + \delta_i^{2*}\hat{p}_i) > S$, alors on ajoute la contrainte ci-dessous dans (MP) :

$$\sum_{i\in V} y_i(p_i + \delta_i^{2*}\hat{p}_i) \leqslant S$$

3 Résolution par dualisation

3.1 Reforumlation de l'objectif du problème robuste

Premièrement, on reécrit l'objectif du problème robuste (PR) afin d'isoler la variable δ^1_{ij} :

$$\min_{x_{ij}} \max_{\delta^1_{ij}} \quad \sum_{ij\in A} d_{ij}(1+\delta^1_{ij})x_{ij} = \min_{x_{ij}} \sum_{ij\in A}(d_{ij}x_{ij}) + \max_{\delta^1_{ij}} \sum_{ij\in A} d_{ij}\delta^1_{ij}x_{ij} \tag{26}$$

$$\text{s.c.} \quad \sum_{ij\in A} \delta^1_{ij} \leqslant d_1 \tag{27}$$

$$\delta^1_{ij} \leqslant D_{ij}, \forall ij \in A \tag{28}$$

$$\delta^1_{ij} \geqslant 0, \forall ij \in A \tag{29}$$

contrainte du poids : (12)

contraintes du chemin s à t : (3) − (8)

variables : (9) − (10)

3.2 Problème interne lié aux variables δ^1_{ij}

Prenons le problème interne dans la reformulation précédente :

$$\begin{aligned} &\max_{\delta^1_{ij}} \sum_{ij\in A} d_{ij}\delta^1_{ij}x_{ij} \\ \text{s.c.} \quad &\sum_{ij\in A} \delta_{ij} \leqslant d_1 \\ &\delta^1_{ij} \leqslant D_{ij}, \forall ij \in A \\ &\delta^1_{ij} \geqslant 0, \forall ij \in A \end{aligned}$$

3.3 Dualisation du problème interne

Maintenant, on dualise le problème interne lié aux variables δ^1_{ij} :

$$\min_{\lambda_{ij},\,\alpha} d_1\alpha + \sum_{ij\in A} D_{ij}\lambda_{ij} \tag{30}$$

$$\text{s.c} \quad \alpha + \lambda_{ij} \geqslant d_{ij}x_{ij}, \forall ij \in A \tag{31}$$

$$\lambda_{ij} \geqslant 0, \forall ij \in A \tag{32}$$

$$\alpha \geqslant 0 \tag{33}$$

3.4 Reformulation de la contrainte robuste

La contrainte robuste (12) peut s'écrire sous la forme ci-dessous en isolant les variables δ_i^2 :

$$\sum_{i \in V} p_i y_i + \max_{\delta_i^2} \sum_{i \in V} \delta_i^2 \hat{p}_i y_i \leqslant S \tag{34}$$

$$\text{t.q} \quad \sum_{i \in V} \delta_i^2 \leqslant d_2 \tag{35}$$

$$\delta_i^2 \leqslant 2, \forall i \in V \tag{36}$$

$$\delta_i^2 \geqslant 0, \forall i \in V \tag{37}$$

3.5 Problème interne lié aux variables δ_i^2

Prenons le problème interne lié aux variables δ_i^2 dans la reformulation de la contrainte robuste précédente :

$$\begin{aligned} &\max_{\delta_i^2} \sum_{i \in V} \delta_i^2 \hat{p}_i y_i \\ \text{s.c.} \quad &\sum_{i \in V} \delta_i^2 \leqslant d_2 \\ &\delta_i^2 \leqslant 2, \forall i \in V \\ &\delta_i^2 \geqslant 0, \forall i \in V \end{aligned}$$

3.6 Dualisation du problème interne

De même, on dualise le problème interne précédente :

$$\min_{\beta,\, \omega_i} d_2 \beta + \sum_{i \in V} 2\omega_i \tag{38}$$

$$\text{s.c.} \quad \beta + \omega_i \geqslant \hat{p}_i y_i, \forall i \in V \tag{39}$$

$$\omega_i \geqslant 0, \forall i \in V \tag{40}$$

$$\beta \geqslant 0 \tag{41}$$

3.7 Le problème robuste sous forme PLNE

Par la forte dualité, on peut reexprimer le problème robuste en intégrant les deux dualisations de problèmes internes précédentes sous forme PLNE :

$$(\widetilde{\mathrm{PR}}) \quad \min_{x_{ij},\,\alpha,\,\lambda_{ij}} \sum_{ij\in A} d_{ij}x_{ij} + d_1\alpha + \sum_{ij\in A} D_{ij}\lambda_{ij} \tag{42}$$

$$\text{s.c.} \quad \alpha + \lambda_{ij} \geqslant x_{ij}d_{ij},\ \forall ij \in A \tag{43}$$

$$\sum_{i\in V} p_i y_i + d_2\beta + \sum_{i\in V} 2\omega_i \leqslant S \tag{44}$$

$$\beta + \omega_i \geqslant \hat{p}_i y_i,\ \forall i \in V \tag{45}$$

$$\lambda_{ij} \geqslant 0,\ \forall ij \in A \tag{46}$$

$$\omega_i \geqslant 0,\ \forall i \in V \tag{47}$$

$$\alpha \geqslant 0 \tag{48}$$

$$\beta \geqslant 0 \tag{49}$$

contraintes du chemin s à t : (3) − (8)

variables : (9) − (10)

4 Vérification la faisabilité

Après avoir obtenu la solution exacte (statique et robuste) et la solution heuristique, il est nécessaire de vérifier si elle est bien réalisable. Pour la solution statique, on simplement vérifie la solution est bien un chemin de s à t telle que la somme de poids des sommets ne dépasse pas à la seuille S.

