3 – Datotečki sistem OS

USLUGE OS U ORGANIZACIJI DATOTEKA

DATOTEČKI (FILE) SISTEM OS

ZADACI:

1. Upravljanje datotekama – fizičkim strukturama podataka (FSP) na eksternim memorijskim uređajima.
2. Upravljanje i realizacija razmene podataka između aplikativnih programa i datoteka
3. Obezbeđenje mehanizma zaštite od neovlašćenog pristupa datotekama i oštećenja podataka u uslovima višekorisničkog i multiprogramskog režima rada
4. Obezbeđenje podrške različitih pogleda na LSP datoteke – preslikavanja LSP u FSP i obrnuto

USLUGE DATOTEČKIH SISTEMA

- **USLUGE NISKOG NIVOA** – Pokrivaju pobrojane zadatke ali obezbeđuju pogled na LSP datoteke samo kao niz bajtova i njeno preslikavanje u FSP niza blokova

  Usluge niskog nivoa su servisi koji isključivo pripadaju OS. U njih spadaju:

  1. **Upravljanje memorijskim prostorom**
  2. **Upravljanje katalogom**
  3. **Upravljanje fizičkom razmenom podataka**
  4. **Obezbeđenje veze programa i datoteke**
  5. **Sistemski pozivi**

- **USLUGE VISOKOG NIVOA** – Usluge visokog nivoa su servisi koji mogu biti ugrađeni u OS, biblioteke programskih jezika ili SUBP, ili mogu biti prepušteni nivou aplikativnog programa, najčešće u slučaju razvoja specijalnih aplikacija ili korišćenja specijalnih procesorskih uređaja za obradu podataka.

  Obavezno uključuju sve usluge niskog nivoa. Obezbeđuju različite poglede na LSP datoteke kao različitih struktura nad skupom slogova/blokova/bajtova i njihovih preslikavanja u FSP niza blokova. Takođe obezbeđuju izgradnju specijalnih pomoćnih struktura za poboljšanje efikasnosti obrade podataka kao i traženja zasnovana na vrednosti podataka.

  U njih spadaju

  6. **Metode pristupa**

Svaka od navedenih usluga uključuje odgovarajuće rutine i specijalizovane strukture podataka

UPRAVLJANJE MEMORIJSKIM PROSTOROM

Rutine za upravljanje prostorom eksternog memorijskog uređaja – jedinice diska. U ove rutine se ubrajaju:

1. **Uspostava adresnog prostora i fajl sistema**
   - formatiranje diska, niskog i visokog nivoa
   - kreiranje strukture podataka sa evidencijom slobodnog i zauzetog prostora diska (indeks slobodnih i zauzetih blokova)
2. **Održavanje strukture podataka sa evidencijom slobodnog i zautetog prostora na disku**
   - Alociranje i dealociranje prostora na zahtev drugih rutina OS
   - Reorganizacija (defragmentacija) slobodnog prostora na zah. administratora OS
3. **Arhiviranje, restauracija i oporavak sadržaja diska (backup, restore i recovery rutine)**

UPRAVLJANJE KATALOGOM

Katalog je hijerarhijska struktura direktorijuma (foldera) tipa stabla. Koren predstavlja korenski direktorijum i formira se automatski u postupku formatiranja diska. Svaki čvor u strukturi može biti direktorijum ili datoteka. Uvodi se pojam tekućeg direktorijuma.

Katalog može biti formiran na jednoj ili grupi jedinica diskova s direktnim pristupom. Takođe, katalog dozvoljava relativno i apsolutno referenciranje čvorova u strukturi.

Rutine za upravljanje katalogom:

1. kreiranje, brisanje, preimenovanje i prevezivanje direktorijuma u strukturi
2. izlistavanje i pretraživanje sadržaja direktorijuma i datoteka
3. kreiranje i brisanje datoteka u direktorijumu upotrebom sistemskih poziva
4. preimenovanje, kopiranje i premeštanje datoteka u strukturi
5. dodela i ukidanje prava pristupa nad direktorijumima i datotekama (konkretnim korisnicima ili dodelom uloga na nivou OS)
6. izmena atributa datoteka

UPRAVLJANJE FIZIČKOM RAZMENOM PODATAKA

Rutine za upravljanje fizičkom razmenom podataka (rutine fizičkog U/I, U/I supervizor, drajveri uređaja) zadužene su za upravljanje razmenom blokova kroz U/I podsistem, između kontrolera diska i operativne memorije.

Rutine za upravljanje fizičkom razmenom podataka:

1. Inicijalizacija fizičkog prenosa podataka jednog fizičkog bloka i zadavanje parametra prenosa
2. Razmena poruka sa programima za fizički prenos podataka, koji se realizuje kao CPU ili DMA prenos
3. Prosleđivanje statusa uspešnosti obavljenog prenosa podataka

OBEZBEĐENJE VEZE PROGRAMA I DATOTEKE

Upravljanje strukturama podataka o:

A. **Eksternim memorijskim uređajima**
B. **Datotekama na eksternom memorijskom uređaju**
C. **Upotrebi datoteka u aplikativnim programima**
D. **Datotekama u operativnoj upotrebi**

TABELA OS - Struktura podataka za opis nekog resursa kojim OS operativno upravlja. Gorenavedene strukture podataka su jednim delom predstavljene putem tabela OS.

1. ***TABELA UREĐAJA***
2. ***SISTEMSKA TABELA DATOTEKE***
3. ***ALOKACIONA TABELA DATOTEKE***
4. ***TABELA LOGIČKIH IMENA DATOTEKE***
5. ***TABELA PROCESA***
6. ***TABELA OTVORENIH DATOTEKA***
7. ***TABELA OPISA DATOTEKE***

[A] – Podaci o eksternom memorijskom uređaju

OS održava na jedinici diska strukture podataka o:

- proizvođačkim karakteristikama same disk jedinice (naziv, adresa i tip uređaja, ukupan broj cilindara, staza po cilindru i sektora po stazi)
- numeraciji sektora
- ispravnim i neispravnim sektorima
- zamenskim sektorima za neispravne sektore
- definisanom kapacitetu bloka
- korenskom direktorijumu i sistemu kataloga
- slobodnom prostoru

TABELA UREĐAJA – TU

Zapis sa podacima neophodnihm za korišćenje uređaja. Formira se prilikom pokretanja OS ili montiranja uređaja. Koriste je rutine za upravljanje fizičkim U/I (drajveri)

Sadrži:

1. Podatke o karakteristikama uređaja
2. Adrese rutina (drajvera) za upravljanje fizičkim U/I za dati uređaj

[B] – Podaci o datotekama na eksternom memorijskom uređaju

OS održava na jedinici diska strukture podataka o datotekama razmeštenih po sistemu kataloga.

SISTEMSKA TABELA DATOTEKE – STD

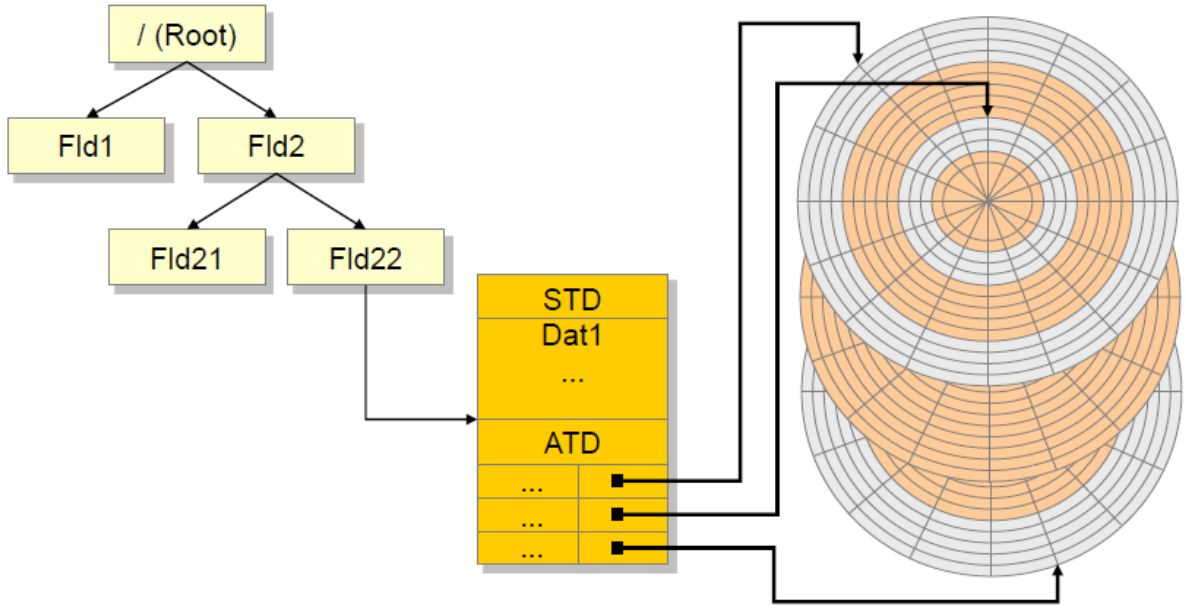
Predstavlja trajan zapis o datoteci koji održava OS. Formira se prilikom kreiranja datoteke, uništava prilikom brisanja a modifikuje pri izmeni sadržaja datoteke. Učitava se u operativnu memoriju kada se datoteka operativno koristi. Spregnuta je sa sistemom kataloga, adresirana iz pripadajućeg direktorijuma.

Sadržaj:

1. naziv datoteke
2. ekstenzija i verzija datoteke
3. vrsta datoteke (bin/txt)
4. podaci o vlasniku i ovlašćenjima za korišćenje datoteke
5. veličina datoteke
6. datum i vreme kreiranja ili poslednje modifikacije sadržaja
7. podaci o ostalim atributima (sistemska, skrivena datoteka, itd.)
8. **alokaciona tabela datoteke**

ALOKACIONA TABELA DATOTEKE – ATD

Predstavlja mapu alociranog prostora diska za datoteku i služi za vođenje evidencije o alociranom prostoru datoteke. Sadrži niz parova tipa (pokazivač, broj blokova) gde je pokazivač trojka (c, t, s) tj. adresa početka zone alociranog prostora, a broj blokova ukazuje na veličinu zone iskazanu brojem fizičkih blokova. Svako alociranje novog prostora izaziva formiranje novog para u nizu, a svako dealociranje nepotrebnog prostora izaziva brisanje para iz niza.



[C] – Podaci o upotrebi datoteka u aplikativnim programima

Omogućavaju vezu između aplikativnih programa I OS. Formira ih kompajler a koristi I dopunjava OS na osnovu specifikacija upotrebe datoteka u programu. Nalaze se u delu operativne memorije rezervisanom za aplikativni program.

Sadržaj ovih podataka može zavisiti od nivoa usluga OS – mogu biti uključeni podaci vezani za format sloga ili bloka datoteke, načine upotrebe i načine pristupa podacima datoteke.

TABELA LOGIČKIH IMENA DATOTEKA – TLI

Sastoji se od niza parova (ime datoteke, pokazivač) gde je indeks niza redni broj specificirane datoteke (fajl deskriptor, 0,1 ili 2 vrednost), ime datoteke ukazuje na povezani naziv datoteke iz STD a pokazivač ukazuje na zapis sa podacima o otvorenoj datoteci u TOD.

[D] – Podaci o datotekama u operativnoj upotrebi

Obezbeđuju uvezivanje podataka sadržanih u TU, STD I TLI. Formira ih I koristi OS kada je potrebno operativno korišćenje datoteke iz aplikativnog programa (priprema datoteke za operativno korišćenje podataka od strane aplikativnog programa tj. procesa OS). Nalaze se u sistemskom delu operativne memorije kojim upravlja OS.

TABELA PROCESA – TP

Sadrži id oznaku procesa I neophodne podatke o procesu, kreiranom na osnovu aplikativnog programa. Takođe, uključuje podatke o trenutnom stanju procesa na nivou OS I podatke iz TLI

> **TABELA OTVORENIH DATOTEKA PROCESA – TOP**
>
> Predstavlja TLI u sistemskom delu operativne memorije, sadrži pokazivače prema zapisima o otvorenim datotekama tj. tabeli otvorenih datoteka (TOD).

