Razpršilni indeksi

Iztok Savnik, FAMNIT

Prosojnice & učbenik

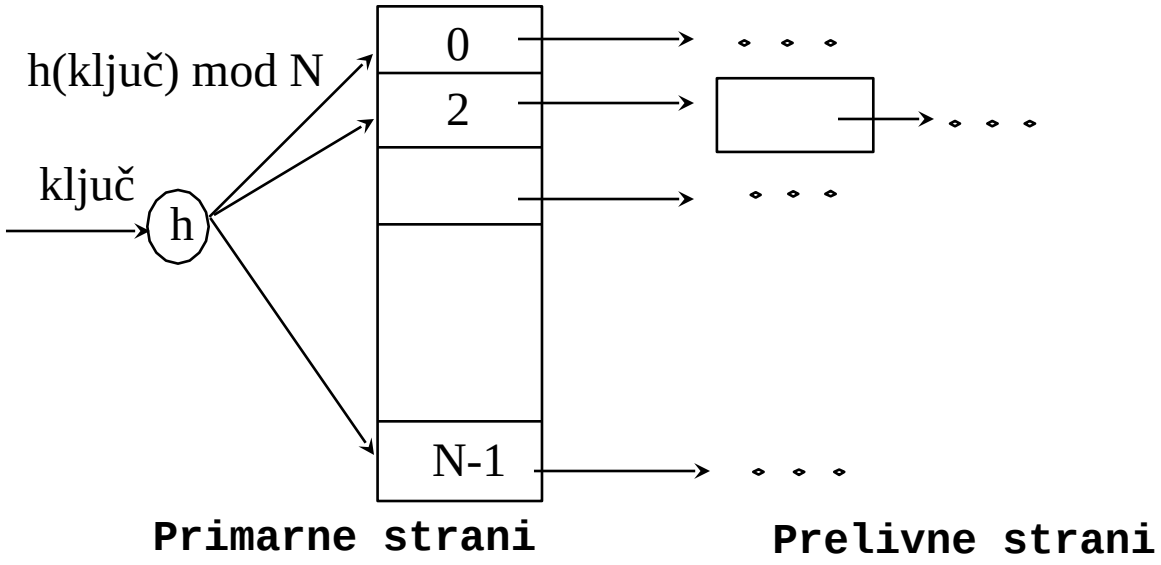
- Učbenik:
  - Raghu Ramakrishnan, Johannes Gehrke, *Database Management Systems, McGraw-Hill, 3rd ed., 2007.*
- *Prosojnice:*
  - *From „Cow Book“: R.Ramakrishnan, http://pages.cs.wisc.edu/~dbbook/*

Uvod

- Za vsak indeks velja, da imamo 3 alternative za podatkovni vpis k*:
  1) Podatkovni zapis z vrednostjo ključa **k.**
  2) <**k**, rid pod. zapisa z iskalnim ključem **k**>
  3) <**k**, seznam rid-jev pod. zapisov z isk. ključem **k**>
  – Izbira alternative je ortogonalna izbiri indeksne tehnike
- Razpršilni indeksi so najboljši za iskanje po enakosti. Ne podpirajo iskanje po področju.
- Obstajajo statični in dinamični razpršilni indeksi; kompromisi so podobni ISAM vs. B+ drevesa.

Statični razpršilni indeks

- # primarne strani so fiksne, zasežene sekvenčno ter niso nikoli sproščene.
- **h**(*k*) mod N = skupek h kateremu pripada podatkovni vpis s ključem *k*. (N = # skupkov)



