

Travaux Dirigés d'informatique n°8

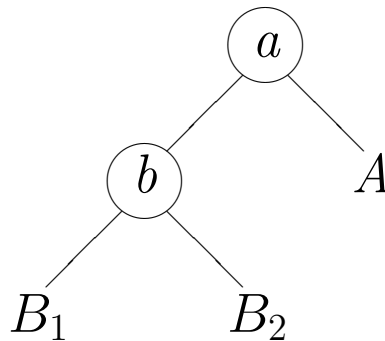
Algorithmique

Arbres binaires de recherche équilibrés.

On notera $h(T)$ la hauteur de l'arbre T . Si T_1 est le fils gauche de T et T_2 le fils droit de T , on note $balance(T) = h(T_2) - h(T_1)$ la balance de l'arbre T .

Exercice 1 Manipulations.

1. Effectuer respectivement une rotation droite sur la racine de l'arbre suivant, puis une rotation gauche sur la racine de l'arbre résultant:



2. Supposons que, dans l'arbre ci-dessus, les sous-arbres A, B_1, B_2 soient équilibrés.

(a) Que dire du sous-arbre dont la racine est b si $|balance(b)| \leq 1$.

Dans la suite, on supposera cette propriété vérifiée et on supposera que la balance de l'arbre, dont la racine est a , est -2 ;

(b) effectuer les rotations nécessaires afin d'équilibrer l'arbre ci-dessus dans le cas où la balance du sous-arbre dont le nœud b est la racine vaut 0 ou -1 ;

(c) effectuer les rotations nécessaires afin d'équilibrer l'arbre ci-dessus dans le cas où la balance du sous-arbre dont le nœud b est la racine vaut 1 ;

(d) calculer la balance de chacun des sous-arbres de l'arbre obtenu après la/les rotation(s) effectuée(s);

3. construire l'AVL par insertion successive des entiers suivants : 7, 12, 25, 16, 21, 13, 2, 5, 4, 9, 30, 27, 6, 11;

4. supprimer successivement les entiers 30, 27, 16 dans l'arbre construit précédemment en respectant la structure d'arbre AVL;

Exercice 2 Programmatons.

Dans tout cet exercice, on utilisera la structure suivante pour définir un AVL:

```
typedef struct noeud
{
    int valeur;
    int balance;
    struct noeud *fg, *fd;
} Noeud, *AVL;
```

- *Ecrire une fonction `int EstAVL(AVL A)` qui teste si l'arbre binaire `A` est un arbre binaire de recherche équilibré;*
- *écrire une fonction `void RotationD(AVL *A)` qui effectue une rotation droite de l'arbre `A`. On n'oubliera pas de rectifier la balance des nœuds concernés;*
- *écrire une fonction `void RotationG(AVL *A)` qui effectue une rotation gauche de l'arbre `A`. On n'oubliera pas de rectifier la balance des nœuds concernés;*
- *écrire une fonction `void RotationGD(AVL *A)` qui effectue une rotation gauche/droite de l'arbre `A`. On n'oubliera pas de rectifier la balance des nœuds concernés. Une rotation gauche/droite est la double rotation effectuée à la question 2.c de l'exercice 1;*
- *écrire une fonction `void Equilibre(AVL *A)` qui choisit et appelle les fonctions de rotation afin d'équilibrer l'arbre `A`. On supposera que les fils de `A` sont des AVL;*
- *écrire une fonction `void Insertion(AVL *A, int x)` qui effectue l'insertion de l'entier `x` dans l'AVL `A`. On effectuera les modifications de la balance des nœuds ainsi que les rotations, si nécessaire, afin de conserver la structure d'arbre AVL;*
- *écrire une fonction `AVL Extraction(AVL *A, int x)` qui effectue la suppression de l'entier `x` dans l'AVL `A`. On effectuera les modifications de la balance des nœuds ainsi que les rotations, si nécessaire, afin de conserver la structure d'arbre AVL.*