TELECOM Nancy 2A 2019-20
Graphes et Recherche Opérationnelle

Feuille 1 : Modélisation

Exercice 1. *Résolution graphique d'un programme linéaire (pb de production)*

Une usine fabrique deux produits P_1 et P_2 en utilisant 4 machines $M_1, \cdots, M_4$. Chacun de ces deux produits nécessite un temps de passage donné (en heures) sur chacune des 4 machines. Les temps de passage par unité de produit sont indiqués dans le tableau ci-dessous. Par ailleurs, chaque machine n'est disponible que pendant une certaine durée, également indiquée dans le tableau.

	M_1	M_2	M_3	M_4
P_1	0h	1,5h	2h	3h
P_2	3h	4h	3h	0h
disponibilité	39h	60h	57h	57h

Les produits P_1 et P_2 rapportent respectivement les bénéfices $B_1 = 1700$ euros et $B_2 = 3200$ euros par unité. On cherche à déterminer le plan de production pour maximiser le bénéfice total.

1. Modéliser ce problème sous forme de programmation linéaire (PL). Donner la forme canonique pure et la forme standard du programme linéaire.
2. Résoudre **graphiquement** le PL.

Exercice 2. *Le problème du restaurateur*

Un restaurateur a constaté que sa clientèle aime les coquillages. Lorsqu'il en propose dans son restaurant, ses clients les consomment jusqu'à épuisement de sa livraison du jour. Sa carte comporte deux plateaux :

— un plateau "riche" à P_r euros qui comporte n_1^r oursins, n_2^r palourdes, n_3^r huîtres
— un plateau "simple" à P_s euros qui comporte n_1^s oursins, n_2^s palourdes, n_3^s huîtres

1. Un jour donné, son fournisseur lui a livré k_1 oursins, k_2 palourdes et k_3 huitres.
   Comment doit-il répartir les coquillages entre les deux sortes de plateaux pour réaliser un chiffre d'affaire maximal ? Modéliser ce problème sous forme de programmation linéaire. Ecrire la forme canonique pure et la forme standard du PL.
2. En début de soirée, une de ses amies vient le trouver. Elle a plus d'invités que prévu et manque de coquillages alors que les magasins sont fermés. Elle lui propose de lui racheter tous ses coquillages.
   Quel prix minimum peut-elle proposer pour chaque oursin, chaque palourde et chaque huître afin qu'il ne soit pas tenté d'en conserver une partie pour composer quelques plateaux ? Modéliser ce nouveau problème et comparer-le au précédent.

Exercice 3. *Répartition de tâches*

On découpe un programme en n procédures (tâches) que l'on veut exécuter sur un ordinateur possédant m processeurs. A chaque procédure effectuée sur un processeur donné, il correspond un coût en temps de calcul. Ainsi pour la procédure i effectuée sur le processeur j, le coût est $c_{ij} > 0$. Chacune des n tâches doit être effectuée par un et un seul processeur. Mais chaque processeur peut exécuter plusieurs tâches ou pas du tout. On cherche à minimiser le coût total d'exécution des n procédures.

1. Modéliser ce problème sous forme d'un problème de programmation linéaire faisant intervenir des variables binaires.
2. Modéliser la variante suivante (plus réaliste...) qui consiste à minimiser le plus grand coût d'exécution.

Exercice 4. *Stockage*

Une entreprise de service vidéo à la demande (vod) stocke tous ses fichiers vidéos sur un ensemble de disques durs identiques qui ont tous la même taille. L'entreprise souhaite utiliser le minimum de disques durs pour le stockage. Elle dispose de n fichiers vidéo de taille $c_1, \cdots, c_n$ avec m disques durs identiques de taille C. On stocke tous les fichiers sur les disques durs. Pour qu'un stockage soit admissible il faut que la somme des tailles des fichiers stockés sur un disque soit inférieure ou égale à C (on ne peut pas dépasser la capacité d'un disque...). De plus, un fichier n'est stocké qu'une seule fois sur un seul disque. Déterminer un stockage

consiste alors à déterminer sur quel disque j est stocké le fichier i donné. On cherche à déterminer le stockage (admissible) des fichiers sur les disques durs de façon à minimiser le nombre de disques durs utilisés.