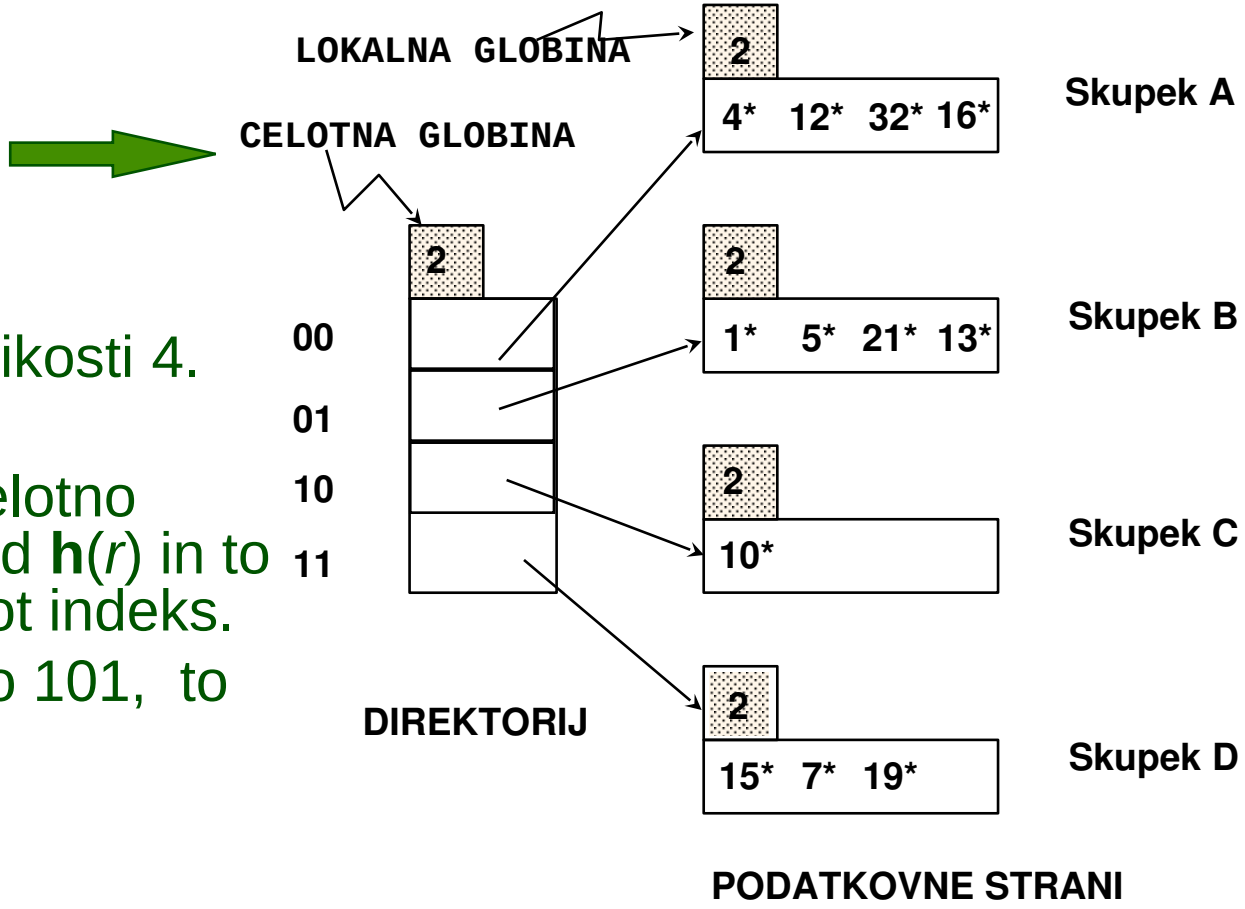
Statični razpršilni indeks

- Skupki vsebujejo *podatkovne vpise*.
- Razpršilna fn se uporabi na *iskalnem ključu.* Vrednosti se morajo *razpršiti* na interval 0 ... N-1.
  - **h**(*k*) = (a * *k* + b) običajno deluje dobro.
  - a in b so konstante; veliko je znano o tem kako umeriti **h**.
- Razvije se lahko dolga veriga prelivnih strani kar poslabša učinkovitost podatkovne strukture.
  - *Dinamični razpršilni indeks* reši problem.
  - Imamo razširljive in linearne razpršilne indekse.

