Numéro d'anonymat :

Examen LU2IN003

Mercredi 12 Mai 2021, 1.5 heures
aucun document autorisé

Exercice 1 – Arbres binaires balisés (10 points)

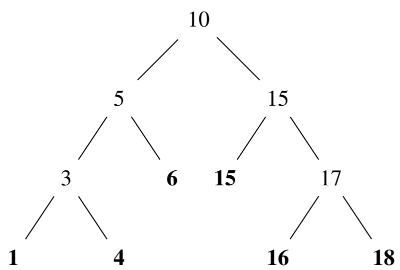
On rappelle qu'un arbre binaire **strict** est un arbre binaire **non vide** dans lequel tout nœud a 0 ou 2 fils.
Un *arbre binaire balisé* sur $\mathbb{N}$ (noté ABB) est un arbre binaire **strict** étiqueté sur $\mathbb{N}$ dans lequel tout nœud a une clé qui est supérieure ou égale à toutes les clés de son sous-arbre gauche et qui est strictement inférieure à toutes les clés de son sous-arbre droit.
Les clés des nœuds internes sont appelées *balises* et les clés des feuilles sont appelées *valeurs*. Dans un ABB, on ne s'intéresse qu'aux valeurs (stockées aux feuilles), les balises ne servent qu'à diriger la recherche.
Dans les preuves, on utilisera la définition inductive des ABB. B est un ABB si :

Base $B = (x, \emptyset, \emptyset)$ avec $x \in \mathbb{N}$;

Induction $B = (b, G, D)$ avec $b \in \mathbb{N}$ et :

- G et D sont des ABB ;
- toutes les clefs de G sont inférieures ou égales à b ;
- toutes les clefs de D sont strictement supérieures à b.

Voici un exemple d'ABB, que l'on appellera $B1$:



Les valeurs de $B1$ sont : 1, 4, 6, 15, 16, 18. Les balises de $B1$ sont : 10, 5, 15, 3, 17.
Pour manipuler les ABB, on utilise les primitives définies sur les arbres binaires : `AB.clef`, `AB.gauche`, `AB.droit`, auxquelles on ajoute les fonctions :
– `ABfeuille(x)` qui retourne un arbre binaire réduit à une feuille d'étiquette x,
– `estABfeuille(B)` qui teste si un arbre binaire B est réduit à une feuille.

Question 1

On note $ni(B)$ le nombre de nœuds internes et $f(B)$ le nombre de feuilles d'un arbre balisé B.

1. Que valent $ni(B1)$ et $f(B1)$?
2. Donner une définition inductive de $ni(B)$ et de $f(B)$.
3. Prouver par induction que, pour tout arbre balisé B : $f(B) = ni(B) + 1$.

Solution:

1. $ni(B_1) = 5$ et $f(B1) = 6$.
2. **Base** Si $B = (x, \emptyset, \emptyset)$ alors $ni(B) = 0$ et $f(B) = 1$.
   Induction Si $B = (b, G, D)$ alors $ni(B) = ni(G) + ni(D) + 1$ et $f(B) = f(G) + f(D)$.

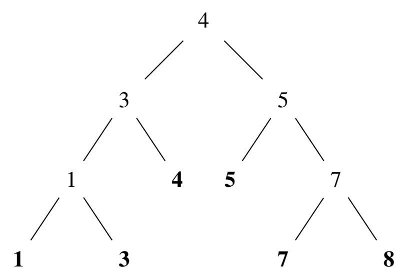
3. **Base** Si $B = (x, \emptyset, \emptyset)$ alors $ni(B) = 0$ et $f(B) = 1$ donc $f(B) = ni(B) + 1$.
   Induction Soit $B = (b, G, D)$ un ABB tel que $f(G) = ni(G) + 1$ et $f(D) = ni(D) + 1$, alors :
   $f(B) = f(G) + f(D) = ni(G) + 1 + ni(D) + 1 = ni(B) + 1$.
   Conclusion On a prouvé par induction que $f(B) = ni(B) + 1$ pour tout ABB B.

