Projet Modélisation, graphes et algorithmes :

Séquence destructrice

Guillaume Aguenier
Vincent Renard
Thibaut Vacheron

1- Sequence2destructive du graphe B :

Solution :

[v2 v6 v1 v5 v4 v3 v7 v8]

3- Fonctionnement et complexité de la fonction :

Description de la fonction de détermination de séquence: isSeq2destr(seq)

La séquence à vérifier est parcourue par ses sommets : lors du parcours la vérification de la somme des sommets voisins (exceptés les sommets mémorisés car déjà parcourus auparavant) de couleur rouge est effectuée , si la somme vérifie la contrainte des “moins de 2 sommets rouges compris” (au plus 2) le sommet est mémorisé. Lorsque l’extrémité de la séquence est atteinte, la taille de la séquence doit correspondre au nombre de sommets du graphe.

Complexité de la fonction:

Parcours de la séquence :
n x(parcours des voisins = n-1)

$O(n * (n-1)) \mp O(n^2)$ avec $n = |V|$ (le nombre de sommets du graphe).

4- Construction de la séquence :

Tant que la taille de la séquence ne correspond pas au nombre de sommets du graphe : on ajoute un sommet ayant le minimum de sommets de couleur Rouge, puis ce sommet est oblitéré.

Entrée = G(V,E) // le graphe est séquence-compatible
Sortie = séquence2destructive
S ← {}
E’ = E #on ne cherche pas à modifier le Graphe ,
TantQue S.taille <> |V| :
map ← (e,redNexts) # (e > sommet , redNext > nombre de voisins rouges de e)*
minSommet ← map.getE_Min() # sommet ayant le moins de voisins rouge
S ∪ {minSommet}
E’ = E’ \ minSommet # $\forall\ \alpha\beta \in E' \mid (\ \alpha = minSommet \vee \beta = minSommet\ )\ E' \setminus \alpha\beta$
FinTantQue
Retourner S

*ici redNext est forcément < 3 car G est séquence-compatible

7- Degré :

Un sommet quelconque **q** possède une probabilité de **p** d'être relié à un sommet du graphe (probabilité qu'une arête existe). Le nombre de voisin de **q** correspond donc au nombre d'arêtes partant de **q**. Le maximum de voisins possible pour **q** étant **n-1** avec **n** le nombre de sommets, le degré attendu de **q** est: **p*n-1.**

9- tableau des valeurs :

	p=0.1	p=0.3	p=0.5	p=0.7
n=50	52.59%	19.87%	13.44%	10.70%
n=100	26.17%	10.14%	6.78%	5.34%

Annexe 1 - Indication d'exécution du programme :

Main : java -jar projetGrapheMain.jar [args]

Main

Main : java -jar projetGrapheMain.jar [args]

Après compilation, l'exécution de Main sans argument abouti à l'exécution de

- La fonction testA(nSommets,probabilite,rougirSommets)
- La fonction testB(nSommets,probabilite)

avec pour paramètres les valeurs par défaut définies au début de la classe Main.

Exécution avec arguments :

- testa nSommets probabilite probaRougirSommets
- testb nSommets probabilite

Avec (int) nSommets pour le nombre de sommets dans le graphe, (double) probabilite , probabilité qu'une arête soit représentée dans le graphe et (double) probaRougirSommets pour la probabilité qu'un sommet soit rouge.

MainTest :

MainTest : java -jar projetGrapheMainTest.jar

Contient des tests des différentes méthodes de la classe Graphe :
Il y a 2 façons de créer un Graphe:

De façon aléatoire, en joignant le nombre de sommets désirés , la propension des sommets à se voisiner et enfin celle de ces derniers à rougir.Tout usage de nombres non cohérents résultera en un graphe vide et terne.

Un graphe peut aussi être généré suivant une description dans un fichier plat.
Il y a quelques règles à suivre :

- Toute ligne commençant par un # est ignorée.
- Les sommets se nomment "vn" (avec n entier)
- Les couleurs des sommets s'expriment : "vn BLEU\ROUGE" dans une partie balisée par @Couleurs (n étant un entier).
- Les arêtes sont listées sommet-i voisins_de_i (sans réciprocité ni séparateur autre que' ') dans une partie balisée par @Aretes.

Un fichier txt pour chaque graphe d'exemple est fourni ainsi qu'un autre graphe test dans le répertoire /res

Annexe 2 - Choix d'implémentations et solutions proposées :

Un sommet est représenté par une chaîne de caractères (String) ,ce qui permet de l'identifier.
Le Graphe est représenté par sa ListeAdjacence ainsi qu'un objet CouleurSommet.

La liste d'adjacence du Graphe est représentée par une map<Sommet,ListeSommets>.

- La liste correspond à la liste des arêtes partants du sommet. (ses voisins)
- Accès à la liste des voisins d'un sommet en temps constant.

Annexe 3 - Diagramme de classe :