# Kap.6 Datenstruktur für Peripheriespeicher
# Kap.6.0 Plattenspeicher

| | Großr. 1980 | MARS µVAX 1987 | Siemens 4868 1987 | PC 1995 | PC 2001 |
|---|---|---|---|---|---|
| Anz. Oberflächen | 20 | 12 | 60 | | 6 |
| Anz. Zylinder | 400 | 1024 | 555 | | |
| Kapazität/Block | 3KB | 1KB | | 4KB | 8KB |
| /Spur | 12KB | 8KB | 19KB | | |
| /Zylinder | 240KB | 88KB | 1,2MB | | |
| gesamt | 96MB | 90MB | 635MB | 4GB | 120GB |

1

| | | | | | |
|---|---|---|---|---|---|
| Umdrehungszeit | 20 ms | 16 | 17 | | 8,5 |
| mittl. Rotationz. | 10 ms | 8 | 8,4 | | |
| Übertr. Rate | 600 KB/ s | 500 KB/ s | 1,2 MB/ s | 3 MB/s | 23 -48 MB/s |
| Suchz. Nächster Zgl. | 10 ms | 10 | 6 | | |
| mittl. Suchzeit | 30 ms | 25 | 20 | | |
| max. Suchzeit | 55 ms | 40 | 40 | | |
| mittl. Suchz. gesamt | 40 ms | 33 | 28,4 | 10ms | 8,5ms |
| Zugänglichkeit $\frac{MB}{MB/sec}$ | 160 sec | 180 | 552 | 1333 | 2400 |
| Zugr. HS | 500 ns | 300 ns | 120 ns | | 20ns |
| Zeit Platte/ HS | $0,8 \cdot 10^5$ | $1,1 \cdot 10^5$ | $2,4 \cdot 10^5$ | | $4,2 \cdot 10^5$ |
| HS Größe | 100 KB | 256 KB | 32 MB | 16 MB | 256 MB |
| Periph. Größe | 100 MB | 40 MB | 9 GB | | 80 GB |
| Quotient P/ HS | 1000 | 150 | 280 | 250 | 320 |

2

**Fazit:**

1. Zugr. Zeit relativ zu HS-Zeit sehr groß, ≈ $10^5$
2. Heutige Mikro-Rechner haben Peripherie wie Groß-rechner vor einigen Jahren ⇒ Downsizing, Client/ Server
3. I/ O Engpaß wird immer schlimmer, Zugänglichkeit!!
   z. B. nächste Generation von CD
   ≈ 8 $GB/_{CD}$ • 100 $CD/_{Jukebox}$ = 800 $GB/_{Jukebox}$
   in Größe eines Mikrowellen-Ofens

   **Heilmittel:** - RAID s
   - Striping

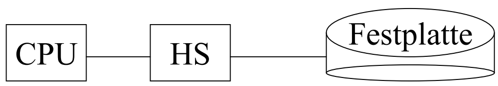
4. HS werden schneller schnell als Peripheriespeicher



5. Verhältnis Plattenkap./ HS-Kap. Bleibt bei ca. 250
6. Weitere Ebene in Peripherie-Speicher-Hierarchie:
   Jukeboxen und Netze (CAN u. WAN)

⇒HS-Datenbanken wird es auf absehbare Zeit **nicht** geben!!
Datenstr. Für Peripheriespeicher nehmen an Bedeutung zu.

**Ein Analogon:** Faktor $10^5$




| | | |
|---|---|---|
| 4 B | 4 KB | $10^3$ mehr |
| 200 ns | 20 ms | $10^5$ langsamer |
| 1 kg | 1 t | |
| 600 000 km/h | 6 km/ h | |


**Ziel:** möglichst wenig Platenzugriffe

- Algorithmen u. DS
- Pufferungstechniken

**Faustregel:** von untergeordneter Bedeutung

- Rechenaufwand, CPU
- PS Platz

5

**Beispiel:** AVL-Baum auf Platte u. 1 Plattenzugriff pro AVL Knoten:

$10^6$ Knoten ~ Höhe 30

1 Suchvorg. ~ 30 Plattenzugriffe

= 30 • 30 ms = 0,9 sec

Aufbau aus leerem Baum durch einzelne **insert** ohne Pufferung ca. 10 Tage.

