

Indexation et structures d'arbres

Talel.Abdessalem@enst.fr

Introduction

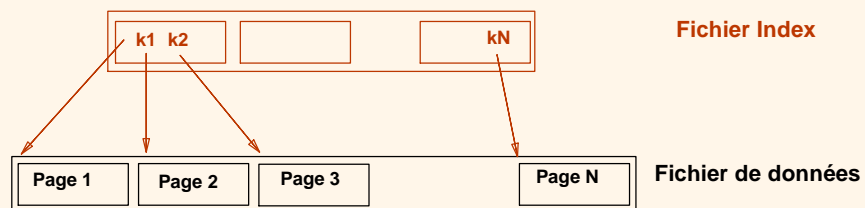
- *Accès aux données via des clés \mathbf{k} d'index.*
- *3 possibilités d'organisation des points d'accès aux données \mathbf{k}^* :*
 - \mathbf{k}^* = enregistrement de données avec la valeur clé \mathbf{k}
 - \mathbf{k}^* = $\langle \mathbf{k}, \text{rid d'enregistrement de clé } \mathbf{k} \rangle$
 - \mathbf{k}^* = $\langle \mathbf{k}, \text{liste de rid d'enregistrements de clé } \mathbf{k} \rangle$
- *Le choix est orthogonal à la technique d'indexation utilisée pour localiser les données de clé \mathbf{k} .*
- *Les index ayant une structure d'arbre supportent les deux types de recherche : **intervalle de valeurs** ou **égalité**.*
- ***ISAM (Indexed sequential access method)**: fichier trié et indexé sur la clé primaire, structure statique à la manière d'un carnet d'adresses.*
- ***Arbres B+**: dynamique, ajustement de la structure balancée de l'index à chaque insertion/suppression.*

Recherche sur un intervalle de valeurs

☞ ``Trouver tous les élèves dont l'âge < 20``

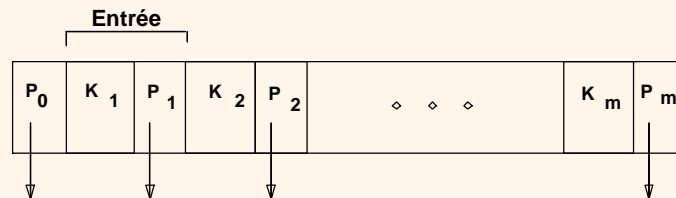
- Données dans un fichier trié : recherche séquentielle du premier élève, puis, lecture des suivants.
- Coût de la recherche séquentielle peut être très élevé.

☞ Idée : Créer un fichier index.

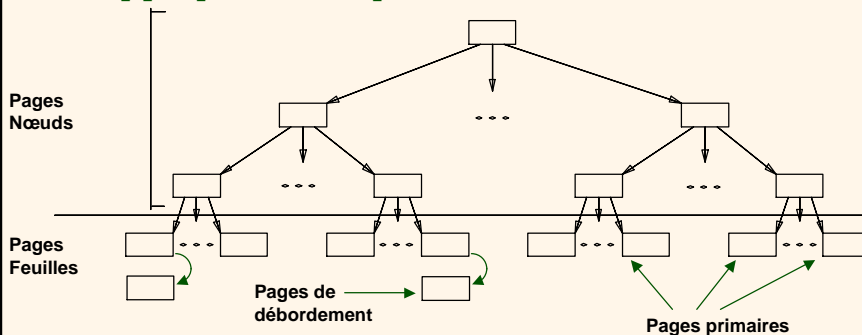


☞ Recherche séquentielle sur fichier index (plus petit)!

ISAM



☞ Le fichier index peut être volumineux. On peut appliquer l'idée plusieurs fois !



☞ Pages feuilles contiennent les points d'accès aux données

Commentaires sur ISAM

- *Création de fichier*: pages feuilles (données) sont allouées séquentiellement, triées sur la clé; puis les pages index sont allouées, ainsi que l'espace pour les pages de débordement.
 - *Entrées de l'index*: <clé, valeur de la clé, Id page>; Renvoient vers les *points d'accès aux données*, dans les pages feuilles.
 - *Recherche*: commence à la racine; comparaison de clé pour arriver aux feuilles. Coût $\log_F(N)$; $F = O(\text{nb entrées/page index})$, $N = O(\text{nb pages feuilles})$
 - *Insertion*: rechercher la page feuille d'accès aux données, et insertion à l'endroit trouvé.
 - *Suppression*: Recherche et retrait à partir de la page feuille. Si une page de débordement est rendue vide, la supprimer.
- ✍ **Structure d'arbre statique :**
inserts/deletes n'affectent que les pages feuilles.

