Devoir 3

BLAIS REGOUT, Lucien
SAVOIE, Olivier

IFT436 - Algoritmes et structures de données

Faculté des Sciences
Université de Sherbrooke

Présenté à Pr BLONDIN, Michael

Jeudi, 10 octobre 2019

Question 1

a)

V = Les bassins du manège aquatique (sommets).

E = Les corridors entre les bassins, potentiellement des glissades(arêtes).

G = Manège aquatique est dirigé.

b)

G ne peut pas être un cycle dû aux contraintes stipulées. Cette contrainte, ci-dessous, en témoigne.

> Il n'y a *pas* de corridor passant d'un bassin vers lui-même.

Il est impossible qu'un cycle soit établi avec un seul bassin considérant qu'il y a aucun corridor qui part d'un dit bassin en arrivant dans ce même bassin.

Finalement, cette seconde contrainte, implique que, dû à l'inclinaison, l'un ne peut pas revenir au bassin précédent. Cela implique qu'il y a aucun cycle possible et que de ce fait, le manège aquatique ***G*** est acyclique et dirigé.

> Puisque les bassins sont situés de plus en plus bas, le long d'une colline, si on peut atteindre un bassin j à partir d'un bassin i, alors on ne peut pas atteindre le bassin i à partir du bassin j.

c)

Si le graphe ***G*** est considéré non dirigé, il y a plusieurs arrangements d'arêtes qui rendrait ***G*** cyclique et pour lesquels $|\boldsymbol{E}| > |\boldsymbol{V}| - 1$. Ainsi, dans la majorité des cas, ***G*** n'est pas un arbre, car il ne respecte pas les propriétés essentielles pour en être un. Toutefois, il y a toujours la possibilité qu'aucun cycle, simple ou non, ne s'y retrouve et que le nombre d'arête soit équivalent au nombre de sommet -1. De sorte que les propriétés suivantes soit respectées.

— ***G*** est $connexe$ et $\mathbf{E}| = |\mathbf{V}| - 1$

d)

$$min = n - 1, \quad \forall n \geq 1$$

Dans le cas minimal, les arêtes placées représenterons un graphe ***G*** qui sera forcément un arbre, respectant les contraintes, ainsi que l'explication élaborée à la lettre précédente, **c)**.

$$max = \sum_{i=1}^{n} n - i = \sum_{i=1}^{n} n - \sum_{i=1}^{n} i = n^2 - \frac{n}{2}(n+1) = \frac{n^2 - n}{2}$$

Advennant que chacun des bassins présents se voient attribués un corridor vers chacun des bassins soudjacents, l'équation maximal reflètera la composition du chemin le plus long.

e)

Bassin départ : De toutes les paires de sommets composants les arêtes, le sommet qui se retrouvera uniquement en index 0 de la paire, à la gauche, qui est donc toujours le sommet de départ, sera donc considéré comme sommet de départ.

Bassin d'arrivée : Suivre le même résonnement, mais pour le sommet qui se retrouve toujours en index 1 de la paire et qui est donc toujours le bassin d'arrivée dans toutes les paires, sera donc considéré comme sommet d'arrivée.

f)

Algorithme 1 : Calcul de la séquence qui possède un temps maximal passé dans une attraction ***G***

Entrées : Manège ***G*** tel que ***G*** $= ($***V***,***E***$)$ et sommet initial u tel que u appartien à ***V***.

Résultat : Une séquence S de bassins de temps maximal dans le manège ***G***.

```
S ← []
Ttot ← 0
Parent ← u
parcours (x) :
    SPrime ← S
    Temps ← Ttot
    si x non marquée alors
        Temps ← Temps + c[Parent, x]
        marquer x
        ajouter x à SPrime
        si Temps > Ttot alors
            Ttot ← Temps
            S ← SPrime
        pour u ∈V : x → y faire
            Parent ← x
            parcours(y)
parcours(u) retourner S
```

g)

Algorithme 2 : Calcul de la séquence qui possède un temps maximal passé dans une attraction ***G***, donné 5 remontées

Entrées : Manège ***G*** tel que ***G*** $= (\boldsymbol{V},\boldsymbol{E})$, que le sommet initial u tel que $u \in \boldsymbol{V}$, que $v \in \boldsymbol{V}$, ainsi que v est le bassin finale.

