Une variante de la méthode de Benders adverse pour le problème de lot-sizing robuste avec budget d'incertitude

Tom Portoleau[1,2], Romain Guillaume[2], Christian Artigues[1]

[1] LAAS-CNRS, Université de Toulouse, CNRS, France,
{tom.portoleau,christian.artigues}@laas.fr

[2] IRIT-CNRS, Université de Toulouse, France, romain.guillaume@irit.fr

Mots-clés : *production sous incertitude, budget d'incertitude, Benders adverse*

Résumé : *Dans ce papier nous présentons une variante de la méthode de Benders adverse pour résoudre un problème d'optimisation robuste pour le lot-sizing avec budget d'incertitude sur la demande cumulée. Nous rappelons d'abord l'approche classique avant de détailler cette variante qui se base sur une représentation compacte de scénarios dans un type de graphe particulier. Nous concluons avec des résultats expérimentaux comparant les deux approches, qui indiquent que notre variante peut être plus intéressante que l'approche classique lorsque le budget d'incertitude est bas.*

1 Introduction et description du problème

Pour résoudre des problèmes d'optimisation robuste, il existe de nombreuses approches. L'une d'entre elle, appelée la méthode de Benders adverse (ou parfois méthode des plans coupants) a été introduite dans [3]. Cette approche se base notamment sur la résolution itérée d'un problème dit adverse, c'est-à-dire problème où l'on cherche à calculer un pire scénario pour une solution réalisable donnée. Cette méthode est donc indiquée lorsque ce problème adverse est facile à résoudre et dépend entre autres de la nature de l'ensemble des scénarios. Dans [4], les auteurs présentent une variante de cette méthode où des pires scénarios sont calculés de manière approximative, et prouvent la convergence de leur méthode. Dans ce papier, nous proposons une autre variante de cette méthode, dans laquelle on cherche à calculer non pas un seul scénario pire cas mais un ensemble de pire scénarios, avec l'idée que l'on peut gagner du temps de calcul en réduisant le nombre d'itérations de l'algorithme. Nous présentons cette méthode pour un problème de lot-sizing avec contraintes de capacités, sous budget d'incertitude qui portera sur la demande cumulée. Typiquement, pour modéliser ce problème, on se donne un horizon découpé en T périodes de temps, et on associe à chaque période $t \leq T$ une demande d_t, et une production x_t. L'ensemble de ces variables de production constitue un plan de production. Le coût de production d'une période t est un coût de lancement c^P. A ce coût, des coûts linéraires liés à chaque pas de temps t et aux trois quantités suivantes sont ajoutées. Tout d'abord, si l'on produit plus que la demande, le surplus stocké sera noté I_t. Ensuite, la quantité effectivement vendue, qui satisfait au moins une partie de la demande est notée s_t. Et finalement, la quantité B_t représente la part de la demande non satisfaite. Ces trois quantités sont pondérées, respectivement par le coût de stockage c^I, le prix de vente b^P et le coût de backordering c^B. On supposera que la quantité produite à chaque pas de temps est bornée, et donc que les x_t prennent leurs valeurs dans un ensemble borné $\mathbb{X}$. On peut noter que ces différents coûts sont indépendants de la période de temps et que ce problème de lot sizing déterministe est NP-Difficile [1]. L'incertitude du problème porte sur la demande cumulée. Pour modéliser cela, on pose $X_t = \sum_{i=1}^{t} x_t$ et $D_t = \sum_{i=1}^{t} d_t$, qui représentent respectivement, la production cumulée jusqu'à la période t et la demande cumulée jusqu'à t. Concernant les incertitudes des demandes cumulées, on suppose que chaque demande D_t prend sa valeur dans un intervalle symétrique de la forme $[\hat{D}_t - \Delta_t, \hat{D}_t + \Delta_t]$, avec $\hat{D}_t \geq \Delta_t \geq 0$ où $\hat{D}_t$