Razširljiv razpršilni indeks

- Dinamični razpršilni indeks
  - *Situacija*: Skupek (primarna stran) se napolni. Zakaj nebi podvojili # skupkov?
    - Branje in pisanje strani skupka je potratno.
- *Ideja*: *direktorij kazalcev na skupke*
  - Podvojimo # skupkov (strani) s podvojitvijo direktorija in razcepimo samo skupek (stran), ki je napolnjen.
  - Direktorij je veliko manjši kot celotna datoteka zato se podvojitev izvrši hitro. *Ni prelivnih strani* !
  - Trik je v prilagoditvi razpršilne funkcije!

Primer

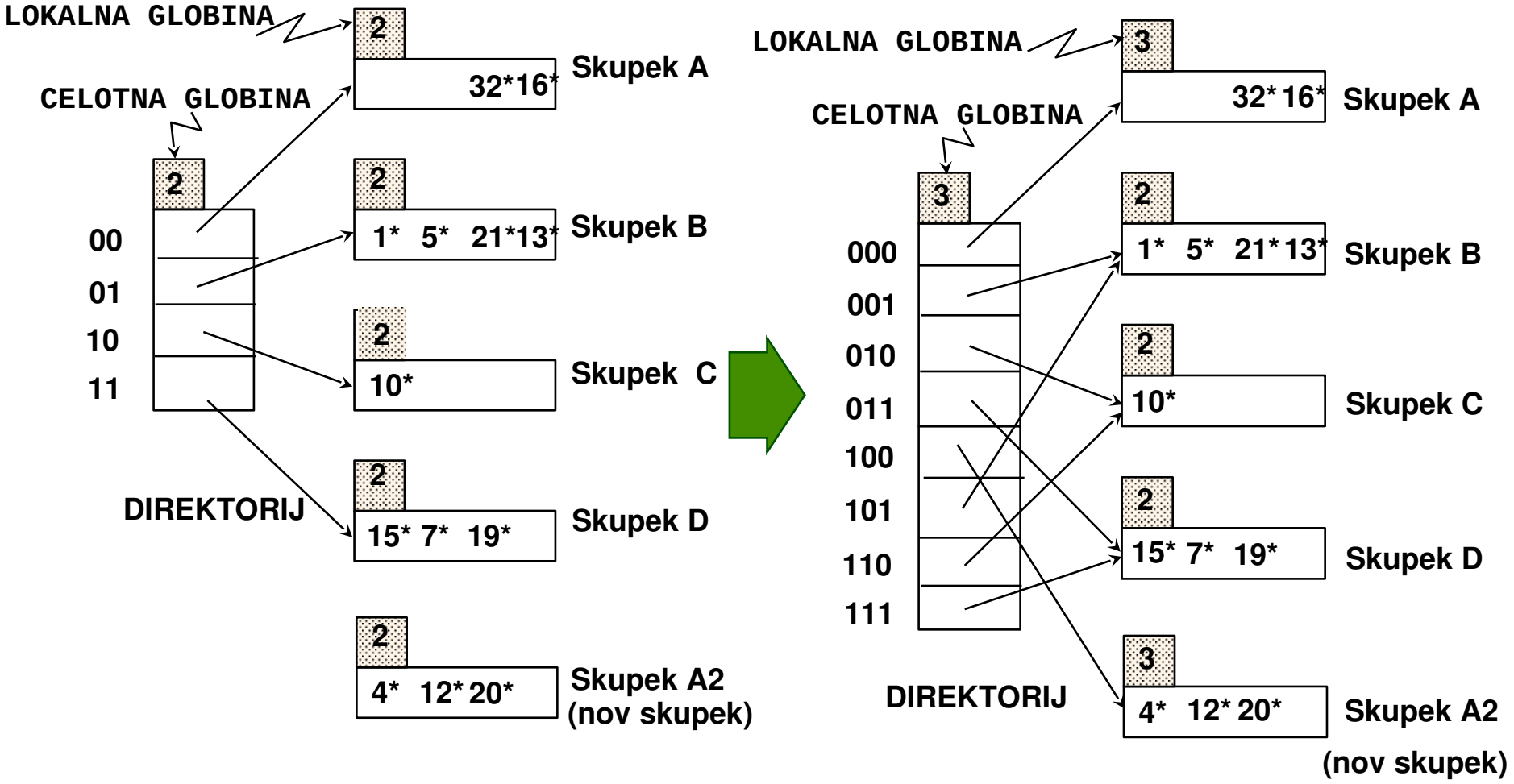


- Direktorij je polje velikosti 4.
- Iskanje skupka za *r:*
  - Vzemi zadnjo `celotno globino' # bitov od **h**(*r*) in to število uporabi kot indeks.
    - **h**(*r*) = 5 = binarno 101, to je skupek 01.

