

Optimisation combinatoire

• Données

• Variable \rightarrow à déterminer.

• Contraintes

• Objectif \rightarrow minimiser \rightarrow maximiser

Une combi° linéaire: une somme et des coeffs.

Programmation linéaire:

$$\text{Min } \sum_{i=1}^n c_i x_i \rightarrow c = \text{contraintes}, x = \text{variables}$$

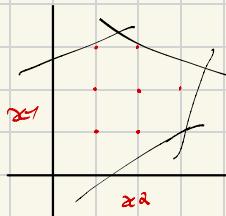
$$a_1 x_1 + a_n x_n \leq b_1$$

:

$$a_k x_1 + \dots + a_n x_n \leq b_k$$

$x_1, \dots, x_n \geq 0$, on Rⁿ c'est des valeurs réel R

Domaine des contraintes, quand on a 2 variables



En optimisation combi°:

les variables sont des entiers: $x_i \in \mathbb{N}$ \downarrow entier

Exemple: le sac à dos de 15kg.

formulation en français

P: limite de poids du sac

n Objets: objet i a un poids P_i

a une valeur d'utilité U_i

objectif: constitue un sac d'utilité maxima.

formulation en math:

pour l'objet i, on a une variable $x_i \in \{0, 1\}$

si objet i est dans le sac

x_i ↗ si non

$$\sum_{i=0}^n x_i p_i \leq P \rightarrow \text{contrainte du poids max}$$

$$\max \sum_{i=0}^n w_i x_i \rightarrow \text{Fonction d'objectif.}$$

difference avec un prof° linéaire en RO en opt° combi° on a des variables binaires.

P _i	3	4	2	5	1	6	3	6	2
w _i	30	8	-18	15	8	16	-12	-12	44

Idees d'algorithmes:

Algo 1:

- ① Trier les objets par w_i → complexité $O(n \log(n))$
- ② On essaye de les placer un par un dans cet ordre $O(n)$

avec notre tableau ok. Mais avec le tableau suivant non:

P _i	15	8	4
w _i	30	-16	16

Algo 2:

- ① on tue par w_i/p_i)
- ② on les place dans cet ordre.

P _i	3	4	2	5	1	6	3	6	2
w _i	30	8	-18	15	8	16	-12	-12	44
w _i /p _i	10	2	9	3	8	4	6	2	7

Algo 3:

meilleure = 0

$O(2^n)$ { enumérer les vecteurs binaires (x_0, \dots, x_n)

pour chacun :

Complexité : linéaire $O(n)$ { si $\sum_{i=0}^n p_i x_i \leq P$ et $\sum_{i=0}^n w_i x_i > \text{meilleure}$
 $\text{meilleure} = \sum_{i=0}^n w_i x_i$

$O(n2^n)$

Plan de la matière :

I. Modélisation et bases (1 session → 2 cours sur la semaine du 17/03)

II. Méthodes exactes (utilisent l'enumeration) (1 session → 3 semaines)

- Méthode arborescente (cours du 26/03)

- Programmation dynamique (interv. pendant la dernière session, le 27/03)

III. Méthodes approchées

Théorie la qualité des solutions Projet (en groupe)

- Algo glouton

- Théorématiques

Ce document contient des exercices/archives d'écran qu'on va faire en cours.



40 exercices en lien avec le cours 276.6 Ko

Exercice 1.1 Binômes

Modéliser le problème suivant à l'aide d'un programme mathématique. On considère un ensemble de personnes $P = \{1, \dots, n\}$ qui doivent travailler en binôme sur des projets (on supposera que n est pair). Chaque personne possède une capacité c_i connue à l'avance et qui mesure son efficacité au travail : plus c_i est petit, plus la personne est rapide dans l'exécution de sa part du projet. Si on constitue un binôme entre deux personnes i et j alors le temps que mettra le binôme pour terminer son projet est $c_i + c_j$. \rightarrow capa* pour terminer le projet en binôme.

On dispose de plus d'un graphe d'incompatibilité G , dont les sommets sont les personnes. L'existence d'une arête $\{i, j\}$ signifie que les personnes i et j ne peuvent pas être en binôme.

Le but est de répartir les personnes en binômes compatibles de sorte que tous les projets soient terminés en un minimum de temps (Les binômes travaillent en parallèle).

