Cuplaje maxime

Capitolul

10

- ❖ **Cuplaje**
- ❖ **Grafuri bipartite**
- ❖ **Cuplaje maxime în grafuri bipartite**
- ❖ **Rezumat**
- ❖ **Implementări sugerate**
- ❖ **Probleme propuse**
- ❖ **Soluțiile problemelor**

În cadrul acestui capitol vom prezenta noțiunea de cuplaj în graf, vom defini grafurile bipartite şi vom arăta modul în care poate fi determinat un cuplaj maxim într-un graf bipartit.

10.1. Cuplaje

Un ***cuplaj*** într-un graf poate fi definit ca o submulțime a mulțimii muchiilor astfel încât această mulțime să nu conțină muchii adiacente.

Practic vom lega nodurile două câte două, fiecare nod fiind legat cel mult o dată de un alt nod. Aşadar, pentru un graf care conține *N* noduri, un cuplaj va consta din cel mult [*N* / 2] muchii.

De exemplu, pentru graful din figura 10.1 muchiile îngroşate sunt cele care fac parte din cuplaj.

Evident, un cuplaj maxim reprezintă cuplajul pentru care cardinalul mulțimii muchiilor alese este cât mai mare posibil.

Cuplajul din figura 10.1 este maxim, deoarece graful conține zece noduri, iar cuplajul conține cinci muchii (numărul maxim posibil).

Totuşi, nu în orice graf vom putea determina un cuplaj care să conțină [*N* / 2] muchii.

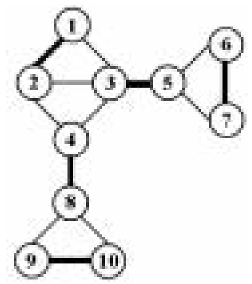


Figura 10.1: Cuplaj într-un graf oarecare

10.2. Grafuri bipartite

Un ***graf bipartit*** este un graf ale cărui noduri pot fi partiționate în două submulțimi disjuncte astfel încât să nu existe nici o muchie (arc) care să unească două noduri aflate în aceeaşi submulţime. În cazul grafurilor neorientate toate muchiile vor uni perechi de noduri aflate în submulţimi diferite. În cazul grafurilor orientate toate arcele vor porni de la noduri aflate în una dintre cele două submulţimi şi vor ajunge la noduri aflate în cealaltă submulţime.

Un graf bipartit cu şase noduri este prezentat în figura 10.2. Una dintre mulţimi este formată din nodurile 1, 2 şi 3, iar cealaltă din nodurile 4, 5 şi 6.

Este uşor de observat în imagine că toate muchiile au una dintre extremităţi în mulţimea {1, 2, 3} şi cealaltă extremitate în mulţimea {4, 5, 6}.

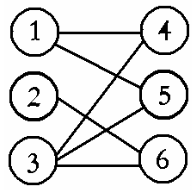


Figura 10.2: Graf bipartit

Deşi pentru graful considerat cele două mulţimi au acelaşi număr de elemente, această condiţie nu trebuie obligatoriu îndeplinită.

10.3. Cuplaje maxime în grafuri bipartite

În cadrul acestei secţiuni vom particulariza noţiunea de cuplaj pentru grafuri bipartite, vom defini problema determinării cuplajului maxim în astfel de grafuri şi vom arăta cum poate fi redusă această problemă la determinarea unui flux maxim într-o reţea de transport.

10.3.1. Preliminarii

Este evident faptul că pentru grafurile bipartite cuplajul va consta din stabilirea unei "corespondenţe" între nodurile din prima mulţime şi nodurile din cea de-a doua.

Fiecărui nod din prima mulţime îi va corespunde cel mult un nod din cea de-a doua, deci această corespondenţă trebuie să fie o funcţie injectivă.

De exemplu, pentru graful din figura 10.2 un cuplaj ar putea fi: (1, 4), (2, 6), (3, 5). Aşadar nodurilor 1, 2 şi 3 le corespund nodurile 4, 6 şi 5 (în această ordine).