TABELA OTVORENIH DATOTEKA – TOD

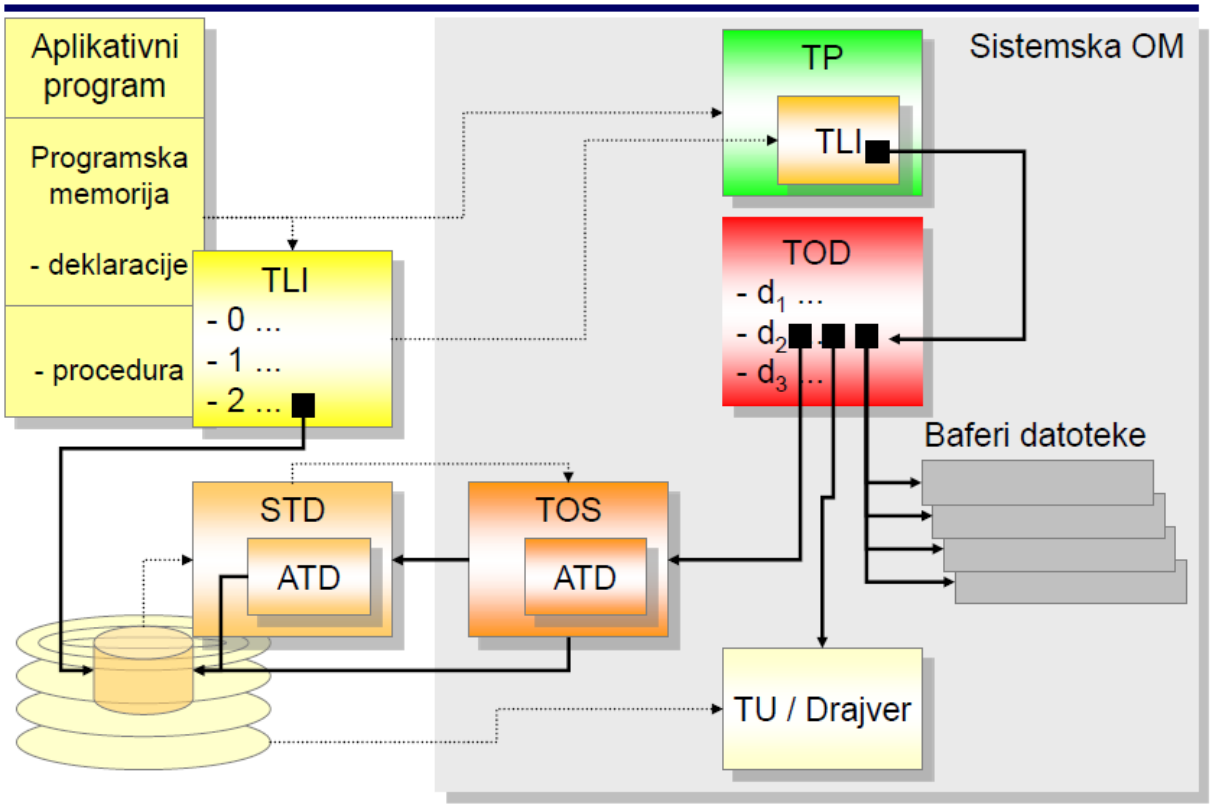
Predstavlja niz zapisa o otvorenim datotekama u celom sistemu. Svako otvaranje datoteke stvara jedan zapis u TOD, a indeks niza označava redni broj otvorene datoteke u sistemu. Svaki zapis sadrži podatke neophodne da bi se razmena podataka između aplikativnog programa i datoteke obavljala uspešno.

Sadržaj zapisa:

- mogući načini korišćenja datoteke
- pokazivač na tekuću poziciju u datoteci (tekući pokazivač)
- niz pokazivača prema rezervisanim sistemskim baferima
- veze između blokova i sistemskih bafera, u obliku (rbr_bloka, status, bafer)
  - rbr_bloka – redni broj učitanog bloka u sistemski bafer
  - status – indikator izmene sadržaja bloka nakon poslednjeg učitavanja
  - bafer – oznaka sistemskog bafera u koji je blok učitan
- pokazivač prema TU / drajveru uređaja u tabeli drajvera
- pokazivač na tabelu opisa datoteke (TOS)

TABELA OPISA DATOTEKE – TOS

Predstavlja prekopirani sadržaj STD zajedno sa ATD u operativnu memoriju. Ažurira se tokom upotrebe datoteke, a ažurirani sadržaj se prenosi u STD pri završetku rada sa datotekom.



SISTEMSKI POZIVI

Sistemski pozivi su pozivi rutina OS za upravljanje datotekama.

A. Pružaju usluge niskog nivoa tj. obezbeđuju:

   1. **POGLED NA DATOTEKU KAO NIZA BAJTOVA** – razmena podataka između aplikativnog programa i datoteke, organizovanih kao nizovi bajtova
   2. **UPRAVLJANJE BLOKOVIMA DATOTEKE** – grupisanje nizova bajtova u blokove i razmena kompletnih blokova između diska i sistemskih bafera u operativnoj memoriji
   3. **UPRAVLJANJE SISTEMSKIM BAFERIMA** – nalozi za rezervisanje i otpuštanje bafera kao i evidencija smeštanja blokova u bafere
   4. **NEZAVISNOST APLIKATIVNOG PROGRAMA OD FIZIČKIH KARAKTERISTIKA DISKA** – transformacija rednog broja bajta u redni broj bloka i rednog broja bloka u apsolutnu adresu bloka na disku.

B. Vode računa o karakteristikama datoteke – početku i kraju datoteke, kao i indikatoru aktuelnosti.

C. podržavaju **REDOSLEDNI PRISTUP** bajtovima datoteke pri operacijama učitavanja i zapisivanja. Oni zahtevaju zadavanje ukupnog broja bajtova za operaciju i automatski održavaju vrednost tekućeg pokazivača.

D. podržavaju i **DIREKTNI PRISTUP** bajtovima datoteke pri operacijama pozicioniranja. Oni zahtevaju zadavanje vrednosti tekućeg pokazivača tj. rednog broja bajta u odnosu na koji započinje sledeća operacija.

E. Preuzimaju parametre poziva iz pozivajućeg okruženja aplikativnog programa ili okruženja zaduženog za pružanje usluga visokog nivoa.

F. Prosleđuju u pozivajuće okruženje informacije o statusu izvršenja sistemskog poziva što je osnova za obradu izuzetka. *Izuzetak je događaj koji izaziva prekid normalnog toga obrade podataka.*

TIPOVI SISTEMSKIH POZIVA

1. **create** - kreiranje datoteke
2. **open** - otvaranje datoteke
3. **read** - učitavanje dela sadržaja datoteke
4. **write** - zapisivanje podataka u datoteku
5. **seek** - pozicioniranje na željenu lokaciju
6. **close** - zatvaranje datoteke
7. **sync** - pražnjenje izmenjenih bafera
8. **delete** - brisanje i uništavanje datoteke
9. **truncate** - brisanje sadržaja datoteke
10. **stat** - preuzimanje informacija o datoteci

1. **CREATE** – sistemski poziv za kreiranje datoteke na osnovu zadatih parametara

   Kreiranje potpuno nove datoteke i otvaranje za pisanje - **ZADACI**:

   - Proverava sve uslove neophodne za kreiranje datoteke (prostor, putanja, prava)
   - Alocira inicijalni prostor za novu datoteku na disku
   - Kreira STD sa ATD
   - Formira zapis u direktorijumu i uvezuje ga sa STD
   - Vraća podatak o uspešnosti operacije I fajl deskriptor

2. **OPEN** – sistemski poziv za otvaranje datoteke na osnovu zadatih parametara

   Priprema datoteke za operativnu upotrebu:

   - U operativnoj upotrebi datoteke zahteva se pristup podacima u STD I TLI
   - Da bi se izbeglo repetitivno pristupanje disku pogodno je smestiti te podatke u operativnu memoriju

   NAČINI OTVARANJA DATOTEKE:

   a. Otvaranje nepostojeće datoteke
   b. Otvaranje postojeće datoteke
   c. Otvaranje u režimu dozvole čitanja I pisanja
   d. Otvaranje u režimu ekskluzivnog čitanja sadržaja
   e. Otvaranje u režimu ekskluzivnog pisanja sadržaja

   ZADACI:

   - Ukoliko je specificiran zahtev kreiranja nove datoteke, sprovođenje poziva tipa CREATE
   - U slučaju zahteva za otvaranje postojeće datoteke, provera ostvarenosti potrebnih uslova (prostor, putanja, naziv datoteke) I ovlašćenja za izvođenje operacije OPEN
   - Prenos sadržaja STD u operativnu memoriju, kreiranje TOS I uvezivanje sa STD
   - Formiranje zapisa u TOD I uvezivanje sa TOS I TLI u TP (povezivanje fizičkog imena I logičke oznake datoteke u TLI, I povezivanje adresa odgovarajućih programa za opsluživanje sistemskih poziva za razmenu podataka)
   - Inicijalizacija vrednosti tekućeg pokazivača
   - Inicijalno rezervisanje sistemskih bafera (upisivanje pokazivača na bafere u odgovarajući zapis u TOD I punjenje bafera početnim blokovima datoteke)
   - Predaja pozivajućem okruženju podataka o uspešnosti operacije I fajl deskriptor

UPRAVLJANJE SISTEMSKIM BAFERIMA

REZERVISANJE I OTPUŠTANJE BAFERA DATOTEKE – Vrši se korišćenjem servisa jezgra OS. Otvorenoj datoteci mora biti dodeljen najmanje jedan bafer.

EVIDENTIRANJE SMEŠTANJA BLOKOVA U BAFERE – u zapisu iz TOD, ili u referenciranoj strukturi podataka koju održava jezgro OS

TEHNIKE REZERVISANJA I OTPUŠTANJA BAFERA – može biti fiksna (statička) ili dinamička dodela bafera datoteci.

- FIKSNA DODELA – fiksni broj bafera se dodeljuje pri otvaranju a svi baferi se otpuštaju pri zatvaranju datoteke
- DINAMIĆKA DODELA – svi sistemski baferi OS-a su na raspolaganju svim otvorenim datotekama, a baferi se dodeljuju I otpuštaju prema potrebi

VIŠESTRUKO OTVARANJE ISTE DATOTEKE

Nije dozvoljeno kod nekih OS. Datoteka se ekskluzivno otvara I zaključava od strane jednog procesa. Za svaku datoteku, u TOD može postojati naviše jedan zapis. Ostali procesi mogu otvoriti samo datoteku za čitanje. Ovim se očuvava konzistentnost podataka u višekorisničkom režimu rada ali se bitno snižava stepen mogućeg paralelizma u obradi I korišćenju podataka datoteke.

Kod nekih OS je dozvoljeno. Datoteka se može otvoriti od strane više različitih procesa istovremeno ili od strane jednog procesa više puta. Svako otvaranje datoteke formira poseban zapis u TOD koji je uvezan sa odgovarajućim zapisom u TLI iz TP. Dobija se visok stepen paralelizma u obradi podataka ali konzistentnost ne može biti očuvana.

3. **READ** – sistemski poziv za učitavanje niza bajtova iz otvorene datoteke na osnovu zadatih parametara

Stvarni prenos dela sadržaja datoteke u pozivajuće okruženje, zadati broj bajtova od pozicije tekućeg pokazivača.

ZADACI:

- Provera da li je datoteka otvorena I da li dozvoljava čitanje
- Izračunavanje rednog broja prvog bloka I dužine niza izvornih blokova u kojima se nalaze traženi podaci
- Provera da li se traženi blokovi već nalaze u baferima. Ako se ne nalaze, biraju se baferi za smeštanje blokova, izračunavaju se apsolutne adrese traženih blokova I vrši se incijalizacija fizičkog prenosa blokova sa diska
- Prenos traženog sadržaja iz bafera u ciljnu promenljivu
- Uvećanje vrednosti tekućeg pokazivača
- Predaja pozivajućem okruženju podataka o uspešnosti I broju preuzetih bajtova

4. **WRITE** – sistemski poziv za zapisivanje niza bajtova u otvorenu datoteku

   Stvarni prenos niza bajtova iz pozivajućeg okruženja u datoteku, zadati broj bajtova od pozicije tekućeg pokazivača.

ZADACI:

- Provera da li je datoteka otvorena I da li dozvoljava pisanje
- Izračunavanje rednog broja prvog bloka I dužine niza ciljnih blokova u koje treba smestiti izvorne podatke
- Provera da li je potrebno imati dopremljen prethodni sadržaj ciljnih blokova. Ako jeste, proverava se da li se traženi blokovi nalaze u baferima već.
- Ako se ne nalaze u baferima, biraju se baferi za smeštanje blokova, izračunavaju se apsolutne adrese traženih blokova I vrši se inicijalizacija fizičkog prenosa blokova sa jedinice diska
- Prenos traženog sadržaja iz izvorne promenljive u bafere
- Uvećanje vrednosti tekućeg pokazivača
- Predaja pozivajućem okruženju podataka o uspešnosti I broju preuzetih bajtova

5. **SEEK** – pozicioniranje na željenu lokaciju

   Upravljanje sadržajem tekućeg pokazivača u cilju obezbeđenja direktnog pristupa željenom bajtu datoteke.

ZADACI:

- Provera da li je datoteka otvorena
- Postavljanje nove vrednosti tekućeg pokazivača pomeranjem za zadati broj bajtova ili na referentnu tačku
- Predaja pozivajućem okruženju podataka o uspešnosti izvedene operacije

6. **CLOSE** – sistemski poziv za zatvaranje otvorene datoteke

   Uredan prestanak operativne upotrebe datoteke. Garantuje da će datoteka ostati memorisana na disku u konzistentnom stanju. Zatvaranje datoteke nastaje ili automatski iili eksplicitno.