Step 1: les données connues

P = ensemble des personnes $\{1, \dots, n\}$, n est pair

G = graphe d'incompatibilité

i, j = personne i , personne j

Step 2: décrire les variables

pour deux personnes i, j :

$x_{ij} \begin{cases} 1 & \text{si } i \text{ est en binôme avec } j \\ 0 & \text{sinon} \end{cases}$

y_{ij} = une arête d'incompatibilité

T = durée du projet

Step 3: contraintes

- deux personnes incompatibles ne sont pas en binôme $\Rightarrow x_{ij} = 0$, si $\{i, j\}$ est une arête de G .
- une personne i n'a qu'un binôme $\Rightarrow \sum_{j \in P} x_{ij} = 1$ pour tout $i \in \{1, \dots, n\}$
- une personne i avec j est un binôme avec i $\Rightarrow x_{ij} = x_{ji}$
- $x_{ij}(c_i + c_j) \leq T$ pour tout $i, j \in \{1, \dots, n\}$
- $x_{ij} \in \{0, 1\}$ pour tout $i, j \in \{1, \dots, n\}, i \neq j$

Step 4: fonction objectif

$\min T$

Écrire en français les variables puis les contraintes
les données:

O : durée ouvrable de la journée

w_i, v_i : durée d'une tâche sur les machines

i : une tâche

p_i : pénalité d'une tâche non exécutée

les variables:

$x_{ii} = 1$ si i est fait sur machine 1

$x_{ii} = 0$ sinon

$y_{ii} = 1$ si i est fait sur machine 2

$y_{ii} = 0$ sinon

$z_i = 1$ si tâche n'est pas faite

$z_i = 0$ sinon

Objectif: $\min \sum_{i=0}^n p_i z_i$

minimum de pénalité sur les tâches non faites

Exercice 1.6 Ordonnancement à deux machines de pénalité minimale

Modéliser le problème suivant : On considère un ensemble de n tâches à faire. Les tâches peuvent être faites sur l'une ou l'autre de deux machines différentes. Chaque tâche i a une durée u_i si elle est faite sur la première machine, v_i sur la seconde. Chaque machine n'exécute qu'une seule tâche à la fois et réalise donc toutes les tâches qui lui sont affectées en séquence (dans n'importe quel ordre) sans s'arrêter, dans une durée qui ne doit pas dépasser la durée ouvrable de la journée O . Les deux machines fonctionnent en parallèle.

Malheureusement, il n'est pas toujours possible de faire toutes les tâches dans la durée impartie. Chaque tâche i , si on ne l'exécute pas engendre une pénalité notée p_i . On cherche donc à déterminer pour chaque tâche si on l'exécute ou non, et si oui sur quelle machine, de sorte que chaque machine fonctionne sur une durée qui n'excède O et que la somme des pénalités liées aux tâches non-exécutées soit minimale.

les contraintes:

$$x_{ii} + y_{ii} + z_i = 1 \text{ pour chaque } i \in \{1, \dots, n\}$$

$$\sum_{i=0}^n w_{ii} \leq O : \text{durée des tâches sur machine 1}$$

$$\sum_{i=0}^n v_{ii} \leq O : \text{durée des tâches sur machine 2}$$

$$x_{ii}, y_{ii}, z_i \in \{0, 1\} \text{ pour tout } i$$

Exercice 1.3 Gestion de stock

Une usine automobile cherche à planifier sa production sur les H prochaines semaines. Elle dispose de ses prévisions, qui donnent pour chaque semaine i un nombre de véhicules D_i à livrer en fin de semaine.

Selon les aléas de la production (notamment congés, coût des transports, etc), le coût de production des véhicules varie selon les semaines. On note c_i le coût de production d'un véhicule durant la semaine i . Produire x véhicules pendant la semaine i coûte donc $c_i x$. L'usine a, de plus, une capacité de production fixe par semaine, notée Q (c'est à dire ne peut produire plus de Q véhicules par semaine).

Ces deux éléments peuvent conduire à produire des véhicules en avance pour une commande d'une semaine ultérieure. Les véhicules doivent donc être stockés. Chaque véhicule stocké à la fin de la semaine i a un coût unitaire de stockage noté t_i . (stocker x véhicules en fin de semaine i coûte donc $t_i x$).

La production se déroule donc selon le schéma suivant : chaque semaine i , le stock restant de la semaine précédente est complété avec la production de la semaine, le client est livré en fin de semaine, et il peut éventuellement rester un stock pour la semaine suivante. L'entreprise cherche à calculer un plan de production, c'est à dire savoir combien de véhicules doivent être produits chaque semaine, de sorte que le coût total de production et de stockage soit minimum et que tous les clients soient livrés. Modéliser ce problème.

données

H semaines

D_i: nb de véhicules à livrer pour la semaine i.

