Université de Lille

ALGO

1er décembre 2015
Licence ST-A
François Lemaire

Contrôle TP

Avant de commencer à répondre aux questions, veuillez lire complétement le fichier `diametre.c`. Cela vous permettra de comprendre les questions qui vous sont demandées.

Consignes importantes :
— vous ne devez pas changer les entêtes des fonctions, ni le contenu de la fonction `main`
— vous devez commenter votre code
— pour chaque question, vous devez compléter le fichier CR.txt
— compilez avec les options `-Wall -std=c99` en ignorant les éventuels avertissements du type :
`warning: control reaches end of non-void function`

1 Calcul de diamètre

Dans tout le TP, nous ne considérons que des graphes qui sont **connexes** et **non orientés**. L'objectif du TP est de calculer, pour un graphe donné, son diamètre.

Définition 1. *Soit G un graphe connexe non orienté. Pour tout couple de sommets (x, y), on note $d(x, y)$ la distance séparant x et y, c'est-à-dire la longueur d'une plus courte chaîne reliant x à y. On appelle* diamètre *de G le maximum de $d(x, y)$ pour tous les couples de sommets (x, y) possibles.*

Attention à ne pas faire d'erreur sur la définition de $d(x, y)$. Par exemple, sur le graphe de la figure 1, $d(A, D)$ vaut 4, et pas 5.

Q 1 . Pour le graphe de la figure 1, trouver et donner (sans justifier) une plus courte chaîne entre deux sommets, qui soit de longueur 3.

Q 2 . Compléter le tableau du fichier `CR.txt` avec toutes les distances entre sommets, pour le graphe de la figure 1. En déduire que le diamètre vaut 3.

Pour calculer le diamètre d'un graphe, on pourrait calculer $d(x, y)$ pour tous les couples de sommets possibles. Pour faire cela efficacement, on va faire une succession d'appels à l'algorithme `plus_courte_chaine` en faisant varier le sommet de départ s parmi tous les sommets du graphe. On obtient l'algorithme suivant :

```
Fonction : calculeDiametre
Entrée   : un graphe connexe et non orienté
Sortie   : son diamètre

diametre := 0
Pour chaque sommet s du graphe
  calculer d(s,x) pour tous les sommets x du graphe (par l'algo plus_courte_chaine)
  max_d := le max des d(s,x)
  diametre := max(max_d, diametre)
fait
renvoyer diametre
```

Q 3 . Compléter la fonction `max`.

Q 4 . Compléter la fonction `plusCourteChaine`.

Q 5 . Compléter la fonction `calculeDiametre`.

Q 6 . Tester votre programme sur les fichiers `graphe-1.grp`, `graphe-2.grp`, `graphe-3.grp` et `graphe-4.grp`, et reportez les valeurs que vous obtenez avec votre programme.

Indication : vous devez obtenir respectivement 3, 3, 6 et 6.

On considère maintenant un nouvel algorithme de calcul de diamètre, qui ne fonctionne que si le graphe n'a pas de cycle (en plus d'être connexe et non orienté).

```
Fonction : calculeDiametreOptimise
Entrée   : un graphe connexe et non orienté, et sans cycle
Sortie   : son diamètre

choisir un sommet s au hasard
calculer d(s,x) pour tous les sommets x du graphe (par l'algo plus_courte_chaine)
choisir un sommet x_max tel que d(s,x_max) est maximum
calculer d(x_max,y) pour tous les sommets y du graphe (par l'algo plus_courte_chaine)
renvoyer le maximum des d(x_max,y)
```

Q 7 . Est-ce que l'algorithme `calculeDiametreOptimise` a une meilleure complexité que `calculeDiametre` ? Justifier clairement en donnant la complexité des deux algorithmes.

Q 8 . Pour cette question, vous avez le droit de rajouter une nouvelle fonction

`int calculeDiametreOptimise(tGraphe graphe)`,

et de changer l'appel à `calculeDiametre` en un appel à `calculeDiametreOptimise` dans le `main`. Écrire et tester `calculeDiametreOptimise` sur le graphe de la figure 4.

2 Annexe

Voici les représentations sagitalles des fichiers `graphe-1.grp`, `graphe-2.grp`, `graphe-3.grp` et `graphe-4.grp`.

