Projet SDP

CentraleSupélec

Système de planification de projets - CompuOpti

Carl Rizk, Dylan Sechet, David De la Hera Carretero

Sommaire

1. Modélisation
    a. Contraintes
    b. Objectifs

    ~~Algorithme Epsilon Constraint~~

2. Modélisation des préférences

Modélisation: contraintes

- Qualification: $\forall e, \forall p, \forall j, \forall q \notin qualifications(e), projet_{e,j,p,q} = 0$

- Congés: $\forall e, \forall p, \forall q, \forall j \in conges(e), projet_{e,j,p,q} = 0$

- Non-duplication: $\forall e, \forall j, \sum_{p} \sum_{q} projet_{e,j,p,q} \leq 1$

- Travail utile: $\forall p, \forall q, \sum_{e} \sum_{j} projet_{e,j,p,q} \leq duree(p, q)$

Modélisation: premier objectif

Objectif: maximiser le bénéfice

Jour (j)	1	2	3	4	5	6
$realise_{0,j}$	0	0	0	0	1	1
$penalite(0,j)$	0	0	0	5	5	5

Date de livraison

Fin du projet

NB: *penalite vaut en réalité 0.0001 les jours 1, 2 et 3*

Modélisation: premier objectif

$$\max \sum_{p} gain(p) \cdot realise_{p,N+1} - \sum_{p,j \leq N+1} penalite(p,j) \cdot (1 - realise_{p,j})$$

Jour (j)	**1**	**2**	**3**	**4**	**5**	**6**
$realise_{0,j}$	0	0	0	0	1	1
$1 - realise_{0,j}$	1	1	1	1	0	0
$penalite(0,j)$	0	0	0	5	5	5
$penalite(0,j) \cdot (1 - realise_{0,j})$	0	0	0	5	0	0

Date de livraison Fin du projet

Modélisation: premier objectif

$$\max \sum_{p} gain(p) \cdot realise_{p,N+1} - \sum_{p,j \leq N+1} penalite(p,j) \cdot (1 - realise_{p,j})$$

On ajoute la contrainte:

$$\forall p, \forall j, \sum_{q} duree(p,q) - \sum_{j' < min(j,N)} \sum_{e,q} projet_{e,j',p,q} \leq M \cdot (1 - realise_{p,j})$$

Modélisation: deuxième objectif

Objectif: minimiser le nombre de projets auquel chaque collaborateur est affecté :

On introduit $affecte_{e,p}$ qui vaut 1 ssi l'employé *e* travaille sur le projet *p*.

La fonction objectif est: $min \left(\sum_{e,p} affecte_{e,p} \right) = max \left(- \sum_{e,p} affecte_{e,p} \right)$

On ajoute la contrainte: $\forall p, e, \sum_{j,q} projet_{e,j,p,q} \leq M \cdot affecte_{e,p}$

Modélisation: troisième objectif

Objectif: minimiser l'étendue totale des projets

Jour (j)	**1**	**2**	**3**	**4**	**5**	**6**
$debute_{0,j}$	0	1	1	1	1	1
$realise_{0,j}$	0	0	0	0	1	1

Début du travail

Fin du projet

Modélisation: troisième objectif

$$min \sum_p \sum_j debute_{p,j} - realise_{p,j} = max \sum_p \sum_j realise_{p,j} - debute_{p,j}$$

On introduit $debute_{p,j}$ qui vaut 1 ssi le projet *p* a démarré un jour antérieur ou égal à *j*.

On ajoute la contrainte: $\forall p, j, \sum_{j' \leq j} \sum_{e,q} projet_{e,j',p,q} \leq M \cdot debute_{p,j}$

Modélisation des préférences

But: prendre une décision basée sur les scores des 3 objectifs

Méthode:

- Générer des problèmes aléatoires
- Résoudre les problèmes avec le solver
- Avoir un décisionnaire qui, en prenant les 3 scores, décide si la planification est bonne
- Utiliser une régression logistique pour apprendre les facteurs de décisions du décisionnaire

Performance: 30 échantillons montrés au décideur, F1-score de 0.949

Merci pour votre attention

Annexe

Modélisation

Variables

j - jour, numeroté de [1, N]

e - employé, numéroé de [1, E]

q - qualifications numéroté de [1, Q]

p - projet, numéroté de [1, P]