Question 2

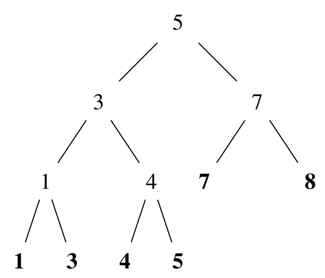
1. Soit $B2$ un arbre balisé dont les valeurs sont 4, 7, 3, 5, 1, 8. Combien $B2$ a-t-il de nœuds internes ?
2. Dessiner un arbre balisé contenant les valeurs 4, 7, 3, 5, 1, 8 (le choix des balises est laissé libre mais il doit bien sûr être judicieux).
3. Pouvez-vous dessiner un arbre balisé contenant les mêmes valeurs (c'est-à-dire 4, 7, 3, 5, 1, 8) et n'ayant pas la même forme ? Si oui, le dessiner.
4. Un arbre balisé est-il nécessairement parfait ? Justifier la réponse.

<u>**Solution**</u>:

1. $B2$ a 5 nœuds internes (un nœud interne de moins que de feuilles).
2. Un arbre balisé possible, parmi beaucoup d'autres...



3. On peut dessiner d'autres arbres balisés contenant les valeurs 4, 7, 3, 5, 1, 8. Par exemple :



   Il y a d'autres ABB possibles, par exemple des peignes.
4. Un arbre balisé n'est pas nécessairement parfait. Par exemple, l'arbre $B1$ est un ABB mais n'est pas parfait (son avant-dernier niveau n'est pas entièrement rempli).

La fonction `ABBinfixe(B)` définie ci-dessous calcule le parcours infixe des valeurs d'un ABB.

```
def ABBinfixe(B):
  if estABfeuille(B):
    return [B.clef]
  return ABBinfixe(B.gauche) + ABBinfixe(B.droit)
```

Question 3

1. Prouver par induction structurelle que, pour tout arbre balisé B, `ABBinfixe(B)` retourne la liste des valeurs de B rangée en ordre strictement croissant.
2. Montrer, par induction structurelle, que le nombre $c(f)$ de concaténations effectuées par `ABBinfixe(B)` est égal à $f-1$, où f est le nombre de valeurs de B.
3. En déduire la complexité de `ABBinfixe(B)` en fonction du nombre de nœuds n de l'arbre balisé B lorsque les listes sont représentées par des listes circulaires doublement chaînées. Justifier votre réponse.

Solution:

1. Par induction structurelle. Notons $P(B)$ la propriété : `ABBinfixe(B)` retourne la liste des valeurs de B rangée en ordre strictement croissant.

   Base Si $B = (x, \emptyset, \emptyset)$ avec $x \in \mathbb{N}$ alors `ABBinfixe(B)` est la liste `[x]`, qui est bien la liste des valeurs de B rangée en ordre strictement croissant.

   Induction Soit $B = (b, G, D)$ où $b \in \mathbb{N}$, G et D sont des ABB tels que toutes les clefs de G sont inférieures ou égales à b et toutes les clefs de D sont strictement supérieures à b. Supposons que $P(G)$ et $P(D)$ soient vraies. Posons $L =$ `ABBinfixe(B)`, $L_G =$ `ABBinfixe(G)` et $L_D =$ `ABBinfixe(D)` alors $L = L_G + L_D$. Par hypothèse de récurrence, L_G, resp. L_D, est la liste des valeurs (donc des étiquettes des feuilles) de G, resp. de D, donc $L = L_G + L_D$ est la liste des étiquettes des feuilles (donc des valeurs) de B.
   Toujours par hypothèse de récurrence, L_G et L_D sont rangées en ordre strictement croissant. De plus tous les éléments de L_G sont inférieurs ou égaux à b, qui est **lui-même strictement inférieur** à tous les éléments de L_D. Par conséquent, tous les éléments de L_G sont strictement inférieurs à tous les éléments de L_D et la liste $L = L_G + L_D$ est rangée en ordre strictement croissant.