Evident, numărul muchiilor care fac parte din cuplajul maxim va fi limitat de cardinalul celei mai "mici" dintre cele două mulţimi disjuncte. Putem trage concluzia că pentru graful din figura 10.2 cuplajul prezentat anterior este maxim, deoarece conţine trei muchii şi fiecare dintre cele două mulţimi conţine trei elemente.

10.3.2. Transformarea problemei

Pentru a determina cuplajul maxim într-un graf bipartit va trebui să alegem cât mai multe muchii, fără ca printre muchiile alese să avem două care au aceeaşi extremitate.

Vom arăta în continuare cum vom rezolva această problemă transformând graful bipartit într-o reţea de transport.

Pentru început, dacă graful bipartit este neorientat, vom stabili o orientare a muchiilor (care vor deveni arce) astfel încât ele să plece de la noduri aflate în una dintre mulţimi şi să ajungă la noduri aflate în cealaltă mulţime. De asemenea, vom stabili că fiecare dintre aceste arce va avea capacitatea 1.

Pentru a obține o rețea de transport avem nevoie de o sursă şi o destinație. Aceste noduri vor fi introduse artificial şi vor fi legate de nodurile grafului bipartit.

De la sursă vor pleca arce spre toate nodurile din prima dintre submulțimi (considerăm că prima submulțime este cea care conține noduri din care pleacă arce în graful bipartit), iar la destinație vor ajunge arce dinspre toate nodurile din cea de-a doua submulțime. Pentru sursa şi destinația introduse artificial sunt folosite uneori denumirile de ***sursă virtuală*** şi ***destinație virtuală***. Toate arcele care pleacă de la sursă şi toate arcele care ajung la destinație vor avea capacitatea 1.

După construirea rețelei de transport vom determina fluxul maxim în rețeaua obținută. Datorită faptului că arcele adiacente sursei şi destinației au capacitatea 1, fiecare nod va apărea o singură dată ca extremitate a unui arc pentru care fluxul este nenul.

Ca urmare, după determinarea fluxului maxim, vom putea determina cuplajul maxim, ca fiind format din muchiile pe care există fluxuri nenule.

10.3.3. Un exemplu

Să presupunem că avem o listă de n elevi şi n licee, precum şi o listă de preferințe de forma (e, l) cu semnificația: *elevul e doreşte să studieze la liceul l.*

Fiecare elev poate să dorească, în egală măsură, să studieze la mai multe licee. Dorim să determinăm, dacă este posibil, o corespondență prin care fiecărui elev să îi fie asociat un liceu unic, astfel încât fiecare elev să fie repartizat într-un liceu în care doreşte să studieze.

Problema se modelează printr-un graf bipartit în care una dintre cele două submulțimi disjuncte reprezintă elevii, iar cealaltă reprezintă liceele. Între un elev şi un liceu va exista un arc doar dacă elevul doreşte să studieze la liceul respectiv.

Va trebui să selectăm un număr maxim de arce ale grafului bipartit, astfel încât fiecărui elev să îi fie asociat un liceu unic şi invers. În cazul în care numărul maxim de perechi valide este n, am obținut o soluție.

Aşa cum am arătat anterior, pentru rezolvare introducem o sursă virtuală s şi o destinație virtuală t. De asemenea, vom introduce câte un arc de la sursă la fiecare elev şi câte un arc de la fiecare liceu la destinație.

Toate arcele vor avea capacitatea 1 pentru a ne asigura că fiecărui elev îi este asociat un singur liceu şi invers. Aşadar, am redus această problemă la determinarea unui flux maxim într-o rețea de transport.

10.3.4. Implementarea şi analiza complexității

Algoritmul de transformare a grafului bipartit în rețea de transport nu ridică nici o problemă deosebită. Practic vor fi adăugate grafului două noduri şi un număr de arce egal cu numărul nodurilor din graful bipartit inițial. În continuare, se va aplica un algoritm de determinare a fluxului maxim şi se vor alege arcele din graful bipartit pentru care fluxul prin arcul corespunzător din rețeaua de transport este nenul.

