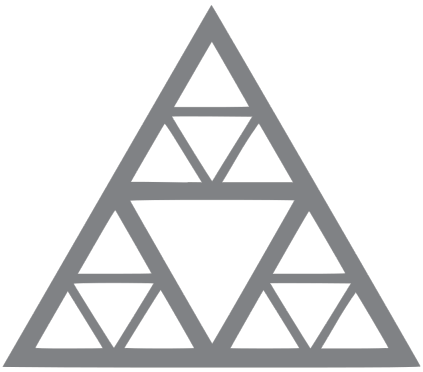
École Nationale des Ponts et Chaussées



École des Ponts

ParisTech

Rapport de projet MOPSI

Piaulage à la résidence Meunier des admissibles au concours Mines-Ponts

---

LE CHENADEC Bastien & TERRISSE Theïlo

Tuteur : Vincent Leclere

2021-2022

1 Présentation du problème

1.1 Contexte général

À la sortie des classes préparatoires aux grandes écoles, les élèves passent des concours pour tenter d'intégrer les meilleures écoles d'ingénieur selon leurs préférences personnelles. Ces concours commencent par une phase écrite, permettant d'éliminer une partie des candidats. Les élèves ayant suffisamment réussi leurs écrits sont dits « admissibles » et peuvent passer la phase orale des concours pour tenter de décrocher une place dans l'école de leur choix.
Dans le cas du concours Mines-Ponts, une partie des admissibles passent leurs oraux sur le campus de l'école des Ponts ParisTech, et doivent loger à proximité de cette dernière pour la durée des épreuves. Dans ce contexte, l'école propose aux élèves de demander à être logés à la résidence Meunier, une résidence étudiante habituellement réservée aux élèves des Ponts et gérée par l'association Arpej.
Cependant, chaque semaine, il y a plus de candidats passant leurs oraux sur le campus que de places disponibles à la résidence Meunier. Pour faire une demande de logement, les élèves passent par un site sur lequel ils renseignent un certain nombre de critères (étranger ou non, distance du campus, statut boursier ou non, genre, préférences des élèves...). L'attribution des places se fait alors sur la base des critères renseignés, complétés selon le principe de "premier arrivé, premier servi" (dit « shotgun » dans la suite de ce rapport) lorsque les critères ne permettent pas de décider l'attribution des chambres. Elle peut être modélisée par un problème d'optimisation linéaire en nombres entiers qu'il s'agit ensuite de résoudre.

1.2 Problématisation et hypothèses

Les oraux s'étalent sur 4 semaines, chaque élève ayant des épreuves durant une seule de ces semaines. Chaque élève demandeur est donc amené à rester au plus une semaine, et la durée réelle de séjour peut varier selon la répartition des épreuves dans la semaine. Cependant, Arpej impose que chaque élève qui fait une demande de logement s'inscrive pour la durée d'une semaine entière. De plus, un shotgun a lieu pour chaque semaine : peu avant la semaine de logement, les étudiants peuvent s'inscrire durant une certaine plage temporelle, à l'issue de laquelle les demandes (et leur date d'envoi sur le site) sont fournies à l'algorithme pour attribution des chambres. En réalité, on peut donc considérer que le problème ne concerne qu'une semaine, si bien que la notion de dates de séjour n'entre pas dans le cadre du problème car est déjà résolue.
Par ailleurs, les élèves sont répartis dans les chambres disponibles de la résidence, qui peuvent être de 3 types :

- **simple :** ne peut accueillir qu'un élève à la fois
- **binômée :** petite chambre pouvant accueillir deux élèves
- **double :** grande chambre pouvant accueillir deux élèves

En termes de capacité de résidents, les chambres binômée et double ne sont pas différentes, mais leur distinction fera sens dans la mesure où les élèves pourront exprimer des préférences en termes de type de chambre.
Naturellement, une chambre ne peut pas recevoir plus de personnes que sa capacité maximale.