C_i: coût de prod d'un véhicule pour la semaine i

Q: capa^e de l'usine à par semaine

c_{i,x}: coût de prod de x véhicule pour la semaine i

b_i: coût unitaire de stockage

variables:

s_i: Stock restant stocké par semaine i.

x_i: qte à produire pour la semaine i

contraintes:

• capacité de prod, Q, de l'usine à pour la semaine i: x_i ∈ Q, pour tout i ∈ {1, ..., n}, x_i ∈ N.

• stock restant de la semaine i après l'usage du stock s_i Si = S_{i-1} + x_i - D_i

• livre D_i pour la semaine i : si > 0 → si il reste du stock, on le stocke est à zéro, ya veux dire on en assez produit si c_{i,x} / la puissance de notre produit

objectifs

$$\text{Max} \sum_{i=1}^n c_i x_i + b_i s_i$$

exercice du sac à dos bidimensionnel

contrainte sur le poids et le volume.

obj_i $\begin{cases} p_i: \text{poids} \\ v_i: \text{volume} \\ w_i: \text{utilité} \end{cases}$

x_i = un objet $\begin{cases} \rightarrow \text{pris} \\ \rightarrow \text{pas pris} \end{cases}$

sac poids max P

volume max V

Objectif: sac d'utilité maximal.

Contraintes

$$\bullet \sum_{i=1}^n p_i x_i \leq P$$

$$\bullet \sum_{i=1}^n v_i x_i \leq V$$

$$\bullet \forall i \quad x_i \in \{1, 0\}$$

Objectif

$$\text{Max} \sum_{i=0}^n w_i x_i$$

données :

G : graphe

S : sommets

A : arêtes

$\{i,j\}$: une arête

c_i : capacité d'émission

Variables :

$x_{i,j}$:
 1 émet actuellement
 0 n'émet pas.

contraintes : $x_{i,j} + x_{j,i} \leq 1$ si $\{i,j\}$ une arête de G

- $\forall i, x_{i,i} \in \{0,1\}$

Objectif

$$\text{Max } \sum_{i=1}^n c_i x_{i,i}$$

Exercice des points de liaison.

données

n points de liaison

visiter chaque point et revenir au point 1

D_{ij} : distance entre i et j

Variables

- a) x_{ijk} :
 1 chemin emprunté
 0 sinon
 ou
 b) y_{ik} :
 1 si point de liaison i est en position k
 0 sinon

Variante b:

Contraintes

- $y_{ik} \in \{0,1\}$
- $\sum_{i=1}^n y_{ik} = 1$ pour tout $i \in \{1, \dots, n\}$ chaque point est dans une position
- $\sum_{i=1}^n y_{ik} = 1$ chaque position a un point
- $y_{11} = 1$: premier point parcouru

objectif :

Soit k une position. On veut exprimer la distance entre la ville en position k et la

ville en position $k+1$

Somme totale

$$\text{Min} \sum_{k=1}^{n-1} \sum_{i=1}^n \sum_{j=i}^n D_{ij} y_{ik} y_{jk} + \sum_{i=1}^n D_{1i} y_{1i}$$

Retour au point 1.
point en position n à la ville i.
Distance de i au point 1.

Version a:

$$\text{Objetif: Min } \sum_{i=1}^n \sum_{j=1}^n D_{ij} x_{ij}$$

Contraintes:

- du point i on va à un seul point: $\sum_{\substack{j \in S \\ j \neq i}} x_{ij} = 1$, pour tout i;

- on arrive en j que d'un seul point:
$$\sum_{\substack{i=1 \\ i \neq j}}^n x_{ij} = 1, \text{ pour tout } j$$

- $x_{ij} \in \{0, 1\}$, $x_{ii} = 0$

① Avoir les questions de recherche et les requêtes

② Définir des critères d'inclusion et d'exclusion hard.

- langues

- au moins 4 pages

- articles de type réc., pas de conférences et de journalisme

③ Filtrer sur les titres

si le titre correspond à mon sujet

↳ si non on l'élimine.

Prendre X articles : max 200

④ filtres sur les mots clés
garder les articles pertinents

⑤ filtres sur les abstracts
garder les articles pertinents

⑥ Ensuite on lit les abstracts et on garde les derniers articles pertinents.

Dans la bibliographie du mémoire mettre les DOI de nos articles.

Dans le mémoire : parler des étapes de recherche et de filtres.

Dans ma problématique ajouter deux types de canaux

Réseau de neurones récurrents.