Numéro d'anonymat :

Examen LU2IN003

Vendredi 20 Mai 2022, 2 heures
Aucun document autorisé

1 Exercice sur les arbres (10 points)

Dans tout cet exercice, les arbres binaires sont étiquetés dans $\mathbb{N}$.

Questions de cours

1. 1. Rappeler la définition inductive de l'ensemble ABR des arbres binaires de recherche.
   2. Dessiner l'arbre binaire de recherche T_0 obtenu à partir de l'arbre vide par insertions successives des clefs $8, 3, 1, 5, 4, 7, 6, 12, 9, 13$ (on rappelle que l'insertion se fait aux feuilles).
      Donner la hauteur de T_0, la taille de T_0 (c'est-à-dire son nombre de nœuds) et le parcours infixe de T_0.

2. 1. Donner une condition nécessaire et suffisante pour qu'une liste d'entiers naturels soit le parcours infixe d'un ABR (on ne demande pas de justification).
   2. Dessiner l'arbre binaire de recherche T_1 dont le parcours préfixe est $[6, 1, 5, 3, 8, 7, 9]$.

Arbres binaires de recherche dénombrés

Un arbre binaire de recherche *dénombré* est un arbre binaire de recherche dont chaque nœud x contient une information supplémentaire : la taille du fils gauche de x.
Voici une définition inductive de l'ensemble $ABRT$ des arbres binaires de recherche dénombrés. $T \in ABRT$ si :

Base $T = \emptyset$;

Induction $T = (x, t, G, D)$ où :

- — $x \in \mathbb{N}$, $G \in ABRT$, $D \in ABRT$, t est égal à la taille de G ;
- — toutes les clefs de G sont inférieures strictement à x ;
- — toutes les clefs de D sont supérieures strictement à x.

Pour manipuler les arbres binaires de recherche dénombrés dans les questions suivantes, nous utilisons les primitives :

- — `estABTvide(T)` teste si l'arbre T est vide ;
- — si T est non vide :
  - — `T.clef` désigne l'étiquette (ou clef) de T,
  - — `T.ng` désigne la taille du sous-arbre gauche de T,
  - — `T.gauche` désigne le sous-arbre gauche de T,
  - — `T.droit` désigne le sous-arbre droit de T.

On considère l'arbre binaire de recherche dénombré `Tex` représenté par la figure 1.

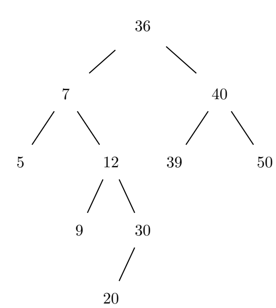

FIGURE 1 – L'arbre binaire dénombré `Tex`. Les nœuds sont étiquetés uniquement par leur clef sur cette figure.

3. Donner les valeurs de `Tex.ng`, `Tex.gauche.ng`, `Tex.droit.gauche.ng`

On définit une fonction `taille` sur les arbres binaires dénombrés :

```
def taille(T):
  if estABRTvide(T):
    res = 0
  else:
    print("appel sur T de racine ", T.clef)
    res = 1 + T.ng + taille(T.droit)
    print("taille de T de racine ", T.clef, " : ", res)
  return res
```

4. Exécuter l'appel de `taille(Tex)`, en donnant les affichages successifs et le résultat final.

5. Montrer, par induction structurelle sur ABRT, que pour tout arbre binaire de recherche dénombré T, `taille(T)` se termine et retourne le nombre de nœuds de T.

6. Pour un arbre binaire de recherche dénombré T de taille n et de hauteur h, calculer la complexité comptée en nombre d'additions effectuées par `taille(T)` dans le meilleur cas et dans le pire cas. Justifier vos réponses.

7. On considère dans cette question que T est un arbre binaire de recherche dénombré H-équilibré de taille n.
   1. En étudiant le meilleur des cas, démontrer que la complexité de `taille(T)` est en $\Omega(\log(n))$.
      Indication : en appelant $c(h)$ la complexité meilleur cas de `taille(T)` pour T de hauteur h, donner une relation de récurrence permettant de calculer $c(h)$. Résoudre alors la suite.
   2. En déduire que, pour T arbre binaire de recherche dénombré H-équilibré de taille n, la complexité de `taille(T)` est en $\Theta(\log(n))$. Justifier votre réponse.

