*Conversia reprezentarii grafelor.*

*De la matricea de adiacenta la lista predecesorilor.*

1. Formularea corecta completa si concreta a problemei

Sa se faca conversia de la reprezentarea maticeala de adiacenta la reprezentarea cu tablouri unidimensionale data de lista predecesorilor a unui graf.

Se da :

Repezentarea matriceala de adiacenta a grafului,

Se cere:

Sa se construiasca lista predecesorilor.

2. Modelarea problemei (mod de abordare)

Fie G=(X,U) graf, unde X-multimea varfurilor, n=│X│; U-multimea arcelor, m=│U│.

X={1..n}

U={1..m}

2.1. Definirea elementelor

Matricea de adiacenta este o matrice patratica, linia si coloana i corespunzand varfului i al grafului. Elementele matricei de adiacenta se definesc astfel:

1, (i,j) ∈ U, i,j ∈ X

aij = 0, (i,j) ∉ U, i,j ∈ X

Lista predecesorilor este reprezentata de vectorii a si b, unde

i-1

ai = ∑ gˉ (j) +1, a are dimensiunea n+1 si an+1=m+1,

j=1

g ˉ(i)= ai+1- ai, , pentru grafe orientate,

g(i)= ai+1- ai , pentru grafe neorientate,

Гˉi={b(ai),b(ai)+1),.., b((ai+1)-1)}

Pentru fiecare varf i ∈ X, ai indica indicele din tabloul b unde sunt inregistrati predecesorii sai, predecesorii varfului i se afla, deci, in tabloul b intre pozitiile ai si ai+1-1 inclusiv, iar daca ai= ai+1, atunci varful i nu are nici un predecesor.

2.2. Metoda de rezolvare:

In elaborarea acestui algoritm folosim observatia ca fiecare valoare nenula din A defineste (refera) un arc si acesta defineste o pozitie in vectorul beta. Deci, numarul valorilor nenule din A este dimensiunea lui beta. Daca se parcurge matricea A pe coloane valorile nenule depistate in ea, definesc predecesorii indicelui de linie in care se afla valoare nenula. Daca se introduc acesti indici de linie succesiv in vectorul beta, acesta va contine multimile Гˉ1, Гˉ2,…, Г ˉn. La schimbarea coloanei se va retine in vetorul alfa pozitia ultimului element memorat in beta plus 1.

3. Descrierea algoritmului Pseudocod

Date: n,(Aij, i=1,..,n, j=1,..,n)

Rezultate: (ai , i=1,..,n), (bj , j=1,..,n\*(n-1)).

Algoritmul transform este:

citeste n, A

k:=1

pentru j := 1,n executa

*alfa*(j) := k

pentru i := 1,n executa

daca aij != 0 atunci

*beta*(k) := i

k := k + 1

sfarsit daca

sfarsit pentru

sfarsit pentru

*alfa*(n+1) := k

Sfarsit algoritm transform

4. Date de test

|  |  |
| --- | --- |
| Date de intrare | Date de iesire |
| 0 1 1 1 0  0 0 0 1 0  0 0 0 0 1  0 0 1 0 1  0 0 0 0 0 | a: 1 1 2 4 6  b: 1 1 4 1 2 3 4 |
| 0 1 0 1 0 0  1 0 1 0 0 0  0 1 0 1 1 0  1 0 1 0 0 0  0 0 1 0 0 0  0 0 0 0 0 0 | a: 1 3 5 8 10 11  b: 2 4 1 3 2 4 5 1 3 3 |