În funcție de modul în care este reprezentat graful bipartit în memorie, operația de transformare a acestuia va avea ordinul de complexitate $O(N)$ sau $O(M)$. După efectuarea transformării, se va aplica algoritmul de determinare a fluxului maxim care are complexitatea cel puțin de ordinul $O(M + N) \cdot M \cdot N)$.

Putem trage imediat concluzia că ordinul de complexitate al algoritmului de determinare a cuplajului bipartit maxim este egal cu ordinul de complexitate al algoritmului ales pentru determinarea fluxului maxim în rețeaua de transport, obținută după transformarea grafului bipartit.

10.4. Rezumat

În cadrul acestui capitol am introdus noțiunea de cuplaj în general şi cuplaj bipartit în particular. De asemenea, am descris modul în care poate fi transformată problema determinării cuplajului maxim în grafuri bipartite într-o problemă de determinare a unui flux maxim într-o rețea de transport. După prezentarea modalității de transformare am descris pe scurt algoritmul şi i-am analizat complexitatea.

10.5. Implementări sugerate

Pentru a vă familiariza cu modul în care trebuie implementate rezolvările problemelor ale căror soluții necesită cunoştințe referitoare la cuplaje în grafuri bipartite, vă sugerăm să încercați să realizați implementări pentru:

1. determinarea cuplajului maxim într-un graf bipartit, folosind algoritmul *Ford-Fulkerson* pentru a determina fluxul în rețeaua de transport obținută;
2. determinarea cuplajului maxim într-un graf bipartit, folosind algoritmul *Edmonds-Karp* pentru a determina fluxul în rețeaua de transport obținută.

De asemenea, vă sugerăm să realizați aceste implementări folosind diferite modalități de reprezentare a grafurilor.

10.6. Probleme propuse

În continuare vom prezenta enunțurile câtorva probleme pe care vi le propunem spre rezolvare. Rezolvarea acestor probleme necesită doar informațiile prezentate în cadrul acestui capitol şi în cadrul capitolului dedicat fluxurilor în rețelele de transport.

10.6.1. Raze laser

Descrierea problemei

Pe partea stângă a unui culoar se află N dispozitive de emisie LASER. Pe cealaltă parte se află N celule care pot detecta astfel de raze. Pentru fiecare dintre cele N dispozitive de emisie se cunosc celulele la care pot ajunge razele emise. Va trebui să alegeți pentru fiecare dispozitiv una dintre celulele la care pot fi trimise raze astfel încât fiecare dispozitiv să emită o singură rază şi fiecare celulă să recepţioneze exact o rază.

Date de intrare

Prima linie a fişierului de intrare **`LASER.IN`** conţine numărul *N* al dispozitivelor de emisie şi al celulelor de recepţie. Fiecare dintre următoarele linii corespunde unui dispozitiv de emisie. Primul număr de pe o astfel de linie reprezintă numărul *K* al celulelor către care pot fi trimise raze, iar următoarele numere reprezintă numerele de ordine ale acestor celule. Numerele de pe o linie vor fi separate printr-un spaţiu. Prima dintre cele *N* linii corespunde dispozitivului identificat prin 1, cea de-a doua corespunde dispozitivului identificat prin 2 etc.

Date de ieşire

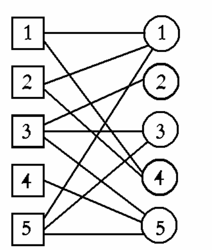
Fişierul de ieşire **`LASER.OUT`** va conţine *N* linii; cea de-a *i*-a linie va conţine numărul de ordine al celei către care dispozitivul de emisie identificat prin *i* emite o rază.

Restricţii şi precizări

- $1 \leq N \leq 100$;
- dispozitivele de emisie sunt identificate prin numere întregi cuprinse între 1 şi *N*;
- celulele sunt identificate prin numere întregi cuprinse între 1 şi *N*;
- va exista întotdeauna cel puţin o soluţie;
- dacă există mai multe soluţii, trebuie determinată doar una dintre ele.

