



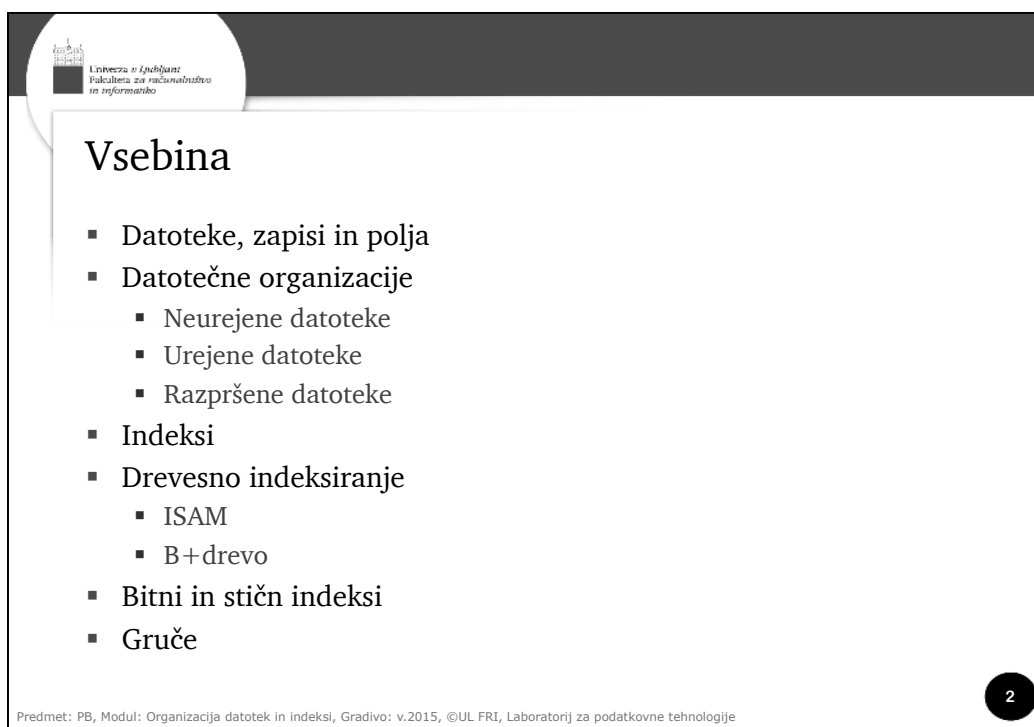
Univerza v Ljubljani
Fakulteta za računalništvo
in informatiko

Predmet:
Osnove podatkovnih baz

Modul:
Organizacija datotek in indeksi

Gradivo:
v.2015

11, 18.3.
2016



Univerza v Ljubljani
Fakulteta za računalništvo
in informatiko

Vsebina

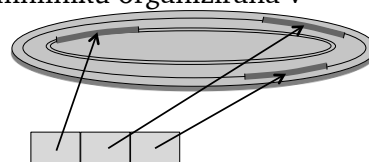
- Datoteke, zapisi in polja
- Datotečne organizacije
 - Neurejene datoteke
 - Urejene datoteke
 - Razpršene datoteke
- Indeksi
- Drevesno indeksiranje
 - ISAM
 - B+drevo
- Bitni in stični indeksi
- Gruče

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

2

Osnovni koncepti...

- Podatkovna baza je na sekundarnem pomnilniku organizirana v eno ali več datotek (*file*)



- Vsaka datoteka zajema enega ali več zapisov (*record*).
- Zapis sestavljajo polja (*field*).
- Zapisi običajno označujejo entitete, polja pa njihove attribute.

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

3

Osnovni koncepti...

- Uporabnik zahteva zapis A10 od SUPB:
 - SUPB naredi preslikavo logičnega zapisa v fizični;
 - Poišče fizični zapis in ga prepíše v primarni pomnilnik oziroma v medpomnilnik;

Logični pogled

Tabela

Šifra	Naziv	Zaloga
A10	Telovadni copati Nike	10
A12	Trenerka Bali	4
BC80	Moška jakna QuickSilver	1
X12	Ženska jakna QuickSilver	0

entiteta/zapis

Fizični pogled

datoteka



atribut/polje

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

4

Osnovni koncepti...

- Fizični in logični zapis: ne velja vedno 1:1
 - Fizični zapis enota prenosa med diskom in primarnim pomnilnikom. Lahko zajema več logičnih zapisov.
 - Večji logični zapis lahko zapisan čez več fizičnih zapisov.
- Fizični zapis ustreza konceptu strani.

5

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

Osnovni koncepti

ARTIKEL

Šifra	Naziv	Zaloga	Stran
A10	Telovadni copati Nike	10	1
A12	Trenerka Bali	4	
BC80	Moška jakna QuickSilver	1	2
X12	Ženska jakna QuickSilver	0	

6

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

Datotečna organizacija...

- Fizična urejenost podatkov v zapise in strani na sekundarnem pomnilniku → datotečna organizacija.
- Osnovne vrste datotečnih organizacij:
 - Kopica ali neurejena datoteka: zapisi so na disku shranjeni v nedefiniranem vrstnem redu.
 - Zaporedno urejena datoteka: zapisi so urejeni po vrednosti določenega polja.
 - Razpršena datoteka: zapisi so razpršeni z uporabo hash funkcije.

Datotečna organizacija...

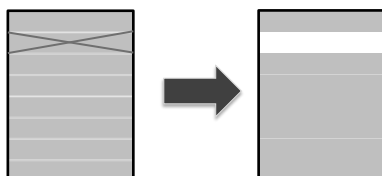
- Za delo z zapisi v datotekah obstajajo različne tehnike ali metode dostopa (*access methods*):
 - določajo korake, ki jih je potrebno izvesti za shranjevanje ali iskanje nekega zapisa v datoteki.
 - odvisne od datotečne organizacije.

Neurejene datoteke...

- Imenujemo tudi kopica (*heap*).
- Najenostavnejša datotečna organizacija:
 - Zapisi shranjeni v vrstnem redu, kot so bili dodani
 - Nov zapis dodan na zadnjo stran datoteke
 - Če ni dovolj prostora, se doda nova stran
- Dobra lastnost:
 - Zelo učinkovito dodajanje zapisov - ni potrebno računati, na katero stran bomo zapis vstavili.
- Uporabljamo za masovni vnos.

Neurejene datoteke

- Slabosti:
 - Neučinkovitost iskanja: linearno iskanje (zapisi so neurejeni)
 - Neizkoriščenost prostora: brisani zapisi puščajo prazen prostor na straneh... neurejene datoteke potrebno občasno reorganizirati!



Urejene datoteke...

- Zapisi v datotekah urejeni po enem ali več poljih → urejena ali zaporedna datoteka.
- Možno binarno iskanje.
- Primer:
 - Poišči artikel s šifro 14

1	2	3	4	5	6	7	8	9	10	11	12	13
10	12	14	18	23	34	35	65	72	78	89	90	99

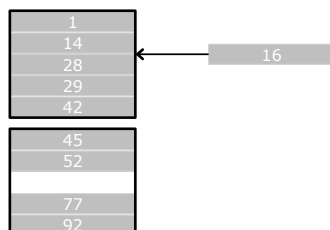
↑

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

11

Urejene datoteke...

- Prednosti:
 - Učinkovitost iskanja
- Problem
 - Dodajanje in brisanje zapisov → potrebno vzdrževati vrstni red
 - Kaj če na strani ni dovolj prostora?



Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

12

Urejene datoteke

- Dodajanje zapisa na začetek velike datoteke posebej problematično.
- Možna rešitev: uporaba dodatne neurejene datoteke (*overflow file*):
 - Nov zapis dodan v neurejeno datoteko
 - Neurejena datoteka se periodično prepíše v urejeno
 - Pri iskanju zapisa se najprej pogleda urejena datoteka. Če zapisa ne najdemo, se linearno pregleda še neurejena.



Razpršene ali hash datoteke...

- Zapisi razpršeni v skladu s hash funkcijo.
- Hash funkcija za vsak zapis izračuna naslov strani (naslov bloka na disku), kamor zapis sodi glede na vrednost določenega polja (hash polje).

$f_{\text{hash}}(P) \rightarrow \text{naslov strani};$
 P: vrednost hash polja.

- Iskanje zapisa na strani se izvede v primarnem pomnilniku. Za večino zapisov moramo prebrati le eno stran.

Razpršene ali hash datoteke...

- Operacije za hash funkcijo mora zagotavljati enakomerno porazdeljenost zapisov po datoteki.
- Tipična hash funkcija: ostanek pri deljenju (MOD) z določenim številom (n)

Primer hash funkcije

- MOD 11



Razpršene ali hash datoteke...

- Slabosti razpršenih datotek:
 - Zaloga vrednosti hash polja navadno večja od števila naslovov, ki jih lahko vrne hash funkcija.
 - Vsak naslov ustreza določeni strani (*bucket*), ki ima mesta za več zapisov.
 - Linerano iskanje zapisov po strani: znotraj strani zapisi urejeni po vrsti, kot so bili vstavljeni.
 - Ko hash funkcija za nek zapis vrne naslov strani, ki je polna, pride do kolizije.

Razpršene ali hash datoteke...

- Tehnike za reševanje kolizije:
 - Odprto naslavljanje (*open addressing*)
 - Nepovezane dodatne strani (*unchained overflow*)
 - Povezane dodatne strani (*chained overflow*)
 - Večkratno razprševanje (*multiple hashing*)

Odprto naslavljanje...

- **Pravila:**
 - Zapisovanje: kolizija → poiščemo prvo stran, ki ima še kakšno prosto mesto. Ko pridemo do zadnje strani, gremo na začetek.
 - Iskanje: enako kot pri zapisovanju. Če naletimo na prazno mesto preden na iskani zapis → zapisa ni v datoteki.

Primer odprtega naslavljanja

- Hash operacija: MOD 3
- Dodajamo zapis SL 41

?

Stanje prej	Stran
SA 9	0
SG 37	1
SG 5 SG 14	2

Stanje potem	Stran
	0
	1
	2

Nepovezane dodatne strani

- Vzdržujemo seznam dodatnih strani

Primer

Osnovne strani	Stran	Dodatni prostor	Stran
SA 9	0	SL 41	3
SG 37	1		4
SG 5	2		
SG 14			

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

21

Povezane dodatne strani

- Dodatne strani povezane z osnovnimi
- Vsaka stran ima dodatno polje, ki pove, ali je prišlo pri tej strani do kolizije ali ne. V polju je naslov dodatnega polja, kamor so razvrščeni zapisi, pri katerih pride do kolizije. Vrednost 0 pomeni, da kolizije ni bilo.

Primer

Osnovne strani	Stran	Dodatni prostor	Stran
SA 9 SK 12	0	SL 41	3
SG 37	1		4
SG 5 SG 14	2		

Diagram showing a dashed arrow labeled "synonym pointer" from the entry "3" in the "Stran" column of the "Osnovne strani" table to the entry "0" in the "Stran" column of the "Dodatni prostor" table.

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

22

Večkratno razprševanje

- V primeru kolizije uporaba dodatne hash funkcije (vrača drugačen naslov).
- Dodatna hash funkcija navadno razvršča v dodati prostor.

Primer:

- H_1 : MOD 3, H_2 : $3 + \text{MOD } 5$; dodajamo SL 11

Osnovne strani	Stran	Dodatni prostor	Stran
SA 9	0		3
SG 37	1	SL 41	4
SG 5	2		
SG 14			

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

23

Razpršene ali hash datoteke...

- Razprševanje v splošnem
 - Prinaša dobre rezultate pri iskanju (ob uporabi ene izmed tehnik reševanja kolizij).
 - Spreminjanje zapisov enostavno, razen v primerih, ko spremenimo hash polje.

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije


24

Dinamične razpršene datoteke...

- Obravnavane hash tehnike statične: ko zapisu določimo naslov, se ta ne spremeni, če se ne spremeni vrednost hash polja .
- Problem: datoteka postane premajhna → vzamemo večjo, določimo novo hash funkcijo, vse zapise ponovno razpršimo.
- Alternativa: uporaba dinamičnih razpršenih datotek: velikost datoteke spreminjamo po potrebi.
- Več tehnik realizacije dinamičnih razpršenih datotek.

Razširljivo razprševanje (*extendible hashing*)

- Pravila:
 - Strani kreiramo po potrebi. V začetku gredo zapisi v prvo stran.
 - Ko stran polna, razdelimo glede na prvih i bitov, kjer velja $0 \leq i < b$
 - Izbranih i bitov določa naslov oziroma offset v naslovni tabeli strani (BAT - *Bucket Address Table*). Vrednost i se spreminja z velikostjo datoteke.
 - V glavi imenika je zapisana trenutna vrednost i (globina) skupaj z 2^i kazalci.
 - Vsaka stran ima tudi lokalno globino, ki pove, pri katerem i dobimo naslov te strani.

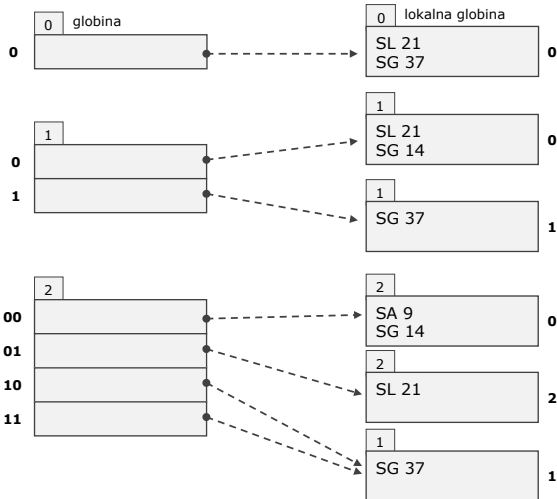


Univerza na Ljubljani
Fakulteta za računalništvo
in informatiko

21 = 010101
37 = 100101
14 = 001110
09 = 001001

Primer razširljivega razprševanja

Imenik



The diagram illustrates the growth of a hash table using separate chaining. It shows three stages of insertion:

- Stage 1:** The hash table has one slot (index 0) labeled 'globina'. It points to a file entry with 'SL 21' and 'SG 37'.
- Stage 2:** A second slot (index 1) is added. Slot 0 still points to 'SL 21, SG 37'. Slot 1 points to a new file entry 'SL 14'.
- Stage 3:** A third slot (index 2) is added. Slot 0 now points to two entries: 'SA 9, SG 14' and 'SL 21, SG 37'. Slot 1 points to 'SL 21, SG 37'. Slot 2 points to a new file entry 'SL 21'.

