Grafuri bipartite

Lecție de probă, informatică clasa a XI-a

Mihai Bărbulescu
b12mihai@gmail.com
Facultatea de Automatică și Calculatoare, UPB

Colegiul Național de Informatică Tudor Vianu București

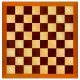
27 februarie 2013

Probleme interesante pentru mintea voastră

- Fie o tablă de șah 8×8, căreia îi ștergem pătratul din stânga sus și din dreapta jos. Demonstrați că nu putem acoperi tabla cu piese de domino 1×2 fără să existe nici o suprapunere între piese

Probleme interesante pentru mintea voastră

- Fie o tablă de șah 8×8, căreia îi ștergem pătratul din stânga sus și din dreapta jos. Demonstrați că nu putem acoperi tabla cu piese de domino 1×2 fără să existe nici o suprapunere între piese

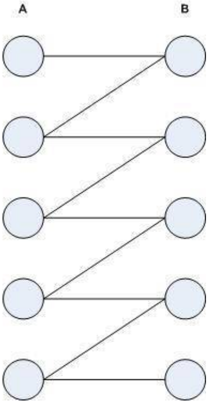


- Există băieți și fete în clasa a XI-a. Fiecărui băiat îi place de o fată. Un tovarăș comun al unei potențiale perechi încearcă să îi facă fericiți pe amândoi și să joace rolul lui Cupidon. Misiunea tovarășului e îndeplinită doar dacă băiatul o place pe fată și invers. E posibilă împerecherea tuturor? Ce proprietate trebuie să aibă această clasă a XI-a pentru a fi toată lumea îndrăgostită?

- Felicitări celor care au găsit soluție la problemele de mai sus fără a folosi grafuri bipartite! ☺

- Felicitări celor care au găsit soluție la problemele de mai sus fără a folosi grafuri bipartite! ☺
- Eu din păcate știu altfel ☹

- Felicitări celor care au găsit soluție la problemele de mai sus fără a folosi grafuri bipartite! ☺
- Eu din păcate știu altfel ☹
- Fie un graf $G = (V, E)$. G se numește **bipartit** dacă mulțimea nodurilor, V, poate fi împărțită în două mulțimi disjuncte A și B astfel încât: $V = A \cup B$ și $E \subset A \times B$ (adică orice muchie leagă un nod din A cu un nod din B).



Amintiri din copilărie ...

- **Cozi** - structuri de date FIFO (First In, First Out) - primul venit, primul servit/prelucrat
- **BFS** - **breadth first search** - parcurgere în lățime a grafurilor
  - Vizitare + inspectarea unui nod
  - Obținerea accesului la vecinii nodului curent vizitat
- **DFS** - **depth first search** - parcurgere în adâncime a grafurilor
  - Vizitare + inspectarea unui nod
  - Obținerea accesului la vecinii nodului curent vizitat
- Și ultimul, dar nu cel din urmă: reprezentarea grafurilor în memorie

- Putem folosi atât BFS cât și DFS pentru a detecta dacă un graf e bipartit sau nu
- Care e mai bună pentru problema noastră?
  - DFS - **nu** e optim, dar parcurge **tot** graful
  - BFS - **e optim**, dar **nu** parcurge tot graful

Folosim BFS pentru a detecta dacă un graf e bipartit

- În timp ce efectuez parcurgerea atribui etichete nodurilor (A sau B - cele două mulțimi din definiție)

Folosim BFS pentru a detecta dacă un graf e bipartit

- În timp ce efectuez parcurgerea atribui etichete nodurilor (A sau B - cele două mulțimi din definiție)
- Etichete atribuite conform cu paritatea nivelului (A - nivel par, B - nivel impar)

Folosim BFS pentru a detecta dacă un graf e bipartit

- În timp ce efectuez parcurgerea atribui etichete nodurilor (A sau B - cele două mulțimi din definiție)
- Etichete atribuite conform cu paritatea nivelului (A - nivel par, B - nivel impar)
- Apoi verific etichetele vecinilor nodului în care mă aflu acum

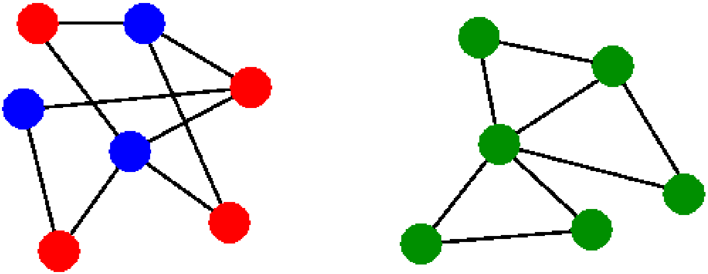
Folosim BFS pentru a detecta dacă un graf e bipartit

- În timp ce efectuez parcurgerea atribui etichete nodurilor (A sau B - cele două mulțimi din definiție)
- Etichete atribuite conform cu paritatea nivelului (A - nivel par, B - nivel impar)
- Apoi verific etichetele vecinilor nodului în care mă aflu acum
- Momentul nasol: Unul din vecini are aceeași etichetă ca cea a nodului curent $\Rightarrow$ Graful nu e bipartit. Opresc algoritmul!