❖ Dodajanje: Če je skupek poln potem ga razcepi ter zaseži novo stran in porazdeli vpise.

❖ *Če je potrebno* se podvoji direktorij. Videli bomo, da razcep strani velikokrat ne zahteva podvojitev; to vidim tako, da primerjamo celotno in lokalno globino skupka.

Vstavi **h**(r)=20 (podvojitev)

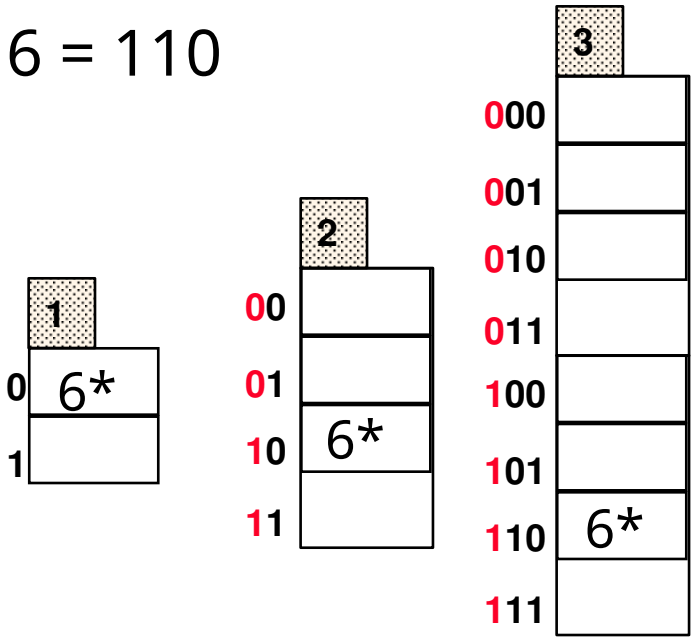


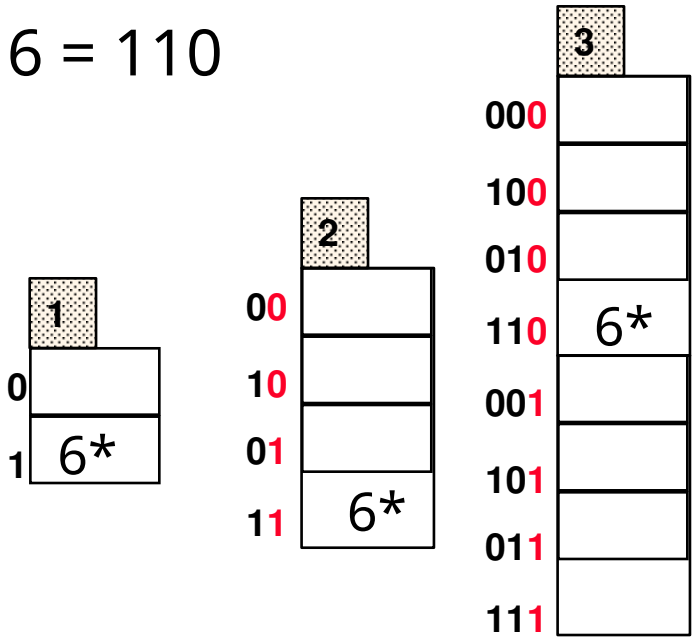
Komentarji

- 20 = binarno 10100. Zadnja **2** bita (00) nam povesta, da *r* pripada A ali A2; **3.** bit pove v kateri.
  - *Celotna globina direktorija*: Max # bitov potrebnih za določitev skupka h kateremu pripada vpis.
  - *Lokalna globina skupka*: # bitov potrebnih za določitev ali vpis pripada danemu skupku.
- Kdaj razcep skupka povzroči podvojevanje direktorija?
  - Pred vstavljanjem: *lokalna globina* skupka = *celotna globina*.
  - Vstavljanje povzroči: *lokalna globina* postane > *celotna globina.*
  - Direktorij se podvoji s kopiranjem.
  - V direktorij se vpiše kazalec na novo stran.
  - Uporaba najmanj pomembnih bitov omogoča učinkovito podvojevanje s kopiranjem direktorija!

