

Dumitru Fanache

**TEORIA ALGORITMICĂ
A GRAFURILOR**
NOȚIUNI FUNDAMENTALE

Volumul I

ÎNVĂȚARE DE EXCELENȚĂ[®]
supersucces



Cuprins

<i>Cuvânt-înainte</i>	9
Capitolul I. Noțiuni generale despre grafuri	15
I.1. Scurt istoric al teoriei grafurilor	15
I.2. Teoria grafurilor în optimizarea combinatorie.....	18
I.3. Câteva aplicații de colorare și acoperire cu vârfuri	20
I.4. Utilizarea grafurilor în biologia computațională	22
I.5. Grafuri neorientate, incidență, izomorfism.....	24
I.6. Subgrafuri, supergrafuri, grafuri parțiale, cicluri.....	28
I.7. Conexitate, operații cu grafuri	34
I.8. Probleme propuse pentru grafuri neorientate.....	42
I.9. Grafuri orientate, mulțimi dominante	48
I.10. Algoritmul Havel–Hakimi	57
I.11. Probleme propuse pentru grafuri orientate	60
I.12. Probleme rezolvate	63
Capitolul II. Memorarea grafurilor	72
II.1. Reprezentarea matriceală a grafurilor neorientate.....	72
II.1.1. Spațiul vectorial asociat unui graf	72
II.1.2. Matricea de adiacență.....	73
II.1.3. Matricea de incidență	84
II.1.4. Matricea ciclurilor	88
II.1.5. Matricea lanțurilor între două vârfuri fixate.....	91
II.1.6. Matricea ciclurilor fundamentale	93
II.2. Reprezentarea matriceală a grafurilor orientate	95
II.3. Graful tranzitiv asociat unui graf orientat	100
II.4. Reprezentarea grafurilor prin liste de adiacență.....	102
II.5. Probleme rezolvate	105
II.6. Probleme propuse	119
Capitolul III. Arbori de acoperire de cost minim	128
III.1. Arbori, proprietăți	128
III.2. Reprezentarea mulțimilor disjuncte prin păduri de arbori	132
III.3. Caracterizări ale arborilor de acoperire.....	138
III.4. Arbori de acoperire în grafuri ponderate.....	140
III.4.1. Algoritmul lui Prim.....	144
III.4.2. Algoritmul lui Kruskal	147
III.4.3. Algoritmul lui Borůvka.....	152

III.5. Arbori Steiner.....	153
III.6. Enumerarea arborilor	157
III.6.1. Reprezentarea arborilor prin codul lui Prüfer	157
III.6.2. Teorema matriceală pentru arbori	162
III.7. Probleme propuse.....	165
Capitolul IV. Algoritmi de traversare.....	178
IV.1. Traversarea în lățime	178
IV.2. Implementări ale algoritmului BFS	185
IV.3. Traversarea în adâncime	187
IV.3.1. Principii de execuție a traversării în adâncime	187
IV.3.2. Proprietăți ale căutării în adâncime.....	189
IV.3.3. Clasificarea muchiilor.....	192
IV.4. Implementări ale algoritmului DF	194
IV.5. Aplicații ale traversării grafurilor	196
IV.5.1. Componente conexe.....	196
IV.5.2. Verificarea existenței unui lanț între două vârfuri	197
IV.5.3. Distanță minimă într-un graf grid	199
IV.5.4. Componente tari conexe	201
IV.5.4.1. Algoritmul Kosaraju–Sharir	201
IV.5.4.2. Algoritmul lui Tarjan.....	204
IV.5.5. Sortarea topologică a vârfurilor unui graf.....	206
IV.5.6. Puncte de articulație.....	209
IV.5.7. Muchii critice.....	213
IV.5.8. Componente biconexe.....	214
IV.5.9. Supraveghere eficientă.....	217
IV.5.10. Reamenajări cu costuri minime	218
IV.5.11. Excentricitatea unui vârf, diametrul unui graf	220
IV.5.12. Partiționarea mulțimii vârfurilor unui graf	222
IV.6. Probleme propuse	223
Capitolul V. Drumuri optime, centre, mediane.....	228
V.1. Problema celui mai scurt drum	228
V.2. Drumuri minime cu sursă unică	229
V.2.1. Algoritmul lui Dijkstra.....	229
V.2.2. Analiza algoritmului lui Dijkstra	238
V.2.3. Algoritmul lui Dantzig	240
V.2.4. Algoritmul lui Bellman–Ford.....	243
V.2.5. Algoritmul lui Bellman–Kalaba.....	247
V.3. Drumuri minime între toate perechile de vârfuri	250
V.3.1. Algoritmul Roy–Floyd–Warshall.....	250
V.3.2. Comparație între algoritmul lui Floyd și algoritmul lui Dijkstra	253

