Leçon 12

INTRODUCTION A LA PROGRAMMATION LINEAIRE EN NOMBRES ENTIERS

Nous abordons dans cette leçon une nouvelle catégorie de problèmes : les problèmes de programmation linéaire en nombres entiers.
Nous nous intéresserons essentiellement à la modélisation de ces problèmes et non à leur résolution et présenterons quelques-uns des très nombreux problèmes susceptibles d'être modélisés par un problème de programmation linéaire en nombres entiers.
Les exemples présentés, bien que simplifiés, font référence à des situations réelles.

I Le problème du sac à dos : Knapsack-Problem

Un alpiniste se préparant à partir en expédition est confronté au douloureux problème de savoir ce qu'il doit mettre dans son sac.
Chaque objet présente pour lui une utilité plus ou moins importante et le poids total du sac est limité.
La modélisation de ce problème est a priori très simple.
On connaît pour chaque objet son poids p_j, ainsi que l'utilité c_j que l'on peut en retirer. On connaît aussi le poids maximum P du sac.

Les **décisions** concernent le nombre de chacun des objets à mettre dans le sac. Elles sont représentées par des variables x_j : $x_j \geq 0$

La seule **contrainte** porte sur le poids des objets.

$$\sum_{j=1}^{n} p_j x_j \leq P$$

L'objectif concerne la maximisation de l'utilité totale. On fait l'hypothèse que pour chaque type d'objets, l'utilité est proportionnelle au nombre d'objets.

$$\text{Max} \sum_{j=1}^{n} c_j x_j$$

Pour le moment, on a, a priori, un problème très simple de programmation linéaire :

$$\left| \begin{array}{l} \text{Max} \sum_{j=1}^{n} c_j x_j \\ \sum_{j=1}^{n} p_j x_j \leq P \\ x_j \geq 0 \quad j = 1, \ldots, n \end{array} \right.$$

Cependant, si on doit tenir compte du fait que les objets sont non fractionnables et donc que, pour chacun d'eux, on ne peut en sélectionner qu'un nombre entier qui peut être pour certains limité à une unité ; il faut ajouter que **les variables ne peuvent pendre que des valeurs entières, voire même uniquement la valeur 0 ou 1**.

Ce problème est l'exemple le plus simple de problème de programmation linéaire en nombres entiers.
Le challenge est que ce type de problème fait partie des problèmes difficiles, ceux pour lesquels le temps de calcul croit de manière exponentielle avec la taille du problème.

II Définition d'un problème de programmation linéaire en nombres entiers

La définition est immédiate. Un problème de programmation linéaire en nombres entiers (PLNE) est un problème de programmation linéaire (PL) avec tout ou partie des variables qui doivent être entières, voire restreintes à 0 et 1 comme valeur.
On dit que les variables sont soumises à des **contraintes d'intégrité**.
Un problème de programmation linéaire classique (PL) sera, a contrario, dit "en variables continues".
Dans certains problèmes, les variables de décision sont, par nature, entières, mais nous allons voir que, dans de nombreux cas la modélisation nécessite l'introduction de **variables booléennes,** c'est-à-dire qui ne peuvent prendre que la valeur 0 ou 1.

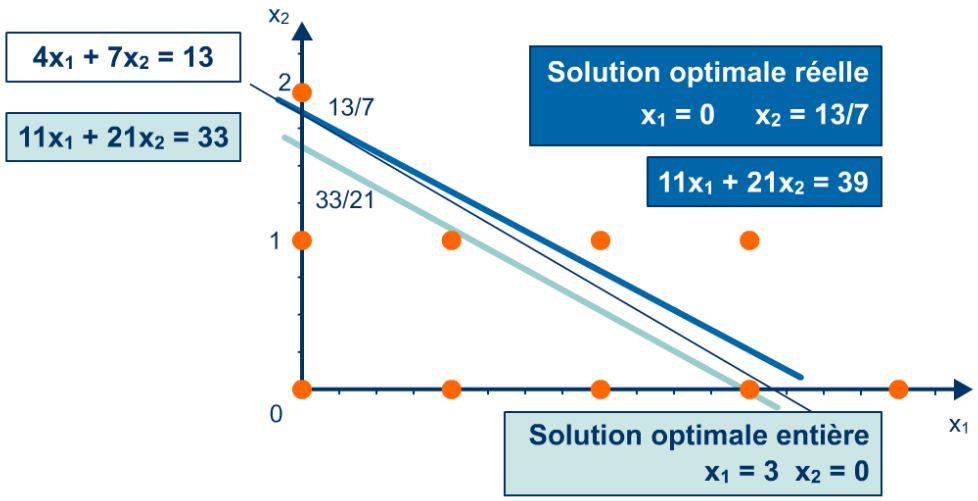
Remarques sur la résolution d'un PLNE
Les problèmes de PLNE font partie des problèmes que nous avons déjà rencontrés, ceux pour lesquels on ne dispose pas d'algorithmes dont le temps de calcul croit de manière polynomiale avec la taille du problème.
On pourrait, par exemple, modéliser le problème du voyageur de commerce, le problème de coloration, voire certains problèmes d'ordonnancement, par un problème de PLNE.

