Projet Optimisation Robuste

Benoit DUVAL, Raphaël TAISANT

16 février 2024

Résolution des sous-problèmes

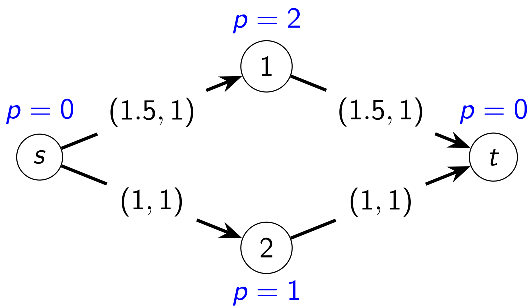
$$
\begin{array}{lll}
\max_{\delta^1} & \sum_{a\in A} d_a \left(1+\delta_a^1\right) x_a^* & \\
st & \sum_{ij\in A} \delta_a^1 \leq d_1 & \\
 & \delta_a^1 \in [0, D_a] & \forall\, a \in A
\end{array}
\tag{1}
$$

Entrée: G, P ensemble d'arcs utilisés
Trier les arcs de P selon d_a décroissant
budget $\leftarrow 0.0$, $\delta_a^1 \leftarrow 0$, score $\leftarrow \sum_{a\in P} d_a$
while budget $< d_1$ et il reste des arcs dans P **do**
    Sélectionner l'arc a avec le $+$ grand d_a
    $\delta_a^1 \leftarrow \min(D_a, d_1 -$ budget$)$
    budget $\leftarrow$ budget $+\delta_a^1$
    score $\leftarrow$ score $+\delta_a^1 d_a$
end while

Contrainte Borne inférieure

- Beaucoup de temps passé à prouver l'optimalité en montant la borne inf
- A chaque itération, on obtient une borne inférieure (solution optimale)
- On ajoute la contrainte $z \geq$ borne inférieure précédente
- Toujours vraie car on ajoute des contraintes et permet de clore la résolution plus rapidement

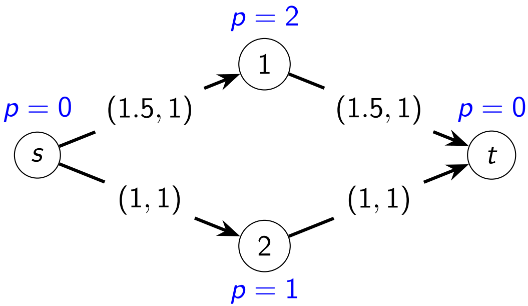
Sauvegarde Solution



$S = 1$

- itération 1: s-1-t, z=2. Non admissible car score pas robuste, mais solution admissible en terme de p

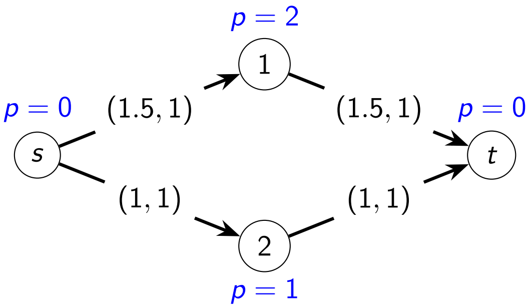
Sauvegarde Solution



$S = 1$

- itération 1: s-1-t, z=2. Non admissible car score pas robuste, mais solution admissible en terme de p
- itération 2: s-2-t. Non admissible car score pas robuste et solution non admissible en p

Sauvegarde Solution



$S = 1$

- itération 1: s-1-t, z=2. Non admissible car score pas robuste, mais solution admissible en terme de p
- itération 2: s-2-t. Non admissible car score pas robuste et solution non admissible en p
- Fin de l'algorithme (limite de temps)

Warm starts

- A chaque itération, on résout le problème maitre à la racine
- Pour accélerer sa résolution, on peut lui ajouter des *warm starts*
- Ce qui a marché le mieux, c'est de créer un seul *model* au début, ainsi qu'un objet *cplex* et ajouter à chaque itération les contraintes au *model* courant et laisser *cplex.solve()*

Réduction des symétries (1/2)

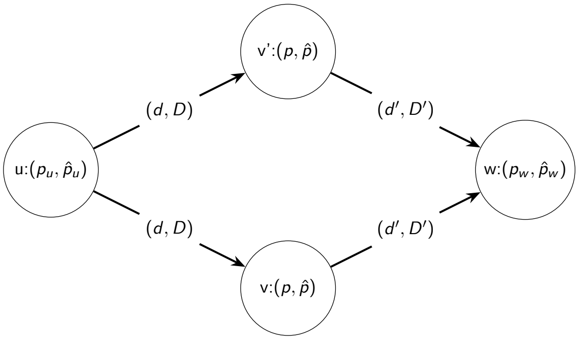


Figure: Sous-chemins entre u et w

Réduction des symétries (2/2)

Équivalence entre les sous-chemin $(u - v - w)$ et $(u - v' - w)$: on interdit $(u - v' - w)$

$$x_{uv'} + x_{v'w} \leq 1$$

Trop de contraintes ajoutées pour être réalisé en pré-traitement.
Dans les CallBacks: pour tout triplet de sommet (u, v, w), on réduit les symétries des sous chemins entre u et w.