V.3.3. Determinarea traseelor drumurilor minime.....	255
V.4. Algoritmul lui Johnson.....	257
V.5. Centrul relativ al unui graf.....	257
V.6. Centrul absolut al unui graf, metoda Hakimi.....	259
V.7. Mediana unui graf.....	262
V.8. Problema acoperirii cu vârfuri.....	263
V.9. Probleme propuse.....	269
Capitolul VI. Grafuri hamiltoniene, grafuri euleriene.....	275
VI.1. Grafuri hamiltoniene.....	275
VI.2. Algoritmi pentru determinarea ciclurilor hamiltoniene.....	279
VI.2.1. Problema comis-voiajorului.....	282
VI.2.2. Algoritmul lui Foulkes.....	285
VI.2.3. Algoritmul lui Chen.....	287
VI.2.4. Algoritmul lui Kauffman.....	291
VI.3. Grafuri euleriene.....	294
VI.3.1. Construcția lanțurilor și ciclurilor euleriene.....	298
VI.3.2. Algoritmul lui Fleury.....	299
VI.3.3. Algoritmul lui Hierholzer.....	302
VI.4. Algoritmul lui Christofides.....	306
VI.5. Problema poștaşului chinez.....	308
VI.6. Probleme propuse.....	310
Bibliografie.....	316



I.1. Scurt istoric al teoriei grafurilor

În problemele care apar în matematică, economie, inginerie, în general, și în multe alte domenii, apare adeseori necesitatea reprezentării unor relații arbitrare între diferite obiecte, respectiv a interconexiunilor dintre acestea.

Spre exemplu, dându-se traseele aeriene ale unui stat, se cere să se precizeze drumul optim dintre două orașe. Criteriul de optimalitate poate fi timpul sau prețul, drumul optim putând să difere pentru cele două situații.

Circuitele electrice sunt alte exemple în care interconexiunile dintre obiecte joacă un rol central. Piese (tranzistoare, rezistențe, condensatoare) sunt interconectate prin fire electrice. Astfel de circuite pot fi reprezentate și prelucrate de către un sistem de calcul în scopul rezolvării unor probleme simple, cum ar fi: *Sunt toate piesele date conectate în același circuit?* Sau a unor probleme mai complicate, de tipul: *Este funcțional un anumit circuit electric?*

Un al treilea exemplu îl reprezintă planificarea activităților, în care obiectele sunt *task-uri* (activități, procese), iar interconexiunile precizează care dintre activități trebuie finalizate înaintea altora. Întrebarea la care trebuie să oferim un răspuns este: *Când trebuie planificată fiecare activitate?*

Structurile de date care pot modela în mod natural situații de genul celor de mai sus sunt cele derivate din conceptul matematic cunoscut sub denumirea de *graf*.

Probleme care conduceau implicit la grafuri erau cunoscute de foarte multă vreme, fără să se cunoască însă prezența acestora.