6

## Kap. 6.1 B-Bäume

Bayer, Mc Creight, Herbst 1969

**Problemstellung:** Wie bei AVL-Bäumen:
Verwaltung einer Menge $\Sigma = \{(x, \alpha)\}$
$x \in X$, Schlüsselmenge
$\alpha$ : assoziierte Info

**Operationen:** **insert** $(x, \alpha)$
**search** $(x)$
**delete** $(x)$

zusätzlich sequentielle Verarbeitung
nach $<$ oder $\leq$ auf X:

**reset**
**next**
**eof**

} scan, cursor über $\Sigma$

$<$ ist meist lexikographische Ordnung.

7

**Beispiele:**

1. Telefonbuch: $\{(N,T)\}$
   Vor- u. Nachteile von $<$ ?
2. Dateienkatalog
3. Kundendatei: $<$ ?
4. Freie Speichergebiete
5. Lagerhaltung mit Teile #
6. Zeitreihen von Aktienkursen, Meßwerten mit mehrdimensionalem X

8

**Problem:** $\Sigma = \{(x, \alpha)\}$ ist meist sehr große, dyn. Menge von **Variablen**, aber zur Übersetzungszeit unbekannt, Deklaration und Manipulation interaktiv.

**Manuelle Lösung:** Sortierte Kartei mit Indexkarten

**HS-Lösung:** AVL-Baum

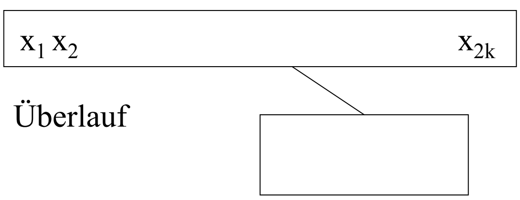


**Grundidee:** Transporteinheit = 1 Plattenseite = 1 Baumknoten

**insert** in leeren Baum

| | | | |
|---|---|---|---|
| $x_1, \alpha_1$ | $x_2, \alpha_2$ | . . . | $x_{2k}\ \alpha_{2k},$ |

$(x, \alpha)$ einsortieren in $<$ ,
Maximal 2k Einträge.
Ab jetzt ignoriere $\alpha_i$
durch insert des nächsten Elementes:

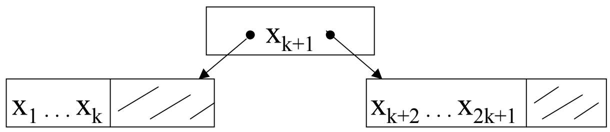



$X_{2k+1}$ ?

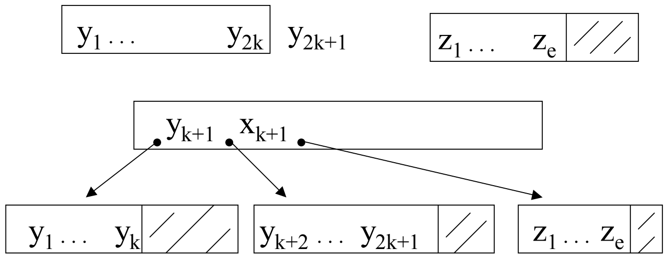
Überlauf

Scientific American!

**Spaltung:**




u. rekursive Fortsetzung bei Spaltung eines weiteren Knotens






**Hinw:** Höhenwachstum nicht durch „Austreiben“ neuer Blätter nach unten, sondern durch Überlauf u. Spaltung der Wurzel.

- Baumstruktur?
- Knotenstruktur?

**Def:** von ***B-Bäumen***: Seien h,k ganz, $h \geq 0$, $k > 0$. Ein ***B-Baum T*** der Klasse $\tau$ (k, h) ist entweder

a) $\emptyset$, d.h. $h = 0$ oder
b) Ein geordneter Baum, in dem
   i) jeder Pfad von Wurzel zu einem Blatt hat Länge $h - 1$
   ii) Wurzel ist selbst ein Blatt oder hat $\geq 2$ Söhne; andere interne Knoten haben $\geq k + 1$ Söhne
   iii) jeder Knoten hat $\leq 2\,k+1$ Söhne

**Lemma:** Für $N \neq 2 \wedge N \geq 0 \wedge N \neq 2k+3 \wedge N \neq 2k+4$ gibt es

B-Baum mit N Knoten

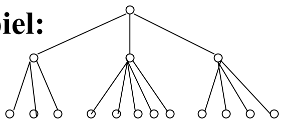
In Klasse $\tau$ (k, h) für geeignetes h u. jedes k:

$\forall N \geq 0 \wedge N \neq 2 \wedge N \neq 2k+3 \wedge N \neq 2k+4 \; \forall k \; \exists h \; \exists T \in \tau(k, h)$:

Anzahl Knoten (T) = N

**Bew:** siehe spätere Konstruktionen.

**Beispiel:**



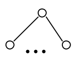
$\in \tau$ (2, 3)
{Verzweigungsgrad
zwischen 3 u. 5,
Höhe 3}

**Übung:** Konstruiere kleinsten u. größten Baum in $\tau$ (2, 3) u. zähle Knoten.



**Bew**. für Lemma auf Folie 13

1. Für N = 2k + 3 gibt es keinen Baum, weil mit
   h = 2



Voller Baum hat
2k + 2 Knoten

Wurzelspaltung liefert Baum mit 2k + 5 Knoten,
2k + 3, 2k + 4 fehlen.

h=1: + 1
h=2: 1 + 1
h=3: 2k+1 + 1

**Ansatz:** wenn gilt: $N_{min}\ (k, h+1) \leq N_{max}\ (k, h)$, dann kann man zu $N_{min}$ einen zusätzlichen Knoten durch Spaltung hinzunehmen.

**Induktionsbeweis:** Basis $h = 3$ :

$N_{min}\ (k, \underbrace{h+1}_{4}) \leq N_{max}\ (k, h)$ weil

$$1 + \tfrac{2}{k}\ ((k+1)^3 - 1) \leq \tfrac{1}{2k}\ ((2k+1)^3 - 1)$$

mit $(a+b)^3 = a^3 + 3a^2 b + 3a\, b^2 + b^3$ :

$$1 + \tfrac{2}{k}\ (k^3 + 3k^2 + 3k + 1 - 1) \leq$$

$$\tfrac{1}{2k}\ (8k^3 + 3 \cdot 4k^2 + 3 \cdot 2k + 1 - 1)$$

$$1 + 2\ (k^2 + 3k + 3) \leq \tfrac{1}{2}\ (8k^2 + 12k + 6) = 4k^2 + 6k + 3$$

15

$$1 + 2k^2 + \cancel{6k} + \leq 4k^2 + \cancel{6k +} 3$$

$$4 \leq 2k^2 \quad \text{mit } k \geq 2$$

$$4 \leq 8 \quad \text{stimmt für Induktionsbasis}$$

Ungleichung gilt nicht für $h = 2$ :

$$1 + \tfrac{2}{k}\ ((k + 1)^2 - 1) \underset{?}{\leq} \tfrac{1}{2k}\ ((2k + 1)^2 - 1)$$

$$1 + \tfrac{2}{k}\ (k^2 + 2k + \cancel{1} - \cancel{1}) \underset{?}{\leq} \tfrac{1}{2k}\ (4k^2 + 4k + \cancel{1} - \cancel{1})$$

$$1 + 2\ (k + 2) \leq \tfrac{1}{2}\ (4k + 4)$$

$$1 + 2k + 2 \not\leq 2k + 2$$

16

**Induktionsschritt:** von $h \geq 3 \Rightarrow h + 1$

$$1 + \tfrac{2}{k}\,((k+1)^h - 1) \leq \tfrac{1}{2k}\,((2k+1)^h - 1) \;/\cdot 2k$$

$$2k + 4\,((k+1)^h - 1) \leq (2k+1)^h - 1$$

$$\underbrace{\underbrace{2k - 3}_{\substack{\text{bleibt,}\\ \text{Ist} > 0}} + 4\,\underbrace{(k+1)^h}_{\substack{\text{wird m.}\\ k+1\\ \text{multipl.}}}}_{\substack{\text{d.h. Teil der linken}\\ \text{Seite mit kleinerem}\\ \text{Faktor } (k+1)\\ \text{multipliziert}}} \leq \underbrace{\underbrace{(2k+1)^h}_{\substack{\text{wird m.}\\ 2k+1\\ \text{multipl.}}}}_{\substack{\text{ganze rechte}\\ \text{Seite mit größerem}\\ \text{Faktor } (2k+1)\\ \text{multipliziert}}}$$

stimmt nach Ind. Vorauss.