En considérant (entre autres) les variables binaires y_j pour indiquer si le disque dur j est utilisé ou non, modéliser ce problème par un programme linéaire portant uniquement sur des variables binaires.

Exercice 5. *Affectation de bureaux*

Une entreprise va emménager dans de nouveaux locaux. Ceux-ci se composent de n bureaux identiques. Chacun de ces bureaux peut accueillir indistinctement l'un des n services de l'entreprise. Chaque bureau accueille un et un seul service. De plus, chaque service est accueilli entièrement dans un et un seul bureau. La disposition des bureaux est connue et on note d_{jl} la distance entre les bureaux j et l. L'entreprise a effectué dans les anciens locaux des statistiques sur les va-et-vient entre les n services à installer dans les n bureaux. Elle dispose du nombre c_{ik} de fois où des employés vont du service i au service k et où des employés vont du service k au service i. L'entreprise souhaite affecter les services aux bureaux de sorte que la distance totale parcourue par les employés soit la plus petite possible.

1. Combien y-a-t-il de possibilité d'affecter les n services aux n bureaux ? L'ordinateur dont vous disposez peut effectuer 1 milliard d'opérations à la seconde. Pour $n = 30$, estimer le temps mis pour obtenir par une méthode exhaustive (c'est-à-dire en essayant toutes les solutions), l'une des meilleures configurations donnant la distance totale parcourue minimale (*indication :* $30! \simeq 10^{32}$).
2. Modéliser le problème qui consiste à placer les services dans les bureaux de manière à minimiser la somme des distances parcourues par les employés. Vous pourrez introduire une variable binaire x_{ij} qui indique si le service i est placé ou non dans le bureau j. Vous obtiendrez ainsi un problème de programmation *quadratique*.
3. *Linéarisation.* Pour linéariser le problème quadratique précédent, on introduite de nouvelles variables binaires à 4 indices $y_{ijkl} = x_{ij}x_{kl}$. Montrer que

$$y_{ijkl} = x_{ij}x_{kl} \quad \Leftrightarrow \quad \begin{cases} y_{ijkl} \leq x_{ij} \\ y_{ijkl} \leq x_{kl} \\ x_{ij} + x_{kl} \leq 1 + y_{ijkl} \end{cases}$$

   Ecrire le problème de programmation *linéaire* correspondant.

Exercice 6. *Gestion de projet* (extrait du partiel de GRO 2012)

Les élèves d'une célèbre école d'ingénieurs de l'Est de la France doivent réaliser en 2ème année un Projet d'Introduction Douloureuse à la Recherche (PIDR). L'enseignant responsable des Projets est chargé de répartir les projets entre élèves. Il y a autant d'élèves que de projets et chaque élève doit être affecté à un projet et un seul. De même, chaque projet doit être attribué à un élève et un seul. Il y a n projets ($n \geq 3$). Pour décider de la répartition des projets, l'enseignant demande à chaque élève de choisir 3 projets parmi les n et de les classer par ordre strictement croissant de préférence. Chaque projet retenu par un élève se voit ainsi attribué une note de 1 à 3 et on considère que les projets non-retenus par l'élève ont la note 0. L'idée de l'enseignant (qui veut oeuvrer pour le bien collectif) est de maximiser la satisfaction générale. Cette satisfaction est définie comme étant la somme (ou la moyenne) des notes données par les élèves aux projets auxquels ils sont affectés. Il vous est demandé de venir en aide à l'enseignant en mettant en oeuvre toutes vos compétences acquises en programmation linéaire.

1. Modéliser ce problème sous forme de programmation linéaire, afin de déterminer une répartition optimale des projets en maximisant la satisfaction générale. Vous introduirez une matrice E des notes projets/élèves dont les coefficients sont les notes données à chaque projet par chaque élève.
2. Pour $n = 4$, donner un exemple de matrice E des notes projets/élèves. Pour $n = 4$, écrire le problème de programmation linéaire obtenu précédemment, sous la forme

$$\begin{cases} \max\limits_{\mathbf{x}} \left[F(\mathbf{x}) = \mathbf{e}^\top \mathbf{x} \right] \\ A\mathbf{x} = \mathbf{b} \\ \mathbf{x} \in \{0,1\}^{n^2} \end{cases} \tag{1}$$

   en précisant la variable $\mathbf{x}$ et en explicitant les vecteurs $\mathbf{e}$ et $\mathbf{b}$ ainsi que la matrice A.
3. La modélisation précédente souffre d'un manque de considération individuelle ... Dans la répartition optimale précédente, des élèves peuvent très bien se voir attribuer des projets qu'il n'avaient pas classés initialement (i.e. avec la note 0). L'enseignant décide alors que chaque élève doit être affecté à un projet parmi les 3 projets classés initialement.

