Pretraživanje - AISP
Vrsta: Seminarski | Broj strana: 24

U ovom radu ćemo razmotriti problem pronalaženja određene informacije u velikom skupu podataka. Kao što ćemo videti, izvesne metode organizacije podataka (t.j. strukture podataka) čine proces pronalaženja efikasnijim. S obzirom da je proces pretraživanja vrlo čest u obradi podataka, poznavanje metoda i tehnika organizacije podataka pretraživanja je vrio važno.
Pre nego što predemo na konkretne metode uvedimo neke osnovne termine. Tabela ili datoteka je grupa elemenata od kojih se svaki naziva zapis ili čvor. Ove termine upotrebljavaćemo u najopštijem smislu i ne bi trebalo da se pomešaju sa sličnim terminima u PASCAL-u ili COBOL-u.
Proces pretraživanja je algoritam koji prihvata argument a i pronalazi jedan ili više zapisa u datoteci ili tabeli čija vrednost ključa je a. Može se desiti da proces pretraživanja bude neuspešan, t.j. da ne postoji zapis sa takvim ključem. U tom slučaju, često je poželjno u datoteku ubaciti zapis sa takvim ključem. Takvo pretraživanje se naziva pretraživanje sa ubacivanjem.
Primetite da do sada nismo ništa rekli o strukturi podataka u kojoj se datoteka implementira. To može biti niz, spregnuta lista, stablo ili čak mreža (graf). Pošto su različite metode pretraživanja pogodnije za neke strukture podataka, izbor implementacije datoteke (t.j. izbor strukture podataka) je često određen metodom pretraživanja koja se ima na umu.
Sekvencijalno pretraživanje
Najprostiji algoritam pretraživanja je sekvencijalno pretraživanje koji se primenjuje na nizove i spregnute liste. On se sastoji u tome da se ključ svakog zapisa ispituje u redosledu kako su zapisi smešteni sve dok se ne nađe traženi zapis. Neka se u nizu čuvaju zapisi deklarisani kao
public int Key; public object Value;
}
{
int Poz = -1;
for (int i = 0; i < aArray.length;
{
if ((aArray[i].Key == aTargetKey) {
Poz = i;
break;
}
}
return Poz;
}
Realizacija datoteke kao spregnute liste ima prednost u tome što se veličina memorijskog prostora potrebnog za smeštanje datoteke može menjati prema potrebi. Definišimo prvo čvor liste na sledeći način:
{
int Kljuc; CvorListe Sledeci;
public CvorListe(int aKljuc, CvorListe aSled)
{
Kljuc = aKljuc; Sledeci = aSled;
}
}
Pretpostavimo da promenljiva Glava pokazuje na prvi zapis datoteke, a polje Kljuc sadrži vrednost ključa zapisa koji tražimo. Algoritam koji sekvencijalno pretražuje datoteku i ako je pretraživanje neuspešno vrši ubacivanje, izgleda ovako: