Algorithmique II
Année 2006-2007

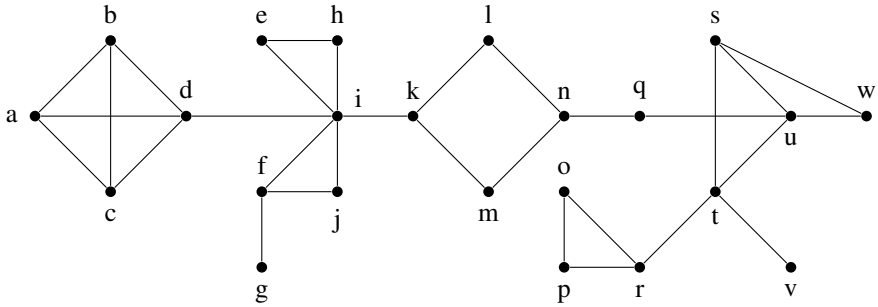
Florent Bouchez
Jean-François Pineau

Partiel d'Algorithmique II

1 Le cours, le cours, il furet (à rendre sur la copie 1)

Question 1.1. Quand on évalue la complexité d'algorithmes s'appliquant sur les graphes représentés par des listes d'adjacences, on considère souvent que ces listes sont triées. Pourquoi ? Et si elles ne le sont pas ?

2 Et les coudes, et les genoux, et les chevilles (à rendre sur la copie 2)



Soit $G = (V, E)$ un graphe non orienté connexe. Un *point d'articulation* de G est un sommet dont la suppression déconnecte le graphe. Un *isthme* est une arête dont la suppression déconnecte le graphe. Une *composante 2-connexe* de G est un ensemble maximal de sommets tel que le sous-graphe induit par ces sommets n'a pas de point d'articulation.

On notera $T = (V, F)$ un arbre de parcours de G obtenu par parcours en profondeur et $deb(v)$ le début de visite de v dans la numérotation associée à ce parcours en profondeur.

Question 2.1. Sur l'exemple ci-dessus, déterminer les points d'articulation, les isthmes et les composantes 2-connexes.

Question 2.2. Expliquer pourquoi dans le parcours en profondeur d'un graphe non orienté et connexe on n'obtient que deux sortes d'arcs : les arcs de l'arbre et les arcs retours.

Question 2.3. Montrer que la racine de T est un point d'articulation si et seulement si elle a au moins deux fils dans T.

Question 2.4. Soit x un sommet de T distinct de la racine. Prouver que x est un point d'articulation si et seulement si il existe un fils y de x dans l'arbre tel qu'il n'existe pas d'arc retour de la forme (z, t) où z est un descendant (au sens large) de y et t un ancêtre propre de x.

Question 2.5. On définit la fonction `au_dessous` de V dans $\mathbb{N}$ par :

$$\texttt{au\_dessous}(x) = \min\{deb(x); deb(z) : (y, z) \text{ est un arc retour avec } y \text{ descendant de } x\}$$

Donner un algorithme en $\mathcal{O}(n + m)$ de calcul de la fonction `au_dessous`. On prouvera la complexité.

Question 2.6. Montrer comment calculer tous les points d'articulation en $\mathcal{O}(n + m)$.

Question 2.7. Prouver qu'une arête est un isthme si et seulement si elle n'appartient à aucun cycle élémentaire.

Question 2.8. Montrer comment calculer les isthmes en $\mathcal{O}(n+m)$.

Question 2.9. Considérer le graphe dont les sommets sont les composantes 2-connexes de G et tel que deux composantes sont adjacentes si et seulement si elles ont au moins un sommet en commun. Qu'est-ce que vous pouvez dire de ce graphe ? Justifier votre réponse.

3 Les femmes et les plus petits d'abord **(à rendre sur la copie 1)**

On souhaite maintenir à jour un ensemble dynamique T d'éléments de $\{1, 2, \cdots, n\}$; cet ensemble, initialement vide, peut être augmenté ou diminué par les opérations `insérer` et `extraire_min`. La première ajoute un élément choisi dans T tandis que la seconde retire le plus petit élément de T.

On considère une séquence S contenant n appels à `insérer` et m appels à `extraire_min`, où chaque élément de $\{1, 2, \cdots, n\}$ est inséré exactement une fois. On souhaite déterminer les éléments renvoyés par chaque appel à `extraire_min`, c'est-à-dire remplir un tableau `extrait[1..m]` où pour $i \in \{1, \cdots, n\}$, `extrait[i]` est l'élément renvoyé par le $i$ème appel à `extraire_min`.

Question 3.1. On considère la séquence S suivante, dans laquelle chaque opération `insérer` est représentée par le nombre qu'elle insère dans T, et chaque opération `extraire_min` est représentée par la lettre `E` :

$$S = 4,8,E,3,E,9,2,6,E,E,E,1,7,E,5$$

Quel est le tableau `extrait` correspondant à cette séquence ?

Question 3.2. Une manière de résoudre ce problème est de maintenir l'ensemble T et d'y appliquer S itérativement. On remplit alors les cases de `extrait` de 1 à m dans l'ordre. Quelle structure de données envisageriez-vous pour maintenir à jour T ? Quelle serait alors la complexité de calcul de `extrait` en fonction de n et m ?

En fait, il est possible de traiter entièrement la séquence S avant de déterminer n'importe lequel des éléments extraits. Pour cela, on découpe la séquence S en sous-séquences homogènes, autrement dit, on représente S par :

$$\mathrm{I}_1,\mathrm{E},\mathrm{I}_2,\mathrm{E},\mathrm{I}_3,\ldots,\mathrm{I}_m,\mathrm{E},\mathrm{I}_{m+1}$$

où chaque E représente un unique appel à `extraire_min` et chaque I_j représente une séquence (éventuellement vide) d'appels à `insérer`. Pour chaque sous-séquence I_j, on commence par placer tous les éléments insérés par ces opérations dans un ensemble T_j, qui est vide si I_j est vide. On exécute alors la procédure `Extraire` suivante :

pour $i \leftarrow$ *1 à n* **faire**
  soit j tel que $i \in T_j$
  si $j \neq m+1$ **alors**
    `extrait`$[j] \leftarrow$ i
    soit $k > j$ minimum tel que T_k existe
    $T_k \leftarrow T_j \cup T_k$ et détruire T_j
  fin
fin
retourner `extrait`

Question 3.3. Expliquer pourquoi le tableau `extrait` renvoyé par `Extraire` est correct.

Question 3.4. Quelle structure de données utiliser pour implanter efficacement `Extraire` ? Donner une borne approchée pour le temps d'exécution dans le pire cas.