Exemplu

LASER.IN	**LASER.OUT**
`5`	`4`
`2 1 4`	`1`
`2 1 4`	`2`
`3 2 3 5`	`5`
`1 5`	`3`
`3 1 3 5`	



Timp de execuţie: 1 secundă/test

10.6.2. Culori

Descrierea problemei

Pe o masă se află *N* pătreţele şi *M* cerculeţe, pe fiecare dintre acestea fiind desenate mai multe benzi colorate. Un cerculeţ poate fi aşezat peste un pătrăţel dacă există cel puţin o culoare care apare pe ambele.

Pentru fiecare pătrăţel şi pentru fiecare cerculeţ sunt cunoscute codurile culorilor benzilor. Va trebui să determinaţi o modalitate de amplasare a cerculeţelor pe pătrăţele astfel încât să fie create cât mai multe perechi de acest tip.

Date de intrare

Prima linie a fişierului de intrare **`CULORI.IN`** conţine numărul N al pătrăţelelor. Pe fiecare dintre următoarele N linii sunt descrise benzile corespunzătoare unui pătrăţel. Primul număr de pe o astfel de linie este numărul b al benzilor, iar următoarele b numere reprezintă codurile celor b culori. Următoarea linie conţine numărul M al cerculeţelor, iar pe următoarele M linii sunt descrise benzile corespunzătoare unui cerculeţ. Din nou, primul număr de pe o astfel de linie este numărul b al benzilor, iar următoarele b numere reprezintă codurile celor b culori. Numerele de pe o linie vor fi separate prin spaţii. Pătrăţelele şi cerculeţele vor fi descrise în ordinea dată de numărul lor de ordine.

Date de ieşire

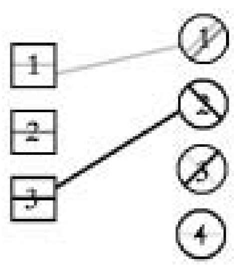
Prima linie a fişierelor de ieşire **`CULORI.OUT`** va conţine numărul k al perechilor formate. Următoarele k linii vor conţine câte două numere x şi y, separate printr-un spaţiu, reprezentând numerele de ordine ale unui pătrăţel, respectiv cerc, care formează o pereche.

Restricţii şi precizări

- $1 \leq N, M \leq 100$;
- pătrăţelele sunt identificate prin numere întregi cuprinse între 1 şi N;
- celulele sunt identificate prin numere întregi cuprinse între 1 şi M;
- numărul benzilor de pe cerculeţe şi pătrăţele este cuprins între 1 şi 10;
- un pătrăţel sau un cerc poate face parte din cel mult o pereche;
- dacă există mai multe soluţii, trebuie determinată doar una dintre ele.

Exemplu

CULORI.IN	**CULORI.OUT**
`3`	`2`
`1 1`	`1 1`
`1 2`	`3 2`
`1 3`	
`4`	
`2 1 2`	
`1 3`	
`2 3 4`	
`1 4`	



Timp de execuţie: 1 secundă/test

10.6.3. Jedi

Consiliul Jedi trebuie să trimită pe mai multe planete ale galaxiei câte un cavaler. Pentru fiecare dintre *Jedi* sunt cunoscute planetele în care acesta este deja celebru. Identitatea trimişilor trebuie să rămână secretă, motiv pentru care un cavaler nu poate fi trimis pe o planetă în care este celebru. Va trebui să determinaţi care cavaler va fi trimis pe fiecare dintre planete, astfel încât să fie trimişi în galaxie cât mai mulţi *Jedi*.

Date de intrare

Prima linie a fişierului de intrare **`JEDI.IN`** conţine numărul N al planetelor şi numărul M al cavalerilor. Pe fiecare dintre următoarele M linii este descrisă "celebritatea" unui cavaler. Primul număr de pe o astfel de linie este numărul p al planetelor pe care cavalerul este celebru, iar următoarele p numere reprezintă numerele de ordine ale celor p planete. Aceste p numere vor fi distincte, iar numerele de pe o linie vor fi separate prin spaţii.