Lorsque la taille du problème le permet, on peut le résoudre avec des méthodes de résolution basées sur une exploration intelligente du domaine des solutions.
Il arrive aussi que, dans certains (rares) cas, la solution obtenue sans tenir compte des contraintes d'intégrité, par exemple avec l'algorithme du simplexe, soit a posteriori entière ; le problème est alors résolu.

Comme on l'a vu, on dispose de bons algorithmes pour résoudre les problèmes de programmation linéaire classiques, on peut se demander s'il ne serait pas possible de résoudre, sans tenir compte des contraintes d'intégrité, puis arrondir la solution trouvée à l'entier le plus proche, ceci n'ayant évidemment pas de sens pour une variable booléenne.

Considérons l'exemple suivant :

$$\begin{aligned} &\text{Max } 11\,x_1 + 21\,x_2 \\ &4\,x_1 + 7\,x_2 \leq 13 \\ &x_1, x_2 \in N \end{aligned}$$



Sans la contrainte d'intégrité, la solution optimale est au point $x_1 = 0$ et $x_2 = 13/7$
Si on ne considère que les points à coordonnées entières, l'optimum est atteint au point $x_1 = 3$ et $x_2 = 0$, qui ne peut évidemment pas être obtenu par arrondi de la solution réelle.
De plus la valeur optimale de l'objectif n'est que de 33 alors qu'elle est égale à 39 dans le cas des variables réelles.

III Un problème de localisation

L'objectif de cette leçon est de présenter quelques-uns des très nombreux problèmes qui se modélisent par un problème de programmation linéaire en nombres entiers. Nous commençons par le problème dit de localisation.
Afin d'approvisionner ces principaux centres de distribution, la firme Nacege a décidé d'implanter plusieurs usines. Pour cela, elle a identifié plusieurs sites susceptibles de convenir parmi lesquels il faudra en choisir au maximum un certain nombre.
On connaît le coût d'implantation des usines en fonction du site.
Le coût de raccordement d'un centre à une usine est également connu.
Le problème est de sélectionner les sites à retenir et, pour chaque centre, l'usine qui le desservira de manière à minimiser le coût total.

Afin de modéliser ce problème, on dispose des **données** sur les sites et les centres :
n sites j = 1,..., n p centres i = 1,...,p
f_j = coût d'implantation d'une usine sur le site j.
c_{ij} = coût de raccordement du centre de distribution i au site j
K = nombre maximum d'usines.

Pour chaque site, il s'agit de savoir si on y construit ou non une usine. Cette **décision** peut être représentée par une variable booléenne qui aura l'interprétation suivante : si elle vaut 1 c'est "oui" ; si elle vaut 0, c'est "non".

$$y_j = \begin{cases} 1 \text{ si le site j est retenu} \\ 0 \text{ sinon} \end{cases}$$

Pour le raccordement des centres aux usines, la décision est de même nature. Pour chacun des centres i et pour chaque usine située sur le site j, il s'agit de savoir si le centre est, ou non, raccordé à l'usine. Une variable booléenne x_{ij} permet de représenter cette décision.

$$x_{ij} = \begin{cases} 1 \text{ si le centre i est affecté au site j} \\ 0 \text{ sinon} \end{cases}$$

Les contraintes
On ne veut pas plus de K usines construites.
Cette contrainte se traduit par : au maximum K variables y_j peuvent prendre la valeur 1 :

$$\sum_{j=1}^{n} y_j \leq K$$

Chaque centre est affecté à une seule usine c'est-à-dire que, pour chaque i, une seule variable x_{ij} vaut 1 :

$$\sum_{j=1}^{n} x_{ij} = 1$$

Un centre ne peut être raccordé qu'à une usine construite ; si x_{ij} vaut 1 alors y_j doit valoir 1, ce qui s'exprime encore par : si y_j vaut 0 alors x_{ij} vaut 0.
Cette contrainte peut être traduite par : $x_{ij} \leq y_j$

