Optimisation: Reactive Covering

Adrien M
Othmane N

Sommaire

Algorithme glouton naïf

L'algorithme glouton naïf est donné pour prendre en main le sujet.

Description de l'algorithme

1. Pour chaque passe, il regarde les accès associés à cette passe.
   - Pour chaque accès, il vérifie les demandes d'observation restantes (= non couvertes).
     - Si l'accès figure dans la demande d'observation, cela signifie que la demande peut être couverte par cette passe.
2. Si une demande d'observation est couverte, elle est marquée comme couverte.
3. Si au moins une demande est couverte par une passe, la passe est sélectionnée.

<u>Critère d'arrêt:</u>
Lorsque toutes les demandes d'observation sont couvertes.

Résultats

La solution présente un caractère aléatoire.
Elle dépend de l'ordre dans lequel les passes sont traitées.

Voici la moyenne des résultats du glouton sur 1000 calculs:

Réactivité	**12h**	**4h**	**2h**
Temps de calcul	Quelques ms	Quelques ms	Quelques s
Passes sélectionnées	10	42	57
Coût	3 600 €	14 900 €	19 600 €
Satisfaction	100%	100%	100%

Résultats

Voici les meilleures solutions obtenues sur 1000 calculs:

Réactivité	**12h**	**4h**	**2h**
Temps de calcul	Quelques ms	Quelques ms	Quelques s
Passes sélectionnées	4	34	37
Coût	1 530 €	12 259 €	13 309 €
Satisfaction	100%	100%	100%

Passages complets - Approche gloutonne

Dans l'optique d'améliorer l'algorithme glouton donné, un algorithme glouton guidé sur la rentabilité de chaque passe a été mis en place.

Description de l'algorithme

Définition d'un rendement:

Le rendement d'une passe est défini comme **le rapport entre le nombre d'accès qui satisfont des requêtes non couvertes par la passe et le coût total de la passe**.
A chaque itération, l'algorithme **calcule le rendement de chaque passe restante** et **sélectionne la passe ayant le plus grand rendement**.

```
def bestPassId(remainingSatPassesIds, covered):
    passesEff = {}
    for i in remainingSatPassesIds:
        satPass = satPasses[i]

        coveredByPass = 0
        for accessId in satPass['accessIds']:
            access = accesses[accessId]
            for j in range(nObsRequests):
                obsRequest = obsRequests[j]
                if not covered[j] and accessId in obsRequest['accessIds']:
                    coveredByPass += 1

        startPass = satPass['start']
        endPass = satPass['end']

        passDuration = endPass - startPass
        satPassCost = fixedPassCost + passDuration*passCostPerTimeUnit
        passEfficiency = coveredByPass/satPassCost

        passesEff[i] = passEfficiency


    return max(passesEff, key=passesEff.get)
```

Figure: Sélection de la passe la plus rentable

Résultats

La solution n'a plus aucun caractère aléatoire.

Réactivité	**12h**	**4h**	**2h**
Temps de calcul	10ms	10s	30s
Passes sélectionnées	6	30	35
Coût	2 122 €	10 863 €	13 059 €
Satisfaction	100%	99.29%	97.88%

Pour les contraintes de réactivité de 4h et 2h, toutes les contraintes ne peuvent pas être satisfaites

Passages complets - MILP

- **Variables de décision:**
  - $p_i \in \{0;1\}$: sélection de la passe i, $\quad i \in \mathcal{P}$
  - $r_j \in \{0;1\}$: remplissage de la requête j, $\quad j \in \mathcal{R}$
  - $a_k \in \{0;1\}$: sélection de l'accès k de la passe i, $\quad k \in \mathcal{A}$
- **Objectif:**

$$\min \sum_{i \in \mathcal{P}} c_i \cdot p_i, \quad \text{avec} \quad C_i \text{ le coût de la passe i.}$$

Passages complets - MILP

- **Contraintes:**

$$\forall i \in \mathcal{P}, \forall k \in \mathcal{A}, \, p_i \geq a_k \, :$$

si un accès k de la passe i est sélectionné, alors la passe i est sélectionnée.

$$\forall j \in \mathcal{R}, r_j = 1 \Rightarrow \sum_{k \in \mathcal{A}(j)} a_k \geq 1 \quad :$$

pour remplir une requête, il faut sélectionner au moins un accès qui contient l'identifiant de la requête. On linéarise cette contrainte:

$$\forall j \in \mathcal{R}, \quad \sum_{k \in \mathcal{A}(j)} a_k \geq r_j$$

$$\sum_{j \in \mathcal{R}} \frac{r_j}{nObsRequests} \geq satisfaction\_score \quad :$$

le nombre de requêtes satisfaites doit dépasser un certain seuil (à définir).

