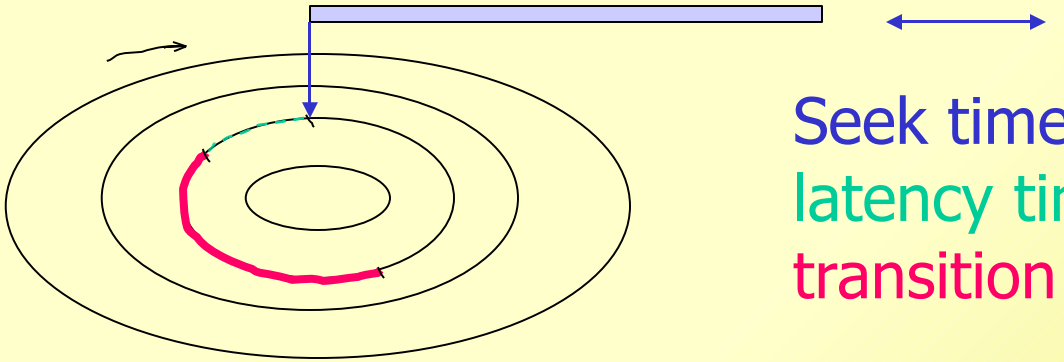
Fyzická organizácia dát

- Organizácia pamätí (vnútorná a vonkajšie pamäte)
  - rotačné, diskové a bubnové pamäte
  - sekvenčné (páskové) pamäte
  - alternatívne pamäte
- Čas prístupu
  - seek time (čas vyhľadania cylindra)
  - latency time (čas pretočenia na začiatok na stope)
  - transmission time (čas prenosu dát)
- Formát blokov a záznamov
  - pevná alebo premenná dĺžka
  - pripichnuté (pinned), nepripichnuté alebo polpripichnuté záznamy

Disková pamäť



Seek time
latency time
transition time

- Čas prístupu
  - seek time (čas vyhľadania cylindra) *cca* *10 ms*
  - latency time(čas pretočenia na začiatok na stope) *0.1 ms*
  - transmission time (čas prenosu dát) *cca* $5.10^{-7} \times b\ s$

Sekvenčné média (pásky)

Rewind, locate, read (forward, backward)

Organizácia súborov

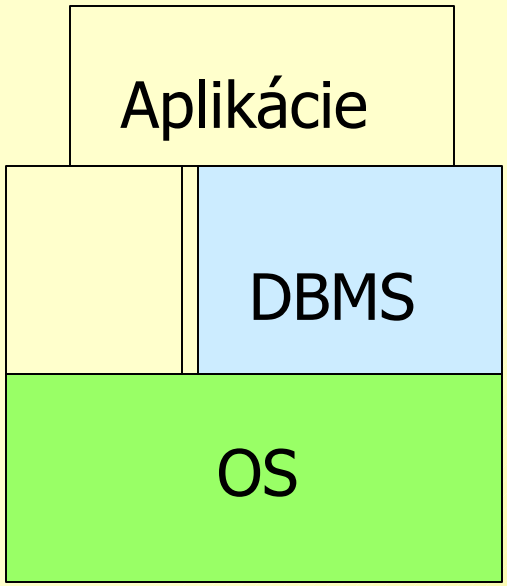
- Operačný systém
- Databázový systém

Možné architektúry

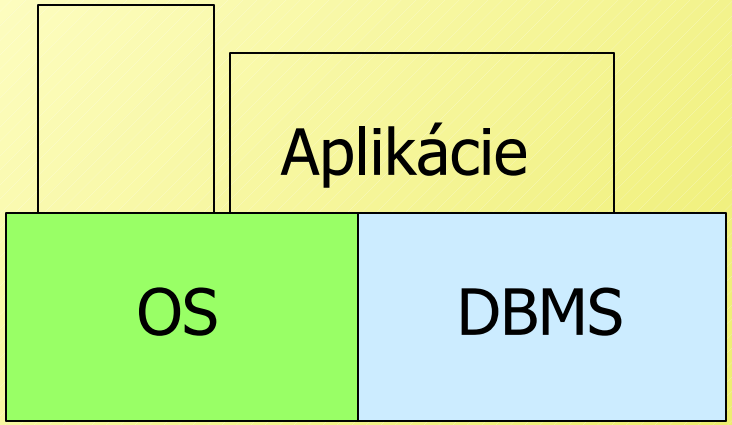
Vyrovnávacie pamäte (buffers)

Odsun blokov:

- v poradí
- ktorý je v pamäti najdlhšie
- s ktorým sa najdlhšie nepracovalo



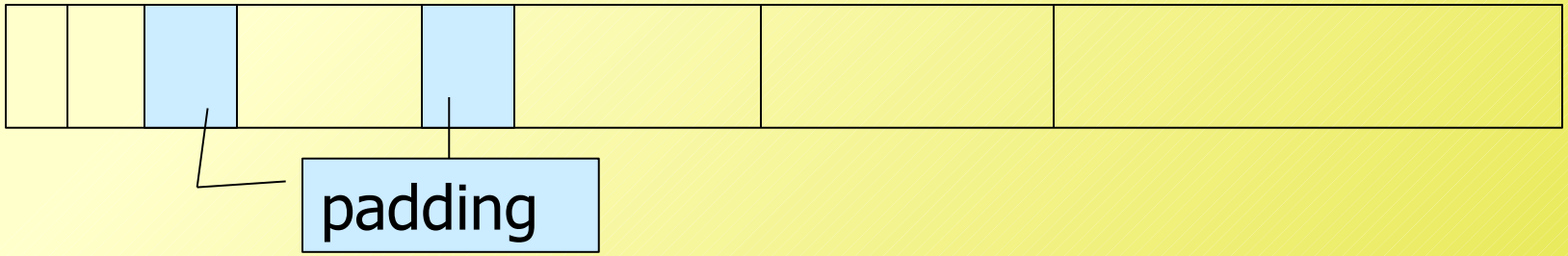
a



b

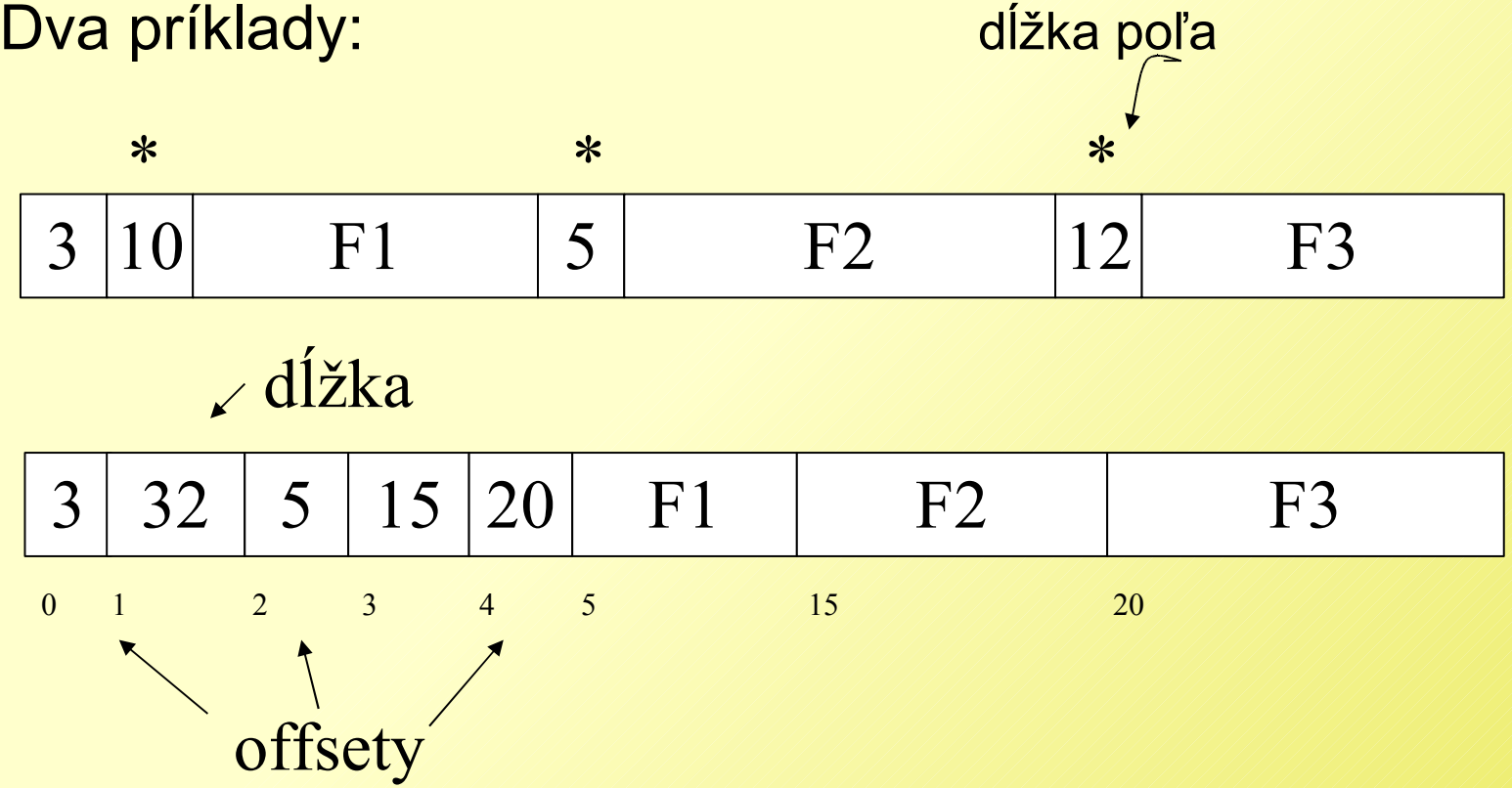
Údajová štruktúra záznamu

- Stavová informácia
  - used / unused bit
  - deleted bit
- Formát (typ) záznamu
  - informácia o štruktúre záznamu
  - informácia o dĺžke a pozícii jednotlivých položiek (priamo alebo prostredníctvom oddelovačov)
- Výplň (padding, waste)
- Vlastná uložená informácia



Sú možné rôzne organizácie záznamov

Dva príklady:



Údajová štruktúra bloku

- Hlavička bloku
  - veľkosť bloku, typ bloku
  - smerníky na predošlý a nasledujúci blok
  - počet záznamov, smerníky na jednotlivé záznamy
  - údaj o volných pozíciách a vynechaných záznamoch
- Uložené záznamy
- Volné miesto

Zjednodušenie:

Bloky a záznamy pevnej dĺžky.

Spôsoby uloženia záznamov v bloku

- Oddelovanie záznamov
  - pevná dĺžka
  - oddelovač
  - dĺžka alebo ofset
    - v každom zázname
    - v hlavičke bloku
- Môžu ísť záznamy cez bloky ? (spanned vs. unspanned)
  - nie, jednoduchšie
  - potrebné, ak je záznam dlhší ako blok
- Rôzne typy záznamov v bloku (clustering), optimalizácia
  - lepšie nedovoliť, ale cluster na jednom cylindri
- Rozdelenie záznamov s premenlivou časťou (split records) premenne část v inom bloku
  - rýchly scan fixných častí / viac IO operácií pri celom zázname

Adresácia

- Fyzická adresa
  - adresa bloku: Disk, cylinder, stopa, blok
  - offset (poloha v bloku)
  - „pripichnuté záznamy“
    - Nepriama adresácia zobrazenie:
      RecordID → Fyzická adresa
  - trade off: flexibilita (premiestnenie, vynechanie) proti rýchlosti prístupu (indirekcia), „nepripichnuté záznamy“
- Možnosti rôznych kombinácií
  - Napr.: Priama adresa bloku + nepriama adresa v bloku „polopripichnuté záznamy“

Pridelovanie pamäti pre bloky

- First fit
- Best fit
- Worst fit (next free in cyclic address space)

