2018-2019, Semestre d'automne
L3, Licence Sciences et Technologies
Université Lyon 1

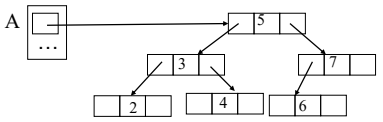
LIFAP6: Algorithmique, Programmation et Complexité

Chaine Raphaëlle (responsable semestre automne)
E-mail : raphaelle.chaine@liris.cnrs.fr
http://liris.cnrs.fr/membres?idn=rchaine

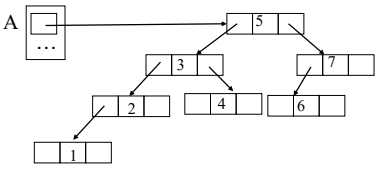
ABR équilibré

- Un ABR est en équilibre parfait si sa hauteur est minimale
- Pour chaque nœud, le nombre d'éléments du sous arbre gauche et du sous arbre droit diffèrent de 1
- Configuration obtenue en insérant l'élément médian m, puis l'élément médian des éléments <= m puis l'élément médian des éléments > m …
- Dur à maintenir, une simple insertion peut entrainer toute une réorganisation de l'arbre!

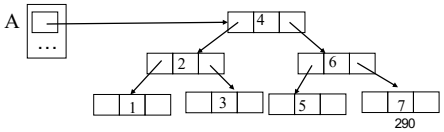
Arbre initial



Arbre initial dans lequel on insère 1



Retour à l'équilibre



- Pour éviter des réorganisations douloureuses en temps de calcul, on assouplit les conditions d'équilibrage,
  - pour limiter le coût de la réorganisation
  - tout en gardant des performances logarithmiques pour la recherche, l'insertion et la suppression!

AVL

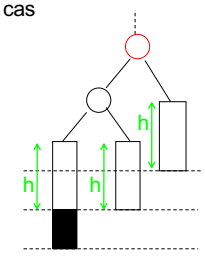
- Du nom des auteurs de la méthode : Adelson-Velskij et Landis
- Garantir après chaque opération que
  - pour chaque nœud, les hauteurs des sous-arbres gauche et droit diffèrent au plus de 1
- Cet équilibre peut-être maintenu à travers l'utilisation judicieuse de 4 opérations :
  - Rotation gauche
  - Rotation droite
  - Rotation droite-gauche (ou rotation double à gauche)
  - Rotation gauche-droite (ou rotation double à droite)

Rééquilibrage après une insertion

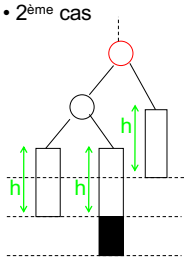
- Si il apparait un nœud ◯ au niveau duquel le déséquilibre devient égal à 2 (mais dont les fils restent équilibrés)

Élément inséré

- 1er cas



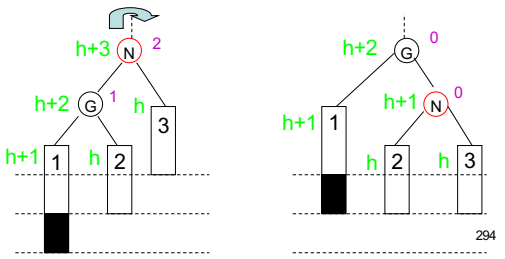
- 2ème cas



Rotation simple à droite

- A faire dans le 1er cas : déséquilibre « à gauche toute »

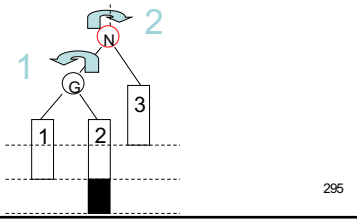
Hauteurs h1=h+1 h2=h3=h



294

Rotation double à droite

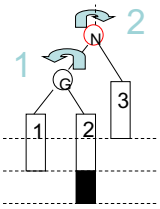
- A faire dans le 2ème cas : déséquilibre « à droite de la gauche»
- La rotation double à droite commence par une rotation gauche sur le sous arbre G **puis** une rotation droite sur N



