Techniques de PLNE pour la résolution des sous-problèmes hydrauliques en gestion de prodution journalière

Alexandre Heintzmann, Pascale Bendotti, Cecile Rottner

EDF Lab Paris-Saclay, 7 bd Gaspard Monge, 91120 Palaiseau
{alexandre.heintzmann,pascale.bendotti,cecile.rottner}@edf.fr

Mots-clés : *polyèdre, branch&cut, Hydro Unit Commitment.*

Le **Hydro Unit Commitment** est un problème de planification journalière de production électrique propre aux centrales hydrauliques. Il s'inscrit comme l'un des sous-problèmes du problème de planification d'unités de production électrique, appelé **Unit Commitment Problem**. Ce problème est aujourd'hui traîté chez EDF par une relaxation Lagrangienne. Cette relaxation sépare le problème en un sous-problème pour chaque unité de même nature : vallée hydraulique, centrale thermique, etc. Le problème **Hydro Unit Commitment** contient une combinatoire très forte, ce qui fait que la résolution de ce problème demande un temps de calcul important. Etudier cette combinatoire permettrait d'une part d'avoir un meilleur contrôle sur les temps d'execution des PLNE, et d'autre part de garantir la qualité d'une solution obtenue. Nous proposons plusieurs familles d'inégalités valides pour le HUC à une usine, afin de mieux comprendre les différents couplages des contraintes du problème et d'améliorer la formulation.

Nous avons considéré une version du **HUC** à une usine, que nous avons modélisé de la manière suivante. Soit une vallée constituée de N usines et de $N+1$ réservoirs. Chaque usine est située entre deux réservoirs. Le principe de production hydroélectrique est le suivant. L'eau d'un réservoir amont se déverse dans le réservoir aval en passant par une usine, mettant ainsi en fonctionnement les turbines de l'usine. Un point de fonctionnement est défini comme étant le couple formé par un débit et une puissance. Pour chaque usine n, il y a un nombre discret de points de fonctionnement m^n, ce nombre est variable d'une usine à l'autre. Lorsqu'une usine n atteint un point de fonctionnement i, elle turbinera un débit d_i^n d'eau, ce qui produira une puissance p_i^n. De plus, si une usine est au point de fonctionnement i, elle est alors aussi à tous les points de fonctionnement $j < i$, ce qui revient à une *forme cumulative*, c'est-à-dire qu'on cumule les débits et les puissances des points de fonctionnement correspondants. Par exemple si l'usine n est au deuxième point de fonctionnement, le débit pour l'usine n sera $d_1^n + d_2^n$ et la puissance $p_1^n + p_2^n$. On peut voir une turbine comme une succession de points de fonctionnement. De plus, les turbines ont un ordre de démarrage imposé, il est alors suffisant de modéliser les points de fonctionnement sans modéliser les turbines. En discrétisant le temps sur un horizon T, chaque usine turbinera, à chaque pas de temps t, un certain volume d'eau, produisant une certaine quantité d'énergie pendant la durée l du pas de temps. Pour chaque réservoir n, il y a un volume d'eau initial V_0^n, mais aussi une capacité maximale $V_{max,t}^n$ et une capacité minimale $V_{min,t}^n$ à chaque pas de temps t. Au delà de la capacité du réservoir, imposer un volume maximal et minimal par pas de temps permet de fixer des volumes cibles à certains pas de temps, ce qui est nécessaire pour la gestion économique de l'eau. A chaque pas de temps t, un réservoir n reçoit un apport a_t^n. Cet apport peut être positif ou négatif. On aura une valeur unitaire v^n de l'eau stockée à la fin de l'horizon de temps dans le réservoir n. La quantité d'énergie produite au pas de temps t sera vendue à une valeur unitaire λ_t, variable au cours du temps. Les revenus prennent en compte la valeur totale de l'eau dans les réservoirs à la fin de l'horizon de temps, et la revente de l'énergie au cours du temps. Le problème **HUC** consiste à maximiser les revenus, en turbinant de manière à respecter les capacités et les volumes cibles des réservoirs

et le fonctionnement des turbines à tout instant. Pour la modélisation utilisée, nous utilisons les variables $x^n_{t,i}$, indiquant si on est au moins au point de fonctionnement i au pas de temps t pour l'usine n.

Nous avons remarqué que le problème était équivalent à plusieurs contraintes de sac-à-dos et sac-à-dos inversés avec des contraintes de précédence. On appelle contrainte de sac-à-dos inversé une contrainte de sac-à-dos où le sens de l'inégalité est inversé. En effet, à chaque pas de temps, les volumes maximaux et minimaux des réservoirs et les apports extérieurs d'eau feront qu'on ne pourra pas turbiner plus d'une certaine quantité d'eau, ou au contraire, on sera obligé de turbiner une quantité d'eau minimale. Comme nous avons des contraintes d'ordre sur nos variables $x^n_{t,j}$, on voit naturellement apparaître des contraintes de précédence dans ces sacs-à-dos. Ces contraintes de sac-à-dos sont imbriquées car les choix faits à un pas de temps t vont modifier les choix possibles pour tous les pas de temps $t', t' > t$. On aura alors, pour chaque pas de temps, une contrainte de sac-à-dos et une contrainte de sac-à-dos inversé prenant en compte les choix depuis le premier pas de temps.

Sur différentes petites instances du problème que nous avons générées, allant jusqu'à 4 pas de temps et 3 points de fonctionnement, l'utilisation du logiciel PORTA nous a permis d'obtenir l'enveloppe convexe du problème à partir d'une formulation. A partir de ces enveloppes convexes, nous avons identifié trois types d'inégalités facettes, les inégalités dites binaires avec des coefficients $\{0, 1\}$, les inégalités dites ternaires avec des coefficients $\{-1, 0, 1\}$ et les inégalités dites entières avec des coefficients entiers positifs.

Une grande partie des inégalités facettes correspond aux inégalités binaires, que nous avons appelées inégalités de Couverture (Inversée) Ordonnée (noté C(I)O). Certaines de ces inégalités ont été générées par un arrondi de Chvàtal-Gomory. Comme le **HUC** est équivalent à plusieurs sac-à-dos imbriqués avec des contraintes de précédence, nous avons comparé les inégalités C(I)O aux inégalités de couverture du sac-à-dos introduites par Balas [1]. Nous avons prouvé que les inégalités C(I)O sont au moins aussi fortes que les inégalités de Balas. Pour renforcer les inégalités C(I)O, nous avons mis au point un processus d'extension pour nos inégalités. Cette extension consiste à utiliser un algorithme de programmation dynamique pour trouver un plus court chemin dans un graphe. Le graphe construit a un sommet par variable de l'inégalité C(I)O à étendre. En trouvant ce plus court chemin, on arrive à déterminer des variables à ajouter à l'inégalité dans le sens du renforcement. Les inégalités C(I)O, avec ou sans le procédé d'extension, sont prouvées comme valides.

Les inégalités ternaires sont au cœur du **HUC**. En effet, elles proviennent du couplage d'inégalités de sac-à-dos, de sac-à-dos inversé et d'inégalités de précédence. Leur génération repose sur les inégalités C(I)O que nous avons obtenues précédemment. A partir des inégalités C(I)O et des contraintes de précédence, on peut alors distinguer qu'il existe des incompatibilités entre certains points de fonctionnement. Ces incompatibilités nous ont permis de construire les inégalités ternaires, que nous avons appelé inégalités de Convolution de Couverture (Inversée) Ordonnée (noté CC(I)O). De plus, nous avons prouvé la validité des inégalités CC(I)O.

En ce qui concerne les inégalités entières, nous avons pu obtenir un début de procédure pour les générer. L'idée est que lorsqu'on turbine sur un point de fonctionnement avec un grand débit, il est possible qu'on ne puisse plus turbiner sur certains points de fonctionnement avec un petit débit pour respecter les contraintes de volume. Les inégalités entières modélisent alors un choix entre turbiner sur un nombre réduit de points de fonctionnement avec des débits élevés, ou sur un plus grand nombre de points de fonctionnement avec des débits faibles.

Nous avons généralisé les inégalités C(I)O et CC(I)O pour des instances avec n'importe quel nombre de pas de temps et de points de fonctionnement Cependant la procédure pour les inégalités entières est une prémisse, il reste à la généraliser.

Parmi les différents exemples d'enveloppes convexes générés par PORTA, les inégalités C(I)O avec l'extension et les inégalités CC(I)O permettent de retrouver toutes les inégalités binaires et ternaires. Nous obtenons aussi certaines des inégalités entières avec la prémisse de procédure que nous avons obtenue.

Nous avons mis au point un prétraitement, qui consiste à modifier, dans le sens du renforcement, la borne des inégalités de sac-à-dos et de sac-à-dos inversé du problème. Cette modification est faite de manière à toujours préserver l'ensemble des solutions réalisables. Avec ce prétraitement, on peut alors éliminer des solutions non-réalisables avant de commencer le développement de l'arbre de recherche.

Pour finir, nous avons pu utiliser les inégalités C(I)O et CC(I)O et le prétraitement au sein d'un Branch and Cut, pour résoudre des instances d'EDF. Des tests ont été faits en comparant le développement de l'arbre de recherche selon l'utilisation de notre prétraitement ou celui de CPLEX, et de nos inégalités ou celles de CPLEX. Ces résultats expérimentaux mettent en avant l'effet positif de l'utilisation de notre prétraitement et de nos inégalités sur la taille de l'arbre de recherche. Ces résultats nous encouragent à essayer d'adapter nos inégalités C(I)O et CC(I)O pour un **HUC** avec une topologie différente, ou à les étendre au cas du **HUC** à plusieurs usines. De plus, il existe d'autres problèmes avec un couplage de contraintes similaire à celui du **HUC**. On pourrait alors chercher à généraliser les inégalités C(I)O et CC(I)O pour que ces inégalités soient pertinentes pour une famille plus large de problèmes.

Références

[1] Egon Balas. “Facets of the knapsack polytope”. In : *Mathematical programming* 8.1 (1975), p. 146-164.