TELECOM Nancy 2A 2017-18
Graphes et Recherche Opérationnelle

Feuille 4 : Flot maximal dans un graphe

Exercice 1. *Parcours dans un graphe orienté*

Le but de cet exercice est d'étudier deux façons différentes de parcourir un graphe orienté.

1) Parcours en profondeur (DFS, Depth First Search) avec une pile.
Avec ce parcours, on cherche à explorer le sommet le plus éloigné du sommet de départ, à la condition qu'il soit le successeur d'un sommet qui a déjà été exploré. Dans le cas du parcours en profondeur, on utilise une pile pour l'exploration des sommets (de type LIFO, Last In, First Out c'est-à-dire : "dernier entré, premier sorti"). A partir d'un sommet courant, on empile et on marque le premier successeur non encore marqué. Le sommet courant est ensuite enlevé de la pile et on utilise le sommet suivant de la pile pour continuer. On procède ainsi jusqu'à ce que la pile soit vide. L'algorithme générique s'écrit :

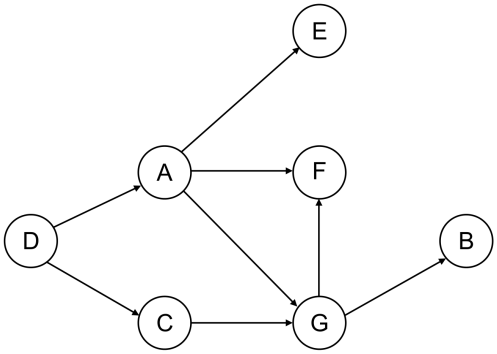
```
Initialement, tous les sommets sont non marqués et la pile est vide.
Tant qu'il existe[1] un sommet S non marqué,
      insérer S au sommet de la pile. Marquer S.
      Tant que la pile n'est pas vide
          Soit Q sommet situé en haut de la pile.
          S'il existe[1] un sommet P non marqué, successeur de Q
          Alors insérer P en haut de la pile. Marquer P.
          Sinon supprimer Q de la pile.
      Fin Tant que
Fin Tant que
```

2) Parcours en largeur (BFS, Breath First Search) avec une file.
Dans ce type de parcours, on essaie toujours d'explorer le sommet le plus proche du sommet de départ et qui n'a pas déjà été visité. A partir d'un sommet courant, on liste et on marque tous ses successeurs non encore marqués. Le sommet courant est ensuite enlevé de la liste et on utilise le premier sommet de la liste pour continuer. On utilise une structure de file d'attente (de type FIFO, First in, first out c'est-à-dire "premier entré, premier sorti"). L'algorithme générique s'écrit :

```
Initialement, tous les sommets sont non marqués et la file est vide.
Tant qu'il existe un sommet S non marqué,
      insérer S dans la file. Marquer S.
      Tant que la file n'est pas vide
          - Supprimer de la file le sommet Q situé en tête de file.
          - Insérer successivement dans la file tous les sommets P non marqués,
            successeurs de Q.
          - Marquer les sommets P.
      Fin Tant que
Fin Tant que
```

1. S'il y a plusieurs sommets concernés, on prend le premier dans l'ordre lexicographique.

1. En partant du sommet D, parcourir le graphe ci-dessous en utilisant les deux méthodes différentes proposées. Pour chaque parcours, vous afficherez un tableau représentant l'état de la pile/file à chaque itération et vous indiquerez l'ordre de parcours des sommets.



2. Ecrire une version récursive de l'algorithme de parcours en profondeur.
3. *Parcours profondeur DFS et tri topologique.*

   Le tri topologique d'un graphe *sans cycle*[2] consiste à ordonner les numéros des sommets de telle sorte que chaque sommet du graphe est toujours visité avant ses successeurs. Dans le parcours en profondeur DFS, en sauvegardant successivement les numéros des sommets dépilés on obtient un tri topologique des sommets en ordre inverse.

   (a) A partir du parcours en profondeur DFS réalisé à la question 1, donner un tri topologique du graphe ci-dessus.