295

Rotation double à droite

- Détail supplémentaire à mettre en œuvre dans le 2ème cas (déséquilibre « à droite de la gauche»)
  - Mise à jour des déséquilibres des nœuds



a) Sous-cas où les sous-arbres 1, 2 et 3 sont vides avant l'insertion (h=0)
b) Sous cas où le sous-arbre 2 est déséquilibré sur sa gauche
c) Sous cas où le sous-arbre 2 est déséquilibré sur sa droite

296

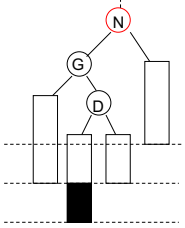
296

Rotation double à droite

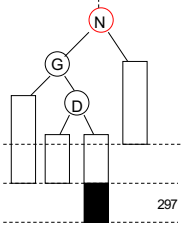
- A faire dans le 2ème cas : déséquilibre « à droite de la gauche» de ◯

■ Élément inséré

• sous-cas b



• sous-cas c



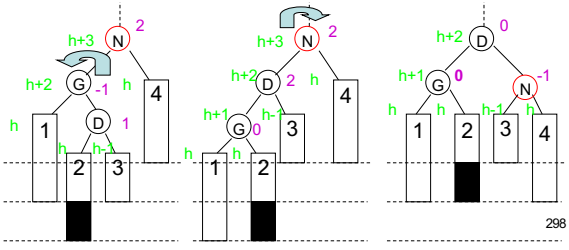
297

Rotation double à droite

- Cas 2b : déséquilibre à gauche de la droite du sous-arbre gauche

Hauteurs h1=h2=h4=h et h3=h-1

- Combinaison d'une rotation gauche puis d'une rotation droite



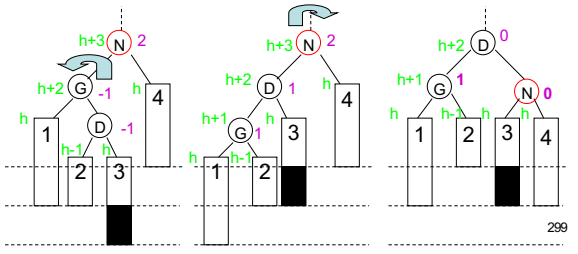
298

Rotation double à droite

- Cas 2c : déséquilibre à droite de la droite du sous-arbre gauche

Hauteurs h1=h3=h4=h h2=h-1

- Combinaison d'une rotation gauche puis d'une rotation droite

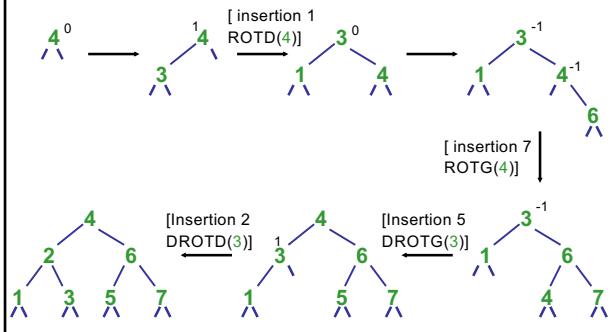


299

- Les symétriques de ces opérations existent dans le cas d'un déséquilibre alourdissant un sous-arbre droit après une insertion
  - Rotation gauche
  - Rotation droite-gauche
- Dans les 4 cas (et leurs symétriques) la hauteur du sous-arbre de racine n retrouve sa hauteur d'avant l'insertion de l'élément ■

- Ajouts successifs de **4**, **3**, **1**, **6**, **7**, **5**, **2** dans l'arbre vide

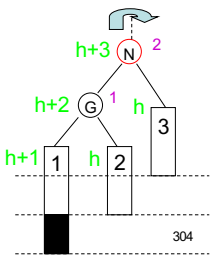
- Ajouts successifs de **4**, **3**, **1**, **6**, **7**, **5**, **2** dans l'arbre vide



