



## Rasuta organizacija datoteke

---

*Struktura, formiranje, traženje,  
obrada, ažuriranje, primena i  
ocena*

# Sadržaj

---

- Rasute organizacije datoteka
- Direktna i relativna organizacija datoteke
- Statička rasuta organizacija datoteke
- Dinamička rasuta organizacija datoteke

# Rasute organizacije datoteka

---

- **Rasuta organizacija datoteke**

- često se, u najširem značenju, naziva i direktnom jer se

- slogu ili grupi slogova pristupa direktno na osnovu poznavanja adrese memorijske lokacije u kojoj su smešteni
    - adresa lokacije dobija transformacijom vrednosti identifikatora sloga u adresu

- **identifikator** – skup obeležja čije vrednosti jednoznačno određuju slogove datoteke

- identifikator može a ne mora pripadati skupu obeležja tipa sloga datoteke – **interni** ili **eksterni**
    - **interni identifikator** - po pravilu, primarni ključ datoteke
    - **eksterni identifikator** - vrednosti identifikatora pridružuju se svakom slogu eksterno, van konteksta sadržaja datoteke

# Rasute organizacije datoteka

---

- **Rasuta organizacija datoteke**

- transformacija vrednosti identifikatora u adresu

- $h: \text{dom}(K) \rightarrow A$

- $K$  - domen identifikatora

- $A$  - skup adresa lokacija memorijskog prostora datoteke

- vrste transformacija vrednosti identifikatora u adresu

- **deterministička**

- funkcija  $h$  je injektivna

- svakoj vrednosti identifikatora odgovara jedna adresa

- svakoj adresi odgovara najviše jedna vrednost identifikatora

- **probabilistička**

- svakoj vrednosti identifikatora odgovara jedna adresa

- jednoj adresi može odgovarati više rezultata transformacije

- metoda za generisanje pseudoslučajnih brojeva

# Rasute organizacije datoteka

---

- **Rasuta organizacija datoteke**

- fizička struktura podataka ne sadrži informaciju o vezama između slogova logičke strukture datoteke
  - u dve fizički susedne lokacije mogu a ne moraju biti memorisani logički susedni slogovi
  - slogovi su, na slučajan način, rasuti po memorijskom prostoru datoteke

- **Baket**

- tradicionalan naziv za blok kod rasutih datoteka
- faktor baketiranja  $b$  ( $b \geq 1$ ) = faktor blokiranja  $f$
- transformacijom  $h$  vrednost identifikatora pretvara se u adresu baketa

# Rasute organizacije datoteka

## • Primer - rasuta datoteka - $D_{ras}$

- slogova  $N = 10$
- faktor baketiranja  
 $b = 3$
- datoteci je dodeljeno  
 $B = 5$  baketa
- identifikator
  - primarni ključ datoteke
- transformacija
  - $h(k_i) = 1 + k(S_i) \pmod{B}$   
 $i \in \{1, \dots, N\}$
  - rezultat : relativna  
adresa baketa  
iz skupa  $\{1, 2, 3, 4, 5\}$

1

15	$p(S_4)$	25	$p(S_7)$		

2


3

07	$p(S_2)$	27	$p(S_9)$		

4

03	$p(S_3)$	23	$p(S_8)$	13	$p(S_{10})$

5

34	$p(S_1)$	19	$p(S_5)$	29	$p(S_6)$

# Rasute organizacije datoteka

---

- **Vrste rasutih datoteka**

- s obzirom na način alokacije memorijskog prostora

- **statičke**

- veličina adresnog prostora određuje se i kompletno alocira unapred, statički
    - ne može se menjati tokom eksploatacije

- **dinamičke**

- veličina dodeljenog adresnog prostora menja se tokom ažuriranja, saglasno potrebama

- istorijski gledano, statičke rasute datoteke nastale su znatno ranije od dinamičkih

# Rasute organizacije datoteka

- **Opšti postupak formiranja statičke rasute**

- statičkoj rasutoj datoteci, u postupku formiranja, dodeljuje se  $Q = bB$  lokacija,  $N \leq Q$ 
  - nakon formiranja,  $Q$  se više ne može menjati
- **faktor popunjenosti**
  - $q = N / Q, \quad 0 < q \leq 1$
- redosled smeštanja slogova u datoteku je nevažan
  - i u slučaju determinističke i probabilističke transformacije
- slogovi se upisuju saglasno hronološkom redosledu nastanka
- upisu sloga prethodi neuspešno traženje, na osnovu obavljene transformacije identifikatora u adresu
- slog se smešta u baket sa izračunatom adresom



# Sadržaj

---

- Rasute organizacije datoteka
- Direktna i relativna organizacija datoteke
- Statička rasuta organizacija datoteke
- Dinamička rasuta organizacija datoteke

# Direktna i relativna organizacija datoteke



- **Direktna organizacija datoteke**

- eksterni identifikator, trivijalna deterministička transformacija  $h: A \rightarrow A$
- vrednost identifikatora je, u isto vreme, i adresa baketa:  $A_i = k_i$ 
  - najjednostavniji vid determinističke transformacije
- preslikavanje veza između sadržaja slogova i adresa ne pripada direktnoj organizaciji datoteke
- vrste direktnih datoteka, s obzirom na vrstu upotrebljavanih adresa
  - vrednosti identifikatora su mašinske adrese baketa
  - vrednosti identifikatora su relativne adrese baketa

# Direktna i relativna organizacija datoteke

---

- **Direktna datoteka s mašinskim adresama**
  - direktna datoteka u užem smislu reči
  - koristi se adresa baketa na disku, oblika  $(u, c, t, s)$
  - ima samo istorijski značaj
  - nedostaci
    - čvrsta povezanost programa sa fizičkim karakteristikama memorijskog uređaja
      - program zavisi od karakteristika eksternog memorijskog uređaja i fizički alociranog prostora na uređaju
      - ne može se koristiti putem programskih jezika treće generacije
    - odsustvo logičke veze između vrednosti identifikatora i sadržaja sloga
      - održavanje veze između sadržaja sloga i adrese je zadatak krajnjeg korisnika, ili u boljem slučaju, samog programa
      - problemi u slučaju reorganizacije ili brisanja pa upisa sloga

# Direktna i relativna organizacija datoteke

---

- **Direktna datoteka s relativnim adresama**
  - relativna organizacija datoteke
  - korišćenje relativnih adresa slogova
    - lokacije alociranog memorijskog prostora numerišu se rednim brojevima od 1 do  $Q$  (alternativno, od 0 do  $Q - 1$ )
    - redni broj sloga u memorijskom prostoru predstavlja eksterni identifikator sloga
    - rešava se problem čvrste povezanosti slogova datoteke sa fizičkim karakteristikama memorijskog uređaja
  - odsustvo logičke veze između vrednosti identifikatora i sadržaja sloga – glavni nedostatak
  - **relativna metoda pristupa**
    - deo sistema za upravljanje podacima koji obavlja transformaciju relativne adrese u mašinsku i ostale operacije

# Direktna i relativna organizacija datoteke



- **Relativna metoda pristupa**

- pruža programu usluge na nivou bloka (baketa)
  - ne vrši blokiranje i rastavljanje blokova na slogove
  - prihvata samo vrednost faktora blokiranja jednaku 1
  - sa stanovišta relativne metode pristupa:

**1 blok = 1 slog**

- aktivnosti blokiranja i rastavljanja blokova na slogove moraju se realizovati u okviru samog programa
  - ukoliko za tim postoji realna potreba
  - čime se može postići veća efikasnost iskorišćenja memorijskog prostora

# Direktna i relativna organizacija datoteke

---



- **Relativna metoda pristupa**

- sistemi za upravljanje datotekama mainframe računarskih sistema
  - kompletna podrška – usluge na nivou bloka
- savremeni SUBP
  - moguća podrška, ali nije tipično u upotrebi
- standardne biblioteke programskih jezika
  - podržana transformacija relativne adrese u apsolutnu
  - moguća kompletna podrška – usluge na nivou sloga, ili
  - podrška samo direktnog pristupa lokacijama datoteke
    - rutinske operacije rada sa slogovima datoteke rešavaju se kroz aplikativni program
    - koriste se pozivi rutina koje obezbeđuju rad sa datotekom kao bajt ili znak-orijentisanom strukturom

# Direktna i relativna organizacija datoteke



- **Relativna metoda pristupa**

- standardne biblioteke programskih jezika, za razliku od sistema za upravljanje datotekama mainframe računarskih sistema
  - ne prave razliku između serijske, sekvencijalne, relativne, spregnute ili rasute osnovne organizacije datoteke
  - pružaju usluge na nivou sloga ili čak samo na nivou bajt-orijentisane strukture
  - za svaku datoteku, podržavaju metode direktnog i sekvencijalnog pristupa
  - za upravljanje svakom datotekom podržavaju iste systemske pozive i odgovarajuće bibliotečne funkcije
    - otvaranje, pozicioniranje, čitanje, upisivanje i zatvaranje
  - aplikativnom programu ostaje da obezbedi svu funkcionalnost koju zahteva određena vrsta organizacije

# Direktna i relativna organizacija datoteke

---



- **Formiranje relativne datoteke**
  - najčešće u posebnom postupku
  - na osnovu vodeće serijske ili neke druge vrste datoteke
    - sukcesivno učitavanje slogova vodeće datoteke
    - pridruživanje vrednosti identifikatora – relativne adrese slogu
      - najčešće automatski, kroz aplikativni program
    - smeštanje sloga u lokaciju s pridruženom relativnom adresom



