AMD5
TD n° 4 : Tri topologique, Composantes fortement connexes

I) Tri topologique

Exercice 1 : Graphes Orientés Acycliques et Tri Topologique

Un graphe orienté acyclique (en anglais Directed Acyclic Graph, ou DAG) est un graphe orienté qui ne contient aucun cycle orienté.

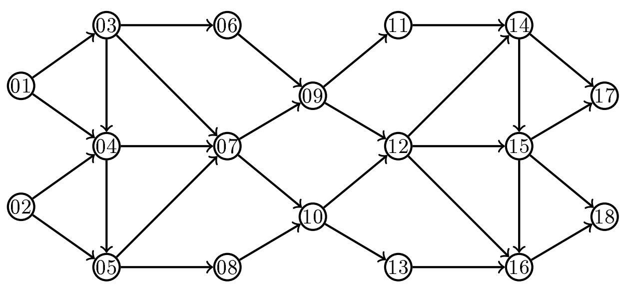
1. Montrer que si C est un cycle dans un graphe orienté G, alors dans toute exécution de parcours en profondeur de G, un des arcs de C sera un arc retour (on pourra utiliser la caractérisation des arcs retour par les fonctions `pre` et `post`). En déduire une modification de l'algorithme de parcours en profondeur qui permet de décider si un graphe orienté est un DAG.

2. On rappelle qu'un *tri topologique* d'un graphe orienté $G = (V, E)$ est une énumération $(s_1, s_2, \ldots, s_n)$ des éléments de V telle que, pour tous i, j, on ait $(s_i, s_j) \in E \Rightarrow i < j$. Montrer que si un graphe orienté admet un tri topologique, alors il est nécessairement acyclique.

3. Montrer la réciproque en prouvant que si on effectue un parcours en profondeur d'un DAG et qu'on trie les sommets par ordre décroissant de la valeur de la fonction $post(v)$, on obtient un tri topologique du graphe.

4. Ceci fournit donc un algorithme pour construire le tri topologique d'un DAG. Quelle est sa complexité ?

5. Appliquer l'algorithme sur le graphe suivant.



6. Montrer que si un graphe orienté est tel que tout sommet a au moins un voisin entrant, alors il contient un cycle.

7. On vient donc de montrer qu'un DAG possède nécessairement une source (un sommet de degré entrant 0). En se basant sur cette observation proposer un autre algorithme permettant de construire le tri topologique d'un DAG. Quelle est sa complexité (on suppose que le graphe est représenté par liste d'adjacence) ?

8. On aurait aussi pu montrer que que tout DAG possède nécessairement un puits (degré sortant 0) et en déduire un algorithme pour le tri topologique qui construit l'ordre par la fin. Aurait-on obtenu une meilleure complexité ?

9. Améliorer la complexité de l'algorithme de la question 7 en construisant, puis en maintenant une liste de tous les sommets qui peuvent être la prochaine source choisie. Quelle est la nouvelle complexité ?

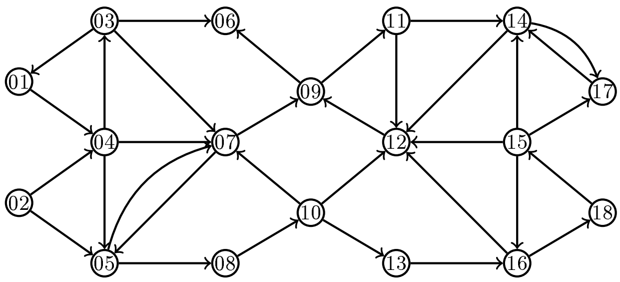
II) Composantes fortement connexes

Exercice 2 : Algorithme de calcul de CFC

1. Rappellez ce qu'est une composante connexe.
On rappelle l'algorithme de Kosaraju qui permet de calculer les composantes fortement connexes :

1. Faire un parcours en profondeur sur G pour obtenir les dates `f` de chaque sommet ;
2. Construire G^t (le graphe transposé de G) ;
3. Faire un parcours en profondeur sur (G^t) en énumérant les sommets par date post décroissante ;
4. Retourner la forêt de parcours obtenue par l'étape précédente : Chaque arbre est une CFC.

2. Appliquer l'algorithme de composantes fortement connexes sur le graphe suivant.



3. Comment implémenter les étapes 2 et 3 pour avoir un algorithme en temps linéaire ?

4. Écrire tout le pseudo-code d'un algorithme qui compte le nombre de composantes fortement connexes d'un graphe.

5. Montrer que si le graphe G est un DAG, chaque relance à l'étape 3 ne visite qu'un seul sommet. Dans quel ordre ces relances sont-elles faites ?

6. Peut-on remplacer l'étape 1 par un parcours en largeur ?

7. Et l'étape 3 ?

Exercice 3 : Chemins et cycles hamiltoniens

Definitions :

- On appelle *chemin hamiltonien* sur un graphe un chemin passant par chaque sommet du graphe une fois et une seule.
- Un *cycle hamiltonien* est un cycle passant par chaque sommet du graphe une fois et une seule.
- Un *tournoi* est un graphe orienté tel que pour tout paire de sommets u,v soit $(u, v) \in A$ soit $(v, u) \in A$.

1. Si un graphe possède un cycle hamiltonien, que dire de son nombre de composantes connexes ?

2. Réciproquement, si un graphe est fortement connexe, est-il nécessaire qu'il ait un cycle hamiltonien ?

3. On a vu exercice 2 qu'un graphe orienté acyclique (DAG, pour *Directed Acyclic Graph*) admet un tri topologique. Montrer qu'un DAG n'a qu'un seul tri si et seulement s'il possède un chemin hamiltonien.

4. Montrer qu'un tournoi G possède un chemin hamiltonien.

5. Montrer qu'un tournoi fortement connexe possède un cycle hamiltonien.