STRUKTURE PODATAKA I ALGORITMI

PROJEKAT I: RIJETKA MATRICA

Adna Čengić
Broj indexa: 5261/m
Prirodno-matematički fakultet Sarajevo

RIJETKE MATRICE

U numeričkoj analizi i drugim primjenama eng. sparse ili rijetke matrice su matrice kod kojih je većina elemenata nula. Ako su elementi većinom nenulti, onda se matrica smatra "gustom". Broj nultih elemenata podjeljen sa brojem elemenata se naziva *sparsity* matrice. Korištenjem ovih definicija, matrica je rijetka tj ima većinski nenulte elemente ako je sparsity veći od 0.5. Veće matrice tj matrice većih dimenzija se često pojavljuju u naučnim ili primjenama inžinjerske prirode.

Različite struktre podataka mogu biti korištene pri čuvanju memorije. Razlikujemo:

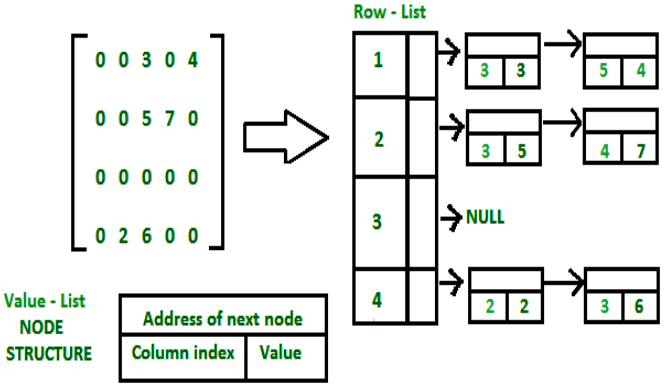
- One koje podržavaju efikasne modifikacije kao što su DOK,LIL ili COO. Ove metode se pretežno koriste pri konstrukciji matrica.
- One koje podržavaju efikasniji pristup i matrične operacije kao što su CSR ili CSC.

DOK ili Dictionary of keys se sastoji od riječnika koji preslikava (redove i kolone) parove u vrijednosti elemenata. Elementi koji nedostaju iz riječnika se postavljaju na 0.

LIL ili List of lists čuva jednu listu po redu pri čemu void računa o indexu kolone I vrijednosti. Tipično, ovi unosi se čuvaju sortirani po indexu kolone za bržu pretragu. Ovo je jos jedan format za inkrementalnu konstrukciju matrica. Lista bi trebala da je sortirana prema ulaznim ključevima.

Konkretno se baziramo na LIL pristup. Dakle, jednu listu koristimo za reprezentaciju redova a svaki red sadrži listu trojki:

1. Index colone
2. Nenultu vrijednost elementa
3. Polje adrese (također za nenulte elemente).



O PROJEKTU

Tema ovog projekta je bilo da se implementira klasa Matrica koja matricu čuva kao listu listi (ali ne koristiti tip list iz STL). Na primjer, ako matrica formata 1000 × 1000 sadrži samo 4 nenulta elementa i to a2,10 = 5, a2,400 = 15, a100,50 = 25, a700,800 = 35, onda klasa Matrica treba čuvati samo listu od 3 liste (od kojih prva ima dva elementa 5 i 15, a preostale dvije po jedan, 25 odnosno 35), vodeći evidenciju na koji se red koja lista odnosi, ali i unutar tih listi treba za svaki element voditi evidenciju u kojoj je koloni. Dakle treba da implementiramo klasu Matrica koja je u suštini sparse matrica. Pored ovoga potrebno je za tu klasu podržati sabiranje, oduzimanje, množenje i stepenovanje matrica, kao i transponovanje matrice. Potrebno je podržati konstruktor sa dva parametra tipa int (kreira matricu datih dimenzija popunjenu nulama), te konstruktor koji prima dimenzije matrice i vektor koji se sadrži od nenultih elemenata matrice, a elementi vektora su tipa pair<pair,double>, pri čemu prvi element para predstavlja red i kolonu elementa, a drugi njegovu vrijednost.

U nastavku ove dokumentacije ćemo opisati (koliko je to moguće) funkcionalnosti implementiranih funkcija, njihovih parametara, primjene I slično.

