GLIN 408 : Analyse algorithme

Aucun document autorisé

1 Notation O et Θ

On rappelle les définitions de base : soit deux fonctions $f(n)$ et $g(n)$ de $\mathbb{N} \to \mathbb{N}$,

$$f \in \Theta(g) \iff \{f \in O(g) \text{ et } g \in O(f)\} \text{ et}$$

$$f \in O(g) \iff \{\exists N_0 \in \mathbb{N},\ \exists k \in \mathbb{N} \mid \forall n \geq N_0\ :\ f(n) \leq kg(n)\}$$

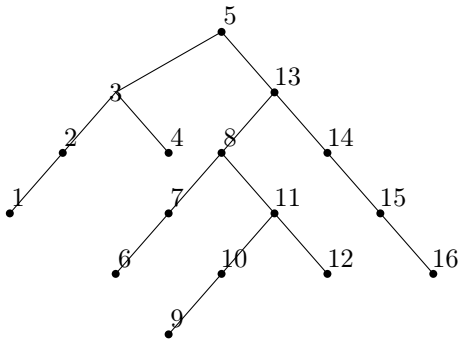
Question 1 *justifiez vos réponses (vrai ou faux) aux deux questions suivantes :*

1. $2^{n+1} \in O(2^n)$
   vrai car $2^{n+1} = 2 * 2^n$
2. $2^n \in \Theta(2^{2n})$ faux
   - $2^n \in O(2^{2n})$ est vrai car $2^n < 2^{2n}$
   - mais $2^{2n} \in O(2^n)$ est faux :

     $$\{2^{2n} < k * 2^n\} \iff \{2^n < k\} \text{ en divisant par } 2^n$$

     ce qui ne peut avoir lieu avec k constant

2 AVL



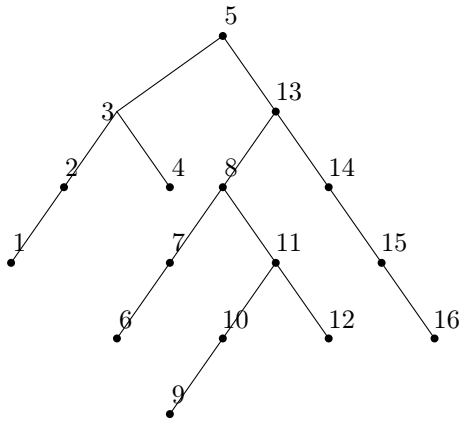
Question 2 *En rajoutant un minimum de valeurs (non entières) transformer l'ABR ci dessus en AVL (indiquer les valeurs à rajouter)*

Regardons la balance de chacun des sommets

1. : zéro
2. : un
3. : un
4. : zéro
5. : moins deux
6. : zéro
7. : un
8. : moins un
9. : zéro
10. : un
11. : un
12. : zéro
13. : un
14. : moins deux
15. : moins un
16. : zéro

Donc il faut augmenter la profondeur de la sous arborescence gauche du sommet 5 et du sommet 14
Mais si en rajoutant la valeur 13,5 on rétablit la balance du sommet 14, pour rétablir la balance du sommet 5 il faut rajouter les sommets 1,5, 2, 5 et 4,5.

3 ABR



Question 3 *On veut supprimer le sommet d'étiquette 8 dans l'ABR ci dessus.*

- indiquez les différentes étapes de l'algorithme

- indiquer comment modifier le dessin pour représenter le résultat obtenu par l'algorithme

- on recherche le sommet minimum dans la sous arborescence de droite du sommet 8 (pour obtenir 9)
- on supprime ce sommet minimum
- on remplace l'étiquette 8 par l'étiquette 9

Question 4 *On suppose que des entiers compris entre 1 et 1000 sont disposés dans un arbre binaire de recherche et que l'on souhaite trouver le nombre 363. Parmi les séquences suivantes, lesquelles* ***ne*** *pourraient* ***pas*** *être la suite des nœuds parcourus ? Justifiez chacune de vos réponses (ne pas justifier les séquences qui peuvent être la suite des nœuds parcourus).*

1. *2, 252, 401, 398, 330, 344, 397, 363.*
2. *924, 220, 911, 244, 898, 258, 362, 363.*
3. *925, 202, 911, 240, 912, 245, 363.*
   240 et 912 ne peuvent être dans la même sous arborescence de racine 202
4. *2, 399, 387, 219, 266, 382, 381, 278, 363.*
5. *935, 278, 347, 621, 299, 392, 358, 363.*
   299 et 621 ne peuvent être dans la même sous arborescence de racine 347

Question 5 *Dans l'arbre binaire de recherche que nous considérons dans ce problème, toutes les clés sont différentes.*
Pour un sommet S, on notera $cle(S)$ la clé de ce sommet.