ZADACI:

- Oslobađanje sadržaja zauzetih bafera, modifikovanog sadržaja (buffer flushing) I iniciranje fizičkog prenosa blokova na jedinicu diska
- Oslobađanje svih zauzetih bafera I vraćanje OS-u
- Prenos sadržaja TOS sa ATD na jedinicu diska, ažuriranje sadržaja STD sa ATD
- Uništavanje TOS I odgovarajućeg zapisa u TOD, raskidanje svih uspostavljenih veza prema TLI, TP I STD
- Predaja pozivajućem okruženju podataka o uspešnosti izvedene operacije

7. **SYNC** – sistemski poziv za pražnjenje dirty bafera, izvršava se na zahtev ili automatski u intervalima

8. **DELETE** – sistemski poziv za brisanje imena I uništavanje datoteke

   ZADACI:

   - Dealociranje prostora datoteke na disku
   - Uništavanje STD sa ATD
   - Uništavanje zapisa o datoteci u direktorijumu
   - Vraćanje podataka o uspešnosti operacije

9. **TRUNCATE** – sistemski poziv za brisanje sadržaja datoteke

   Dealocira prostor datoteke na disku ali zadržava STD

METODE PRISTUPA

Koriste ili uključuju usluge niskog nivoa izabranog OS. Servisi metoda pristupa mogu biti ugrađeni u OS, programski jezik sa bibliotekama ili SUBP.

1. Obezbeđuju **poglede na LSP datoteke kao različitih struktura nad skupom slogova ili blokova**, a obavezno kao niza blokova. Takođe, obezbeđuju preslikavanje ovih pogleda u FSP niza fizičkih blokova.
2. Obezbeđuju **izgradnju specijalnih pomoćnih struktura** za poboljšanje efikasnosti obrade podataka.
3. Obezbeđuju **traženja**, zasnovana na vrednostima podataka.

Zahtevaju razrešavanje pitanja

- organizovanja i memorisanja polja, slogova i blokova
- načina adresiranja i načina memorisanja logičkih veza
- mogućih vrsta usluga na nivou sloga ili bloka
- podrške različitih vrsta organizacije datoteka
- podrške opštih postupaka upravljanja sadržajem datoteka

4 – Metode pristupa i organizacija datoteka

OSNOVNA STRUKTURA DATOTEKE

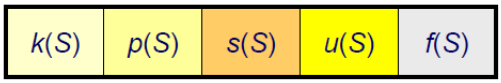
DATOTEKA KAO STRUKTURA SLOGOVA

Organizovana nad tipom sloga kao linearnom strukturom atributa.

FORMAT SLOGA – pravila za struktuiranje i interpretaciju sadržaja sloga

Opšta struktura sloga datoteke kao FSP – Uključuje podatke iz LSP i podatke o organizaciji FSP na eksternom memorijskom uređaju. Svaki slog predstavlja niz polja sa vrednostima atributa.

- k – polje vrednosti primarnog ključa
- p – polje vrednosti ostalih atributa
- s – polje statusa sloga, indikator aktuelnosti sloga u LSP
- u – polje pokazivača za memorisanje veza u LSP
- f – konrolna polja kod slogova promenljive dužine



Datoteka se često posmatra kao linearna struktura slogova uređena u rastućem ili opadajućem redosledu vrednosti primarnog ključa. Redosled polja u formatu sloga ne mora biti isti kao u opštoj strukturi sloga. Format polja sloga je uslovljen specifikacijom domena odgovarajućeg atributa, odnosno primenjenim tipom podatka. Polja u slogovima mogu biti konstantne ili promenljive dužine pri čemu se mora voditi informacija o granicama polja.

VRSTE SLOGOVA PREMA DUŽINI

SLOGOVI KONSTANTNE DUŽINE – sva polja u svakom slogu su konstantne dužine i nije potrebno memorisati informaciju o granicama sloga. Pojavljuju se u praksi, jednostavniji su za pristupanje podacima i ažuriranje, lakša i preciznija procena performansi obrade podataka ali manja efikasnost upotrebe memorijskog prostora.

SLOGOVI PROMENLJIVE DUŽINE – postoji barem jedno polje promenljive dužine u slogu i potpuno je memorisati informaciju o granicama sloga. Izuzetno često se pojavljuju u praksi, teže se pristupa podacima i ažurira, teško je proceniti performanse ali je veća efikasnost upotrebe memorijskog prostora.

VRSTE SLOGOVA PREMA PONAVLJANJU VREDNOSTI

SLOGOVI S PONAVLJAJUĆIM GRUPAMA – višestruko pojavljivanje vrednosti atributa u jednom slogu, moraju uvek biti slogovi varijabilne dužine

SLOGOVI BEZ PONAVLJAJUĆIH GRUPA – nije dozvoljeno višestruko pojavljivanje vrednosti atributa

Polja pokazivača u strukturi sloga predstavljaju adrese lokacija u memorijskom prostoru. Postoje 3 vrste adresa lokacija:

- **APSOLUTNA (MAŠINSKA) ADRESA** – struktuirana prema adresnom prostoru diska. Praktično se ne koristi jer stvara zavisnost od fizičkih karakteristika uređaja. Ne zahteva transformaciju.
- **RELATIVNA ADRESA** – predstavlja redni broj lokacije. Vrlo često se koristi u organizaciji datoteka jer obezbeđuje nezavisnost od fizičkih karakteristika uređaja. Zahteva jednu ili više transformacija do apsolutne adrese.
- **SIMBOLIČKA (ASOCIJATIVNA) ADRESA** – vrednost ključa. Često se koristi u organizaciji datoteka. Zahteva transformaciju u relativnu adresu na nivou metode pristupa.

DATOTEKA KAO NIZ BLOKOVA

Linearna struktura blokova datoteke gde svaki blok obuhvata niz slogova datoteke.

Strogo tipizovana datoteka sa pridruženom semantikom, organizovana kao struktura nad skupom slogova koja se preslikava u strukturu na skupom blokova (koja se dalje preslikava u strukturu nad nizom bajtova pa fizičkih blokova)

BLOK – Logički blok kao organizovana jedinica podataka predstavlja niz slogova i ima konstantni kapacitet. Najčešće prestavlja celobrojni umnožak kapaciteta fizičkog bloka. Uobičajeno, jedan blok predstavlja niz od 2^n fizičkih blokova.

OPŠTA STRUKTURA BLOKA – Blok sadrži adresu bloka (relativnu), relativnu adresu rednog sloga u bloku i faktor blokiranja (broj slogova u bloku). Takođe sadrži i zaglavlje bloka koje nije obavezno i koje može sadržati podatke vezane za FSP datoteke kao npr broj slogova u bloku ili indeks na početke slogova.

VRSTE BLOKOVA

- **BLOKOVI SA SLOGOVIMA PROMENLJIVE DUŽINE** – više slogova može se smestiti u jedan blok, dozvoljeno je da veličina jednog sloga premaši kapacitet bloka, pa se tada vrši ulančavanje blokova jednog sloga
- **BLOKOVI SA SLOGOVIMA KONSTANTNE DUŽINE** – homogena struktura bloka i datoteke, svaki blok datoteke sadrži isti broj slogova. Pri korišćenju blokova sa slogovima konstantne dužine, moguć je proračun potrebnog kapaciteta datoteke.

ZAGLAVLJE DATOTEKE - Potrebno proširenje osnovne strukture datoteke. Uvodi se specijalni slog na početku datoteke sa podacima o organizaciji datoteke i formatu bloka i sloga datoteke.

OZNAKA KRAJA DATOTEKE:

1. Uvođenjem specijalnog sloga za oznaku kraja datoteke
2. Uvođenjem specijalne oznake kraja u polje pokazivača
3. Vođenjem posebne evidencije zauzetosti prostora
4. Kraj datoteke je kraj prostora dodeljenoj datoteci

METODA PRISTUPA

Paket programa (rutina) za podršku usluga visokog nivoa. OBEZBEĐUJE:

1. **Upravljanje strogo struktuiranim datotekama** - organizacijom i memorisanjem polja, slogova i blokova.
2. **Upravljanje baferima metode pristupa** – viši nivo baferisanja u odnosu na nivo sistemskih bafera
3. **Podrška različitih vrsta organizacija datoteke** – različitih načina memorisanja logičkih veza i adresiranja, vođenje brige o kategorijama, podrška izgradnje specijalnih pomoćnih struktura za poboljšanje efikasnosti obrade podataka.
4. **Podrška opštih postupaka upravljanja sadržajem datoteka** (kreiranje, traženje, ažuriranje i reorganizacija)
5. **Koristi ili uključuje usluge niskog nivoa** izabranog OS u zavisnosti od mesta i načina implementacije metode pristupa
6. **Obezbeđuje nezavisnost aplikativnog programa od usluga niskog nivoa OS** – preslikavanje strogo struktuirane datoteke u FSP niza fizičkih blokova i transformacije relativne adrese sloga ili bloka datoteke u relativnu adresu bajta ili fizičkog bloka.

UPRAVLJANJE STROGO STRUKTUIRANIM DATOTEKAMA

- Podrška organizacije slogova i polja konstantne ili promenljive dužine.
- Podrška različitih tipova podataka i kodnih rasporeda
- Konverzije podataka iz tipa podataka programske promenljive u tip podataka atributa datoteke i obrnuto, ili iz tipa podataka atributa strogo struktuirane datoteke u niz bajtova i obrnuto.

USLUGE RAZMENE PODATAKA SA APLIKATIVNIM PROGRAMOM

NA NIVOU SLOGA – grupisanje slogova u blokove pri upisu, i rastavljanje bloka na slogove pri čitanju podataka. Održavanje tekućeg pokazivača kao relativne adrese sloga i transformacija pokazivača u oblik (*rb. bloka, rb. sloga*).

NA NIVOU BLOKA – razmena sadržaja kompletnih logičkih blokova između aplikativnog programa i datoteke, održavanje tekućeg pokazivača kao relativne adrese bloka u obliku rb. bloka u datoteci.

USLUGE PRISTUPA PODACIMA IZ APLIKATIVNIH PROGRAMA

- **SEKVENCIJALNI (REDOSLEDNI) PRISTUP** – automatski održavaju vrednost tekućeg pokazivača pri operacijama učitavanja i upisa podataka
- **DIREKTNI PRISTUP** – zahtevaju eksplicitno zadavanje vrednosti tekućeg pokazivača (rb. sloga ili bloka datoteke pri operacijama pozicioniranja)
- **DINAMIČKI (KOMBINOVANI) PRISTUP** – kombinacija direktnog i sekvencijalnog

POZIVI RUTINA METODE PRISTUPA

Preuzimaju parametre poziva iz pozivajućeg okruženja (putanja, naziv, oznaka itd.) i prosleđuju u pozivajuće okruženje informacije o statusu izvršenja rutine. U pozive spadaju:

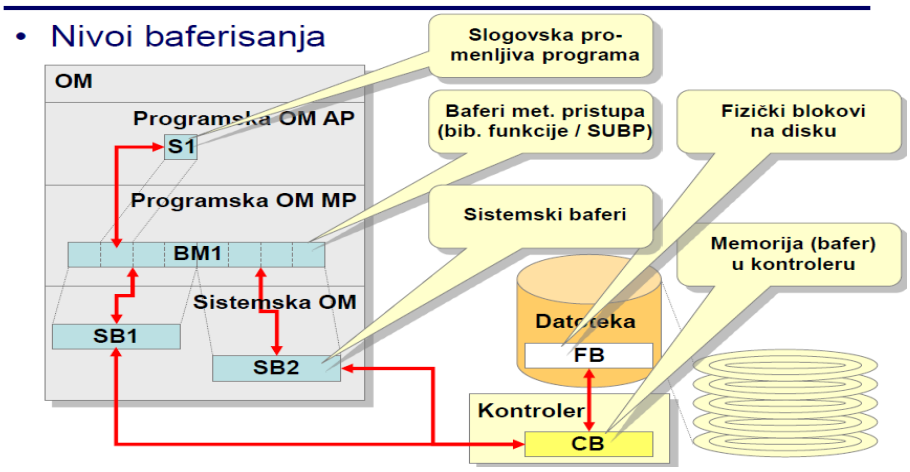
- Otvaranje i zatvaranje datoteke
- Učitavanje i ispis sadržaja sloga ili bloka
- Pozicioniranje na slog ili blok datoteke
- Ispitivanje statusa datoteke
- Kreiranje i brisanje datoteke

Upravljanje baferima metode pristupa

Okruženje u kojem je implementirana metoda pristupa brine o zadacima upravljanja baferima (alociranje, dealociranje, vođenje evidencije o sadržaju itd.)

Postoje tri nivoa baferisanja podataka datoteke u operativnu memoriju:

1. **NIVO SISTEMSKIH BAFERA** – upravlja OS
2. **NIVO BAFERA METODE PRISTUPA** – kojim upravlja okruženje u kojem je metoda implementirana
3. **NIVO LOKACIJA PROMENLJIVIH U APLIKATIVNOM PROGRAMU** – upravlja aplikativni program



Neki servisi metode pristupa mogu biti implementirani direktno u aplikativnom programu

Okruženja koja uključuju metode pristupa su:

- **OPERATIVNI SISTEM**
- **PROGRAMSKI JEZIK SA PRIDRUŽENIM BIBLIOTEKAMA FUNKCIJA**
- **SISTEM ZA UPRAVLJANJE BAZAMA PODATAKA - SUBP**

PARAMETRI ORGANIZACIJE DATOTEKA

ORGANIZACIJA PODATAKA je

- projekat logičke strukture obeležja LSO
- projekat i implementacija FSP
- sa ciljem da se zadovolje korisnički zahtevi i uslovi za efikasnu obradu podataka

REZULTAT: Sistem baze podataka ili sistem datoteka

PROJEKAT I IMPLEMENTACIJA FSP

1. izbor načina dodele lokacija slogovima
2. izbor načina memorisanja logičkih veza između slogova u LSP
3. projektovanje osnovnih struktura podataka
4. projektovanje pomoćnih struktura podataka
5. proračun i rezervisanje potrebnog prostora na eksternim memorijskim uređajima
6. smeštanje slogova sa vezama na eksterne memorijske uređaje
7. proračun, praćenje i analiza performansi postupaka obrade podataka

ORGANIZACIJA DATOTEKE

Projektovanje LSO svodi se na projektovanje tipa entiteta N(Q,C) tj. tipa sloga. Izbor vrste organizacije datoteke zavisi od vrednosti parametara tj. **načina dodele lokacija slogovima**, načina evidentiranja slobodnog i zauzetog prostora i **načina memorisanja logičkih veza između slogova** u LSP.