est sa valeur nominal et Δ_t sa déviation maximale. On appelle scénario une réalisation de ces incertitudes et on note $\mathcal{U}$ un ensemble de scénarios. Dans la suite, on s'intéressera plus particulièrement à l'ensemble de scénarios continus suivant : $\mathcal{U} = \{\mathcal{D} = (D_t)_{t\leq T} : D_t \leq D_{t+1}, D_t \in [\hat{D}_t - \Delta_t, \hat{D}_t + \Delta_t], \|\mathcal{D} - \hat{\mathcal{D}}\|_1 \leq \Gamma\}$ où Γ est le budget d'incertitude. Pour résoudre ce problème de lot-sizing nous avons choisi d'utiliser le critère le robustesse absolu, ou **minmax**. Naturellement puisque le problème déterministe est déjà difficile, il le reste dans sa version incertaine.

2 Benders adverse et Benders adverse étendu

2.1 Benders adverse

Pour résoudre ce genre de problème, on peut utiliser une méthode dite de Benders adverse dont le principe est le suivant. L'idée est de résoudre le problème **minmax** (ou problème maître), d'abord sur un seul scénario, puis en augmentant le nombre de scénario considéré. Pour calculer le prochain scénario à ajouter au problème maître, on effectue une phase adverse, c'est-à-dire que l'on cherche, étant donnée une solution particulière du problème, à calculer le pire scénario pour cette solution. Ce problème est appelé problème adverse (et on le notera **Adv**). Une fois ce scénario calculé, il est ajouté à la liste des scénario du problème maître. On réitère alors jusqu'à ce que le critère d'arrêt soit atteint. Pour appliquer cette méthode à notre problème, il "suffit" donc d'être capable de résoudre le problème **minmax** pour une liste de scénarios donnée, et de résoudre le problème **Adv** pour une solution du problème donnée.
Concernant le problème **minmax**, on peut le modéliser par le programme linéaire en variables mixtes suivant :

$$
\begin{aligned}
& \min z \\
& \text{s.t.} \sum_{t=1}^{T}(c^I I_{t,l} + c^B B_{t,l} + c^P y_t - b^P s_{t,l}) \leq z && l \leq |\mathcal{U}'| && (1)\\
& B_{t,l} + X_t = D_t^l + I_{t,l} && 1 \leq t \leq T, l \leq |\mathcal{U}'| && (2)\\
& \sum_{i=1}^{t} s_{i,l} = D_t^l - B_{t,l} && 1 \leq t \leq T, l \leq |\mathcal{U}'| && (3)\\
& X_t - X_{t-1} \leq My_t && 2 \leq t \leq T && (4)\\
& X_1 \leq My_1 && && (5)\\
& B_{t,l}, I_{t,l}, s_{t,l}, X \in \mathbb{X} && 1 \leq t \leq T, l \leq |\mathcal{U}'| \\
& y_t \in \{0,1\} && 1 \leq t \leq T
\end{aligned}
$$

avec $M = D_T + \Gamma$ et où y_t est une variable binaire qui vaut 1 s'il y a une production au temps t et 0 sinon, $\mathcal{U}'$ est une liste de scénarios et D_t^l est la demande cumulée jusqu'à la période t dans le l-ième scénario de $\mathcal{U}'$. On notera ce programme linéaire $\mathcal{P}_{\mathcal{U}'}$. Quant au problème **Adv**, dans [2], les auteurs ont montré que ce problème était NP-Difficile au sens faible, et ont proposé un algorithme de programmation dynamique de complexité pseudo-polynomial pour le résoudre. L'idée de cet algorithme est de calculer un plus long chemin dans un graphe partitionné construit à partir d'un plan de production X de la manière suivante.

Définition 1 (Graphe de budget)**.** *On note ce graphe $\mathcal{G} = (V, A)$. L'ensemble des noeuds V partitionné en $T + 2$ niveaux et, mis à part le premier et le dernier niveau qui contiennent un unique noeud chacun, chaque niveau contient $\Gamma + 1$ noeuds. On note respectivement v_0 et v_{T+1} les noeuds du premier et du dernier niveau, et $v_{t,i}$ pour $1 \leq t \leq T + 1$ et $0 \leq i \leq \Gamma$ tous les autres noeuds du graphe. L'ensemble des arcs A est construit de sorte que l'arc de $v_{t-1,i}$ vers $v_{t,j}$ existe si et seulement si $i \leq j$ et $j - i \leq \Delta_t$ pour chaque t (ou niveau) compris entre 1 et T, et un arc de $v_{T,i}$ vers v_{T+1} est défini pour i compris entre 0 et Γ. Ce graphe est tel que chaque chemin de v_0 vers v_{T+1} représente un scénario de $\mathcal{U}$, et que si un noeud $v_{t,i}$ est sur ce chemin, alors i unités de budget ont été consommées entre la première période et la période t. Ensuite, on définit les coûts des arcs de sorte qu'ils représentent chacun la contribution à la fonction objectif de la décision de faire varier la demande à une période donnée. Plus formellement, on* *définit les coûts des arcs par :*

$$c_{t-1,i,t,j} = \begin{cases} \max\{c^I(X_t - (\hat{D}_t - (j-i))), c^B(\hat{D}_t + (j-i) - X_t)\} + c^P y_t & si\ 1 \leq t \leq T-1, \\ \max\{c^I(X_t - (\hat{D}_t - (j-i))) - b^P(\hat{D}_t - (j-i)), \\ c^B(\hat{D}_t + (j-i) - X_t) - b^P X_t\} + c^P y_t & si\ t = T, \\ 0 & si\ t = T+1 \end{cases}$$

où $c_{t-1,i,t,j}$ est le coût de l'arc entre les noeuds $v_{t-1,i}$ et $v_{t,j}$ lorsque cet arc existe, et $y_t = 0$ si $X_{t-1} = X_t$ et 1 sinon.

Pour reconstruire un scénario $\mathcal{D} = (D_t)_{t \leq T}$ à partir d'un chemin du graphe de budget, on pose, pour chaque arc sur le chemin $D_t = \hat{D}_t + s(t)(j-i)$ avec, pour $1 \leq t \leq T-1$, $s(t) = 1$ si $c^B(\hat{D}_t + (j-i) - X_t) \geq c^I(X_t - (\hat{D}_t - (j-i)))$ et $s(t) = -1$ sinon, et, pour $t = T$ $s(t) = 1$, si $c^B(\hat{D}_t + (j-i) - X_t) - b^P X_t \geq c^I(X_t - (\hat{D}_t - (j-i))) - b^P(\hat{D}_t - (j-i))$ et $s(t) = -1$ sinon. Puisqu'un chemin dans le graphe représente un scénario unique, le graphe $\mathcal{G}$ représente un ensemble de scénarios que l'on note $\mathcal{U}_{\mathcal{G}}$. On peut voir facilement que $\mathcal{U} = \mathcal{U}_{\mathcal{G}}$. De plus, on peut montrer (voir [2]) que le plus long chemin du graphe et le coût de ce chemin représentent respectivement le pire scénario pour une solution X et la valeur objectif de cette solution sur ce pire scénario. Ainsi, le problème **Adv** revient à un problème de calcul de plus long chemin dans un graphe orienté sans cycle, ce qui peut être résolu dans notre cas en $O(T\Gamma^2)$ par un algorithme de programmation dynamique que l'on ne détaillera pas dans ce papier. Finalement, l'algorithme de Benders adverse peut s'écrire comme présenté dans l'algorithme 1.

