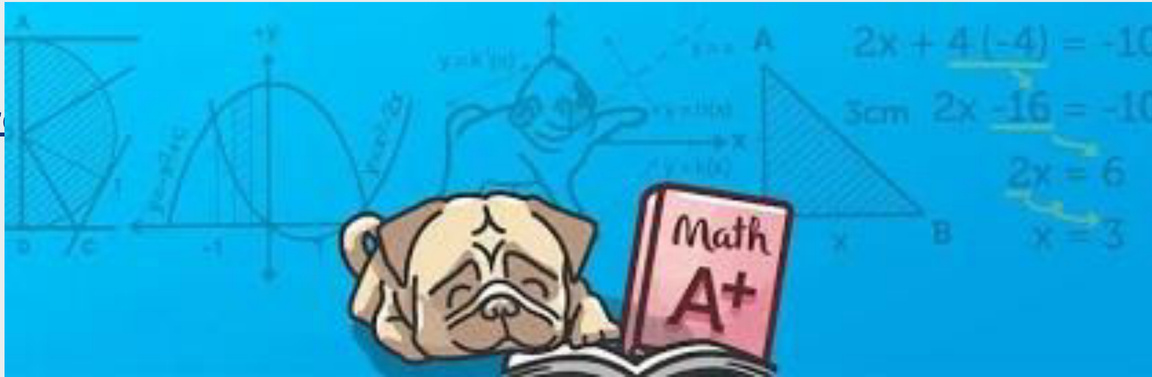
Algoritmi Avansați 2021
c-4
Vertex Cover Problem, Linear Programming

Lect. Dr. Ștefan Popescu

Email: stefan.popescu@fmi.unibuc.r

Grup Teams:



Vertex cover problem

Problema:
Fie o rețea de calculatoare în care trebuie să testăm toate conexiunile.

Pentru a testa conexiunile, trebuie să instalăm un program software pe mai multe calculatoare. Acest program poate testa toate conexiunile directe care pleacă din respectivul calculator.

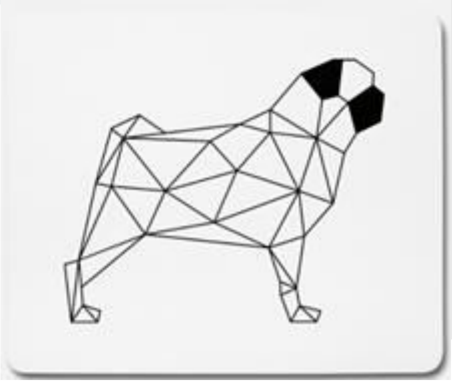
Vertex cover problem

Problema:
Fie o rețea de calculatoare în care trebuie să testăm toate conexiunile.

Pentru a testa conexiunile, trebuie să instalăm un program software pe mai multe calculatoare. Acest program poate testa toate conexiunile directe care pleacă din respectivul calculator.

Evident, putem instala acest program pentru a monitoriza întreaga rețea, dar dorim să minimizam intervenția. Deci se pune problema găsirii unei submulțimi de calculatoare de cardinal minim care să poată monitoriza întreaga rețea.

Vertex cover problem



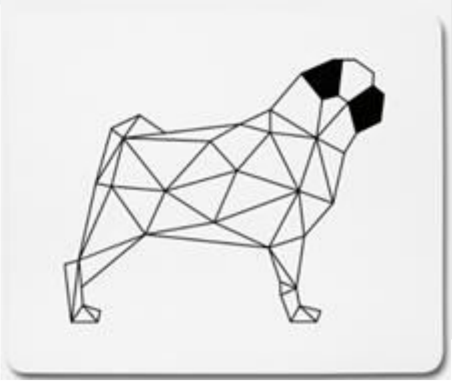
Problema formală:
Fie un graf neorientat G=(V,E).

Numim ”acoperire” o submulțime $S \subset V$ cu proprietatea ca pentru orice $(x,y) \in E$ avem

$x \in S$ sau $y \in S$ (sau $x,y \in S$)

Se pune problema găsirii unei acoperiri S de cardinal minim!

Vertex cover problem



Problema formală:
Fie un graf neorientat G=(V,E).