Bei Überg. von h zu h+1

$\Rightarrow$ Ungleichung bleibt auch für h+1 erhalten q.e.d.

17

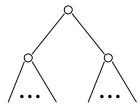
**Lemma:** Der minimale, maximale Baum in $\tau$ (k, h) habe $N_{min}$( k, h), $N_{max}$ (k, h) Knoten.

Dann gilt: $N_{min}^{(k,h)} = 1 + \frac{2}{k}\,((k+1)^{h-1} - 1)$

$$N_{max}^{(k,h)} = \frac{1}{2k}\,((2k+1)^h - 1)$$

18

**Bew:** $N_{min}$: Wurzel hat 2 Söhne,
sonst k+1 Söhne



$$\left.\begin{array}{l} 1 \\ 2 \\ 2 \cdot (k+1) \\ 2 \cdot (k+1)^2 \end{array}\right\} \quad \sum = 1 + 2 \cdot ((k+1)^0 + (k+1)^1 + \ldots + (k+1)^{h-2}) = 1 + \tfrac{2}{k}\,((k+1)^{h-1} - 1)$$

$\vdots$

$$N_{max} = 1 + (2k+1) + (2k+1)^2 + \ldots + (2k+1)^{h-1} = \tfrac{1}{2k}\,((2k+1)^h - 1)$$

**Hinw:** Baum T mit N (T) Knoten kann aus vielen Klassen sein mit unterschiedlichen k und h.
Konstruiere ein $T \in \tau(2, 3) \cap \tau(3, 3)$ sowie
$T` \in \tau(2, 3)$ und $T`` \in \tau(2, 4)$ mit N (T`) = N (T``)

19

**Def: Knotenaufbau:**
1 B-Baum Knoten  1 Plattenseite

Indexelement: $(x, \alpha)$

i) Wurzelseite hat 1 bis 2k Indexelemente, andere Seiten haben k bis 2k Indexelemente

ii) Seite P mit $\ell$ Indexelementen (Schlüsseln) hat $\ell + 1$ Söhne, außer wenn P Blatt ist

iii) $x_1, x_2, \ldots, x_\ell$ sind auf P geordnet nach < auf X. Falls P kein Blatt ist, sind auf P $\ell + 1$ Zeiger $p_0, p_1, \ldots, p_\ell$ auf Söhne von P

iv) Sei $P(p_i)$ Seite, auf die $p_i$ zeigt, $T(p_i)$ Unterbaum von T mit Wurzel $P(p_i)$ $K(p_i)$ Menge der Schlüssel (keys) in $T(p_i)$, dann gilt:

20

$\forall\, y \in K(p_0)\ :\ y < x_1$

$\forall\, y \in K(p_i)\ :\ x_i < y < x_{i+1}$ für $i = 1, ..., \ell - 1$

$\forall\, y \in K(p_\ell)\ :\ x_\ell < y$

**Seitenorganisation:**

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $p_0$ | $x_1$ | $\alpha_1$ | $p_1$ | $x_2$ | $\alpha_2$ | $p_2$ | ... | $x_\ell$ | $\alpha_\ell$ | $p_\ell$ | |

$\forall\, y \in K(p_1) :\ x_1 < y < x_2$

**Durchlaufalg:** ~ Nachordnung für lexikogr. Schlüsselordnung $T(p)$, Wurzel $P(p)$ mit $\ell(p)$ Schlüsseln:

21

**if** $T(p) \neq \emptyset$ **then**

```
begin durchlaufe T (p0);
    for i:= 1 to ℓ (p) do
        begin verarbeite (xi, αi);
              durchlaufe T (pi)
        end
end
```

**Hinw:** Rekursion braucht Keller von Seiten entsprechend Pfad durch B-Baum, i.e. maximal h Seiten.

22

**Suchalgorithmus:** finde beliebiges y in T (p) :

Parameter: y, p:

**if** T(p) = $\varnothing$ **then** Mißerfolg

**else if** $y < x_1$ **then** suche y in T ($p_0$)

**else if** $y = x_i$ **then** {y gef.} verarbeite ($x_i$, $\alpha_i$)

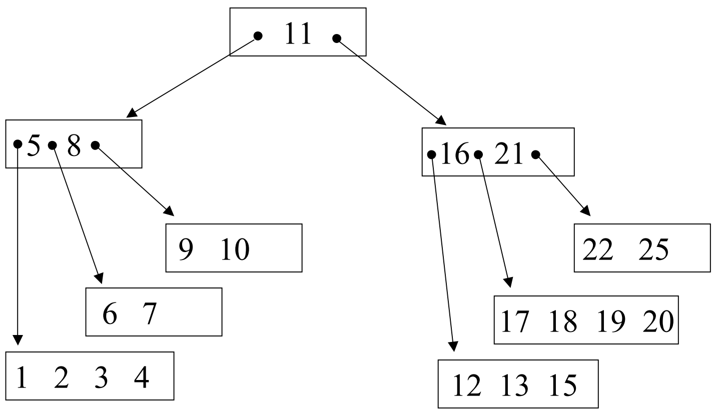
**else if** $x_i < y < x_{i+1}$ **then** suche y in T ($p_i$)

else {$x_\ell < y$} suche y in T($p_\ell$)

**Hinw:** Die Fälle $y < x_1$ , $y = x_i$ , $x_i < y < x_{i+1}$ erfordern Suche innerhalb der Seite P(p). Seiten so organisieren, daß binäre Suche in P(p) möglich ist.

23

**Beispiel:**




24

**Anzahl Indexelemente in T:**

sei $T \in \tau(k, h)$ :

$I_{min}(k, h), I_{max}(k, h)$ ?

$N_{min}(k, h) = 1 + \frac{2}{k}((k + 1)^{h-1} - 1)$

$\Rightarrow I_{min}(k, h) = 2(k + 1)^{h-1} - 1$

$N_{max}(k, h) = \frac{1}{2k}((2k + 1)^{h} - 1)$

$\Rightarrow I_{max}(k, h) = (2k + 1)^{h} - 1$

sei I Anzahl Indexelemente in T

$I_{min} \leq I \leq I_{max}$

$\Rightarrow \frac{I+1}{2} \geq (k + 1)^{h-1}$

$I + 1 \leq (2k + 1)^{h}$

25

**Höhenungleichung:**

$\Rightarrow$ **Logarithmisches Höhenwachstum:**

$\log_{2k+1}(I+1) \leq h \leq 1 + \log_{k+1}(\frac{I+1}{2})$

**Beispiel:** Sei h = 4, k = 200

d.h. zu Blatt von T 2-3 Plattenzugriffe mit Caching

$I \geq 2 \cdot 200^{3} = 1.6 \cdot 10^{7}$

$I \leq 400^{4} = 256 \cdot 10^{8} = 2.6 \cdot 10^{10}$

bei 20 Bytes/ Element:

$320 \text{ MB} = 3.2 \cdot 10^{8} \text{ B} \leq I \leq 5.2 \cdot 10^{11} = 520 \text{ GB}$

$\Rightarrow$ B-Baum kann um Faktor 1000 wachsen, ohne wesentliche Änderung des Zugriffsverhaltens.

$\Rightarrow$ Bedeutung für DBen

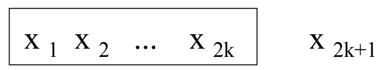
26

**Einfüge-Algorithmus:** $T \in \tau(k, h)$

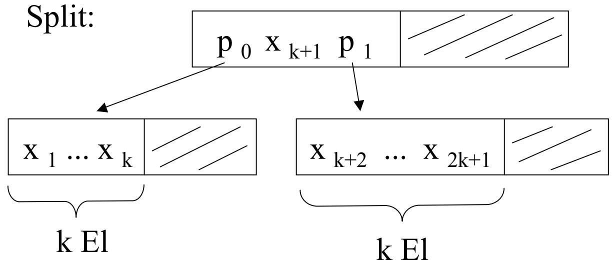
Annahme: $x_i$, $p_i$, $\alpha_i$ feste Länge