Nette amélioration des performances du Branch and Cut et de la méthode des Plans coupants.
Possibilité de généraliser cette méthode avec des sous-chemin plus long ou des sous-chemin strictement dominés.

Heuristique (1/4)

$$\text{cout du chemin } s-u = \text{longueur robuste de } s-u$$
$$+ K \times \text{ poids robuste de } s-u$$

Exploration du graphe avec une structure d'information adaptée.
Pour chaque noeuds:

- le parent dans le sous chemin.
- les parties statique et robuste du cout et du poids du sous-chemin.
- le contenu des sac à dos associés.

Amélioration des performances avec A^*:

$$\text{borne inf de } u-t = \text{distance statique min } u-t$$
$$+ K \times \text{ poids statique min de } u-t$$

Heuristique (2/4)

$$\text{coût du chemin } s - u = \text{longueur robuste de } s - u$$
$$+ K \times \text{ poids robuste de } s - u$$

Utilisation de A^* pour différentes valeurs de K. Objectif: trouver le K^* qui fait coïncider la solution de l'heuristique avec la solution optimale.

- $\forall K < K^*$ on a $sol(K)$ non admissible, et $val(K) < val(K^*)$
- $\forall K > K^*$ on a $sol(K)$ admissible, et $val(K) > val(K^*)$

L'heuristique garde en mémoire un K_{inf} et un K_{sup} pour encadrer K^*.

Heuristique (3/4)

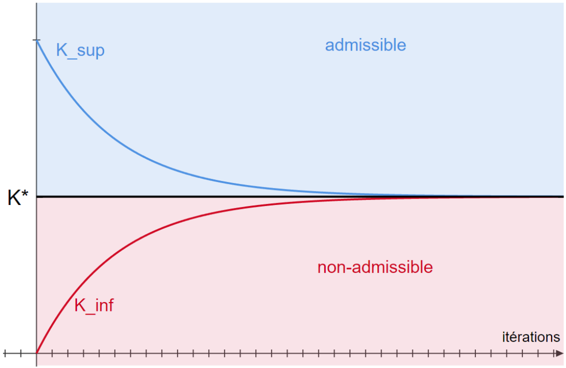


Figure: Convergence de K_{inf} et K_{sup} dans l'heuristique

Heuristique (4/4)

Limites de l'heuristique:

- Perte des propriétés des algorithmes d'origine
- Aucune garantie sur la borne inférieure
- Peu pertinent si il est "facile" de trouver un chemin admissible
- Certains cas pathologique ou l'algorithme n'arrive pas à instantier un K_{sup}

Conclusion sur l'heuristique: très bonne performance en pratique. La solution renvoyée en quelques secondes est la même que CPLEX en plusieurs minutes

Résultats

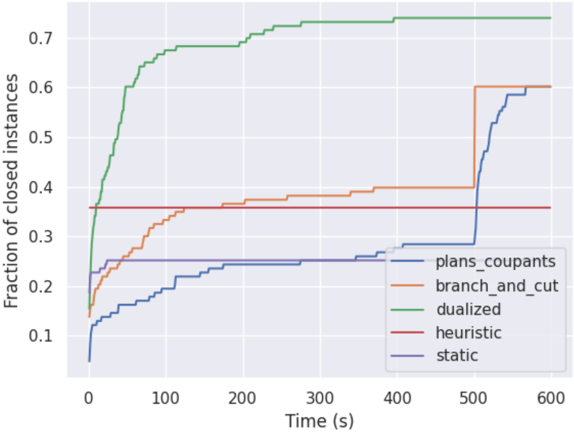


Figure: Fraction d'instances fermées par les différentes méthodes

Résultats

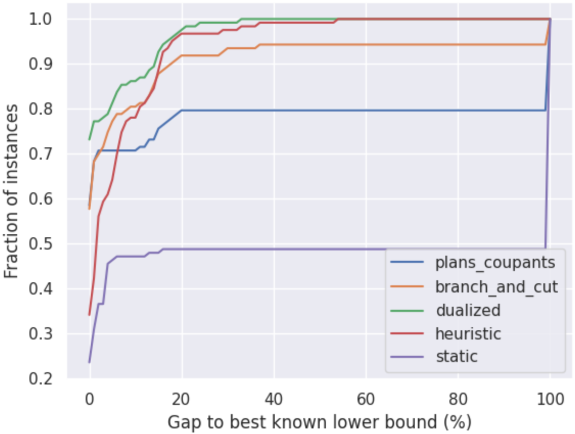


Figure: Fraction d'instances résolues avec un gap inférieur à l'abscisse par les différentes méthodes

Ouverture

Pistes d'approfondissement:

- étendre la méthode actuelle de réduction de symétrie pour les méthodes de Branch and Cut et plans coupants
- étudier l'origine des symétries pour les traiter dans la partie dualisation
- utiliser l'heuristique comme un WarmStart pour la résolution du problème dualisé
- ajouter du preprocessing pour réduire l'instance envoyée au solveur

Une autre façon de comparer les méthodes serait d'évaluer l'instant auquel elles obtiennent leur meilleure solution.