Une nouvelle borne pour les problèmes d'optimisation combinatoire robuste avec des coûts sous forme d'intervalles

Hugo Gilbert[1], Olivier Spanjaard[1]

Sorbonne Universités, UPMC Univ Paris 06, CNRS, LIP6 UMR 7606, Paris, France
{hugo.gilbert,olivier.spanjaard}@lip6.fr

Mots-clés : *optimisation combinatoire robuste, minmax regret, branch and bound.*

1 Introduction

Pour de nombreux problèmes d'optimisation combinatoire (*e.g.*, plus court chemin, arbre couvrant, ...), la fonction objectif peut s'écrire sous la forme suivante :

$$\min_{x \in \mathcal{X}} \sum_{i=1}^{n} c_i x_i \tag{1}$$

où $\mathcal{X} \subset \{0,1\}^n =: \mathbb{B}^n$ représente l'ensemble des solutions admissibles et c_i représente le coût associé à l'élément i.

Cependant, dans de nombreux cas, la fonction de coût est incertaine et le problème devient alors un problème d'optimisation robuste. Nous considérons ici le cas où à chaque élément est associé un intervalle représentant l'ensemble des coûts possibles qu'il peut avoir. L'attribution à chaque élément d'un coût est alors appelé un scénario et nous désignerons par $\mathcal{U}$ l'ensemble des scénarios possibles.

Une approche classique est de chercher à déterminer une solution réalisable qui minimise le regret maximal qu'elle peut engendrer. Étant donnée une solution réalisable x, le regret induit par un scénario c est défini comme la différence entre le coût de x dans le scénario c et le coût d'une solution optimale dans le scénario c. Le regret maximal $Reg(x)$ d'une solution admissible x est alors défini comme le regret maximal induit par un scénario. Plus formellement :

$$Reg(x,c) = \sum_{i=1}^{n} c_i x_i - \min_{y \in \mathcal{X}} \sum_{i=1}^{n} c_i y_i \qquad\qquad Reg(x) = \max_{c \in \mathcal{U}} Reg(x,c) \tag{2}$$

La fonction objectif d'un problème d'optimisation combinatoire minmax regret peut alors s'écrire sous la forme suivante :

$$\min_{x \in \mathcal{X}} Reg(x) \tag{3}$$

Nous désignerons par OPT le regret de la solution minmax regret optimale.

Les variantes robustes de nombreux problèmes d'optimisation combinatoire ont été étudiées [1, 4, 9] et plusieurs solutions algorithmiques sont déjà connues. Malheureusement, le problème minmax regret a souvent une complexité plus élevée que le problème d'optimisation original. Par exemple, la version minmax regret du plus court chemin est NP-difficile pour des coûts sous forme d'intervalles [2].

Les problèmes NP-difficiles étant coûteux à résoudre exactement, on peut alors chercher une solution avec une garantie d'approximation en temps polynomial. Si les coûts sont sous forme d'intervalles, un algorithme d'approximation consiste à déterminer la meilleure solution pour le scénario qui attribue à chaque élément la valeur médiane de l'intervalle de coût correspondant. En effet, cette solution est toujours une 2-approximation de la solution optimale [5]. En d'autres

termes, le regret de cette solution est toujours inférieur à 2 fois celui de la solution optimale. Cette borne est atteinte pour certaines instances. Cependant dans de nombreux cas, cette solution a un regret bien inférieur à deux fois celui de l'optimal, et elle est même souvent une solution optimale. De plus, comme montré récemment par Chassein et Goerigk [3], pour une instance donnée, cette garantie d'approximation peut être améliorée. Leurs travaux reposent sur une nouvelle procédure de calcul d'une borne inférieure sur le regret de la solution optimale.

Dans cette communication, nous montrons que cette garantie d'approximation peut encore être améliorée. Pour cela, à la suite de Chassein et Goerigk, nous proposons également une procédure de calcul d'une borne inférieure sur le regret de la solution optimale et nous montrons que la borne que nous proposons est toujours plus précise. La validité de la borne inférieure proposée se fonde sur des arguments de théorie des jeux, et son calcul est réalisé à l'aide d'un algorithme de double oracle [6] que nous spécifions. Cette borne inférieure est très générale et peut être calculée pour tout problème d'optimisation combinatoire robuste si une primitive résolvant le problème d'optimisation classique est connue. Nous détaillons ensuite comment implanter efficacement cette borne dans le cadre d'un algorithme de type branch and bound.

Enfin nous appliquons notre méthode au problème de plus court chemin robuste. Nous montrons que notre approche permet des gains de temps de calcul significatifs.

2 Une nouvelle borne inférieure sur le regret de la solution optimale

Nous développons dans cette section une analyse orientée théorie des jeux des versions minmax regret des problèmes d'optimisation combinatoire à coût intervalle. Cette analyse nous permettra de définir la borne inférieure sur OPT étudiée dans cette communication.

La version minmax regret d'un problème d'optimisation combinatoire induit un jeu à deux joueurs à somme nulle pour lequel les ensembles des stratégies pures sont : celui des solutions admissibles $\mathcal{X}$ pour le premier joueur (appelé par la suite joueur x) et celui des scénarios possibles $\mathcal{U}$ pour le deuxième joueur (appelé par la suite joueur c). Le gain du jeu est alors donné par $Reg(x, c)$, le regret de la solution admissible x dans le scénario c. Le joueur x (resp. joueur c) peut aussi jouer avec une stratégie mixte en utilisant une distribution de probabilité $P_{\mathcal{X}}$ (resp. $P_{\mathcal{U}}$) sur les stratégies pures. La fonction de regret $Reg(.,.)$ est alors étendue par linéarité aux stratégies mixtes, autrement dit, le regret attaché à une stratégie mixte correspond à une espérance de regret.

Étant donnée la stratégie d'un joueur, une meilleure réponse est une stratégie pure optimale qui peut être jouée par le joueur adverse. Dans notre communication, nous définirons les procédures (aussi appelées *oracles*) qui, étant donnée une stratégie mixte d'un des deux joueurs, retournent une meilleure réponse pour le joueur adverse.

Un graphe avec des coûts de type intervalle est montré sur la Figure 1a. Pour le problème du plus court chemin entre les noeuds s et t, l'oracle du joueur c devra par exemple répondre à la question "quel est le scénario qui maximise le regret si le joueur x joue les chemins (s, c, e, t) et (s, d, f, t) avec une probabilité de 0.5 chacun ?". Une meilleure réponse est montrée Figure 1b.

Nous énonçons maintenant un résultat de Chassein et Goerigk qui nous permet de faire le lien entre la notion de meilleure réponse et celle de borne inférieure sur OPT.

Proposition 1. *[3] Soit $P_{\mathcal{U}}$ une stratégie mixte du joueur c, alors :*

$$OPT \geq \min_{x \in \mathcal{X}} Reg(x, P_{\mathcal{U}}) \tag{4}$$

En d'autres termes, le regret du joueur x quand il joue une meilleure réponse face à une stratégie mixte du joueur c est toujours une borne inférieure sur OPT.

La borne inférieure proposée par Chassein et Goerigk [3] est la meilleure borne inférieure possible du type décrit par la proposition 1 quand le joueur c ne considère qu'un ensemble "très" restreint de stratégies mixtes.

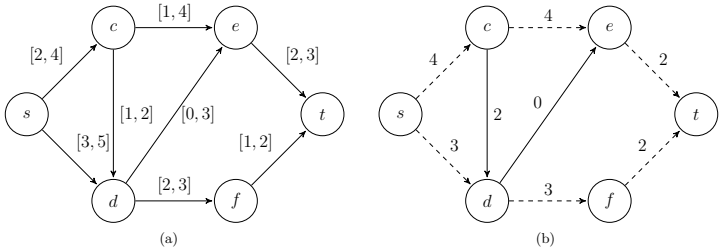

FIG. 1 – (a) Un exemple de graphe avec des coûts intervalles. (b) Une meilleure réponse du joueur c à la stratégie mixte du joueur x qui choisit chaque chemin en pointillés avec une probabilité de 0.5.