Recherche du k-ème plus petit élément

Le *k-ème plus petit élément* d'un ensemble E de n entiers tous différents, avec $n \geq k \geq 1$, est l'élément x de E tel qu'il y a exactement $k-1$ éléments de E strictement inférieurs à x.

Par exemple, le 3-ème plus petit élément de $\{8, 5, 3, 4, 1, 6, 7\}$ est 4.

8. Dans cette question l'ensemble E est représenté par une liste non triée L. Voici une méthode permettant de déterminer le k-ème plus petit élément de L :
(1) trier la liste L en ordre croissant ;
(2) accéder au k-ème élément de la liste triée.

   1. Quelle est la complexité minimale de cette méthode pour une liste représentée par un tableau ? Justifiez votre réponse et précisez un algorithme de tri dont la complexité est égale à cette complexité minimale.
   2. Même question que précédemment si la liste est représentée par une liste doublement chaînée circulaire.

Dans la suite de l'exercice, l'ensemble E est représenté par un arbre binaire de recherche dénombré.

On définit une fonction `ABRT_k_eme(T, k)` sur les arbres binaires de recherche dénombrés :

```python
def ABRT_k_eme(T, k):
    """hypothese : k <= taille(T)"""
    print("appel avec k = ", k, "sur T de racine ", T.clef)

    if k == (T.ng + 1) :
        res = T.clef
    elif k <= T.ng:
        res = ABRT_k_eme(T.gauche, k)
    else:
        res = ABRT_k_eme(T.droit, k - (T.ng + 1))
    print("retour pour T de racine ", T.clef ,": ", res)
    return res
```

9. Exécuter l'appel de `ABRD_k_eme(Tex,6)`, en donnant les affichages successifs et le résultat final.

10. Montrer par induction structurelle sur l'ensemble ABRT que, pour tout arbre binaire de recherche dénombré T et pour tout k tel que $0 < k \leq taille(T)$, `ABRD_k_eme(T, k)` se termine et calcule la k-ème plus petite clef de T.

    Vous veillerez à énoncer clairement la propriété $\mathcal{P}(T)$, pour $T \in ABRT$ à démontrer.

11. 1. Quelle est la complexité, comptée en nombre de comparaisons entre clefs, de la recherche de la k-ème plus petite clef dans un arbre de recherche dénombré de taille n dans le meilleur cas ? Dans le pire cas ?

    2. Mêmes questions pour un arbre de recherche dénombré H-équilibré.

2 Exercice sur les graphes (7.5 points)

Dans cet exercice, $G = (V, E)$ désigne un graphe non orienté. On rappelle que les graphes considérés sont sans boucle ni arête multiple.

1. On considère **uniquement** dans cette question le graphe non orienté $G_1 = (V_1, E_1)$ avec $V_1 = \{1, 2, 3, 4, 5\}$ et les arêtes $E_1 = \{\{1, 2\}, \{1, 3\}, \{1, 4\}, \{2, 5\}, \{3, 5\}, \{4, 5\}\}$.
   1. Donner la représentation de G_1 sous la forme d'une matrice sommet-sommet ;
   2. Donner, pour tous les sommets $u \in V_1$, le degré $d(u)$ de u pour le graphe G_1 ;
   3. Dans le cas d'un graphe $G = (V, E)$ non orienté quelconque, quelle est la complexité du calcul de $d(u)$ pour tout sommet u dans le pire et le meilleur des cas pour une représentation par une matrice sommet-sommet ? Justifier votre réponse.

2. Soit $G = (V, E)$ un graphe non orienté. Démontrer que $\sum_{v \in V} d(v) = 2m$.

3. Soit $G = (V, E)$ un graphe non orienté minimal connexe avec $|V| \geq 1$. Démontrer par récurrence sur le nombre de sommets $n = |V|$ du graphe G que $|E| = |V| - 1$.