# Direktna i relativna organizacija datoteke



- **Traženje sloga u relativnoj datoteci**

- traženje i logički narednog i slučajno odabranog sloga
  - zadavanjem relativne adrese lokacije metodi pristupa
    - primena trivijalne metode transformacije identifikatora u adresu
  - na osnovu adrese, metoda pristupa prenosi blok u radnu zonu programa
- uspešno traženje
  - ako i samo ako traženi slog postoji u prenetom bloku
  - broj pristupa za uspešno ili neuspešno traženje  $R_u$ ,  $R_n$ :

$$R_u = 1, R_n = 1$$

# Direktna i relativna organizacija datoteke



- **Ažuriranje relativne datoteke**

- u režimu direktne obrade

- **upis novog sloga**

- novom slogu pridruži se vrednost identifikatora
    - slog se upisuje u datoteku, ako postoji slobodna lokacija u bloku
    - pre upisa, može se izvršiti provera da li slog sa istom vrednošću ključa već postoji u istom bloku

- **brisanje**

- realizuje se kao logičko
    - pročita se sadržaj adresiranog bloka
    - nakon provere vrednosti ključa, izmena sadržaja statusnog polja sloga
    - modifikovani sadržaj bloka ponovo se upisuje u datoteku

# Direktna i relativna organizacija datoteke

---



- **Ocena karakteristika relativne datoteke**
  - traženje slučajno odabranog sloga
    - najefikasnije moguće
    - uz pretpostavku da korisnik brine o vezi između sadržaja sloga i relativne adrese
  - traženje logički narednog sloga
    - značajno efikasnije nego u slučaju serijske datoteke
    - značajno manje efikasno nego u slučaju sekvencijalne datoteke
  - traženje i logički narednog i slučajno odabranog sloga
    - potreban uvek samo jedan pristup datoteci
  - uvođenje relativne adrese lokacije kao identifikatora
    - rešava problem čvrste povezanosti slogova datoteke sa karakteristikama memorijskog uređaja

# Direktna i relativna organizacija datoteke



- **Ocena karakteristika relativne datoteke**

- nepostojanje veze između vrednosti identifikatora i sadržaja sloga u relativnoj organizaciji datoteke
  - prepreka za šire korišćenje relativnih datoteka u praksi
- moguća primena relativne metode pristupa
  - osnov za izgradnju spregnute datoteke
    - relativna adresa lokacije logički narednog sloga smešta se u polje pokazivača tekućeg sloga
  - osnov za izgradnju rasutih datoteka sa probabilističkom transformacijom ključa u adresu
    - probabilistička transformacija brine o vezi između vrednosti identifikatora i relativne adrese
  - osnov za izgradnju indeksnih datoteka
    - formira se pomoćna struktura podataka koja obezbeđuje memorisanje veze između sadržaja sloga i relativne adrese

# Sadržaj

---

- Rasute organizacije datoteka
- Direktna i relativna organizacija datoteke
- Statička rasuta organizacija datoteke
- Dinamička rasuta organizacija datoteke

# Statička rasuta organizacija datoteke

- **Opšte karakteristike rasute datoteke sa probabilističkom transformacijom**
  - ključ
    - često uzima vrednosti iz veoma velikog opsega mogućih vrednosti
      - ograničenog samo brojem pozicija  $p$  i brojem dozvoljenih vrednosti  $v$  koju svaka pozicija može imati
    - može uzimati jednu od  $v^p$  ili  $v^p - 1$  vrednosti
  - veličina adresnog prostora dodeljenog datoteci
    - broj lokacija  $Q = Bb$ 
      - $B$  - broj baketa,  $b$  - broj lokacija u baketu
    - $N$  - broj aktuelnih slogova u datoteci
      - po pravilu, mnogo manji od broja mogućnih vrednosti ključa

$$v^p \gg Q \geq N$$

- **Metode probabilističke transformacije**
  - uvode se kako bi se prevazišli nedostaci do kojih dovodi deterministička transformacija vrednosti ključa u adresu
- Ciljevi:
  - što ravnomernija raspodela slogova u adresnom prostoru
  - pseudoslučajna transformacija vrednosti ključa u adresu
  - pravilno dizajniranje potrebnog adresnog prostora

# Statička rasuta organizacija datoteke

- **Metode probabilističke transformacije**

- koraci probabilističke transformacije vrednosti ključa

$$h: \text{dom}(K) \rightarrow A$$

- (1) pretvaranje nenumeričke u numeričku vrednost ključa:

$$k(S) \in \{0, 1, \dots, v^p - 1\}$$

- očekivano, osnova brojnog sistema:  $v = 10$
- $p$  – broj cifara numeričke vrednosti ključa

- (2) pretvaranje numeričke vrednosti ključa  $k(S)$  u pseudoslučajan broj  $T(k(S))$ , ili skraćeno  $T$

$$T \in \{0, 1, \dots, v^n - 1\}$$

- $n$  – dozvoljeni broj cifara relativne adrese  $A \in \{1, \dots, B\}$

$$n = \lceil \log_v B \rceil, \quad 1 \leq n \leq p$$



# Statička rasuta organizacija datoteke

- **Metode probabilističke transformacije**

- koraci probabilističke transformacije vrednosti ključa

$$h: \text{dom}(K) \rightarrow A$$

- (3) dovođenje vrednosti pseudoslučajnog broja  $T$  u opseg dozvoljenih vrednosti relativne adrese  $\{1, \dots, B\}$

$$A = 1 + \lfloor kT \rfloor, \quad k = \frac{B}{v^n}, \quad 0,1 < k \leq 1$$

- $A \in \{1, \dots, B\}$
- (4) pretvaranje relativne u mašinsku adresu
  - opšti zadatak metode pristupa
  - rezultat primene prva tri koraka je relativna adresa

- **Metode probabilističke transformacije**
  - tri često upotrebljavane metode
    - metoda ostatka pri deljenju
    - metoda centralnih cifara kvadrata ključa
    - metoda preklapanja

# Statička rasuta organizacija datoteke

- **Metoda ostatka pri deljenju**

- relativna adresa  $A$  – celobrojni ostatak pri deljenju numeričke vrednosti ključa

$$T = k(S)(\text{mod } m)$$

- $m$  – ceo broj, takav da:

$$v^{n-1} < m \leq v^n$$

$$A = 1 + \left\lfloor \frac{B}{m} T \right\rfloor, \quad k = \frac{B}{m}, \quad 0,1 < k \leq 1$$

- kada se izabere da je  $m = B$ :

$$A = 1 + k(S)(\text{mod } B)$$

# Statička rasuta organizacija datoteke

---

- **Metoda ostatka pri deljenju**
  - preporuke za izbor vrednosti  $m$ 
    - kako bi rezultat transformacije bio što slučajniji broj, a transformacija što ravnomernija
  - $m$  ne treba da bude paran broj, jer tada
    - neparne vrednosti ključa daju neparne vrednosti relativne adrese, a
    - parne vrednosti ključa daju parne adrese
  - $m$  ne treba da bude stepen osnove brojnog sistema
    - jer bi tada cifre najmanje težine ključa predstavljale relativnu adresu bez obzira na vrednost ostalih cifara
  - najpogodnije da  $m$  predstavlja
    - prost broj ili
    - neparan broj sa relativno velikim prostim činilcima

# Statička rasuta organizacija datoteke

- Metoda ostatka pri deljenju**

$$A = 1 + k(S)(\text{mod } B)$$

– Primer:  $B = 5, m = B$

$S_i$	$S_1$	$S_2$	$S_3$	$S_4$	$S_5$	$S_6$	$S_7$	$S_8$	$S_9$
$k(S_i)$	34	7	3	15	19	29	64	43	23
$T_i$	4	2	3	0	4	4	4	3	3
$A_i$	5	3	4	1	5	5	5	4	4

- **Metoda ostatka pri deljenju**

- pogodna za primenu kada se vrednosti ključa javljaju u paketima
  - pojedini intervali u opsegu dozvoljenih vrednosti ključa gusto zaposednuti aktuelnim vrednostima ključa
  - između njih intervali sa neaktuelnim vrednostima ključa



- slogovi sa sukcesivnim vrednostima ključa iz paketa dobijaju adrese fizički susednih baketa
  - što rezultuje ravnomernim zaposedanjem baketa

- **Metoda centralnih cifara kvadrata ključa**
  - vrednost ključa diže se na kvadrat
  - uzima se onoliko centralnih cifara kvadrata vrednosti ključa koliko pozicija treba da sadrži relativna adresa
    - formira se pseudoslučajan broj  $T$
  - vrši se centriranje i normiranje pseudoslučajnog broja na zadati opseg relativnih adresa

- **Metoda centralnih cifara kvadrata ključa**

- vrednost ključa  $k(S) \in \{0, 1, \dots, v^p - 1\}$  u polinomijalnom obliku:

$$k(S) = \sum_{i=0}^{p-1} a_i v^i, \quad a_i \in \{0, 1, \dots, v-1\}$$

- kvadrat u vrednosti ključa u polinomijalnom obliku:

$$(k(S))^2 = \sum_{i=0}^{2p-1} c_i v^i, \quad c_i \in \{0, 1, \dots, v-1\}$$



# Statička rasuta organizacija datoteke

- Metoda centralnih cifara kvadrata ključa**

- iz niza cifara kvadrata ključa  $(c_{2p-1}, c_{2p-2}, \dots, c_1, c_0)$  izdvaja se podniz od  $n$  centralnih cifara

$$(c_{t+n-1}, c_{t+n-2}, \dots, c_{t+1}, c_t)$$

- $t$  – pozicija najlakše cifre podniza  $t = \left\lfloor p - \frac{n}{2} \right\rfloor$
- formira se pseudoslučajni broj  $T$ :

$$T = \sum_{i=0}^{n-1} c_{t+i} v^i$$

- relativna adresa matičnog baketa  $A$ :  $A = 1 + \left\lfloor \frac{B}{v^n} T \right\rfloor$

