Yannis Sauzeau
2021-2022
Algorithmique avancée Master 1, Informatique

Mini-projet - Tri topologique d’un graphe orienté sans circuits

Arborescence du projet

```
sauzeau
├── graph
│   ├── graph-100.alists
│   ├── graph-101.alists
│   ├── graph-102.alists
│   ├── graph-103.alists
│   ├── graph-104.alists
│   ├── graph-105.alists
│   ├── graph-106.alists
│   ├── graph-107.alists
│   └── randomdag.jar
├── makefile
├── rapport.pdf
└── src
    ├── Graph.java
    ├── MyStack.java
    ├── TopologicalSorting.java
    └── Vertex.java
```

Dossier graph

Le dossier graph contient 7 graphes et un programme Java sous la forme d'une archive jar qui permet de générer des graphes à n sommets sans circuit avec un probabilité p d'existence d'arc.

Fichier makefile

Le fichier makefile permet de construire l'application avec la commande *make*.
La commande *make help* permet de voir les différentes commandes possibles.
Pour tester un des graphes du dossier graph, vous devez saisir la commande *make test<n>* avec n le numéro du graphe que vous voulez tester, par exemple pour tester le graphe graph/graph104.alists il suffit de faire *make test4*.
Pour tester un graphe aléatoire de 100 sommets avec une probabilité de 0.3 d'existence d'arc il faut utiliser la commande *make testRandom*.
Enfin pour tout tester, exécutez la commande *make testAll.*

Dossier src

src\Graph.java

La classe Graph contient un tableau de sommets qui représente le graphe construit à l'appel du constructeur par lecture de l'entrée standard.
Le format d'entrée des graphes respecte scrupuleusement le format spécifié dans le sujet du projet.

src\MyStack.java

La classe MyStack est une implémentation d'un pile d'entiers en Java, cette classe va servir à stocker les numéros des sommets lorsqu'ils sont traités dans le parcours en profondeur.

src\TopologicalSorting.java

La classe TopologicalSorting est la classe principale qui contient la fonction d'entrée du programme qui construit le graphe pour effectuer un tri topologique sur les sommets. Si un tri est trouvé alors il est retourné sur la sortie standard, sinon il existe un circuit qui lève un message d'erreur sur la sortie d'erreur et qui retourne le circuit sur la sortie standard.

src\Vertex.java

La classe Vertex représente les sommets du graphe. Chaque sommet est défini par un numéro. Chaque sommet est marqué d'une couleur, qui est verte lorsque le sommet est créé, puis passe au orange lorsqu'il est en cours de traitement lors du parcours en profondeur et devient rouge quand il a fini d'être traité par le parcours. Chaque sommet contient aussi un explorateur qui est le numéro du sommet qui l'a découvert lors du parcours en profondeur. Enfin chaque sommet possède une liste des numéros de ses successeurs dans le graphe.

Yannis Sauzeau
2021-2022
Algorithmique avancée Master 1, Informatique

Solution de l’exercice 8 de la feuille 7

Question a

Au cours de l’exécution d’un parcours en profondeur, la rencontre d’un sommet déjà vu mais non encore traité signifie qu’il y a un circuit dans le graphe ou le parcours s’effectue. Ce circuit part de ce sommet déjà vu mais non encore traité et va jusqu’au sommet qui l’a rencontré.

Question b

Si il existe un arc de x à y, dans le parcours en profondeur c’est le sommet x qui découvre le sommet y, l’estampille de début de x est donc inférieure à l’estampille de début de y. Pour l’estampille de fin, c’est l’inverse, en effet le sommet y sera traité avant le sommet x dans la pile d’appel récursif de l’algorithme qui effectue le parcours en profondeur.

Question c

1. Initialisation

```
for x in V loop
   Mark(x) <- Green
end loop
```

2. Lancement du parcours

```
for x in V loop
   if Mark(x) == Green then
      Pi(x) <- Nil
      DFS(x)
   end if
end loop
```

3. Parcours

```
DFS(x)
   Mark(x) <- Orange
   for y in G+(x) loop
      if Mark(y) == Green then
         Pi(y) <- x
         DFS(y)
      end if
    end loop
    Mark(x) <- Red
    P.push(x)
end function
```

4. Affichage

```
while not (P.empty()) do
  P.pop()
done
```

La complexité de cet algorithme est en O(|V| + |E|) avec |V| le nombre de sommets du graphe et |E| le nombre d'arcs du graphe.

Lorsqu'il n'existe pas de tri topologique pour le graphe donné, l'algorithme ci-dessus donne un faux tri topologique dans lequel se trouve un circuit, c'est pour cela que dans l'algorithme programmé durant ce projet lorsqu'il y a un circuit on lève une erreur et on affiche le circuit.