Algorithm 1: VERIFYSTATICSP($vertices$, $arcs$)

Input: $vertices$: l'ensemble des sommets sélectionés; $arcs$: l'ensemble des arcs sélectionés.
Output: *true* si la solution entrée est réalisable; $false$ sinon.

if $|vertices| \neq |arcs| - 1$ **then**
  return $false$;
$predecessor \leftarrow [None,\ \forall v \in V]$;
for $(u, v) \in arcs$ **do**
  $predecessor[v] \leftarrow u$;
for $v \in vertices$ **do**
  if $v = s$ **then**
    continue;
  if $predecessor[v] = None$ **then**
    return $false$;
return $sum(p_v,\ \forall v \in vertices) \leqslant S$

Pour la solution robuste, elle devrait être tout d'abord realisable dans le context statique, c'est-à-dire que elle est bien un chemin de s à t avec le poids total ne dépasse pas à S. Et puis on surtout s'intéresse sur la robustesse de poids. Donc on considère la solution entrée comme les y_i^* et on résout le sous-problème (SP_2) (22) programmation linéaire par CPLEX et voir si la fonction objective (la maximale robustesse) dépasse à S ou non.

Algorithm 2: VERIFYROBUSTSP($vertices$, $arcs$)

Input: $vertices$: l'ensemble des sommets sélectionés; $arcs$: l'ensemble des arcs sélectionés.
Output: *true* si la solution entrée est réalisable; $false$ sinon.

if VERIFYSTATICSP($vertices$,$arcs$) $= false$ **then**
  return $false$;
$z_2 \leftarrow$ résoudre (SP_2);
return $z_2 \leqslant S$

Maintenant, on est prêt d'analyer les résultats expérimentaux de nos différentes propositions. Les tests sont effectués sur une **machine virtuelle** de système LINUX utilisant le CPU `Intel(R) Core(TM) i7-8650U CPU @ 1.90GHz` du `4 processors` avec `RAM 4 GB`. Pour chaque méthode approache, on a fixé le temps limite à **60** seconds. En particulier, étant donné que le nom des instances est long, dans les tableaux d'affichage des résultats, on identifie chaque instance par son nombre de villes et les traiter dans l'ordre de **"*.BAY.gr", "*.COL.gr" et "*.NY.gr"**. Malheureusement, pour les instances très grands, les processors ont été **"Killed"** à cause de mémoire/capacité limité, et on n'arrive pas à utiliser "try catch" pour traiter ce type d'erreur en JULIA, d'autre part on n'a pas assez de temps pour tourner touts les instances, donc on n'a que évalué les différents algorithmes sur les petits instances (le nombre de villes inférieur à 700).

5 Influence des sénarios initiaux

Pour l'algorithme plans coupants, on s'intéresse à la sensibilité sur les sénarios intitiaux. Dans subsection 2.2, on a proposé trois différents initialisations des sénarios:

- **Defaut** : pas d'augementation sur les distances d'arc et ni augmentation sur les poids de sommets.
- **Uniform** : toute la distance d'arc est augmentée uniformément la moyenne de l'augementation totale sans passer sa limite; tout le poid de sommet est aussi augmenté uniformément la moyenne de l'augementation totale sans passer sa limite.
- **Arbitrary** : on sélectionne aléatoiremment un sous-ensemble d'arcs et un sous-ensemble de sommets, et les effectuer une augmentation.

Voici dessous Table 1 le tableau des résultats expérimentaux. Ici on a choisi de tester les instances par l'algorithme plans coupants (avec les sous-problèmes exactes). Chaque colonne, on affichage le temps d'exécution, le gap entre la meilleure borne et la meilleur borne rencontrée.

On observe que pour les petits instances, les différentes initialisations des sénarios n'ont pas d'influence sur les qualités de solutions. L'algorithme plans coupants trouve la solution optimale sur tous les 3 différents sénarios initiaux. Par contre concernant le temps d'exécution, l'initialisation par défaut est légèrement le plus lente pour trouver une solution optimale. Le temps d'exécution avec l'initialisation uniforme et l'initialisation arbitaire sont assez proches. Peut-être l'influence des sénarios initiaux serait plus évident sur les grands instances.

