Probléme de répartition d'un marché

Valentin Ryckewaert Marion Noirbent

9 novembre 2015

Table des matières

Chapitre 1

Introduction

1.1 Organisation de l'environnement

Notre environnement se compose des fichiers suivant :

- Le fichier "repartition_marche" est un script qui automatise la configuration et la résolution par Ampl.
- Le fichier "repartition_marche.data" fourni, que nous avons utilisé sans le modifiier.
- Le fichier "repartition_marche_realisable.model" qui modélise la recherche d'une solution réalisable.
- Le fichier "repartition_marche_somme.model" qui modélise la recherche d'une solution minimisant la somme des variations.
- Le fichier "repartition_marche_variation.model" qui modélise la recherche d'une solution minimisant la variation maximale.

1.2 Utilisation du script

Le script est chargé dans Ampl par la commande : model repartition_marche

Sa première action est un reset, afin de s'assurer que le script s'exectute dans un environnement propre.
Ensuite il configure le solver sur Gurobi, un détaillant ne pouvant pas appartenir à 0.8 à une division.
Puis il charge le fichier de data et le premier modèle (celui de recherche d'une solution réalisable) et lance la résolution.
Ensuite il fera un reset du model, rechargera le modéle corrsepondant et lancera sa résolution pour la minimisation la somme des variations puis pour la minimisation de la variation maximale.

Chapitre 2

Recherche d'une solution réalisable

2.1 Fonctionnement

N'ayant que deux division à affecter, le problème de se rapprocher du rapport 40/60 est équivalent à se rappocher de 60% du total pour la 2^{e} division.
Nous avons donc fait le choix de leur attribuer la valeursd'un booléen (0 pour la 1re division, 1 pour la 2^{e}). Nous multiplions alors les valeurs dans les domaines recherchés (nombre de points de vente, ...) par la valeur du booléen correspondant, puis nous en faisont la somme que nous divisons par le total dans ce domaine, le nombre obtenu correspond au rapport pour la division 2 uniquement.

Nous avons fait des totaux dans les domaines recherchés (nombre de points de vente, ...) des paramètres calculé afin de rendre leur utilisation plus lisible.
La recherche d'une solution réalisable ne necessite ici pas de minimisation de l'objectif.
Bien qu'il soit possible de faire une contrainte contenant deux inégalités, nous avons préféré distinguer les contraintes portant sur le minimum de celles portant sur le maximum.

2.2 Résultat

Réponse donnée par Ampl lors de la recherche d'une solution réalisable :

Gurobi 5.0.2 : optimal solution ; objective 0
div_du_detaillant [*] :=
M1 0 M12 1 M15 1 M18 0 M20 1 M23 1 M5 1 M8 1
M10 1 M13 1 M16 0 M19 1 M21 0 M3 0 M6 1 M9 0
M11 1 M14 1 M17 1 M2 0 M22 0 M4 0 M7 1
;

Chapitre 3

Recherche avec minimisation de la somme des variations

3.1 Fonctionnement

Nous avons repris le modèle de la solution réalisable, nous n'y avons rien soustrait.
Nous avons ajouté un paramètre calculé dans chaque dommaine égale à 60% du total, il correspond à la valeur vers laquelle le rapport de la division 2 doit tendre.
Nous avons ajouté une variable auxiliaire pour représenter la somme des variations. Nous lui affectons sa valeur par l'utilisation d'une contrainte, l'objectif est la minimisation de cette contrainte.

3.2 Problémes rencontrés

Nous nous somme basés sur le support de cours (page 28, 2.6.1) pour définir ce fonctionnement mais nous ne sommes pas parvenus à déterminer ce que devait contenir les membres à additionner dans la contrainte.

Chapitre 4

Recherche avec minimisation de la variation maximale

4.1 Fonctionnement

Nous avons repris le modèle de la solution réalisable, nous n'y avons rien soustrait.
Nous avons ajouté une variable auxiliaire pour représenter le maximum, ainsi que deux contraintes pour chaque domaine recherché pour lui attibuer sa valeur comme étant supérieure ou égale à chacune des variations (les deux contraintes permettent de l'obtenir en valeur absolue des variations).
L'objectif étant de miniser cette contrainte, sa valeur viendra égaliser le maximum des variations et tendra à le minimiser.