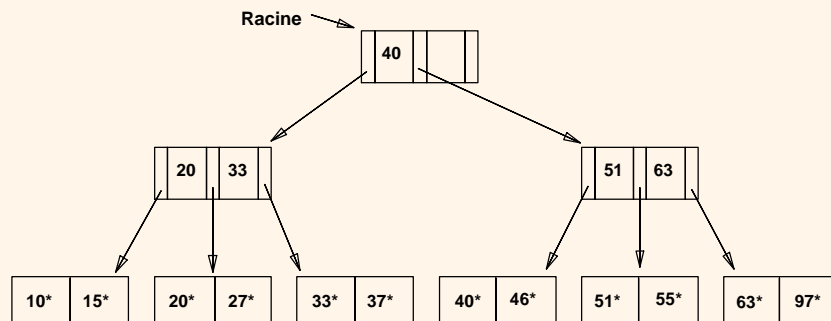
Pages de données

Pages d'Index

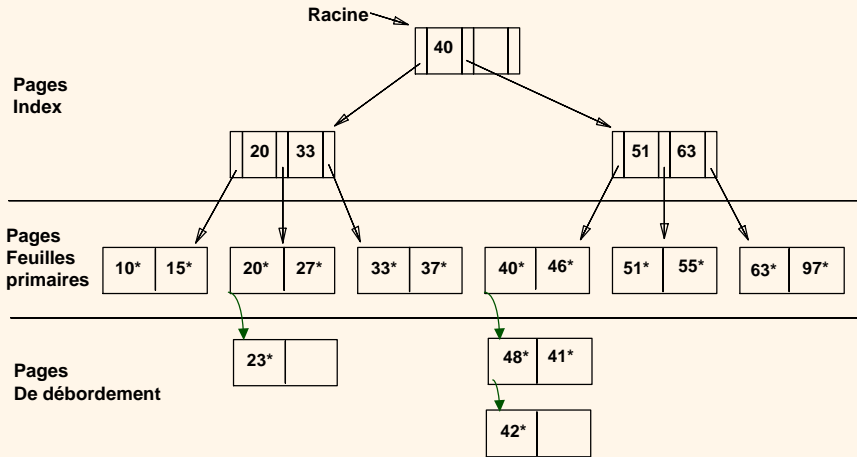
Pages de débordement

Exemple d'arbre ISAM

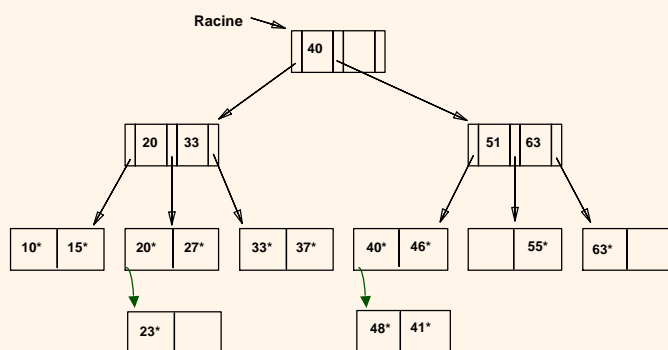
- ✍ Chaque nœud peut contenir deux entrées, pas de pointeur 'page suivante'.



Après Insertion de 23, 48*, 41*, 42* ...*



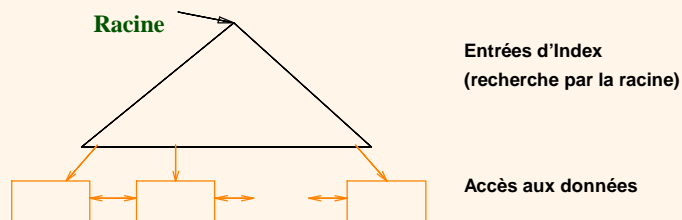
... Puis, Suppression de 42, 51*, 97**



51 est indexé dans les pages index, bien que supprimé des feuilles!*

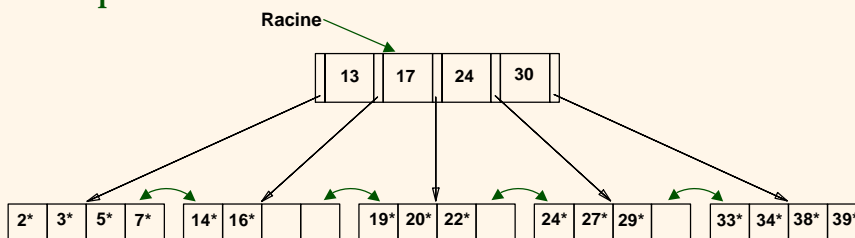
Arbre B+ : La structure d'index la plus répondue