Start : $T = \varnothing$

Aufbau der Wurzel:

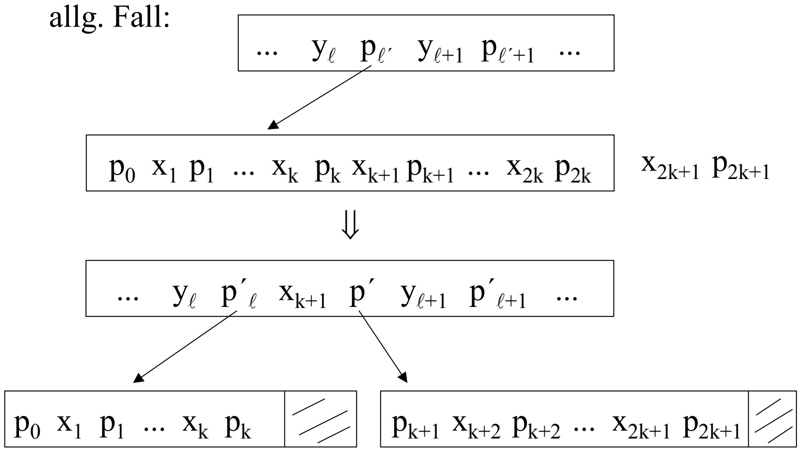



Split:

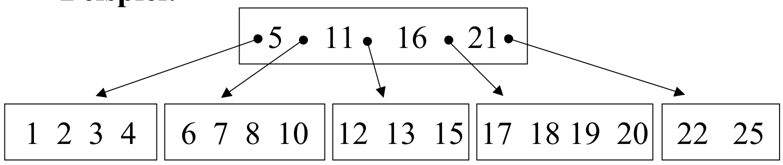





allg. Fall:



**Beispiel:**




durch Eintrag von 9 entsteht Baum auf Folie 24

**Übung:** Wähle Zahlenfolge u. konstruiere
$T \in \tau(2, h)$ durch Einfügung in $\emptyset$.

**Übung:** Andere Reihenfolge der Einfügung derselben
Zahlen führt i. a. zu anderem Baum. Beispiel?

29

**Kostenanalyse Einfügung:**

$f_{min}$ = minimale Anzahl zu *holender* Seiten

$w_{min}$ = minimale Anzahl zu *schreibender* Seiten

$f_{max}$, $w_{max}$ analog

- Suchvorgang: $f_{min} = 1$
  $f_{max} = h$
  $w_{min} = w_{max} = 0$

- Einfügen: $f_{min} = h, f_{max} = h$
  $w_{min} = 1, w_{max} = 2h + 1$

30

Fall $w_{max}$ zeigt **Höhenwachstum**:

spalte Pfad einschl. Wurzel!

tritt sehr selten auf, für $T \in \tau\ (k, h)$ nur h-1 mal

**Amortisierte, durchschnittliche Aufbaukosten:**

**Annahme:** in T wird nur eingefügt u. gesucht, praxisnahe!

Seien I Indexel. in T

n(I) : Anzahl der Knoten von T

$n(I) \leq \frac{I-1}{k} + 1$

beim Aufbau von T ergeben sich

$s(I) \leq \frac{I-1}{k}$ Spaltungen

31

**Bew:** erste Seite pro Ebene entsteht durch Höhenzuwachs,
alle anderen Seiten entstehen durch Splits

Anzahl Splits = n(I)-h =: s(I)