Operácie so súbormi

- Vyhľadanie záznamu (Search, Look-Up)
- Vloženie záznamu (Insert)
- Vynechanie záznamu (Delete)
- Modifikácia záznamu (Update, modification)
- Vytvorenie súboru (Create)
- Ukončenie práce so súborom (close)

Smerníky a ich implementácia

- Interné smerníky
  - radenie blokov súborov
  - smerníky na položky záznamov v záznamoch premennej dĺžky
- Aplikačné smerníky
  - **Absolútne smerníky** - pripichnuté záznamy
  - **Kľúče** - volne pohyblivé záznamy. Musí však byť implementovaná vyhľadávacia štruktúra. Nájdenie záznamu môže vyžadovať viac prístupov k bloku.
  - **Dvojica (b, k)**, kde b je adresa bloku a k je kľúč. Po prinesení bloku hľadanie sa už deje v operačnej pamäti záznamy sa môžu pohybovať v rámci bloku. Polopripichnuté (semi-pinned) adresa pozostáva z dvojice (b, a), kde a je pozícia v adresári bloku.

Organizácia súborov

- Sekvenčné súbory a haldy
- Indexovo - sekvenčné súbory
- Stromovo organizované súbory
- Hašované súbory

Halda (heap) : triviálna schéma absencia organizácie. Bloky tvoria obojsmerne spájaný zoznam. Záznamy sú uložene za sebou ako prichádzali.

Výhody:
- Žiadne náklady na správu
- Ľahké vkladanie a vynechanie

Nevýhody:
- Sekvenčné vyhľadávanie podľa kľúča

Kľúčovo sekvenčné súbory

Kľúčovo sekvenčné súbory sú podobné haldám, odlišujú sa však tým, že súbor sa udržuje utriedený podľa primárneho kľúča.

Problémom je vkladanie vyžaduje odsunutie viet od danej pozície. To je neúnosné. Riešenie **bloky pretečenia** (overflow blocks). Prístup k blokom pretečenia:

- Nepriamo cez primárny súbor
- Prístup s posunom cez smerník v bloku

Zhodnotenie: V porovnaní s predošlou organizáciou sa za minimálne náklady získala možnosť vyhľadávania bisekciou alebo interpoláciou.

Indexovo-sekvenčné súbory

K základnému kľúčovo sekvenčnému súboru sa pridáva pomocná štruktúra index. (**ISAM** - index sequential access method). Index je súbor, ktorého vety postávajú z dvojíc (*kľúč, smerník*). Vlastne stačí začiatok hodnoty kľúča a smerník na blok, v ktorom sú vety s danou hodnotou kľúča.

Vyhľadávanie pozostáva z vyhľadávania v indexe a v primárnom súbore (vlastne v bloku a blokoch preplnenia).

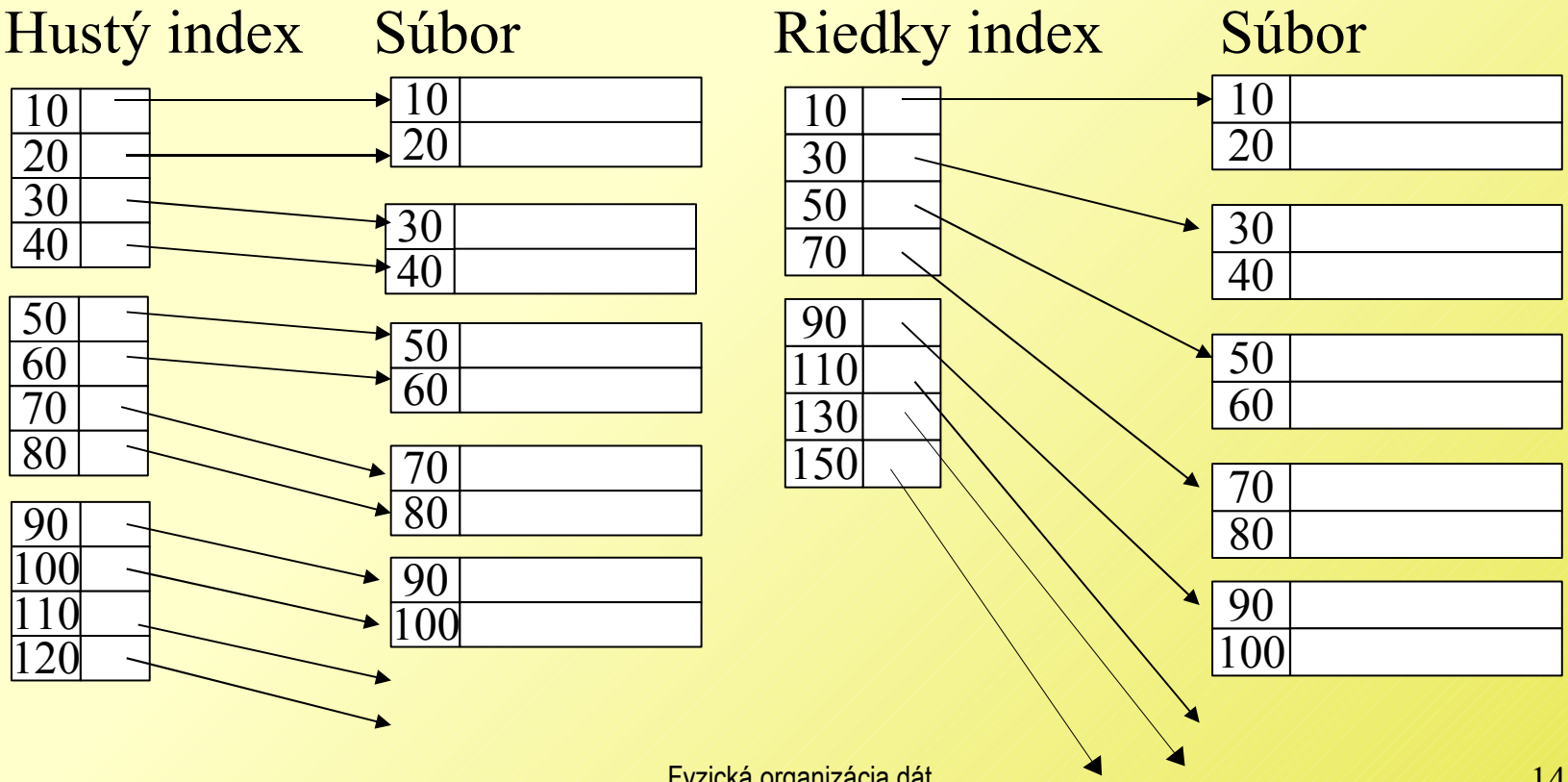
Ak je súbor príliš veľký možno zaviesť index na indexový súbor. Indexy vyššej úrovne. **(HISAM** - hierarchical index sequential access method).

