

Application de l'algorithme de Ford Fulkerson

I – Le problème à résoudre

Tous les ans se tient le HackMIT sur tout u week-end, et les programmeurs qui y participent doivent être hébergé chez des hôtes.

Il faut trouver quel programmeur va dormir chez quel hôte, en fonction de plusieurs critères.

II – Les critères

Les hôtes/programmeurs peuvent être fumeur, non-fumeur (et dans ce cas ils peuvent tolérer ou non la cigarette).

Les hôtes peuvent avoir des animaux, et certains programmeurs peuvent être allergiques aux animaux.

Les hôtes peuvent avoir des chambres individuelles ou bien des dortoirs, et les programmeurs peuvent tolérer ou non de dormir avec des personnes du sexe opposé.

Les programmeurs peuvent avoir besoin d'être héberger soit le vendredi, soit le samedi, soit les deux soirs. De la même manière, les hôtes peuvent accueillir le vendredi, le samedi ou les deux soirs.

III – La résolution du problème

Le but est de résoudre ce problème en utilisant notre code initial, qui permet l'application de l'algorithme Ford Fulkerson à un graphe.

Il faut donc trouver un moyen de créer un graphe à partir du problème.

Pour cela nous avons décidé de partir de deux fichiers txt : un pour les hôtes, et un pour les hackers.

Les fichiers contiendront toutes les informations nécessaires à la construction du graphe.

Format des fichiers

```
1 h X
2
3 p X n X d X/Y a X f X t X
4 .
5 p X n X d X/Y a X f X t X
6 .
7 p X n X d X/Y a X f X t X
8 .
9 p X n X d X/Y a X f X t X
10 .
```

Fichier type Hôte

```
1 h X
2
3 n X m X a X f X t X
4 .
5 n X m X a X f X t X
6 .
7 n X m X a X f X t X
8 .
9 n X m X a X f X t X
10 .
```

Fichier type Programmeur

Lecture des fichiers

- « X » : champs à remplir par des chiffres
- « p » : nombre de places que l'hôte a dans sa maison/son appartement

- « n » : nombre de nuit que l'hôte peut recevoir/que le programmeur souhaite rester
- « d/X/Y » : nombre de dortoirs que l'hôte a, suivi du nombre de place par dortoir
- « a » : 1 si l'hôte à un animal, 0 sinon. 1 si le programmeur est allergique, 0 sinon/
- « f » : 1 si la personne fume, 0 sinon
- « t » : 1 si la personne tolère la cigarette, 0 si elle ne la tolère pas
- « . » : correspond au moment où l'on va créer le(s) nodes ainsi que le(s) arcs

Création des sommets et des arcs

Lors de la lecture du fichier « Hosts.txt », on va créer un node pour chaque hôte (donc à chaque ligne) et on va ajouter cet hôte dans la liste l_thosts (qui nous permettra par la suite de vérifier la compatibilité entre hôte et programmeur). Cette liste contient donc des structures thosts, contenant tous les éléments nécessaires aux tests de compatibilité (allergie, fumeur etc.).

Lors de la lecture du fichier « Hackers.txt », on va créer un node pour chaque programmeur (donc à chaque ligne), puis on va étudier la compatibilité de ce programmeur avec chacun des hôtes. Pour chaque hôte compatible, on va créer un arc entre le programmeur et l'hôte compatible.

Remarque : tous les hôtes seront reliés au puit et tous les programmeurs seront reliés à la source.

Exemple concret pour une seule nuit

Soit les personnes suivantes :

- Hôte 1 : possède une chambre individuelle et fume.
- Hôte 2 : possède une chambre individuelle, ne fume pas et ne tolère pas la cigarette.
- Hôte 3 : possède un dortoir de 2 places, ne fume pas et ne tolère pas la cigarette.
- Programmeur 1 : fume
- Programmeur 2 : souhaite dormir dans une chambre non-mixte, ne fume pas mais tolère la cigarette
- Programmeur 3 : Ne fume pas mais tolère la cigarette
- Programmeur 4 : Ne fume pas mais tolère la cigarette

Les fichiers txt correspondants :

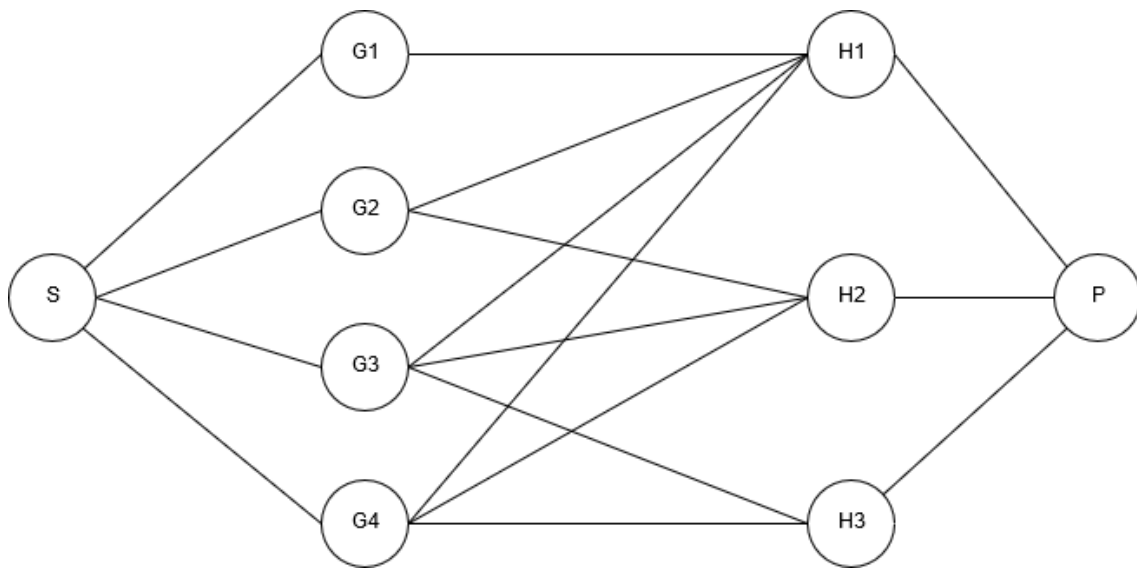
```
1 h 3
2
3 p 1 n 1 d 0 a 0 f 1 t 1
4 .
5 p 1 n 1 d 0 a 0 f 0 t 0
6 .
7 p 2 n 1 d 1/2 a 0 f 0 t 0
8 .
```

Hosts.txt

```
1 h 4
2
3 n 1 m 1 a 0 f 1 t 1
4 .
5 n 1 m 0 a 0 f 0 t 1
6 .
7 n 1 m 1 a 0 f 0 t 1
8 .
9 n 1 m 1 a 0 f 0 t 1
10 .
```

Hackers.txt

Le graphe correspondant :



Les flows et capacités :

Tous les arcs entre la source et les programmeurs auront un flot initial de 0 et une capacité de 1 : un sommet = un programmeur.

Tous les arcs entre les hôtes et le puit auront un flot initial de 0 et une capacité égale au nombre de couchages disponibles chez l'hôte.

Les arcs entre les programmeurs et les hôtes auront également un flot initial de 0 et une capacité égale au nombre de couchages disponibles.