Oficial, actul de naștere al teoriei grafurilor poate fi reprezentat de rezolvarea *problemei celor șapte poduri din Königsberg* de către *L. Euler* în anul 1736: două insule *A* și *B*, formate de râul Prigel în Königsberg (Prusia orientală, acum orașul Kaliningrad din vestul Rusiei) sunt conectate între ele și de malurile *C* și *D* prin șapte poduri, *fig.I.1.(a)*, formând, în acea vreme, patru cartiere. În plimbările lor, localnicii încercau să traverseze toate cele șapte poduri, astfel încât, plecând dintr-un anumit cartier, să se înapoieze în acesta trecând o singură dată peste fiecare pod. Nu reușeau să facă acest lucru.

I.9. Grafuri orientate, mulțimi dominante

Definiția I.37. Fie V o mulțime finită și nevidă. Numim graf orientat (digraf) orice pereche $G = (V, E)$ în care $E \subset V \times V$ este o mulțime finită de perechi ordonate cu componente din V (E este o relație binară pe V).

Elementele mulțimii V se numesc vârfuri sau noduri, iar elementele lui E se numesc arce. Orice arc are forma (a, b) , în care a se numește extremitate inițială, iar b se numește extremitate finală a arcului (a, b) .

Exemplu: Fie graful reprezentat în fig. I.24, unde mulțimea vârfurilor este $V = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ și mulțimea arcelor $E \subset V \times V$: $E = \{(1, 2), (2, 3), (3, 4), (4, 5), (5, 6), (6, 7), (7, 4), (8, 7), (3, 8), (8, 9), (9, 2)\}$.

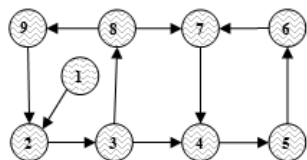


Fig. I.24. Un graf orientat cu 9 vârfuri

Un graf orientat se reprezintă printr-o mulțime de puncte corespunzătoare vârfurilor și printr-o mulțime de segmente orientate (săgeți) corespunzătoare arcelor. O săgeată este orientată de la extremitatea inițială spre extremitatea finală a arcului pe care îl reprezintă. Dacă $u = (x, y) \in E$, spunem că x și y sunt adiacente în G și că nodurile x și y sunt incidente arcului u sau arcul u este incident nodurilor x și y . Mai exact, spunem că u este incident exterior nodului x (u pleacă sau iese din x) și că u este incident interior nodului y (u ajunge sau intră în y). Arcul $(x, y) \in E$ se mai notează și prin xy .

Definiția I.38. Pentru graful $G = (V, E)$ mulțimea $S \subseteq V$ se numește mulțime exterior stabilă dacă $(\forall) x_j \notin S, \exists (x_i, x_j), x_i \in S$. Mulțimea S este una dominantă dacă $S \cup \Gamma^+(S) = V$.

Definiția I.39. Mulțimea dominantă S se numește minimă dacă nu există o altă mulțime dominantă S' pentru care este îndeplinită condiția $S' \subseteq S$.

Nu toate mulțimile minime dominante au același număr de vârfuri $|S|$. Prin urmare, modul de selecție a celei mai bune mulțimi dominante va depinde de condițiile inițiale ale problemei studiate.

Definiția I.40. Fie Q mulțimea tuturor mulțimilor dominante ale grafului $G = (V, E)$. Numărul $\beta_0(G) = \min_{S \in Q} |S|$ se va numi indice de dominanță (număr de stabilitate externă) a grafului G , iar mulțimea S^* pentru care el se obține – mulțime dominantă de putere minimă.

Legătura dintre mulțimile maxim independente și mulțimile dominante este evidentă. Algoritmul folosit pentru determinarea mulțimilor dominante va fi organizat după o tehnică similară celui pentru determinarea mulțimilor independente.