Hardwarovo závislé indexovanie - úrovne: bloky, sektory, stopy, cylindre, disky.

Pri narastaní súboru pribúda blokov preplnenia. Efektívnosť metódy klesá. **Periodické reorganizácie**.

Hustý a riedky index

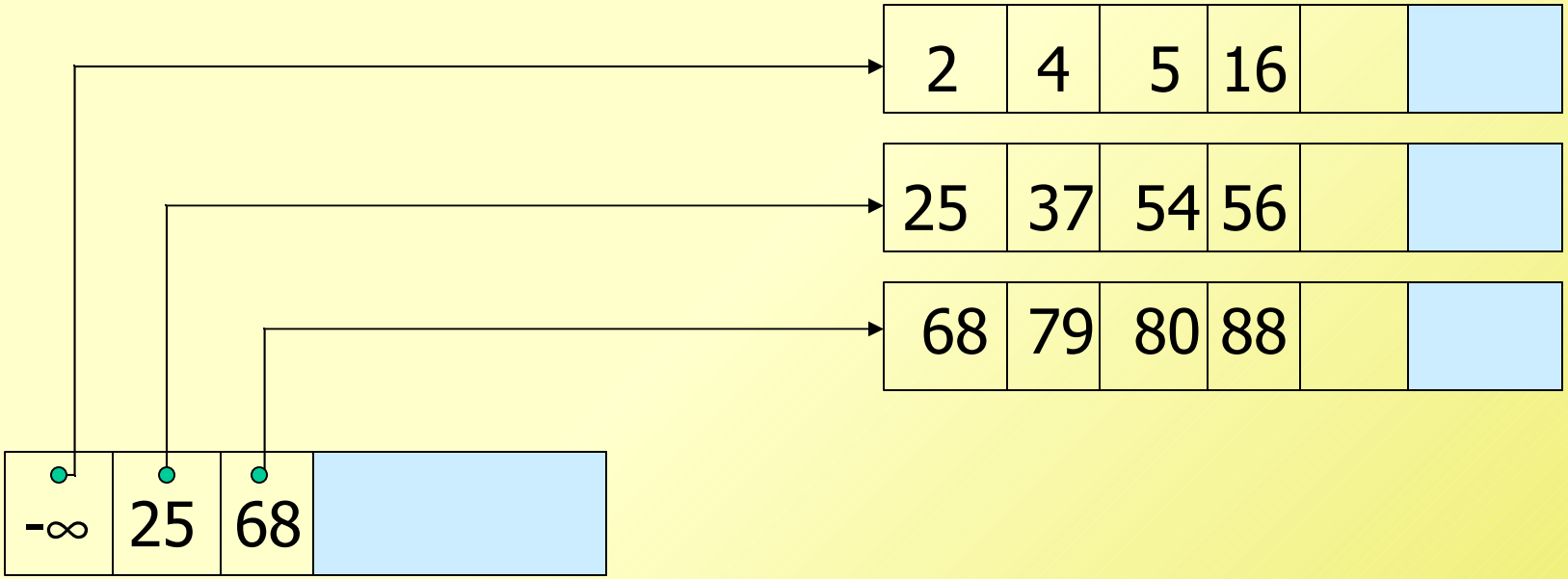
- Hustý (dense) index: Indexy ukazujú na všetky záznamy.
- Riedky (sparse) index: Indexy ukazujú len na niektoré záznamy. Ostatné hľadáme sekvenčne od najbližšieho predošlého záznamu.



Poznámky o indexe

- Smerníky na bloky môžu byť kratšie ako smeníky na záznamy.
- Ak súbor je spojitý a bloky sú pevnej dĺžky môžeme smerníky vynechať a vypočítavať ich.
- Riedky index má menšie nároky na priestor.
- Ak máme hustý index môžeme povedať o existencii záznamu bez prístupu do súboru.
- Do súboru s riedkym indexom sa ľahšie vkladajú záznamy.
- Riedky index k danému súboru môž byť najviac jeden, obvykle pre primárny kľúč.
- Duplikáty
- Vkladanie a vynechanie