4. Dessiner deux arbres différents qui ont chacun 2 sommets de degré 3, 3 sommets de degré 2, et où tous les autres sommets sont des feuilles (c'est à dire des sommets de degré 1). Pour chacun des arbres, indiquer leur nombre de feuilles.

5. Soit $T = (V, E)$ un **arbre**. Soient a son nombre de sommets de degré 3, b son nombre de sommets de degré 2 et c son nombre de feuilles. On suppose que T n'a aucun sommet de degré 4 ou plus.
   1. Exprimer en fonction de a, b et c le nombre de sommets de T ;
   2. Exprimer en fonction de a, b et c la somme des degrés des sommets de T ;
   3. Exprimer c en fonction de a et de b.

6. Soit $G = (V, E)$ un graphe non orienté qui a 2 sommets de degré 3, 3 sommets de degré 2 et aucun sommet de degré 4 ou plus.
   1. Si G est un arbre, calculer son nombre de feuilles c ;
   2. Donner un exemple où G n'est pas un arbre et a strictement moins de c sommets de degré 1 ;
   3. Donner un exemple où G n'est pas un arbre et a strictement plus de c sommets de degré 1.

3 QCM de cours (4 points)

Un seul choix est possible. Une bonne réponse = 0.5 points. Une mauvaise réponse = -0.25 points.

1. (½ point) Dans les deux questions suivantes, on considère la fonction de tri définie comme suit :

```
def triMystere(tab):
    n = len(tab)
    for i in range(n):
        for j in range(0, n-i-1):
            if tab[j] > tab[j+1] :
                tab[j], tab[j+1] = tab[j+1], tab[j]
```

   La dernière ligne échange `tab[j]` et `tab[j+1]`. Quel est l'invariant de boucle $\mathcal{I}(j)$ de la boucle interne de `triMystere` placé juste après cet échange ?

   - ◯ `tab[0..j+1]` est constitué des $j+2$ plus petits éléments du tableau initial et est trié en ordre croissant ;
   - ◯ `tab[j..n-1]` contient les $n-j$ plus grands éléments du tableau initial et est trié en ordre croissant ;
   - ◯ `tab[j+1..n-1]` contient les $n-j-1$ les plus grands éléments du tableau initial et est trié en ordre croissant ;
   - ◯ Aucun des choix précédents.

2. (½ point) Une seule des affirmations est vérifiée, laquelle ?
   - ◯ La fonction `triMystere(tab)` est un tri par sélection et sa complexité est $\Theta(n^2)$;
   - ◯ La fonction `triMystere(tab)` est un tri par sélection et sa complexité est $\mathcal{O}(n \log n)$;
   - ◯ La fonction `triMystere(tab)` est un tri à Bulles et sa complexité est $\Theta(n^2)$;
   - ◯ La fonction `triMystere(tab)` est un tri à Bulles et sa complexité est $\mathcal{O}(n \log n)$;
   - ◯ Aucun des choix précédents.

3. (½ point) Quel est le tas obtenu par insertions successives des clefs 8, 3, 5, 4, 1, 2.
   - ◯ $[\mathbf{6}, 1, 2, 3, 8, 4, 5]$;
   - ◯ $[\mathbf{6}, 1, 3, 2, 8, 4, 5]$;
   - ◯ $[\mathbf{6}, 1, 3, 2, 5, 4, 8]$;
   - ◯ Aucun des choix précédents.

4. (½ point) On considère le tas $[\mathbf{9}, 1, 2, 5, 3, 8, 6, 8, 7, 10]$. Quel est le tas obtenu après suppression du minimum ?
   - ◯ $[\mathbf{8}, 2, 5, 3, 8, 7, 6, 8, 10]$;
   - ◯ $[\mathbf{8}, 2, 3, 5, 7, 8, 6, 8, 10]$;
   - ◯ $[\mathbf{8}, 2, 3, 5, 7, 6, 8, 8, 10]$;
   - ◯ Aucun des choix précédents.
5. (½ point) On considère le graphe non orienté $G_1 = (V_1, E_1)$ représenté par la figure 2.