- Inserts/deletes en $\log_F(N)$; garde une structure **balancée**.
(F = fanout, N = O(pages feuilles))
- Remplissage minimum 50% (excepté pour la racine).
Chaque nœud contient m entrées, $n \leq m \leq 2n$ entrées.
La valeur n est appelée *ordre* de l'arbre.
- Permet une recherche efficace par intervalle de valeur ou sur une condition d'égalité.



Exemple d'arbre B+

- Recherche démarre à la racine, comparaison de clé pour arriver aux feuilles (idem que ISAM).
- Recherche des points d'accès 5^* , 15^* , et tous points d'accès $\geq 24^*$...



15 n'est pas dans l'arbre!*

Arbre B+ dans la pratique

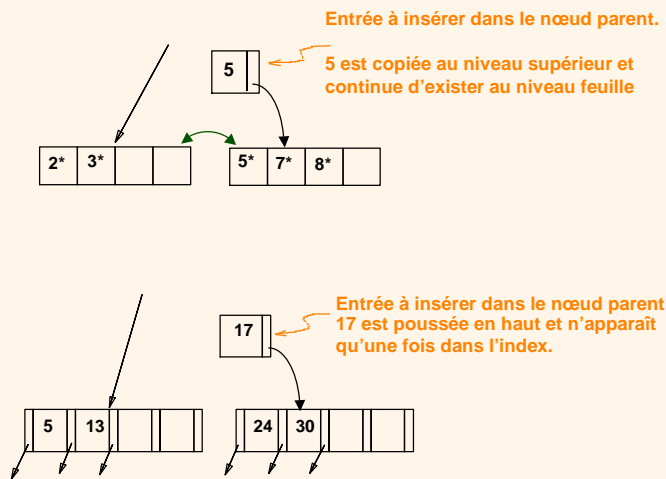
- Typiquement,
 - ordre : 100
 - Facteur de remplissage : 67%
 - fanout en moyenne = 133
- Typiquement, capacité :
 - Hauteur 4 : $133^4 = 312\ 900\ 700$ enregistrements
 - Hauteur 3 : $133^3 = 2\ 352\ 637$ enregistrements
- Souvent les pages du haut de l'arbre sont gardées dans le buffer :
 - Niveau 1 = 1 page = 8 Ko
 - Niveau 2 = 133 pages = 1 Mo
 - Niveau 3 = 17 689 pages = 133 Mo

Insertion dans un arbre B+

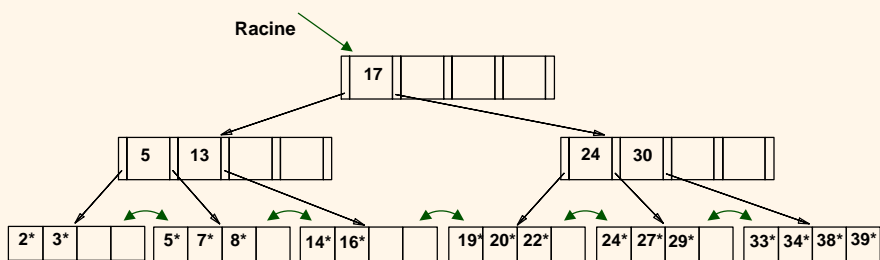
- Rechercher la bonne feuille L .
- Mettre le point d'accès aux données dans L .
 - Si L a suffisamment d'espace, *terminé* !
 - Sinon, *couper* L (en L et une nouvelle feuille $L2$)
 - Partager les points d'accès aux données entre les deux feuilles, déplacer **à partir** de la clé médiane dans $L2$.
 - Créer une entrée d'index au niveau supérieur pour pointer $L2$.
- Mécanisme récursif
 - **Pour couper un nœud index**, redistribuer les entrées d'index entre les deux nœuds, mais en **remontant** la clé médiane. (processus différent pour les nœuds feuilles).
- L'ajout de nœuds fait "grossir" l'arbre; le fait de couper le nœud racine en deux ajoute un niveau à la hauteur de l'arbre.
 - extension de l'arbre : devient *plus large* ou *gagne un niveau en haut*.