NAČINI DODELE LOKACIJA SLOGOVIMA – DLS

1. svaki novi slog upisuje se na kraj datoteke, kao fizički susedan u odnosu na poslednji slog datoteke
2. svaki novi slog dobija prvu slobodnu lokaciju iz spregnute linearne strukture slobodnih lokacija
3. svaki novi slog dobija slobodnu lokaciju čija relativna adresa predstavlja funkciju vrednosti ključa

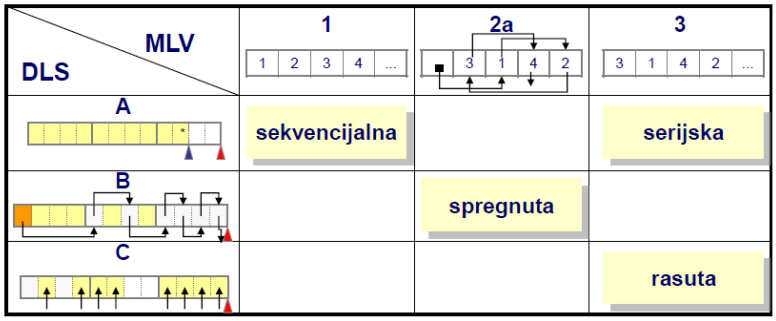
NAČINI MEMORISANJA LOGIČKIH VEZA IZMEĐU SLOGOVA U LSP – MLV

1. **Fizičkim pozicioniranjem** – logički susedni slogovi se smeštaju u fizički susedne lokacije
2. **Pomoću pokazivača kao relativnih adresa –** polja pokazivača koja memorišu relativnu adresu logički susednog sloga mogu biti ugrađena u osnovnu ili u pomoćnu strukturu podataka.
3. **Logičke veze se ne memorišu –** u FSP ne postoje podaci o logički susednim slogovima

VRSTE ORGANIZACIJE DATOTEKA

OSNOVNE ORGANIZACIJE – FSP nad skupom slogova organizovana je u jednoj memorijskoj zoni. U njih spadaju serijska, sekvencijalna, spregnuta i rasuta datoteka sa jedinstvenim memorijskim prostorom (direktna, relativna, statička rasuta i dinamička rasuta).

SLOŽENE ORGANIZACIJE – dobijaju se kombinovanjem osnovnih organizacija gde FSP uključuju barem dve memorijske zone. U njih spadaju rasute datoteke sa zonom prekoračenja, statičke i dinamičke indeksne datoteke



RASUTE (HASH) SA ZONOM PREKORAČENJA

Primarna zona – osnovna struktura, rasuta organizacija

Zona prekoračenja – nastavak osnovne strukture, serijska ili spregnuta organizacija

STATIČKE INDEKSNE (INDEKS-SEKVENCIJALNE)

Primarna zona – osnovna struktura, sekvencijalna organizacija

Zona prekoračenja – nastavak osnovne strukture, spregnuta organizacija

Zona indeksa – pomoćna struktura, spregnuta organizacija u vidu stabla traženja

DINAMIČKE INDEKSNE (sa B-stablom)

Primarna zona – osnovna struktura, serijska ili spregnuta organizacija

Zona indeksa – pomoćna struktura, spregnuta organizacija u vidu B-stabla

Navedene vrste organizacije pojavljuju se u praksi kao:

FIZIČKE ORGANIZACIJE DATOTEKA – u sistemima datoteka, svaka datoteka u sistemu datoteka pojavljuje se kao jedna ili više posebnih OS datoteka

FIZIČKE ORGANIZACIJE TABELA – u SUBP gde svaka tabela BP može biti distribuirana u više datoteka podataka kojim upravlja SUBP i gde u jednoj datoteci može biti smešteno više tabela BP

OPŠTE PROCEDURE NAD DATOTEKAMA

Vrste postupaka tj. operacija nad LSP datoteke: formiranje, pristupanje, traženje, pretraživanje, obrada, ažuriranje i reorganizacija datoteke.

- **FORMIRANJE DATOTEKE** – postupak kreiranja FSP datoteke sa smeštanjem slogova na disk saglasno projektovanoj organizaciji, na osnovu sadržaja neke druge strukture podataka
- **PRISTUPANJE U DATOTECI** – postupak pozicioniranja na željenu lokaciju sloga ili bloka datoteke. Postoje više vrsti pristupa:
    1. **SEKVENCIJALNI PRISTUP** – automatsko održavanje relativne adrese tekućeg pokazivača, operacija se odnosi na neposredno susednu lokaciju u odnosu na lokaciju na kojoj je obavljena prethodna operacija
    2. **DIREKTNI PRISTUP** – eksplicitno zadavanje relativne adrese tekućeg pokazivača koji ukazuje na lokaciju nad kojom će se realizovati neka operacija
    3. **DINAMIČKI** – kombinacija sekvencijalnog i direktnog pristupa
- **TRAŽENJE U DATOTECI** – za zadatu vrednost argumenta traženja u stanju je da generiše i vrati u program indikaciju uspešnosti traženja, relativnu adresu mesta zaustavljanja traženja i sadržaj sloga na mestu zaustavljanja traženja. Primenjuje se da bi se utvrdilo postojanje sloga za upis, brisanje ili izmenu sloga.

    METODE TRAŽENJA NA OSNOVU VRSTE POSTUPKA

    1. **LINEARNO TRAŽENJE** – moguće u serijskim, sekvencijalnim i rasutim organizacijama
    2. **BINARNO TRAŽENJE** – isključivo moguće u sekvencijalnim organizacijama
    3. **TRAŽENJE PRAĆENJEM POKAZIVAČA** – moguće samo u spregnutim organizacijama
    4. **TRAŽENJE TRANSFORMACIJOM ARGUMENTA U ADRESU** – samo u rasutim org.

    METODE TRAŽENJA S OBZIROM NA PREDISTORIJU TRAŽENJA

    1. **TRAŽENJE SLUČAJNO ODABRANOG SLOGA** – ne zavisi od mesta zaustavljanja prethodnog traženja, moguće u svim organizacijama datoteka
    2. **TRAŽENJE LOGIČKI NAREDNOG SLOGA** – početna adresa predstavlja adresu na kojoj je zaustavljeno prethodno traženje, a svaka naredna adresa traženja može biti samo adresa logički narednog sloga
- **PRETRAŽIVANJE U DATOTECI** – za zadatu vrednost argumenta pretraživanja, u stanju je da generiše i vrati u program skup slogova ili skup adresa slogova koji zadovoljavaju logički uslov pretraživanja

- **AŽURIRANJE DATOTEKE** – postupak dovođenja LSP datoteke u sklad sa izmenjenim stanjem klase entiteta u realnom sistemu. Osnovne operacije su:
    1. UPIS NOVOG SLOGA – zahteva prethodno neuspešno traženje, može iziskivati premeštanje određenog broja postojećih slogova
    2. MODIFIKACIJA VREDNOSTI ATRIBUTA – zahteva prethodno uspešno traženje, zabranjena je izmena vrednosti obeležja primarnog ključa jer je po njemu uspostavljena organizacija
    3. BRISANJE SLOGA – zahteva prethodno uspešno traženje, može iziskivati premeštanje određenog broja drugih slogova. Brisanje može biti LOGIČKO (promena polja statusa sloga) ili FIZIČKO (oslobađanje memorijske lokacije sloga u memorijskom prostoru).
- **OBRADA DATOTEKA** – algoritamski iskazani niz operacija nad LSP jedne ili više datoteka, sa ciljem svrsishodne tranformacije podataka datoteka.

PODELA PREMA VRSTAMA PRIMENJENIH OPERACIJA U OBRADI

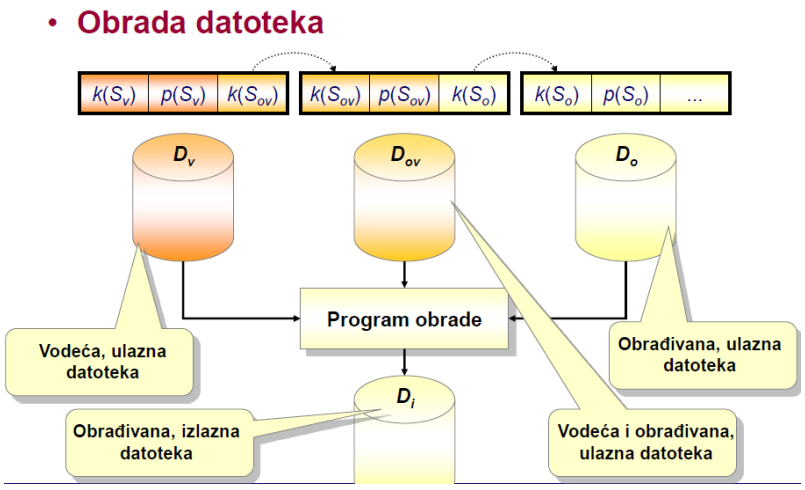
- ULAZNA DATOTEKA – datoteka u kojoj se isključivo vrše čitanja
- IZLAZNA DATOTEKA – datoteka u koju se isključivo zapisuju novi slogovi u obradi
- ULAZNO IZLAZNA DATOTEKA – vrši se i čitanje i modifikacija slogova

PODELA PREMA ULOZI U TRAŽENJIMA

1. **VODEĆA DATOTEKA** – datoteka koja isključivo generiše argumente traženja ili pretraživanja slogova tokom obrade, barem jedna ulazna datoteka u obradi mora biti vodeća
2. **OBRAĐIVANA DATOTEKA** – datoteka u kojoj se isključivo vrše traženja ili pretraživanja, na osnovu generisanih argumenata
3. **VODEĆA I OBRAĐIVANA** – datoteka sa obe uloge, vodeća za neku drugu obrađivanu i obrađivana u odnosu na neku vodeću

PODELA PREMA NAČINIMA TRAŽENJA SLOGOVA U OBRAĐIVANOJ DATOTECI

a. **DIREKTNA OBRADA** – u svakom narednom koraku obrade zahteva se traženje slučajno odabranog sloga
b. **REDOSLEDNA OBRADA** – u svakom narednom koraku obrade zahteva se traženje logički narednog sloga ili sekvencijalni pristup fizički susednoj lokaciji



- **REORGANIZACIJA DATOTEKE** – ponovno formiranje datoteke u cilju dovođenja u sklad FSP sa novim stanjem LSP zbog degradacija performansi rada sa datotekom.

  Organizacije koje traže povremenu reorganizaciju: sekvencijalna, spregnuta, statička rasuta i statička indeksna

  Organizacije koje ne traže povremenu reorganizaciju: serijska, indeksna sa B-stablom i dinamička rasuta.

PERFORMANSE OBRADE DATOTEKE

Mere podobnosti datoteke sa zadatom organizacijom da participira u obradi kao vodeća ili obrađivana, u redoslednoj ili direktnoj obradi.

IDEALNA ORGANIZACIJA DATOTEKE – zahteva tačno onoliko lokacija koliko sadrži slogova, najviše jedan pristup za traženje logički narednog i slučajno odabranog sloga, najviše jedan pristup za pretraživanje i ažuriranje i nikad ne zahteva reorganizaciju.

Izbor vrste organizacije datoteke predstavlja kompromisno rešenje, jer je nemoguće da jedna vrsta organizacije zadovolji sve zahteve. Favorizuju se željene mere performansi u odnosu na zauzeće memorijskog prostora.

Ukupno vreme traženja ili pretraživaja slogova zavisi od broja i vremena pristupa blokovima na jedinici diska, vremenom prenosa bloka sa diska u operativnu memoriju i broja i vremena upoređivanja argumenta sa vrednošću ključa. Iz tog razloga, ovi brojevi su dominantno opredeljeni vrstom organizacije datoteke.

Mere za ocenu performansi su:

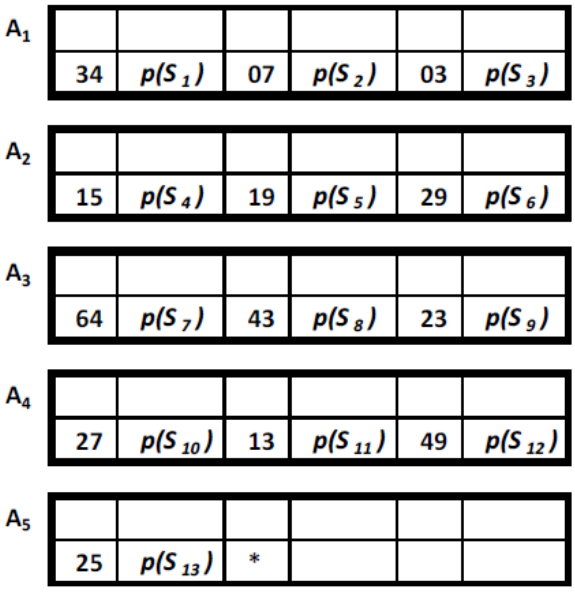
- Broj pristupa blokovima
- Broj upoređivanja argumenta i vrednosti ključa
- Srednji broj
- Broj u najgorem slučaju
- Traženje logički narednog sloga
- Traženje slučajno odabranog sloga
- Operacije ažuriranja
- Uspešna operacija
- Neuspešna operacija

5 – Serijska i sekvencijalna datoteka

SERIJSKA

OSNOVNA STRUKTURA

Slogovi su smešteni jedan za drugim u sukcesivne memorijske lokacije. Fizička struktura ne sadrži informacije o vezama između slogova logičke strukture datoteke. Ne postoji veza između vrednosti ključa sloga I adrese lokacije u koju je smešten. Redosled memorisanja slogova je najčešće prema hronološkom redosledu nastanka. Slogovi mogu a I ne moraju biti blokirani.



FORMIRANJE

OBUHVAT PODATAKA – proces sa zadatkom da obezbedi inicijalno memorisanje ispravnih podataka. Osnovna aktivnost je upis podataka na medijum, a izvršava ga operater ili specijalizovani softver sa odgovarajućim uređajima.

VERIFIKACIJA – postupak suštinske provere ispravnosti unetih podataka, može biti ručni ili automatizovan

FORMAT PROGRAM – Graficki program za obuhvat podataka. Sadrži opis formata dokumenta, raspored polja, pravila navigacije, opise I formatiranje sadržaja polja, specijalne kontrole sadržaja polja I dozvoljene operacije nad sadržajem polja.

Serijska datoteka se generiše najčešće u postupku obuhvata podataka. Vreme obavljanja obuhvata podataka može biti u realnom vremenu (na mestu I u trenutku nastanka podataka) ili naknadno (nakon određenog intervala vremena od nastanka podataka). Slogovi se formiraju prenosom podataka sa različitih izvora (dokumenta ili softveri I uređaji za očitavanje vrednosti u realnom vremenu) I upisuju se jedan za drugim u sukcesivne memorijske lokacije. Svaki novi slog upisuje se na kraj datoteke. Rezultat obuhvata podataka je neblokirana ili blokirana serijska datoteka.

TRAŽENJE SLOGA

Traženje logički narednog i slučajno odabranog sloga je jednako jer ne postoji funkcionalna veza između vrednosti ključa I adrese lokacije sloga. Primenjuje se metoda linearnog traženja – počinje od početka datoteke I pristupa sukcesivno memorisanim blokovima I slogovima.

Uspešno traženje - ***1 <= Ru <= B*** Neuspešno traženje - ***Rn = B*** Ukupan broj blokova – ***B = (N+1)/f***

OBRADA SERIJSKE DATOTEKE

Postoji 2 vrste obrade: **DIREKTNA** I **REDOSLEDNA**

- Može se koristiti kao vodeća u režimu direktne obrade
- Može se koristiti kao vodeća u redoslednoj obradi datoteke čiji ključ sadrži kao svoj strani ključ, kada je uređena saglasno neopadajućim vrednostima tog stranog ključa

Program koji vrši redoslednu obradu serijske datoteke učitava sukcesivne slogove vodeće datoteke. Svaki naredni slog vodeće datoteke sadrži logički narednu vrednost ključa obrađivane serijske datoteke. Te vrednosti ključa se koriste kao argumenti za traženje u serijskoj datoteci metodom linearnog traženja.

U režimu direktne obrade, sukcesivni slogovi vodeće datoteke sadrže slučajno odabrane vrednosti ključa obrađivane serijske datoteke. Traženje je takođe linearno.

Broj pristupa se ne razlikuje za slučaj direktne I redosledne obrade.

AŽURIRANJE SERIJSKE DATOTEKE

- **UPIS NOVOG SLOGA** – vrši se u prvu slobodnu lokaciju na kraju datoteke. Mora mu prethoditi jedno neuspešno traženje. Proces je jednostavan ali zahteva veliki broj pristupa.
- **BRISANJE POSTOJEĆEG SLOGA** – mora mu prethoditi jedno uspešno traženje. Najčešće je samo logičko tj. izmenom statusa aktuelnosti sloga. Fizičko brisanje bi zahtevalo veliki broj pristupa.
- **MODIFIKACIJA SADRŽAJA SLOGA** – mora mu prethoditi jedno uspešno traženje.

Očekivani broj pristupa za brisanje ili modifikaciju sadržaja sloga je ***Rd = Ru + 1***.

OBLASTI PRIMENE I KARAKTERISTIKE

- POGODNE KAO MALE DATOTEKE – kada mogu stati cele u OM, zbog velikog broja pristupa potrebnog za pronalaženje logički narednog ili slučajno odabranog sloga, jer druge vrste organizacije donose samo mala poboljšanja efikasnosti obrade malih datoteka

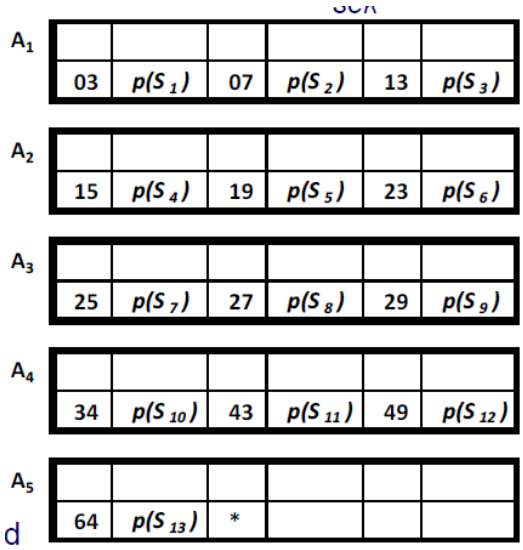
- Serijska organizacija podataka u kombinaciji sa indeksnim strukturama je veoma pogodna za direktnu obradu I osnovna je fizička struktura relacionih baza podataka.

- Serijska datoteka je rezultat obuhvata podataka I polazna je osnova za izgradnju datoteka sa drugim vrstama organizacije podataka.

SEKVENCIJALNA

OSNOVNA STRUKTURA

Slogovi su smešteni sukcesivno jedan za drugim. Logički susedni slogovi smeštaju se u fizički susedne memorijske lokacije. Postoji informacija o vezama između slogova logičke strukture podataka datoteke, ugrađena u fizičku strukturu.



Sekvencijalna datoteka realizovana je kao linearna logička struktura podataka, smeštanjem sloga sa većom vrednošću ključa u lokaciju sa većom adresom. Rastuće uređenje po vrednostima ključa – slog sa najmanjom vrednošću ključa smešta se u prvu lokaciju. Naziva se još I fizički sekvencijalnom organizacijom.

Veza između memorisanih vrednosti ključa I adresa lokacija nije ugrađena u strukturu datoteke I ne predstavlja bilo kakvu matematičku funkciju. Slogovi se smeštaju u blokove od po ***f >= 1*** slogova.

FORMIRANJE

Najčešće se vrši sortiranjem serijske datoteke saglasno rastućim ili opadajućim vrednostima ključa.

TRAŽENJE SLOGA

TRAŽENJE SLUČAJNO ODABRANOG SLOGA

Moguća primena metoda linearnog traženja, binarnog traženja ili traženja po m putanja. Nema praktičnog smisla ako je datoteka velika I smeštena na eksterni memorijski uređaj. Jedino ima praktičnog smisla ako je cela datoteka smeštena u operativnu memoriju. Nju u tom slučaju može predstavljati neka linearna struktura nad skupom slogova ili blok neke druge datoteke.

TRAŽENJE LOGIČKI NAREDNOG SLOGA

Vrši se metodom linearnog traženja. Počevši od prvog, fizički susedni blokovi se učitavaju u operativnu memoriju. U CPU se vrši upoređivanje argumenata traženja I vrednosti ključa sukcesivnih slogova dok se traženi slog ne pronađe, dok argument traženja ne postane manji od vrednosti ključa sloga, ili dok se ne dođe do kraja datoteke. Traženje novog, logički narednog sloga započinje od sloga na kojem se prethodno traženje zaustavilo tj. od tekućeg sloga datoteke.

OBRADA SEKVENCIJALNE DATOTEKE

DIREKTNA OBRADA – Ima smisla ako je sekvencijalna datoteka mala tako da se može smestiti u operativnu memoriju. Performanse obrade se malo razlikuju od performansi obrade serijske datoteke

VODEĆA DATOTEKA U DIREKTNOJ I REDOSLEDNOJ OBRADI – Češća upotreba, sukcesivno učitavanje fizički susednih slogova, počevši od prvog pa do poslednjeg. Ukupan broj pristupa kada se sekvencijalna datoteka koristi kao vodeća u obradi je ***B.***

REDOSLEDNA OBRADA – Iterativan proces, vodeća datoteka generiše logički naredne vrednosti ključa za traženje u obrađivanoj, sekvencijalnoj datoteci. Svaki korak obrade je zapravo traženje logički narednog sloga I vrši se metodom linearnog traženja. Svaki blok datoteke učitava se u OM samo jedanput.

AŽURIRANJE SEKVENCIJALNE DATOTEKE

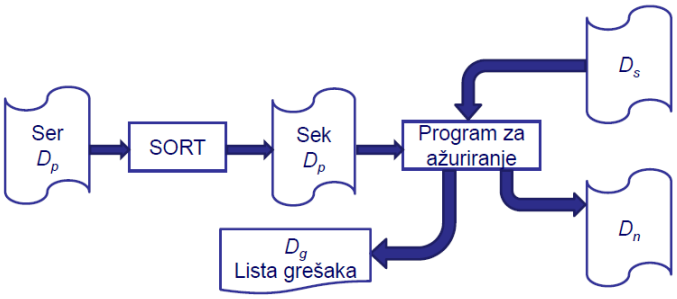
- **UPIS NOVOG SLOGA** – prethodi mu neuspešno traženje. Pronalazi se mesto upisa novog sloga tj. lokacija sloga sa prvom većom vrednošću ključa od datog. Vrši se pomeranje svih slogova sa vrednostima ključa većim od ključa novog sloga za jednu lokaciju udesno
- **BRISANJE POSTOJEĆEG SLOGA** – prethodi mu uspešno traženje. Vrši se pomeranje svih slogova sa većom vrednošću ključa ulevo ako se brisanje vrši fizički.
- **MODIFIKACIJA SADRŽAJA SLOGA** – prethodi joj uspešno traženje.

Upis I brisanje su ozbiljan problem zbog ukupnog broja pristupa.

U režimu direktne obrade - u proseku, pomeranje polovine od ukupnog broja slogova za jednu lokaciju pri upisu ili brisanju sloga. Primenjuje se kada je kompletna datoteka smeštena u operativnu memoriju.

U režimu redosledne obrade – iterativan postupak, kreiranje potpuno nove datoteke na osnovu postojeće. Primeren kada se datoteka može kompletno smestiti u operativnu memoriju. Datoteke I uloge u obradi date su na slici ispod.

- **Ds** – obrađivana, ulazna sekvencijalna datoteka
- **Dn** – obrađena, izlazna sekvencijalna datoteka
- **Dp** – vodeća datoteka promena, serijska, ulazna
- **Dg** – datoteka grešaka



Format sloga datoteke Ds I Dn je isti – predstavlja par ključa I podatka. Format sloga datoteke promena Dp predstavlja trojku u kojoj su ključ sloga, promenjeni podatak I polje statusa izvršene operacije. Datoteka grešaka sadrži slogove koji su formata ključ, podatak, polje opisa greške koji ukazuju na pokušaj upisa već postojećeg, ili brisanje ili modifikaciju nepostojećeg sloga u datoteci.

U režimu redosledne obrade se vrši sekvencijalni pristup sa učitavanjem slogova iz Ds I Dp, upoređivanje vrednosti ključeva tekućih slogova I generisanje I upis novih slogova u datoteku Dn na osnovu sadržaja tekućih slogova.

Dužina intervala između dva ažuriranja se određuje tako da se tokom njega nakupi dovoljan broj promena koji bi opravdao ovaj postupak. Duži interval znači veću efikasnost obrade, ali I duže vreme neusaglašenosti sadržaja datoteke sa realnim stanem.

OBLASTI PRIMENE I KARAKTERISTIKE

PREDNOSTI

1. Najpogodnija fizička organizacija za redoslednu obradu (*zbog činjenice da su logički susedni slogovi smešteni u fizički susedne lokacije, učitavanjem jednog bloka u OM pribavlja se f slogova koji najverovatnije učestvuju u narednim koracima obrade*)
2. Ekonomično korišćenje memorijskog prostora
3. Mogućnost korišćenja I magnetne trake i magnetnog diska kao medijuma

NEDOSTACI

1. Nepogodna za direktnu obradu
2. Potreba sortiranja pri formiranju
3. Relativno dug postupak ažuriranja

6 – Indeks-sekvencijalna datoteka

KARAKTERISTIKE INDEKSNIH DATOTEKA

Postojanje indeksa, pomoćne strukture podataka realizovane u posebnoj datoteci kao stablo traženja. Sadrži parove (vrednost ključa, relativna adresa sloga/bloka) koji služe za preslikavanje argumenta traženja u adresu sloga I za brz pristup slučajno odabranom I logički narednom slogu.

Smeštanje kompletnh slogova vrši se u posebnu datoteku – zonu podataka (primarnu zonu)

VRSTE INDEKSNIH DATOTEKA

STATIČKE – statička alokacija memorijskog prostora, definiše se prilikom projektovanja organizacije, statički indeks se formira I ne ažurira se

DINAMIČKE – nezaobilazne u realnim projektima, dinamička alokacija memorijskog prostora, dinamički indeks koji se ažurira paralelno sa ažuriranjem zone podataka

EFIKASNOST U UPOTREBI

Sekvencijalne datoteke su ideal redosledne obrade I traženja logički narednih slogova dok su rasute ideal direktne obrade I traženja slučajno odabranih slogova. Indeksne datoteke predstavljaju strukturu kompromisa. Daju solidnu podršku direktne I redosledne obrade, kao I obe vrste traženja.

STATIČKA INDEKS SEKVENCIJALNA DATOTEKA

Sadrži tri memorijske zone tj. osnovne organizacije datoteke:

- **Primarnu zonu ili zonu podataka** – sekvencijalna organizacija
- **Zonu indeksa** – spregnuta organizacija u vidu n-arnog stabla
- **Zonu prekoračenja** – spregnuta organizacija u vidu lanaca prekoračilaca

PRIMARNA ZONA

Slogovi su uređeni saglasno rastućim vrednostima ključa I grupisani su u blokove. Primarna zona se kreira u postupku formiranja statičke indeks-sekvencijalne datoteke I nikada se naknadno ne ažurira.

Ciljevi:

- Iskoristiti poželjne osobine sekvencijalne organizacije u redoslednoj obradi podataka
- Izbeći efekat loših performansi ažuriranja sekvencijalno organizovane datoteke.

ZONA INDEKSA

Puno stablo traženja, spregnuta struktura reda n I visine h. Čvor stabla je blok koji sadrži 1 do n elemenata. Svaki element sadrži vrednost ključa sloga S I adresu bloka primarne zone u kom je slog S u slučaju lista, ili drugog čvora stabla traženja koji takođe sadrži ključ, u slučaju neterminalnog čvora. Elementi u čvoru su uređeni saglasno rastućim vrednostima ključa.

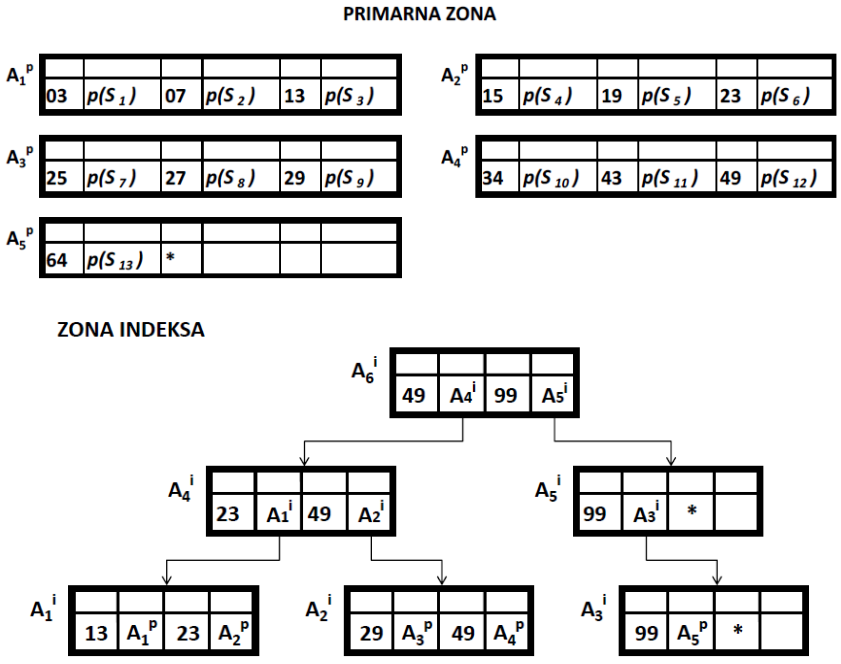
RETKO POPUNJEN INDEKS – jer se reprezentativne vrednosti ključa svakog bloka primarne zone propagiraju u stablo.

Vrednosti ključa u stablu mogu biti najveće ili najmanje vrednosti ključa Iz svakog bloka primarne zone. Elementi listova stabla sadrže po jednu vrednost ključa Iz svakog bloka, a elementi čvorova na višim nivoima hijerarhije stabla sadrže po jednu vrednost ključa iz svakog direktno podređenog čvora. Vrednosti ključa ponavljaju se u čvorovima na svim nižim nivoima hijerarhije. Svaki neterminalni čvor sa m elemenata poseduje m direktno podređenih čvorova.

Stablo traženja obezbeđuje relativno brz pristup za traženje slučajno odabranog sloga.

VRSTE ZONE INDEKSA

- **ZONA INDEKSA S PROPAGACIJOM NAJVEĆIH VREDNOSTI KLJUČA IZ SVAKOG BLOKA** – U slučaju poslednjeg bloka propagira se najveća dozvoljena vrednost ključa umesto aktuelne najveće vrednosti ključa.
- **ZONA INDEKSA S PROPAGACIJOM NAJMANJIH VREDNOSTI KLJUČA IZ SVAKOG BLOKA** – U slučaju prvog bloka propagira se najmanja dozvoljena vrednost ključa umesto aktuelne najmanje vrednosti ključa.



ZONA PREKORAČENJA

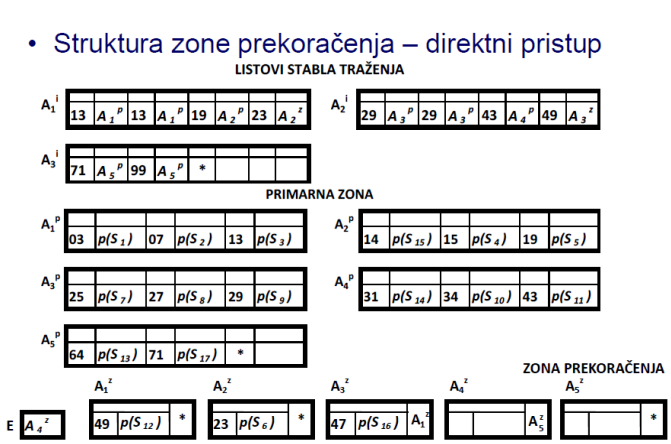
Sadrži kompletne slogove datoteke kao I primarna zona, koji se upisuju u zonu prekoračenja pri upisu novih slogova. Ti slogovi nazivaju se prekoračioci I svaki blok primarne zone može imati svoje prekoračioce.

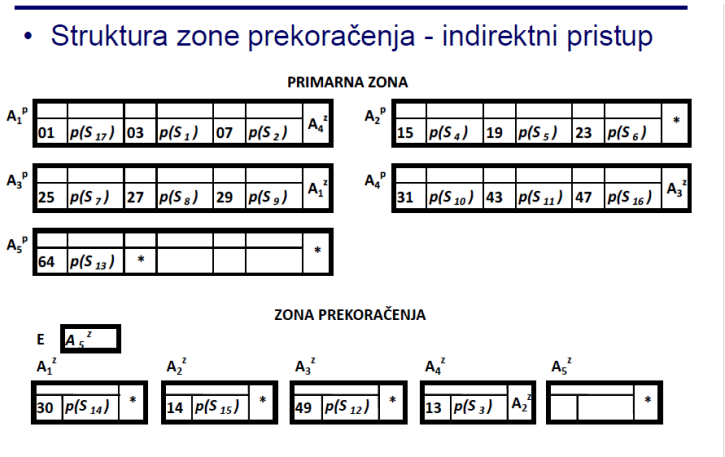
- Kada blok u primarnoj zoni nije kompletan, upis novog sloga dovodi samo do pomeranja slogova u bloku.
- Kada je blok u primarnoj zoni kompletan, upis svakog novog sloga dovodi do upisa jednog od slogova koji pripadaju bloku, u zonu prekoračenja.
    - Ako je ključ sloga koji se upisuje manji od maksimalne vrednosti ključa u bloku, novi slog se upisuje u blok a svi slogovi sa vrednošću ključa većom pomeraju se za jednu lokaciju ka kraju bloka. Slog sa najvećom vrednošću ključa u bloku se upisuje u zonu prekoračenja.
    - Ako je ključ sloga veći od najvećeg ključa u bloku, slog se direktno upisuje u zonu prekoračenja.

Sprežu se logički neposredno susedni prekoračioci iz jednog bloka. Faktor blokiranja je 1. Za svaki blok primarne zone postoji najviše jedan lanac spregntih prekoračilaca. Slogovi u svakom lancu prekoračilaca uređeni su u rastućem ili opadajućem redosledu.

Postoje 2 načina za čuvanje pokazivača na početak lanca:

- **DIREKTNO povezivanje sa listom stabla traženja** – pokazivač na početak lanca smešta se u odgovarajući list
- **INDIREKTNO povezivanje sa listom stabla traženja** – pokazivač na početak lanca smešta se u prateći deo odgovarajućeg bloka u primarnoj zoni





INDEKS SEKVENCIJALNA METODA PRISTUPA

Obezbeđuje formiranje, traženje, ažuriranje I reorganizaciju kao I sekvencijalni, direktni I dinamički način pristupa indeks-sekvencijalnoj datoteci. Podržana je najčešće u sistemima za upravljanje datoteka ugrađenim u OS, ili SUBP.

FORMIRANJE IS DATOTEKE

Program redosledno učitava slogove ulazne sekvencijalne datoteke, smešta blokove u primarnu zonu ili alternativno proglašava već formiranu sekvencijalnu datoteku za primarnu zonu, a zatim vrši formiranje indeksa.

FORMIRANJE ZONE INDEKSA

Iterativan postupak koji se vrši po nivoima stabla traženja – odozdo na gore, sleva na desno. Prvo se formiraju svi listovi tj. čvorovi nivoa h, zatim nivoa h-1 I tako do 1. U svaki čvor na i-tom nivou hijerarhije upisuju se najveće ili najmanje vrednosti ključa iz n sukcesivnih čvorova na nivou hijerarhije i + 1.

Kod propagacije najmanjih vrednosti, u poslednji element krajnjeg desnog čvora upisuje se maksimalna dozvoljena vrednost ključa a kod propagacije najmanjih vrednosti, u prvi element krajnjeg levog čvora upisuje se minimalna dozvoljena vrednost ključa.

FORMIRANJE ZONE PREKORAČENJA

Alocira se prazna zona prekoračenja, svi blokovi se sprežu u lanac slobodnih blokova a početak lanca se upisuje u zaglavlje zone prekoračenja.

TRAŽENJE LOGIČKI NAREDNOG SLOGA

Vrši se kombinovanom primenom metode linearnog traženja I metode traženja praćenjem pokazivača. Počinje u prvom bloku primarne zone, a svako sledeće traženje se nastavlja od tekućeg sloga datoteke u bloku primarne zone – linearna metoda. Po dolasku do poslednjeg sloga bloka, traženje se nastavlja u lancu prekoračilaca ako postoji – metoda praćenja pokazivača.

Kod indirektnog povezivanja prekoračilaca – nastavak traženja direktno u zoni prekoračenja. Traženje logički narednog sloga je **efikasnije** zbog manjeg broja pristupa.

Kod direktnog povezivanja prekoračilaca – pristup direktno nadređenom listu I nastavak traženja u zoni prekoračenja. Pristupa se blokovima primarne zone, prekoračiocima I listovima stabla traženja pa I nije toliko efikasno kao kod indirektnog povezivanja.

TRAŽENJE SLUČAJNO ODABRANOG SLOGA

Vrši se praćenjem pokazivača u stablu pristupa. Započinje u korenu I stiže do lista u stablu traženja. Uvažava organizaciju sa propagacijom maksimalnih ili minimalnih vrednosti ključa iz svakog bloka.

Kod indirektnog povezivanja prekoračilaca – prati se pokazivač odgovarajućeg elementa u listu I prelazi se u blok podataka u primarnoj zoni. Po potrebi se nastavlja traženje praćenjem lanca prekoračilaca.

Kod direktnog povezivanja prekoračilaca – dolaskom do odgovarajućeg elementa u listu donosi se odluka o nastavku traženja u bloku primarne zone ili praćenjem lanca prekoračilaca u zoni prekoračenja. **Efikasnije** je traženje kod direktnog povezivanja nego kod indirektnog.

OBRADA IS DATOTEKE

Moguća je efikasna obrada I u režimu redosledne obrade I u režimu direktne obrade. Pogodne su za korišćenje u ulozi vodeće datoteke u oba režima.

Redosledna obrada putem vodeće datoteke odvija se naizmeničnim pristupanjem blokova primarne zone I njihovim lancima prekoračilaca. Adresa prvog bloka primarne zone nalazi se u zaglavlju datoteke. Redosledna obrada je malo **efikasnija** kod datoteke sa **indirektnim** povezivanjem prekoračilaca.

Direktna obrada je malo **efikasnija** kod datoteke sa **direktnim** povezivanjem prekoračilaca.