# Statička rasuta organizacija datoteke

## • Metoda centralnih cifara kvadrata ključa

– Primer:

- $p = 2, v = 10,$
- $B = 20, n = 2,$
- $t = 1, k = B / v^n = 0,2$

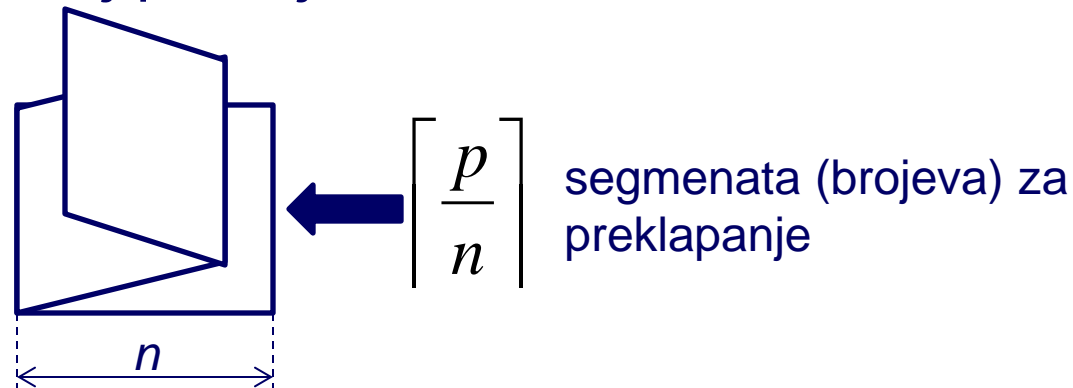
$$A = 1 + \left\lfloor \frac{B}{v^n} T \right\rfloor$$

$S_i$	$S_1$	$S_2$	$S_3$	$S_4$	$S_5$	$S_6$	$S_7$	$S_8$	$S_9$
$k(S_i)$	34	7	3	15	19	29	64	43	23
$(k(S_i))^2$	1156	0049	0009	0225	0361	0841	4096	1849	0529
$T_i$	15	04	00	22	36	84	09	84	52
$A_i$	4	1	1	5	8	17	2	17	11

# Statička rasuta organizacija datoteke

- **Metoda preklapanja**

- cifre ključa premeštaju se kao pri savijanju, tj. preklapanju hartije
- vrši se sabiranje preklapljenih vrednosti po modulu  $v^n$
- pogodna za primenu
  - kada je broj pozicija vrednosti ključa  $p$  mnogo veći od broja pozicija relativne adrese  $n$
- preklapanje se izvodi po osama koje zdesna u levo određuje broj pozicija  $n$  relativne adrese



# Statička rasuta organizacija datoteke

- **Metoda preklapanja**

- vrednost ključa  $k(S) \in \{0, 1, \dots, v^p - 1\}$  u polinomijalnom obliku:

$$k(S) = \sum_{i=0}^{p-1} a_i v^i, \quad a_i \in \{0, 1, \dots, v-1\}$$

- pseudoslučajan broj  $T$

$$T = \left( \sum_{k=0}^q \sum_{i=0}^{n-1} c_r v^i + \sum_{k=1}^q \sum_{i=0}^{n-1} c_s v^i \right) (\text{mod } v^n)$$

$$q = \left\lfloor \frac{p}{2n} \right\rfloor, \quad r = 2kn + i, \quad s = 2kn - i - 1$$

$$c_r = \begin{cases} c_r, & \text{za } r < p \\ 0, & \text{za } r \geq p \end{cases} \quad \text{i} \quad c_s = \begin{cases} c_s, & \text{za } s < p \\ 0, & \text{za } s \geq p \end{cases}$$

# Statička rasuta organizacija datoteke

## • Metoda preklapanja

– relativna adresa matičnog baketa  $A$ :  $A = 1 + \left\lfloor \frac{B}{v^n} T \right\rfloor$

– Primer:

- $p = 6, v = 10,$
- $B = 20, n = 2,$
- $\lceil p / n \rceil = 3, k = B / v^n = 0,2$
- $k(S_1) = 341201, T_1 = (01 + 21 + 34) \bmod 10^2 = 56$

$S_i$	$S_1$	$S_2$	$S_3$	$S_4$	$S_5$	$S_6$	$S_7$	$S_8$	$S_9$
$k(S_i)$	<u>341201</u>	<u>237896</u>	<u>465210</u>	<u>542812</u>	<u>191378</u>	<u>296532</u>	<u>641000</u>	<u>430025</u>	<u>231258</u>
$T_i$	56	6	81	48	28	17	65	68	2
$A_i$	12	2	17	10	6	4	14	14	1

- **Karakteristike probabilističke transformacije**

- pojava slogova **sinonima**

- dve različite vrednosti ključa mogu transformacijom dobiti istu relativnu adresu
- slogovi sinonimi – slogovi koji transformacijom dobiju iste relativne adrese

$$k(S_i) \neq k(S_j), h(k(S_i)) = h(k(S_j))$$

- **matični baket**

- baket čija relativna adresa predstavlja rezultat transformacije
- slogovi se uvek smeštaju u matični baket dok se ne popuni

- **primarni slog**

- slog koji je smešten u matični baket

- metoda transformacije particionira skup mogućih vrednosti ključa na  $B$  skupova sinonima

- **Karakteristike probabilističke transformacije**
  - **prekoračilac**
    - slog koji ne može biti smešten u matični baket, usled njegove popunjenosti
    - mora se smestiti u neki drugi baket
      - definiše se poseban postupak za smeštanje prekoračilaca
  - pojava prekoračilaca je nepoželjna, jer
    - zahteva poseban postupak za pronalaženje nove slobodne lokacije za smeštaj prekoračilaca
    - dovodi do produženja vremena pristupa pri kasnijem traženju slogova prekoračilaca

# Statička rasuta organizacija datoteke

- **Karakteristike probabilističke transformacije**

- verovatnoća pojave sinonima zavisi od
  - raspodele vrednosti ključa unutar opsega dozvoljenih vrednosti
  - odabrane metode transformacije
  - faktora popunjenosti memorijskog prostora

$$q = \frac{N}{Q} = \frac{N}{bB}$$

- broj prekoračilaca će biti manji
  - što su slogovi ravnomernije raspoređeni po baketima
  - što je faktor popunjenosti manji
  - što je faktor baketiranja veći
- pojava prekoračilaca je neminovnost



- **Karakteristike probabilističke transformacije**
  - izbor faktora popunjenosti
    - ostvaruje veliki uticaj na karakteristike rasuto organizovane datoteke
  - za malo  $q$ 
    - verovatnoća pojave više slogova u jednom skupu sinonima je takođe mala ali je malo i iskorišćenje memorijskog prostora
  - za veliko  $q$  (blizu 1)
    - iskorišćenje memorijskog prostora je dobro, ali je velika verovatnoća pojave sinonima i prekoračilaca
  - u praksi se bira  $q \leq 0,8$

# Statička rasuta organizacija datoteke

---

- **Karakteristike probabilističke transformacije**
  - izbor faktora baketiranja
    - utiče na očekivani broj prekoračilaca po jednom baketu, pri datom faktoru popunjenosti  $q$
  - s porastom faktora baketiranja  $b$ 
    - verovatnoća pojave prekoračilaca opada
    - ali raste vreme razmene sadržaja baketa između diska i OM
      - putem jednog baketa učitava se veći broj, potencijalno nepotrebnih slogova u OM
  - sa smanjenjem faktora baketiranja  $b$ 
    - povećava se očekivani broj prekoračilaca
    - za isti  $N$  i  $q$  povećava se broj baketa  $B$  i poboljšava se preciznost transformacije
  - u praksi, bira se  $b \leq 10$

# Statička rasuta organizacija datoteke

---

- **Projektovanje rasute datoteke**

- pri projektovanju utvrđuju se
  - faktor popunjenosti memorijskog prostora  $q$
  - faktor baketiranja  $b$
  - metoda transformacije vrednosti ključa u adresu
  - postupak za smeštanje prekoračilaca
- ova opredeljenja donose se s obzirom na
  - raspodelu vrednosti ključa unutar opsega mogućnih vrednosti
  - veličinu sloga
  - obim i karakter ažuriranja datoteke
  - očekivani srednji broj pristupa datoteci pri uspešnom i neuspešnom traženju

# Statička rasuta organizacija datoteke

---

- **Postupci za smeštaj prekoračilaca**
  - smeštaj svih prekoračilaca unutar jedinstvenog adresnog prostora
    - izbor posebnog postupka za pronalaženje prazne lokacije za smeštaj prekoračioca
  - smeštaj svih prekoračilaca u posebnu zonu adresnog prostora
    - datoteka poseduje dve zone, primarnu i zonu prekoračenja
      - prekoračioc se smeštaju u neku od slobodnih lokacija unutar zone prekoračenja
    - izbor vrste fizičke organizacije zone prekoračenja
  - **kombinacija prethodna dva načina**
    - lokalne zone prekoračenja u okviru primarne zone
    - glavna zona prekoračenja – posebna zona