Date de ieşire

Fişierul de ieşire **`JEDI.OUT`** va conţine N linii, pe fiecare dintre acestea aflându-se un singur număr. În cazul în care pe planeta corespunzătoare unei linii nu poate fi trimis un cavaler, numărul va fi 0, în caz contrar, linia va conţine numărul de ordine al cavalerului *Jedi* trimis pe planeta respectivă.

Restricţii şi precizări

- $1 \leq M, N \leq 100$;
- planetele sunt identificate prin numere întregi cuprinse între 1 şi N;
- cavalerii sunt identificaţi prin numere întregi cuprinse între 1 şi M;
- pe o planetă poate fi trimis cel mult un cavaler;
- dacă există mai multe soluţii, trebuie determinată doar una dintre ele.

Exemplu

JEDI.IN	**JEDI.OUT**
3 4	1
2 2 3	4
2 2 3	0
2 2 3	
0	

Timp de execuţie: 1 secundă/test

10.7. Soluțiile problemelor

Vom prezenta acum soluțiile problemelor propuse în cadrul secțiunii precedente. Pentru fiecare dintre acestea va fi descrisă metoda de rezolvare şi va fi analizată complexitatea algoritmului prezentat.

10.7.1. Raze laser

Vom construi un graf bipartit în care N noduri vor reprezenta dispozitivele de emisie LASER, iar alte N noduri vor reprezenta celulele. Va exista o muchie de la un nod corespunzător unui dispozitiv la o anumită celulă, dacă raza care pleacă de la dispozitiv poate ajunge la celula respectivă.

După construirea grafului bipartit vom determina un cuplaj maxim în acest graf şi pe baza sa vom stabili celula aleasă pentru fiecare dispozitiv.

În final, vom scrie rezultatele în fişierul de ieşire.

Enunțul ne asigură că va exista întotdeauna cel puțin o soluție, aşadar fluxul maxim determinat pentru stabilirea cuplajului va fi întotdeauna N.

Analiza complexității

Operația de citire a datelor de intrare are ordinul de complexitate $\mathbf{O(N^2)}$ deoarece, în cazul cel mai defavorabil, raza care pleacă de la un dispozitiv va putea fi detectată de oricare dintre celule. În momentul citirii poate fi construit şi graful bipartit, neconsumându-se timp suplimentar pentru această operație.

Dacă utilizăm algoritmul *Ford-Fulkerson* pentru determinarea fluxului maxim, acesta va avea ordinul de complexitate $O(N \cdot (N^2 + 2 \cdot N + N)) = \mathbf{O(N^3)}$, deoarece fluxul maxim este N şi rețeaua de transport conține cel mult $N^2 + 2 \cdot N$ arce.

Pentru afişarea soluției va trebui să parcurgem cele cel mult N^2 muchii ale grafului bipartit, deci această operație are ordinul de complexitate $\mathbf{O(N^2)}$.

În concluzie, ordinul de complexitate al algoritmului de rezolvare a acestei probleme este $O(N^2) + O(N^3) + O(N^2) = \mathbf{O(N^3)}$.

10.7.2. Culori

Vom construi un graf bipartit în care N noduri vor reprezenta pătrățelele, iar alte M noduri vor reprezenta cerculețele. Va exista o muchie de la un nod corespunzător unui pătrățel la un nod corespunzător unui cerculeț dacă există cel puțin o culoare care apare atât pe cerculeț, cât şi pe pătrățel.

După construirea grafului bipartit vom determina un cuplaj maxim în acest graf şi pe baza sa vom stabili pătrățelul ales pentru fiecare cerculeț.

În final vom scrie rezultatele în fişierul de ieşire.