Izgled dinamično razpršene datoteke po vpisu zapisov SL 21 in SL 37.

Izgled dinamično razpršene datoteke po vpisu zapisa SG 14

Izgled dinamično razpršene datoteke po vpisu zapisa SA 9.

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

28



Univerza v Ljubljani
Fakulteta za računalništvo
in informatiko

Razpršene ali hash datoteke

- Razprševanje v splošnem
 - Prinaša dobre rezultate pri iskanju (ob uporabi ene izmed tehnik reševanja kolizij).
 - Spreminjanje zapisov enostavno, razen v primerih, ko spremenimo hash polje.
- Omejitve:
 - Učinkovitost razpršenih datotek za iskanje odvisna od hash polja. Uporaba razpršenih datotek ni primerna za:
 - Iskanje po vzorcu
 - Iskanje po nizu vrednosti
 - Iskanje po polju, ki ni hash polje

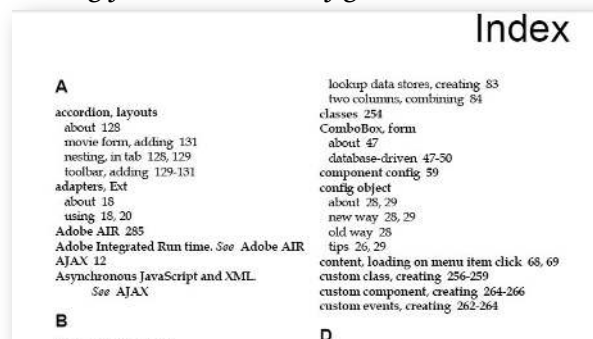
29

Ponovitev

- Kakšne datotečne organizacije poznamo?
- Zakaj je datotečna organizacija pomembna?
- V kakšnih primerih bi uporabili:
 - Urejeno datoteko
 - Neurejeno datoteko
 - Razpršeno datoteko
- Datoteka ima n strani. Koliko I/O operacij je potrebnih pri iskanju, če je datoteka:
 - urejena
 - neurejena
 - razpršena

Indeksi in indeksiranje...

- Indeks je podatkovna struktura, ki SUPB-ju omogoča hitrejše lociranje zapisov v datoteki.
- Analogija z indeksom knjige



Univerza na Ljubljani
Fakulteta za računalništvo in informatiko

Primer indeksa

Indeks

Priimek
Bolčina
Bregar
Bric
Colja
Kavčič
Markovič
Robnik
...

Podatkovna datoteka

Ime	Priimek	Spol	...	Starost
Mateja	Bolčina	Ž		27
Teja	Bregar	Ž		66
Miha	Bric	M		43
Dejan	Bric	M		29
Milka	Colja	Ž		32
Gregor	Kavčič	M		62
Iztok	Kavčič	M		55
Nina	Kavčič	Ž		33
Tadej	Markovič	M		45
Breda	Robnik	Ž		37
...				

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

32

Univerza na Ljubljani
Fakulteta za računalništvo in informatiko

Indeksi in indeksiranje...

- Terminologija:
 - Podatkovna datoteka: datoteka s podatki (imenujemo tudi osnovna datoteka)
 - Indeksna datoteka: datoteka z indeksom. Indeks sestavlja iskalni ključ ter kazalec na zapis v podatkovni datoteki.
 - Iskalni ključ: sestavljen iz vrednosti polj, po katerih je datoteka indeksirana. Imenujemo tudi indeksno polje.

Iskalni ključ

B003, 18000

Indeksna datoteka

Zapis

SG14	David	Ford	18000	B003
------	-------	------	-------	------

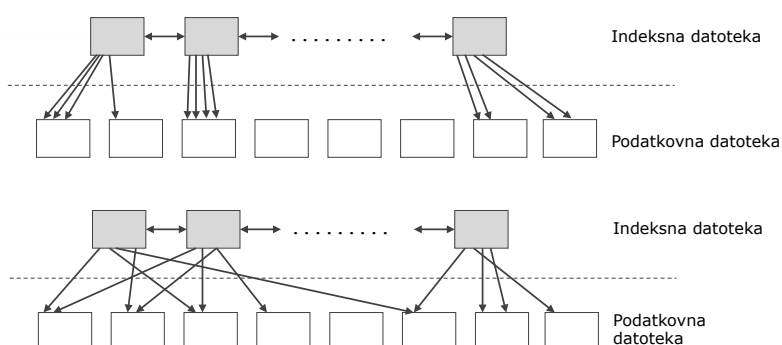
Podatkovna datoteka

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

33

Indeks gruče (*clustered index*)

- Indeks po polju, po katerih je urejena podatkovna datoteka.



Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

34

Primerni in sekundarni indeksi

- Primarni indeks (*Primary index*): indeks po poljih, ki vsebujejo primarni ključ. Vsak iskalni ključ kaže natanko na en zapis v podatkovni datoteki. Datoteka je po ključu urejena.
- Sekundarni indeks (*Secondary key*): vsak indeks, ki ne temelji na poljih, ki bi vsebovala primarni ključ.
- Vsaka datoteka ima lahko:
 - primarni indeks ali indeks gruče ter
 - več sekundarnih indeksov!

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

35

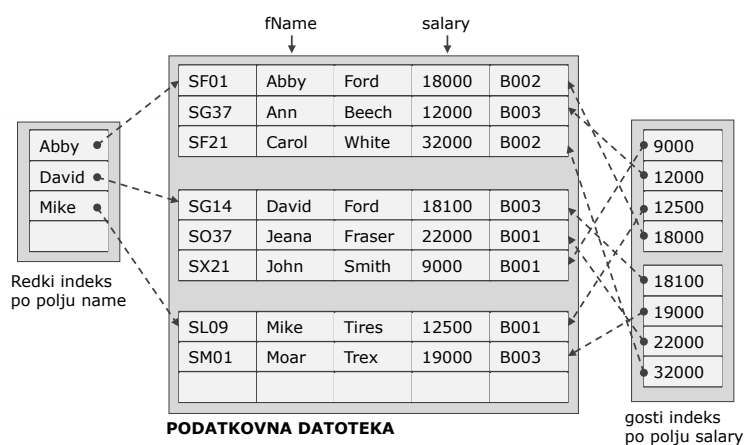
Redki in gosti indeksi

- Redki indeks (*sparse index*): indeksna datoteka vsebuje kazalce, ki kažejo le na določene zapise v podatkovni datoteki. Navadno vsebuje po en zapis za vsako stran v podatkovni datoteki;
- Gosti indeks (*dense index*): indeksna datoteka vsebuje kazalce na vse zapise v podatkovni datoteki.