AŽURIRANJE IS DATOTEKE

Vrši se u režimu direktne obrade.

UPIS NOVOG SLOGA

- Ako se neuspešno traženje zaustavilo u bloku primarne zone:
    1. Vrši se pomeranje slogova sa većom vrednošću ključa od vrednosti ključa novog sloga za jednu lokaciju ka kraju bloka
    2. Novi slog se upisuje u lokaciju koju je zauzimao slog sa prvom većom vrednošću ključa, a slog sa do tada najvećom vrednošću ključa u bloku upisuje se u zonu prekoračenja
    3. Prekoračilac se upisuje u lokaciju čiju adresu sadrži indeks slobodnih lokacija I povezuje se sa ostalim prekoračiocima iz bloka
- Ako se neuspešno traženje zaustavilo na nekom od prekoračilaca, novi slog se upisuje u prvu slobodnu lokaciju I uvezuje se sa ostalim prekoračiocima.

BRISANJE SLOGA

Najčešće je logičko – lokacija logički izbrisanog sloga može se upotrebiti za upis novog sloga u slučaju kada se vrednost ključa novog sloga nalazi tačno u odgovarajućim granicama. Fizičko brisanje zahteva pomeranje slogova sa ažuriranjem lanca prekoračilaca I zahteva mnogo veći broj pristupa datoteci.

MODIFIKACIJA SADRŽAJA SLOGA

Nakon uspešnog traženja, potreban je samo jedan pristup kako bi se modifikovani slog upisao u datoteku.

REORGANIZACIJA IS DATOTEKE

Usled upisa slogova u zonu prekoračenja I logičkog brisanja slogova, dolazi do značajne degradacije performansi traženja slogova I obrade datoteke u realnom vremenu. Iz tog razloga, potrebno je vršiti periodičnu reorganizaciju datoteke tj. uklanjanje negativnih posledica ažuriranja.

Postupak se sastoji iz ponovnog formiranja primarne zone putem redosledne obrade – traženjima logički narednih slogova u postojećoj primarnoj zoni I zoni prekoračenja, zatim generisanje novog stabla traženja I formiranje nove, prazne zone prekoračenja.

Interval vremena između dve reorganizacije može biti fiksan ili dinamički određen.

DISTRIBUIRANI SLOBODNI PROSTOR – ublažava problem degradacije performansi obrade zbog upisa novih slogova. Blokovi podataka se pri formiranju datoteke popunjavaju samo delimično (60-80%) I time se obezbeđuje prostor za upis novih slogova I produžava se ujedno interval vremena između dve reorganizacije.

IS datoteka može se ažurirati I kao sekvencijalna u režimu redosledne obrade – po završetku ažuriranja, generiše se novo stablo traženja. Ovakav postupak je istovremeno I reorganizacija datoteke.

OBLASTI PRIMENE I KARAKTERISTIKE

PREDNOSTI:

1. Kada iste podatke treba obrađivati I u režimu redosledne I u režimu direktne obrade
2. Intenzivno se koristi u paketnoj obradi, a može I u interaktivnoj
3. Peformanse redosledne obrade ne zaostaju za performansama redosledne obrade sekvencijalne datoteke
4. Performanse direktne obrade ne zaostaju za performansama direktne obrade rasute datoteke

NEDOSTACI:

1. Upis slogova u zonu prekoračenja dovodi do degradacije performansi obrade. Jedino rešenje je periodična reorganizacija datoteke koja je nepogodna.

FSP zasnovane na statičkoj IS organizaciji se intenzivno koriste u mrežnim sistemima baza podataka. Imaju istorijski značaj ali su zamenjene indeksnim datotekama sa B-stablima.

Osnovna ideja za primenu IS datoteka je:

1. Kada se podaci ne ažuriraju često I u velikom obimu
2. Kada je potrebno obezbediti efikasnu redoslednu obradu I istovremeno dobre performanse direktne obrade
3. Brz pristup slučajno odabranom slogu u sekvencijalnoj strukturi vrši se korišćenjem stabla traženja kao funkcije koja preslikava vrednost ključa u adresu.

7 – Rasuta organizacija datoteke

RASUTA ORGANIZACIJA DATOTEKE

Naziva se I direktnom jer se slogu ili grupi slogova pristupa direktno na osnovu poznavanja adrese memorijske lokacije u kojoj su smešteni. Adresa lokacije dobija se transformacijom vrednosti identifikatora sloga u adresu. Transformacija vrednosti je hash funkcija koja preslikava domen identifikatora u skup adresa lokacija memorijskog prostora datoteke.

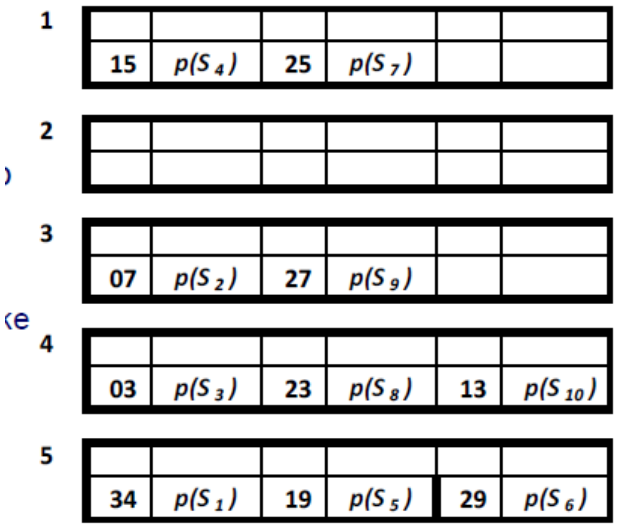
Fizička struktura podataka ne sadrži informaciju o vezama između slogova logičke strukture datoteke – u dve fizički susedne lokacije ne moraju biti memorisani logički susedni slogovi već su oni na slučajan način rasuti po memorijskom prostoru datoteke.

IDENTIFIKATOR – skup obeležja čije vrednosti jednoznačno određuju slogove datoteke. Identifikator može a ne mora da pripada skupu obeležja tipa sloga datoteke. Interni identifikator po pravilu je primarni ključ a eksterni identifikator kada se vrednost identifikatora pridružuje van konteksta sadržaja datoteke.

VRSTE TRANSFORMACIJA VREDNOSTI IDENTIFIKATORA U ADRESU

- **DETERMINISTIČKA** – funkcija h je injektivna, svakoj vrednosti identifikatora odgovara jedna adresa a svakoj adresi najviše jedna vrednost identifikatora
- **PROBABILISTIČKA** – svakoj vrednosti identifikatora odgovara jedna adresa a jednoj adresi može odgovarati više rezultata transformacije

BAKET – tradicionalan naziv za blok kod rasutih datoteka, transformacijom h vrednost identifikatora pretvara se u adresu baketa



VRSTE RASUTIH DATOTEKA

STATIČKE – veličina adresnog prostora određuje se I kompletno alocira unapred, I ne može se menjati tokom eksploatacije

DINAMIČKE – veličina dodeljenog prostora menja se tokom ažuriranja saglasno potrebama.

OPŠTI POSTUPAK FORMIRANJA STATIČKE RASUTE DATOTEKE

U postupku formiranja dodeljuje se Q lokacija I nakon formiranja se više ne može menjati. 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.

DIREKTNA I RELATIVNA ORGANIZACIJA DATOTEKE

DIREKTNA ORGANIZACIJA

Eksterni identifikator je trivijalna deterministička transformacija. Vrednost identifikatora je u isto vreme I adresa baketa. Preslikavanje veza između sadržaja slogova I adresa ne pripada direktnoj organizaciji datoteke. Postoje 2 vrste direktnih datoteka s obzirom na vrstu upotrebljavanih adresa:

- Vrednosti identifikatora su mašinske adrese baketa

  DIREKTNA DATOTEKA SA MAŠINSKIM ADRESAMA

  Koristi se adresa baketa na disku oblika (u, c, t, s). Nedostaci su čvrsta povezanost programa sa fizičkim karakteristikama memorijskog uređaja I odsustvo logičke veze između vrednosti identifikatora I sadržaja sloga.

- Vrednosti identifikatora su relativne adrese baketa

  DIREKTNA DATOTEKA SA RELATIVNIM ADRESAMA

  Koriste se relativne adrese slogova. Lokacije memorijskog prostora numerišu se rednim brojevima, redni broj sloga u memorijskom prostoru predstavlja eksterni identifikator sloga, a rešava se problem čvrste povezanosti slogova datoteke sa fizičkim karakteristikama memorijskog uređaja. Glavni nedostatak je odsustvo logičke veze između vrednosti identifikatora I sadržaja sloga.

RELATIVNA ORGANIZACIJA

RELATIVNA METODA PRISTUPA

Deo sistema za upravljanje podacima koji obavlja transformaciju relativne adrese u mašinsku I ostale operacije. Pruža programu usluge na nivou baketa. Ne vrši blokiranje I rastavljanje blokova na slogove već prihvata samo vrednost faktora blokiranja jednaku 1. Aktivnosti blokiranja I rastavljanja blokova na slogove moraju se realizovati u okviru samog programa.

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 I smeštanje sloga u lokaciju s pridruženom relativnom adresom.

TRAŽENJE SLOGA U RELATIVNOJ DATOTECI

Traženje I logički narednog kao I slučajno odabranog sloga se vrši zadavanjem relativne adrese lokacije metodi pristupa. Na osnovu adrese, metoda pristupa prenosi blok u radnu zonu programa. Uspešno traženje – samo ako traženi slog postoji u prenetom bloku.

AŽURIRANJE RELATIVNE DATOTEKE

Vrši se u režimu DIREKTNE obrade.

UPIS NOVOG SLOGA – novom slogu se pridružuje vrednost idenifikatora, slog se upisuje u datoteku, ako postoji slobodna lokacija u bloku, pre upisa može se izvršiti provera da li slog sa istom vrednošću ključa postoji u istom bloku

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

PERFORMANSE RELATIVNE DATOTEKE

Traženje slučajno odabranog sloga – najefikasnije moguće

Traženje logički narednog sloga – efikasnije od serijske ali manje efikasno od sekvencijalne

U oba slučaja potreban je samo jedan pristup datoteci. Uvođenje relativne adrese lokacije kao identifikatora rešava problem čvrste povezanosti slogova datoteke sa karakteristikama diska.

S druge strane, nepostojanje veze između vrednosti identifikatora I sadržaja sloga u relativnoj organizaciji datoteke predstavlja nedostatak.

Moguća primena relativne metode pristupa:

- Kao osnova za izgradnju spregnute datoteke
- Kao osnova za izgradnju rasutih datoteka sa probabilističkom transformacijom
- Kao osnova za izgradnju indeksnih datoteka

STATIČKA RASUTA ORGANIZACIJA DATOTEKE

OPŠTE KARAKTERISTIKE RASUTE DATOTEKE SA PROBABILISTIČKOM TRANSFORMACIJOM

Ključ često uzima vrednosti iz veoma velikog opsega mogućih vrednosti. Veličina adresnog prostora dodeljenog datoteci određuje se kao proizvod broja baketa I broja lokacija u baketu. Broj aktuelnih slogova u datoteci je po pravilu mnogo manji od broja mogućih vrednosti ključa.

METODE PROBABILISTIČKE TRANSFORMACIJE

Uvode se kako bi se prevazišli nedostaci do kojih dovodi deterministička transformacija vrednosti ključa u adresu.

CILJEVI:

1. Što ravnomernija raspodela slogova u adresnom prostoru
2. Pseudoslučajna transformacija vrednosti ključa u adresu
3. Pravilno dizajniranje potrebnog adresnog prostora

KORACI:

1. Pretvaranje nenumeričke u numeričku vrednost ključa
2. Pretvaranje numeričke vrednosti ključa u pseudoslučajan broj
3. Dovođenje vrednosti dobijenog pseudoslučajnog broja u opseg dozvoljenih vrednosti relativne adrese
4. Pretvaranje relativne u mašinsku adresu

METODA OSTATKA PRI DELJENJU – relativna adresa A je celobrojni ostatak pri deljenju numeričke vrednosti ključa po modulu m ($T = k(S) (mod\ m)$, $A = 1 + T$).

Kako bi rezultat transformacije bio što slučajniji, a transformacija što ravnomernija, m ne treba da bude paran broj niti stepen osnove brojnog sistema već prost broj ili neparan broj sa velikim prostim činiocima.

Ova metoda je pogodna za primenu kada se vrednosti ključa javljaju u paketima – pojedini intervali u opsegu dozvoljenih vrednosti ključa gusto su popunjeni aktuelnim vrednostima ključa. Između njih, nalaze se intervali sa neaktuelnim vrednostima ključa. Slogovi sa sukcesivnim vrednostima ključa iz paketa dobijaju adrese fizički susednih baketa, što rezultuje ravnomernim popunjavanjem baketa.

METODA CENTRALNIH CIFARA KVADRATA KLJUČA – vrednost ključa diže se na kvadrat I uzima se onoliko centralnih cifara kvadrata vrednosti ključa koliko pozicija treba da sadrži relativna adresa. Zatim se vrši centriranje I normiranje pseudoslučajnog broja na zadati opseg relativnih adresa.

