*Diametrul grafului*

1. Formularea corecta completa si concreta a problemei

Sa se elaboreze un algoritm care determina diametrul unu graf conex G = (X,U) dat.

2. Modelarea problemei (mod de abordare)

2.1 Descrierea algoritmului

Dacă A este matricea de adiacenta a unui graf de ordin n, atunci matricea Ap

indica prin elementul de pe linia i si coloana j numarul de drumuri de lungime p de la varful i la varful j.

2.2 Metoda de rezolvare

Deoarece raza grafului se define te în mod indirect utilizând notiunea de lant , iar lanturile se definesc doar pentru grafuri neorientate, primul pas trebuie sa fie transformarea grafului într-un graf neorientat.

Algoritmul va calcula apoi matricea distantelor D folosind teorema enuntatt anterior, în următorul mod:

* + fiecare element al matricii D = (dij)1<=i, j <= n, i ≠ j va fi initializat cu ∞
  + pentru p = 1 .. n se va calcula matricea Ap = (apij)1<= i, j< = n
  + pentru un p fixat dij = p, daca dij = ∞ si apij ≠ 0

Utilizând matricea D, se va calcula vectorul excentricitatilor varfurilor:

* + E = (ei)i <= i <= n , ei = max { dij | j ∈ X }

In final algoritmul va calcula diametrul grafului R conform definitiei

* + R = min { ei | i ∈ X }

3. Descrierea algoritmului Pseudocod

Date: n, A

Rezultate: d - diametrul grafului

Algoritmul diametru este:

functia **diametru**(A, n) este:

A' := simetrizare(A, n)

D := distante(A', n)

E := excentricitati(D, n)

raza := maxim(E, n)

sfarsit functie

functia **simetrizare**(A, n) este:

pentru i := 1 .. n executa

pentru j := i+1 .. n executa

A'[i,j] := A'[j,i] := max(A[i,j], A[j,i])

sfarsit pentru

sfarsit pentru

sfarsit functie

functia **distante**(A, n) este:

pentru i := 1 .. n executa

pentru j := 1 .. n executa

D[i,j] := ∞

sfarsit pentru

D[i,i] := 0

sfarsit pentru

AP := A

p := 1

repeta

modificat := false

pentru i := 1 .. n executa

pentru j := 1 .. n executa

daca D[i,j] = ∞ si AP[i,j] ≠ 0 atunci

D[i,j] := p

modificat := true

sfarsit daca

sfarsit pentru

sfarsit pentru

AP := inmultire(AP, A, n)

p := p + 1

pana cand modificat = false sau p > n

distante := D

sfarsit functie

functia **inmultire**(A, B, n) este:

pentru i := 1 .. n EXECUTA

pentru j := 1 .. n EXECUTA

AB[i,j] := 0

pentru k := 1 .. n EXECUTA

AB[i,j] := AB[i,j] + A[i,k] \* B[k,j]

sfarsit pentru

sfarsit pentru

sfarsit pentru

inmultire := AB

sfarsit functie

functia **excentricitati**(D, n) este:

pentru i := 1 .. n EXECUTA

E[i] := maxim(D[i], n)

sfarsit pentru

excentricitati := E

sfarsit functie

functia **maxim**(V, n) este:

m := -∞

pentru i := 1 .. n EXECUTA

m := max(V[i], m)

sfarsit functie

maxim := m

sfarsit functie

Sfarsit algoritm.

4. Date de test

|  |  |
| --- | --- |
| Date de intrare | Date de iesire |
| n = 8  0 1 1 1 1 0 0 0  1 0 1 1 0 0 1 0  1 1 0 1 0 1 0 0  1 1 1 0 0 0 0 1  1 0 0 0 0 0 0 0  0 0 1 0 0 0 0 0  0 1 0 0 0 0 0 0  0 0 0 1 0 0 0 0 | 3 |