Cities	Defaut			Uniform			Arbitrary		
	Time(s)	Gap	Best bound	Time(s)	Gap	Best bound	Time(s)	Gap	Best bound
20	1.22	0.0	15332.56	0.99	0.0	15332.56	1.0	0.0	15332.56
20	0.55	0.0	7076.52	0.59	0.0	7076.52	0.64	0.0	7076.52
20	0.9	0.0	8699.36	0.96	0.0	8699.36	1.16	0.0	8699.36
40	16.42	0.0	12664.33	14.3	0.0	12664.33	13.42	0.0	12664.33
40	60.78	0.0	14119.91	60.12	0.0	14119.91	60.09	0.0	14119.91
40	52.38	0.0	17330.1	50.96	0.0	17330.1	49.62	0.0	17330.1
60	33.12	0.0	10633.33	32.61	0.0	10633.33	34.71	0.0	10633.33
60	3.71	0.0	23914.23	3.51	0.0	23914.23	3.4	0.0	23914.23
60	13.07	0.0	21277.0	12.46	0.0	21277.0	12.26	0.0	21277.0
80	62.21	0.0	10447.73	60.42	0.0	10411.68	61.75	0.0	10411.68
80	40.04	0.0	11610.8	34.67	0.0	11610.8	35.36	0.0	11610.8
80	7.58	0.0	18619.6	7.26	0.0	18619.6	7.61	0.0	18619.6
100	61.19	0.0	8374.96	60.42	0.0	8374.96	60.99	0.0	8374.96
100	60.3	0.0	12575.96	60.27	0.0	12575.96	60.81	0.0	12575.96
100	60.99	0.0	18042.84	60.19	0.0	18373.17	60.28	0.0	18373.17
120	60.34	0.0	9761.52	61.28	0.0	9833.16	61.18	0.0	9761.52
120	61.34	0.0	12575.96	61.74	0.0	12575.96	60.62	0.0	12213.52
120	61.99	0.0	23395.45	61.89	0.0	24676.03	61.68	0.0	23395.45
140	60.88	0.0	12989.0	61.5	0.0	12989.0	60.98	0.0	12989.0
140	47.35	0.0	14380.14	47.65	0.0	14380.14	44.8	0.0	14380.14
140	62.09	0.0	30755.78	64.95	0.0	30755.78	60.92	0.0	30755.78
160	61.31	0.0	9365.32	60.44	0.0	9365.32	61.86	0.0	9365.32
160	61.13	0.0	13655.26	61.34	0.0	13655.26	60.52	0.0	13655.26
160	60.21	0.0	12331.87	61.44	0.0	12331.87	61.1	0.0	12331.87
180	62.41	0.0	9075.08	62.18	0.0	9075.08	63.45	0.0	9303.68
180	61.75	0.0	19643.42	60.23	0.0	19643.42	61.79	0.0	19643.42
180	61.62	0.0	11753.74	61.13	0.0	11753.74	60.28	0.0	11753.74
200	60.59	0.0	9075.08	60.35	0.0	9075.08	61.08	0.0	9075.08
200	62.66	0.0	19369.02	61.11	0.0	19251.76	61.5	0.0	19369.02
200	60.56	0.0	27835.61	61.39	0.0	27835.61	61.12	0.0	27635.63
250	64.03	0.0	15094.09	63.91	0.0	15094.09	62.6	0.0	15094.09
250	64.1	0.0	19187.8	62.96	0.0	19187.8	62.66	0.0	19187.8
250	60.77	0.0	34525.13	60.74	0.0	34525.13	61.06	0.0	34525.13
300	67.44	0.0	17902.5	67.33	0.0	17902.5	69.2	0.0	17955.82
300	66.92	0.0	16046.36	60.01	0.0	16046.36	62.94	0.0	16046.36
300	63.68	0.0	28943.4	62.01	0.0	28943.4	61.91	0.0	28943.4
350	67.12	0.0	16902.64	69.01	0.0	16902.64	68.94	0.0	16902.64
350	65.89	0.0	20387.11	71.01	0.0	20387.11	60.89	0.0	20387.11
350	72.52	0.0	23044.65	69.13	0.0	23044.65	68.44	0.0	23044.65
400	67.68	0.0	16619.9	67.63	0.0	16619.9	68.07	0.0	16619.9
400	66.88	0.0	20690.32	66.6	0.0	20690.32	66.15	0.0	20690.32
400	68.04	0.0	19288.16	67.76	0.0	19288.16	67.51	0.0	19288.16

Table 1: Influences d'initialisation des sénarios.

6 Influence les heuristiques pour des sous problèmes

Concernant l'algorithme plans coupants et Branch&Cut, on typiquement résout les sous-problèmes exactes par CPLEX. Une autre approache est que on peut résoudre les sous-problèmes heuristiquement, et il est intéressant de voir le temps accéleré par les heuristiques et ses performances. Vue que les sous-problèmes sont les problèmes de programmation linéaire très simple avec seulement une variable, on propose les heuristiques gloutons. L'idée est que on laisse les arcs (resp. sommets) ayant le maximale distance robuste (resp. le maximal poids robuste) augementer au maximum d'abord, jusqu'à le seuil atteint.

Algorithm 3: HEURISTICSP1(x^*)

Input: x^*: la solution courante de problème maître.
Output: δ^1: l'augmentation de charque arc; z_1 la valeur de distance robuste heuristique.

$\delta^1 \leftarrow [0.0,\ \forall ij \in A]$;
$z_1 = 0.0$;
$max\_augmentation \leftarrow [x^*_{ij} \cdot d_{ij} \cdot (1 + D_{ij}),\ \forall ij \in A]$;
$arcs \leftarrow sorted(ij,\ by = max\_augmentation[ij],\ reversed)$;
; `// trier les arcs par la distance robuste maximale augmentée décroissante`
for $ij \in arcs$ **do**
  if $sum(\delta^1) + D_{ij} \leqslant d_1$ **then**
    $z_1 \leftarrow z_1 + max\_augmentation[ij]$;
    $\delta^1[ij] \leftarrow D_{ij}$;
  else if $sum(\delta^1) < d_1$ **then**
    $\delta^1[ij] \leftarrow d_1 - sum(\delta^1)$;
    $z_1 \leftarrow z_1 + x^*_{ij} \cdot d_{ij} \cdot (1 + \delta^1[ij])$;
  else
    $z_1 \leftarrow z_1 + x^*_{ij} \cdot d_{ij}$;
return $z_1,\ \delta^1$

Donc dans le problème maître, il recupère les valeurs z_1 et z_2 en le comparant avec les conditions optimales dans subsection 2.4 où on a discuté, et décide d'ajouter les coupes ou non avec les variables heuristiques δ^1 et δ^2.