Algorithme 1 Benders adverse

$\mathcal{U}' \leftarrow \{\hat{\mathcal{D}}\}$
$(X, v) \leftarrow$ solution optimale et valeur objectif optimale de $\mathcal{P}_{\mathcal{U}'}$
$(D, v') \leftarrow$ pire scénario et sa valeur objectif du problème **Adv** pour la solution X
tant que $v \neq v'$ **faire**
  $v \leftarrow v'$
  $\mathcal{U}' \leftarrow \mathcal{U}' \cup \{\mathcal{D}\}$
  $(X, v) \leftarrow$ solution optimale et valeur objectif optimale de $\mathcal{P}_{\mathcal{U}'}$
  $(D, v') \leftarrow$ pire scénario et sa valeur objectif du problème **Adv** pour la solution X
retourne X

2.2 Benders adverse étendu

Nous proposons maintenant une variante de la méthode de Benders adverse présentée dans la section précédente. L'idée de cette variante est qu'à chaque phase adverse, on ne renvoie pas seulement le pire scénario pour une solution donnée, mais un ensemble de mauvais scénarios. Cependant, le problème maître $\mathcal{P}_{\mathcal{U}'}$, défini pour une liste de scénarios $\mathcal{U}'$, passe difficilement à l'échelle lorsque le nombre de scénarios explose. Pour pallier cela, on propose de réécrire le problème maître en se basant cette fois non pas sur une liste de scénarios, mais sur le graphe de budget introduit dans la section précédente. Une manière de voir le graphe de budget est de le considérer comme une représentation relativement compacte de l'ensemble des scénarios. On définit un sous-graphe de budget, qui représente un sous-ensemble de scénarios.

Définition 2 (Sous-graphe de budget)**.** *Soit $\mathcal{G} = (V, A)$ un graphe de budget défini suivant la définition 1. On dit que $\mathcal{G}' = (V, A')$ est un sous-graphe de budget de $\mathcal{G}$ si $A' \subset A$ et pour tout arc $a \in A'$, il existe un chemin de v_0 vers v_{T+1} passant par l'arc a. Les poids des arcs de ce graphe sont calculés de la même manière que pour le graphe de budget défini précédemment. On a alors $\mathcal{U}_{\mathcal{G}'} \subset \mathcal{U}_{\mathcal{G}}$.*

On propose maintenant un programme linéaire en variables mixtes qui étant donné un sous-graphe de budget $\mathcal{G}'$, calcule le planning robuste X, par rapport à l'ensemble des scénarios représentés par $\mathcal{G}'$. On le notera $\mathcal{P}'_{\mathcal{G}'}$ et il s'écrit de la manière suivante :

$$
\begin{aligned}
&\min \pi_{T+1} && \\
&\text{s.t. } \pi_{t_j} - \pi_{(t-1)_i} \leq c^I(X_t - (\hat{D}_t - (j-i))) + c^P y_t && 1 \leq t \leq T-1, (v_{t-1,i}, v_{t,j}) \in A' \quad (6)\\
&\pi_{t_j} - \pi_{(t-1)_i} \leq c^B(\hat{D}_t + (j-i) - X_t) + c^P y_t && 1 \leq t \leq T-1, (v_{t-1,i}, v_{t,j}) \in A' \quad (7)\\
&\pi_{t_j} - \pi_{(t-1)_i} \leq c^I(X_t - (\hat{D}_t - (j-i))) + c^P y_t - b^P(\hat{D}_t - (j-i)) && t = T, (v_{t-1,i}, v_{t,j}) \in A' \quad (8)\\
&\pi_{t_j} - \pi_{(t-1)_i} \leq c^B(\hat{D}_t + (j-i) - X_t) + c^P y_t - b^P X_t && t = T, (v_{t-1,i}, v_{t,j}) \in A' \quad (9)\\
&\pi_{T+1} - \pi_{T,i} \geq 0 && 0 \leq i \leq \Gamma \quad (10)\\
&X_t - X_{t-1} \leq M y_t && 2 \leq t \leq T \quad (11)\\
&X_1 \leq M y_1 && (12)\\
&\pi_0 = 0, \pi_v \in \mathbb{R} && v \in V \quad (13)\\
&y_t \in \{0,1\}, X_t \in \mathbb{X} && 1 \leq t \leq T \quad (14)
\end{aligned}
$$