# Rasute organizacije datoteka

1

15	$p(S_4)$	25	$p(S_7)$		

2


Primarna zona

3

07	$p(S_2)$	27	$p(S_9)$		

4

03	$p(S_3)$	23	$p(S_8)$	13	$p(S_{10})$

5

34	$p(S_1)$	19	$p(S_5)$	29	$p(S_6)$

Zona prekoračenja



# Statička rasuta organizacija datoteke

- **Projektovanje i formiranje rasute datoteke**

- za zadati  $b$ ,  $q$  i  $N$ , broj baketa je:

$$B = \left\lceil \frac{N}{bq} \right\rceil$$

- aktivnosti formiranja rasuto organizovane datoteke realizuju se

- potpuno uz pomoć metode pristupa, ako je podržava ili
    - delom uz pomoć aplikativnog programa i delom uz pomoć relativne metode pristupa
      - aplikativni program: transformacija vrednosti ključa, smeštanje prekoračilaca, formiranje baketa od više slogova, izdvajanje slogova iz baketa
      - relativna metoda: direktni upis i čitanje sadržaja baketa

- **Formiranje rasute datoteke**
  - (1) inicijalno alociranje prazne datoteke
    - statička alokacija kompletnog praznog prostora datoteke
    - izbor postupka prepoznavanja slobodnih lokacija unutar baketa
      - putem sadržaja statusnog polja
      - upisom specijalnih znakova u lokaciju
      - putem indeksa slobodnih lokacija i njihovog sprezanja
  - (2) upisivanje slogova u rasutu datoteku
    - saglasno opštem postupku formiranja rasutih datoteka
    - na osnovu sadržaja vodeće datoteke kojoj se sekvencijalno pristupa ili
    - direktnim upisivanjem slogova u realnom vremenu

- **Formiranje rasute datoteke**
  - (A) Formiranje u jednom prolazu
  - (B) Formiranje u dva prolaza
  
- (A) Formiranje u jednom prolazu
  - slogovi se upisuju u hronološkom redosledu nastanka
    - bilo čitanjem sadržaja vodeće datoteke u sekvencijalnom pristupu, bilo direktnim unosom podataka u realnom vremenu



# Rasute organizacije datoteka

39,24,33,6

06 →

1					
	15	$p(S_4)$	25	$p(S_7)$	5
					$p(S_{11})$

2					
	39		24		33

3					
	07	$p(S_2)$	27	$p(S_9)$	06

4					
	03	$p(S_3)$	23	$p(S_8)$	13
					$p(S_{10})$

5					
	34	$p(S_1)$	19	$p(S_5)$	29
					$p(S_6)$

Transformacija  
 $h(k_i) = 1 + k(S_i) \pmod{B}$

- **Formiranje rasute datoteke**
  - **(B) Formiranje u dva prolaza**
    - slogovi se učitavaju iz vodeće datoteke i upisuju u rasutu
    - **(I) Prvi prolaz**
      - upisuju se samo oni slogovi koji će biti smešteni u matične bakete
    - **(II) Drugi prolaz**
      - upisuju se preostali slogovi – prekoračioc, saglasno izabranom postupku za smeštanje prekoračilaca
  - ima smisla kod datoteka sa jedinstvenim adresnim prostorom
    - time se umanjuje ukupan broj prekoračilaca

- **Traženje sloga u rasutoj datoteci**
  - traženje logički narednog = slučajno odabranog sloga
    - vrši se metodom transformacije argumenta u adresu baketa
  - ako se slog ne nađe u matičnom baketu, a matični baket ima prekoračilaca, traženje se nastavlja
    - saglasno izabranom postupku za smeštanje i traženje prekoračilaca
      - linearna metoda
      - ponovna transformacija
      - metoda praćenja pokazivača
  - za zadati  $q$  i  $b$ , efikasnost traženja zavisi od primenjenog postupka za smeštaj prekoračilaca

# Statička rasuta organizacija datoteke

---

- **Vrste statičkih rasutih organizacija**
  - **rasute datoteke s jedinstvenim adresnim prostorom**
    - datoteka sa linearnim traženjem lokacije za smeštaj prekoračilaca – sa otvorenim načinom adresiranja
      - s fiksnim korakom  $k$ ,  $k \geq 1$
      - sa slučajno odbarnim korakom  $k$ ,  $k \geq 1$
    - datoteka sa sprezanjem prekoračilaca u jedinstvenom adresnom prostoru – primarnoj zoni
  - **rasute datoteke sa zonom prekoračenja**
    - sa serijskom zonom prekoračenja
    - sa spregnutom zonom prekoračenja

# Statička rasuta organizacija datoteke

- **Rasuta s linearnim traženjem prekoračilaca**

- s fiksnim korakom  $k$ ,  $k = 1$

- ukoliko je matični baket popunjen, slog se smešta u prvu narednu slobodnu lokaciju

- s obzirom na poziciju matičnog baketa

$$A_0 = h(k(S))$$

$$A_n = 1 + A_{n-1} \bmod B, \text{ za } n \geq 1$$

- **traženje slučajno odabranog sloga - prekoračioca**

- linearnom metodom

- zaustavlja se na

- a) pronađenom slogu, ako je uspešno

- b) prvoj slobodnoj lokaciji, ako je neuspešno

- c) ponovnim nailaskom na matični baket, ako je neuspešno, a cela datoteka kompletno popunjena

# Statička rasuta organizacija datoteke

---

- **Rasuta s linearnim traženjem prekoračilaca**
  - s fiksnim korakom  $k$ ,  $k = 1$
  - **upis novog sloga - prekoračioca**
    - pronalaženjem prve slobodne lokacije iza matičnog baketa
  - **brisanje postojećeg sloga**
    - **logičko**, potrebna tri statusa sloga
      - a) aktuelan, b) neaktuelan i c) slobodna lokacija
    - **fizičko**, uz lokalnu reorganizaciju memorijskog prostora
      - primarnog sloga, kada ne postoje prekoračioci
        - » oslobađa se lokacija
      - primarnog sloga, kada postoje prekoračioci ili prekoračioca
        - » svi prekoračioci (ne samo za dati matični baket) pomeraju se za jednu poziciju prema matičnom baketu
        - » pri pomeranju, prekoračilac ne sme da pređe ispred svog matičnog baketa

# Statička rasuta organizacija datoteke

---

- **Rasuta s linearnim traženjem prekoračilaca**
  - s fiksnim korakom  $k$ ,  $k = 1$
- glavni nedostaci
  - **(A) efekat nagomilavanja prekoračilaca**
    - prekoračioc i iz jednih baketa izazivaju pojavu prekoračilaca iz drugih baketa
    - sve više raste verovatnoća zauzeća prve prazne lokacije iza sve dužeg lanca zauzetih lokacija
  - **(B) neefikasno traženje**
    - traženje se vrši i u baketima koji ne sadrže slogove iz istog skupa sinonima
  - **(C) neefikasno neuspešno traženje**
    - zaustavlja se tek nailaskom na prvu slobodnu lokaciju

# Statička rasuta organizacija datoteke

- **Rasuta s linearnim traženjem prekoračilaca**
  - s fiksnim korakom  $k$ ,  $k > 1$
  - ukoliko je matični baket popunjen, slog se smešta u narednu slobodnu lokaciju, udaljenu za  $k > 1$  pozicija
    - s obzirom na poziciju matičnog baketa

$$A_0 = h(k(S))$$

$$A_n = 1 + (A_{n-1} + k - 1) \bmod B, \text{ za } n \geq 1$$

- $k$  i  $B$  moraju biti uzajamno prosti brojevi
  - kako bi se obezbedio, u najgorem slučaju, siguran obilazak svih mogućih baketa
    - u cilju pronalaska prve slobodne lokacije



# Statička rasuta organizacija datoteke

---

- **Rasuta s linearnim traženjem prekoračilaca**
  - s fiksnim korakom  $k$ ,  $k > 1$
- glavna motivacija
  - odlaganje efekta nagomilavanja prekoračilaca
- pozitivan efekat
  - prekidanje dugačkih lanaca zauzetih lokacija
    - pokušaj da se, koliko je to moguće, očuva približno jednaka verovatnoća zauzeća bilo koje prazne lokacije

- **Rasuta s linearnim traženjem prekoračilaca**
  - sa slučajno odabranim korakom  $k$ ,  $k \geq 1$
  - **Rasuta datoteka sa slučajnim traženjem**
  - ukoliko je matični baket popunjen, slog se smešta u narednu slobodnu lokaciju, udaljenu za  $k \geq 1$  pozicija
    - s obzirom na poziciju matičnog baketa
  - $k$  se određuje na slučajan način
    - predstavlja rezultat druge probabilističke transformacije, primenjene na vrednost identifikatora – ključa sloga

# Statička rasuta organizacija datoteke

- **Rasuta s linearnim traženjem prekoračilaca**

- sa slučajno odabranim korakom  $k$ ,  $k \geq 1$
- dve probabilističke transformacije

$$A_0 = h_1(k(S)) \in \{1, \dots, B\}$$

$$k = h_2(k(S)) \in \{1, \dots, B - 1\}$$

$$A_n = 1 + (A_{n-1} + k - 1) \bmod B, \text{ tj.}$$

$$A_n = 1 + (A_{n-1} + h_2(k(S)) - 1) \bmod B, \text{ za } n \geq 1$$

- $k$  i  $B$  moraju biti uzajamno prosti brojevi
  - kako bi se obezbedio, u najgorem slučaju, siguran obilazak svih mogućih baketa
    - u cilju pronalaska prve slobodne lokacije
  - pošto je  $k$  slučajna veličina, bira se da  $B$  bude prost broj
  - često,  $k = h_2(k(S)) = 1 + k(S) \bmod (B - 1)$