36

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

Primer redkega in gostega indeksa

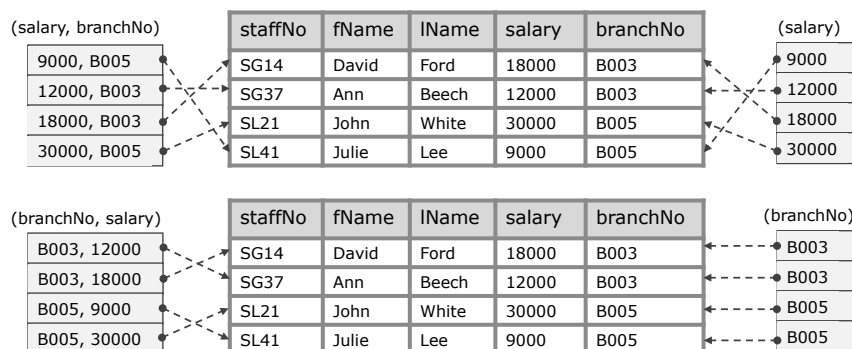


37

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

Sestavljen indeks

- Indeksi s sestavljenim iskalnim ključem (*composite key*):
 - Indeksi po več kot enem polju
 - Uporabljamo za kombinacije polj, po katerih pogosto iščemo



Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

38

Vaja

- Imamo tabelo finančnih transakcij z 50 milioni zapisov. Vsak zapis ima naslednje atribute:
 - ID transakcije, datum, znesek, tip transakcije, plačnik, prejemnik, valuta
- Transakcije bomo predvidoma iskali po:
 - Datumu – npr. vse transakcije iz določenega dne ali obdobja,
 - Plačniku – npr. vse transakcije določenega plačnika,
 - Plačniku in prejemniku – npr. vse transakcije določenega plačnika, agregirano po prejemniku.
- Kakšno datotečno organizacijo in indekse bi uporabili?

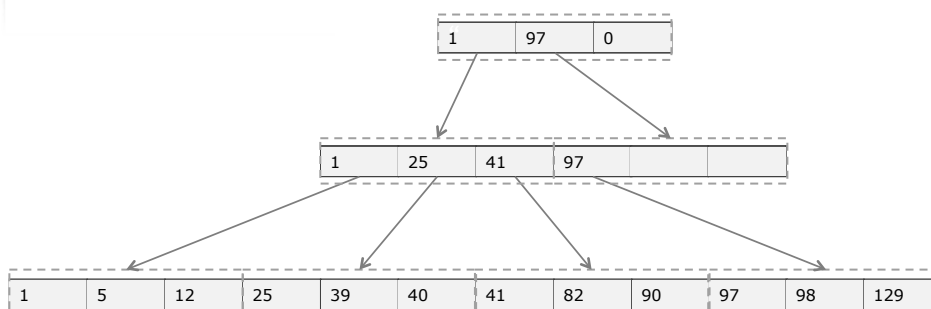
Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

39

Drevesno indeksiranje...

- Primer:
 - Imamo datoteko oseb, urejeno po polju *starost*.
 - Če želimo najti vse osebe starejše od 30 let, moramo najprej najti prvo (s pomočjo binarnega iskanja), ki je starejša od 30 in od te naprej prebrati preostale zapise datoteke.
- Možnost za pohitritev iskanja:
 - Izdelamo dodatno datoteko, s po enim zapisom za vsako stran v podatkovni datoteki in jo uredili po polju *starost*.
 - Vsak zapis v dodatni datoteki vsebuje par
 <ključ prvega zapisa na strani, kazalec na zapis>

Primer drevesnega indeksa



Drevesno indeksiranje...

- Drevesno indeksiranje učinkovito pri:
 - intervalnem iskanju ter
 - dodajanju in brisanju (za razliko od urejenih datotek).
- Iskanje po enakosti boljše pri hash indeksiranju.
- Dve tipični indeksni strukturi, ki temeljita na drevesni organizaciji:
 - ISAM (*Indexed Sequential Access Method*):
 - B+ drevesna struktura

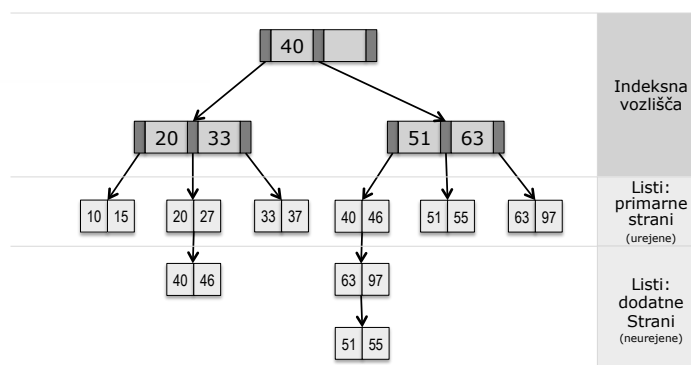
Drevesno indeksiranje...

- ISAM:
 - statičen indeks,
 - učinkovit v primerih, ko se podatkovna datoteka ne spreminja pogosto,
 - ni učinkovit pri datotekah, ki se hitro povečujejo ali krčijo.
- B+ drevo:
 - dinamična struktura; se zelo dobro prilagaja spremembam v podatkovni datoteki,
 - najbolj uporabljana vrsta indeksa; omogoča hitro iskanje zapisov, intervalno iskanje in se učinkovito prilagaja spremembam v podatkovni datoteki.

ISAM...

- ISAM statična struktura (izjema dodatne strani)
- Gradnja ISAM indeksa:
 - Ko se indeksna datoteka kreira, so vse strani v listih urejene zaporedno in po iskalnem ključu.
 - Vstavljanje novih podatkov lahko zahteva kreiranje dodatnih strani (če je stran, kamor podatek sodi, polna).
- Podprte operacije: operacije iskanje, brisanje in vstavljanje.

Primer ISAM drevesa

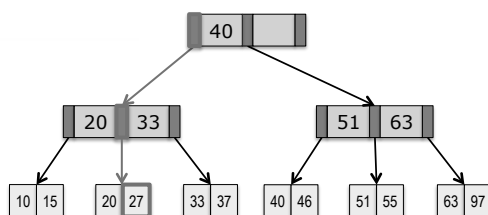


Iskanje po ISAM indeksu

- Postopek:
 - Začnemo v korenu
 - Če iskana vrednost manjša ali enaka ključu korena, sledimo levemu kazalcu, sicer desnemu.
 - Ponavljamo, dokler ne pridemo do listov drevesa. Če iskanega podatka ni v listu, iščemo v dodatnih straneh.

Primer iskanja po ISAM drevesu

- Iščemo element 27

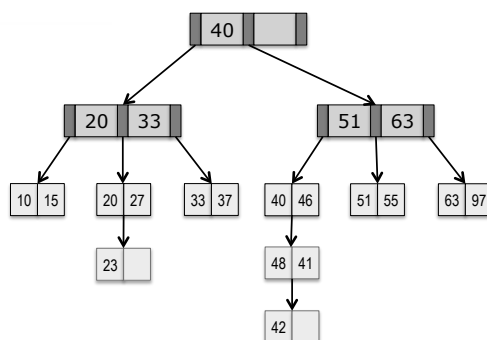


Dodajanje v ISAM indeks

- Postopek:
 - Stran v listih, kamor zapis sodi, poiščemo enako kot pri iskanju
 - Če je prosto mesto v primarnem listu, moramo element dodati na ustrezno mesto (primarni listi so urejeni)
 - Če v primarnih listih ni prostora, dodamo element na prvo prosto mesto v dodatnih listih. Po potrebi dodamo nov list v seznam dodatnih listov.

Primer dodajanja v ISAM drevo

- Dodajamo elemente 23, 48, 41, 42



Brisanje iz ISAM indeksa

- Postopek:
 - Zapis, ki ga brišemo, poiščemo enako kot pri iskanju
 - Če je zapis na dodatni strani in gre za zadnji zapis na strani, stran zberemo.
 - Če je zapis na primarni strani in gre za zadnji zapis na strani, pustimo stran prazno. Služi kot mesto za kasnejšo rabo.
 - Lahko se zgodi, da se iskalni ključ, ki nastopa v indeksnem delu, ne pojavi v listih.

Stroški iskanja v ISAM strukturi

- Primer izračuna stroškov iskanja zapisa v ISAM strukturi:
 - Število I/O operacij = $\log_F N$, kjer je N število primarnih strani listov, F število otrok vsake indeksne strani.
 - Število I/O operacij pri binarnem iskanju po urejeni datoteki je $\log_2 N$. Pri iskanju po eno-nivojskem indeksu: $\log_2(N/F)$
 - Primer: datoteka z 1.000.000 zapisi, 10 zapisov na stran v listih in 100 zapisov v indeksnih straneh
 - Strošek branja cele datoteke: 100.000
 - Strošek binarnega iskanja po urejeni datoteki: 17
 - Strošek binarnega iskanja po eno-nivojskem indeksu: 10
 - Strošek binarnega iskanja po ISAM strukturi: 3 (brez dodatnih strani)

Omejitve ISAM indeksne strukture

- Problem ISAM strukture se pokaže, ko naraste število dodatnih strani
 - Podatki v dodatnih straneh so načeloma lahko urejene, običajno pa niso (zaradi učinkovitosti dodajanja zapisov)
 - Problem omilimo tako, da v začetku, ko izgradimo indeks, pustimo nekaj praznega prostora v straneh listov.

52

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

Kje smo?



- Datoteke, zapisi in polja
- Datotečne organizacije
 - Neurejene datoteke
 - Urejene datoteke
 - Razpršene datoteke
- Indeksi
- Drevesno indeksiranje
 - ISAM
 - ➔ ▪ B+drevo
- Bitni in stični indeksi
- Gruče

53

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

B+ drevesa...

- B+ indeks je dinamičen → njegova struktura se dinamično prilagaja spremembam v podatkovni datoteki.
- Odpravlja težave ISAM indeksa, npr. s povečevanjem števila dodatnih strani pada učinkovitost...
- B+ je uravnoteženo drevo, katerega vozlišča usmerjajo iskanje, listi pa vsebujejo podatke (ključe).

B+ drevesa

- Zaradi dinamične narave B+ drevesa, strani v listih ni možno alocirati zaporedno. Uporabimo kazalce.
- Liste B+ drevesa uredimo z dvosmernim seznamom.



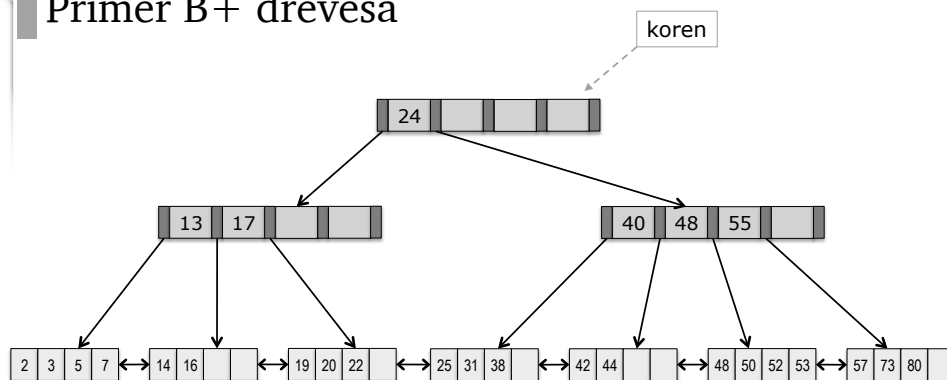
Lastnosti B+ dreves

- Operacije dodajanja in brisanja ohranjajo drevo uravnoteženo;
- Vozlišča (razen korena) so vsaj 50% zasedena, če uporabimo ustrezen algoritem brisanja;
- Iskanje določene vrednosti zahteva le pot od korena do ustreznega lista.
- Poti do vseh vozlišč so zaradi uravnoteženosti enake in določajo višino drevesa.
- Zaradi velikega razvejanja (parameter F) je višina običajnih B+ dreves majhna (redko več kot 3 ali 4).

56

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

Primer B+ drevesa



d je parameter B+ drevesa imenovan red drevesa;

2^d predstavlja kapaciteto vozlišča. Edina izjema je korensko vozlišče, za katerega velja $1 \leq m \leq 2^d$.

Vsako vozlišče B+ drevesa vsebuje m vpisov, pri čemer velja $d \leq m \leq 2^d$.

57

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

Prednosti B+ dreves

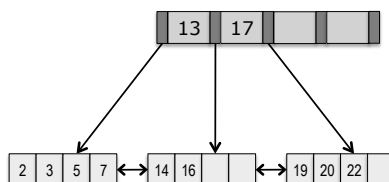
- Uporaba B+ dreves se splača v primerih, ko se podatki velikokrat spreminjajo, obenem pa potrebujemo zaporedno iskanje.
- Prednosti pred uporabo urejenih datotek:
 - Za ceno dodatnega prostora, ki ga porabimo z indeksno datoteko, pridobimo vse prednosti urejene datoteke, obenem pa ne izgubimo na učinkovitosti dodajanja in brisanja.
- Prednosti pred uporabo ISAM-a:
 - Zaradi dodajanja ni potrebno ustvarjati dodatnih oziroma presežnih strani.

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

58


Operacije z B+ drevesi

- Iskanje - enostavno
 - Dodajanje
 - Bisanje
- kompleksno, potrebno zagotavljati uravnoteženost drevesa



Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

59



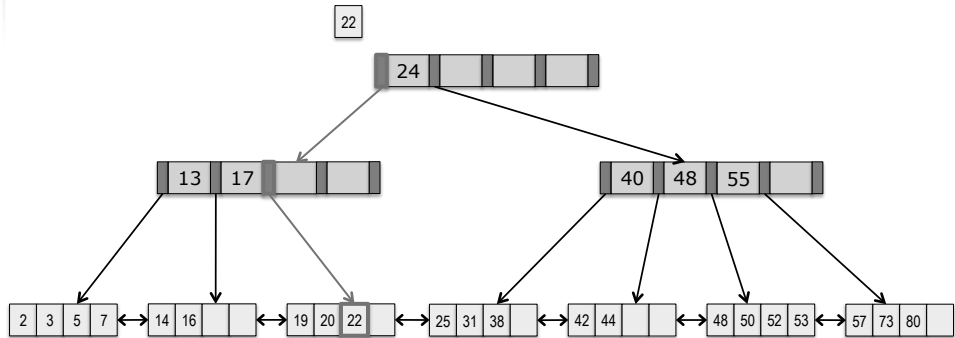
Univerza na Ljubljani
Fakulteta za računalništvo
in informatiko

Iskanje v B+ drevesu

- Poišči element: 22


Postopek: iščemo element k

1. Premakni se v koren
2. Poišči ustrezen kazalec k_i
3. Premakni se na naslednje vozlišče prek kazalca k_i
4. Vrni se na korak 2



60

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije



Univerza na Ljubljani
Fakulteta za računalništvo
in informatiko

Algoritem za iskanje v B+ drevesu

K_1	K_2	...	K_m
P_1	P_2	P_3	P_m

P_{m+1}

```

func find (search key value  $K$ ) returns nodepointer
// Given a search key value, finds its leaf node
return tree_search(root,  $K$ );                                // searches from root
endfunc

func tree_search (nodepointer, search key value  $K$ ) returns nodepointer
// Searches tree for entry
if *nodepointer is a leaf, return nodepointer;
else,
    if  $K < K_1$  then return tree_search( $P_0$ ,  $K$ );
    else,
        if  $K \geq K_m$  then return tree_search( $P_m$ ,  $K$ );           //  $m = \#$  entries
        else,
            find  $i$  such that  $K_i \leq K < K_{i+1}$ ;
            return tree_search( $P_i$ ,  $K$ )
        endfunc
    endfunc
  
```

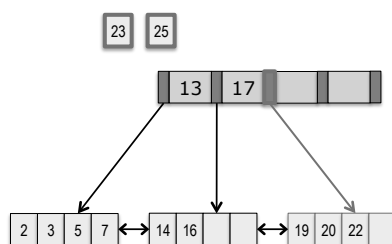
61

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

Dodajanje elementov v B+ drevo

- Dodajanje elementov zahteva razdelitev vozlišča, če je to že polno.