   Conclusion On a prouvé, par induction structurelle, que $P(B)$ est vraie pour tout ABB B.
2. On montre, par induction structurelle, que le nombre de concaténations $c(f)$ effectué par `ABBinfixe(B)` est égal à $f-1$:
   – si $B = (x, \emptyset, \emptyset)$ alors $f = 1$ et il y a 0 concaténation ;
   – si $B = (b, G, D)$ alors $c(f) = c(f_G) + c(f_D) + 1 = f_G - 1 + f_D - 1 + 1 = f_G + f_D - 1 = f - 1$ (les arbres étant stricts, le nombre f de feuilles de B est égal à la somme du nombre de feuilles de G (f_G) et du nombre de feuilles de D (f_D)).
3. Les listes étant représentées par des listes circulaires doublement chaînées, chaque concaténation est en $\Theta(1)$. La complexité de `ABBinfixe(B)` est donc en $\Theta(f)$. D'après la question 1, le nombre de nœuds $n(B)$ de B vérfie $n(B) = f(B) + ni(B) = 2f(B) - 1$. On en déduit que la complexité est ainsi en $\Theta(n)$.

La fonction `ABBcherche(B,x)` définie ci-dessous teste si un entier est une valeur d'un ABB.

```
def ABBcherche(B,x):
  b = B.clef
  print("Appel␣a␣partir␣de␣b␣=␣", b)
  if estABfeuille(B):
    res = (x == b)
  else:
    if x <= b:
      res = ABBcherche(B.gauche,x)
    else:
      res = ABBcherche(B.droit,x)
  print("Valeur␣de␣retour␣en␣b␣=␣", b, "␣:␣", res)
  return res
```

Question 4

1. Exécuter l'appel de `ABBcherche(B1,5)`, en ne donnant que les affichages. Préciser la valeur retournée.
2. Calculer `ABBcherche(B1,1)`, `ABBcherche(B1,10)`, `ABBcherche(B1,15)`.
3. Prouver que, pour tout arbre balisé B et pour tout $x \in \mathbb{N}$, `ABBcherche(B,x)` se termine et retourne la valeur `True` si x est une valeur de B et la valeur `False` sinon.
4. Pour un arbre balisé B ayant f valeurs, calculer la complexité pire cas et la complexité meilleur cas de `ABBcherche(B,x)` en fonction de f.

Solution:

1. ```
   Appel a partir de b =  10
   Appel a partir de b =  5
   Appel a partir de b =  3
   Appel a partir de b =  4
   Valeur de retour en b =  4  :  False
   Valeur de retour en b =  3  :  False
   Valeur de retour en b =  5  :  False
   Valeur de retour en b =  10  :  False
   ```

   La valeur retournée est `False`.
2. `ABBcherche(B1,1)` retourne `True`, `ABBcherche(B1,10)` retourne `False`, `ABBcherche(B1,15)` retourne `True`.
3. Par induction structurelle. Notons $P(B)$ la propriété : `ABBcherche(B,x)` se termine et retourne la valeur `True` si x est une valeur de B et la valeur `False` sinon.

   Base Si $B = (b, \emptyset, \emptyset)$ avec $b \in \mathbb{N}$ alors B est une feuille et sa seule valeur est b. Dans ce cas `ABBcherche(B,x)` se termine et retourne la valeur `True` si $x = b$, donc si x est une valeur de B et la valeur `False` sinon.

   Induction Soit $B = (b, G, D)$ où $b \in \mathbb{N}$, G et D sont des ABB tels que toutes les clefs de G sont inférieures ou égales à b et toutes les clefs de D sont strictement supérieures à b. Supposons que $P(G)$ et $P(D)$ soient vraies. Il y a deux cas possibles : ou bien $x \leq b$ ou bien $x > b$.
   Dans le cas où $x \leq b$ alors `ABBcherche(B,x)` fait appel à `ABBcherche(G,x)` qui se termine.
   Remarquons que x est une valeur de B ssi x est une valeur de G (car toutes les valeurs de D sont strictement supérieures à b). Par hypothèse de récurrence, `ABBcherche(G,x)` retourne la valeur `True` ssi x est une valeur de G et donc ssi x est une valeur de B.
   Le cas où $x > b$ est analogue.

   Conclusion On a prouvé, par induction structurelle, que $P(B)$ est vraie pour tout ABB B.
4. Dans le pire cas, l'arbre balisé B est un peigne et la valeur cherchée est tout en bas de B (ou n'est pas dans B mais devrait se trouver tout en bas de B), le pire cas est donc en $O(f)$.

   Dans le meilleur cas, l'arbre balisé B est un peigne et la valeur cherchée est à la feuille la plus proche de la racine de B (ou devrait s'y trouver), le meilleur cas est donc en $\Omega(1)$.
   ```