4.2 Problémes rencontrés

Nous avons manqué de temps pour implémenter ce fonctionnement.

Annexe A

repartition_marche

```
reset ;

option solver gurobi ;

/* Question 1 : recherche d'une solution réalisable */
model repartition_marche realisable.model
data repartition_marche.data
solve ;

display div_du_detaillant ;

/* Question 2 : optimisation */

/* en somme des écart */
reset model ;
model repartition_marche_somme.model
solve ;

/* en variation maximale */
reset model ;
model repartition_marche_variation.model
solve ;
```

Annexe B

repartition_marche_realisable.model

```
set DETAILLANTS;
set REGIONS;
set CATEGORIES;

/* ---------------- */
/* Paramètres caractérisant un détaillant */

/* ******* */
/* La région dans laquelle il est implanté */
param region{DETAILLANTS} symbolic in REGIONS;

/* Son nombre de vente d'huile */
param huile{DETAILLANTS} >= 0;

/* Son nombre de points de vente */
param nb_pts_vente{DETAILLANTS} >= 0;

/* Son nombre de vente de spiriteux */
param spiritueux{DETAILLANTS} >= 0;

/* Sa catégorie */
param categorie{DETAILLANTS} symbolic in CATEGORIES;

/* ---------------- */
/* Paramètre calculés */

/* Nombre total de points de ventes */
param pts_vente_totaux :=
sum {m in DETAILLANTS}
nb_pts_vente[m];

/* Nombre total de vente de spiriteux */
param spiritueux_totaux :=
sum {m in DETAILLANTS}
spiritueux[m];

/* Nombre de points d'huile dans chaque région */
param huile_par_region {r in REGIONS} :=
sum {m in DETAILLANTS : region[m]=r}
```

```
huile[m] ;

/* Nombre de détaillants de chaque catégorie */
param nb_par_categorie{c in CATEGORIES} :=
sum {m in DETAILLANTS : categorie[m]=c}
1 ;

/* ---------------- */
/* Répartition recherchée */

/* Division à laquelle rattaché chaque détaillant
div_du_detaillant[m] vaut 0 si m appartient à la division D1, et vaut 1 sinon */
var div_du_detaillant {m in DETAILLANTS} binary ;

/* ---------------- */
/* Rapprochement de l'objectif */

/* Cas où on cherche une solution réalisable -> seules les contraintes importent */
minimize res : 0 ;

/* ---------------- */
/* Contraintes */

/* ******* */
/* Contraintes sur le rapport 40/60 dans chaque division (+/- 5%) */

/* Le nombre total de points de vente */
subject to rapport_pts_vente_min :
sum {m in DETAILLANTS}
div_du_detaillant[m]*nb_pts_vente[m]*100 / pts_vente_totaux >= 55 ;

subject to rapport_pts_vente_max :
sum {m in DETAILLANTS}
div_du_detaillant[m]*nb_pts_vente[m]*100 / pts_vente_totaux <= 65 ;

/* Le nombre total de vente de spiriteux */
subject to rapport_spiritueux_min :
sum {m in DETAILLANTS}
div_du_detaillant[m]*spiritueux[m]*100 / spiritueux_totaux >= 55 ;

subject to rapport_spiritueux_max :
sum {m in DETAILLANTS}
div_du_detaillant[m]*spiritueux[m]*100 / spiritueux_totaux <= 65 ;

/* Le nombre de points d'huile dans chaque région */
subject to rapport_huile_region_min {r in REGIONS} :
sum {m in DETAILLANTS : region[m]=r}
div_du_detaillant[m]*huile[m]*100 / huile_par_region[r] >= 55 ;

subject to rapport_huile_region_max {r in REGIONS} :
sum {m in DETAILLANTS : region[m]=r}
div_du_detaillant[m]*huile[m]*100 / huile_par_region[r] <= 65 ;
```

```
/* Le nombre de détaillants de chaque catégorie */
subject to rapport_categorie_min {c in CATEGORIES} :
sum {m in DETAILLANTS : categorie[m]=c}
div_du_detaillant[m]*100 / nb_par_categorie[c] >= 55;

subject to rapport_categorie_max {c in CATEGORIES} :
sum {m in DETAILLANTS : categorie[m]=c}
div_du_detaillant[m]*100 / nb_par_categorie[c] <= 65;
```