avec $M = D_T + \Gamma$. L'idée de ce PLNE est de considérer notre problème comme un problème de plus long chemin, en termes de potentiels, dans le graphe $\mathcal{G}'$ avec des arcs dont le poids dépend des variables de décisions, à savoir X et y. Plus précisément, les termes à droite des inégalités des contraintes de (6) à (9) représentent les coûts des arcs, et chaque variable $\pi_{t,j}$ représentent la longueur du plus long chemin entre le noeud v_0 et le noeud $v_{t,j}$. Cette formulation a l'avantage d'être compacte alors qu'elle permet de résoudre le problème **minmax** pour un nombre possiblement exponentiel de scénarios, ce que la formulation précédente du problème ne permettait pas de faire. On peut maintenant considérer le problème de la phase adverse, qui cette fois consiste, étant donné un plan de production X, à calculer un sous-graphe de budget $\mathcal{G}'$ qui représente un sous-ensemble de mauvais scénarios pour X. Pour être plus précis, si l'on note C la longueur du plus long chemin dans un graphe de budget, on considère un paramètre τ. Alors l'algorithme 2 renvoie un sous-graphe de budget qui contient tous les noeuds et arcs constituant tous les chemins de longueur au moins τC dans le graphe de budget. Pour calculer un tel sous-graphe, on utilise un algorithme de programmation dynamique. D'abord on calcule le plus long chemin dans le graphe, ce qui est facile puisque le graphe est orienté et sans cycle. Une fois la valeur C de ce plus long chemin calculée, on identifie par backtracking les arcs qui contribuent aux chemins de longueur au moins τC. Ce sous-ensemble d'arcs correspond à celui du sous-graphe recherché.

Algorithme 2 Adv_subgraph

Entrée: Un graphe de budget $\mathcal{G} = (V, A)$ et $\tau \in [0, 1]$
  $\pi_0 \leftarrow 0$
  pour $t = 1 \rightarrow T+1$ **faire**
    pour $j = 0 \rightarrow \Gamma + 1$ **faire**
      $\pi_{t,j} \leftarrow \max\{\pi_{t-1,i} + c_{t-1,i,t,j} | (v_{t-1,i}, v_{t,j}) \in A\}$
  $A' = \{\}$
  pour $t = T \rightarrow 1$ **faire**
    pour $j = 0 \rightarrow \Gamma + 1$ **faire**
      si $\pi_{T,j} \geq \tau \pi_{T+1}$ **et** $t = T$ **alors**
        $A' \leftarrow A' \cup \{(v_{T,j}, v_{T+1})\}$
        $v_{T,j} \leftarrow$ marqué
      si $v_{t,j}$ est marqué **alors**
        pour $i = 0 \rightarrow \Gamma + 1$ **faire**
          si $\pi_{t-1,i} + c_{t-1,i,t,j} = \pi_{t,j}$ **alors**
            $A' \leftarrow A' \cup \{(v_{t-1,i}, v_{t,j})\}$
            $v_{t-1,i} \leftarrow$ marqué
  retourne $\mathcal{G}' = (V, A'), \pi_{T+1}$

Avant de pouvoir adapter cette étape de phase adverse pour une approche de Benders adverse, il reste une dernière étape. En effet dans l'approche classique, on ajoute les scénarios un à un dans une liste. Dans notre cas, cela signifie que l'on a besoin d'une opération d'union ou de fusion de graphes. On propose donc la fonction de fusion de graphes suivante : $Merge((V, A'), (V, A'')) = (V, A' \cup A'')$. En utilisant cette opération, on ne change ni la correction ni la terminaison de l'algorithme, l'important étant que le graphe contienne au moins le pire scénario pour la solution de l'itération en cours. On peut maintenant présenter l'algorithme de Benders adverse étendu, qui est décrit dans l'algorithme 3. De la même manière que l'on initialisait la liste de scénarios du Benders adverse classique avec le scénario nominal $\hat{\mathcal{D}}$, on initialise cet algorithme avec le sous-graphe de budget qui ne contient que le scénario nominal, que l'on note $\hat{\mathcal{G}}$.