Exercice 2 – Graphes biconnexes et point d'articulation (12 points + 2 bonus)

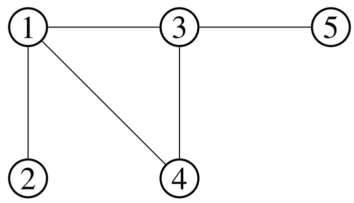
Dans cet exercice, $G = (V, E)$ est un graphe **non orienté connexe**. On rappelle qu'un graphe est *connexe* si pour tout couple de sommets $(x, y) \in V^2$, il existe une chaîne qui relie x à y dans G. Une chaîne (resp. un cycle) est *élémentaire* si elle (resp. il) ne passe pas deux fois par le même sommet.

Question 1
On considère dans cette question le graphe non orienté connexe $G_1 = (V_1, E_1)$ avec $V_1 = \{1, 2, 3, 4, 5\}$ et $E_1 = \{\{1, 2\}, \{1, 3\}, \{1, 4\}, \{3, 4\}, \{3, 5\}\}$.

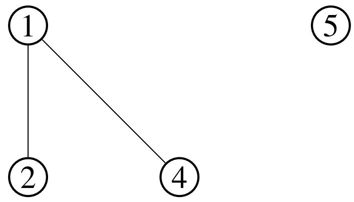
1. Représentez G_1 graphiquement.
2. Donnez la représentation de G_1 sous la forme d'une matrice sommet-sommet.
3. Donnez la représentation de G_1 sous la forme de listes d'adjacence.
4. Représentez graphiquement le sous-graphe induit $G_2 = (V_1 - \{3\}, E_1)$. G_2 est il connexe ? Justifiez votre réponse dans la négative.

Solution:

1.



2. On obtient la matrice 5×5 suivante : $M_1 = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$.
3. On obtient les listes $L[1] = (2, 3, 4)$, $L[2] = (1)$, $L[3] = (1, 4, 5)$, $L[4] = (1, 3)$, et $L[5] = (3)$.
4. Le graphe G_2 n'est pas connexe. Par exemple, il n'y a pas de chaîne reliant les sommets 2 et 5.



Question 2
Soit G un graphe non orienté connexe et la relation $\mathcal{R}$ définie sur V^2 de la manière suivante : pour tout couple $(x, y) \in V^2$, $x\mathcal{R}y$ si il existe un cycle élémentaire qui contient x et y.

Par convention, on considère que, pour tout $x \in V$, $x\mathcal{R}x$. De même, pour tout arête $e = \{x, y\} \in E$, $c = (x, y, x)$ est un cycle élémentaire et donc $x\mathcal{R}y$ et $y\mathcal{R}x$.

Un graphe $G = (V, E)$ est *biconnexe* si $\mathcal{R} = V \times V$.

1. Donnez la relation $\mathcal{R}$ associée au graphe G_1. On pourra la représenter sous la forme d'une matrice $M_{\mathcal{R}}$ à valeur dans $\{0, 1\}$ de taille $V \times V$ avec $M_{\mathcal{R}}[x, y] = 1$ si $x\mathcal{R}y$.
2. Est-ce que $\mathcal{R}$ est une relation d'équivalence ? Justifiez votre réponse.
3. Est-ce que le graphe G_1 de la question 1 est biconnexe ? Dans la négative, quelles sont les arêtes que l'on peut rajouter au minimum pour obtenir à partir de G_1 un graphe biconnexe.