Insertion de 8* dans l'arbre exemple

- Observer comment le remplissage minimum est respecté dans les nœuds feuilles et ceux des niveaux supérieurs.
- La clé médiane est copiée au niveau supérieur *copy-up* lorsqu'il s'agit d'un nœud feuille. Elle est poussée au-dessus *push-up* lorsqu'il s'agit d'un nœud intermédiaire.



Arbre B+ après insertion de 8*

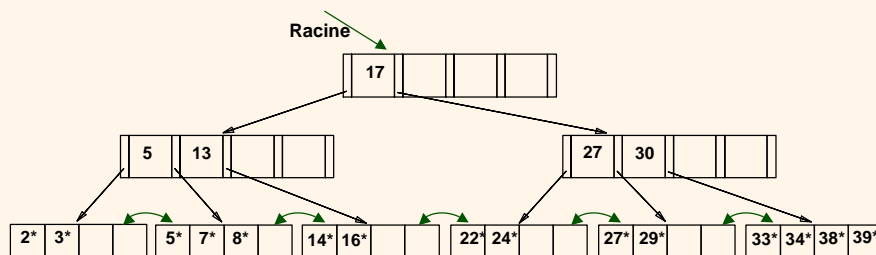


- ✂ La racine a été coupée en deux, l'arbre a gagné un niveau.
- ✂ Dans cet exemple, une réorganisation des clés aurait pu permettre d'éviter d'ajouter un niveau à l'arbre, mais cela ne se fait pas dans la pratique.

Suppression d'un point d'accès aux données d'un arbre B+

- A partir de la racine, rechercher la feuille L contenant le point d'accès.
- Supprimer le point d'accès.
 - Si L est au moins à moitié pleine, *terminé !*
 - Si L a $n-1$ éléments,
 - Essayer une **redistribution**, avec des feuilles ayant le même parent que L .
 - Si la redistribution ne donne rien, **fusionner** L avec un nœud voisin.
- Si la fusion est effectuée, il faut supprimer l'entrée qui pointe sur L , ou sur le nœud voisin qui a servi à la fusion, du parent de L .
- La fusion peut se propager jusqu'à la racine et faire baisser d'un niveau l'arbre.

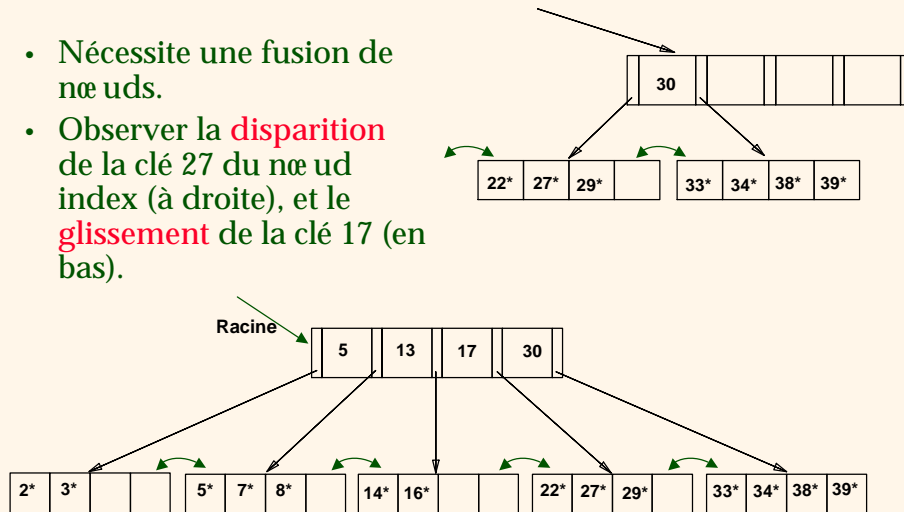
Arbre exemple après insertion de 8^* , puis, suppression de 19^* et de 20^* ...



- La suppression de 19^* est facile.
- La suppression de 20^* est faite avec une redistribution d'éléments avec une feuille voisine. Vérifier comment la clé médiane est copiée au-dessus..