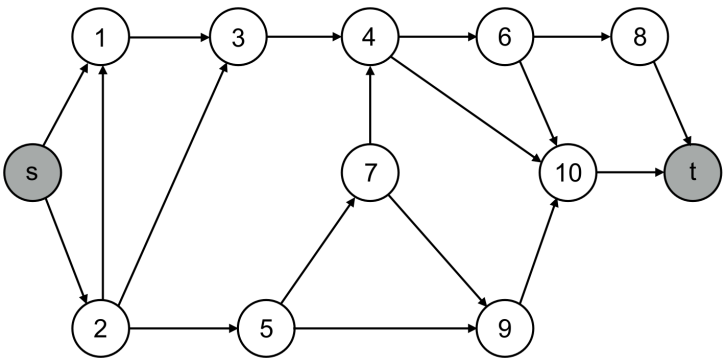
   (b) Modifier l'algorithme de parcours en profondeur DFS pour réaliser un tri topologique des sommets d'un graphe sans cycle.

4. *Parcours largeur BFS et distance minimale.*

   La distance entre deux sommets d'un graphe orienté est définie comme le nombre d'arêtes du chemin reliant les deux sommets. On considère un graphe orienté tel que tous les sommets sont accessibles à partir d'un sommet initial s par un chemin. Le parcours en largeur (BFS) permet de calculer la distance minimale de tous les sommets au sommet initial. Pour calculer la distance minimale, on introduit la fonction L telle que $L(s) = 0$ pour le sommet s initial et on fait $L(P) = L(Q) + 1$ si on rencontre (et marque) le successeur non-marqué P de Q.

   (a) Modifier l'algorithme du parcours en largeur (BFS) pour calculer les distances minimales de chacun des sommets du graphe au sommet initial s.

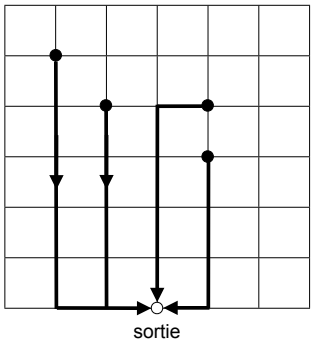
   (b) Déterminer en effectuant un parcours largeur (BFS) la distance minimale de s à t pour le graphe suivant.



2. un cycle dans un graphe orienté est un chemin dont les deux sommets extrémités sont identiques

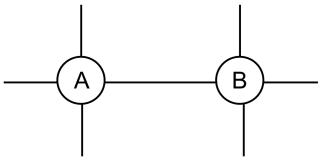
Exercice 2. *Modélisation par flot maximal dans un graphe*

On considère un problème d'évacuation de p personnes disposées sur une grille $n \times n$. Les personnes doivent rejoindre la sortie E en suivant des trajets **qui ne se coupent pas à l'intérieur de la grille** (voir la figure ci-dessous ; les disques noirs représentent les personnes).



Une grille $n \times n$ est un graphe non-orienté de n^2 sommets correspondants aux noeuds de la grille. Tous les sommets du graphe - à l'exception des sommets du bord de la grille - ont exactement 4 voisins. On veut modéliser ce problème d'évacuation comme un problème de flot maximal dans un graphe. A chaque arête du graphe, on associe un flot qui vaut 1 si un trajet d'évacuation emprunte cette arête et vaut 0 sinon. La contrainte que les trajets d'évacuation à l'intérieur de la grille ne doivent pas se couper est prise en compte en rendant le graphe orienté et en introduisant de nouveaux sommets : à chaque sommet S situé à l'intérieur de la grille, on introduit un nouveau sommet S' relié à S par une arête orientée avec une capacité de 1, de sorte que la somme des flots entrants en S ne peut dépasser 1. Il y aura ainsi au plus un trajet d'évacuation passant pas S. Toutes les arêtes entrantes vers S dans le graphe initial passent désormais par S'. Enfin, on ajoute un sommet source S_0 relié à tous les sommets où sont positionnées les personnes, par des arêtes orientées avec une capacité de 1.