Algorithme 3 Benders adverse Étendu

$\mathcal{G}' \leftarrow \hat{\mathcal{G}}$
$(X, v) \leftarrow$ solution optimale et valeur objectif optimale de $\mathcal{P}'_{\mathcal{G}'}$
$(\mathcal{G}'', v') \leftarrow$ sous-graphe de budget et son plus long chemin du problème **Adv_subgraph** pour la solution X
tant que $v \neq v'$ **faire**
  $v \leftarrow v'$
  $\mathcal{G}' \leftarrow Merge(\mathcal{G}', \mathcal{G}'')$
  $(X, v) \leftarrow$ solution optimale et valeur objectif optimale de $\mathcal{P}'_{\mathcal{G}'}$
  $(\mathcal{G}'', v') \leftarrow$ sous-graphe de budget et son plus long chemin du problème **Adv_subgraph** pour la solution X
retourne X

3 Expérimentations numériques et conclusion

Afin de comparer les deux méthodes, nous avons pris des instances de **csplib.org**. Ces instances comportent entre 200 et 500 périodes de temps et décrivent un problème de lot-sizing multi-produit avec un coût de stockage variant avec le temps. Pour adapter ces instances à notre problème, on a tout d'abord agrégé tous les produits et leurs demandes associées en un seul produit. Pour les différents coûts de la fonction objectif, on a choisi le coût de stockage c^I de la première période de l'instance initiale puis l'avons étendu à toutes les périodes. Tous les autres coûts (c^B, c^P, et b^P) ont été tirés aléatoirement en suivant une loi uniforme dans l'intervalle $[\frac{c^I}{2}, \frac{3c^I}{2}]$. Pour contraindre les x_t à chaque pas de temps, on les a borné par une valeur σ_t tirée uniformément dans l'intervalle $[0.8\hat{d}_t, 1.2\hat{d}_t]$. Chacun de ces tirages a été réalisé une unique fois par instance, puis cette même instance randomisée a été réutilisée plusieurs fois dans les expérimentations. Les différents programmes linéaires ont été résolus avec CPLEX 12.9 et tous les calculs ont été effectués avec un processeur Intel Xeon CPU E5-2695 v4 2.10GHz sous Linux Ubuntu 16.04.4. Pour ces expérimentations, nous avons considéré deux paramètres. Tout d'abord le budget d'incertitude Γ que l'on a fait varier entre 1 et 10, puis τ notre paramètre de seuillage des sous-graphes de budget que l'on a fait varier entre 0 et 1. Les deux critères qui sont observés sont : le nombre d'itération de l'algorithme et le temps de calcul. Les résultats pour l'algorithme de Benders adverse étendu (BAE) sont affichés dans la Table 1 avec à gauche dans les cellules le nombre d'itérations moyen, et à droite le temps de calcul. Quant à l'algorithme de Benders adverse (BA), les résultats ne sont pas présentés dans la table tout simplement parce que le nombre d'itérations et le temps de calcul n'ont pas varié avec les différentes valeurs de ces paramètres. Cela peut s'expliquer par deux choses. Tout d'abord il est clair que l'algorithme BA ne dépend pas de τ. Ensuite la valeur de Γ n'intervient que dans la complexité du problème **Adv**, dont la complexité est négligeable devant le temps de résolution du problème maître. Ainsi, indépendemment des paramètres en jeu ici, le nombre moyen d'itérations de BA est de 5 et le temps de calcul moyen est 2.81 secondes. Les résultats en gras dans la Table 1 indiquent les combinaisons de paramètres pour lesquelles BAE est plus intéressant que BA. On peut d'abord remarquer que pour $\tau = 1$ le nombre d'itérations de BAE est le même que BA à savoir 5. Ce n'est pas très étonnant puisque avec ce choix de τ le sous-graphe