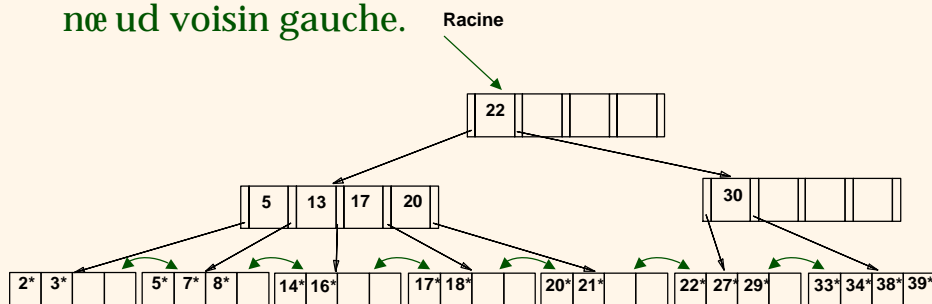
... Puis, après suppression de 24*

- Nécessite une fusion de nœuds.
- Observer la **disparition** de la clé 27 du nœud index (à droite), et le **glissement** de la clé 17 (en bas).



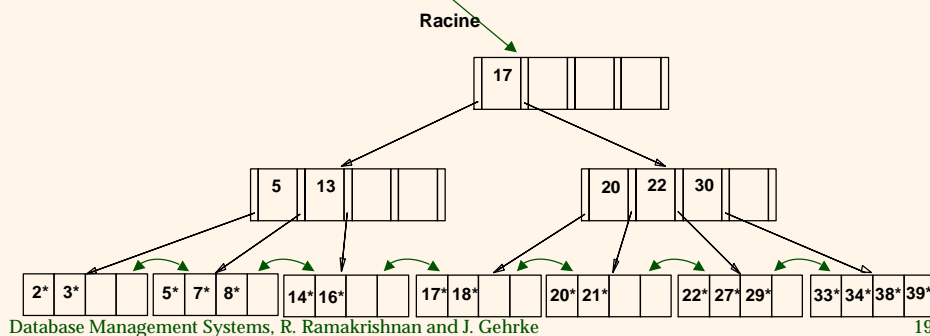
Exemple de redistribution au niveau des nœuds intermédiaires

- ☞ L'exemple donné ci-dessous peut correspondre à l'état obtenu en cours d'équilibrage à la suite de la suppression de 24* (que peut être l'arbre initial ?).
- ☞ Ici, il est possible de faire une redistribution avec le nœud voisin gauche.



Après redistribution

- Intuitivement, la redistribution des entrées est réalisée par un déplacement de gauche à droite, *en transitant* par le nœud parent.
- On aurait pu s'arrêter à la clé 20.



Vocabulaire

- Non-dense** : toutes les valeurs clés existantes ne sont pas représentées dans l'index. Lorsque les données sont triées sur l'attribut clé (index sur clé primaire), les seules clés représentées sont celles apparaissant au début de chaque bloc de données.
- Dense** : utilisé pour indexer des données non triées sur la clé de recherche. L'index est basé sur toutes les valeurs de clé qui existent dans les données en les associant à chaque enregistrement et non à chaque adresse de bloc.
- Index **primaire** / **secondaire** : index sur clé primaire (non dense) / index supplémentaire (dense) sur des attributs servant de critère de recherche.
- Lorsqu'il n'y a pas de garantie que les tuples partageant la même valeur clé soient dans un même bloc, l'index est dit **non-plaçant**.

Résumé

- La structure d'arbre est idéale pour une recherche par intervalle, comme pour une recherche par valeur.
- ISAM est une structure statique.
 - Seules les pages feuilles sont modifiées; des pages de débordement sont nécessaires.
 - Les pages de débordement peuvent dégrader la performance de l'index.
- Les Arbres B+ offrent une structure dynamique.
 - Inserts/deletes gardent l'équilibre de l'arbre; recherche en $\log_F(N)$.
 - Un fanout (**F**) élevé veut dire une hauteur d'arbre rarement supérieure à 3 ou 4 niveaux .

Résumé (suite)

- En général, un taux de remplissage moyen des nœuds de **67%**.
- Préféré à ISAM, sans considération des pbs de **verrouillage** nécessaires lors de l'équilibrage de l'arbre.
- Si les points d'accès aux données sont les enregistrements mêmes, le partage d'un nœud feuille peut provoquer le changement des rids !
- Les clés peuvent être compressées pour augmenter le fanout, et réduire la hauteur de l'arbre.
- Le chargement d'un volume de données important nécessite des mesures spécifiques (ex. tri des données à insérer) pour ne pas être freiné par la construction de l'arbre.
- Arbres B+ : la structure d'index la plus utilisée. Un composant essentiel des SGBD.