METODA PREKLAPANJA – cifre ključa premeštaju se kao pri preklapanju hartije, vrši se sabiranje preklopljenih vrednosti po modulu Vn. Pogodna za primenu kada je broj pozicija vrednosti ključa mnogo

veći od broja pozicija relativne adrese. Preklapanje se izvodi po osama koje zdesna ulevo određuje broj pozicija n relativne adrese.

KARAKTERISTIKE PROBABILISTIČKE TRANSFORMACIJE

SLOGOVI SINONIMI – dve različite vrednosti ključa mogu transformacijom dobiti istu relativnu adresu. Metoda transformacije particionira skup mogućih vrednosti ključa na B skupova sinonima. Verovatnoća pojave sinonima zavisi od raspodele vrednosti ključa unutar opsega dozvoljenih vrednosti, odabrane metode transformacije I faktora popunjenosti memorijskog prostora.

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

PREKORAČILAC – slog koji ne može biti smešten u matični baket usled njegove popunjenosti mora se smestiti u neki drugi baket. Za smeštanje prekoračilaca definiše se poseban postupak. Njihova pojava je nepoželjna jer zahteva poseban postupak za pronalaženje nove slobodne lokacije za smeštanje prekoračilaca I dovodi do produženja vremena pristupa pri kasnijem traženju slogova prekoračilaca. Pojava prekoračilaca je neminovna ali broj prekoračilaca će biti manji što su slogovi ravnomernije raspoređeni po baketima, što je faktor popunjenosti manji I faktor baketiranja veći.

Izbor faktora popunjenosti ostvaruje veliki uticaj na karakteristike rasuto organizovane datoteke. Za malo q verovatnoća pojave više slogova u jednom skupu sinonima je mala ali je malo I iskorišćenje memorijskog prostora. Za veliko q iskorišćenje memorijskog prostora je dobro, ali je velika verovatnoća pojave sinonima I prekoračilaca. U praksi se bira da je **q <= 0.8.**

Izbor faktora baketiranja utiče na očekivani broj prekoračilaca po jednom baketu, pri datom faktoru popunjenosti q. S porastom faktora baketiranja, verovatnoća pojave prekoračilaca opada, ali raste vreme razmene sadržaja baketa između diska I operativne memorije. Sa smanjenjem faktora baketiranja, povećava se očekivani broj prekoračilaca, ali I broj baketa čime se poboljšava preciznost transformacije.

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.
- KOMBINACIJA PRETHODNA DVA – lokalne zone prekoračenja u primarnoj zoni, glavna zona prekoračenja posebna zona

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 mogućih vrednosti, veličinu sloga, obim I karakter ažuriranja datoteke Itd.

FORMIRANJE RASUTE DATOTEKE

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 vrši transformaciju vrednosti ključa, smeštanje prekoračilaca, formiranje baketa od više slogova I izdvajanje slogova iz baketa, dok relativna metoda vrši direktni upis I čitanje sadržaja baketa.

1. **INICIJALNO ALOCIRANJE PRAZNE DATOTEKE –** statička alokacija kompletnog praznog prostora datoteke. Izbor postupka prepoznavanja slobodnih lokacija unutar baketa – putem statusnog polja, upisom specijalnih znakova u lokaciju ili putem indeksa slobodnih lokacija
2. **UPISIVANJE SLOGOVA U RASUTU DATOTEKU –** na osnovu sadržaja vodeće datoteke kojoj se sekvencijalno pristupa ili direktnim upisivanjem slogova u realnom vremenu.

A. **FORMIRANJE U JEDNOM PROLAZU** – Slogovi se upisuju u hronološkom redosledu nastanka, bilo čitanjem sadržaja vodeće datoteke u sekvencijalnom pristupu ili direktnim unosom podataka u realnom vremenu

B. **FORMIRANJE U DVA PROLAZA** – slogovi se učitavaju iz vodeće datoteke I upisuju u rasutu. U prvom prolazu upisuju se samo oni koji će biti smešteni u matične bakete, a u drugom prolazu se upisuju prekoračioci, saglasno izabranom postupku za njihovo smeštanje.

TRAŽENJE SLOGA U RASUTOJ DATOTECI

Traženje logički narednog je jednako traženju slučajno odabranog sloga I vrši se metodom transformacije argumenta u adresu baketa. Ako se slog ne nađe u matičnom baketu, a matični baket ima prekoračilaca, traženje se nastavlja linearnom metodom, ponovnom transformacijom ili praćenjem pokazivača.

VRSTE STATIČKIH RASUTIH ORGANIZACIJA

A. RASUTE DATOTEKE SA JEDINSTVENIM ADRESNIM PROSTOROM

1. Rasuta sa linearnim traženjem lokacije za smeštaj prekoračilaca - otvoreni način adresiranja
    - *Sa fiksnim korakom (k = 1, k > 1)*
    - *Sa slučajno odabranim korakom*
2. Rasuta sa sprezanjem prekoračilaca u jedinstvenom adresnom prostoru – primarnoj zoni

B. RASUTE DATOTEKE SA ZONOM PREKORAČENJA

1. **Sa serijskom zonom prekoračenja**
2. **Sa spregnutom zonom prekoračenja**

RASUTA SA LINEARNIM TRAŽENJEM PREKORAČILACA SA FIKSNIM KORAKOM

- Ukoliko je matični baket popunjen, slog se smešta u prvu narednu slobodnu lokaciju (k = 1) ili u lokaciju udaljenu za k pozicija (k > 1).

- **Traženje** slučajno odabranog sloga vrši se linearnom metodom, zaustavlja se na pronađenom slogu ako je uspešno, na prvoj slobodnoj lokaciji ako je neuspešno ili ponovnim nailaskom na matični baket, ako je neuspesno a cela datoteka je kompletno popunjena.

- **Upis** novog sloga prekoračioca se vrši pronalaženjem prve slobodne lokacije iza matičnog baketa

- **Brisanje** postojećeg sloga može biti logičko, putem tri statusa sloga (aktuelan, neaktuelan, slobodna) ili fizičko uz lokalnu reorganizaciju memorijskog prostora (kada ne postoje prekoračioci oslobađa se lokacija, a kada postoje, svi prekoračioci pomeraju se za jednu poziciju prema matičnom baketu, pri čemu prekoračilac ne sme da pređe ispred svog matičnog baketa).

Nedostaci kod k = 1:

1. **Efekat nagomilavanja prekoračilaca** – prekoračioci iz jednih baketa izazivaju pojavu prekoračilaca iz drugih baketa, sve veća je verovatnoća zauzeća prve prazne lokacije iza sve dužeg lanca zauzetih lokacija. Ovaj efekat se može odložiti korišćenjem promenljiv. koraka k > 1.
2. **Neefikasno traženje** – jer se vrši I u baketima koji ne sadrže slogove iz istog skupa sinonima
3. **Neefikasno neuspešno traženje** – jer se zaustavlja tek nailaskom na prvu slobodnu lokaciju.

Glavna motivacija za k > 1 – izbegavanje efekta nagomilavanja prekoračilaca I prekidanje dugačkih lanaca zauzetih lokacija (bolji pokusaj da se očuva približno jednaka verovatnoća zauzeća bilo koje prazne lokacije)

RASUTA SA LINEARNIM TRAŽENJEM PREKORAČILACA SA SLUČAJNIM KORAKOM

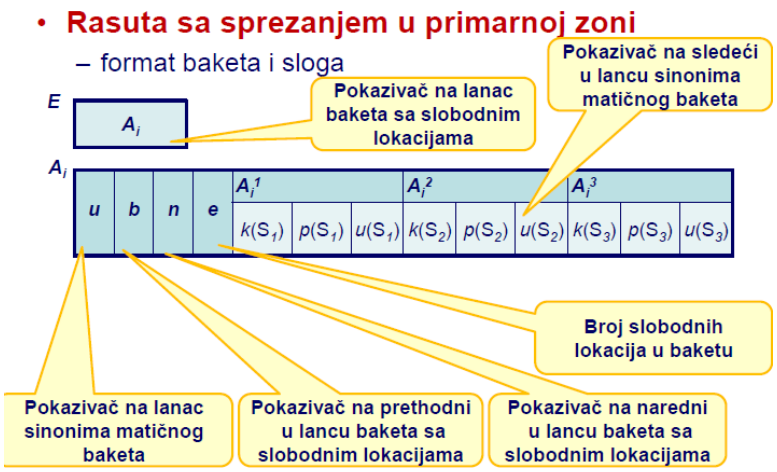
Ukoliko je matični baket popunjen, slog se smešta u narednu slobodnu lokaciju udaljenu za k pozicija, s obzirom na poziciju matičnog baketa. Korak k se određuje na slučajan način – predstavlja rezultat druge probabilističke transformacije primenjene na vrednost identifikatora – ključa sloga.

Glavna motivacija – izbegavanje efekta nagomilavanja prekoračilaca I prekidanje dugačkih lanaca zauzetih lokacija (bolji pokusaj da se očuva približno jednaka verovatnoća zauzeća bilo koje prazne lokacije)

Pogodne su za upotrebu u slučaju manje popunjenosti I 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, I specijalan nulti baket sa pokazivačem na početak lanca.



Vrši se sprezanje svih sinonima u odnosu na matični baket. Za svaki matični baket se kreira po jedan lanac sinonima, pokazivač na početak lanca u zaglavlju matičnog baketa I pokazivač na sledeći u lancu sinonima ugrađen u svaki slog.

Traženje slučajno odabranog sloga – transformacija ključa u adresu I pristupanje matičnom baketu, praćenje lanca sinonima započinjući od pokazivača u. Zaustavlja se na pronađenom slogu ako je uspešno, ili na kraju lanca sinonima ako je neuspešno.

Upis novog sloga – nakon neuspešnog traženja, vrši se uvezivanjem u lanac sinonima. Izborom prve prazne lokacije iz lanca baketa sa slobodnim lokacijama

Brisanje – FIZIČKO, uklanjanjem sloga iz lanca sinonima uz potrebno prevezivanje. Oslobađa lokacije uz eventualno vraćanje baketa u lanac baketa sa slobodnim lokacijama.

Glavna motivacija – izbegavanje efekta neefikasnog traženja, traženje se vrši samo u baketima koji sadrže slogove iz istog skupa sinonima, a neuspešno traženje se zaustavlja dolaskom do kraja lanca spregnutih slogova – sinonima.

Poboljšana je efikasnost traženja, naročito neuspešnog, u odnosu na datoteke s otvorenim načinom adresiranja, ali je s druge strane, fizička struktura je komplikovanja a I dalje je moguć efekat nagomilavanja prekoračilaca.

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 I 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 I pokazivač na sledeći u lancu prekoračilaca ugrađen u svaki slog u zoni prekoračenja. Tipičan faktor blokiranja je 1 jer je mala verovatnoća da se dva prekoračioca iz istog lanca sinonima nađu u susednim lokacijama.

Formiranje – vrši se uvek u jednom prolazu. Svi prekoračioci su u zoni prekoračenja koja je odvojena od primarne zone.

Traženje slučajno odabranog sloga – transformacija vrednosti ključa u adresu 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. Zaustavlja se na pronađenom slogu ako je uspešno, ili na kraju lanca prekoračilaca ako je neuspešno.

Upis novog sloga – nakon neuspešnog traženja, upisuje se u matični baket ako ima mesta, ili se uvezuje 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 sloga – FIZIČKO, uklanjanjem sloga iz matičnog baketa uz prebacivanje prvog prekoračioca u matični baket ili uklanjanjem sloga iz lanca prekoračilaca uz potrebno prevezivanje.

Glavna motivacija – izbegavanje efekta neefikasnog traženja, traženje se vrši samo u baketima koji sadrže slogove iz istog skupa sinonima, a neuspešno traženje se zaustavlja dolaskom do kraja lanca spregnutih slogova – sinonima. Takođe, uklanjanje efekta nagomilavanja jer su svi prekoračioci u zoni prekoračenja koja je spregnuta datoteka.

Poboljšana je efikasnost traženja, naročito neuspešnog, u odnosu na datoteke s jedinstvenim adresnim prostorom.

RASUTA SA SERIJSKOM ZONOM PREKORAČENJA

Uvođenje zone prekoračenja koja je 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 je u slučaju manjeg očekivanog broja prekoračilaca jer se ne isplati sprezanje u zoni prekoračenja.

OBRADA RASUTE DATOTEKE SA PROBABILISTIČKOM TRANSFORMACIJOM

Nepogodne su za korišćenje u ulozi osnovne (prve) vodeće datoteke. Ne mogu se koristiti kao vodeće u režimu redosledne obrade pošto fizička struktura ne sadrži informaciju o logičkoj strukturi podataka. Mogu se koristiti kao obrađivane I vodeće u režimu direktne obrade. Mogu se obrađivati I u režimu redosledne I u režimu direktne obrade. Performanse su u oba slučaja iste zbog iste efikasnosti traženja I logički narednog I slučajno odabranog sloga.

OBLASTI PRIMENE RASUTIH DATOTEKA

U svim mrežnim SUBP, u nekim relacionim SUBP, u interaktivnoj obradi podataka kao I u režimu paketne obrade. Prednost je mali očekivani broj pristupa pri traženju slučajno odabranog sloga.

Nedostaci su:

- Potreba da se unapred odredi veličina datoteke
- Problem izbora probabilističke transformacije – ravnomerna raspodela broja sinonima po baketima pri formiranju I ažuriranju
- Broj pristupa pri traženju može biti nepredvidivo velik