Résultats

Réactivité	**12h**
Temps de calcul	10ms
Passes sélectionnées	4 ([6, 31, 34, 67])
Coût	1 525 €
Satisfaction	100%

L'écart à la meilleure borne est ici de 0%.

Pour les contraintes de réactivité de 4h et 2h, **l'algorithme ne peut pas trouver une solution où le score de satisfaction est de 100%.**
Il faut donc jouer avec le taux de satisfaction minimum acceptable.

Résultats

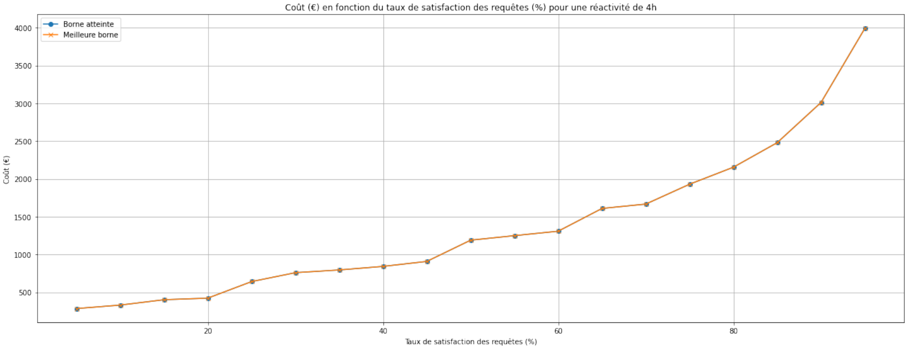


Figure: Coût en fonction du taux de satisfaction (4h)

L'écart à la meilleure borne est ici de 0%.

Remarque: Il aurait été idéal de ne pas interpoler les points lors du tracé du front de Pareto.

Résultats

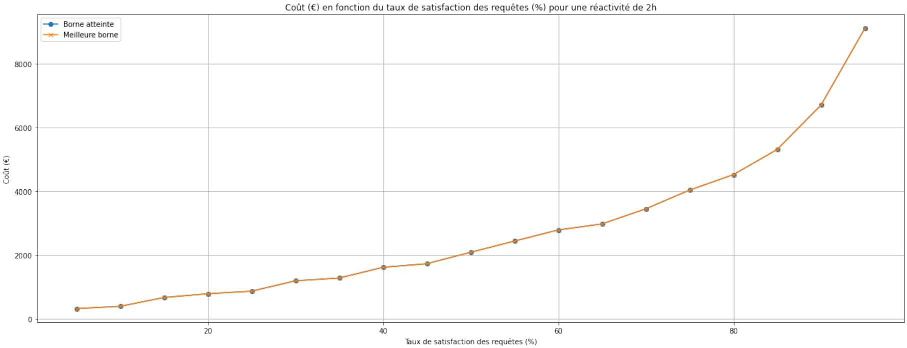


Figure: Coût en fonction du taux de satisfaction (2h)

L'écart à la meilleure borne est ici de 0%.

Remarque: Il aurait été idéal de ne pas interpoler les points lors du tracé du front de Pareto.

Résumé - Passages Complets

Critère	Glouton naïf	Glouton guidé	MILP
Caractère aléatoire	Oui	Non	Non
Taux de satisfaction max.			
12h	100%	100%	100%
4h	100%	99.29%	$<$100%
2h	100%	97.88%	$<$100%
Nombre de passes min.			
12h	10	6	4
4h	42	30	-
2h	57	35	-
Coût min.			
12h	3,600€	2,122€	1,525€
4h	14,900€	10,863€	$<$4,000€
2h	19,600€	13,059€	$<$9,000€

Table: Comparison des méthodes

Passages partiels - MILP

Pour minimiser le coût, il est possible de ne réserver qu'une partie de la passe lorsqu'elle est sélectionnée, au lieu de réserver la totalité de la durée de la passe.

Définition du problème

- **Variables de décision:**
  - $p_i \in \{0; 1\}$: sélection de la passe i, $\quad i \in \mathcal{P}$
  - $r_j \in \{0; 1\}$: remplissage de la requête j, $\quad j \in \mathcal{R}$
  - $a_k \in \{0; 1\}$: sélection de l'accès k de la passe i, $\quad k \in \mathcal{A}$
  - $e \in \mathbb{N}^{nPasses}$: Fin de réservation de la passe i
  - $s \in \mathbb{N}^{nPasses}$: Début de réservation de la passe i
- **Objectif:**

$$\min \sum_{i \in \mathcal{P}} c_i \cdot p_i, \quad \text{avec} \quad C_i \text{ le coût de la passe i.}$$

Définition du problème

- **Contraintes:**

$\forall i \in \mathcal{P}, \forall k \in \mathcal{A}, p_i \geq a_k$:
si un accès de la passe i est sélectionné, alors la passe i est sélectionnée.