Solution:

1. On obtient $M_{\mathcal{R}} = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix}$.
2. La relation $\mathcal{R}$ n'est pas une relation d'équivalence car elle n'est pas transitive. En effet, on a par exemple, $2\mathcal{R}1$, $1\mathcal{R}3$ et 1 et 3 ne sont pas en relation.
3. G_1 n'est pas biconnexe, car on peut avoir des sommets de G_1 qui ne sont pas en relation, par exemple 1 et 5. Pour obtenir un graphe biconnexe, on peut par exemple rajouter à G_1 les arêtes $\{2, 4\}$ et $\{4, 5\}$.

Question 3

On considère dans cette question la liste $L = (1, 3, 5, 4, 2)$.

1. Est-ce que L est un parcours générique de G_1 ? Dans l'affirmative, donnez un graphe de liaison associé. Est-ce que le graphe de liaison est unique ? Justifiez votre réponse.
2. Rappeler la définition d'un parcours en largeur. Est-ce que L est un parcours en largeur ? Dans l'affirmative, donnez un graphe de liaison en largeur associé. Dans le cas général, est-ce que le graphe de liaison d'un parcours en largeur est unique ? Justifiez votre réponse.
3. Rappeler la définition d'un parcours en profondeur. Est-ce que L est un parcours en profondeur ? Dans l'affirmative, donnez un graphe de liaison en profondeur associé.

Solution:

1. L est bien un parcours générique de G_1. $\mathcal{A} = (V_1, \{(1, 2), (1, 3), (3, 4), (3, 5)\})$ est un graphe de liaison associé à L. Le graphe de liaison n'est pas unique, par exemple $\mathcal{A}' = (V_1, \{(1, 2), (1, 3), (1, 4), (3, 5)\})$ est également un graphe de liaison associé à L.
2. Soit $G = (V, E)$ un graphe non orienté connexe et $L = (v_1, \ldots, v_n)$ un parcours de G d'origine v_1. L est un *parcours en largeur* si pour tout sous-parcours $L_k = (v_1, \ldots, v_k)$ avec $k < n$, v_{k+1} est un sommet adjacent du **premier** sommet ouvert de L_k.

   L n'est pas un parcours en largeur car le sous-parcours $L_1 = (1, 3)$ ne vérifie pas la définition. En effet, le premier sommet ouvert de L_1 est 1 et 4 n'a pas encore été visité quand on visite 5.

   Le graphe de liaison en largeur est unique car il relie, pour tout $k \in \{1, \ldots, n-1\}$, le premier sommet ouvert de L_k à v_{k+1}.
3. Soit $G = (V, E)$ un graphe non orienté connexe et $L = (v_1, \ldots, v_n)$ un parcours de G d'origine v_1. L est un *parcours en profondeur* si pour tout sous-parcours $L_k = (v_1, \ldots, v_k)$ avec $k < n$, v_{k+1} est un sommet adjacent du **dernier** sommet ouvert de L_k.

   L est bien un parcours en profondeur. $\mathcal{A} = (V_1, \{(1, 2), (1, 3), (3, 4), (3, 5)\})$ est le graphe de liaison en profondeur associé à L.

Question 4

Un *point d'articulation* de $G = (V, E)$ est un sommet $x \in V$ tel que le sous-graphe induit $G' = (V - \{x\}, E)$ n'est pas connexe. Est-ce que le graphe G_1 de la question 1 possède un ou plusieurs points d'articulation ? Dans l'affirmative, quels sont-ils ?

Solution:

Les deux points d'articulation de G_1 sont 1 et 3.

Question 5

On rappelle que $G = (V, E)$ est un graphe non orienté connexe. Montrez que, si G est biconnexe, alors G ne possède pas de point d'articulation. Pour cela, vous pouvez démontrer la contraposée.

Solution:

On démontre la contraposée : soit $G = (V, E)$ un graphe connexe possèdant un point d'articulation x. Montrez que G n'est pas biconnexe.