# Statička rasuta organizacija datoteke

---

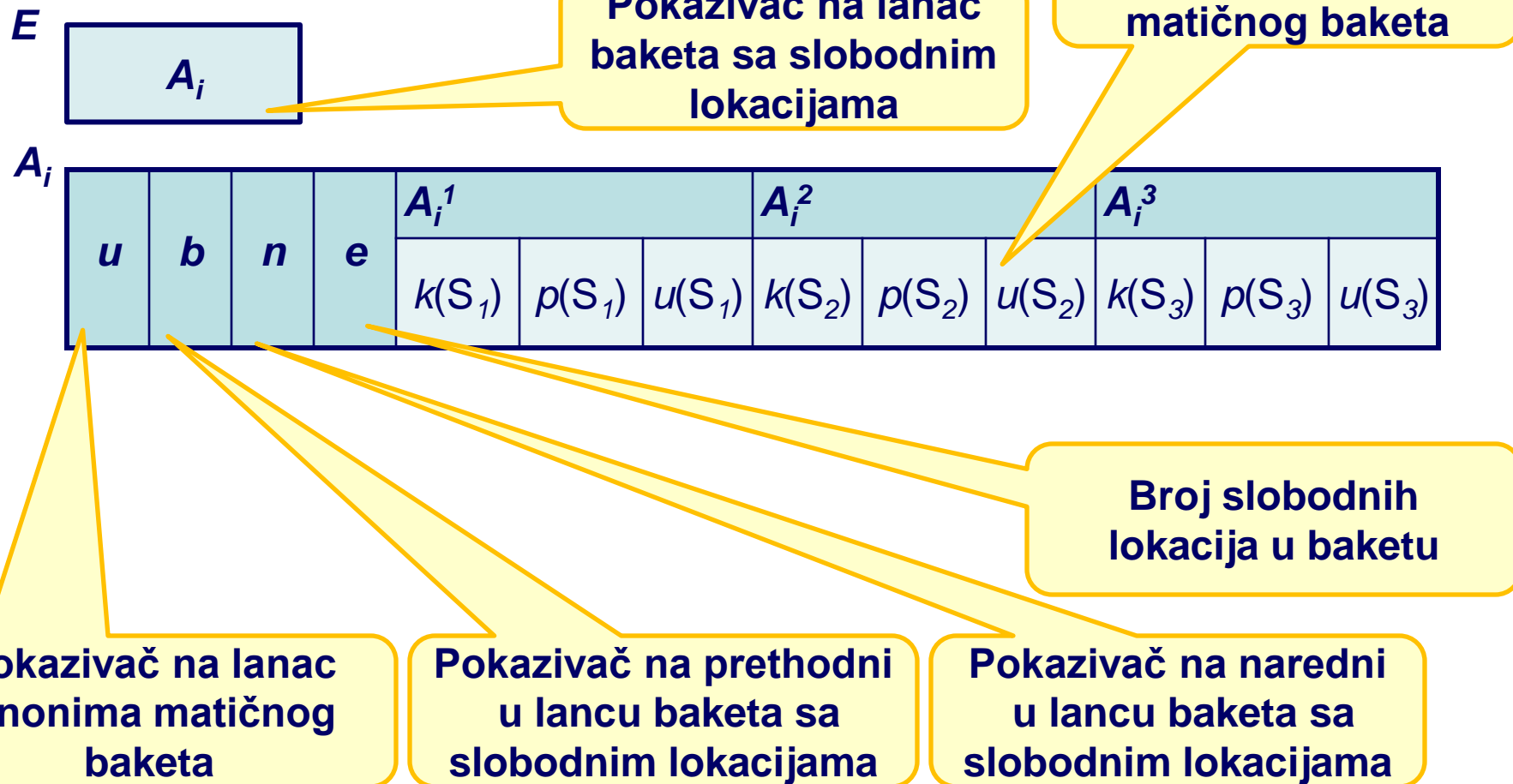
- **Rasuta s linearnim traženjem prekoračilaca**
  - sa slučajno odabranim korakom  $k$ ,  $k \geq 1$
- glavna motivacija
  - izbegavanje efekta nagomilavanja prekoračilaca
- pozitivan efekat
  - prekidanje dugačkih lanaca zauzetih lokacija
    - bolji pokušaj da se, koliko je to moguće, očuva približno jednaka verovatnoća zauzeća bilo koje prazne lokacije
- **Rasute s otvorenim načinom adresiranja**
  - pogodne za upotrebu u slučaju
    - manje popunjenosti,  $q \leq 0,7$
    - nižeg intenziteta ažuriranja

- **Rasuta sa sprezanjem u primarnoj zoni**
  - primena tehnike sprezanja i složenija fizička struktura
  - ukoliko je matični baket popunjen, slog se smešta u prvu slobodnu lokaciju iz lanca slobodnih lokacija
    - sprežu se dvostruko baketi sa slobodnim lokacijama
    - specijalan (nulti) baket s pokazivačem na početak lanca
  - vrši se sprezanje svih sinonima u odnosu na matični baket
    - za svaki matični baket po jedan lanac sinonima
    - pokazivač na početak lanca u zaglavlju matičnog baketa
    - pokazivač na sledeći u lancu sinonima ugrađen u svaki slog

# Statička rasuta organizacija datoteke

## • Rasuta sa sprezanjem u primarnoj zoni

– format baketa i sloga



- **Rasuta sa sprezanjem u primarnoj zoni**
  - **traženje slučajno odabranog sloga**
    - transformacija vrednosti ključa u adresu  $A = h(k(S))$  i pristupanje matičnom baketu
    - praćenje lanca sinonima, započinjući od pokazivača  $u$ 
      - metodom praćenja pokazivača
    - zaustavlja se na
      - a) pronađenom slogu, ako je uspešno
      - b) kraju lanca sinonima, ako je neuspešno

# Statička rasuta organizacija datoteke

---

- **Rasuta sa sprezanjem u primarnoj zoni**
  - **upis novog sloga**
    - nakon neuspešnog traženja, uvezivanjem u lanac sinonima
    - izborom prve prazne lokacije iz lanca baketa sa slobodnim lokacijama
      - na koju ukazuje  $E$
  - **brisanje postojećeg sloga**
    - **fizičko**
    - uklanjanjem sloga iz lanca sinonima, uz potrebno prevezivanje
    - oslobađanje lokacije, uz eventualno vraćanje baketa u lanac baketa sa slobodnim lokacijama



# Statička rasuta organizacija datoteke

---



- **Rasuta sa sprezanjem u primarnoj zoni**
- glavna motivacija
  - izbegavanje efekata neefikasnog traženja (B) i (C)
    - traženje se vrši samo u baketima koji sadrže slogove iz istog skupa sinonima
    - neuspešno traženje zaustavlja se dolaskom do kraja lanca spregnutih slogova - sinonima
- pozitivan efekat
  - poboljšana efikasnost traženja (naročito neuspešnog)
    - u odnosu na datoteke s otvorenim načinom adresiranja
- negativan efekat
  - i dalje moguć efekat nagomilavanja prekoračilaca (A)
  - komplikovanja fizička struktura

# Statička rasuta organizacija datoteke

---

- **Rasuta sa sprezanjem u zoni prekoračenja**
  - uvođenje zone prekoračenja – spregnuta datoteka
  - primena tehnike sprezanja i složenija fizička struktura
  - ukoliko je matični baket popunjen, slog se smešta u prvu slobodnu lokaciju iz lanca slobodnih lokacija u zoni prekoračenja
    - sprežu se jednostruko baketi sa slobodnim lokacijama u zoni prekoračenja
    - specijalan (nulti) baket s pokazivačem na početak lanca
  - vrši se sprezanje svih prekoračilaca
    - za svaki matični baket po jedan lanac prekoračilaca
    - pokazivač na početak lanca u zaglavlju matičnog baketa
    - pokazivač na sledeći u lancu prekoračilaca ugrađen u svaki slog u zoni prekoračenja

- **Rasuta sa sprezanjem u zoni prekoračenja**
  - dimenzionisanje spregnute zone prekoračenja
  - tipičan faktor blokiranja  $f = 1$ 
    - mala je verovatnoća da se dva prekoračioca iz istog lanca sinonima nađu u susednim lokacijama
  - **formiranje**
    - uvek u jednom prolazu
    - svi prekoračioci su u zoni prekoračenja, koja je odvojena od primarne zone

- **Rasuta sa sprezanjem u zoni prekoračenja**
  - **traženje slučajno odabranog sloga**
    - transformacija vrednosti ključa u adresu  $A = h(k(S))$  i pristupanje matičnom baketu
    - praćenje lanca prekoračilaca, započinjući od pokazivača na početak lanca u matičnom baketu
      - ukoliko slog nije pronađen u matičnom baketu, a postoji lanac prekoračilaca
      - metodom praćenja pokazivača
    - zaustavlja se na
      - a) pronađenom slogu, ako je uspešno
      - b) kraju lanca prekoračilaca, ako je neuspešno