On fait l'hypothèse que chaque élève formule une seule demande.
Une demande se compose :

- du genre de la personne demandeuse (homme, femme ou non précisé) ;
- d'une éventuelle préférence en termes de type de chambre. Cette préférence peut-être «stricte», auquel cas l'étudiant n'est pas accepté si sa préférence ne peut pas être satisfaite ;
- d'une éventuelle demande d'être en collocation avec un candidat précis ;
- de critères de sélection :
  - distance entre le logement du demandeur et le campus des Ponts ;
  - si la personne est boursière ou non ;
  - rang dans le shotgun.

Classification des critères

Les éléments constitutifs d'une demande cités ci-dessus correspondent en fait à des critères de choix et de répartition des élèves. On introduit une classification de ces critères :

- Les critères dits « primaires » sont les critères qui ne dépendent que de l'étudiant et servent à la sélection des candidats qui auront droit à une chambre : ce sont la **distance des Ponts**, le caractère **boursier** ou non et le **shotgun**.

— Les critères dits « secondaires » sont les critères qui servent à répondre aux préférences des élèves une fois ceux-ci sélectionnés : **préférence de type de chambre** et **demande à être avec un ami**.
— Les critères « de contrainte » sont des conditions à respecter strictement, à savoir les **contraintes de genre** et **préférences de chambre strictes**.

Description des critères

Les critères primaires et secondaires sont quantifiés par des coefficients (bonus ou malus) appliqués lors de leur (non)-respect (*cf.* section 2.2). Une répartition des demandes dans les chambres se voit ainsi attribuée un score qui quantifie la qualité de cette solution.

Critères primaires :
— Pour le critère de distance, on établit deux paliers associés à deux bonus croissants. Ainsi, un élève peut être à distance courte des Ponts (<50km), à distance moyenne (entre 50km et 800km) ou à grande distance (>800km).
— Le simple fait d'être boursier applique le bonus associé.
— Une pénalité est attribuée en fonction croissante du rang dans le shotgun, mais cette pénalité reste toujours inférieure en valeur absolue aux autres critères primaires.

Le choix de ces coefficients permet de trier les critères primaires par importance : distance > boursier > shotgun, ce dernier ne servant qu'à départager des élèves de profils égaux par ailleurs.

Critères secondaires :
— La préférence de type de chambre s'exprime comme un bonus appliqué en cas de respect de la préférence.
— Un bonus est appliqué en cas de respect d'une demande d'être ensemble. À noter que l'on vérifie en amont que les demandes de colocation sont compatibles (chaque personne a demandé à être avec l'autre). Toutes celles qui ne le sont pas seront traitées comme des demandes normales.

Critères de contrainte :
— Contrainte de genre : deux personnes de genres différents ne peuvent être mis dans la même chambre, sauf s'ils ont demandé à être ensemble. Si un colocataire ne précise pas son genre, il peut se retrouver avec quelqu'un de n'importe quel genre.
— Préférence de chambre stricte : une personne ne peut se retrouver dans une chambre d'un autre type que celui préféré si la préférence est stricte.

2 Modélisation

2.1 Données d'entrée

— N_R chambres de 3 types :

$$\forall j \in [\![1, N_R]\!], \quad c_j = \begin{cases} 0 & \text{si simple} \\ 1 & \text{si binômée} \\ 2 & \text{si double} \end{cases}$$

— N_D préférences de type de chambre :

$$\forall i \in [\![1, N_D]\!],\ \forall k \in [\![0, 2]\!], \quad p_{ik} = \begin{cases} 1 & \text{si préférence pour le type } k \\ 0 & \text{sinon} \end{cases} \qquad (\text{tq } \sum_{k=0}^{2} p_{ik} \leqslant 1)$$

En outre, l'élève peut préciser s'il accepte d'avoir un autre type de chambre si sa préférence ne peut être satisfaite (dans le cas contraire la demande est dite "stricte") :

$$\forall i \in [\![1, N_D]\!], \quad q_i = \begin{cases} 1 & \text{si la demande } i \text{ est stricte} \\ 0 & \text{sinon} \end{cases}$$

— Le genre des N_D demandes :

$$\forall i \in [\![1, N_D]\!], \quad g_i = \begin{cases} -1 & \text{si homme} \\ 0 & \text{si ne se prononce pas} \\ 1 & \text{si femme} \end{cases}$$

— Les éventuelles volontés de loger avec un candidat précis :

$$\forall i_1 \in [\![1, N_D]\!],\ \forall i_2 \in]\!]i_1, N_D]\!] \quad b_{i_1 i_2} = \begin{cases} 1 & \text{si } i_1 \text{ et } i_2 \text{ demandent à être ensembles} \\ 0 & \text{sinon} \end{cases}$$

Satisfaction des critères

— N_D indications du critère boursier :

$$\forall i \in [\![1, N_D]\!], \quad s_i = \begin{cases} 1 & \text{si le candidat } i \text{ est boursier} \\ 0 & \text{sinon} \end{cases}$$

— N_D indications de proximité des Ponts :

$$\forall i \in [\![1, N_D]\!], \quad d_i = \begin{cases} 1 & \text{si le candidat } i \text{ est à plus de } 50km \text{ des Ponts} \\ 0 & \text{sinon} \end{cases}$$

— N_D indications de résidence hors de la métropole :

$$\forall i \in [\![1, N_D]\!], \quad f_i = \begin{cases} 1 & \text{si le candidat } i \text{ est à plus de } 800km \text{ des Ponts} \\ 0 & \text{sinon} \end{cases}$$

— N_D rangs dans le shotgun :

$$\forall i \in [\![1, N_D]\!], \quad r_i \in [\![1, N_D]\!]$$

2.2 Paramètres

On introduit les paramètres de pénalité et bonus liés aux critères de sélection :
— B_p : bonus sur la satisfaction de la préférence de type de chambre.
— B_b : bonus sur la satisfaction d'une demande d'être ensemble.
— B_s : bonus sur la satisfaction de la demande d'un boursier.
— B_d : bonus sur la satisfaction de la demande d'un élève loin des Ponts.
— B_f : bonus sur la satisfaction de la demande d'un élève hors métropole.
— P_r : pénalité sur le rang dans le shotgun.
Les valeurs de ces paramètres sont choisies comme suit :

B_f	B_d	B_s	P_r	B_p	B_b
1	0.3	0.2	1×10^{-4}	1×10^{-6}	1×10^{-6}

Détail du choix des paramètres :
— $B_p = 1 \times 10^{-5}$: la préférence souple de chambre ne doit pas être déterminante dans la sélection d'un candidat. Simplement, s'il y a le choix, le candidat doit préférentiellement être placé dans la chambre qui a sa préférence.
— $B_b = 1 \times 10^{-5}$: De même que pour la préférence de type de chambre, la demande d'être avec une connaissance ne doit pas être déterminante dans la sélection, et doit seulement influencer la répartition *a posteriori* des demandes retenues.

— $B_s = 0.2,\ B_d = 0.3,\ B_f = 1$: Ces critères dépendent uniquement de l'étudiant et sont les plus déterminants dans la sélection des demandes. Les valeurs sont choisies de sorte à classer ces critères par importance, le critère "vivre très loin" étant de loin le plus déterminant.
— $P_r = 1 \times 10^{-4}$: le critère de shotgun dépend uniquement de l'élève et est déterminant dans la sélection, mais il sert uniquement à départager des demandes dont les dossiers sont comparables par ailleurs. Le choix d'ordre de grandeur permet donc d'éviter que ce critère dépasse les autres en importance, sachant que le nombre de demande ne dépassera jamais les 10000.