$\forall j \in \mathcal{R}, \quad \sum_{k \in \mathcal{A}(j)} a_k \geq 1 - nbAccesses \cdot (1 - r_j) \quad$:
pour remplir une requête, il faut sélectionner au moins un accès qui contient l'identifiant de la requête.

$\sum_{j \in \mathcal{R}} \frac{r_j}{nObsRequests} \geq satisfaction\_score \quad$:
le nombre de requêtes satisfaites doit dépasser un certain seuil (à définir).

Définition du problème

- **Nouvelles contraintes:**

$\forall i \in \mathcal{P},\ s_i = satPasses_i[\text{'start'}]$:
initialisation de la variable s de la passe i.

$\forall i \in \mathcal{P},\ e_i = satPasses_i[\text{'end'}]$:
initialisation de la variable e de la passe i.

$\forall i \in \mathcal{P},\ e_i \geq s_i$:
la fin de la réservation est après le début de la réservation.

Définition du problème

Nouvelles contraintes:

$\forall i \in \mathcal{P},\ e_i + s_i \geq 30$: la durée de réservation est d'au moins 30.

Pour une passe donnée, si un accès k de la passe i est sélectionnée:

$a_k = 1 \Rightarrow accesses_k[\text{'end'}] - s_i \geq 30$

$a_k = 1 \Rightarrow e_i - accesses_k[\text{'start'}] \geq 30$

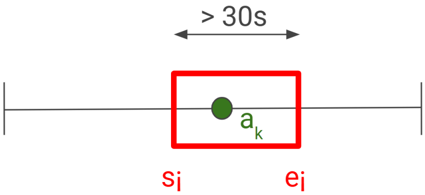


Figure: Schéma d'un accès

Conditions d'Accès

$$a_k = 1 \Rightarrow accesses_k[\text{'end'}] - s_i \geq 30 \quad \Leftrightarrow \quad \begin{aligned} M_1 &= satPasses_i[\text{'end'}] + 30 - accesses_k[\text{'end'}] \\ s_i + 30 &- accesses_k[\text{'end'}] \leq M_1 \times (1 - a_k) \end{aligned}$$

$$a_k = 1 \Rightarrow e_i - accesses_k[\text{'start'}] \geq 30 \quad \Leftrightarrow \quad \begin{aligned} M_2 &= 2 \times (30 + accesses_k[\text{'start'}]) \\ accesses_k[\text{'start'}] &+ 30 - e_i \leq M_2 \times (1 - a_k) \end{aligned}$$

Résultats

Réactivité	**12h**
Temps de calcul	10ms
Passes sélectionnées	4 ([6, 31, 34, 67])
Coût	1 126 €
Satisfaction	100%

A noter que la meilleure limite possible pour l'algorithme est 595. La solution trouvée est donc à 47.18% de l'optimale.
Pour les contraintes de réactivité de 4h et 2h, **l'algorithme ne peut pas trouver une solution où le score de satisfaction est de 100%.**

Résultats

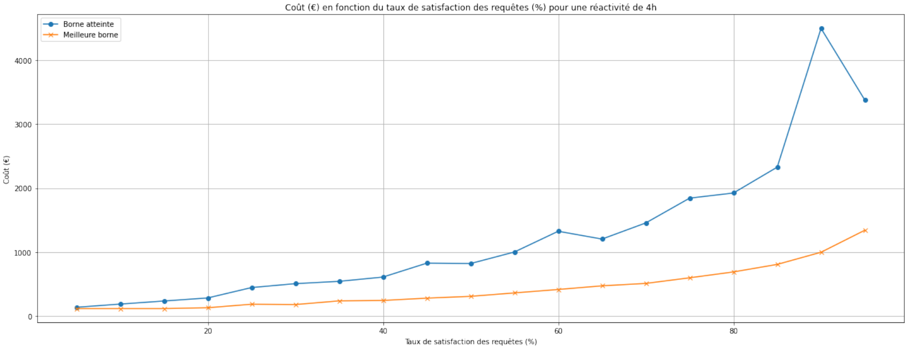


Figure: Coût en fonction du taux de satisfaction (4h)

Le pic est peut-être du à une limite de temps de calcul trop petite ($20s$).

