Strukture podataka i algoritmi

Šesto predavanje

Skupovi

Općenito

- U mnogim algoritmima se kao matematički model javlja skup.
- U skup pohranjujemo kolekciju podataka istog tipa koje zovemo elementi
- U jednoj kolekciji ne mogu postojati dva elementa (podatka) s istom vrijednosti
- Unutar kolekcije se ne zadaje nikakav eksplicitni linearni ili hijerarhijski uređaj
- Među podacima nema povezanosti

… no ipak

- Pretpostavimo da među podacima možemo uspostaviti neku prirodnu relaciju ≤ (npr.)
- Dakle možemo govoriti o tome koji je element veći a koji manji (relacija uređaja)
- Skup definiramo kao apstraktni tip podataka s operacijama koje su uobičajene u matematici

Operacije

- MAKE_NUL(A) -operacija pretvara skup u prazan skup
- INSERT(x,A) -ubacuje element x u skup A
- DELETE(x,A) -izbacuje element x iz skup A
- MEMBER(x,A) -logička funkcija; istina ako je $x \in A$
- MIN(A),MAX(A) -vraća najmanji odnosno najveći element skupa
- SUBSET(A,B) -logička funkcija; istinita ako je $A \subseteq B$
- UNION(A,B,C) -procedura koja pretvara skup C u uniju skupova A i B; dakle C mijenja u $A \cup B$
- INTERSECTION(A,B,C) -procedura koja pretvara skup C u presjek skupova A i B; dakle C mijenja u $A \cap B$
- DIFFERENCE(A,B,C) procedura koja pretvara skup C u razliku skupova A i B; dakle C mijenja u $A \setminus B$

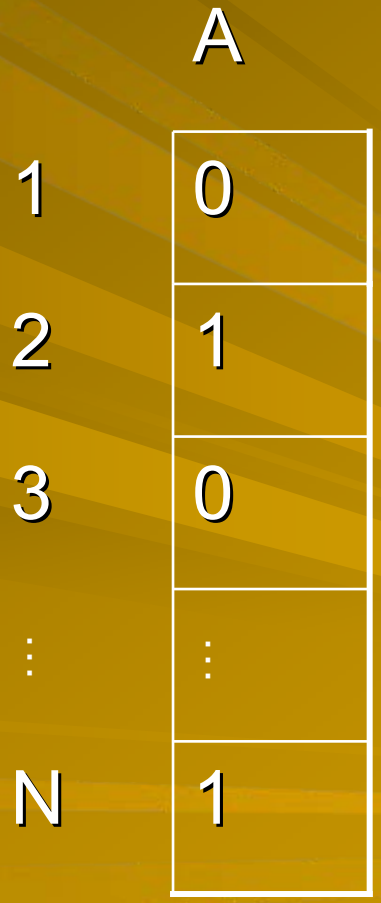
Problematika

- Ovako zadani apstraktni tip podataka (SET ili SKUP) veoma je blizak svakodnevnoj logici (problematici)
- Relativno ga je teško implementirati
- Postizanje efikasnog obavljanja neke operacije sporo će se obavljati neke druge

Implementacija skupa pomoću bit vektora

- Uzmimo da je tip elementa cijeli broj (element_type={1,2,…,N}) gdje je N dovoljno veliki cijeli broj
- Skup prikazujemo poljem bitova; *i*-ti bit je 1 (odnosno 0) ako i samo ako *i*-ti element pripada (odnosno ne pripada) skupu

Implementacija skupa pomoću bit vektora



karakteristike

- Operacije INSERT, DELETE i MEMBER zahtijevaju konstantno vrijeme budući da se svode na direktni pristup *i*-tom bitu.
- UNION, INTERSECTION, DIFFERENCE, SUBSET zahtijevaju vrijeme proporcionalno s *N*

Implementacija skupa pomoću sortirane vezane liste

- Prikazujemo skup kao listu
- Od dvije poznate implementacije bolja je ona pomoću pokazivača
- Da bi se operacije mogle bolje izvršavati pogodnije je da lista bude sortirana
- Potprogrami za UNION, DIFFERENCE, SUBSET i INSERTION uspoređuju dvije liste
- Zadaća: usporedi efikasnost algoritma za sortirane odnosno nesortirane liste A i B

Rječnik

- Često nije potrebno obavljati složene operacije $\cup$ $\cap$ $\subseteq$ \ nego se umjesto toga pamti jedan skup nad kojim se vrše operacije dodavanja, oduzimanja ili provjere prisutnosti elemenata

Primjeri

- Pravopis
- Spell checker