Voici dessous Table 2 le tableau de résultats expérimentaux où on compare les performances de plans coupants et branch&cut en résolvant les sous problèmes exactes ou heuristiques. Pour chaque méthode, on s'intérrese au temps d'exécution, le gap entre la meilleure borne et également la meilleure borne rencontrée. Pour **PC Heur** et **BC Heur**, on calcule également le gap entre la solution heuristique et la solution exacte par $\frac{|v^*(exact) - v^*(heur)|}{v^*(exact)}$.

A cause de difficile gérer le contrôl du temps, l'algorithme de plans coupants ne s'arrête pas exactement dans 60 seconds. Parce que avant de commencer à la nouvelle itération, si le temps courant ne dépasse pas à limite, mais la prochaine itération la somme de temps résolvant le problème maître et les deux sous-problèmes peut être dépassé à la limite. Par conséquent, l'algorithme plans coupants souvant se terminent dans 2 ou 3 minutes environ.

Algorithm 4: HEURISTICSP2(y^*)

Input: y^*: la solution courante de problème maître.
Output: δ^2: l'augmentation de charque sommet; z_2 la valeur du poids robuste heuristique.

$\delta^2 \leftarrow [0.0,\ \forall v \in V]$;
$z_2 = 0.0$;
$max\_augmentation \leftarrow [y_v^* \cdot (p_v + 2 \cdot \hat{p}_v),\ \forall v \in V]$;
$vertices \leftarrow sorted(v,\ by = max\_augmentation[v],\ reversed)$;
; `// trier les sommets par le poids robuste maximale augmentée décroissante`
for $v \in vertices$ **do**
  if $sum(\delta^2) + 2 \leqslant d_2$ **then**
    $z_2 \leftarrow z_2 + max\_augmentation[v]$;
    $\delta^2[v] \leftarrow 2$;
  else if $sum(\delta^2) < d_2$ **then**
    $\delta^2[v] \leftarrow d_2 - sum(\delta^2)$;
    $z_2 \leftarrow z_2 + y_v^* \cdot (p_v + \delta^2[v] \cdot \hat{p}_v)$;
  else
    $z_2 \leftarrow z_2 + y_v^* \cdot p_v$;
return $z_2,\ \delta^2$

Comparaison l'algorithme plans coupants exact et le branch&cut exact Si on regarde les instances résolus dans 60 seconds (les villes entre 20 et 100), on trouve que l'algorithme branch&cut est signifitivement plus rapide que l'algorithme plans coupants. Les même valeurs objectives impliquent que nos algorithmes sont bien implémentés.

On constate que pour les instances de 140 à 300, l'algorithme plans coupants arrive à trouver une solution optimale environs dans 60 seconds, cependant, l'algorithme branch&cut s'arrête au tour de 60 seconds mais avec un gap moins 10% une garantie pas male.

Pour les instances de villes entre 400 et 650, l'algorithme plans coupants réussie à trouver la solution optimale dans 2 minutes, et clairement le branch&cut a eu le time out pour la plus part de ces instances et avec un gap environ 15%.

Comparaison les sous-problèmes exactes et heuristiques Tout d'abord, on observe que les heuristiques accélèrent beaucoup environ 2 fois plus vite sur les grands instances 550 et 650 pour l'algorithme plans coupants. Et les heuristiques sur le branch&cut permettent de résoudre presque tous les instances moins de 700 villes.

Au niveau du gap entre la borne heuristique et la borne exacte, pour le branch&cut ils sont quasiment 0 gap entre la solution exacte. Pour l'algorithme plans coupants, le gap est environ moins de 5% pour les instances moins de 300 villes, pour les instances entre 400 et 650, le gap augmente à 10% et 20%.

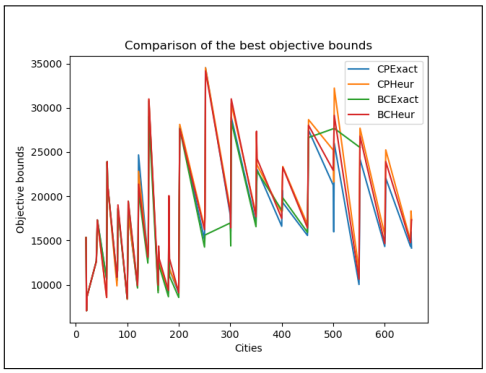
Dans Figure 1 on voit que les bornes pour les petits instances sont presque les mêmes (assze proches), mais l'écart devient large quand instances deviennent plus grands. Sur le Figure 1b, il est évident que le méthode B&C exact est le plus lent, mais le méthode B&C heuristique est le plus rapide, et les deux méthodes de plans coupants sont aussi pas males.

