

Langages recommandés : Java ou Python

1 Prise en main de `lp_solve`

Dans tout le TP, nous allons utiliser le programme `lp_solve`, qui est un solveur de programmes linéaires.

Voilà un exemple de programme linéaire sous le format de base de `lp_solve` :

```
max: 2 x1 + x2;  
x1 + 2 x2 <= 20;  
3.2 x1 + x2 <= 15;
```

Pour utiliser le programme, il suffit de taper la commande

```
> lp_solve fichier.lp
```

où `fichier.lp` contient l'exemple ci-dessus. Le résultat est affiché comme ceci :

```
Value of objective function: 12.77777778  
  
Actual values of the variables:  
x1                      1.85185  
x2                      9.07407
```

Si on veut se restreindre à des `x1` et `x2` entiers il suffit d'ajouter la ligne suivante :

```
int x1,x2;
```

► **Question 1** ◀ Vérifiez que tout fonctionne correctement sur l'exemple ci-dessus. Quelles sont les valeurs optimales de `x1` et `x2` si on se restreint à des nombres entiers ?

► **Question 2** ◀ Vérifiez que vous trouvez bien la même solution à l'exercice 1 de la feuille de TD 2 avec le solveur que celle que l'on a déterminée graphiquement.

2 Un problème générique

On considère un problème plus général où l'on dispose de n types de marchandises $M_1, \dots, M_n$. Il y a k types de ressources utilisées pour confectionner les marchandises, $R_1, \dots, R_k$. Chaque marchandise i rapporte un bénéfice de B_i pour chaque unité vendue.

Les différentes statistiques associées aux marchandises sont donnée dans un fichier texte formaté de la façon suivante :

M8 16 29 15 22 27 15 442
M9 28 12 25 15 19 24 436

La première colonne est le nom de la marchandise (ex: M8). La dernière colonne est le bénéfice réalisé par unité de cette marchandise qui est vendue (ex: 442). Les autres sont les besoins en ressources, ici il y a donc $k = 6$ ressources différentes. On peut produire des portions d'unité de marchandises, et on considère que tout sera vendu.

► **Question 3** ◀ En utilisant le fichier que vous pouvez récupérer « [ICI](#) », écrivez un programme qui prend en entrée un tel fichier et écrit sur la sortie standard le programme linéaire associé, avec les contraintes que chacune des k ressources est limité à 1000 unités.

- ▷ Faites tourner le solveur dessus, quelle est le bénéfice optimal ?
- ▷ Combien de marchandises différentes faut-il produire ?
- ▷ Combien de temps a mis le solveur pour calculer la solution optimale ?

► **Question 4** ◀ Rajoutez la contrainte qu'on ne peut plus faire de portions de marchandises, il faut en produire un nombre entier. Mêmes questions qu'à la question 3.

► **Question 5** ◀ Essayez avec le fichier (beaucoup) plus volumineux que vous pouvez trouver « [ICI](#) ». Chaque ressource est maintenant limitée à 100 000 unités.

3 Un problème de découpe

► **Question 6** ◀ Reprendre l'exercice de la feuille de TD 1 sur la découpe de barres de métal et trouvez la solution optimale avec le solveur.

► **Question 7** ◀ On considère maintenant que les tiges de bases font 5,0m et qu'on veut les découper en barres de longueurs 2,0m, 1,2m, 1m et 0,5m. Ecrire un programme qui génère tous les découpages maximaux possibles. Adaptez le pour générer le programme linéaire

correspondant à une commande qui minimise le nombre de tiges utilisées pour produire 60 barres de 2,0m, 100 de 1,2m, 150 de 1m et 350 de 0,5m.

4 Le marchand de glace

► **Question 8** ◀ Reprendre le problème du marchand de glace vu en cours. Trouver la solution optimale avec un coût de stockage de 20 € par tonne et un coût de réajustement de 50 € par tonne pour les demandes suivantes :

jan	350
fev	320
mar	440
avr	630
mai	630
jun	550
jui	680
aou	660
sep	350
oct	420
nov	380
dec	620

► **Question 9** ◀ Observez le changement de stratégie à adopter quand le coût de stockage est nul.