Podvojevanje direktorija

Zakaj uporabiti najmanj pomembne bite v direktoriju?
Možna podvojitev direktorija s kopiranjem!





Najmanj pomembni biti vs. Najbolj pomembni biti

Komentarji o razširljivem indeksu

- Če direktorij gre v dinamični spomin potem so poizvedbe na osnovi enakosti izvedene z branjem enega bloka, sicer z dvemi.
  - 100MB datoteka, 100 zlogov/zapis, 4K stran. 1,000,000 zapisov (podatkovnih vpisov), direktorij ima 25,000 elementov, velika verjetnost da bo cel direktorij v dinamičnem spominu.
  - Direktorij raste s podvojitvijo. Če je imamo neenakomerno porazdelitev vrednosti razpršilne funkcije potem direktorij zelo hitro raste.
- Brisanje: Če izbris podatkovnega vpisa izprazni stran, jo lahko združimo z ’razcepljeno sliko’. Če vsak element kaže na isto stran kot njegova razcepljena slika potem lahko razpolovimo direktorij.

Linearni razpršilni indeks

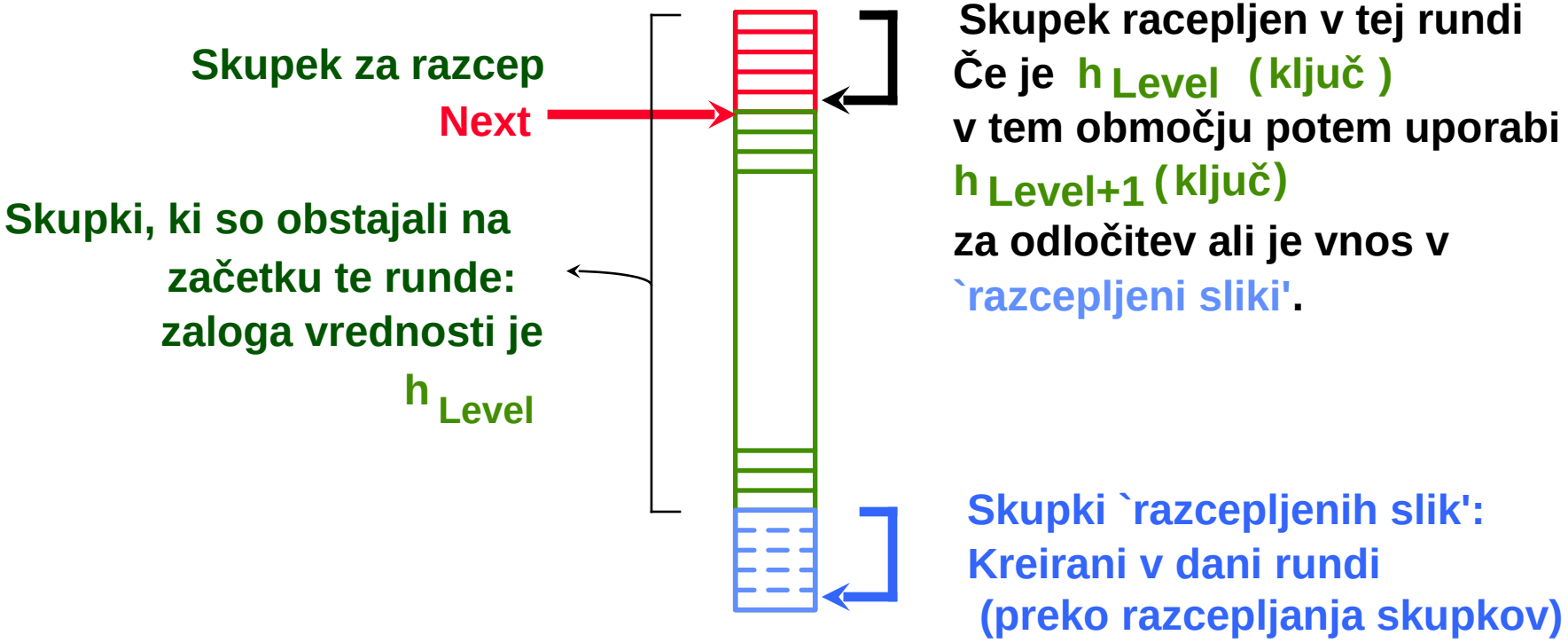
- Še ena shema z dinamičnim razprševanjem. Alternativa razširljivem indeksu.
- Linearni razpršilni indeks reši problem z dolgimi prelivnimi vrstami brez uporabe direktorija.
- *Ideja*: Uporabi družino razpršilnih funkcij $\mathbf{h}_0$, $\mathbf{h}_1$, $\mathbf{h}_2$, ...
  - $\mathbf{h}_i(key) = \mathbf{h}(key) \bmod (2^i N)$; N = začetno # skupkov
  - **h** je neka razpršilna funkcija (zaloga vrednosti *ni* [0,N-1])
  - Če $N = 2^{d0}$, za nek *d0*, $\mathbf{h}_i$ aplicira **h** in gleda zadnjih *di* bitov, kjer *di* = *d0* + *i*. Vrednost *i* imenujemo *Level*.
  - $\mathbf{h}_{i+1}$ podvoji zalogo vrednosti $\mathbf{h}_i$ (podobno podvojevanju direktorija)