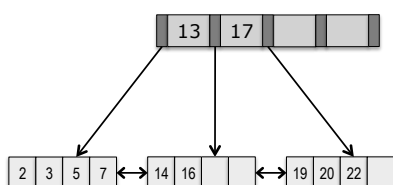
Primer: dodajamo elementa 23 in 25



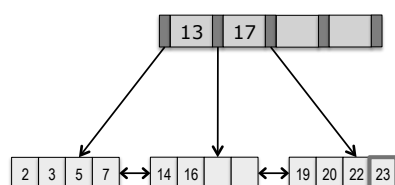
Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

62

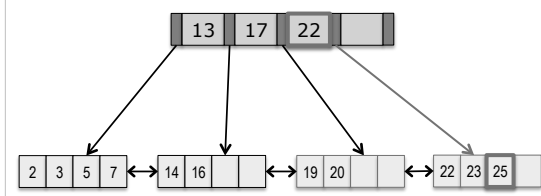
Korak 1: začetno drevo



Korak 2: po dodajanju elementa 23

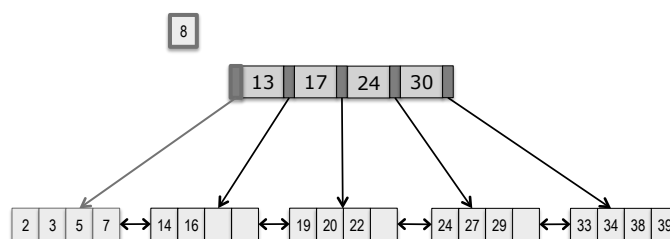


Korak 3: končno drevo – po dodajanju elementa 25



Primer: dodajanje nivoja

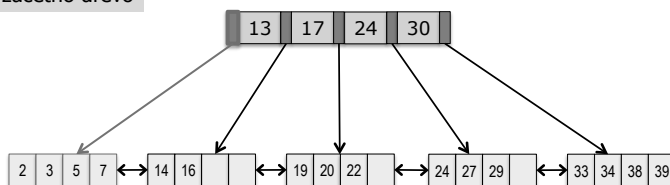
Dodajamo element: 8



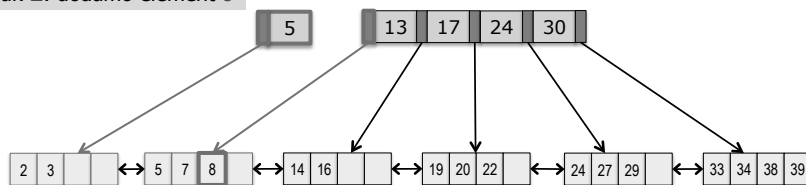
Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

64

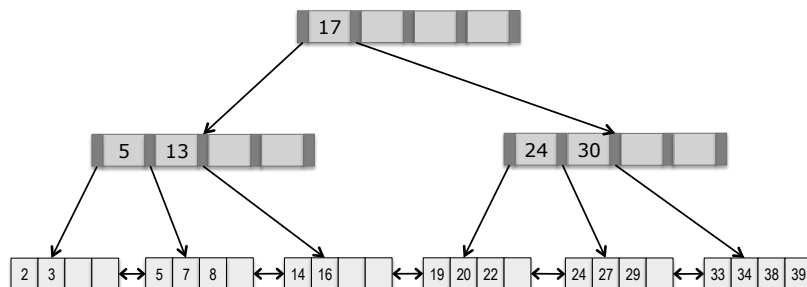
Korak 1: začetno drevo



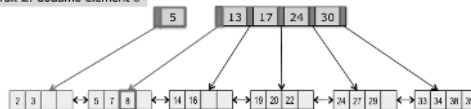
Korak 2: dodamo element 8



Korak 3: končno drevo - po delitvi indeksnega vozlišča



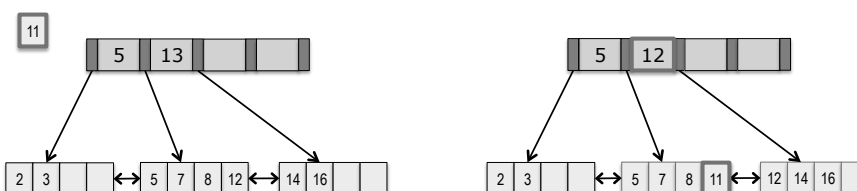
Korak 2: dodamo element 8



Dodajanje elementov v B+ drevo

- Če je vozlišče polno, lahko namesto delitve uporabimo redistribucijo, če je možna.

Primer: dodajamo element 11

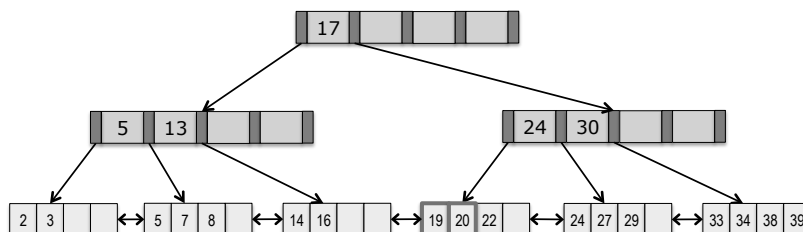


Brisanje elementov iz B+ drevesa

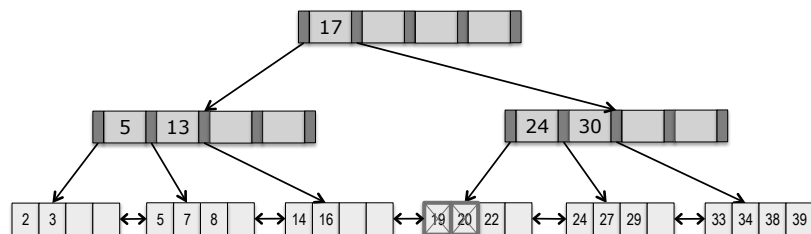
- Postopek:
 - Poiščemo zapis in ga brišemo.
 - Če se zgodi, da zasedenost lista pade pod mejo, moramo redistribuirati podatke s sosednjimi listi ali pa združiti več listov skupaj.
 - Če podatke redistribuiramo med dve vozlišči, moramo ustrezno spremeniti vsebino nad-vozlišča.
 - Če podatke združimo iz dveh vozlišč, moramo vsebino njunih nad-vozlišč spremeniti tako, da brišemo indeksno polje za drugo vozlišče.