# Statička rasuta organizacija datoteke

---

- **Rasuta sa sprezanjem u zoni prekoračenja**
  - **upis novog sloga**
    - nakon neuspešnog traženja
    - u matični baket, ako ima mesta
    - uvezivanjem u lanac prekoračilaca, ako u matičnom baketu nema mesta
    - izborom prve prazne lokacije iz lanca baketa sa slobodnim lokacijama u zoni prekoračenja
      - na koju ukazuje pokazivač na početak lanca
  - **brisanje postojećeg sloga**
    - **fizičko**
    - uklanjanjem sloga iz matičnog baketa uz, eventualno, prebacivanje prvog prekoračioca u matični baket, ili
    - uklanjanjem sloga iz lanca prekoračilaca, uz potrebno prevezivanje

# Statička rasuta organizacija datoteke

---



- **Rasuta sa sprezanjem u zoni prekoračenja**
- glavna motivacija
  - izbegavanje efekata neefikasnog traženja (B) i (C)
    - traženje se vrši samo u baketima koji sadrže slogove iz istog skupa sinonima
    - neuspešno traženje zaustavlja se dolaskom do kraja lanca spregnutih slogova – sinonima
  - uklanjanje efekta nagomilavanja prekoračilaca (A)
    - svi prekoračioci u zoni prekoračenja – spregnuta datoteka
- pozitivan efekat
  - poboljšana efikasnost traženja (naročito neuspešnog)
    - u odnosu na datoteke s jedinstvenim adresnim prostorom

- **Rasuta sa serijskom zonom prekoračenja**
  - uvođenje zone prekoračenja – serijska datoteka
  - ukoliko je matični baket popunjen, slog se smešta u prvu slobodnu lokaciju u serijskoj zoni prekoračenja
  - jednostavna struktura
    - nema dodatnih polja pokazivača
  - pogodna u slučaju manjeg očekivanog ukupnog broja prekoračilaca
    - ne isplati se sprezanje prekoračilaca u zoni prekoračenja

# Statička rasuta organizacija datoteke

- **Ocena traženja sloga u rasutoj datoteci**

- očekivani broj prekoračilaca po jednom slogu

$$\frac{\bar{L}B}{N} = \frac{q-1}{q} - \frac{1}{qb} \sum_{i=0}^b (i-b)P(i)$$

- pri zadatom  $q$  i  $b$  ne zavisi od broja slogova  $N$ 
  - odnos  $N/B$  je konstantan
- očekivani broj pristupa pri uspešnom i neuspešnom traženju zavisi od  $q$  i  $b$ , a ne od  $N$
- karakteristike velikih rasutih datoteka mogu se procenjivati poređenjem s malim datotekama, ali sa istim  $q$  i  $b$



# Statička rasuta organizacija datoteke

- **Ocena traženja sloga u rasutoj datoteci**

- očekivani broj pristupa za uspešno traženje slučajno odabranog sloga, kod svih vrsta

$$\bar{R}_u = \frac{1}{N} \sum_{i=1}^N R_i^u$$

- prebroji se potreban broj pristupa za svaki pojedinačni slog
- očekivani broj pristupa za neuspešno traženje slučajno odabranog sloga, kod svih vrsta
  - osim kod datoteke sa slučajnim traženjem prekoračilaca

$$\bar{R}_n = \frac{1}{B} \sum_{i=1}^B R_i^n$$

- prebroji se potreban broj pristupa za svaki matični baket

# Statička rasuta organizacija datoteke

- **Ocena traženja sloga u rasutoj datoteci**
  - očekivani broj pristupa za neuspešno traženje slučajno odabranog sloga
    - kod datoteke sa slučajnim traženjem prekoračilaca

$$\bar{R}_n = \frac{1}{T} \sum_{i=1}^T R_i^n \quad \text{ili} \quad \bar{R}_n = \frac{1}{B(B-1)} \sum_{i=1}^{B(B-1)} R_i^n$$

- prebroji se potreban broj pristupa za
  - svaku neaktuelnu vrednost ključa,  $T = v^p - N$  ili
  - svaki matični baket i svaku moguću vrednost koraka  $k \in \{1, \dots, B - 1\}$

- **Obrada rasute datoteke sa probabilističkom transformacijom**
  - nepogodne za korišćenje u ulozi osnovne (prve) vodeće datoteke
  - mogu se koristiti kao obrađivane i vodeće u režimu direktne obrade
  - ne mogu se koristiti kao vodeće u režimu redosledne obrade
    - pošto fizička struktura ne sadrži informaciju o logičkoj strukturi podataka
  - mogu se obrađivati i u režimu redosledne i u režimu direktne obrade

# Statička rasuta organizacija datoteke

- **Obrada rasute datoteke sa probabilističkom transformacijom**

- performanse redosledne i direktne obrade rasute datoteke su iste
  - zbog iste efikasnosti traženja i logički narednog i slučajno odabranog sloga
- očekivani ukupni broj pristupa

$$\overline{R}_{uk} = N_v^u \overline{R}_u + N_v^n \overline{R}_n$$

- broj slogova vodeće datoteke  $N_v = N_v^u + N_v^n$
- $\overline{R}_u$  očekivani broj pristupa pri uspešnom traženju 1 sloga
- $\overline{R}_n$  očekivani broj pristupa pri neuspešnom traženju 1 sloga



- **Oblasti primene rasutih datoteka**
  - u svim mrežnim SUBP
  - u pojedinim relacionim SUBP
  - u interaktivnoj obradi podataka
  - u režimu paketne obrade
  - prednost
    - mali očekivani broj pristupa pri traženju slučajno odabranog sloga

- **Oblasti primene rasutih datoteka**
  - nedostaci
    - potreba da se unapred odredi veličina datoteke
      - pogodne samo za datoteke čiji se sadržaj ređe menja
      - intenzivno upisivanje novih slogova dovodi do degradacije performansi obrade
    - problem izbora probabilističke transformacije
      - ravnomerna raspodela broja sinonima po baketima i pri formiranju i pri ažuriranju
    - broj pristupa pri traženju može biti nepredvidivo velik

# Sadržaj

---

- Rasute organizacije datoteka
- Direktna i relativna organizacija datoteke
- Statička rasuta organizacija datoteke
- Dinamička rasuta organizacija datoteke



- **Dinamička rasuta datoteka**
  - osnovni nedostatak statičkih rasutih datoteka
    - uzajamna zavisnost metode transformacije identifikatora u relativnu adresu i veličine adresnog prostora dodeljenog datoteci
    - dolazi do pojave prekoračilaca
    - dolazi do degradacije performansi obrade datoteke
  - za datoteke koje se skoro isključivo obrađuju u režimu direktne obrade i čiji se sadržaj intenzivno menja
    - moguće je uvesti rasutu organizaciju koja će se dinamički prilagođavati aktuelnom broju slogova u datoteci



# Dinamička rasuta organizacija datoteke

---

- **Dinamička rasuta datoteka**

- više vrsta struktura dinamičkih rasutih datoteka

- rasute datoteke koje se mogu širiti
- dinamičke rasute datoteke
- linearne dinamičke rasute datoteke

- zajedničke osobine

- metoda transformacije  $h$ 
  - ne zavisi od veličine adresnog prostora dodeljenog datoteci
  - ne menja se zbog upisa novih slogova ili brisanja postojećih
- rezultat primene  $h$  na vrednost ključa
  - binaran broj maksimalne dužine  $d_{max}$
  - vodećih  $d$  ( $0 \leq d \leq d_{max}$ ) bitova rezultata transformacije je vrednost transformacije  $vt$
  - $vt$  se koristi za pronalaženje adrese baketa u kojem je slog
- dužina  $d$  i broj baketa  $B$  se povećavaju i smanjuju dinamički

# Dinamička rasuta organizacija datoteke

- **Struktura dinamičke rasute datoteke**
  - dva dela
    - **adresar**
      - sadrži niz pokazivača dužine  $2^d$  ( $d \geq 0$ ) sa adresama baketa i druga polja (uključujući i  $d$  i faktor baketiranja  $b$ )
    - **zona podataka sa baketima**
      - svaki baket sadrži zaglavlje i bar jedan slog
      - zaglavlje baketa sadrži
        - » polje  $d'$  ( $0 \leq d' \leq d$ ) sa lokalnom dužinom vrednosti transformacije
        - » polje  $m$  sa brojem aktuelnih slogova u baketu
      - $d'$  govori koliko istih bitova najveće težine vrednosti transformacije  $vt$  moraju imati svi slogovi u baketu
  - adresar i zona podataka
    - realizuju se kao dve posebne datoteke

- **Adresar**

- mala pomoćna struktura podataka
  - kao indeks koristi se za pristup baketima na disku
- najčešće se realizuje kao linearna struktura
- adresar sadrži
  - jednodimenzionalni niz od  $2^d$  ( $\geq B$ ) pokazivača ka baketima u zoni podataka, gde je  $B$  broj aktuelnih baketa u zoni podataka
  - promenljivu  $d$ 
    - broj bitova najveće težine funkcije  $h(k)$  koji se trenutno koriste za indeksiranje niza pokazivača
  - faktor baketiranja  $b$

- **Adresar**

- koristi se samo stvarno neophodni broj bitova za adresiranje baketa

$$d = \lceil \log_2 B \rceil$$

- adresar zahteva relativno mali kapacitet memorijskog prostora
    - ceo adresar može smestiti u OM
    - pri otvaranju datoteke adresar se prenosi u OM
    - u OM ostaje do zatvaranja datoteke, kada se upisuje nazad na disk
  - elementima niza pokazivača
    - pristup na osnovu binarne vrednosti njihove pozicije u nizu
      - ta vrednost izražena je putem  $d$  bitova