- Comment pister / stocker les déséquilibres?
  - 1ère solution : Conserver l'info de hauteur dans chaque nœud
  - 2ème solution : Ne conserver dans chaque nœud que la **différence** de hauteur entre le sous-arbre gauche et le sous-arbre droit

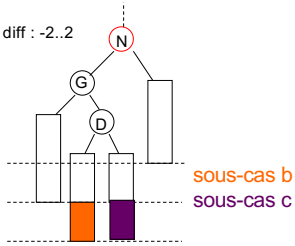
Rotation droite

```
procédure AVL_Rotation_Droite
    (donnée-résultat a : ABR, pN : pointeur sur Nœud)
    Précondition : pN->diff=2 et pn->gauche->diff=1 (cas 1)
    variables :
        pG : pointeur sur Nœud
    début
        pG ← pN->gauche
        pN->gauche ← pG->droite
        pG->droite ← pN
        pN ← pG
        pN->diff ← 0; pn->droite->diff ← 0
    fin
```



Rotation double à droite

```
procédure AVL_Rotation_Double_Droite
    ( donnée-résultat a : ABR, pN : pointeur sur Nœud )
    Précondition : pN->diff=2 et pn->gauche->diff=-1 (cas 2 ou 3)
    variables :
        pG, pGD : pointeur sur Nœud, diff : -2..2
    début
        pG ← pN->gauche
        pGD ← pN->gauche->droite
        pG->droite ← pGD->gauche
        pGD->gauche ← pG
        pN->gauche ← pGD->droite
        pGD->droite ← pN
        diff ← pGD->diff ;
        pN ← pGD
        pN->gauche->diff ← max(0,-diff) ; pN->droite->diff ← min(0,-diff);
        pN->diff ← 0
    fin
```



```
procédure équilibrer( donnée-résultat a : ABR, pN : pointeur sur Nœud )
   Précondition : pN->diff=2 ou -2
   début
      si pN->diff=2 alors
         si pN->gauche->diff=1 alors
            AVL_Rotation_Droite(a,pN)
         sinon
            AVL_Rotation_Double_Droite(a,pN)
         finsi
      sinon
         si pN->droite->diff=-1 alors
            AVL_Rotation_Gauche(a,pN)
         sinon
            AVL_Rotation_Double_Gauche(a,pN)
         finsi
      finsi
   fin
```


306

Insertion dans un AVL

- Insertion aux feuilles : parcours d'un chemin menant de la racine au nouvel élément
- Parcours du chemin en sens inverse,
  - pour mettre à jour les différences de hauteur des nœuds ancêtres,
  - et effectuer une opération de rééquilibrage si nécessaire!
- Parcours du chemin en sens inverse :
  - facile si la procédure d'insertion est récursive, (il suffit de mettre les instructions relatives à la remontée après l'appel récursif)
  - la remontée peut s'arrêter à partir du moment où le champ diff d'un nœud n'est plus modifié

307

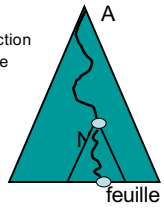
Complexité

- Temps d'une rotation : constant
- Note : Insertion exécute AU PLUS UNE rotation car une rotation sur un nœud rétablit la hauteur initiale (avant insertion) du sous-arbre correspondant
- *A* arbre AVL à *n* nœuds
- Temps total d'un ajout : $O(h(A)) = O(\log n)$ car une seule branche de l'arbre est examinée

308

- Version itérative
  - C'est uniquement sur le chemin racine / nouvelle feuille qu'il peut y avoir des rotations
  - On considère sur ce chemin le nœud N le plus bas (le dernier rencontré à la descente) dont le champ *diff != 0*
  - On peut démontrer que si rotation il doit y avoir après l'insertion ca sera sur ce Noeud

Entre A et N l'adjonction n'a pas pu causer de problème.



N= nœud le + bas t. que N->diff != 0

309