Dumitru Fanache

**TEORIA ALGORITMICĂ
A GRAFURILOR**
REȚELE, CUPLAJE,
COLORĂRI, PLANARITATE

Volumul II

ÎNVĂȚARE DE EXCELENȚĂ[®]
supersucces



Cuprins

Capitolul VII. Rețele de flux, drumuri critice	7
VII.1. Rețele de flux.....	7
VII.2. Drumuri de ameliorare	11
VII.3. Algoritmul Ford–Fulkerson.....	13
VII.3.1. Metoda saturării celui mai scurt drum.....	14
VII.3.2. Algoritmul Edmonds–Karp	16
VII.4. Algoritmul lui Dinic	20
VII.5. Teorema de flux maxim – tăietură minimă	26
VII.6. Problema cuplajului maxim de cost minim tratată ca o problemă de flux maxim	30
VII.7. Drumuri critice în grafuri de activități.....	35
VII.8. Probleme propuse	43
Capitolul VIII. Cuplaje în grafuri bipartite	45
VIII.1. Considerații generale despre cuplaje.....	45
VIII.2. Determinarea unui cuplaj maxim folosind o rețea de transport	48
VIII.3. Algoritmul ungar pentru determinarea unui cuplaj maxim	52
VIII.4. Algoritmul Hopcroft–Karp.....	59
VIII.5. Algoritmul Kuhn–Munkres.....	63
VIII.6. Probleme propuse.....	68
Capitolul IX. Colorarea grafurilor	71
IX.1. Colorarea vârfurilor	71
IX.1.1. Numărul cromatic al unui graf.....	72
IX.1.2. Polinomul cromatic.....	77
IX.1.3. Teorema Birkhoff–Lewis.....	79
IX.1.4. Algoritmi clasici de colorare a vârfurilor unui graf	85
IX.2. Colorarea muchiilor	96
IX.2.1. Indexul cromatic al unui graf.....	96
IX.2.2. Graful conjugat asociat unui graf.....	97
IX.2.3. Incidența cromatică.....	101
IX.2.4. Colorarea muchiilor grafurilor bipartite.....	102
IX.2.5. Teorema lui Vizing, clasificarea grafurilor.....	106
IX.3. Elemente de teoria extremală a grafurilor.....	111
IX.4. Probleme propuse	117
Capitolul X. Grafuri planare, grafuri aleatoare	121
X.1. Caracterizarea grafurilor planare.....	121
X.2. Teorema lui Kuratowski.....	126

X.3. Reprezentarea pe o g -sferă a unui graf.....	133
X.4. Planaritate și hamiltonietate	136
X.5. Algoritmi pentru desenarea/testarea planarității unui graf	138
X.5.1. Algoritmul forței directe	139
X.5.2. Algoritmul Hopcroft–Tarjan	140
X.5.3. Algoritmul lui Tutte	142
X.5.4. Algoritmul lui Schnyder.....	147
X.6. Dualul unui graf planar	147
X.7. Relația dintre un graf și dualul său.....	148
X.8. Teorema celor patru culori.....	151
X.9. De ce avem nevoie de grafuri aleatoare?	153
X.10. Modele de grafuri aleatoare	154
X.11. Grafuri aleatoare cu generatorul Mersenne–Twister.....	159
X.12. Probleme propuse.....	162
Capitolul XI. Biblioteci de grafuri	170
XI.1. Matgraph Library	170
XI.2. Apelarea funcțiilor în Matgraph prin exemple.....	173
XI.2.1. Adăugări și ștergeri.....	174
XI.2.2. Vecini și grade	176
XI.2.3. Partiții	179
XI.2.4. Permutări.....	181
XI.3. Operații cu grafuri.....	182
XI.3.1. Calcule în grafuri	186
XI.3.2. Colorarea grafurilor	188
XI.4. Boost Graph Library	190
XI.4.1. Construirea unui graf	192
XI.4.2. Accesarea mulțimii vârfurilor.....	193
XI.4.3. Accesarea mulțimii muchiilor.....	194
XI.4.4. Structura de adiacență a grafului	194
XI.4.5. Arce de ieșire, arce de intrare, descriptori de muchii	195
XI.4.6. Vârfuri adiacente	197
XI.4.7. Algoritmul lui Dijkstra în BGL	197
XI.4.8. Utilizarea conceptului de vizitator	199
XI.4.9. Parcurgerea DF folosind BGL	202
XI.4.10. Un arbore de acoperire minim cu BGL.....	202
Bibliografie	204

capitolul

VIII

Rețele de flux, drumuri critice

VII.1. Rețele de flux

Definiția VII.1. O rețea de flux este un graf orientat ponderat și fără circuite $G = (V, E)$ ce îndeplinește condițiile următoare:

- ✓ există un vârf unic $s \in V$ în care nu intră niciun arc (sursa rețelei);
- ✓ există un vârf unic $t \in V$ din care nu iese niciun arc (destinația rețelei);
- ✓ G este conex și există drumuri de la s la t ;
- ✓ s-a definit o funcție $c : E \rightarrow \mathbb{R}$ astfel încât $c(u) \geq 0$ pentru $(\forall) u \in E$, numărul $c(u)$ se numește capacitatea rețelei.

Problema determinării fluxului maxim este cea mai simplă problemă legată de rețelele de flux (transport), care cere să se determine cantitatea maximă de material care poate fi transportată de la sursă la destinație ținând cont de restricțiile de capacitate.

Definiția VII.2. Un flux într-o rețea de transport $G = (V, E)$ este o funcție $\varphi : V \times V \rightarrow \mathbb{R}^+$, care respectă următoarele condiții:

- ✓ $(\forall) u, v \in V, c(u, v) \geq \varphi(u, v)$ (limitarea impusă de capacități);
- ✓ $(\forall) u, v \in V, \varphi(u, v) = -\varphi(v, u)$ (antisimetrie);
- ✓ $(\forall) u \in V, u \neq s, t, \text{ unde } s, t \in V, \sum_{v \in V} \varphi(u, v) = 0$ (conservarea fluxului).

Observații: Aceste 3 condiții reies și intuitiv:

- Fluxul pe fiecare muchie nu trebuie să depășească capacitatea muchiei;
- Fluxul dintre nodurile u și v care nu sunt legate prin niciun arc, nu poate fi decât zero. Dacă $(u, v) \notin E$ și $(v, u) \notin E$, atunci $c(u, v) = c(v, u) = 0$; din cauza restricției de capacitate avem $\varphi(u, v) \leq 0$ și $\varphi(v, u) \leq 0$. Dar, din condiția de antisimetrie avem $\varphi(u, v) = -\varphi(v, u)$, prin urmare $\varphi(u, v) = \varphi(v, u) = 0$. Rezultă că existența fluxului nenul între vârfurile u și v implică $(u, v) \in E$ sau $(v, u) \in E$ sau ambele;
 - Condiția de conservare a fluxului afirmă că pentru orice vârf x cu $x \neq s$ și $x \neq t$, suma fluxurilor de pe arcele care intră în x este egală cu suma fluxurilor pe arcele care ies din x .
 - Funcția φ nu este unică.

VII.3.2. Algoritmul Edmonds–Karp

Așa cum am văzut, *algoritmul Ford–Fulkerson* nu definește o metodă de alegere a drumului de ameliorare pe baza căruia se modifică fluxul în graf.

Edmonds³ și Karp au sugerat ca metodă de construcție a fluxului φ^* în rețeaua reziduală *determinarea celui mai scurt drum între sursă și destinație folosind o căutare în lățime în graful rezidual unde fiecare arc are ponderea 1 și saturarea acestui drum*.