La borne inférieure que nous étudions dans cette communication est plus précise car elle est la meilleure borne possible du type décrit par la proposition 1, i.e. $\max_{P_\mathcal{U}} \min_{x \in \mathcal{X}} (Reg(x, P_\mathcal{U}))$, où $P_\mathcal{U}$ n'est restreint à aucun sous-ensemble de stratégies mixtes particulier. Cela revient à trouver un équilibre de Nash du jeu induit par les solutions admissibles et les scénarios possibles.

Malheureusement, ce jeu est de taille combinatoire et ne peut pas être résolu directement. Dans la prochaine section nous présentons donc une approche de type double oracle pour résoudre ce jeu sans spécifier explicitement l'ensemble des stratégies pures des deux joueurs.

3 Résolution du jeu

L'approche de type double oracle pour les jeux comportant un ensemble combinatoire de stratégies a été proposée par McMahan et al. [6]. Elle permet de déterminer un équilibre de Nash pour un jeu fini à somme nulle et à deux joueurs dès lors qu'une procédure de meilleure réponse (appelée *oracle*) est connue pour chaque joueur (procédures notées $MR_x(\cdot)$ et $MR_c(\cdot)$). Étant donnée une stratégie mixte $P_\mathcal{X}$ (resp. $P_\mathcal{U}$), nous notons $MR_c(P_\mathcal{X})$ (resp. $MR_x(P_\mathcal{U})$) la stratégie pure c (resp. x) maximisant $Reg(P_\mathcal{X}, c)$ (resp. minimisant $Reg(x, P_\mathcal{U})$).

Il est possible de décrire l'approche double oracle comme suit pour le jeu induit par un problème d'optimisation minmax regret. Cet algorithme commence avec des ensembles S_x et S_c de stratégies pures réduits (des singletons dans l'Algorithme 1), et ajoute progressivement des éléments à ces deux ensembles en appliquant les oracles à l'équilibre de Nash du jeu restreint à S_x et S_c. L'équilibre de Nash du jeu restreint est calculé par programmation linéaire [8].

L'algorithme continue jusqu'à convergence. La convergence est atteinte quand les meilleures réponses générées par les deux oracles sont déja contenues dans S_x et S_c. À chaque itération, d'après la proposition 1, le regret $Reg(MR_x(P_\mathcal{U}), P_\mathcal{U})$ est toujours une borne inférieure sur OPT. Ainsi nous pouvons espérer obtenir une "bonne" borne inférieure sur OPT après seulement quelques itérations de l'algorithme.

Algorithme 1 : Algorithme de Double Oracle

Données : Solutions admissibles $\mathcal{X}$ et scénarios possibles $\mathcal{U}$, singletons $S_x = \{x\}$ contenant une solution admissible arbitraire et $S_c = \{c\}$ contenant un scénario arbitraire.

Résultat : Un équilibre de Nash (possiblement mixte)

converge ← Faux
tant que *converge est Faux* **faire**
  Calculer équilibre de Nash $(P_\mathcal{X}, P_\mathcal{U}) \in G = (S_x, S_c, Reg|_{S_x, S_c})$
  Calculer $x = MR_x(P_\mathcal{U})$ et $c = MR_c(P_\mathcal{X})$
  si $x \in S_x$ *et* $c \in S_c$ **alors** converge ← Vrai **sinon** ajouter x à S_x et c à S_c
retourner $(P_\mathcal{X}, P_\mathcal{U})$

En pratique, nous observons en effet que l'algorithme de double oracle fournit rapidement une borne inférieure précise et cela souvent bien avant convergence. Ce calcul de borne est donc un bon candidat pour pouvoir être utilisé dans un algorithme de type branch and bound.

4 Adaptation de la borne inférieure pour un algorithme de type branch and bound

Nous montrons maintenant comment adapter notre méthode de calcul de borne inférieure pour l'intégrer à un algorithme de type branch and bound visant à determiner une solution minmax regret optimale.