Résultat : Une séquence S de bassins, de temps maximal dans le manège ***G***, considérant que l'utilisateur peut remonter au bassin initial un maximum de 5 fois.

$S \leftarrow [\,]$
$Ttot \leftarrow 0$
$Parent \leftarrow u$
$Ascension \leftarrow 5$
`parcours` (x) **:**
    $SPrime \leftarrow S$
    $Temps \leftarrow Ttot$
    si x *non marquée* **alors**
        $Temps \leftarrow Temps + c[Parent, x]$
        marquer x
        ajouter x à $SPrime$
        si $Temps > Ttot$ **alors**
            $Ttot \leftarrow Temps$
            $S \leftarrow SPrime$
        si *prochain voisin de* $x = v$ **alors**
            $Ascension \leftarrow Acension - 1$
            retirer les marqueurs
            $Parent \leftarrow S[0]$
            `parcours` $(Parent)$
        sinon
            pour $u \in \boldsymbol{V} : x \rightarrow y$ **faire**
                $Parent \leftarrow x$
                `parcours` (y)
`parcours` (u) **retourner** S

Ici, il fera exactement comme l'algorithme précédent, à l'exception que lorsqu'il se rendra au plus long parcours précédent le bassin final, et se 5 fois. La 5e fois, le sera possiblement différent mais restera cependant couvers par l'algorithme.

Question 2

a)

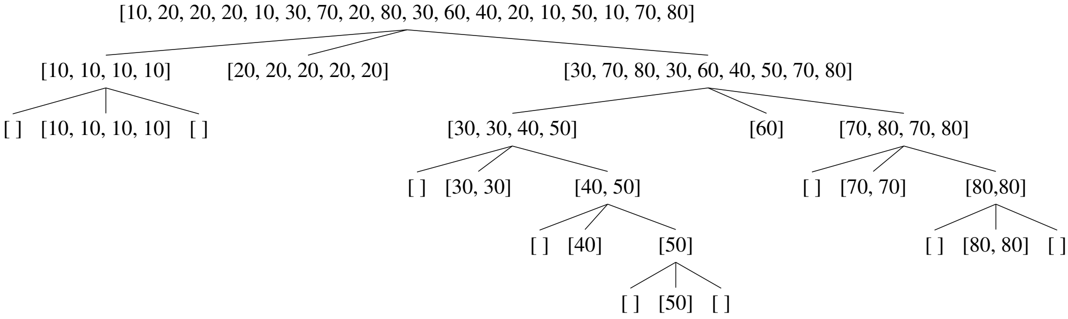
Algorithme 3 : Tri d'une séquence et calcul des valeurs modales

Entrées : Une séquence S non vide d'éléments comparables.

Résultat : Les valeurs modales$m$ trouvées dans la séquence S

$mode \leftarrow [\,]$
$modeQty \leftarrow 0$
`trouverMode` (x) **:**
    si $|x| = 0$ **alors**
        retourner $mode$
    sinon
        $pivot \leftarrow$ `mediane` (x)
        $gauche \leftarrow [x \in S : x < pivot]$
        $centre \leftarrow [x \in S : x = pivot]$
        $droite \leftarrow [x \in S : x > pivot]$
        si $|centre| = modeQty$ **alors**
            ajouter $pivot$ à $mode$
        si $|centre| > modeQty$ **alors**
            $modeQty \leftarrow |centre|$
            $mode \leftarrow [pivot]$
        sinon
            `trouverMode(`$gauche$`)`
            `trouverMode(`$droite$`)`
retourner `trouverMode(`S`)`

b)



Question 3

a)

b)

c)