$n(I) \leq \frac{I-1}{k} + 1$

$s(I) \leq \frac{I-1}{k} + 1 - h \leq \frac{I-1}{k}$

32

Beim Aufbau von T ergeben sich
2 • s (I) *writes* wegen Spaltungen

$w = I + 2 \cdot s(I)$ *writes* insgesamt

mittlere Anzahl writes pro **insert**

$w_a = \frac{w}{I} = 1 + 2 \cdot \frac{s(I)}{I} < 1 + \frac{2}{k}$

heute : $k \geq 100$

$\Rightarrow$ weniger als 1.02 writes/ insert

**Kostendurchschnitt** pro Einfügung:

$f_a = h$ ; $w_a < 1 + \frac{2}{k}$ ~ 50ms/insert

$10^6$ inserts ~ 50.000 sec ~ 1 Tag
10 GB ~ $10^8$ inserts ~ 100 Tage

33

**Löschalgorithmus:** delete (x):

**Schritt 1**: search (x)

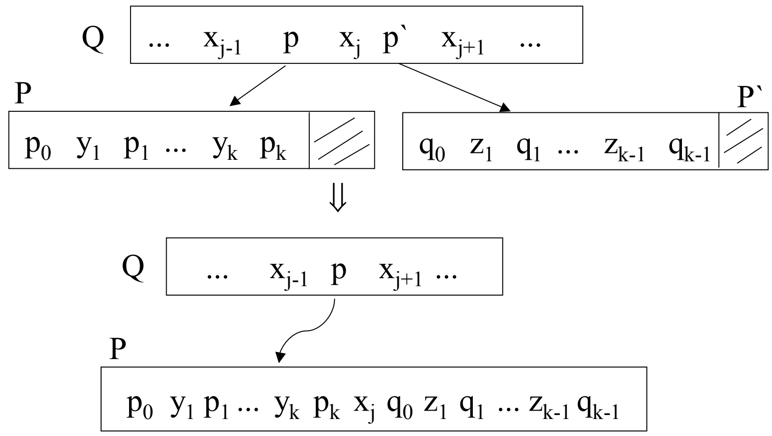
**Schritt 2**: eigentliches delete

2a) x steht auf Blatt

2a.1) noch mindestens k Elemente auf Blatt nach delete

2a.2) Konkatenation: k-1 El. auf Blatt
k El. auf einer Nachbarseite

34




2k Einträge. Rekursive Fortpflanzung bis zur Wurzel

35

2a.3) Unterlauf:

k - 1 El. auf P`, $\ell > k$ auf P
konkateniere P` mit P u. spalte
keine Fortpflanzung der Spaltung zur Wurzel hin
Verschiebung P $\rightarrow$ P` mit Anpassung
von Trennschlüssel in Vaterknoten.

36

2b) x steht auf internem Knoten:

Ersetze x durch nächstgrößeres
(nächstkleineres) El. y {y auf Blatt}
Jetzt Löschung von Blatt wie 2a mit Unterfällen.
(Siehe auch Löschung von Zwischenknoten
in AVL-Bäumen)

37

**Kosten des Löschalg:**

1. $f_{min} = h$ ; $w_{min} = 1$
2. lösche x von internem Knoten
   $f = h$ ; $w = 2$
3. $f_{max} = 2h - 1$
   $w_{max} = h + 1$

   Pfadkonkatenation plus Unterlauf
   in Wurzelnachfolger

38

## Speichernutzung

ungünstigster Fall 50 % Nutzung,
z.B bei Einfügung in Sortierordnung

**Verbesserung:** Verzögerung von Splits durch Überlauf: i.e. Konkatenation mit Nachbarn u. Split.

$$\underbrace{100\% \quad 50\% \quad 50\%}_{2/3} \quad \underbrace{100\% \quad 50\% \quad 50\%}_{2/3} \quad 100\%$$

= 66% Mindestausnutzung, durchschnittlich ca. 83%

**allg. Fall:** Split nur, wenn n Nachbarseiten voll sind

39

## Optimale Seitengröße

Parameter k ?

**Aufwand pro Seite** * h

$\alpha$: Zugriffszeit, unabhängig von k
$\beta$: Übertragungszeit für Tupel (x $\alpha$ p)
$\gamma$: Konstante für binäres Suchen in Seite
$\nu$: Besetzungsfaktor für Seite: $1 \Leftrightarrow \nu \Leftrightarrow 2$

