

TD n°7

Tri Topologique – Composantes Fortement Connexes

Exercice 1 [Tri Topologique]

Soit $G = (S, A)$ un graphe orienté ayant n sommets. On interprète un arc (x, y) comme un ordre entre ces sommets : x précède y ou x est avant y . L'objectif du *tri topologique* est de construire une liste de tous les sommets de G tel qu'aucun sommet n'apparaisse avant un de ses prédécesseurs. Un tri topologique est un ordre linéaire des sommets de G . Le tri topologique d'un graphe peut être vu comme un alignement de ses sommets le long d'une ligne horizontale tel que tous les arcs soient orientés de gauche à droite.

Question 1. Effectuer le tri topologique du graphe de la figure 1. Y a-t-il une solution unique ?

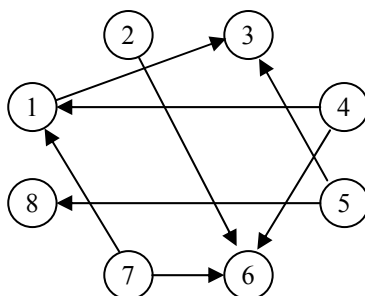


FIGURE 1 – Un graphe orienté

Question 2. Ecrire une fonction `DegreeInt` qui renvoie le nombre d'arcs de G ayant pour destination le sommet x (ceci est appelé le degré intérieur).

- dans le cas où le graphe est représenté par matrice d'adjacence
- dans le cas où le graphe est représenté par liste d'adjacence

Quelle est sa complexité ?

Question 3. L'idée du tri topologique est de choisir un sommet sans prédécesseur (son degré intérieur est nul), de l'ajouter à la liste résultat et de recommencer avec le graphe privé du sommet que l'on vient de sélectionner.

1. A partir de ceci écrire une fonction récursive utilisant `DegreInt` et `SupprimerSommet`
2. Définir un algorithme itératif, en utilisant un tableau D ayant n éléments tel que :
 $D[s] = -1$ si le sommet d'indice s a déjà été supprimé
 $D[s] =$ le degré intérieur de s mis à jour à partir des sommets déjà supprimés
3. Dérouler l'algorithme à la main.
4. Quelle est sa complexité? Peut-on faire mieux?

Question 4. Que se passe-t-il si le graphe contient un cycle? Modifier l'algorithme pour détecter les cycles dans un graphe.

Exercice 2 [CFC]

Soit $G = (S, A)$ un graphe orienté. On définit sur S la relation R comme suit :

pour tout couple $x, y \in S : xRy \iff$ il existe un chemin de x à y et un chemin de y à x

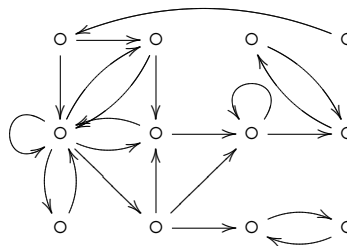
R s'appelle relation de *forte connexité*

Question 1. Montrer que R est une relation d'équivalence. Donner le nombre maximal et minimal de ses classes d'équivalence.

Soit G/R le graphe *quotient* qui a pour sommets les classes d'équivalence C_i de R et pour arcs les couples (C_i, C_j) tels qu'il existe un sommet x de C_i et un sommet y de C_j respectivement origin et extrémité d'un arc de G .

Question 2. Montrer que G/R est acyclique.

Question 3. Donner le graphe quotient du graphe ci-dessous :



On donne l'algorithme suivant (Algorithme de Tarjan) :

```

1 Procédure CFC( $G$  : graphe,  $s$  : sommet)
2 début
3    $couleur[s] := gris$ ;
4    $temps ++$ ;
5    $d[s] := temps$ ;
6    $P.$ Empiler( $s$ );
7   pour  $(s, y) \in A$  faire
8     si  $couleur[s] = blanc$  alors
9       CFC( $G, y$ );
10       $r_A[s] := \min(r_A[s], r_A[y])$ ;
11     sinon
12       si  $(d[y] < d[s]) \wedge (y \in P)$  alors
13          $r_A[s] := \min(r_A[s], d[y])$ ;
14    $couleur[s] := noir$ ;
15    $temps ++$ ;
16    $f[s] := temps$ ;
17   si  $(r_A[s] = d[s])$  alors
18      $nbcfc ++$ ;
19     tant que  $(P.nonVide) \wedge (d[P.tête] \geq d[s])$  faire
20        $y := P.tête$ ;
21        $P.$ depiler;
22        $NumCFC[y] := nbcfc$ ;
23 fin

```

```

1 Procédure Tarjan – CFC( $G$  : graphe)
2 début
3    $temps := 0$ ;
4    $nbcfc := 0$ ;
5    $P := PileVide$ ;
6   pour  $x \in S$  faire
7      $couleur[x] := blanc$ ;
8      $NumCFC[x] := undef$ ;
9   pour  $x \in S$  faire
10    si  $couleur[x] = blanc$  alors
11      CFC( $G, x$ );
12  retourner  $NumCFC[ ]$ ;
13 fin

```

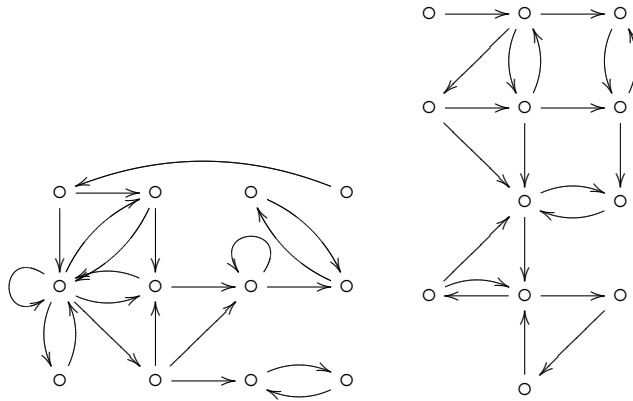


FIGURE 2 – Graphe orientés

Question 4. Exécuter l'algorithme de Tarjan sur les graphes de la figure 2

Question 5. Expliquer comment l'algorithme de Tarjan donne les composantes fortement connexes d'un graphe.