L'objectif
2 types de coût sont pris en compte.
Pour chaque usine, son coût d'implantation sur le site j vaut f_j si elle est construite, donc si y_j est égal à 1, et 0 sinon.
Il en est de même pour le coût de raccordement d'un centre i à une usine. Il vaut soit c_{ij} si x_{ij} vaut 1, soit 0 si x_{ij} vaut 0.
D'où l'expression de l'objectif à minimiser :

$$\sum_{j=1}^{n} f_j x_j + \sum_{i=1}^{p} \sum_{j=1}^{n} c_{ij} x_{ij}$$

Le problème de localisation est donc modélisé par le PLNE suivant :

$$\begin{cases} \text{Min } \sum_{j=1}^{n} f_j x_j + \sum_{i=1}^{p} \sum_{j=1}^{n} c_{ij} x_{ij} \\ \sum_{j=1}^{n} y_j \leq K \\ \sum_{j=1}^{n} x_{ij} = 1 \quad i = 1, \ldots, p \\ x_{ij} \leq y_j \quad i = 1, \ldots, p \; j = 1, \ldots, n \\ y_j \in \{0, 1\} \quad j = 1, \ldots, n \\ x_{ij} \in \{0, 1\} \quad i = 1, \ldots, p \; j = 1, \ldots, n \end{cases}$$

IV Prise en compte de contraintes logiques

Certaines contraintes logiques peuvent être prises en compte dans les modèles de programmation linéaire en nombres entiers.
Par exemple, dans le problème de localisation précédent, on peut introduire les hypothèses supplémentaires suivantes concernant, par exemple, les sites numéro 1 et 2.

Le site 1 ne peut être retenu en même temps que le site 2.
La traduction à partir des variables associées au site est :
x_1 et x_2 ne peuvent être simultanément égales à 1 ; ce qui se traduit par l'inégalité linéaire suivante :

$$x_1 + x_2 \leq 1$$

Au moins un des 2 sites 1 et 2 doit être retenu.
Une au moins des deux variables x_1 et x_2 vaut 1 :

$$x_1 + x_2 \geq 1$$

Un et un seul des sites1 et 2 doit être retenu.
Une et une seule des deux variables x_1 et x_2 vaut 1 :

$$x_1 + x_2 = 1$$

Le site 2 ne peut être retenu que si le site 1 l'a été.
Si x_2 vaut 1, alors x_1 vaut 1 :

$$x_2 \leq x_1$$

On peut aussi par introduction de variables booléennes traduire des contraintes comme :
Une variable ne peut prendre que certaines valeurs.
Par exemple si une variable x ne peut prendre que la valeur 0, 100 ou 1000, on peut, en introduisant 2 variables booléennes y_1 et y_2, écrire :

$x = 100\, y_1 + 1000\, y_2$ avec $y_1 + y_2 \leq 1$

Si $y_1 = y_2 = 0$, on aura $x = 0$
Si $y_1 = 1$ et $y_2 = 0$, on aura $x = 100$

Si $y_1 = 0$ et $y_2 = 1$, on aura $x = 1000$

Dans le cas où une variable est telle que :
soit x = 0 soit x ≥ Q,
situation que l'on rencontre, par exemple en production, lorsqu'on ne veut produire que si la quantité est suffisamment importante, on introduit aussi une variable booléenne y, ainsi qu'un nombre M représentant une majoration des valeurs que peut prendre x. La condition x = 0 ou x ≥ Q est traduite par la double inégalité :

$Q\,y \leq x \leq M\,y$

Si y = 0 alors x = 0
Si y = 1 alors Q ≤ x≤ M

V Un problème de recouvrement (Covering Problem)

Le problème suivant est un exemple de problème dit de recouvrement.
Il concerne ici l'implantation de casernes de pompiers dans une ville.
La ville est découpée en quartiers. Par ailleurs, un certain nombre de localisations possibles pour les casernes ont été identifiées ainsi que le coût d'implantation correspondant.
Le problème est de localiser les casernes de manière à ce que chaque quartier soit à moins de 5 minutes au moins d'une caserne. Et ceci au moindre coût !
Pour chaque quartier, on évalue donc s'il est ou non, à moins de 5 minutes de chacun des différents sites.
Ce problème a été posé dans la ville de Denver avec 246 quartiers et 112 sites possibles. Il peut aussi bien concerner la localisation de sirènes que d'émetteurs de télévisions ou bien d'autres choses du moment qu'il s'agit de couvrir des zones, d'où son nom.

Exemple avec 5 quartiers et 4 sites
Le tableau suivant indique pour chaque quartier les sites situés à moins de 5 minutes.

	Sites			
Quartiers ↓	1	2	3	4
1	OK		OK	
2		OK	OK	OK
3	OK	OK		
4		OK		OK
5	OK			OK

On dispose aussi des différents coûts d'implantation

Coût d'implantation			
1	2	3	4
500	350	580	400

La seule **décision** concerne le choix des casernes à construire.
A chaque site, on associe une variable booléenne x_j.

$$x_j = \begin{cases} 1 \text{ si le site j est retenu} \\ 0 \text{ sinon} \end{cases}$$

Pour les **contraintes**, il s'agit d'exprimer que chaque quartier sera bien couvert : parmi les sites qui intéressent le quartier i, il doit en avoir au moins 1 de retenu ($y_j = 1$)
Par exemple, pour le quartier 1, d'après les données, seuls les sites 1 et 3 conviennent. Il faut donc qu'une au moins des variables x_1 et x_3 soit égale à 1 d'où l'inégalité :

$x_1 + x_3 \geq 1$

On a aussi :
$x_2 + x_3 + x_4 \geq 1$
$x_1 + x_2 \geq 1$
$x_2 + x_4 \geq 1$
$x_1 + x_4 \geq 1$

La représentation de **l'objectif** est immédiate ; le coût total d'implantation est égal à :
$500\ x_1 + 350\ x_2 + 580\ x_3 + 400\ x_4$

D'où le problème de PLNE

$$\text{Min } (500\ x_1 + 350\ x_2 + 580\ x_3 + 400\ x_4)$$

$$\begin{aligned} x_1 \quad & + x_3 \quad \geq 1 \\ x_2 & + x_3 + x_4 \geq 1 \\ x_1 + x_2 & \quad \geq 1 \\ x_2 \quad & + x_4 \geq 1 \\ x_1 \quad & + x_4 \geq 1 \end{aligned}$$

$x_j = 0$ ou $1 \quad j = 1, \ldots, 4$

Dans le cas général, un problème de recouvrement est de la forme suivante :

$$\text{Min} \quad \sum_{j=1}^{n} c_j x_j$$

$$\sum_{j=1}^{n} a_{ij}\ x_j \geq 1$$

$x_j = 0$ ou $1 \quad j = 1, \ldots, n$

Les coefficients a_{ij} sont égaux à 0 ou à 1.

V Un problème de partitionnement (partitionning problem)

A partir d'un centre de distribution et avec des camions d'une capacité maximale donnée, on doit effectuer des tournées afin de livrer des clients.
On souhaite répartir les clients dans différentes tournées.

Le problème est de sélectionner parmi un ensemble de tournées possibles celles qui permettront d'effectuer toutes les livraisons au moindre coût.
Ce modèle, dit de partitionnement, permet de représenter bien d'autres types de problèmes, par exemple l'affectation de chauffeurs à des lignes d'autobus !

Un travail préalable a permis d'identifier un certain nombre de tournées possibles avec leur coût.
On suppose ici qu'on a 4 clients et 5 tournées.

Les données du problème sont les suivantes :

Tournées possibles

Clients ↓	*1*	*2*	*3*	*4*	*5*
1	*		*		
2		*	*	*	
3	*				*
4			*	*	*

	1	*2*	*3*	*4*	*5*
Coût	1000	800	1500	700	600

(Par exemple, la tournée 1 dessert les clients 1 et 3).

Ici encore les variables de **décision** sont booléennes et associées à chaque tournée possible.

$$x_j = \begin{cases} 1 \text{ si la tournée j est retenue} \\ 0 \text{ sinon} \end{cases}$$

Les **contraintes** traduisent que chaque client doit être livré par une et une seule tournée.
Pour le client 1, qui peut être desservi par la tournée 1 ou par la tournée 3, ceci se traduit par :
$x_1 + x_3 = 1$
On a des contraintes analogues pour les autres clients.
$x_2 + x_3 + x_4 = 1$
$x_1 + x_5 = 1$
$x_3 + x_4 + x_5 = 1$

La traduction de **l'objectif** ne présente aucune difficulté.
$1000\ x_1 + 800\ x_2 + 1500\ x_3 + 700\ x_4 + 600\ x_5$

d'où le problème :

$$\begin{aligned} &\text{Min } (1000\ x_1 + 800\ x_2 + 1500\ x_3 + 700\ x_4 + 600\ x_5 \\ &x_1 + x_3 = 1 \\ &x_2 + x_3 + x_4 = 1 \\ &x_1 + x_5 = 1 \\ &x_3 + x_4 + x_5 = 1 \\ &x_j = 0 \text{ ou } 1 \quad j = 1,\ldots,5 \end{aligned}$$

Les tournées qui seront finalement retenues, à l'issue de la résolution, permettent d'effectuer une partition des différents clients, d'où le nom de problème de partitionnement attribué à ce modèle, dont la forme générale est :

$$\begin{aligned} &\text{Min } \sum_{j=1}^{n} c_j x_j \\ &\sum_{j=1}^{n} a_{ij}\ x_j = 1 \\ &x_j = 0 \text{ ou } 1 \quad j = 1, \ldots, n \end{aligned}$$

Les coefficients a_{ij} sont égaux à 0 ou à 1.