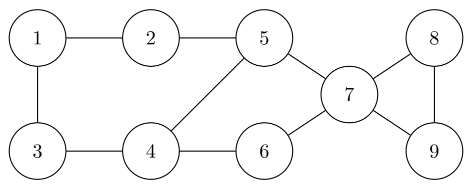

FIGURE 2 – Un graphe non orienté $G_1 = (V_1, E_1)$.

Une seule affirmation est correcte, laquelle ?
   - ◯ $L = (5, 7, 4, 6, 2, 3, 8, 9, 1)$ est un parcours générique de G_1 ;
   - ◯ $L = (7, 5, 4, 1, 2, 3, 6, 8, 9)$ est un parcours générique de G_1 ;
   - ◯ $L = (9, 7, 5, 4, 3, 1, 2, 6, 8)$ n'est pas un parcours générique de G_1.
   - ◯ Aucun des choix précédents.
6. (½ point) On considère le graphe non orienté G_1 représenté par la figure 2. Une seule des affirmations est vérifiée, laquelle ?
   - ◯ Le parcours en largeur $L = (5, 2, 4, 6, 7, 1, 3, 8, 9)$ de G_1 a pour graphe de liaison en largeur $\mathcal{A} = (\{1, 2, 3, 4, 5, 6, 7, 8, 9\}, \{(5, 2), (5, 4), (5, 7), (2, 3), (2, 1), (4, 6), (7, 8), (7, 9)\})$;
   - ◯ Le parcours en largeur $L = (5, 2, 4, 7, 1, 3, 6, 8, 9)$ de G_1 a pour graphe de liaison en largeur $\mathcal{A} = (\{1, 2, 3, 4, 5, 6, 7, 8, 9\}, \{(5, 2), (5, 4), (5, 7), (2, 1), (4, 3), (4, 6), (7, 8), (7, 9)\})$;
   - ◯ Le parcours en largeur $L = (7, 8, 9, 5, 6, 2, 4, 3, 1)$ de G_1a pour graphe de liaison en largeur $\mathcal{A} = (\{1, 2, 3, 4, 5, 6, 7, 8, 9\}, \{(7, 8), (7, 5), (7, 6), (8, 9), (5, 2), (5, 4), (2, 3), (3, 1)\})$;
   - ◯ Aucun des choix précédents.

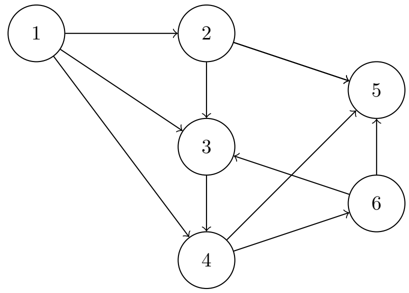

FIGURE 3 – Le graphe orienté $G_2 = (V_2, A_2)$

7. (½ point) On considère le graphe orienté $G_2 = (V_2, A_2)$ représenté par la figure 3. Que valent les intervalles $[pre[u], post[u]]$, $u \in V_2$ pour le parcours en profondeur $L = (1, 4, 5, 6, 3, 2)$?

○

$u \in V_2$	1	2	3	4	5	6
$[pre[u], post[u]]$	[1,12]	[8,11]	[9,10]	[2,5]	[3,4]	[6,7]

○

$u \in V_2$	1	2	3	4	5	6
$[pre[u], post[u]]$	[1,12]	[10,11]	[6,7]	[2,9]	[3,4]	[5,8]

○

$u \in V_2$	1	2	3	4	5	6
$[pre[u], post[u]]$	[1,10]	[11,12]	[7,8]	[2,5]	[3,4]	[6,9]

8. (½ point) On considère le parcours en profondeur $L = (1, 4, 5, 6, 3, 2)$ du graphe orienté $G_2 = (V_2, A_2)$ représenté par la figure 3. Une seule affirmation est correcte, laquelle ?

○ $(6, 5)$ et $(2, 5)$ sont des arcs transverses, et $(3, 4)$ est un arc avant ;

○ $(3, 4)$ est un arc arrière, $(2, 5)$ est un arc transverse et $(2, 3)$ est un arc de liaison ;

○ $(3, 4)$ est un arc arrière, $(1, 3)$ est un arc avant et $(2, 5)$ est un arc transverse ;

○ Aucun des choix précédents.