RCP104- Optimisation en Informatique

TP- Problème pmédian

Par Sourour Elloumi

Objectif : A travers le problème pmédian, comprendre la notion de bonne formulation en Programmation Linéaire en Nombres Entiers

Ce TP utilise le mini-site :

http://cedric.cnam.fr/~elloumi/RCP104/TPlocalisation/

Le problème pmédian:

Etant donnés :

- N : nombre de sites clients
- M : nombre de sites potentiels pour des dépôts
- P : nombre de dépôts à ouvrir
- Dij : distance entre le site client i et le site dépôt j

On veut ouvrir P dépôts et affecter chaque client au dépôt ouvert qui lui est le plus proche, de façon à minimiser la somme des distances entre les clients et les dépôts auxquels ils sont affectés.

Question 1 –

Télécharger le fichier `pmedianF1.mos`. Ce fichier contient une modélisation du problème pmédian. Laquelle ?

Le programme `pmedianF1.mos` lit les données d'une instance du problème dans le fichier `pmedian.dat`

Remarque :

Sous linux, en utilisant le shell bash, on peut exécuter :

```
mosel -c "exec pmedianF1.mos"
```

Question 2 –

Télécharger le fichier `pmedianF2.mos`. Ce fichier contient une modélisation du problème pmédian. Laquelle ?

Quelles différences remarquez-vous dans l'exécution sur le même fichier `pmedian.dat` ?

Question 3 –

Exécuter `pmedianF1.mos` et `pmedianF2.mos` sur les autres instances fournies sur le mini-site. Quelles observations pouvez-vous faire ?

Fichier pmedianF1.mos

```
model pmedian
 uses "mmxprs"               ! utilise le solveur Xpress-Optimizer

!déclaration des paramètres, ici le nom du fichier de donnees
 parameters
  F="pmedian.dat";
 end-parameters

!#####################################################
declarations
  n : integer; !nombre de clients
  m : integer; !nombre de depots potentiels
  p : integer; !nombre de depots à ouvrir
end-declarations
!#####################################################
!Lecture des données
 fopen (F, F_INPUT)
 write ("---Lecture du fichier ", F, " ...")
 fskipline ("#\n") ! on saute les lignes commençant pat # et
                   ! les lignes vides
 readln (n, m, p)
 writeln ("n=",n, " m= ", m, " p= ",p)

 declarations
  ! les donnees
  D : array (1..n, 1..m) of real;  ! distances
 end-declarations

! lecture de D
 forall (i in 1..n)
     forall (j in 1..m) read (D(i,j))
     readln;
! fermeture du fichier
 fclose (F_INPUT)

!#####################################################
 declarations
 ! les variables
  x: array (1..n, 1..m) of  mpvar              ! variables
  y: array (1..m) of  mpvar              ! variables
 end-declarations

!#####################################################
! le modèle
! fonction objectif
cout:=  sum (i in 1..n, j in 1..m) D(i,j)*x(i,j)
! les contraintes
sum (j in 1..m) y(j) = p
forall(i in 1..n) affect(i):=sum (j in 1..m)x(i,j) = 1

forall(j in 1..m) sum (i in 1..n)x(i,j) <= n*y(j) ! contrainte faible

forall (i in 1..n,j in 1..m) x(i,j) is_binary
forall (j in 1..m) y(j) is_binary
!#####################################################
! résolution du problème
!pour rendre xpress plus bavard
 setparam("XPRS_VERBOSE",true);
 ! pour le rendre encore plus bavard en resolution mip
 !setparam("XPRS_MIPLOG",3)

! résolution du problème
minimize (cout)

!#####################################################
!Affichages
 writeln("Solution: ", getobjval) ! affichage de la valeur de cout
 forall (j in 1..m) if  (getsol(y(j))<> 0) then
     writeln("valeur de y(",j,")= ", getsol(y(j)));
 end-if
 forall (i in 1..n,j in 1..m) if  (getsol(x(i,j))<> 0) then
   writeln("valeur de x(",i,",",j,")= ",getsol(x(i,j)));
 end-if
end-model
```