Linearni razpršilni indeks

- Direktoriju se izognemo z uporabo prelivnih strani in s cikličnim razceplanjem skupkov.
  - Razceplanje potek v `rundah'. Runda R se konča, ko so vsi začetni skupki (N_R) razcepljeni.
    - Skupki 0 do *Next-1* so bili razcepljeni
    - *Skupki Next* do N_R morajo še biti razcepljeni
  - Trenutna številka runde je *Level*.
  - **Iskanje:** Da bi poiskali skupek za *r, poišči* $\mathbf{h}_{Level}(r)$*:*
    - Če $\mathbf{h}_{Level}(r)$ je v področju [*Next*,N_R], smo našli ...
    - Sicer, r lahko pripada skupku $\mathbf{h}_{Level}(r)$ ali skupku $\mathbf{h}_{Level}(r) + N_R$. Aplicirati je potrebno $\mathbf{h}_{Level+1}(r)$, da bi izvedeli.

Pregled LH datoteke

- V sredini ene runde.

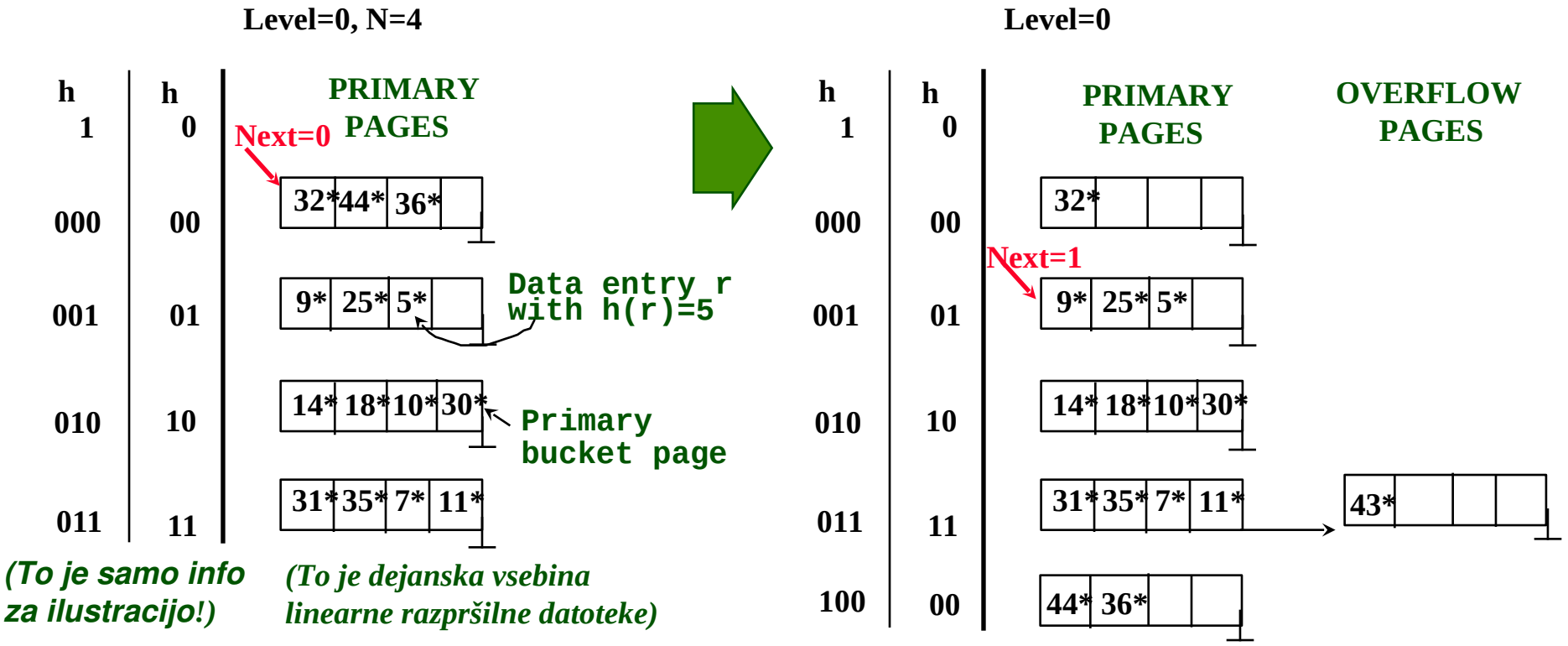


Linearno razprševanje

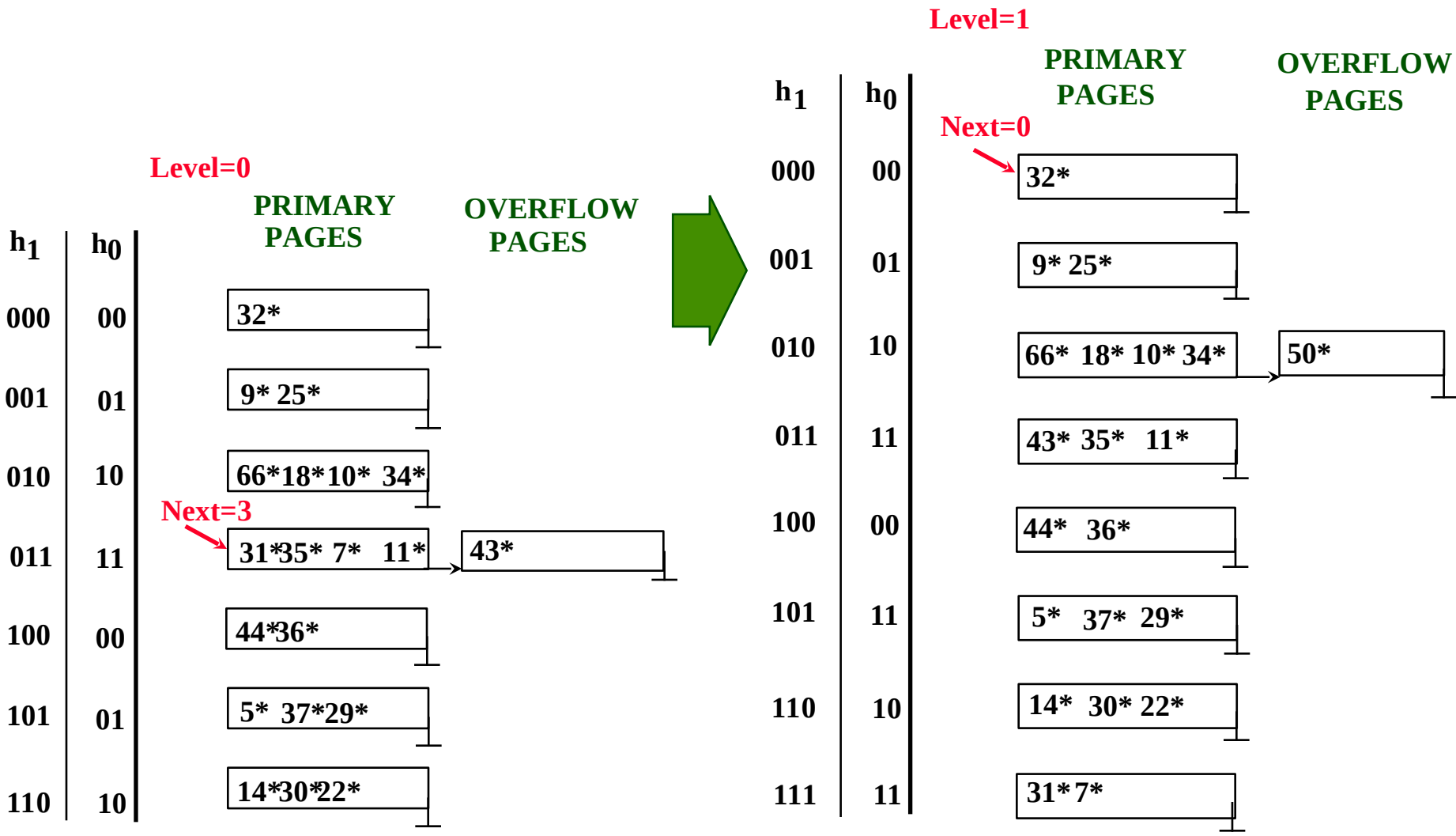