1. *à quelle condition un sommet S est-il le sommet ayant la plus petite clé ?*
   Il faut
   – que S n'aie pas de sous arbo gauche
   – et que S soit dans les sous arborescences gauches de tous ses ancêtres

   Si S n'est pas le sommet qui a la plus petite clé, et si toutes les clés sont différentes, on appelle $Pred(S)$ le sommet qui a la clé imédiatement inférieure à celle de S
2. *Dans le premier cas que nous examinons, S a une sous arborescence gauche*
   (a) *où se trouve alors $Pred(S)$? appeler ce sommet $Max(S)$ dans les explications suivantes*
   c'est le plus grand sommet de la sous arborescence gauche de S
   (b) *expliquer pourquoi pour tout sommet T*
      i. *qui se trouve dans la sous arborescence gauche de S, $T \neq Max(S) \Rightarrow$ T ne peut pas être le prédecesseur de S*
      car $cle(T) < cle(Max(S)) < cle(S)$ Q.E.D.
      ii. *qui se trouve dans la sous arborescence droite de S, T ne peut pas être le prédecesseur de S*
      car $cle(T) > cle(S)$ Q.E.D.
      iii. *qui se trouve sur le chemin de S à la racine, T ne peut pas être le prédecesseur de S (il faudra examiner deux cas, suivant que S est dans la sous arborescence gauche ou droite de T)*
      – si S est dans la sous arborescence gauche de T, $cle(S) < cle(T)$ Q.E.D.
      – si S est dans la sous arborescence droite de T, alors $Max(S)$ aussi et $cle(T) < cle(Max(S)) < cle(S)$ Q.E.D.
      iv. *qui se trouve ailleurs dans l'arbre, T ne peut pas être le prédecesseur de S (il faudra considérer U l'ancêtre commun à S et T le plus bas dans l'arbre, puis examiner deux cas)*
      Remarquons d'abord que S et T ne peuvent être dans la même sous arborescence de U (car sinon S et T auraient un ancêtre commun plus bas)
      – si S est dans la sous arborescence gauche de U et T dans la sous arborescence droite alors $cle(S) < cle(U) < cle(T)$ Q.E.D.
      – si S est dans la sous arborescence droite de U et T dans la sous arborescence gauche alors $cle(T) < cle(U) < cle(S)$ Q.E.D.

3. *Dans le deuxième cas que nous examinons, S n'a pas de sous arborescence gauche :*
   (a) *indiquer alors où se trouve $Pred(S)$ et l'appeler $Bas(S)$ dans les explications ultèrieures.*
   c'est le plus bas ancêtre de S, nomons le $Bas(S)$, pour lequel S est dans la sous arborescence droite
   (b) *le prouver de la même manière que pour le premier cas*
   On va regarder les différents cas suivants pour un sommet $T \neq Bas(S)$
   - T dans la sous arborescence gauche de S : cas impossible, car S n'a pas de sous arborescence gauche
   - T dans la sous arborescence droite de S : alors $cle(T) > cle(S)$ Q.E.D.
   - T un ancêtre de S ; il faut examiner deux cas
     - S est dans la sous arborescence gauche de T, mais alors $cle(S) < cle(T)$ Q.E.D.
     - S est dans la sous arborescence droite de T, mais alors $Bas(S)$ aussi et $cle(T) < cle(Bas(S)) < cle(S)$ Q.E.D.
   - T est ailleurs dans l'arborescence, appelons alors U le plus bas ancêtre commun à T et S. Comme précédement, T et S ne peuvent appartenir à la même sous arborescence de U.
     - Si S est dans la sous arborescence droite de U et T dans la sous arborescence gauche, alors $cle(T) < cle(U) < cle(S)$ Q.E.D.
     - Si S est dans la sous arborescence gauche de U et T dans la sous arborescence droite, alors $cle(S) < cle(U) < cle(T)$ Q.E.D.

Question 6 *Dans un arbre binaire de recherche dont toutes les clés sont différentes, pour lequel on dispose des fonctions (en* $\Theta(1)$*)*

- *racine?*(S) *qui indique si un sommet* S *est la racine de l'arbre*
- *pere*(S)*, défini pour les autres sommets que la racine et qui renvoie le père du sommet* S
- *val*(S) *qui renvoie l'étiquette du sommet* S
- *sag*(S) *qui renvoie la sous arborescence gauche du sommet* S
- *sad*(S) *qui renvoie la sous arborescence droite du sommet* S
- $fd?(S)$*, définie pour les autres sommets que la racine et qui indique si un sommet* S *est le fils droit de son père*
- $fg?(S)$*, définie pour les autres sommets que la racine et qui indique si un sommet* S *est le fils gauche de son père*
- *nonVide?*(A) qui indique que l'arborescence A n'est pas vide

1. *écrire la fonction pred*(S) *(une solution avec des fonctions auxiliaires sera mieux notée).*

   Max
   Données: un ABR non vide de racine S
   Résultat: le sommet de plus grande clé de l'ABR
   si *nonVide?*$(sad(S))$ **alors**
   retourner *Max(sad*(S)*)*
   sinon
   retourner S
   fin

   Bas
   Données: un ABR non vide de racine S
   Résultat: le premier ancêtre (s'il existe) pour lequel S est dans la sous arborescence droite
   si *racine?*(S) **alors retourner** Null;
   si $fd(S)$ **alors retourner** *pere*(S);
   retourner *Bas(pere*(S)*)*

   Pred
   Données: un sommet S d'un ABR
   Résultat: le sommet (s'il existe) de clé maximum mais inférieure à celle de S
   si *nonVide*$(sad(S))$ **alors**
   retourner Max$(sad(S))$
   sinon
   retourner Bas(S)
   fin

2. *Donner et justifier la classe de complexité de votre fonction. (on choisira le bon paramètre, pour que votre fonction soit intéressante si l'ABR est aussi un AVL).*
   $\Theta(h)$