Remarque : la somme des pénalités doit être inférieure à 1, sinon l'algorithme préfère laisser une chambre vide que de l'occuper par une personne avec toutes les pénalités

2.3 Variables

$$\forall (i,j) \in [\![1, N_D]\!] \times [\![1, N_R]\!], \quad x_{ij} = \begin{cases} 1 & \text{si la demande } i \text{ est satisfaite avec la chambre } j \\ 0 & \text{sinon} \end{cases}$$

et des variables de linéarisation des produits :

$$\forall i_1 \in [\![1, N_D]\!],\ \forall i_2 \in]\!]i_1, N_D]\!],\ \forall j \in [\![1, N_R]\!], \quad z_{i_1 i_2 j} = \begin{cases} 1 & \text{si les demandes } i_1 \text{ et } i_2 \text{ sont satisfaites avec la chambre } j \\ 0 & \text{sinon} \end{cases}$$

2.4 Problème de maximisation

$$\max_{x,z} \quad \sum_{i,j} x_{ij} + B_d \sum_{i,j} x_{ij} d_i + B_f \sum_{i,j} x_{ij} f_i + B_s \sum_{i,j} x_{ij} s_i - P_r \sum_{i,j} x_{ij} r_i + \sum_{i,j} x_{ij} B_p p_{ic_j} + B_b \sum_{i_1 < i_2, j} z_{i_1 i_2 j} b_{i_1 i_2}$$

$$\text{s.c.} \quad \begin{cases} \forall (i,j), \quad x_{ij} \geqslant 0 \text{ et } x_{ij} \leqslant 1 \\ \forall i, \quad \sum_j x_{ij} \leqslant 1 \\ \forall j, \quad \sum_i x_{ij} \leqslant \min(2, r_j + 1) \\ \forall (i_1, i_2, j) \text{ tq } i_1 < i_2, \begin{cases} z_{i_1 i_2 j} \geqslant 0 \\ z_{i_1 i_2 j} \leqslant x_{i_1 j} \\ z_{i_1 i_2 j} \leqslant x_{i_2 j} \\ z_{i_1 i_2 j} \geqslant x_{i_1 j} + x_{i_2 j} - 1 \end{cases} \\ \forall j \text{ tq } c_j >= 1, \quad \sum_{i_1 < i_2} z_{i_1 i_2 j} |g_{i_1} g_{i_2} (g_{i_1} - g_{i_2})| (1 - b_{i_1 i_2}) = 0 \\ \forall (i,j), \quad (1 - p_{ic_j}) q_i x_{i,j} = 0 \end{cases}$$

3 Algorithmes pour cas simples

3.1 Seulement des chambres simples

On a un ensemble D de $N_D = n$ demandes (sans préférence rigide autre que simple) et un ensemble R de $N_R = m$ chambres simples, avec $n \geqslant m$. Dans ce cas, il suffit de réaliser un tri sur les demandes par ordre décroissants de scores « semi-absolus» (score primaire + critères de préférence de chambre), puis de sélectionner les demandes dans l'ordre.

Algorithme 1 Exact

1: `sol` ← `[]`
2: Trier D par ordre décroissant de score «semi-absolu».
3: **pour** k allant de 0 à $m - 1$ **faire**
4: `sol` ← $\{D[k] \in R[k]\}$
5: **fin pour**
6: Renvoyer `sol`

La complexité de cet algorithme est celle d'un tri à laquelle on ajoute l'affectation linéaire dans les chambres, donc en $O(n\log(n)+m) = O(n\log(n))$

3.2 Seulement des chambres doubles

On a un ensemble D de n demandes (sans préférence rigide autre que double) et un ensemble R de m chambres doubles avec $n \geqslant 2m$. Un algorithme exact mais naïf consiste à tester toutes les arrangements possibles des demandes en $2m$ paires sélectionnées.