- **Vstavi**: Poišči skupek z uporabo $\mathbf{h}_{Level}$ / $\mathbf{h}_{Level+1}$:
  - Če je skupek poln:
    - Dodaj prelivno stran in vstavi podatkovni vpis.
    - (*Mogoče*) Razcepi skupek *Next* in povečaj *Next* za 1.
- Izberemo lahko katerikoli kriterij za proženje razcepa.
- Ker so skupki razcepljeni ciklično (round-robin) se ne morejo razviti dolge prelivne verige!
- Podvojevanje direktorija v razširljivem razprševanju je podobno; preklop med razpršilnimi funkcijami je *impliciten* s povečevanjem # uporabljenih bitov.

Primer linearnega razp. indeksa

- Pri razcepu, je $\mathbf{h}_{Level+1}$ uporabljena za prerazdelitev vnosov.



Primer: Konec runde



LH je varianta EH

- Shemi sta precej podobni
  - Začni z EH indeksom, ki ima direktorij z *N* elementi.
  - Uporabi prelivne strani in razcepi skupke ciklično
  - Najprej razcepi skupek 0
    - Ustreza “podvojevanju direktorija” v EH.
      - Elementi <1,*N*+1>, <2,*N*+2>, ... se kreirajo “po vrsti”.
    - Ko se skupek razcepi, kreiraj element *N*+1, itd.
- Direktorij se podvojuje postopoma. Skupki se kreirajo fizično zaporedno na disku. Če so tudi alocirani v sekvenci (da je iskanje i-tega skupka enostavno), potem dejansko ne potrebujemo direktorija. In dobimo LH!

Povzetek

- Razpršilni indeksi: dobri za iskanje z enačajem in se jih ne uporablja za iskanje po področju.
- Statični razpršilni indeksi lahko vodijo do dolgih prelivnih verig.
- Razširljiv razpršilni indeks nima prelivnih strani razen v primeru duplikatov.
  - *Duplikati zahtevajo uporabo prelivnih strani!*
- Ko se skupek zapolni se podvoji direktoriij.
  - Problemi v primeru neenakomerne porazdelitve vrednosti razpršilnih funkcij.