Primer: redistribucija pri brisanju

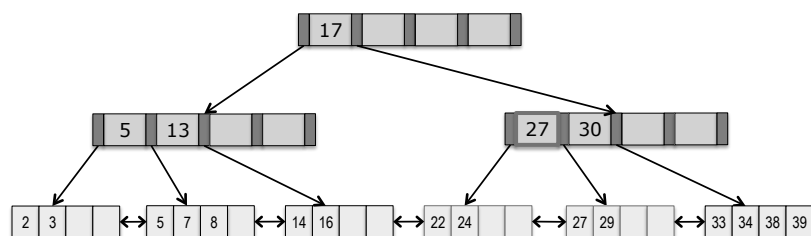
- Brišemo elementa 19 in 20



Korak 1: začetno drevo - brišemo elemente 19 in 20



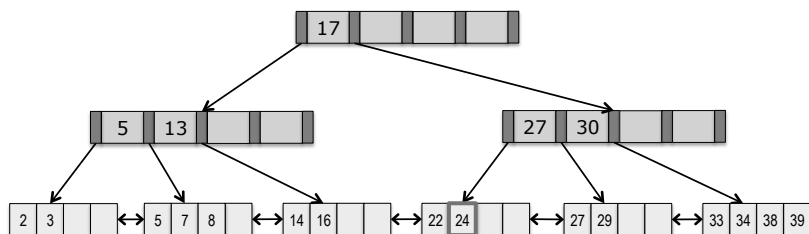
Korak 2: končno drevo – po redistribuciji



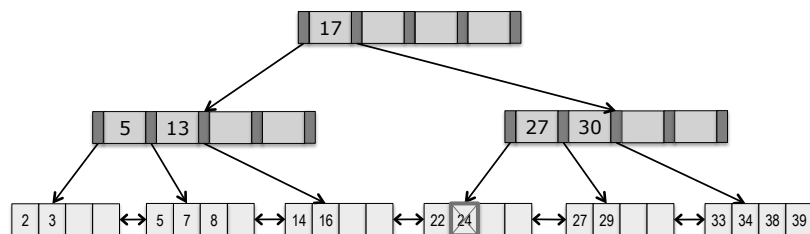
Univerza v Ljubljani
Fakulteta za računalništvo
in informatiko

Primer: združevanje pri brisanju

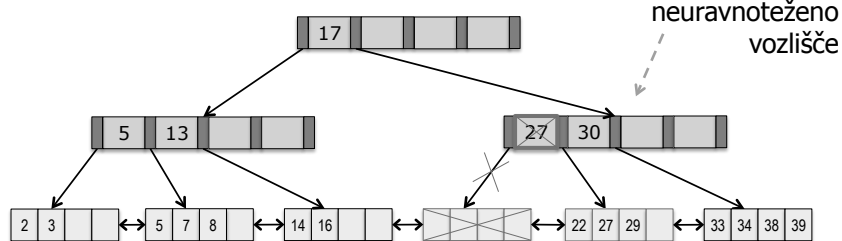
- Brišemo element 24



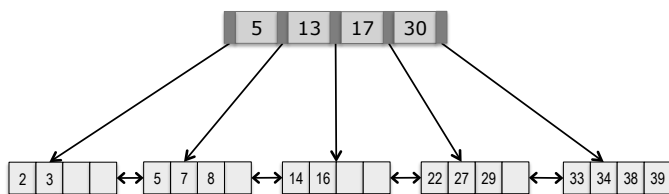
Korak 1: začetno drevo - brišemo element 24



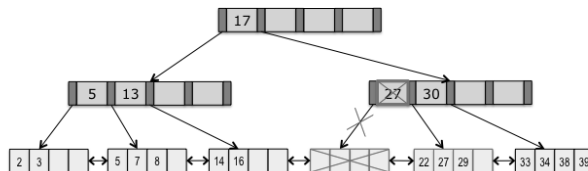
Korak 2: vmesno drevo – po brisanju in združevanju



Korak 3: končno drevo – po združitvi v indeksnem delu



Korak 2: vmesno drevo – po brisanju in združevanju

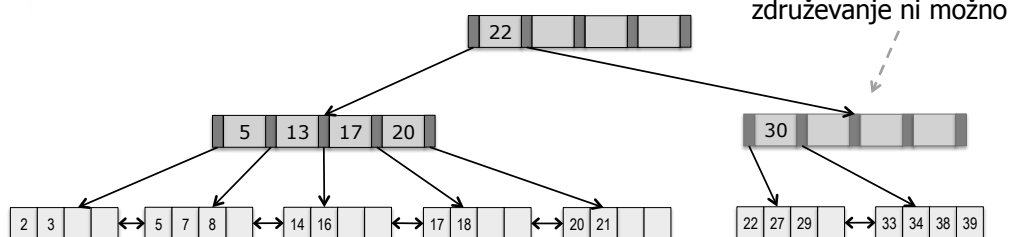


Brisanje elementov iz B+ drevesa

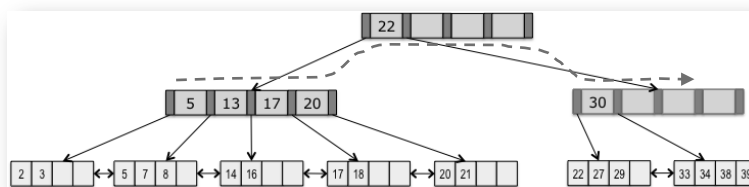
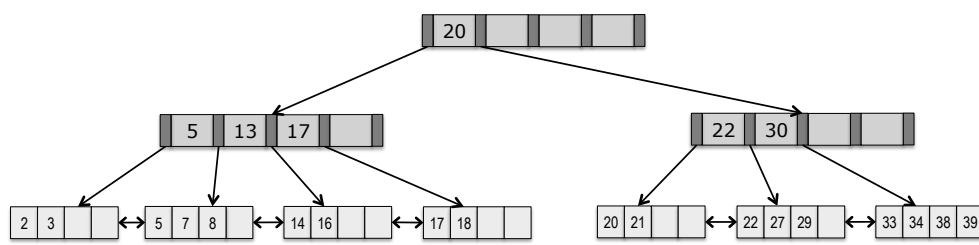
- Pogledali smo, kako deluje:
 - redistribucija v listih,
 - združevanje v listih in
 - združevanje v indeksnem delu
- Ostane:
 - redistribucija v indeksnem delu

Primer: redistribucija v indeksnem delu

- Primer vmesnega neuravnoveženega drevesa



Korak 3: končno drevo – po zamiku indeksnih elementov za eno mesto v desno



Univerza v Ljubljani
Fakulteta za računalništvo
in informatiko

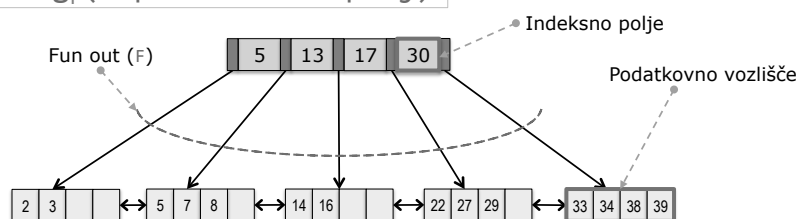
B+ drevesa, ponovitev

	List	Indeksni del
Delitev	Ključ iz lista kopiraj nivo višje	Delitveni ključ prenesi nivo višje in poveži na novo vozlišče
Združevanje	Briši ključ iz indeksnega dela	Prenesi ključ iz višjega nivoja
Redistribucija	Ključ v indeksnem delu nadomesti s ključem v listu	Ključ v indeksnem delu zamakni za eno mesto v desno

Stiskanje ključev

- Višina drevesa je premo sorazmerna številu podatkovnih vozlišč in obratno sorazmerna velikosti indeksnih polj (F).
- Število I/O operacij za branje podatkovnega polja = h .