Remarque: Il aurait été idéal de ne pas interpoler les points lors du tracé du front de Pareto.

Résultats

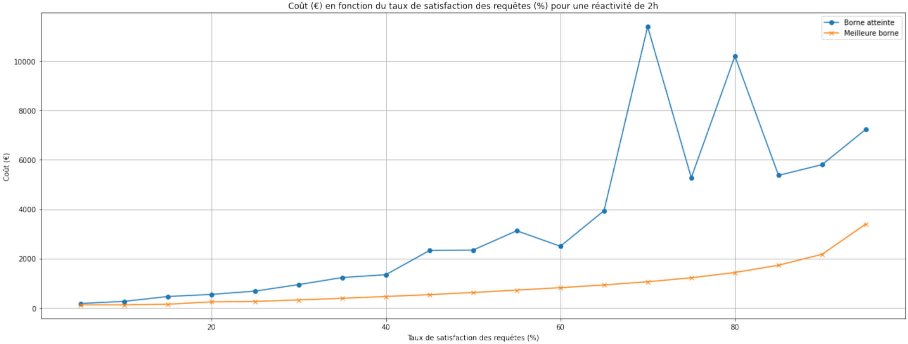


Figure: Coût en fonction du taux de satisfaction (2h)

Les pics sont peut-être dus à une limite de temps de calcul trop petite ($20s$).

Remarque: Il aurait été idéal de ne pas interpoler les points lors du tracé du front de Pareto.

Passages partiels- MILP multi-objectifs

Une nouvelle fonction objectif a été défini:

- Minimisation du coût
- Maximisation de la satisfaction

Définition du problème

Nouvelle fonction objectif:

$$f = \alpha \times \frac{\min(C_1)}{C_{1_{max}}} - (1 - \alpha) \times \min(C_2) \tag{1}$$

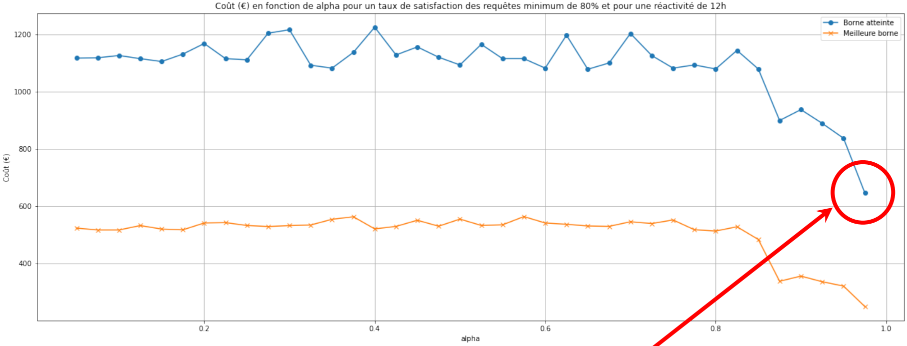
Avec:

$C_1 = \sum_{i \in P} p_i \times fixedPassCost + (e_i - s_i) \times passCostPerTimeUnit$
$C_2 = \sum_{j \in R} \frac{r_j}{\text{nObsRequests}}$

Résultats

Il est possible de jouer sur le paramètre α pour privilégier la minimisation des coûts au remplissage des requêtes (α élevé) ou privilégier la maximisation des requêtes à remplir au coûts associés (α faible).
Par exemple, en fixant le taux de requêtes satisfaites à **80% minimum**, il est possible de forcer le non remplissage de toutes les requêtes, même pour une réactivité de 12h et ainsi trouver des solutions très peu coûteuses.

Résultats



80% de requêtes satisfaites
2 passes sélectionnées
coût de 646€ (meilleure borne: 249€)
$\alpha = 0.975$

Figure: Coût en fonction de α

Coût en fonction de α

Il est possible de faire la même chose pour les exigences de réactivité de 4h et 2h.

Il est également possible de changer le taux de satisfaction minimum et afficher à nouveau le coût en fonction de α.

Conclusion

Quelques remarques annexes:

- Il existe des requêtes qui ne peuvent jamais être satisfaites, notamment pour les réactivités de 4h et 2h, car aucun accès ne les couvre.
  Il aurait été possible de supprimer ces requêtes qui de toute manière font baisser le taux de satisfaction sans que l'on puisse faire quelque chose.
- Il aurait été possible de comparer les deux fronts de Pareto pour le MILP mono-object et le MILP multi-objectif sur un même graphique.