(a) Modéliser cette nouvelle contrainte.

(b) Pour $n = 4$, écrivez cette contrainte sous la forme

$$B\mathbf{x} \geq \mathbf{d} \tag{2}$$

en explicitant la matrice B et le vecteur $\mathbf{d}$.

(c) Donner un exemple de matrice des notes projet/élève pour laquelle il n'y a pas de solution réalisable au problème (1),(2).

Exercice 7. *Problème d'une entreprise de vols "charter"*

Une compagnie aérienne de vols "charter" doit effectuer sur une plage de temps donné, un certain nombre de vols pour lesquels on connaît :

— AD_i l'aérogare de départ
— AA_i l'aérogare d'arrivée
— HD_i l'heure de départ
— HA_i l'heure d'arrivée

Un avion peut effectuer un vol j à la suite d'un vol i à condition qu'un intervalle de temps suffisant existe entre l'arrivée de i à l'instant HA_i en AA_i et le départ de j à l'instant HD_j en AD_j de manière à permettre éventuellement son transfert en vol à vide de AA_i à AD_j qui dure T_{ij} et la préparation du vol suivant qui dure p. Pour leur premier vol, on suppose les avions disponibles au bon endroit et après leur dernier vol, on les laisse là où ils viennent d'atterrir.

1. Un premier problème à résoudre consiste à minimiser le nombre total d'avions nécessaires.
2. Un deuxième problème consiste à fixer le nombre d'avions et à minimiser la durée des trajets à vide.

Modéliser ces deux problèmes de programmation linéaire sous les formes canoniques et standards.

Exercice 8. *Modélisation du problème du voyageur de commerce par programmation linéaire (Miller-Tucker-Zemlin, 1960).*

Le problème du voyageur de commerce s'énonce de la façon suivante. Etant données n villes, trouver le plus court trajet passant par toutes les villes une et une seule fois, en revenant à la ville de départ à la fin du trajet. Le but de cet exercice est de modéliser ce problème sous forme d'un programme linéaire. On introduit les variables d'affectation x_{ij} $(1 \leq i, j \leq n)$ telles que

$$x_{ij} = \begin{cases} 1 & \text{si on va de la ville } i \text{ à la ville } j \\ 0 & \text{sinon.} \end{cases}$$

On introduit les variables supplémentaires u_i pour $1 \leq i \leq n$, représentant la position de la ville i dans la tournée du voyageur. On suppose que la ville 1 est la ville de départ, on a donc $u_1 = 1$ et par ailleurs $u_i \in \{2, \cdots, n\}$ pour $i \in \{2, \cdots, n\}$.

1. Les variables x_{ij} et u_i sont liées par le fait que si $x_{ij} = 1$ alors nécessairement $u_i < u_j$. Montrer que cette contrainte peut se traduire algébriquement par

$$u_i - u_j + 1 \leq \alpha(1 - x_{ij}) \quad \forall i, j,\ i \neq 1, j \neq 1 \tag{1}$$

où α est un réel à déterminer.

2. On note d_{ij} la distance séparant la ville i de la ville j (a priori $d_{ij} \neq d_{ji}$). Modéliser le problème du voyageur de commerce sous forme d'un programme linéaire pour les variables x_{ij} $(1 \leq i, j \leq n)$ et u_i $(2 \leq i \leq n)$.

