TME - semaines 1 à 4

1 Problème et affectation

On s'intéresse au problème d'affection des étudiants dans le master informatique de Sorbonne Université. Ce master comporte 9 parcours : ANDROIDE, BIM, DAC, IMA, RES, SAR, SESI, SFPN, STL. Vous pouvez aller sur le site du master, à l'adresse `www-master.ufr-info-p6.jussieu.fr/`, pour avoir une description de ces différents parcours. Le recrutement dans chacun de ces parcours se fait selon les dossiers, et selon les vœux exprimés par les étudiants. L'objectif de cet exercice est d'utiliser l'algorithme de Gale-Shapley pour déterminer des affectations stables dans ce problème. Commencer par récupérer les fichiers "PrefSpe.txt" et "PrefEtu.txt" sur le Moodle de l'UE.

Fichier PrefSpe.txt : il correspond aux préférences des 9 parcours sur un groupe fictif de 11 étudiants (les parcours sont numérotés de 0 à 8, et les étudiants de 0 à 11). La première ligne contient le nombre d'étudiants. La deuxième contient les capacités d'accueil des parcours (2 pour ANDROIDE, 1 pour BIM, etc). Les 9 lignes suivantes correspondent aux préférences des parcours sur les étudiants. Par exemple, la ligne "0 ANDROIDE 7 9 [...] 2" signifie que l'étudiant 7 est classé premier par le parcours numéroté 0 (ANDROIDE), que l'étudiant 9 est classé deuxième par ce même parcours, etc, et que l'étudiant 2 est le dernier.

Fichier PrefEtu.txt : il correspond aux préférences des étudiants. La première ligne contient le nombre d'étudiants (ici 11). Puis, le fichier contient une ligne par étudiant donnant ses préférences sur les parcours. Par exemple, la ligne "0 Etu0 5 [...] 4" signifie que l'étudiant numéro 0, nommé Etu0, a classé le parcours SAR (numéro 5) premier, etc, et que le parcours RES (numéro 4) est le dernier de son classement.

Q1. En langage Python, écrivez une fonction lisant le fichier des préférences des étudiants sur les masters (PrefEtu.txt) et qui retourne une matrice des préférences correspondantes (ou une liste de listes). Écrivez une autre fonction pour les préférences des spécialités sur les étudiants.

Note : deux fichiers contenant quelques instructions utiles en Python vous sont fournis ("main.py" et "exemple.py"). Si vous n'avez jamais utilisé Python, lisez la fiche d'aide sur le langage python.

Q2. Codez l'extension de l'algorithme de Gale-Shapley au problème des hôpitaux "côté étudiants", et appliquez l'algorithme sur les deux fichiers tests.

Q3. Codez une adaptation de Gale-Shapley "côté parcours". Appliquez cet algorithme sur les deux fichiers tests.

Q4. Écrivez une méthode prenant en entrée une affectation (et les matrices de préférences), et renvoyant la liste des paires instables. Vérifiez que les affectations obtenues dans les questions précédentes sont bien stables.

2 Evolution du temps de calcul

Q5. Ecrivez deux méthodes prenant en paramètre un nombre n d'étudiants :

- l'une générant un tableau des préférences de ces n étudiants sur les 9 parcours du master (préférences aléatoires),
- l'autre générant un tableau des préférences des 9 parcours du master sur les n étudiants. Les préférences seront aléatoires. Pensez à définir les capacités d'accueil des parcours (la somme devant faire n). On pourra générer des capacités de manière déterministe (et par exemple à peu près équilibrées entre les parcours).

Q6. Mesurez le temps de calcul de votre algorithme de Gale-Shapley pour différentes valeurs de n (vous ferez plusieurs tests pour chaque valeur de n pour avoir une valeur significative). On pourra par exemple faire varier n de 200 à 2000 par pas de 200, et faire 10 tests pour chaque valeur de n. On pourra ensuite tracer une courbe représentant le temps de calcul (moyen) en fonction de n.

Q7. Quelle complexité obtient-on ? Est-ce cohérent avec la complexité théorique ?

Q8. Faites de même avec le nombre d'itérations. Combien fait-on d'itérations en moyenne ? Est-ce cohérent avec une analyse théorique de l'algorithme ?

3 Equité et PL(NE)

On souhaite maintenant trouver une solution (pas forcément stable) qui maximise l'utilité minimale des étudiants. Autrement dit, on veut trouver le plus petit k tel qu'il existe une affectation où tout étudiant a un de ses k premiers choix (utilité au moins $m - k$ pour chaque étudiant).

Q9. Écrivez, pour un certain k, un PLNE permettant de savoir s'il existe une affectation où tout étudiant a un de ses k premiers choix.

Q10. Sur l'exemple, et pour $k = 3$, écrivez une méthode permettant de générer un fichier .lp correspondant au PLNE, puis résolvez le PLNE à l'aide de Gurobi (voir la fiche d'aide sur la programmation linéaire). Existe-t-il une solution ?

Q11. Trouvez le plus petit k tel qu'il existe une solution. Décrivez une solution maximisant l'utilité minimale. Quelle était l'utilité minimale des solutions fournies par l'algorithme de Gale-Shapley (côté étudiants et côté parcours) ?

Q12. Écrivez un PLNE maximisant l'utilité totale des étudiants, puis résolvez-le avec Gurobi. Quelle utilité moyenne obtient-on ?

Q13. Soit k^* le plus petit k trouvé à la question 11. Écrire un PLNE maximisant l'utilité totale des étudiants parmi les solutions où chaque étudiant a un de ses k^* premiers choix.

Q14. Comparez les différentes solutions obtenues (GS côté étudiant, GS côté parcours, solutions des questions 11, 12 et 13) : stabilité, utilité moyenne, utilité minimale. Utilisez la question 4 pour les comparer aussi selon le nombre de paires instables.