Algorithme 2 Exact - complexité aberrante

1: `sol` $\leftarrow$ `[]` (dont la portée s'étend à l'algorithme de récursion 3)
2: `val` $\leftarrow -\infty$ (dont la portée s'étend à l'algorithme de récursion 3)
3: **pour** chaque partie S à $2m$ éléments de D **faire**
4: Appliquer la récursion (algorithme 3) pour S, `[]`, 0
5: **fin pour**
6: Renvoyer `sol`

Algorithme 3 Récursion de partition

Entrée: P_{rec}, $\texttt{sol}_{rec}$, $\texttt{val}_{rec}$
1: **si** $P_{rec} ==$ `[]` **alors**
2: **si** $val_{rec} < val$ **alors**
3: $val \leftarrow val_{rec}$
4: $sol \leftarrow sol_{rec}$
5: **fin si**
6: **fin si**
7: **pour** $0 \leqslant i_1 < \text{len}(P_{rec})$ **faire**
8: **pour** $i_1 < i_2 < \text{len}(P_{rec})$ **faire**
9: $d_1 \leftarrow P_{rec}[i_1]$
10: $d_2 \leftarrow P_{rec}[i_2]$
11: **si** la paire (d_1, d_2) est compatible en genre **alors**
12: Appliquer la récursion pour $P_{rec} \setminus \{d_1, d_2\}$, $\texttt{sol}_{rec} \cup \{(d_1, d_2)\}$, $\texttt{val}_{rec} + \text{score}(d_1, d_2)$
13: **fin si**
14: **fin pour**
15: **fin pour**

$\rightarrow$ Complexité en $O(\binom{n}{2m}(2m)!)$.
Pour être exact, en notant $C(m)$ la complexité de l'algorithme 3 pour un ensemble à partitionner de taille $2m$:

$$C(m) = (2m-1)C(m-1) = ... = (2m-1)(2m-3)...1 = \prod_{\substack{k=1 \\ k \text{ impair}}}^{k=2m} k$$

D'où une complexité en $O(\binom{n}{2m}(2m-1)(2m-3)...1)$.

3.3 Que des doubles et une simple

On a un ensemble D de n demandes (sans préférence rigide autre que double ou simple) et un ensemble R de m chambres doubles, ainsi qu'une chambre simple r_s, avec $n \geqslant 2m+1$.

Algorithme 4 Exact

```
sol ← []
val ← −∞
pour chaque demande d avec une préférence compatible avec une chambre simple faire
    score ← score(d)
    sol_step, val_step ← résultat de l'algorithme 2 pour D \ {d} et R \ {r_s}
    si val_step + score > val alors
        val ← val_step + score
        sol ← sol_step ∪ {d ∈ r_s}
    fin si
fin pour
Renvoyer sol, val
```

→ complexité en $n \times C_2(n, m)$ où C_2 est la complexité de l'algorithme 2.

4 NP-complétude

D'après Wikipédia [1], pour un problème de la forme

$$\min_{x \in \mathcal{R}^n} \frac{1}{2} x^T Q x + c^T x \quad \text{s.c.} Ax \leqslant b \tag{P}$$

où, pour $m \in \mathbb{N}$ et $n \in \mathbb{N}$:
- Q est une matrice symétrique réelle $n \times n$
- $c \in \mathbb{R}^n$
- $A \in \mathbb{R}^{m \times n}$
- $b \in \mathbb{R}^m$,

le problème (P) est polynomial si Q est définie positive, et NP-hard si Q est indefinite (càd ni définie positive/négative, ni semi-définie positive/négative).
On adopte l'abus de notation :

$$x_{i,j} = x_{(j-1)N_D+i}$$

$$c_{i,j} = c_{(j-1)N_D+i}$$

$$Q_{i_1,i_2,j_1,j_2} = Q_{(j_1-1)N_D+i_1,(j_2-1)N_D+i_2}$$

On prend alors :

$$c_{i,j} = 1 + B_d d_i + B_f f_i + B_s s_i - P_r r_i + B_p p_{ic_j} - P_{sp}(1 - p_{ic_j}) q_i$$

et :

$$Q = \begin{pmatrix} \tilde{Q} & 0 & \dots & 0 \\ 0 & \tilde{Q} & \dots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ 0 & \dots & 0 & \tilde{Q} \end{pmatrix}$$

où :