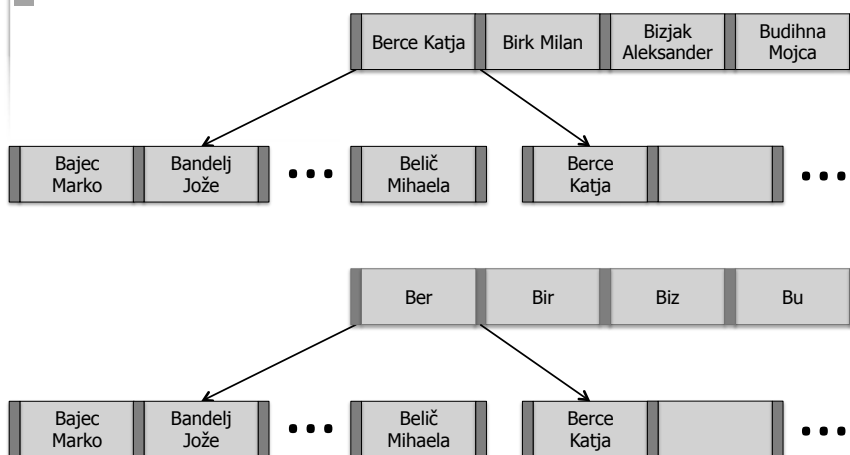
$$h = \log_F(\# \text{ podatkovnih polj})$$



Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

78

Primer



Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

79

Bitni indeks...

staffNo	fName	lName	position	salary	branchNo
SG14	David	Ford	Manager	18000	B003
SG37	Ann	Beech	Assistant	12000	B003
SL21	John	White	Supervisor	30000	B005
SL41	Julie	Lee	Assistant	9000	B005
SF56	Tim	Becker	Manager	32000	B005

Tabela Staff

Manager	Assistant	Supervisor
1	0	0
0	1	0
0	0	1
0	1	0
1	0	0

Bitni indeks po
polju position

B003	B005
1	0
1	0
0	1
0	1
0	1

Bitni indeks po
polju branchNo

Bitni indeks

- Prednosti bitnega indeksa v primerjavi z B+ drevesi:
 - Kompaktnost
 - Večja učinkovitost pri iskanju po več predikatih (če za njih obstajajo bitni indeksi)

Primer:

```

SELECT staffNo, salary
FROM Staff
WHERE position = 'Supervisor' AND branchNo = 'B003'

```

Za rezultat pomnožimo bitna vektorja
za vrednost Supervisor in B003.

Supervisor
0
0
1
0
0

B003
1
1
0
0
0

Stični indeksi...

- Stični indeksi (*Join index*) podobno kot bitni indeksi popularni na področju podatkovnih skladišč.
- Koncept:
 - Uporabljajo se za tabele, ki imajo skupna polja (npr. tuji ključi).
 - Indeks vsebuje za vsako vrednost ključa kazalec na vse tabele, ki so indeksirane z istim stičnim indeksom.
 - Način sortiranja indeksa je poljuben.

Primer stičnega indeksa

Tabela Branch

rowID	branchNo	street	city	postcode
20001	B005	22 Deer Rd	London	SW1 4EH
20002	B007	16 Argyll St	Aberdeen	AB2 3SU
20003	B003	163 Main St	Glasgow	G11 9QX
20004	B004	32 Manse Rd	Bristol	BS99 1NZ
20005

branch rowID	property rowID	city
20001	30002	London
20002	30001	Aberdeen
20003	30003	Glasgow
20003	30004	Glasgow
...

Tabela Property

rowID	propertyNo	street	city	postcode	...
30001	PA14	16 Holhead	Aberdeen	AB2 3SU	
30002	PL94	6 Argyll St	London	SW1 4EH	
30003	PG4	6 Lawrence St	Glasgow	G11 9QX	
30004	PG36	2 Manor Rd	Glasgow	G11 9QX	
30005

Koliko nepremičnin je v mestih, kjer so locirane organizacijske enote nepremičninske agencije?


Gruče...

- Nekateri SUPB nudijo združevanje relacij v gruče (*clustered relations*)
- Gruča označuje skupino relacij, ki so fizično shranjene skupaj, ker imajo nekatere stolpce enake in se pogosto uporabljajo skupaj.
- Gruče izboljšajo čas dostopa do podatkov na disku.
- Stolpce, ki so za relacije v gruči skupni, imenujemo ključ gruče (*cluster key*).

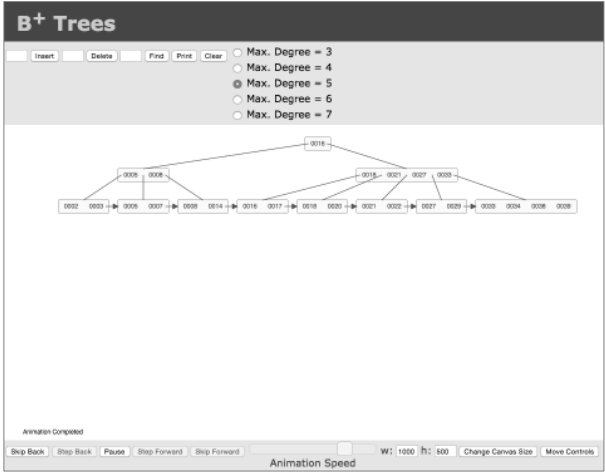
Primer gruče

street	city	postcode	branchNo	staffNo	fName	lName	position	...
22 Deer Rd	London	SW1 4EH	B005	SL21	John	White	Manager	...
				SL41	Julie	Lee	Assistant	...
163 Main St	Glasgow	G11 9QX	B003	SG37	Ann	Beech	Assistant	...
				SG14	David	Ford	Supervisor	...
				SG5	Susan	Brand	Manager	...

Diagram illustrating a clustered relation. The table is divided into two logical tables: **Tabela Branch** (columns: street, city, postcode, branchNo) and **Tabela Staff** (columns: staffNo, fName, lName, position, ...). The **branchNo** column is identified as the **Ključ gruče** (cluster key).


 Univerza v Ljubljani
 Fakulteta za računalništvo
 in informatiko


Vizualizacija B+ dreves



<https://www.cs.usfca.edu/~galles/visualization/BPlusTree.html>

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

86


 Univerza v Ljubljani
 Fakulteta za računalništvo
 in informatiko

Ponovitev

- Kje so shranjene tabele podatkovne baze?
- Kakšna je najmanjša enota, ki jo preberemo iz diska?
- Koliko zapisov ima ena stran (blok)?
- Kaj vpliva bolj na učinkovitost poizvedovanja, višina ali širina B+ drevesa?
- Ali imamo lahko več primarni indeksov?
- Kdaj uporabljamo bitne indekse?

Predmet: PB, Modul: Organizacija datotek in indeksi, Gradivo: v.2015, ©UL FRI, Laboratorij za podatkovne tehnologije

87