Par définition, le sous-graphe $G' = (V - \{x\}, E)$ n'est pas connexe. Soient alors i et j deux sommets dans deux composantes connexes différentes dans G'. Alors, toute chaîne qui relie i à j dans G passe forcément par x. On ne peut donc pas construire un cycle élémentaire dans G qui passe par i et j, G n'est donc pas biconnexe.

Question 6

On suppose dans cette question que G ne possède pas de point d'articulation. On souhaite démontrer que G est alors biconnexe.

Pour cela, on suppose le résultat suivant : soient x, y et z trois sommets distincts tels que x et y sont inclus dans un cycle élémentaire C et que l'arête $\{y, z\} \in E$. Alors, il existe un cycle élémentaire C' qui contient x et z.

1. Démontrez par récurrence sur i la propriété $\Pi(i)$ suivante pour $i \geq 2$: soit $x_1, x_2, \ldots x_i$ une chaîne élémentaire constituée de i sommets. Alors il existe un cycle contenant x_1 et x_i.
2. En déduire que si G ne possède pas de point d'articulation, alors G est biconnexe.

Solution:

1. On démontre $\Pi(i)$ par récurrence faible sur i.

   Base : Pour $i = 2$, $c = (x_1, x_2, x_1)$ est par convention un cycle élémentaire. Ainsi, $\Pi(2)$ est vérifiée.

   Induction : Supposons maintenant par récurrence faible que, pour une valeur $i > 2$, $\Pi(i - 1)$ soit vérifiée. Supposons également que G possède une chaîne élémentaire $x_1, x_2, \ldots, x_i$. Alors, il y a un cycle élémentaire qui contient x_1 et x_{i-1} et une arête $e = \{x_{i-1}, x_i\}$. D'après le résultat supposé, il y a donc un cycle élémentaire qui contient x_1 et x_i, et $\Pi(i)$ est vérifiée.

   Conclusion : Pour tout $i \geq 2$, $\Pi(i)$ est vérifiée par récurrence faible.
2. Supposons que G est connexe et ne possède pas de point d'articulation. Pour tout couple de sommets $(x, y) \in V^2$, par connexité de G, il existe une chaîne élémentaire qui les relie ; donc d'après la question 6.1, il existe un cycle élémentaire contenant x et y. On en déduit que G est biconnexe.

Question 7

On souhaite maintenant développer un algorithme qui détermine si un graphe connexe $G = (V, E)$ est biconnexe.

1. Citez le nom d'un algorithme du cours qui permet de déterminer si un graphe G est connexe.
2. En déduire le principe d'un algorithme (décrit en maximum trois phrases) pour déterminer si un graphe G connexe est biconnexe.

Solution:

1. D'après le cours, G est connexe si et seulement si il existe un parcours du graphe. Il suffit donc de construire un parcour générique L et vérifier que tous les sommets ont été visités par L.
2. Pour tout sommet $x \in V$, on teste que le sous-graphe induit $G_x = (V - \{x\}, E)$ est un graphe connexe en construisant un parcours générique. Si tous les graphes G_x obtenus sont connexes, G est biconnexe. Sinon, il existe au moins un point d'articulation, et donc G n'est pas biconnexe.

Question 8 – Bonus
Supposons que G ne possède pas de point d'articulation. Soient x, y et z trois sommets distincts tels que x et y sont inclus dans un cycle élémentaire C et que l'arête $\{y, z\} \in E$. Démontrez qu'il existe un cycle élémentaire C' qui contient x et z.

__Solution__:
Si z est un sommet de C, alors $C' = C$ vérifie la propriété. On suppose par la suite que z n'est pas un sommet de C. Par hypothèse, y n'est pas un point d'articulation. Donc, il existe une chaine élémentaire ν de x à z qui ne passe pas par y. Soit k le dernier élément de cette chaine qui est dans C. On construit un circuit élémentaire qui contient x et z de la manière suivante :

— aller de x à k en suivant C sans passer par y ;
— aller de k à z en prenant une sous-chaine de ν ;
— l'arête $\{z, y\}$;
— et revenir à x en utilisant la partie de C non encore utilisée.