Kosten pro Seite : $\alpha + \beta\,(2k + 1) + \gamma \ln (\nu k + 1)$

$\delta h$: Seitentransporte pro Operation

Zeit $t \approx \delta h\,(\alpha + \beta\,(2k + 1) + \gamma \ln (\nu k + 1))$

40

approximiere: $h \approx \log_{\nu k+1}$ (I) (Höhenungleichung S.26)

$$t \approx t_a = \delta \log_{\nu k+1} (I) (\alpha + \beta (2k + 1) + \gamma \ln (\nu k + 1))$$

mit $\log_{\nu k+1} (I) = \frac{\ln (I)}{\ln (\nu k + 1)}$ :

$$t_a = \delta \ln (I) \left[ \frac{\alpha + \beta (2k + 1)}{\ln (\nu k + 1)} + \gamma \right]$$

$$t_a` = \delta \ln (I) \left[ \frac{\ln (\nu k + 1) \cdot 2\beta - (\alpha + \beta (2k+1)) \frac{\nu}{\nu k + 1}}{\ln^2 (\nu k + 1)} \right]$$

41

$t_a` = 0$ liefert:

$$\ln (\nu k + 1) \cdot 2\beta - (\alpha + \beta ( 2k + 1)) \frac{\nu}{\nu k + 1} = 0$$

$$2 \frac{\nu k + 1}{\nu} \ln (\nu k + 1) = \frac{\alpha}{\beta} + (2k + 1) \qquad /: \beta$$

$$\frac{\alpha}{\beta} = 2 \frac{\nu k + 1}{\nu} \ln (\nu k + 1) - (2k + 1) = : f (k, \nu)$$

wähle k so, daß $f (k, \nu) \approx \frac{\alpha}{\beta}$

mit $\nu : 1 \Leftrightarrow \nu \Leftrightarrow 2$

42

***Wichtig: optimale Seitengröße***
***unabhängig von I, γ, δ***

f hat flachen Verlauf, d.h. Berücksichtigung
von Kenngrößen der Platte, z.B. Spurkapazität o. k.

**Konkrete Blockgrößen:**

**Fall 1:** (x, α, p) = 45 Bytes

$\beta = \frac{45\ \text{B} \cdot \text{sec}}{3\ \text{MB}} = 15\ \mu\text{s};\ \alpha = 20\ \text{ms}$

$\frac{\alpha}{\beta} = \frac{20\ \text{ms}}{15\ \mu\text{s}} = 1.3 \cdot 10^3;\ k \approx 150$

Seitengröße = 13.5 KB



**Fall 2:** (x, α, p) = 300 Bytes

$\beta = \frac{300\ \text{B} \cdot \text{s}}{3\ \text{MB}} = 10^{-4}\ \text{s};\ \frac{\alpha}{\beta} = 200$

$k \approx 32$ Seitengröße 9600 Bytes

**Fall 3:** Vergleich Festplatte mit CD-ROM

| | Festplatte | CD-ROM |
|---|---|---|
| | 20 ms | 200 ms |
| | 3 MB/ s | 300 KB/ s |

$\frac{\alpha}{\beta}$ bleibt, gleiche Blockgröße
glücklicher Zufall ??

**Fall 4:** Vergleich Festplatte mit Juke-Box

| | Festplatte | Juke-Box |
|---|---|---|
| | 20 ms | 10s |
| | 3 MB/ s | 300 kB/ s |

$\frac{\alpha}{\beta}$ wird größer

$\frac{\alpha}{\beta} = \frac{10s}{150\,\mu s} = 6.6 \cdot 10^4$ ; $k \approx 4000$

Blockgröße: 8000 · 45 Bytes = 360 KB



**Baumhöhe u. Indexelemente:**

$2\,(k+1)^{h-1} - 1 \Leftrightarrow \mathbf{I} \Leftrightarrow (2k+1)^{h} - 1$

sei k = 200

| h | $I_{min}$ | ≤ | I | ≤ | $I_{max}$ |
|---|---|---|---|---|---|
| 1 | 1 | | | | 400 |
| 2 | 400 | | | | 160 000 |
| 3 | $8 \cdot 10^4$ | | | | $400^3 = 64 \cdot 10^6$ |
| 4 | $2 \cdot 200^3 = 1.6 \cdot 10^7$ | | | | $400^4 = 2.56 \cdot 10^{10}$ |

Anm: $64 \cdot 10^6 \cdot 20$ Bytes ≈ 1 GB

⇒ bei Workstations heute ca. 2 Plattenzugriffe

**Kfz-Datei Flensburg:** $< 50 \cdot 10^6$ Fahrer

$= 5 \cdot 10^7$ Einträge, d.h. $h = 4$,
mit großem Cache 2 Zugriffe

47