Cities	PC Exact			PC Heur				BC Exact			BC Heur			
	Time(s)	Gap	Best bound	Time(s)	Gap	Best bound	Gap Heur	Time(s)	Gap	Best bound	Time(s)	Gap	Best bound	Gap Heur
20	2.82	0.0	15332.56	1.98	0.0	15332.56	0.0	0.81	0.0	15332.56	0.21	0.0	15332.56	0.0
20	1.06	0.0	7076.52	1.62	0.0	7076.52	0.0	0.4	0.0	7076.52	0.11	0.0	7076.52	0.0
20	2.49	0.0	8699.36	1.12	0.0	8699.36	0.0	1.33	0.0	8699.36	0.16	0.0	8699.36	0.0
40	16.49	0.0	12664.33	13.53	0.0	12664.33	0.0	1.26	0.0	12664.33	0.3	0.0	12664.33	0.0
40	60.05	0.0	14119.91	60.49	0.0	14119.91	0.0	10.75	0.0	15058.97	1.59	0.0	15058.97	0.0
40	57.71	0.0	17330.1	60.4	0.0	17290.59	0.0	2.74	0.0	17330.1	0.62	0.0	17330.1	0.0
60	35.47	0.0	10633.33	19.29	0.0	8571.81	0.19	3.84	0.0	10633.33	0.54	0.0	8571.81	0.19
60	3.52	0.0	23914.23	3.52	0.0	23914.23	0.0	2.5	0.0	23914.23	0.72	0.0	23914.23	0.0
60	12.2	0.0	21277.0	19.84	0.0	21277.0	0.0	3.65	0.0	21277.0	0.87	0.0	21277.0	0.0
80	60.42	0.0	10483.55	62.47	0.0	9884.89	0.06	14.78	0.0	10857.06	44.31	0.0	10857.06	0.0
80	28.05	0.0	11610.8	23.24	0.0	11610.8	0.0	6.88	0.0	11610.8	44.08	0.0	11610.8	0.0
80	5.5	0.0	18619.6	18.63	0.0	18619.6	0.0	2.6	0.0	18619.6	13.15	0.0	19029.52	0.02
100	60.39	0.0	8374.96	60.86	0.0	8374.96	0.0	29.46	0.0	8462.96	4.34	0.0	8462.96	0.0
100	60.5	0.0	12575.96	59.06	0.0	12938.4	0.03	16.85	0.0	12938.4	1.99	0.0	12938.4	0.0
100	60.03	0.0	17999.39	60.51	0.0	17999.39	0.0	23.72	0.0	19446.56	3.75	0.0	19446.56	0.0
120	60.38	0.0	9761.52	60.58	0.0	9833.16	0.01	60.14	0.03	9642.0	12.94	0.0	9921.16	0.0
120	60.24	0.0	12575.96	60.1	0.0	12575.96	0.0	56.17	0.0	12938.4	5.55	0.0	12938.4	0.0
120	60.5	0.0	24676.03	63.05	0.0	22827.8	0.07	60.0	0.22	20571.54	60.0	0.19	21366.75	0.0
140	60.77	0.0	12989.0	60.76	0.0	12967.44	0.0	60.31	0.13	12462.63	60.02	0.09	13135.53	0.0
140	52.93	0.0	14380.14	26.24	0.0	14380.14	0.0	28.43	0.0	14380.14	2.19	0.0	14380.14	0.0
140	60.62	0.0	30755.78	61.96	0.0	30935.84	0.01	60.07	0.12	29003.36	60.02	0.05	31001.15	0.01
160	60.06	0.0	9365.32	60.19	0.0	9365.32	0.0	60.56	0.09	9105.9	42.14	0.0	9972.16	0.0
160	60.04	0.0	13655.26	60.86	0.0	13836.48	0.01	60.37	0.12	12634.73	14.68	0.0	14380.14	0.0
160	61.91	0.0	12331.87	60.53	0.0	12384.85	0.0	60.47	0.06	12148.77	6.75	0.0	12962.98	0.0
180	62.39	0.0	9075.08	62.62	0.0	9365.32	0.03	61.01	0.13	8657.15	60.02	0.06	9365.32	0.0
180	60.31	0.0	19643.42	60.78	0.0	19824.64	0.01	60.16	0.04	19358.17	16.96	0.0	20069.82	0.0
180	60.81	0.0	11753.74	61.79	0.0	11806.72	0.0	60.08	0.15	10990.1	34.52	0.0	12962.98	0.0
200	60.55	0.0	9075.08	61.8	0.0	9303.68	0.03	65.57	0.14	8573.31	60.12	0.09	9100.48	0.0
200	63.01	0.0	19369.02	63.16	0.0	19643.42	0.01	61.32	0.1	18091.96	33.14	0.0	20069.82	0.0
200	62.62	0.0	27835.61	60.69	0.0	28126.49	0.01	60.03	0.1	27635.63	60.1	0.1	27619.52	0.0
250	60.11	0.0	15094.09	61.42	0.0	16466.18	0.09	62.96	0.16	14271.17	60.27	0.05	16230.8	0.0
250	64.62	0.0	19187.8	60.79	0.0	19187.8	0.0	75.23	0.22	15628.45	60.22	0.12	17623.11	0.0
250	62.57	0.0	34525.13	62.2	0.0	34525.13	0.0	-	-	-	60.17	0.08	34155.8	-
300	64.86	0.0	17902.5	65.12	0.0	18575.32	0.04	82.6	0.19	16986.5	60.04	0.13	18195.21	0.0
300	63.05	0.0	16046.36	64.39	0.0	18034.72	0.12	159.16	0.22	14396.05	60.27	0.11	16441.41	0.0
300	61.82	0.0	28943.4	62.58	0.0	30479.67	0.05	127.11	0.13	28481.76	60.26	0.06	31008.54	0.0

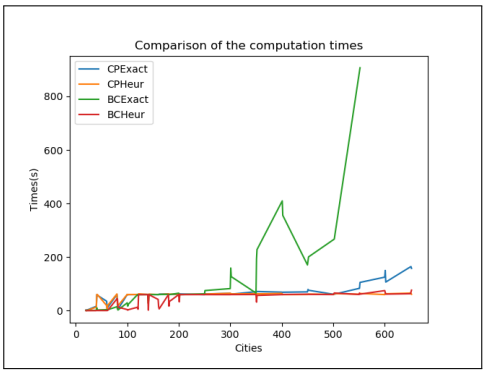
Table 2: Comparaison influences des heuristiques des sous-problèmes.