Numim ”acoperire” o submulțime S⊂V cu oriorietatea ca pentru orice (x,y) ∊ E avem

x∊S sau y∊S (sau x,y∊S)

Se pune problema găsirii unei acoperiri S de cardinal minim!

Această problemă este NP-hard.

Vertex cover problem

Fie următorul algoritm:

INPUT: G=(V,E)

E'=E; S=ø;

cât timp E'≠ø:

aleg (x,y)∊E';

S=S∪{x}

ștergem din E' toate muchiile incidente lui x

return S

Vertex cover problem

Fie următorul algoritm:

INPUT: G=(V,E)

E'=E; S=ø;

cât timp E'≠ø:

aleg (x,y)∈E';

S=S∪{x}

ștergem din E' toate muchiile incidente lui x

return S

Q1. Mulțimea de noduri S este o acoperire pentru graful G?
DA/NU

Vertex cover problem

Fie următorul algoritm:

INPUT: G=(V,E)

E'=E; S=ø;

cât timp E'≠ø:

aleg (x,y)∊E';

S=S∪{x}

ștergem din E' toate muchiile incidente lui x

return S

Q1 Mulțimea de noduri S este o acoperire pentru graful G?
DA!

Vertex cover problem

Fie următorul algoritm:

INPUT: G=(V,E)

E'=E; S=ø;

cât timp E'≠ø:

aleg (x,y)∊E';

S=S∪{x}

ștergem din E' toate muchiile incidente lui x

return S

Q2. Algoritmul de alături:

a) Este un algoritm care generează mereu soluția optimă
b) Este un algoritm 3-aproximativ pentru VCP
c) poate furniza si un răspuns de 100 de ori mai slab decât soluția optimă

Vertex cover problem

Fie următorul algoritm:

INPUT: G=(V,E)

E'=E; S=ø;

cât timp E'≠ø:

aleg (x,y)∊E';

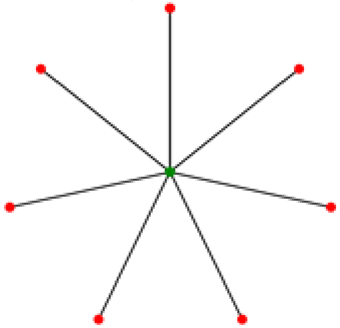
S=S∪{x}

ștergem din E' toate muchiile incidente lui x

return S

Q2. Algoritmul de alături:

poate furniza si un răspuns de 100 de ori mai slab decât soluția optimă



Vertex cover problem

Fie următorul algoritm:

INPUT: G=(V,E)

E’=E; S=ø;

cât timp E’≠ø:

aleg (x,y)∊E’;

S=S∪{x}

ștergem din E’ toate muchiile incidente lui x

return S

Q3. Cum putem modifica algoritmul alăturat astfel încât să îmbunătățim rezultatul?

Vertex cover problem

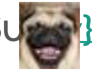
Fie următorul algoritm:

INPUT: G=(V,E)

E'=E; S=ø;

cât timp E'≠ø:

aleg (x,y)∈E';

S=SU  }

ștergem din E' toate muchiile incidente lui x și lui y

return S

Q3. Cum putem modifica algoritmul alăturat astfel încât să îmbunătățim rezultatul?

Vertex cover problem

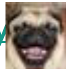
Fie următorul algoritm:

ApproxVertexCover (V,E)

E'=E; S=ø;

cât timp E'≠ø:

aleg (x,y)∊E';

S=S∪{x,y 

ștergem din E' toate muchiile incidente lui x și lui y

return S

Deși pare o abordare cel puțin ciudată, algoritmul alăturat este un algoritm 2-aproximativ pentru vertex cover problem!

Vertex cover problem

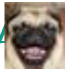
Fie următorul algoritm:

ApproxVertexCover (V,E)

E'=E; S=ø;

cât timp E'≠ø:

aleg (x,y)∊E'

S=S∪{x,y 

ștergem din E' toate muchiile incidente lui x și lui y

return S

Deși pare o abordare cel puțin ciudată, algoritmul alăturat

1) generează o acoperire validă
2) este un algoritm 2-aproximativ

Vertex cover problem

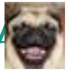
Fie următorul algoritm:

ApproxVertexCover (V,E)

E'=E; S=ø;

cât timp E'≠ø:

aleg (x,y)∈E';

S=S∪{x,y

ștergem din E' toate muchiile incidente lui x și lui y

return S

Lema 1. Fie G=(V,E) un graf neorientat și OPT cardinalul unei acoperiri de grad minim a lui G. Fie E'⊂E o mulțime de muchii nod disjuncte.