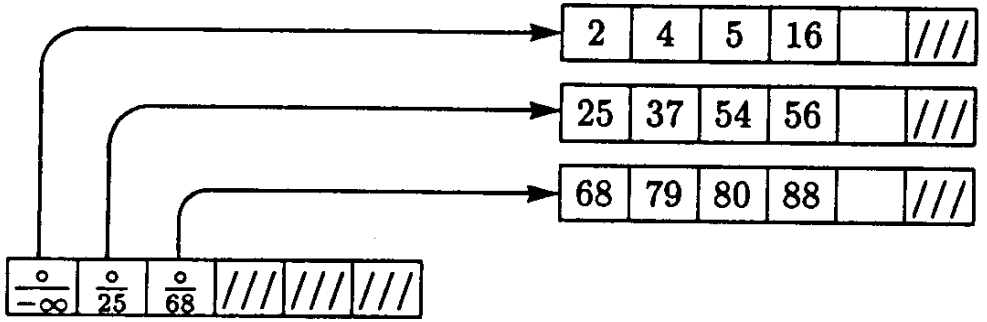
Príklad indexovo-sekvenčného súboru



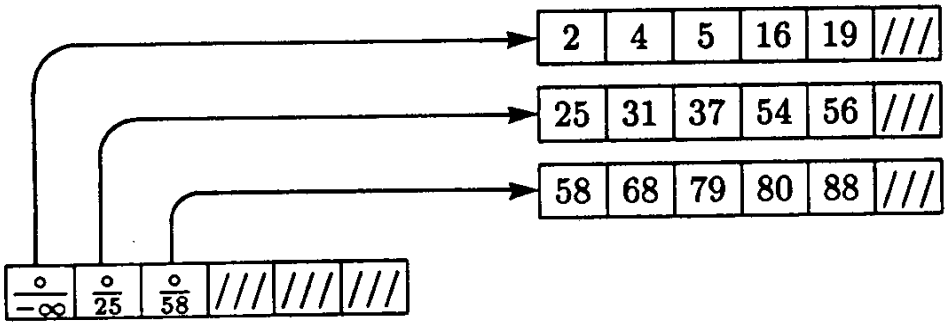
Insert{ 19, 58, 31, 52}

- Pripichuté versus nepripichnuté záznamy
- Utriedené súbory s pripichnutými záznamami

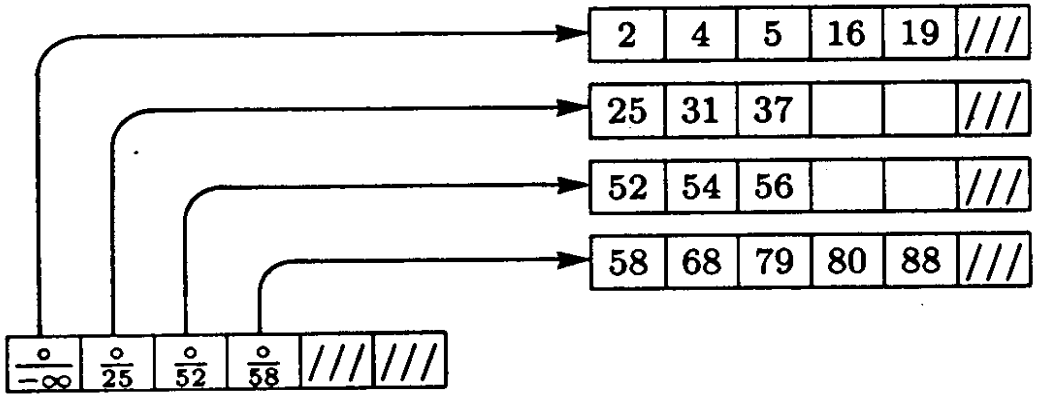
Pôvodná štruktúra index sekvenčného súboru.



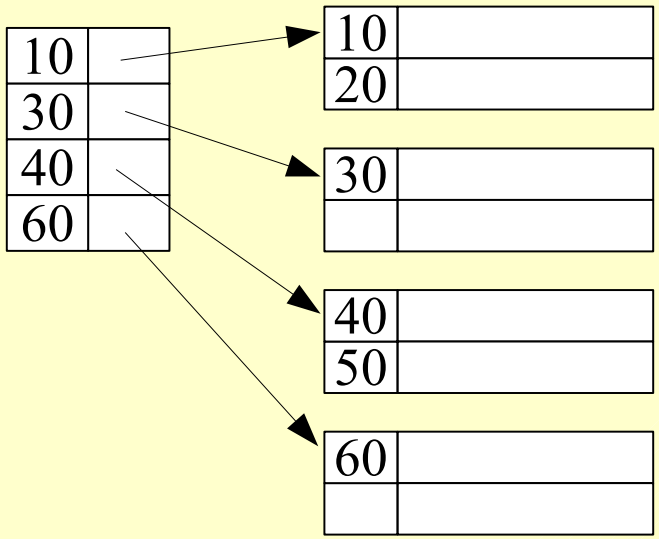
Po inzercií 19, 58 a 31.



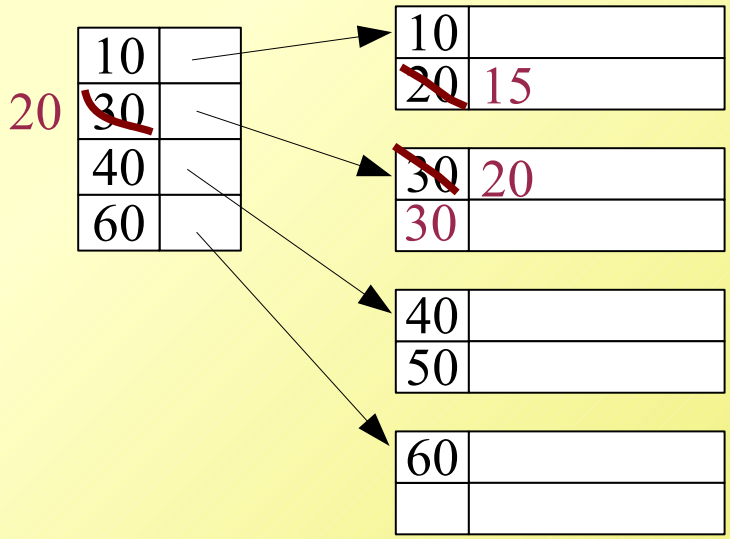
Po následnom vložení klúča s hodnotou 52.