Cities	PC Exact			PC Heur				BC Exact			BC Heur			
	Time(s)	Gap	Best bound	Time(s)	Gap	Best bound	Gap Heur	Time(s)	Gap	Best bound	Time(s)	Gap	Best bound	Gap Heur
400	69.32	0.0	16619.9	64.63	0.0	18164.8	0.09	-	-	-	60.11	0.08	17500.85	-
400	68.75	0.0	20690.32	63.04	0.0	21632.04	0.05	409.83	0.45	18340.16	60.03	0.05	20925.75	0.34
400	69.05	0.0	19288.16	60.0	0.0	23369.58	0.21	355.93	0.18	19828.64	60.63	0.04	23278.33	0.0
450	70.37	0.0	15578.0	60.46	0.0	16817.0	0.08	170.79	0.16	15925.3	61.72	0.14	16424.79	0.0
450	79.41	0.0	20690.32	61.23	0.0	21632.04	0.05	-	-	-	-	-	-	-
450	76.34	0.0	27474.56	62.14	0.0	28665.62	0.04	200.78	0.16	26644.95	60.95	0.11	28029.36	0.0
500	61.53	0.0	21264.0	60.97	0.0	25222.26	0.19	-	-	-	60.36	0.23	22936.97	-
500	62.32	0.0	15990.0	60.27	0.0	21632.04	0.35	-	-	-	-	-	-	-
500	61.59	0.0	25991.0	63.93	0.0	32226.63	0.24	267.26	0.19	27675.98	66.14	0.15	29148.99	0.0
550	83.2	0.0	10036.0	61.5	0.0	11597.74	0.16	-	-	-	60.6	0.09	10643.28	-
550	83.43	0.0	15990.0	63.3	0.0	21632.04	0.35	-	-	-	66.09	0.24	18340.16	-
550	105.63	0.0	24190.0	64.41	0.0	27695.65	0.14	906.59	0.15	25518.69	61.81	0.11	26797.29	0.0
600	125.25	0.0	14332.0	60.21	0.0	15637.2	0.09	-	-	-	75.11	0.16	14658.3	-
600	150.2	0.0	15990.0	67.26	0.0	20690.32	0.29	-	-	-	67.56	0.17	18340.16	-
600	106.94	0.0	22042.0	63.31	0.0	25244.8	0.15	-	-	-	61.84	0.12	23917.74	-
650	163.81	0.0	14332.0	65.73	0.0	14984.6	0.05	-	-	-	63.44	0.16	14767.07	-
650	164.57	0.0	15990.0	64.81	0.0	18340.16	0.15	-	-	-	-	-	-	-
650	157.96	0.0	14158.0	61.38	0.0	17346.74	0.23	-	-	-	76.78	0.0	17346.74	-

Table 3: Comparaison influences des heuristiques des sous-problèmes.



(a) Comparaison la borne.



(b) Comparaison du temps d'exécution.

Figure 1: Diagrammes des performances en fonction de taille d'instance.

7 Heuristique Primal

Pour le problème le plus court chemin robuste, on propose également une approche heuristique gourmande. Vue que l'objective est de minimiser le prix robustesse d'un plus court chemin, notre idée principale est que on suppose chaque arc sa distance est augmentée au maximum, et on y recherche le plus court chemin par algorithme Dijkstra. En considérant la robustesse du poids, on également suppose que chaque sommet son poids est augmenté au maximum, afin que ce type de plus court chemin devrait respecter le seuil du poids total S, on est censé de chercher le plus court chemin au poids minimum avec chaque arc et sommet sa valeur est augmentée au maximum. Subséquemment, dans l'algorithme Dijkstra, on considère le côut est la somme de distance robuste et poid robust. A la fin de l'algorithme Dijkstra, on vas désider les variables robustes (i.e. les augmentations) qui sont résolus par les heuristiques de sous-problèmes introduites dans la section précédemment.

Algorithm 5: HEURISTICPRIMAL()

Input:

Output: δ^1: l'augmentation de charque arc; z_1 la valeur de distance robuste heuristique; δ^2: l'augmentation de charque sommet; z_2 la valeur du poids robuste heuristique.

`// initialisation les données`

$todo \leftarrow [v,\ \forall v \in V]$;
$predecessor \leftarrow [None,\ \forall v \in V]$;
$dist \leftarrow [\infty,\ \forall v \in V]$;
$dist[s] \leftarrow 0.0$;
$max\_augmentation \leftarrow [0,\ \forall uv \in A]$;

`// initialisation les coûts : la somme de distance maximale et le poids maximale`

for $(u, v) \in A$ **do**
  $poids \leftarrow p_v + \hat{p}_v \cdot 4$;
  if $v = s$ ***or*** $v = t$ **then**
    $poids \leftarrow 0$;
  $max\_augmentation[uv] \leftarrow d_{uv} \cdot (1 + D_{uv}) + poids$;

`// tant qu'il reste des sommets à traiter`

while $|todo| > 0$ **do**
  $u \leftarrow min(dist[v],\ \forall v \in V)$;
  $delete(todo,\ u)$;
  if $u = t$ **then**
    break;
  $neigbours \leftarrow [v,\ v \in todo$ **and** $v \in N(u)]$;
  for $v \in neigbours$ **do**
    $alt \leftarrow dist[u] + max\_augmentation[uv]$;
    if $alt < dist[v]$ **then**
      $dist[v] \leftarrow alt$;
      $predecessor[v] \leftarrow u$;

`// préparation la solution`

$arcs \leftarrow []$;
$vertices \leftarrow []$;
$u \leftarrow t$;
while $predecessor[u] \neq None$ **do**
  $vertices.add(u)$;
  $arcs.add((predecessor[u], u))$;
  $u \leftarrow predecessor[u]$;
$vertices.add(s)$;
$z_1, \delta^1 \leftarrow$ HEURISTICSP1(arcs);
$z_2, \delta^2 \leftarrow$ HEURISTICSP2(vertices);
return $z_1, \delta^1, z_2, \delta^2$

8 Comparaison performances des modèles

Dans Figure 2, on illustre que pour chaque algorithme, le temps d'exécution et le nombre d'instances résolus. Parmis les approaches exactes, on trouve que le dualisation a la meilleure performance et puis le branch&cut heuristiques est moins vite. Quand les instances deviennent plus grands, le branch&cut exacte et plans coupants sont beaucoup plus lentes.