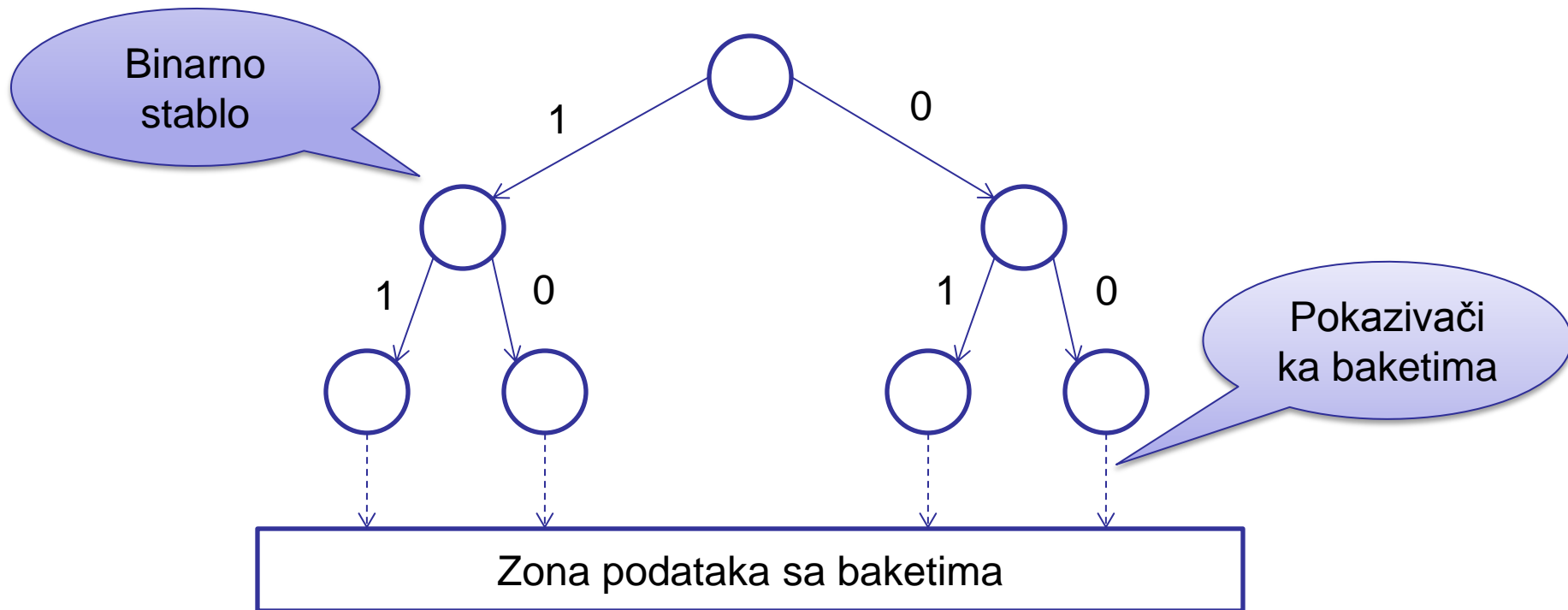


- **Adresar**

- indeksi niza pokazivača se ponekad predstavljaju kao kompletno binarno stablo
  - u čijim listovima se nalaze pokazivači ka baketima
  - levom odlaznom potezu iz jednog čvora pridružuje se binarni broj 1, a desnom 0
  - nizu dužine  $2^d$  odgovara kompletno binarno stablo visine  $h = d + 1$
  - svakom putu od korena do nekog lista odgovara  $h - 1 = d$  ivica i nosi informaciju o jednom od  $2^d$  indeksa u jednodimenzionalnom nizu pokazivača

# Dinamička rasuta organizacija datoteke

- Adresar - primer
  - binarno stablo,  $d = 2$ , koje reprezentuje niz pokazivača dužine  $2^d = 4$





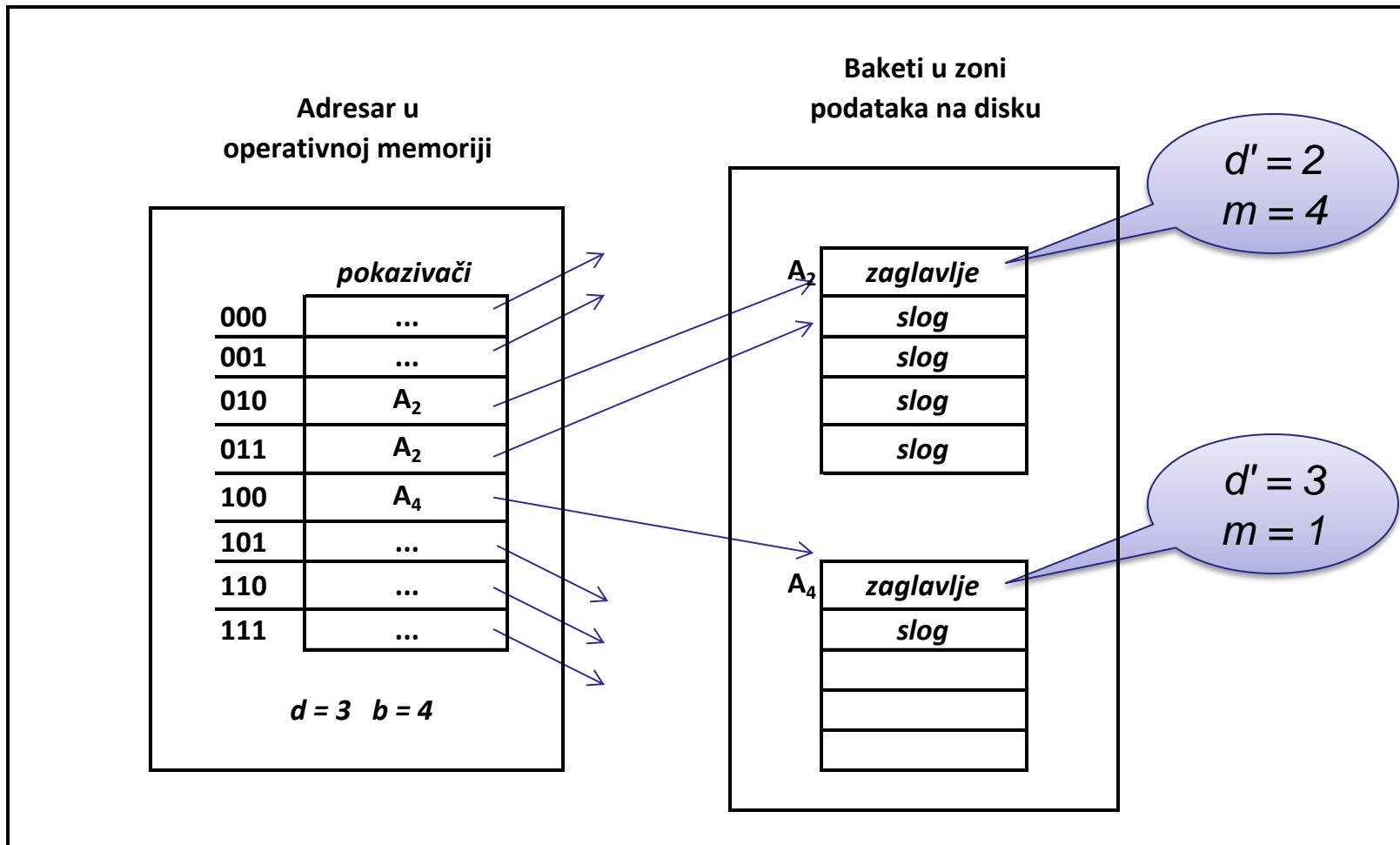
- **Veza između adresara i zone podataka**
  - zona podataka sadrži bakete
    - svaki sadrži najviše  $b$  slogova
    - svaki sadrži zaglavlje sa poljima  $d'$  i  $m$
    - svi slogovi u baketu moraju imati istih samo prvih  $d'$  bitova vrednosti transformacije
    - svaki element niza pokazivača u adresaru ukazuje na jedan baket koji poseduje slogove sa istih  $d'$  vodećih bitova vrednosti transformacije  $vt$

- **Veza između adresara i zone podataka**
  - ako za neki baket važi  $d = d'$ 
    - svi slogovi u baketu imaju iste vrednosti transformacije  $vt$  dužine  $d$
    - samo jedan pokazivač vodi od niza pokazivača u adresaru ka baketu
    - za  $d > 0$  vrednosti transformacije  $vt$  slogova u baketu su jednake onoj vrednosti transformacije koja predstavlja indeks pokazivača ka baketu  $i$
    - za  $d = d' = 0$  vrednosti transformacije slogova u baketu su nebitne, a niz pokazivača je jednočlan
  - pošto broj pokazivača adresara koji ukazuju na isti baket iznosi  $2^{d-d'}$ , ako za neki baket važi  $d' < d$  tada
    - slogovi u baketu moraju imati istih  $d'$  bitova najveće težine za  $vt$ , a mogu imati neke od  $d - d'$  bitova najmanje težine različite
    - $2^{d-d'}$  susednih pokazivača čiji indeksi imaju istih  $d'$  bitova najveće težine za  $vt$ , ukazuje ka posmatranom baketu