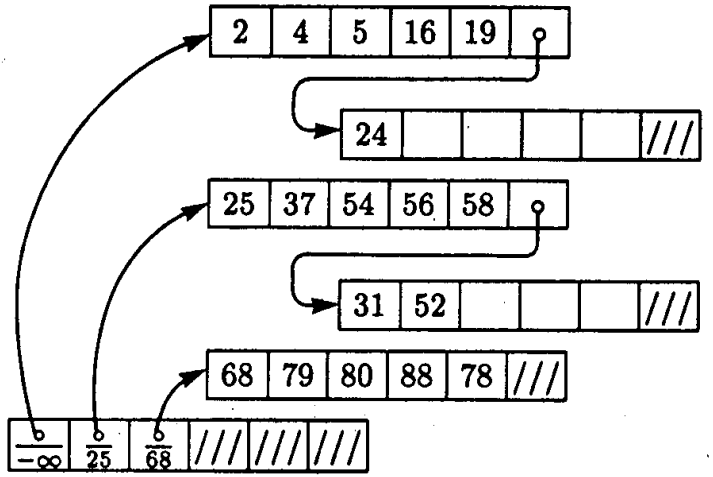
Vkladanie s reorganizáciou



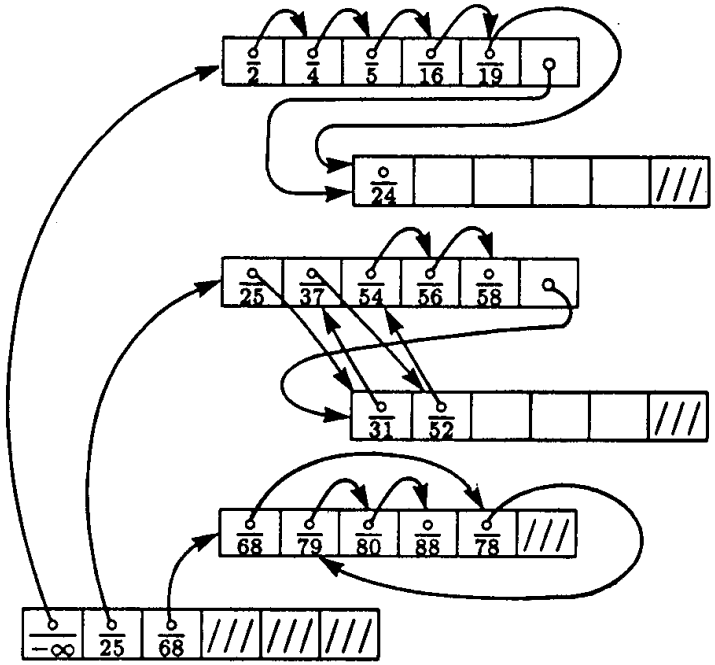
– insert record with key 15



- Ukážka: Reorganizácia v súbore
- Alternatíva:
  - vložiť nový blok (chained file)
  - a aktualizovať index



Vkladanie do súboru
s pripichnutými záznamami.



Smerníky umožňujúce
udržovanie usporiadania
v indexovo sekvenčnom
súbore s pripichnutými
záznamami.

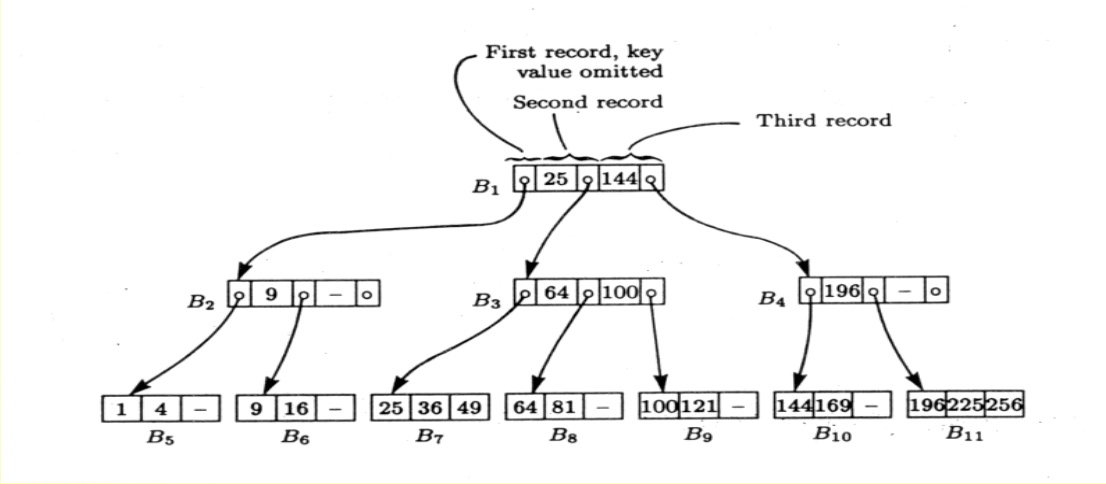
Stromové súbory: B-stromy

Vyhľadávacie stromy (AVL - stromy, 2 - 3 stromy) nie sú pre viacúrovňovú pamäť celkom vhodné, lebo každý uzol obsahuje iba jeden záznam.

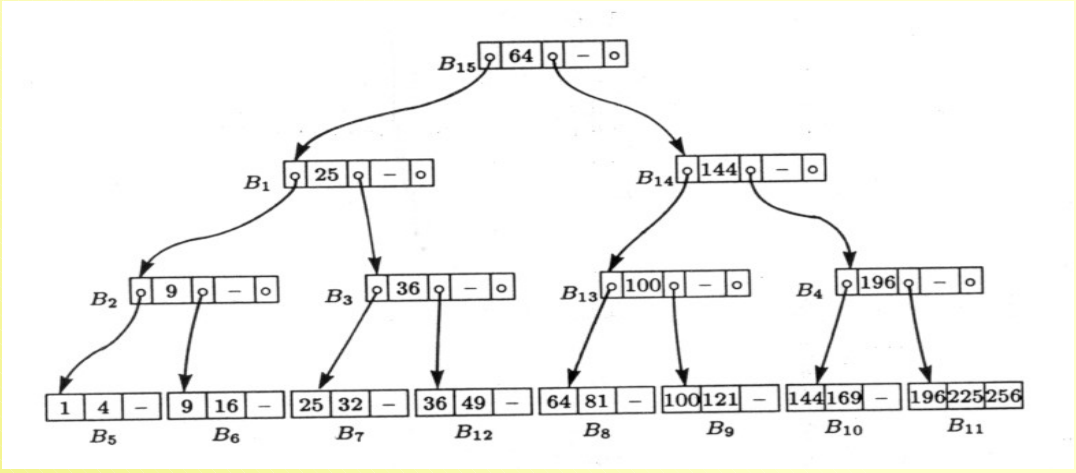
B(m) - strom:

- Každý uzol okrem koreňa obsahuje *k* záznamov, kde $[m/2] \leq k \leq m$. ($[2m/3] \leq k \leq m$, B* stromy)
- Koreň ak nie je listom obsahuje aspoň *1* a najviac *m záznamov*.
- Vnútorný uzol s *k* záznami má *k+1* synov.
- Všetky listy sú na tej istej úrovni.