Analiza complexităţii

Operaţia de citire a datelor de intrare are ordinul de complexitate **O(M + N),** deoarece se citesc date referitoare la N pătrăţele şi M cerculeţe. Considerăm că citirea datelor referitoare la un pătrăţel sau un cerculeţ se realizează în timp constant, deoarece pe acestea sunt desenate cel mult zece benzi.

După citirea datelor va trebui să creăm graful bipartit. Acesta va conţine cel mult $M \cdot N$ muchii. Verificarea existenţei unei muchii între un nod corespunzător unui pătrăţel la un nod corespunzător unui cerculeţ se realizează în timp constant, deoarece va trebui să efectuăm cel mult o sută de comparaţii. Aşadar, operaţia de creare a grafului bipartit va avea ordinul de complexitate **O(M · N)**.

După determinarea grafului bipartit va trebui să determinăm cuplajul cu ajutorul unui flux. Valoarea maximă a fluxului va fi cel mult egală cu minimul valorilor M şi N. Reţeaua de transport va conţine cel mult $M \cdot N + M + N$ arce deoarece sunt introduse N arce care pleacă de la sursă şi M arce care ajung la destinaţie. Ca urmare, ordinul de complexitate al operaţiei de determinare a cuplajului va fi $O(\min(M, N) \cdot (M \cdot N + M + N + M + N))$ = **O(min(M, N) · (M · N))**. În cazul în care avem mai multe pătrăţele decât cercuri, ordinul de complexitate va fi $O(M^2 \cdot N)$, iar în caz contrar va fi $O(M \cdot N^2)$.

Pentru afişarea soluţiei va trebui să parcurgem cele cel mult $M \cdot N$ muchii ale grafului bipartit, deci această operaţie are ordinul de complexitate **O(M · N)**.

În concluzie, ordinul de complexitate al algoritmului de rezolvare a acestei probleme este $O(M + N) + O(M \cdot N) + O(\min(M, N) \cdot (M \cdot N)) + O(M \cdot N)$ = **O(min(M, N) · (M · N))**.

10.7.3. Jedi

Vom construi un graf bipartit în care N noduri vor reprezenta planetele, iar alte M noduri vor reprezenta cavalerii Jedi. Va exista o muchie de la un nod corespunzător unei planete la un nod corespunzător unui cavaler, dacă pe planeta respectivă cavalerul nu este deja celebru. Pentru simplitate, putem crea iniţial un graf bipartit complet (există muchii între oricare două noduri care fac parte din submulţimi diferite) şi putem elimina muchii pe parcursul citirii.

După construirea grafului bipartit vom determina un cuplaj maxim în acest graf şi pe baza sa vom stabili pătrăţelul (cavalerul) ales pentru fiecare cerculeţ (planetă).

În final vom scrie rezultatele în fişierul de ieşire.

Analiza complexităţii

Operaţia de citire a datelor de intrare are ordinul de complexitate **O(M · N)** deoarece, în cazul cel mai defavorabil, cavalerii pot să fie celebrii pe toate planetele. În momentul citirii poate fi construit şi graful bipartit, neconsumându-se timp suplimentar pentru această operaţie.

După determinarea grafului bipartit va trebui să determinăm cuplajul cu ajutorul unui flux. Valoarea maximă a fluxului va fi cel mult egală cu minimul valorilor M şi N. Rețeaua de transport va conține cel mult $M \cdot N + M + N$ arce, deoarece sunt introduse N arce care pleacă de la sursă şi M arce care ajung la destinație. Ca urmare, ordinul de complexitate al operației de determinare a cuplajului va fi $O(\min(M, N) \cdot (M \cdot N + M + N + M + N))$ = **O(min(M, N) · (M · N))**.

Pentru afişarea soluției va trebui să parcurgem cele cel mult $M \cdot N$ muchii ale grafului bipartit, deci această operație are ordinul de complexitate **O(M · N)**.

În concluzie, ordinul de complexitate al algoritmului de rezolvare a acestei probleme este $O(M + N) + O(M \cdot N) + O(\min(M, N) \cdot (M \cdot N)) + O(M \cdot N)$ = **O(min(M, N) · (M · N))**.