

RECHERCHE EXTERNE

Recherche dans mémoire externe

Supposons qu'on veut stocker un grand nombre de données : **enregistrements** avec **clés**

On veut minimiser l'accès au disque dur : 10^5 fois plus de temps que l'accès à la mémoire principale

Concept principal : **page** — bloc contigu de données (unité de lecture sur le disque)

Taille typique : 4k, 8k, ...

Accéder à une partie de la page est essentiellement aussi coûteux que d'accéder à toutes les données de la page

accès à une page : **probe**

Accès séquentiel indexé

«nœuds» du tableau d'indices : paire de (clef, adresse) où adresse est l'adresse d'une page

clés sont stockées de manière ordonnée, avec M paires en une page — correspond à un arbre M -aire

clés sont copiées à des nœuds internes

recherche : accès de $\log_M n$ pages mais insertions/suppression très compliquée

Arbre de recherche

Stocker un arbre rouge et noir??

Arbre 2-3-4 est plus efficace : on modifie «quelques» nœuds seulement

Comment améliorer?

on généralise à M sous-arbres au lieu de 4.

Une autre bonne idée : on va mettre les données aux feuilles, et ne stocker que des clés à des nœuds internes (« B^+ -arbre»)

Arbre B

Données stockées aux feuilles : L enregistrements à une feuille

Clés stockés aux nœuds internes : $(M - 1)$ clés à un nœud interne, M enfants

Clé i : valeur minimale dans le sous-arbre $(i + 1)$

Racine : $2..M$ enfants

Nœuds internes : $\lceil M/2 \rceil .. M$ enfants (taille : $M \cdot |\text{clé}| + (M - 1) \cdot |\text{pointeur}|$)

Feuilles : $\lceil L/2 \rceil .. L$ enregistrements (taille : $L \cdot |\text{enregistrement}|$)

Feuilles ont la même profondeur

Choix de M et L : on utilise une page par nœud

Arbre B

Thm. La hauteur h de B arbre est bornée par

$$h \leq 1 + \log_{\lceil M/2 \rceil} \frac{N}{2} \leq 1 + \frac{\lg \frac{n}{L}}{\lg M - 1}$$

où N est le nombre de feuilles et n est le nombre d'enregistrements.

Exemple : blocs de 8k, clés de 32 octets, enregistrements de 256 octets, $M = 228$,
 $L = 32$

$h = 4$ suffit jusqu'à $N = 2.9 \cdot 10^6$ ou $n = 47 \cdot 10^6$

\Rightarrow nombre d'accès au disque est déterminé par h : très peu (en plus, on peut garder la racine et peut-être même le premier niveau en mémoire principale)

Arbre B (cont)

Insertion d'un enregistrement : s'il y a de la place dans la feuille, aucun problème

s'il n'y a pas de place : **débordement** de la feuille

solution : découpage de la feuille \rightarrow éléments distribués en deux feuilles de tailles $\lfloor \frac{L}{2} \rfloor + 1$ et $\lceil \frac{L}{2} \rceil$.

peut causer un débordement au parent : découpage («éclatement» dans le livre) si nécessaire en ascendant vers la racine

\Rightarrow la hauteur croît en découplant la racine

Arbre B (cont)

Suppression d'un élément : si la feuille est toujours assez complète, aucun problème et si le nombre d'éléments tombe en-dessous de $\lceil L/2 \rceil$?

1. prendre des éléments des sœurs immédiates
 2. si elles sont au minimum, alors fusionner les feuilles \rightarrow le parent perd un enfant
 3. continuer avec le parent de la même manière si nombre d'enfants $< \lceil M/2 \rceil$
- \Rightarrow la hauteur décroît en enlevant la racine (quand elle a un enfant seulement)