1. `https://en.wikipedia.org/wiki/Quadratic_programming#Complexity` explique qu'il suffit que **Q** soit indefinite pour avoir la NP-difficulté, mentionnant la source Sahni, S. (1974). "Computationally related problems". Cependant, je n'ai pas trouvé l'information dans cette source. J'ai trouvé uniquement la source Pardalos, Panos M. ; Vavasis, Stephen A. (1991). "Quadratic programming with one negative eigenvalue is (strongly) NP-hard", qui parle du cas où Q a une valeur propre strictement négative. En outre, le wikipédia français de la même page mentionne seulement que dans le cas convexe, la complexité est polynomiale, et que dans le cas général le pb est NP-dur, mais pas que dans tous les cas non-convexes le pb est NP-dur... Je doute donc que le fait que Q soit indefinite soit suffisant...

$$\tilde{Q}_{i_1,i_2} = \begin{cases} B_b b_{i_1,i_2} - P_g |g_{i_1} g_{i_2} (g_{i_1} - g_{i_2})| (1 - b_{i_1 i_2}) & \texttt{si } i_1 \neq i_2 \\ 0 & \text{sinon} \end{cases}$$

(de sorte que $Q_{i_1 i_2 j_1 j_2} = \tilde{Q}_{i_1 i_2}$ si $j_1 = j_2$ et 0 sinon).
On a bien :

$$\frac{1}{2} x^T Q x = \frac{1}{2} \sum_j \sum_{i_1 \neq i_2} x_{i_1 j} x_{i_2 j} Q_{i_1 i_2 j j} = \sum_j \sum_{i_1 < i_2} x_{i_1 j} x_{i_2 j} \tilde{Q}_{i_1 i_2}$$

$\frac{1}{2} x^T Q x + c^T x$ correspond alors bien à la fonction objectif de la formulation pénalisée de notre problème.
La matrice A et le vecteur b se déduisent aisément des contraintes du problèmes.
Pour montrer que Q est indefinite, il suffit de trouver x et x' tels que $x^T Q x > 0$ et $x'^T Q x' < 0$. On peut commencer par remarquer, en notant $G_{i_1 i_2} = |g_{i_1} g_{i_2} (g_{i_1} - g_{i_2})|$, que :

- Si $\exists (i_1, i_2), b_{i_1 i_2} = 1$, prendre $x^T = \begin{pmatrix} 0 & \dots & 1 & \dots & 1 & \dots 0 \end{pmatrix}$ (avec les 1 en position i_1 et i_2) et $x'^T = \begin{pmatrix} 0 & \dots & 1 & \dots & -1 & \dots 0 \end{pmatrix}$
- Sinon,
  - si $\exists (i_1, i_2), G_{i_1 i_2} > 0$, on prend les mêmes x et x'.
  - sinon, $Q = 0$ et le problème est linéaire (et peu intéressant...).

On montre ainsi que le relâché continu de la formulation pénalisée de notre problème est NP-hard (sous réserve que le wikipédia anglais est exact, sinon il faut exhiber une valeur propre > 0 de Q (en effet, on a ici un problème de maximisation et non de minimisation).
Le problème non relâché est-il alors NP-hard ?...

5 Borne supérieure

Question : un gain sur une meilleure optimisation des critères secondaires peut-il compenser une perte sur le remplacement d'une personne par une autre de moins bon classement ?

Données : n demandes, m_S chambres simples, m_D chambres doubles, m_B chambres binomées.

Solution :

- Perte minimale lors d'un remplacement : P_r
- Gain maximal lors de l'optimisation des critères secondaires : $B_b(m_D + m_B) + B_p(m_S + m_D + m_B)$

$\rightarrow$ compensation $\iff B_b(m_D + m_B) + B_p(m_S + m_D + m_B) > P_r$

A.N. : Pour $m_S = m_D = m_B = 100$ et $P_r = 10^{-4}$, sous l'hypothèse $B_p = B_b$, on a compensation ssi $B_p + B_b > 2.10^{-7}$