Folosim BFS pentru a detecta dacă un graf e bipartit

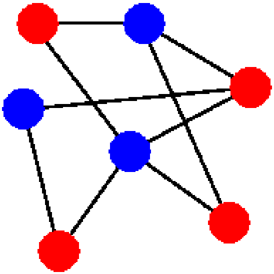
- În timp ce efectuez parcurgerea atribui etichete nodurilor (A sau B - cele două mulțimi din definiție)
- Etichete atribuite conform cu paritatea nivelului (A - nivel par, B - nivel impar)
- Apoi verific etichetele vecinilor nodului în care mă aflu acum
- Momentul nasol: Unul din vecini are aceeași etichetă ca cea a nodului curent $\Rightarrow$ Graful nu e bipartit. Opresc algoritmul!
  - Cu alte cuvinte, graful are o muchie între noduri de pe același nivel și deci nu are cum să fie bipartit
- Momentul fericit: nu s-a oprit algoritmul! ☺

De ce merge?

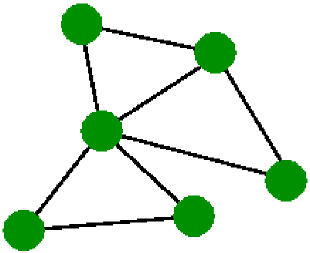


A) A Bipartite Graph B) A non–Bipartite Graph

De ce merge?



A) A Bipartite Graph



B) A non–Bipartite Graph

- Demonstrația corectitudinii algoritmului nu merită făcută acum!

$\textsc{isBipartite}(G = (V, E), s)$

for $(u \in V \setminus \{s\})$ {
    $color[u] \leftarrow WHITE$
    $dist[u] \leftarrow \infty$
    $partition[u] \leftarrow 0$
}
$color[s] \leftarrow GRAY$
$dist[s] \leftarrow 0$
$partition[s] \leftarrow 1$
$enqueue(Q, s)$

Pseudocod

```
while (Q ≠ ∅) {
    u = dequeue(Q)
    for (v ∈ vecini(u)) {
        if (partition[u] == partition[v])
            return 0
        else if (color[v] == WHITE) {
                color[v] = GRAY
                dist[v] = dist[u] + 1
                partition[v] = 3 − partition[u]
                enqueue(Q, v)
            }
    }
    color[u] = BLACK
}
return 1
```

Cum detectăm că un graf e bipartit folosind DFS?

- Simplificăm algoritmul DFS pentru a testa dacă un graf dat $G = (V, E)$ e bipartit
- Modificăm definiția inițială astfel: G este **bipartit** dacă nodurile sale pot fi colorate cu două culori $\textcolor{red}{\bullet}$ și $\textcolor{yellow}{\bullet}$ astfel încât următoarea proprietate e adevărată pentru $\forall (u, v) \in E$:

$$color[u] \neq color[v] \text{ și } color[u] \in \{\textcolor{red}{\bullet}, \textcolor{yellow}{\bullet}\} \text{ și } color[v] \in \{\textcolor{red}{\bullet}, \textcolor{yellow}{\bullet}\}$$

Cum detectăm că un graf e bipartit folosind DFS? (cont.)

- Proprietatea poate fi rescrisă în raport cu nodurile grafului:

$$P(V) : \forall u \in V \text{ și } \forall v \in \text{vecini}(u) \text{ unde}$$

$$P(u) : color[u] \neq color[v] \text{ și } color[u] \in \{{\color{red}\bullet}, {\color{yellow}\bullet}\} \text{ și } color[v] \in \{{\color{red}\bullet}, {\color{yellow}\bullet}\}$$

Cum detectăm că un graf e bipartit folosind DFS? (cont.)

- Dacă reușim să colorăm v în culoarea complementară lui $color[u]$ și proprietatea $P(V)$ e adevărată atunci $P(u)$ e adevărată
- Verificarea cu DFS, practic, pornește de la parcurgerea în adâncime și testeză dacă $P(u)$ e adevărată

Cum detectăm că un graf e bipartit folosind DFS? (cont.)

CULOARECOMPLEMENTARĂ(c)

if ($c ==$ )
    return
return 

ISBIPARTITE($G = (V, E)$)

for ($u \in V$)
    $color[u] = WHITE$
for ($u \in V$)
    if ($color[u] == WHITE$) // Verificarea $P(u)$
        if (EXPLORE(u, ) $== 0$)
            return 0
return 1

Cum detectăm că un graf e bipartit folosind DFS? (cont.)

```
EXPLORE(u, cu)
cv = CULOAReComplementară(cu)
color[u] = cu

for (v ∈ VECINI(u))
        if (color[v] == WHITE) // Verificarea P(v)
                if (EXPLORE(v, cv) == 0)
                        return 0
        if (color[u] == color[v])
                return 0

return 1 // P(u) e adevărată
```

Întrebări

- graf bipartit
- colorare noduri
- mulțimi disjuncte
- partiționare
- împerecheri