Exercice 9. *Identification de paramètres et programmation linéaire*

On considère deux grandeurs physiques t et y qui sont liées entre-elles par une relation fonctionnelle du type $y = f(t)$. On dispose d'un modèle pour la fonction f. On sait que f est une combinaison linéaire de n fonctions élémentaires $\phi_1, \cdots, \phi_n$ connues :

$$f(t) = f_{\mathbf{x}}(t) = x_1\phi_1(t) + \cdots + x_n\phi_n(t)$$

où $\mathbf{x} = (x_1, \cdots, x_n)^\top \in \mathbb{R}^n$ sont des paramètres réels qu'on cherche à identifier. Pour cela, on réalise une série de m mesures $(t_i, y_i)_{1 \leq i \leq m}$ et on cherche à minimiser le plus grand écart absolu entre le modèle donné par $f_{\mathbf{x}}$ et les mesures :

$$\min_{\mathbf{x} \in \mathbb{R}^n} \max_{1 \leq i \leq m} |f_{\mathbf{x}}(t_i) - y_i| \tag{2}$$

Remarques : si l'on remplace l'objectif (2) par la somme des carrés des écarts (i.e. $\min_{\mathbf{x}\in\mathbb{R}^n} \sum_{i=1}^{m} |f_{\mathbf{x}}(t_i) - y_i|^2$, alors on obtient exactement la formulation en moindres carrés...).

On notera $\|\mathbf{x}\|_\infty = \max_{1\leq i\leq n} |x_i|$ et $\|\mathbf{x}\|_1 = \sum_{i=1}^{n} |x_i|$ les normes sur $\mathbb{R}^n$ d'un vecteur $\mathbf{x} = (x_1, \cdots, x_n)^\top$.

1. Ecrire le problème (2) sous la forme

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|A\mathbf{x} - \mathbf{b}\|_\infty := \min_{\mathbf{x}\in\mathbb{R}^n} \max_{1\leq i\leq m} |A\mathbf{x}_i - b_i| \tag{3}$$

   en précisant ce que valent le vecteur $\mathbf{b} = (b_1, \cdots, b_m)^\top$ et la matrice $A \in \mathcal{M}_{m\times n}$.
2. Transformer le problème précédent en un problème de programmation linéaire sous forme canonique pure. (*Indication :* introduire la variable $e = \max_i |A\mathbf{x}_i - b_i|$).
3. On considère à présent la fonction objectif

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|A\mathbf{x} - \mathbf{b}\|_1 := \min_{\mathbf{x}\in\mathbb{R}^n} \sum_{i=1}^{m} |A\mathbf{x}_i - b_i| \tag{4}$$

   Transformer ce nouveau problème en un problème de programmation linéaire sous forme canonique pure.

Exercice 10. *Sudoku*

Un sudoku est composé d'une grille de 3×3 sous-grilles elles-mêmes constituées de 3×3 cases, soit au total 81 cases. Le but du jeu est de remplir la grille avec une série de chiffres allant de 1 à 9 tous différents, qui ne se trouvent jamais plus d'une fois sur une même ligne, dans une même colonne ou dans une même sous-grille. Quelques chiffres sont déjà placés dans la grille.

2	·	·	·	·	**4**	·	**3**	·
·	**9**	·	·	·	·	·	**7**	·
·	·	·	·	**6**	·	·	**4**	·
3	**8**	·	·	·	**7**	·	·	·
6	·	·	·	·	·	**5**	·	·
·	·	**2**	·	·	·	**8**	·	·
·	·	·	**8**	·	·	**9**	·	·
·	·	·	·	**9**	·	·	·	**2**
·	·	**4**	·	·	**2**	·	**8**	·

FIGURE 1 – Exemple d'une grille de sudoku

L'objectif de cet exercice est de modéliser le jeu de Sudoku par programmation linéaire en nombres entiers. Dans ce but, on introduit les variables binaires d'affectation suivantes : pour une case $(i, j) \in [\![1, 9]\!]^2$,

$$x_{ijk} = \begin{cases} 1 & \text{si le chiffre } k \in [\![1, 9]\!] \text{ est affecté à la case } (i, j) \\ 0 & \text{sinon} \end{cases} \tag{5}$$

1. Ecrire les différentes contraintes de la règle du jeu comme des contraintes algébriques linéaires portant sur les variables x_{ijk}.
2. Dans le problème de Sudoku, il n'y a rien à optimiser, il n'y a que des contraintes. On peut cependant fixer l'objectif $f \equiv 1$ ou bien introduire des variables artificielles a_l dans les contraintes et l'objectif devient $\min \sum_l a_l$. Ecrire le problème de Sudoku comme un PL.