M une constante arbitrairement grande

Penalite_cste une penalité journalière en cas de retard

Fonctions

Jours congés associées à un employé e

$$conges : [\![1; E]\!] \rightarrow \mathscr{P}([\![1; N]\!])$$

Qualifications associées à un employé e

$$qualifications : [\![1; E]\!] \rightarrow \mathscr{P}([\![1; Q]\!])$$

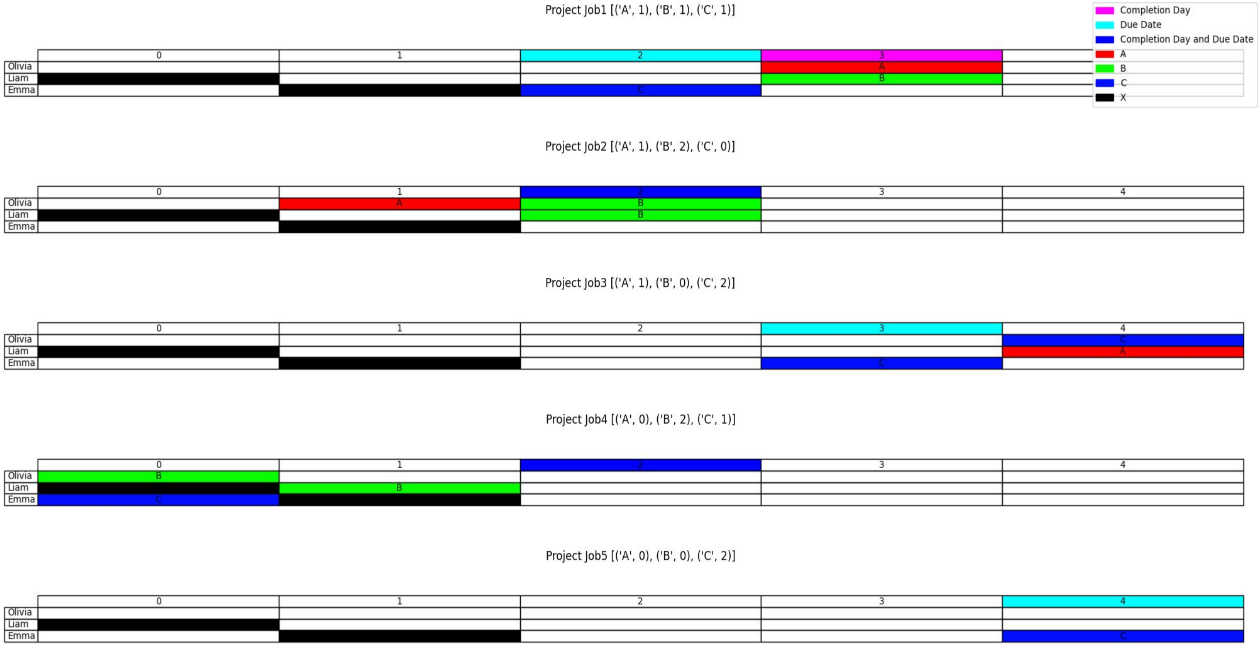
Durée(p,q) indique le nombre d'heures de compétence q nécessaires pour le projet p

Gain (p) indique le gain réalisé si le projet p est complété

Date limite associé au projet p

$$due\_date : [\![1; P]\!] \rightarrow [\![1; N]\!]$$

Premiers résultats



Qualification à faire par projet, par jour et employée

Premiers résultats

Qualification à faire par employé, par jour et projet

Algorithme epsilon-constraint

Tous les objectifs ont été réécrits comme des maximisations.

Première idée:

- Trouver x0 une solution optimale de max f1
- ND <- ND U {x0}
- Prendre epsilon2 = f2(x0) et epsilon3 = f3(x0)
- Tant que max f1 : f2>= epsilon2 a une solution x
    - epsilon2 = f2(x) + 1
    - Tant que max f1: f2 >= epsilon2 & f3 >= epsilon3 a une solution x
        - Epsilon3 = f3(x) + 1
        - ND <- ND U {x}

Algorithme epsilon-constraint

Problème: que se passe-t-il si la première solution x0 maximise aussi f2 ?

Imaginons par exemple que f1 est affecte, f2 est longueur et f3 est le gain. L'algorithme minimise le nombre d'employés affectés et le met à 0. Aucun projet n'est donc fait et la longueur vaut 0. Dans l'algorithme ci-dessus, on ne rentre jamais dans les boucles internes, et l'algorithme ne renvoie que {x0}.
De façon générale, on manque tous les points (xf1, xf2, xf3) tels que xf2 <= f2(x0), qui pourraient être ND.

Solution: On applique epsilon constraint classique à f1,f3, et en enregistre toutes les valeurs de f2 parcourues. On note min_f2 la plus petite de ces valeurs, et on l'utilise pour initialiser epsilon2 à la place de f2(x0).