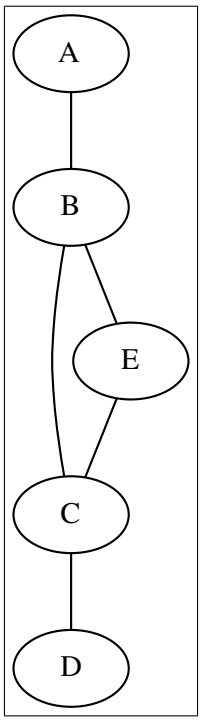

FIGURE 1 – graphe-1.grp

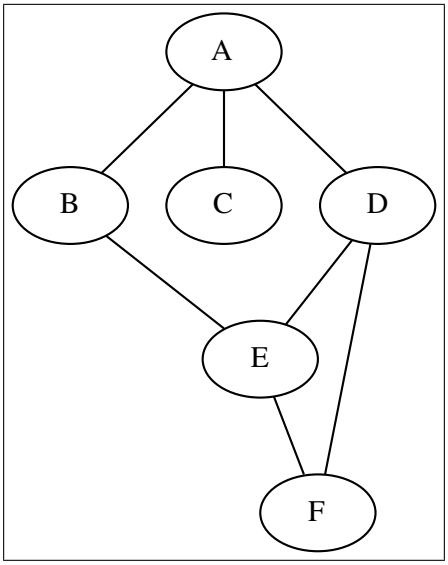

FIGURE 2 – graphe-2.grp

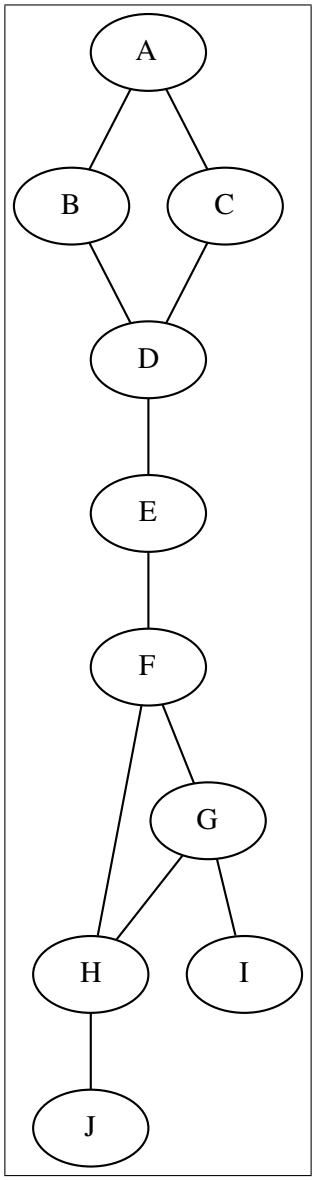

FIGURE 3 – graphe-3.grp

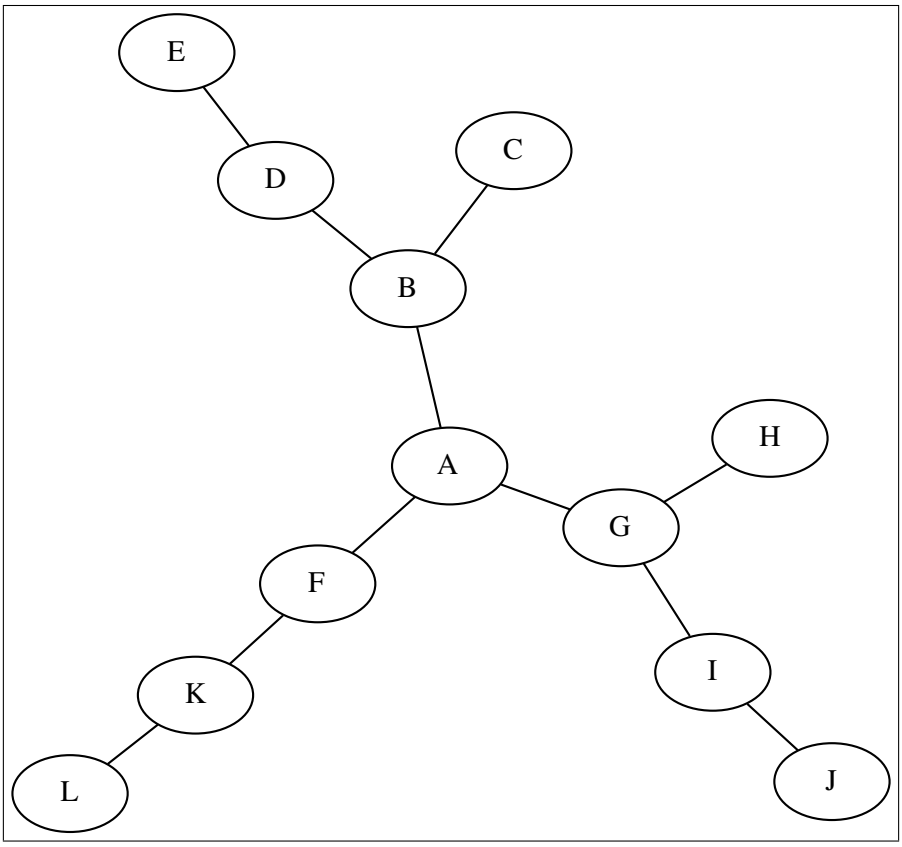

FIGURE 4 – graphe-4.grp