# Dinamička rasuta organizacija datoteke

- Predstava dinamičke rasute datoteke



- **Generisanje vrednosti transformacije**
  - kako bi se izbeglo snižavanje poželjnog stepena slučajnosti transformacije
    - vrednosti ključa  $k$  se podvrgavaju netrivialnoj transformaciji
    - redosled binarnih pozicija rezultata transformacije  $h(k)$  se invertuje
  - za transformaciju  $h$  se bira neka metoda generisanja pseudoslučajnih brojeva
    - čiji je cilj da od datih vrednosti ključa sloga proizvede niz vrednosti sa što ravnomernijom raspodelom
    - pojava više od  $b$  slogova sinonima u okviru jedne klase može dovesti do potrebe da se menja metoda transformacije jer
    - dinamičke rasute datoteke ne poseduju mehanizam za smeštaj prekoračilaca

- **Formiranje dinamičke rasute datoteke**
  - u režimu direktne obrade
    - vrednosti ključa slogova ulazne serijske datoteke podvrgavaju se transformaciji
    - od rezultata koristi se  $d$  bitova najveće težine kao indeks u jednodimenzionalnom nizu pokazivača ka baketima
    - baket čija je adresa dobijena korišćenjem tog niza u adresaru prenosi se u OM
    - ako slog sa istom vrednošću ključa u baketu ne postoji, novi slog se upisuje u baket
    - svakom upisu prethodi neuspešno traženje

- **Formiranje dinamičke rasute datoteke**
  - tri slučaja upisa
    - prost upis
      - izvršava se u baketu sa  $m < b$  slogova
      - novi slog se upisuje u prvi slobodnu lokaciju u baketu
      - broj zauzetih lokacija  $m$  se povećava za jedan
    - upis koji dovodi do deljenja baketa i udvostručavanja dužine niza pokazivača u adresaru
      - ako važi sledeće:  $m = b$ ,  $d' = d$
    - upis koji dovodi samo do deljenja baketa
      - ako važi sledeće:  $m = b$ ,  $d' < d$



- **Traženje sloga u dinamičkoj rasutoj datoteci**
  - traženje logički narednog i slučajno odabranog sloga vrši se korišćenjem istog algoritma
  - koraci algoritma za traženje slučajno odabranog sloga
    - vrednost ključa  $k$  se podvrgava transformaciji  $h$
    - rezultat transformacije  $h(k)$  se pretvara u vrednost transformacije dužine  $d_{max}$  bita
    - korišćenjem vrednosti transformacije dužine  $d \leq d_{max}$  u adresaru se pronalazi adresa baketa, u kojem bi traženi slog trebalo da bude
    - ako je traženi slog u baketu, traženje je uspešno, inače je neuspešno



- **Traženje sloga u dinamičkoj rasutoj datoteci**
  - transformacija ključa i generisanje vrednosti
    - vrši se u OM, ne zahteva pristup disku
  - kompletan adresar nalazi u OM
    - ne zahteva pristup disku
  - jedini pristup disku u cilju čitanja baketa
  - potrebno je najviše  $R = 1$  pristupa
    - ako se baket već nalazi u OM važi  $R = 0$

- **Ažuriranje dinamičke rasute datoteke**
  - u režimu direktne obrade
    - svakom upisu ili brisanju prethodi jedno traženje
  - **upis novog sloga**
    - vrši po principima upisa novih slogova pri formiranju datoteke
    - u najnepovoljnijem slučaju upis zahteva
      - dva pristupa datoteci, ako upis ne dovodi do prepunjenja baketa
        - » jedan pristup za neuspešno traženje
        - » drugi za upis baketa
      - tri pristupa datoteci, ako upis dovodi do prepunjenja baketa
        - » jedan pristup za neuspešno traženje
        - » drugi i treći pristup za upis polaznog i jednog novog baketa u datoteku
    - broj pristupa (najnepovoljniji slučaj):  $2 \leq R_i \leq 3$



- **Ažuriranje dinamičke rasute datoteke**
  - **brisanje postojećih slogova**
    - dva baketa su prijatelji ako zadovoljavaju uslove
      - na njih ukazuju dva takva pokazivača u adresaru čiji se indeksi razlikuju samo na poziciji najmanje težine
      - za oba baketa važi  $d' = d$
      - važi  $d > 0$



- **Ažuriranje dinamičke rasute datoteke**
  - tri slučaja brisanja
    - prosto brisanje
      - u baketu sa  $m > 1$  slogova za koji važi da je ukupan broj njegovih slogova i slogova njegovog prijatelja veći od  $b$
      - nakon uspešnog traženja svi slogovi u baketu koji su smešteni iza sloga koji se briše,
        - » pomeraju se za jednu poziciju ulevo
        - » parametar  $m$  se smanjuje za jedan
      - poslednji slog u baketu se briše
        - » smanjivanjem parametra  $m$  za jedan
        - » nakon čega se mora učitati prijatelj baketa da bi se proverio ukupni broj slogova u ta dva baketa

# Dinamička rasuta organizacija datoteke

---

- **Ažuriranje dinamičke rasute datoteke**
  - **tri slučaja brisanja**
    - spajanje susednih baketa, bez uticaja na veličinu adresara
      - dešava se kada nakon brisanja sloga ukupan broj slogova u dva baketa prijatelja nije veći od  $b$
      - baketi prijatelji se spajaju
      - u spojenom baketu lokalna vrednost transformacije postaje  $d' = d' - 1$
      - parametar  $m$  u baketu dobijenom spajanjem postaje  $m \leq b$
      - svi elementi niza pokazivača koji su ukazivali na bakete prijatelje pre spajanja dobijaju pokazivač ka baketu dobijenom spajanjem

- **Ažuriranje dinamičke rasute datoteke**
  - **tri slučaja brisanja**
    - spajanje susednih baketa sa smanjenjem dužine niza pokazivača na pola
      - dešava se kada na svaki baket ukazuju najmanje po dva pokazivača u adresaru
      - u adresaru se vrednost transformacije  $d$  umanjuje za jedan
        - » čime se dužina niza pokazivača smanjuje na pola
      - svaka dva pokazivača čiji se indeksi razlikuju samo na binarnoj poziciji najmanje težine transformišu se u jedan

# Dinamička rasuta organizacija datoteke

---

- **Ažuriranje dinamičke rasute datoteke**
  - **brisanje**
    - pod pretpostavkom da datoteka sadrži bar dva baketa, i u slučaju prostog i u slučaju brisanja sa spajanjem susednih baketa, u najnepovoljnijem slučaju je potrebno izvršiti tri pristupa datoteci
      - jedan za pronalaženje sloga
      - jedan za učitavanje susednog baketa
      - jedan za upis bilo ažuriranog polaznog baketa ili baketa dobijenog spajanjem
    - broj pristupa:  $R_d = 3$
    - redukcija veličine adresara ne zahteva pristupe disku



- **Obrada dinamičke rasute datoteke**
  - dinamičke (kao i statičke) rasute datoteke su nepogodne za korišćenje u ulozi osnovne vodeće datoteke
  - mogu se koristiti kao obrađivane i vodeće u režimu direktne obrade
  - ne mogu se koristiti kao vodeće u režimu redosledne obrade
    - pošto im fizička struktura ne sadrži podatke o logičkoj strukturi podataka

# Dinamička rasuta organizacija datoteke

- **Obrada dinamičke rasute datoteke**

- mogu se obrađivati i u režimu redosledne i u režimu direktne obrade

- iste su performanse redosledne i direktne obrade rasute datoteke

- zbog iste efikasnosti traženja i logički narednog i slučajno odabranog sloga

- pod pretpostavkom da se adresar može smestiti u OM broj pristupa za pronalaženje jednog sloga:  $R = 1$

- ukupni broj pristupa

$$R_{uk} = N_v^u + N_v^n$$

- važi i pri redoslednoj i pri direktnoj obradi

- ako je broj slogova vodeće datoteke

$$N_v = N_v^u + N_v^n$$

- **Ocena karakteristika i oblasti primene**
  - ne proizvodi slogove prekoračioce
  - datoteka se širi i skuplja u zavisnosti od broja aktuelnih slogova
  - broj pristupa pri traženju ne zavisi od veličine datoteke
    - ako se adresar može kompletan smestiti u OM, tada je broj pristupa i u najnepovoljnijem slučaju 1
    - zato su ove datoteke veoma pogodne za direktnu obradu
  - nisu pogodne za redoslednu obradu
    - ali broj pristupa za pronalaženje jednog logički narednog sloga u obrađivanoj rasutoj datoteci iznosi jedan



- **Ocena karakteristika i oblasti primene**
  - zauzeće memorijskog prostora na disku
    - očekivani broj baketa

$$\bar{B} \approx \frac{N}{b \ln 2}$$

- $N$  – broj slogova
- $b$  – faktor blokiranja



- **Ocena karakteristika i oblasti primene**
  - rasute datoteke sa dinamičkom transformacijom
    - zasnovane su na ideji deljenja baketa, koju su pozajmlele od *B*-stabala
      - u traženju slučajno odabranog sloga su efikasnije od *B*-stabala
    - popularnost ovih vrsta datoteka zaostaje za *B*-stablama
    - postoje SUBP-ovi
      - koji podržavaju korišćenje fizičkih struktura zasnovanih na principima dinamičkih rasutih datoteka
      - podrazumevaju korišćenje *B*-stabala dok se izgradnja rasutih struktura mora posebno zahtevati

# Sadržaj

- Rasute organizacije datoteka
- Direktna i relativna organizacija datoteke
- Statička rasuta organizacija datoteke
- Dinamička rasuta organizacija datoteke

# Pitanja i komentari

---



## Rasuta organizacija datoteke

---

*Struktura, formiranje, traženje,  
obrada, ažuriranje, primena i  
ocena*