Se observă intuitiv că fiecare execuție a acestui proces saturează cel puțin o muchie. După $O(m)$ astfel de execuții, toate drumurile minime dintre sursă și destinație sunt saturate și distanța dintre sursă și destinație trebuie să crească. Aceasta implică faptul că după $O(n \times m)$ astfel de execuții, distanța dintre s și t va deveni mai mare decât n , sau altfel spus, t nu va mai fi accesibil din s . Folosind *algoritmul Edmonds–Karp* pentru a determina fluxul φ^* în rețeaua reziduală, complexitatea algoritmului Ford–Fulkerson devine $O(n \times m^2)$ (vezi codul asociat algoritmului din *fig. VII.8*).

```

Input  $G=(V,E),s,t$ 
Output  $\varphi$ 
while ( $BFS(G_{\varphi},s,t)=true$ ) do
   $P \leftarrow (s \rightarrow t)$ 
  find  $\varphi^* = \min\{(i,j) \mid (i,j) \in P\}$ 
  for each  $(i,j) \in P$  do
     $c_{ij} \leftarrow c_{ij} - \varphi^*$ 
     $c_{ji} \leftarrow c_{ji} + \varphi^*$ 
   $\varphi \leftarrow \varphi + \varphi^*$ 
return  $\varphi$ 

```

Fig. VII.6. Algoritmul Edmonds–Karp

În urma execuției *algoritmului Edmonds–Karp*, obținem pentru rețelele din *fig. VII.1, fig. VII.5* rezultatele din *fig. VII.7(a, b)*.

³ **Jack Edmonds**, matematician canadian, considerat ca unul din cei mai importanți specialiști în optimizare combinatorie; în 1985 a primit premiul John von Neumann.

VIII.2. Determinarea unui cuplaj maxim folosind o rețea de transport

Pentru a determina cuplajul maxim putem transforma graful bipartit într-o rețea de transport adăugând un nod sursă (s) și un nod destinație (t) și vom atribui capacitatea 1 pe toate arcele (vezi fig. VIII.3). În acest fel am transformat o problemă de afectare într-o problemă de flux (vezi cap. VII).

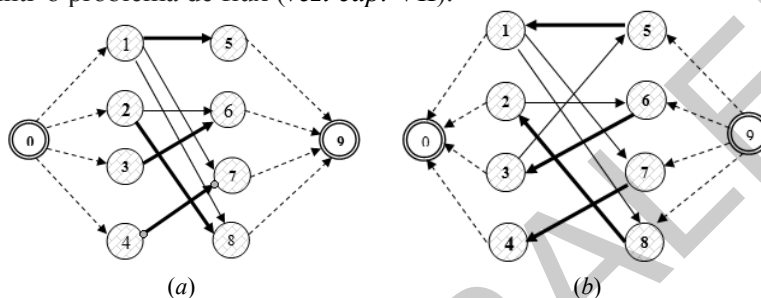


Fig.VIII.3. Transformarea unei probleme de cuplaj într-o problemă de flux
 (a) rețeaua de transport inițială obținută prin adăugarea a două vârfuri la graful bipartit;
 (b) rețeaua a fost modificată prin inversarea arcelor pe drumurile de creștere

a) Cuplaj maxim pentru un graf memorat prin liste de adiacență

Pentru graful bipartit din fig. VIII.3, am adăugat vârfurile 0 și 9, ca fiind sursa și destinația rețelei de transport. Listele de adiacență, gradele vârfurilor precum și drumurile de creștere de la vârfurile sursă(0) și destinație (9) determinate prin parcurgerea BF sunt date în fig. VIII.4.

Liste	$d^-(x)$	$d^+(x)$	Drumuri de creștere
0: 1, 2, 3, 4	0	4	I) . 0, 1, 5, 9 II) . 0, 2, 8, 9 III) . 0, 3, 6, 9 IV) . 0, 4, 7, 9
1: 5, 7, 8	1	3	
2: 6, 8	1	2	
3: 5, 6	1	2	
4: 7	1	1	
5: 9	2	1	
6: 9	2	1	
7: 9	2	1	
8: 9	2	1	
9: 0	4	0	

Fig.VIII.4. Lista de adiacență a vârfului de start nu este vidă

După determinarea unui drum de ameliorare, se schimbă sensul arcelor drumului respectiv și evident că listele de adiacență se vor modifica astfel că, după patru pași ele vor arăta ca în fig. VIII.5. Observăm că lista de adiacență a vârfului de start (0) devine vidă.