

Résolution de problèmes de Lot-Sizing avec GLPK

Cours M06

Devoir à faire par groupe de 2 ou 3 et à rendre pour le 1 février 2012

Le problème de Lot-Sizing est un problème fondamental en planification de la production qui est particulièrement difficile pour les solveurs de programmation linéaire en nombres entiers. Il existe de nombreuses variantes de ce problème plus ou moins difficiles à résoudre. Dans ce devoir nous allons essayer résoudre des instances de grandes tailles de la variante la plus simple de ce problème : le problème de lot-sizing sans contrainte de capacités (notez que les méthodes utilisées sont typiquement celles utilisées pour attaquer les variantes plus compliquées).

Ce problème peut être résolu en temps polynomial par la programmation dynamique, mais nous essaierons ici de le résoudre efficacement par la programmation linéaire en nombre entiers.

1 Modélisation du problème

Une entreprise veut déterminer sa production pour les N prochaines périodes de temps. A chaque période de temps l'entreprise peut décider de produire ou non et peut stocker une quantité illimitée. Si l'entreprise produit à la période de temps t , elle doit payer un coût fixe f_t . Par ailleurs, pour chaque unité produite à la période t , elle doit payer un coût c_t et pour chaque unité stockée elle doit payer un coût h_t . Pour chaque période de temps elle dispose de prévisions de ventes données par son carnet de commandes d_t . L'entreprise veut planifier la production et le nombre d'unités stockées à chaque période de manière à minimiser le coût total donné par la somme des coûts fixes, des coûts variables (coûts unitaires multipliés par le nombres d'unités produites), et des coûts de stockage.

6 instances du problème vous sont fournies ayant respectivement 10, 20, 50, 100, 1000 et 2000 périodes de temps dans les fichiers `ULS_10.dat`, `ULS_20.dat`, `ULS_50.dat`, `ULS_100.dat`, `ULS_1000.dat` et `ULS_2000.dat`. Les valeurs optimales des problèmes à 10 et 20 périodes sont connues et sont de 107 et 213 respectivement.

Question 1.a Rappelez et justifiez la formulation du problème vue en cours. Utiliser des variables x_t pour les taux de productions, y_t pour les coûts fixes et s_t pour les montants stockés.

Question 1.b Écrivez ce modèle à l'aide des bibliothèques C de glpk. Vous pourrez utiliser les fonctions C et les structures fournies pour lire une instance du problème dans les fichiers : `uls.h`, `uls.c`, `uls_read.c` (ne tenez pas compte des capacités dans `uls.h` et `uls_read.c`). Quelles instances pouvez vous résoudre en moins de deux minutes ? Donnez les valeurs optimales et les temps de résolutions des instances résolues.

2 Génération automatique de coupes

Pour accélérer la résolution des problèmes nous allons étudier des méthodes de génération de coupes. Si on ne produit pas à la période t il faut nécessairement que la quantité stockée à la période $t - 1$ soit supérieure à la demande en période t . Ceci peut s'exprimer par l'inégalité :

$$s_{t-1} \geq d_t(1 - y_t) \quad (1)$$

Nous allons ajouter ces inégalités dynamiquement au problème en utilisant les fonctions callback de glpk, un prototype de fonction callback est donné dans le fichier `uls_cut_callback.c`.

Question 2.a Écrivez un algorithme de séparation pour les inégalités (1) et ajoutez le au code de résolution.

Question 2.b Combien d'instances pouvez vous résoudre en deux minutes ? Donnez les valeurs optimales et les temps de résolutions des instances résolues.

L'inégalité (1) prend seulement en compte la production sur une période. En supposant que l'on ne produit pas sur plusieurs période consécutives on peut généraliser cette inégalité.

Question 2.c Supposez que l'on ne produit pas entre les périodes t et t' , pour $1 \leq t \leq t' \leq N$. Quelle quantité doit nécessairement être en stock à la période t ? En déduire une inégalité valide. Combien d'inégalités de ce type peuvent potentiellement être ajoutées au problème ?

Question 2.d Écrivez et implémentez un algorithme de séparation pour ces inégalités. Étant donné le grand nombre de ces inégalités, il est souhaitable de générer seulement l'inégalité la plus violée pour chaque période de temps t .

Question 2.e Combien d'instances pouvez-vous résoudre en deux minutes ? Donnez les valeurs optimales et les temps de résolutions des instances résolues.