Atunci avem că OPT≥|E'|

Demonstratie:

Seria 23 & Seria 24

Vertex cover problem

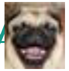
Fie următorul algoritm:

ApproxVertexCover (V,E)

E'=E; S=ø;

cât timp E'≠ø:

aleg (x,y)∈E';

S=S∪{x,y} 

ștergem din E' toate muchiile incidente lui x și lui y

return S

Teorema 2. Algoritmul alăturat este un algoritm 2 aproximativ pentru VCP.

Demonstratie:
Seria 23 & Seria 24

Complicam Problema!
Weighted Vertex Problem.

Fie un graf G=(V,E) - un graf simplu, si $f:V \to R_+$ care asociază fiecărui vârf, un cost

Trebuie să găsim o acoperire de varfuri S astfel încât să minimizăm: $\sum_{v \in S} f(v)$

Este dificil să găsim un algoritm aproximativ pt aceasta problemă prin metodele ”tradiționale”

Tb sa gasim o abordare noua!

Programare Liniara

O problemă de programare liniară arată în felul următor:

- o funcție de ”cost” cu *d* variabile $x_1, x_2, ..., x_d$
- un set de *n* constrângeri liniare peste variabilele $x_1, x_2, ..., x_d$

  Scopul este asignarea de valori pentru variabilele de tip x_i astfel încât să minimizăm (sau, după caz, să maximizăm) funcția de cost, respectand totodata toate cele *n* constrangeri

Programare Liniara

O problemă de programare liniară arată în felul următor:

Ex:

Tb minimizat $c_1x_1 + \cdots + c_dx_d$

astfel încât

$a_{1,1}x_1 + \cdots + a_{1,d}x_d \leq b_1$

$a_{2,1}x_1 + \cdots + a_{2,d}x_d \leq b_2$

...

$a_{n,1}x_1 + \cdots + a_{n,d}x_d \leq b_n$

Programare Liniara

O constrângere poate conține adunări de variabile, poate folosi inegalități de orice tip (<,>,>=,<=,=)

O constrângere nu poate fi opțională! Toate constrângerile sunt ”binding”

În constrangeri nu pot apărea elemente de forma ”x_i*x_j” sau ”x^2” - trebuie sa fie liniare!

Programare Liniara

O problemă de programare liniară arată în felul următor:

Ex:

Tb minimizat $c_1x_1 + \cdots + c_dx_d$

astfel încât

$a_{1,1}x_1 + \cdots + a_{1,d}x_d \leq b1$

$a_{2,1}x_1 + \cdots + a_{2,d}x_d \leq b2$

...

$a_{n,1}x_1 + \cdots + a_{n,d}x_d \leq b1$

Astfel de sisteme pot fi rezolvate în timp polinomial prin algoritmi *simplex* (vezi cursul de Tehnici de Optimizare).

Programare Liniara

O problemă de programare liniară arată în felul următor:

Ex:

Tb minimizat $c_1x_1 + \cdots + c_dx_d$

astfel încât

$a_{1,1}x_1 + \cdots + a_{1,d}x_d \leq b1$

$a_{2,1}x_1 + \cdots + a_{2,d}x_d \leq b2$

…

$a_{n,1}x_1 + \cdots + a_{n,d}x_d \leq b1$

Astfel de sisteme pot fi rezolvate în timp polinomial prin algoritmi *simplex* (vezi cursul de Tehnici de Optimizare).

OBSERVAȚIE:

Algoritmii simplex rezolvă inegalitatea pentru **x_i - numere reale**!

Revenim la WVCP (slide 16)

Putem formula această problemă ca o problemă de programare liniară:

Seria 23 & Seria 24

Astfel de sisteme pot fi rezolvate în timp polinomial prin algoritmi *simplex* (vezi cursul de Tehnici de Optimizare).

OBSERVAȚIE:

Algoritmii simplex rezolvă inegalitatea pentru **x_i - numere reale**!

Further reading:

[Suport de curs saptamanile 4-5 (engl)]