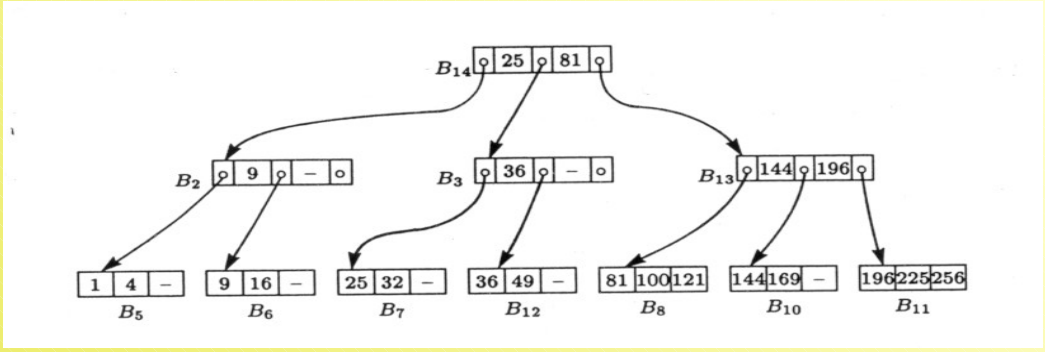
B^+ stromy majú vo vnútorných uzloch kľúče (index) a záznamy sú iba v listoch. Je možné aj rôzne ohraničenie na vnútorný uzol a list.



B - strom



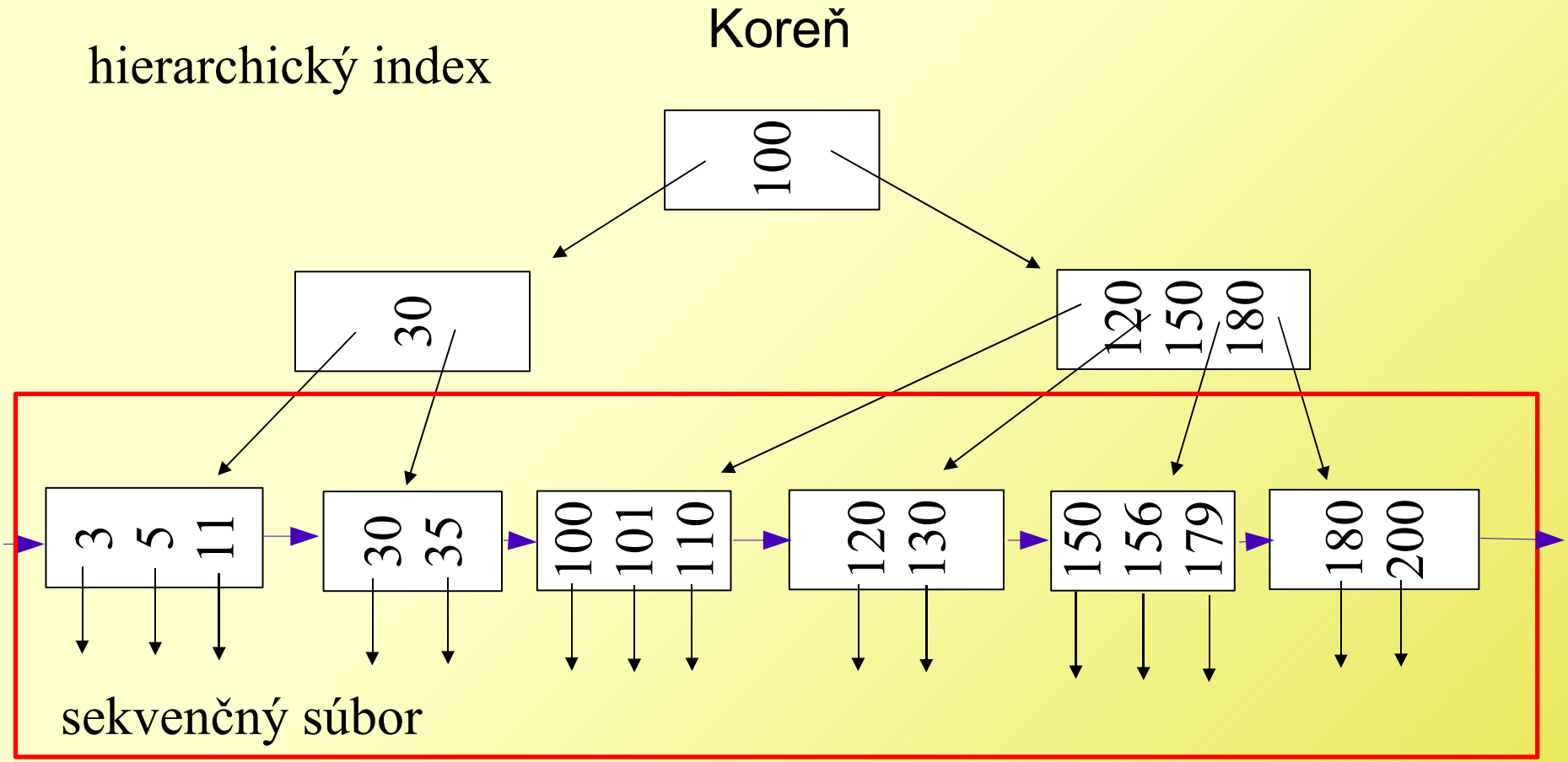
Po vložení 32



Po vynechaní 64

Príklad $B^+(3)$ stromu

Podporuje vyhľadávanie aj sekvenčný scan.