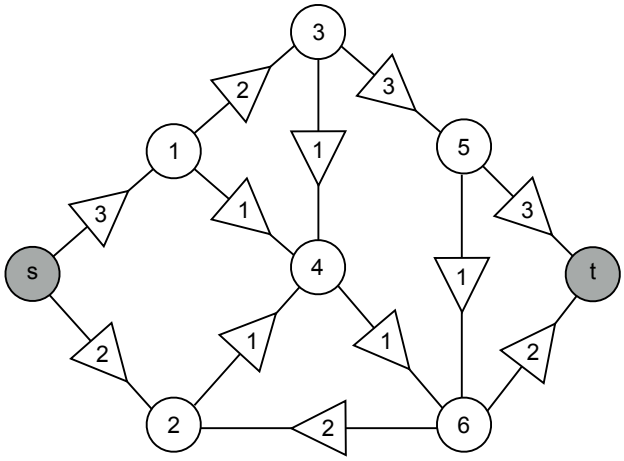
1. Effectuer la transformation décrite ci-dessus sur deux sommets A et B intérieurs à la grille.



2. Pour une grille 3×3, écrire le graphe complété pour le problème de flot maximal.

Exercice 3. *Flot maximal dans un graphe*

Calculer le flot maximal dans le graphe valué suivant, à l'aide de l'algorithme de Ford-Fulkerson.



Exercice 4. *Réseau commercial*

Une entreprise gère un réseau commercial qui est composé de 3 guichets de prises de commandes (G_1, G_2, G_3), de 2 ateliers de préparation des commandes (A_1, A_2) et de deux centres de distribution (C_1, C_2). Le réseau est constitué de la façon suivante :

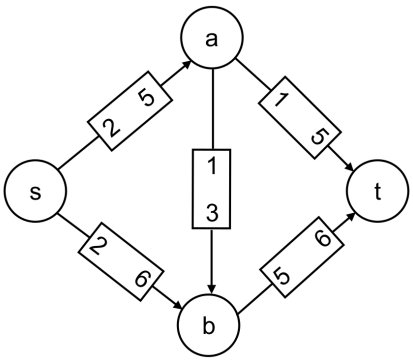
- chaque guichet peut passer commande aux 2 ateliers mais le faible nombre de personnels aux guichets limite à 20 le nombre maximum de commandes passées en un jour entre un guichet et un atelier.
- L'atelier A_1 fournit uniquement le centre C_1 et A_2 fournit uniquement C_2.
- Le centre C_2 peut transférer sur C_1 une partie de la production reçue, sans modifier sa capacité de distribution.
- Les capacités journalières de prises de commandes des guichets G_1, G_2 et G_3 sont de 30, 30 et 10 respectivement.
- Les capacités journalières de préparation des commandes des ateliers A_1 et A_2 sont de 10 et 60 respectivement.
- Les capacités journalières de distribution des centres C_1 et C_2 sont de 30 et 50.

L'entreprise souhaite évaluer la capacité journalière du réseau c'est-à-dire le nombre maximum de commandes qui peuvent être prises, préparées et distribuées.

1. Etablir le graphe associé au réseau décrit ci-dessus.
2. Utiliser l'algorithme de Ford-Fulkerson pour déterminer le nombre maximum de commandes que l'entreprise peut traiter par jour. Interpréter les résultats obtenus.
3. Déterminer une coupe minimale.

Exercice 5. *Flot maximal avec capacités supérieures et inférieures*

On considère le graphe suivant $G = (E, \Gamma, (\{\alpha_{ij}\}, \{\beta_{ij}\}))$ valué par des capacités supérieures et inférieures (non nécessairement nulles).



On rappelle qu'un flot $\{f_{ij}\}$ est réalisable sur G si $\alpha_{ij} \leq f_{ij} \leq \beta_{ij}$ pour tout $(i,j) \in \Gamma$.

1. *Flot réalisable sur G.*
   - Considérer le changement de variable $f'_{ij} = f_{ij} - \alpha_{ij}$ pour se ramener au cas de flots positifs dans un graphe auxiliaire G' associé à G. Ce graphe G' possède une seule source s' et un seul puits t' de sorte que déterminer un flot réalisable sur G est équivalent à déterminer un flot maximal sur G' pour lequel toutes les arêtes sortant de la source s' et toutes les arêtes arrivant au puits t' sont saturées.
   - Déterminer le flot maximal sur G' par l'algorithme de Ford-Fulkerson. Initialiser le flot $\{f'_{ij}\}$ en tenant compte des capacités des arêtes liées à la source s' et au puits t' de G'.
   - En déduire un flot réalisable sur G.
2. Expliquer comment modifier l'algorithme de Ford-Fulkerson pour prendre en compte les capacités inférieures.
3. Déterminer le flot maximal sur G.