Il est bien connu que deux éléments sont particulièrement importants pour l'efficacité d'un algorithme de type branch and bound : i) une borne inférieure rapide à calculer et précise doit être disponible à chaque noeud de l'arbre de recherche, ii) l'information obtenue dans un noeud doit pouvoir être transmise à ses enfants pour accélérer les calculs qui y sont réalisés.

La stratégie de branchement que nous utilisons consiste à développer un noeud de recherche en deux nouveaux noeuds en forçant un élément e à appartenir ou à ne pas appartenir aux solutions réalisables considérées. Un noeud de l'arbre de recherche définit donc un ensemble restreint de solutions admissibles $\mathcal{X}' \subseteq \mathcal{X}$ à partir de deux ensembles : i) l'ensemble $IN(\mathcal{X}')$ des éléments qui doivent faire partie de chaque solution admissible de $\mathcal{X}'$ et ii) l'ensemble $OUT(\mathcal{X}')$ des éléments qui ne doivent faire partie d'aucune solution de $\mathcal{X}'$.

Pour calculer une borne inférieure sous les contraintes induites par $IN(\mathcal{X}')$ et $OUT(\mathcal{X}')$, il est facile d'adapter notre approche de type double oracle : le seul changement consiste à calculer les meilleures réponses du joueur x dans $\mathcal{X}'$ plutôt que dans $\mathcal{X}$. De plus, deux remarques permettent d'accélérer les calculs dans l'algorithme 1 en tirant parti des calculs réalisés au noeud père : i) l'ensemble des scénarios générés S_c n'est pas dépendant du noeud de recherche exploré car la stratégie de branchement n'induit pas de contraintes sur les scénarios ; ii) l'ensemble des solutions admissibles S_x générées à un noeud de recherche est lui partitionné en deux ensembles $S_x^e = \{x \in S_x | x_e = 1\}$ et $S_x^{\overline{e}} = \{x \in S_x | x_e = 0\}$. L'ensemble S_x^e (resp. $S_x^{\overline{e}}$) est transmis au noeud fils dont les solutions admissibles doivent contenir (resp. interdire) e.

Nous avons implanté l'approche proposée pour l'appliquer au problème de plus court chemin robuste. Les résultats numériques obtenus montrent que notre approche permet d'obtenir un gain de temps significatif par rapport aux branch and bounds précédemment proposés dans la littérature [3, 7].

Références

[1] H. Aissi, C. Bazgan, and D. Vanderpooten. Min-max and min-max regret versions of combinatorial optimization problems : A survey. *Eur. Jour. of Oper. Res.*, 197(2) :427–438, 2009.

[2] I. Averbakh and V. Lebedev. Interval data minmax regret network optimization problems. *Discrete Applied Mathematics*, 138(3) :289 – 301, 2004.

[3] A. B. Chassein and M. Goerigk. A new bound for the midpoint solution in minmax regret optimization with an application to the robust shortest path problem. *Eur. Jour. of Oper. Res.*, 244(3) :739–747, 2015.

[4] O. E. Karasan, M. C. Pinar, and H. Yaman. Robust Shortest Path Problem with Interval Data. *Comp. and Oper. Res.*, 2002.

[5] A. Kasperski and P. Zielinski. An approximation algorithm for interval data minmax regret combinatorial optimization problems. *Inf. Proc. Lett.*, 97 :177–180, 2006.

[6] H. B. McMahan, G. J. Gordon, and A. Blum. Planning in the presence of cost functions controlled by an adversary. In *ICML*, pages 536–543, 2003.

[7] R. Montemanni, L. M. Gambardella, and A. V. Donati. A branch and bound algorithm for the robust shortest path problem with interval data. *Oper. Res. Lett.*, 32(3) :225–232, 2004.

[8] T.E.S Raghavan. Zero-sum two-person games. *Handbook of game theory*, 2 :735–768, 1994.

[9] H. Yaman, O. E. Karasan, and M. Ç. Pinar. Minimum spanning tree problem with interval data. *Oper. Res. Lett.*, 29, 1999.