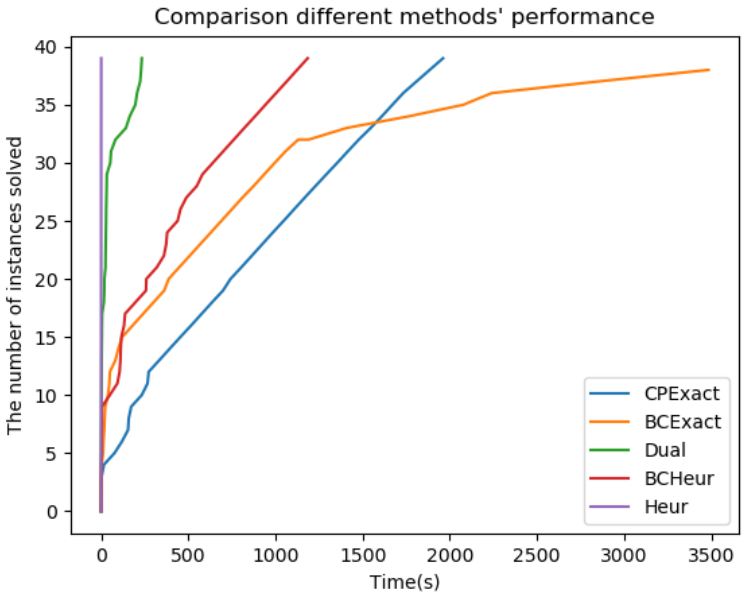


Figure 2: Comparaison les performances de différentes approaches.

Voici Table 4 le tableaux expérimentaux de différentes approaches. Pour les instances avec le nombre de villes moins de 400, le prix robuste est entre 15% et 39%. L'algorithme plans coupants et l'approache dualisation arrivent à trouver une solution optimale pour tous les petits instances. Le branch&cut avec les sous-problèmes heuristiques résout les problèmes signifitivement plus rapide que le branch&cut exacte, avec le gap moins 10% une bonne qualité de solution. Néanmoins, notre heuristique primale résout le problème tout de suite mais avec un gap un peu plus large entre la solution exacte, entre 0% et 85% la garantie de qualité de solution n'est pas stable.

9 Perspectives

En manquant du temps, pour le dernier comparaison des performances des modèles, on n'a que testé sur les petits instances, par suite il est intéressant de tester les grands instances et voir les influences et les performances de différents algorithmes.

Dans le projet, on a proposé une heuristique primale gourmande, donc il est intéressant d'y contin-

uer de réfléchir une heuristique l'arrondi de relaxation de dualisation par exemple d'étudier la garantie de performance.

Vue que on a déjà une solution heuristique primale, il est également intéressant de l'ajouter à la initialisation de CPLEX, et voir l'impact de solution initiale sur le branchement de CPLEX.

Enfin, il est aussi intéressant de chercher les inégalitées valides pour ce problème pour améliorer le processus de branch&cut.

Malheureusement, on a trouvé que le modèle n'élimine pas des cycles sur le chemin, on pouurais donc ajouter les contraintes MTZ d'élimination des sous-tours.