$\tau \backslash \Gamma$	1	3	5	7	9
0	**2.0** | **0.16**	**2.0** | **0.518**	**2.0** | **1.25**	**2.0** | **2.07**	**2.0** | 3.26
0.2	**2.0** | **0.16**	**2.0** | **0.509**	**2.0** | **1.27**	**2.0** | **2.15**	**2.0** | 3.22
0.4	**3.0** | **0.24**	**3.0** | **0.777**	**3.0** | **1.88**	**3.0** | 3.17	**3.0** | 4.76
0.6	**3.0** | **0.24**	**3.0** | **0.783**	**3.0** | **1.86**	**3.0** | 3.23	**3.0** | 4.68
0.8	**4.0** | **0.31**	**4.0** | **1.01**	**4.0** | **2.5**	**4.0** | 4.19	**4.0** | 6.6
1	5.0 | **0.39**	5.0 | **0.886**	5.0 | 3.09	5.0 | 5.17	5.0 | 8.54

TAB. 1 – Nombre moyen d'itérations et temps de calcul (en secondes) de l'algorithme de BAE en fonction du paramètre de seuillage τ et du budget d'incertitude Γ.

retourné par le problème adverse de BAE ne représente qu'un unique scénario, le même que celui retourné par le problème adverse de BA. Aussi, on observe que le temps de calcul de BAE croît lorsque la valeur de Γ et τ augmente. Ce qui peut paraître plus étonnant en revanche, c'est que le nombre d'itérations ne semble dépendre que de τ, et pas de Γ. Nous pensons que cela est dû au fait que la "structure" du problème est invariante par rapport à ces paramètres. Finalement, on remarque sur lorsque le budget d'incertitude Γ dépasse 7 pour nos instances, le temps de calcul de BAE devient plus élevé que celui de BA. En revanche, on observe que BAE est plus intéressant que BA lorsque le paramètre τ est assez bas, car il semble impacter le nombre d'itération, et donc le temps de calcul, bien que les expérimentations ne permettent pas de conclure sur l'intérêt de faire les calculs avec un τ élevé. Nous pensons cependant qu'il existe des problèmes pour lesquels le temps de calcul de BAE ne serait pas monotone avec τ, ou autrement dit, qu'il existerait une valeur de ce paramètre différente de 0 ou 1 qui permettrait de gagner en temps de calcul. Pour conclure, nous avons présenté une variante de la méthode de Benders adverse pour résoudre un problème d'optimisation robuste pour le lot-sizing avec budget d'incertitudes sur la demande cumulée. Les résultats expérimentaux sont encourageants et indiquent que notre variante peut être plus intéressante que l'approche classique si le budget d'incertitude est bas. Nous pensons également qu'il est possible d'améliorer les résultats en passant par des approches de compilation de connaissances, afin de trouver un langage de représentation des scénarios un peu plus compact que les sous-graphes de budget, et d'adapter notre problème maître afin de gagner en temps de calcul lors de sa résolution.
Ce travail a été partiellement financé par l'Agence Nationale de la Recherche (ANR), projet ANR-18-CE10-0007 “PER4MANCE”.

Références

[1] Gabriel R Bitran and Horacio H Yanasse. Computational complexity of the capacitated lot size problem. Management Science, 28(10) :1174–1186, 1982.

[2] Romain Guillaume, Adam Kasperski, and Paweł Zieliński. Production planning under demand uncertainty : a budgeted uncertainty approach. In Operations Research Proceedings 2019, pages 431–437. Springer, 2020.

[3] Almir Mutapcic and Stephen Boyd. Cutting-set methods for robust convex optimization with pessimizing oracles. Optimization Methods and Software, 24(3) :381–406, 2009.

[4] Julius Pätzold and Anita Schöbel. Approximate cutting plane approaches for exact solutions to robust optimization problems. European Journal of Operational Research, 284(1) :20–30, 2020.