VI Un problème d'affectation

Une grande compagnie aérienne a centralisé ses correspondances dans un hub.
Elle souhaite coupler les vols arrivant et partant de ce hub, afin qu'un maximum de passagers puisse continuer leur voyage sans changer d'avion.
On considère n vols arrivant au hub et n vols quittant le hub.
n_{ij} est le nombre de passagers arrivant par le vol numéro i (i = 1, …,n) et poursuivant sur le vol numéro j (j = 1, …,n).

Dans ce problème, le but est de déterminer, pour chaque vol arrivant, le vol partant avec lequel il sera couplé.
Etant donné le vol arrivant numéro i, il s'agit de savoir si le vol partant numéro j lui est ou non affecté, d'où l'introduction des variables de **décision** x_{ij}.

$$x_{ij} = \begin{cases} 1 \text{ si le vol arrivant i est couplé au vol partant j} \\ 0 \text{ sinon} \end{cases}$$

Il faut que chaque vol arrivant soit couplé avec un seul vol partant.
Pour le vol arrivant numéro i, il suffit de compter le nombre de variables x_{ij} qui valent 1.
Cette **contrainte** peut donc s'écrire :

$$\sum_{j=1}^{n} x_{ij} = 1 \quad i = 1, \ldots,n$$

Ceci ne suffit pas car plusieurs vols arrivant pourraient être couplés au même vol quittant le hub ; on ajoute donc les contraintes qui expriment que chaque vol quittant est couplé avec un seul vol arrivant :

$$\sum_{i=1}^{n} x_{ij} = 1 \quad j = 1, \ldots,n$$

L'objectif est de maximiser le nombre de passagers qui ne changeront pas d'avion.

$$\sum_{i=1}^{n}\sum_{j=1}^{n} n_{ij}x_{ij}$$

Ce problème est un problème d'affectation dont la forme générale est:

$$\left|\begin{array}{l} \text{Min (ou max)} \sum_{i=1}^{n}\sum_{j=1}^{n} n_{ij}x_{ij} \\ \sum_{j=1}^{n} x_{ij} = 1 \quad i = 1, \ldots,n \\ \sum_{i=1}^{n} x_{ij} = 1 \quad j = 1, \ldots,n \\ x_{ij} = 0 \text{ ou } 1 \quad i = 1, \ldots,n \quad j = 1, \ldots,n \end{array}\right.$$

Le modèle d'affectation sert à représenter des problèmes dans lesquels on dispose de deux ensembles de n éléments, tout élément de l'ensemble 1 devant être affecté à un et seul élément de l'ensemble 2 et réciproquement, l'affectation d'un élément à un autre induisant un coût. L'objectif est de réaliser cette affectation en minimisant le coût total.
Les 2 ensembles peuvent, par exemple, correspondrent à des tâches et des personnes susceptibles de les réaliser ou, comme vous avez pu le voir dans la leçon 1, à des jetons et des récipients !

Le problème d'affectation est un des rares problèmes de PLNE qui soit facile à résoudre.
Il existe de très bons algorithmes pour cela.
On peut aussi le résoudre comme un problème de programmation linéaire standard, sans tenir compte des contraintes d'intégrité, on constatera a posteriori que les variables à l'optimum valent 0 ou 1.

VII Conclusion

Cette leçon nous a permis de présenter quelques problèmes types de programmation linéaire en nombres entiers.
La programmation linéaire en nombres entiers est loin de clore la panoplie des modèles d'aide à la décision.
D'autres types de problèmes nécessiteront d'autres types de modèles.

- Dans ce cours, nous nous sommes restreints à des situations déterministes, c'est à dire que les données étaient connues avec certitude. Si ce n'est pas le cas, on devra faire appel à des **modèles probabilistes**.

Les modèles de **théorie des jeux** permettent de représenter des situations de concurrence.
On aurait pu aussi évoquer le cas où plusieurs critères doivent être pris en compte simultanément : c'est l'objet des modèles **multicritères**

La difficulté majeure de l'aide à la décision reste de choisir le "bon" modèle, ce qui est, comme on a pu le prétendre, plus un art qu'une technique.