Cities	PR	Cutting Planes			Branch&Cut			Dualization			B&C Heur			Heuristic		
		Time(s)	Gap	Best bound	Time(s)	Gap	Best bound	Time(s)	Gap	Best bound	Time(s)	Gap	Best bound	Time(s)	Gap	Best bound
20	0.39	1.37	0.0	15332.56	0.48	0.0	15332.56	0.03	0.0	15332.56	0.71	0.0	15332.56	0.18	0.0	15332.56
20	0.24	0.57	0.0	7076.52	0.18	0.0	7076.52	0.03	0.0	7076.52	0.08	0.0	7076.52	0.0	0.0	7076.52
20	0.21	1.24	0.0	8699.36	0.55	0.0	8699.36	0.1	0.0	8699.36	0.12	0.0	8699.36	0.0	0.09	9454.47
40	0.38	12.95	0.0	12664.33	1.79	0.0	12664.33	0.3	0.0	12664.33	0.25	0.0	12664.33	0.0	0.0	12664.33
40	0.33	60.52	0.0	14119.91	8.4	0.0	15058.97	0.16	0.0	15058.97	1.26	0.0	15058.97	0.0	0.0	15058.97
40	0.28	42.76	0.0	17330.1	2.22	0.0	17330.1	0.33	0.0	17330.1	0.51	0.0	17330.1	0.0	0.0	17330.1
60	0.34	35.18	0.0	10633.33	4.08	0.0	10633.33	0.23	0.0	10633.33	0.71	0.19	8571.81	0.0	0.19	8571.81
60	0.29	3.77	0.0	23914.23	2.72	0.0	23914.23	0.24	0.0	23914.23	0.78	0.0	23914.23	0.0	0.0	23914.23
60	0.42	14.36	0.0	21277.0	3.01	0.0	21277.0	0.3	0.0	21277.0	0.9	0.0	21277.0	0.0	0.23	16299.13
80	0.19	60.41	0.0	10411.68	16.04	0.0	10857.06	0.5	0.0	10857.06	44.8	0.0	10857.06	0.0	0.0	10857.06
80	0.18	33.15	0.0	11610.8	8.02	0.0	11610.8	0.23	0.0	11610.8	42.96	0.0	11610.8	0.0	0.23	14277.49
80	0.34	6.28	0.0	18619.6	2.8	0.0	18619.6	0.29	0.0	18619.6	13.04	0.02	19029.52	0.0	0.12	16299.13
100	0.15	61.07	0.0	8374.96	31.82	0.0	8462.96	0.32	0.0	8462.96	4.51	0.0	8462.96	0.0	0.28	10857.06
100	0.18	60.2	0.0	12575.96	16.76	0.0	12938.4	0.59	0.0	12938.4	1.94	0.0	12938.4	0.0	0.96	25320.15
100	0.22	60.85	0.0	18042.84	22.25	0.0	19446.56	0.84	0.0	19446.56	3.65	0.0	19446.56	0.0	0.08	20939.94
120	0.15	61.86	0.0	9761.52	60.14	0.03	9633.51	0.48	0.0	9921.16	15.41	0.0	9921.16	0.01	0.23	12218.98
120	0.18	60.21	0.0	12575.96	60.05	0.03	12575.96	0.94	0.0	12938.03	5.73	0.0	12938.4	0.0	0.96	25320.15
120	0.29	60.42	0.0	24676.03	60.12	0.22	20571.54	10.01	0.0	26508.07	60.0	0.0	22001.72	0.0	0.13	22991.44
140	0.19	61.95	0.0	12989.0	60.28	0.13	12546.48	2.64	0.0	14378.52	60.0	0.0	13576.94	0.01	0.09	13118.65
140	0.2	43.0	0.0	14380.14	25.6	0.0	14380.14	0.44	0.0	14380.14	2.1	0.0	14380.14	0.0	0.71	24651.02
140	0.23	61.83	0.0	30755.78	60.35	0.12	29003.36	5.93	0.0	33079.16	60.02	0.01	31457.44	0.0	0.04	31738.91
160	0.17	60.59	0.0	9365.32	60.68	0.06	9365.32	0.74	0.0	9972.16	40.76	0.0	9972.16	0.01	0.34	13395.74
160	0.2	60.2	0.0	13655.26	60.33	0.11	12752.85	0.94	0.0	14380.14	12.98	0.0	14380.14	0.0	0.71	24651.02
160	0.21	60.03	0.0	12331.87	60.61	0.05	12331.87	0.81	0.0	12962.98	5.51	0.0	12962.98	0.0	0.85	24039.44
180	0.17	61.89	0.0	9303.68	60.35	0.13	8657.15	1.04	0.0	9972.16	60.03	0.0	9365.32	0.01	0.34	13395.74
180	0.28	60.27	0.0	19643.42	60.56	0.03	19384.67	0.64	0.0	20069.82	16.07	0.0	20069.82	0.01	0.52	30526.01
180	0.21	60.78	0.0	11753.74	60.09	0.15	10990.1	1.47	0.0	12962.98	33.75	0.0	12962.98	0.01	0.85	24039.44
200	0.17	61.64	0.0	9075.08	64.47	0.14	8573.31	1.36	0.0	9972.16	60.09	0.0	9082.72	0.01	0.34	13395.74
200	0.28	62.07	0.0	19369.02	60.35	0.1	18055.72	0.93	0.0	20069.82	32.24	0.0	20069.82	0.01	0.57	31592.01
200	0.2	62.8	0.0	27835.61	61.0	0.1	27635.63	19.76	0.0	30595.12	60.07	0.0	27668.87	0.01	0.0	30574.8
250	0.2	61.42	0.0	15094.09	61.87	0.16	14271.17	4.63	0.0	17046.05	60.06	0.0	16538.74	0.02	0.13	19247.97
250	0.28	62.23	0.0	19187.8	72.62	0.22	15628.45	24.98	0.0	20069.82	60.13	0.0	17605.87	0.04	0.52	30526.01
250	0.14	67.81	0.0	34000.13	0.0	0.0	-	60.02	0.03	35859.05	60.12	0.09	34000.98	0.04	0.09	37057.41
300	0.21	64.0	0.0	17283.0	217.21	0.22	16986.5	21.78	0.0	20888.14	61.12	0.0	17394.0	0.06	0.05	21954.18
300	0.24	60.78	0.0	14565.08	355.71	0.22	14396.05	32.28	0.0	18546.4	60.05	0.0	15554.32	0.05	0.74	32337.01
300	0.21	61.88	0.0	28331.61	314.25	0.13	28481.76	9.76	0.0	32839.96	60.61	0.0	28908.76	0.04	0.09	35802.77
350	0.19	75.72	0.0	15790.0	161.62	0.14	16573.88	17.92	0.0	19379.24	60.84	0.0	16845.25	0.08	0.13	21954.18
350	0.31	76.07	0.0	20253.75	608.16	0.23	20942.53	5.33	0.0	27353.87	60.01	0.0	26545.48	0.06	0.12	30611.05
350	0.25	76.42	0.0	23044.65	635.01	0.04	23203.28	4.24	0.0	24253.84	60.45	0.0	23373.95	0.05	0.6	38848.64

Table 4: Comparaison les performances de différentes approaches.