Vkladanie do $B^x(m)$ stromov

- Najdi list do ktorého záznam patrí a vlož
- Ak sa list nepreplnil skonči.
- Ak je list preplnený (m+1 záznamov)
  - Prekontroluj či sa nedajú redistribuovať záznami medzi susednými listami.
    - Ak ano, urob a uprav index (spoločný otec, „rôzni otcovia“)
  - Rozdel list na dva a index pre druhú časť vlož do nadradeného uzla (otca)
- Vloženie do vnútorného uzla je podobné vloženiu do listu, ale vkladá sa aj príslušný smerník. Pri redistribúcii sa redistribujú aj príslušné smerníky.
- Ak sa preplnil koreň, vznikne nový koreň s jednou hodnotou a smerníkami na ľavý a pravý podstrom

Vynechanie z $B^x(m)$ stromov

- Najdi a vynechaj záznam
- Ak je list podplnený
  - ak je brat dostatočne plný
    - redistribuj záznamy medzi listom a bratom
    - aktualizuj otca
  - Inak
    - zlúč list s bratom
    - Vynechaj smerník a index z otca
- Vynechanie z vnútorného uzla je podobné ako vynechanie z listu. Môže sa propagovať až ku koreňu.
- Ak vynecháme poslednú hodnotu z koreňa, tento zanikne. Novým koreňom sa stane jeho jediný syn.
- Často sa zlučovaniu uzlov vyhýbame. Súbory majú tendenciu rásť.

Stratégie indexovania

Vo všetkých predošlých prípadoch išlo riedky index.
B stromy vyžadujú nepripichnuté záznamy.

Ak používame sekundárne indexy vzniká potreba hustého indexu a pripichnutých záznamov.

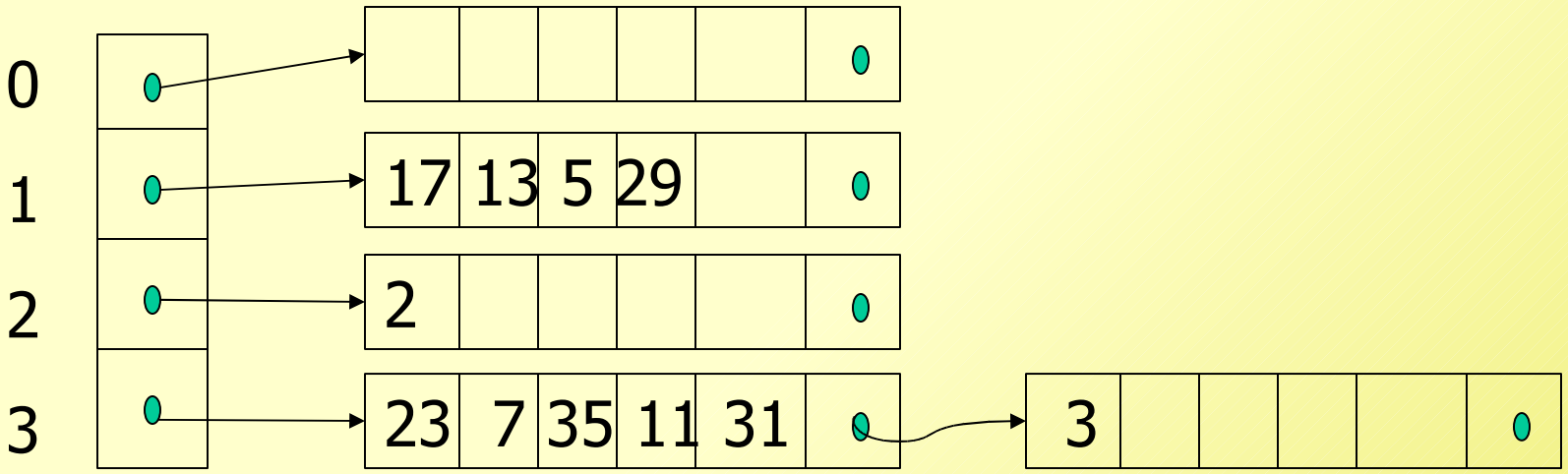
Záznamy možno zmeniť na nepripichnuté ak sekundárny index zostrojíme pre primárne kľúče. Invertovaný súbor obsahuje len záznamy tvaru dvojíc:
<index, primárny kľúč záznamu v pôvodnom súbore>.
Redukuje sa tým významne potreba reorganizácie.

Hašovanie - transformácia kľúča

Vytvorenie čísla z kľúča:
Nech $m = \log_2 M$, *kde je* M počet priečinkov (buckets).

- Polámanie na úseky dĺžky *m* ich XOR alebo súčet *mod M.*
- Nejakých *m* bitov zo stredu mocniny
- Kľúč k berieme ako postupnosť bitov $h(k) = k \bmod M.$
- $h(k) = [M.(k.A \bmod 1)]$, kde $0 < A < 1$. (Fibonacciho transformačná funkcia $A = (\sqrt{5} - 1)/2$.)
- Polynomiálna hašovacia funkcia
- $h(k) = [(M*(k-a)/(b-a)]$, kde $a \leq k < b$. „Hašovacia" funkcia zachovávajúca usporiadanie.

Štruktúra hašovaného súboru



adresár základné bloky bloky preplnenia

Technické problémy

- prázdne buckety
- pripichnuté záznamy
- vynechanie

Faktor naplnenia: $\alpha = N/M$
Ak $\alpha < b$ a adresár je v hlavnej pamäti očakávaný prístup k jednému alebo niekoľkým málo blokom.

Inžinierské pravidlo (rule of thumb)

- Využitie priestoru $\beta = \alpha b^{-1}$ je to pomer obsadeného priestoru k celkove dostupnému priestoru.

Pravidlo: Udržovať $0.5 < \beta < 0.8$

- Ak je $\beta < 0.5$ dochádza k zbytočnému vastovaniu priestoru a zbytočným prenosom skoro prázdnych blokov.
- Ak je $\beta > 0.8$ počet preplnení blokov silne závisi na kvalite hašovacej funkcie a počte záznamov v bloku b.

Podľa štatistickej teórie: pri kvalitnej hašovacej funkcii môžeme ísť až k β blízske 1. Očakávaný počet záznamov v bloku ak $\beta = 1$ je b a disperzia tiež b. Stredná kvadratická odchylka $O(b^{1/2})$.