Klasa izgleda ovako:

projmatrica.cpp | projmatrica.h

```
1  #ifndef PROJMATRICA_H_INCLUDED
2  #define PROJMATRICA_H_INCLUDED
3  #include<iostream>
4  #include<algorithm>
5  #include<utility>
6  using namespace std;
7
8  struct node {
9    node *nextcol; //pokazivac na sljedeci u koloni
10   node *nextrow; //pokazivac na sljedeci u redu
11   int row; //red
12   int col; //kolona
13   int value; //vrijednost
14 };
15
16 class matrica {
17   node **colhead; //pokazivac kolona
18   node **rowhead; // pokazivac reda
19   int rows,cols; //varijabla broj redova i kolona
20   public:
21     matrica(int m, int n); //matrica sa m redova i n kolona
22     matrica(int m, int n, vector<pair<pair<int,int>,double>>);
23     void sabiranje(matrica b); //sabiranje 2 matrica
24     void oduzimanje(matrica b); //oduzimanje 2 matrica
25     void mnozenje(matrica b); //mnozenje matrica
26     void stepenovanje(matrica b); //stepenovanje matrica
27     void trans(matrica b); //transponovanje matreica
28     void insert_remove(int row,int col,float value);//dodajemo elemente ili ih izbacujemo ako je vrijednost elementa 0
29                                                     //pomocna funkcija
30     int getRed() //geter broja redova matrice
31     {
32         return rows;
33     }
34     int getKol() //geter broja kolona matrice
35     {
36         return cols;
37     }
```

IMPLEMENTIRANE FUNKCIJE

U nastavku ćemo reći nešto više o implementiranim funkcijama.

Na samom početku smo implementirali strukturu nazvanu node. Strukturu koristimo za jednostavniju implementaciju rijetke ili engl sparse. S obzirom da koristimo liste potrebno nam je da koristimo liste za redove i kolone, pa s toga koristimo dva pokazivača, na sljedeći u ovom redu i sljedeći u ovoj koloni(*nextrow, *nextcol). Također čuvamo I podatke o redu i koloni kao i vrijednost koja treba da bude spašena.

Sada možemo preći na implementaciju rijetke matrice. Kao private atriute matrice čuvamo dva dvostruka pokazivača, jedan na elemnt kolone drugi na element reda. Isto tako čuvamo i broj redova i kolona.

Na početku imamo konstruktor koji treba da formira matricu dimenzije mxn, koje se proslijeđuju kao parametri. Da kreiranje takve matrice koristimo se struktorom koju smo ranije kreirali te privatnim atributima klase.

Pređimo sada na operacije sabiranja, oduzimanja i množenja.
Da bismo izračunali vrijednost elementa u novoj matrici, kao prvo moramo da nađemo elemnte u obje matrice respektivno. Ako ne postoji barem jedan element to znači da rezultirajuća matrica nema vrijednost u toj ćeliji (ćeliju čine red i kolona tj par (red,kol)). Ako dođe slučaj da barem jean elemnt postoji onda računamo vrijednost elemnta koji se nalazi u toj ćeliji. Nakon što izračunamo vrijednost tog elementa (podrazumjeva se da je vrijednost elemnta nenulta) tada spašavamo vrijednost tog elementa. Tačnije, za sabiranje matrica koristimo se klasičnim pravilom prvi elemnt prve vrste prve matrice sabiramo sa prvim elementom prve vrste matrice, pa drugi sa drugim i tako redom. Dimenzije matrice moraju biti jednake, tj matrice moraju biti jednakih dimenzija. Ukoliko dimenzije matrice nisu adekvatne ispisuje se odgovarajuća poruka.

Analogno radimo sa operacijom oduzimanja.

Sličnim postupkom kao i za sabiranje matrica implementirat ćemo i funkciju za množenje dvije matrice. Da bismo pomnožili dvije matrice dimenzije matrice moraju biti reguralne. Dimenzije matrice za množenje su regularne ako je broj kolona prve matrice jednak broju vrsta druge matrice. Pa na početku provjeravamo da li su matrice saglasnih dimenzija. Ukoliko nisu ispisujemo poruku da dimenzije nisu saglasne. Ukoliko jesu prelazimo na množenje. Množimo prvi elemnte vrste prve matrice sa prvim elementom kolone druge matrice, pa na to dodajemo drugi element vrste prve matrice pomnožen

sa drugim elementom prve kolone druge matrice i tako redom itd. Vodeći se ovim implementirali smo funkciju za množenje matrica.