Algorithme et Structures de Données

Wurtz Jean-Marie

Université Louis Pasteur

Wurtz@igbmc.u-strasbg.fr

Généralisation des 2-3-4 Arbres

(B Tree)

Les B-Arbres

- Introduit par Bayer et McCreigth en 1970, premiers à étudier des arbres à liens multiples
- B-arbre :
  - chaque nœud a :
    - au plus M entrées
    - et au moins M/2 entrées
    - sauf la racine qui doit avoir au moins 1 entrée (2 liens)
- B-tree : nom générique ou nom spécifique pour une structure de données

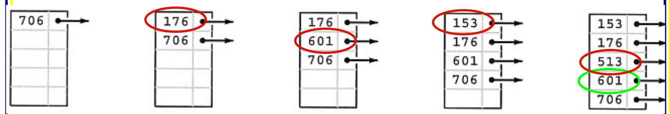
Généralisation des 2-3-4-Arbres : exemple d'un 4-5-6-7-8 arbre

- Un nœud peu contenir jusqu'à 8 liens
- Insertion de « J » ?
  - éclatement en 2 nœuds et remonté de « M » dans le nœud racine

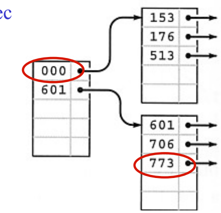
B-Arbre : Structure de donnée

```
class ST {   // Table des Symboles
    private class entry {
        KEY key; ITEM item; Node next;
        entry(KEY v, ITEM x) { key = v; item = x; }
        entry(KEY v, Node u) { key = v; next = u; }
    }
    private class Node {
        int m;                          // nb d'éléments dans le noeud
        entry[] b;
        Node(int k) { b = new entry[M]; m = k; }
    }
    private Node head;
    private int HT;                     // hauteur de l'arbre

    ST() { HT = 0; head = new Node(0); }

    ITEM search(KEY key) {  ... }
    void insert(ITEM x) { ... }
}
```


Insertion dans un B-tree

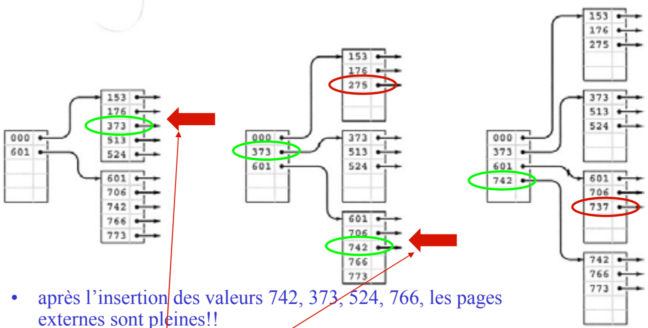


- 6 insertions dans un B-tree avec M = 5
- Une page peut contenir 5 clefs
- insertion des valeurs : 706, 176, 601, 153, 513
- puis : 000, qui éclate le nœud racine en deux
- puis insertion de 773



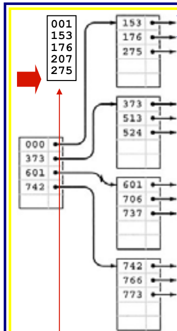
D O N N E E S

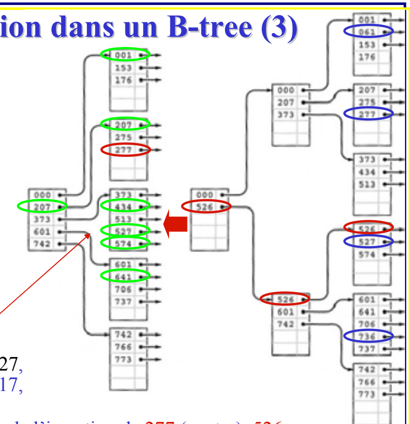
Insertion dans un B-tree (2)



- après l'insertion des valeurs 742, 373, 524, 766, les pages externes sont pleines!!
- L'insertion de 275 entraîne l'éclatement de la première page, la valeur 373 est remontée dans le nœud parent
- insertion de : 737

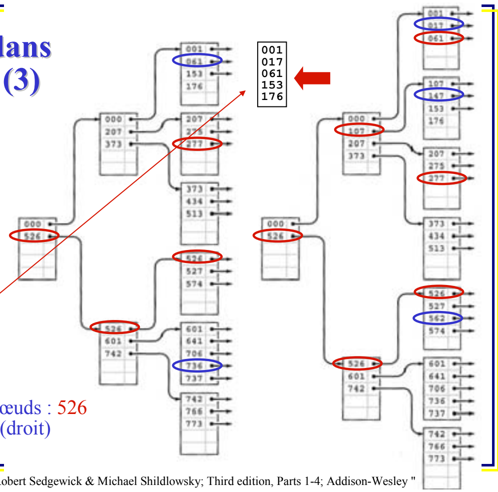
Insertion dans un B-tree (3)





- insertion de 14 nombres : 574, 434, 641, 207, 001, 527, 277, 061, 736, 526, 562, 017, 107 et 147.
- Éclatement de nœuds : lors de l'insertion de 277 (centre), 526 (gauche) et 107 (diapositive suivante)

Insertion dans un B-tree (3)



- insertion de : 277, 061, 736, 526, 562, 017, 107 et 147.
- Éclatement de nœuds : 526 (gauche) et 107 (droit)

L'insertion dans un B-Arbre

```
private Node insertR(Node h, ITEM x, int ht) {
    int i, j; KEY v = x.key(); Node u;
    entry t = new entry(v, x);
    if (ht == 0)        // une feuille
        for (j = 0; j < h.m; j++)
            { if (less(v, (h.b[j]).key)) break; }
    else                // un noeud interne
        for (j = 0; j < h.m; j++)
            if ((j+1 == h.m) || less(v, (h.b[j+1]).key)) {
                u = insertR(h.b[j++].next, x, ht-1);
                if (u == null) return null;        // pas de réarrangement du noeud
                t.key = (u.b[0]).key;
                t.next = u;
                break;
            }
    for (i = h.m; i > j; i--)
        h.b[i] = h.b[i-1];
    h.b[j] = t; h.m++;
    if (h.m < M) return null;
    else return split(h);
}
```

```
void insert(ITEM x) {
    Node u = insertR(head, x, HT);
    if (u == null) return;  // racine inchangée
    Node t = new Node(2);
    t.b[0] = new  entry((head.b[0]).key, head);
    t.b[1] = new entry((u.b[0]).key, u);
    head = t; HT++;
}
```


B-Arbre : Eclatement d'noeud

```
private Node split(Node h) {
    Node t = new Node(M/2);
    h.m = M/2;
    for (int j = 0; j < M/2; j++)
        t.b[j] = h.b[M/2+j];
    return t;
}
```

M : est paire
On ne permet que M-1 ITEM par noeud, cela permet d'insérer le Mème élément dans le nœud avant l'éclatement du noeud

La recherche dans un B-Arbre

```
private ITEM searchR(Node h, KEY v, int ht) {
    if (ht == 0)                    // une feuille de l'arbre
        for (int j = 0; j < h.m; j++) {
            entry e = h.b[j];
            if (equals(v, e.key)) return e.item;
        }
    else                            // sinon recherche le bon noeud à suivre
        for (int j = 0; j < h.m; j++)
            if ((j+1 == h.m) || less(v, h.b[j+1].key))
                return searchR(h.b[j].next, v, ht-1);
    return null;
}
ITEM search(KEY key)
    { return searchR(head, key, HT); }        // HT : hauteur de l'arbre
```

Propriétés d’un B-Arbre

- Une recherche ou une insertion dans un B-arbre d’ordre M avec N éléments requière entre $\log_M N$ et $\log_{M/2} N$ test
  - Car les noeuds ont entre M/2 et M éléments
  - M=1000 la hauteur de l’arbre est inférieur à 3 pour N=125 millions
  - En chargeant en mémoire la racine cela revient à effectuer 2 accès
- Un B-Arbre d’ordre M construit à partir de N nombres aléatoires compte 1.44*N / M feuilles (des pages)