AA - TEMA 2
- REDUCERI POLINOMIALE-

Responsabili:
Horia Nedelcu, Radu Nichita

Deadline soft: **21.01.2022**
Deadline hard: **21.01.2022**

CUPRINS

1 Introducere 3
1.1 Obiectiv . . . 3
1.2 Definiții . . . 3
1.3 Exemplu de reducere la problema SAT . . . 3
1.3.1 Problema acoperirii cu Varfuri (Vertex Cover) . . . 3
1.3.2 Reducere Vertex Cover $\leqslant_p$ CNF-SAT . . . 3
1.3.3 Exemplu . . . 4
2 Rețele sociale 5
2.1 Enunț . . . 5
2.2 Cererea către Oracol . . . 5
2.2.1 Date de intrare . . . 5
2.2.2 Date de ieșire . . . 5
2.2.3 Restricții și precizări . . . 6
2.2.4 Exemplu . . . 6
2.3 Descifrarea Oracolului . . . 6
2.3.1 Date de intrare . . . 6
2.3.2 Date de ieșire . . . 6
2.3.3 Exemplu . . . 7
3 Reclame buclucase 8
3.1 Enunț . . . 8
3.2 Cererea către Oracol . . . 8
3.2.1 Date de intrare . . . 8
3.2.2 Date de ieșire . . . 8

3.2.3 Restricții și precizări 8
3.2.4 Exemplu 9
4 Alocarea regiștrilor 10
4.1 Enunț 10
4.2 Cererea către Oracol 10
4.2.1 Date de intrare 10
4.2.2 Date de ieșire 11
4.2.3 Restricții și precizări 11
4.2.4 Exemplu 11
4.3 Descifrarea Oracolului 12
4.3.1 Date de intrare 12
4.3.2 Date de ieșire 12
4.3.3 Exemplu 12
5 Clarificări pentru folosirea Oracolului 13
6 Punctare 14
6.1 Checker 14
7 Format arhivă 14

1 INTRODUCERE

1.1 Obiectiv

- Asocierea și modelarea unor aplicații practice cu probleme NP.
- Implementarea reducerilor polinomiale la problema SAT.

1.2 Definiții

oracol = este o entitate teoretica capabilă de a rezolva o problemă. Poate fi privit ca o "cutie magica" (black box), care poate produce o soluție pentru orice instanță dată a unei probleme.
CNF = forma conjunctiv normală.
SAT = problema satisfiabilității. Este o problemă NP - Completă, conform Teoremei lui Cook.

1.3 Exemplu de reducere la problema SAT

1.3.1 Problema acoperirii cu Varfuri (Vertex Cover)

Se dă un graf neorientat G și un număr k. Există pentru graful dat o acoperire de k noduri? (O acoperire de k noduri este o submulțime de k noduri ale lui G, cu proprietatea că pentru oricare muchie (u,v), cel puțin unul dintre nodurile u și v aparțin submulțimii)

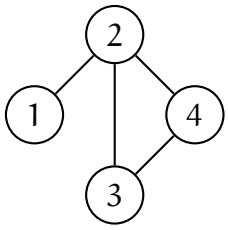
1.3.2 Reducere Vertex Cover $\leqslant_p$ CNF-SAT

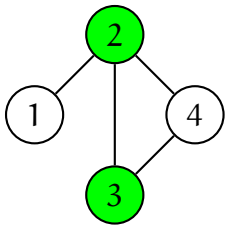
Fie n - numărul de noduri al grafului, construim o expresie booleană cu proprietatea că aceasta este satisfiabilă dacă și numai dacă G are o acoperire de dimensiune k.
Pentru $1 \leqslant i \leqslant n$ si $1 \leqslant r \leqslant k$ se introduc variabilele $y_{i,r}$ cu semnificația: $y_{i,r}$ este adevărată dacă nodul i este al r-lea din acoperire.
Se introduc următoarele clauze:

- $(y_{1,r} \vee y_{2,r} \vee ... \vee y_{n,r})$, pentru fiecare $r, 1 \leqslant r \leqslant k$ - **aceste clauze forțează că există măcar un nod pentru fiecare element al acoperirii**).
- $(\neg y_{i,r} \vee \neg y_{i,s})$ pentru $1 \leqslant i \leqslant n$, $1 \leqslant r, s \leqslant k$ și $r \neq s$ - **aceste clauze forțează că un nod se află cel mult o dată în acoperire (nu este ales pentru mai multi indecsi ai acoperii**).
- $(y_{u,1} \vee y_{u,2} \vee ... \vee y_{u,k} \vee y_{v,1} \vee y_{v,2} \vee ... \vee y_{v,k})$, pentru fiecare muchie (u,v) din G - **aceste clauze forțează ca oricare muchie să aibă cel puțin un capăt în acoperire**).

Având în vedere construcția de mai sus, expresia este satifiabilă dacă și numai dacă G conține o acoperirie de dimensiune k.

1.3.3 Exemplu





k = 2

Pentru graful dat, o posibilă acoperire este mulțimea formată din nodurile 2 și 3. Expresia booleana echivalenta a problemei este:

$E(y_{1,1}, y_{1,2}, y_{2,1}, y_{2,2}, y_{3,1}, y_{3,2}, y_{4,1}, y_{4,2}) =$

$(y_{1,1} \vee y_{2,1} \vee y_{3,1} \vee y_{4,1}) \wedge$ *primul element posibil al acoperirii*
$(y_{1,2} \vee y_{2,2} \vee y_{3,2} \vee y_{4,2}) \wedge$ *al doilea element posibil al acoperirii*
$(\neg y_{1,1} \vee \neg y_{1,2}) \wedge$ *nodul 1 poate fi considerat cel mult o data in acoperire*
$(\neg y_{2,1} \vee \neg y_{2,2}) \wedge$ *nodul 2 poate fi considerat cel mult o data in acoperire*
$(\neg y_{3,1} \vee \neg y_{3,2}) \wedge$ *nodul 3 poate fi considerat cel mult o data in acoperire*
$(\neg y_{4,1} \vee \neg y_{4,2}) \wedge$ *nodul 4 poate fi considerat cel mult o data in acoperire*
$(y_{1,1} \vee y_{1,2} \vee y_{2,1} \vee y_{2,2}) \wedge$ *muchia (1,2) are cel putin un capat in acoperire*
$(y_{2,1} \vee y_{2,2} \vee y_{3,1} \vee y_{3,2}) \wedge$ *muchia (2,3) are cel putin un capat in acoperire*
$(y_{2,1} \vee y_{2,2} \vee y_{4,1} \vee y_{4,2}) \wedge$ *muchia (2,4) are cel putin un capat in acoperire*
$(y_{3,1} \vee y_{3,2} \vee y_{4,1} \vee y_{4,2})$ *muchia (3,4) are cel putin un capat in acoperire*

Aceasta este satisfiabilă, iar o soluție validă este: $y_{2,1}$ = true, $y_{3,2}$ = true, iar restul variabilelor false.

2 REȚELE SOCIALE

2.1 Enunț

Gigel petrece mult timp în spațiul online și găsește o nouă rețea sociala, numită Metabook. Platforma se află la început, așa că oferă beneficii utilizatorilor care găsesc grupuri cât mai bune pentru a expune reclame. Gigel, pasionat de psihologie și algoritmică, știe că cel mai bun grup de influențat este cel care are cât mai multe persoane cu interese comune și care se cunosc între ei.

Scopul vostru e să îl ajutați pe Gigel să facă un ban pentru sărbătorile de iarnă și eventual să prindă un internship de ads provider la Metabook.

2.2 Cererea către Oracol

Fiind dată reteaua socială, determinați dacă există un grup de persoane bun pentru expus reclame. Un grup este bun, dacă toate persoanele grupului se cunosc între ele și au interese comune. Va trebui să reduceți instanța primită la o instanță pentru problema SAT. Vă încurajăm să porniți de la acestă reducere.

2.2.1 Date de intrare

Toate datele se vor citi de la **STDIN**.

```
N M K
u1 v1
u2 v2
...
um vm
```

Pe prima linie se află 3 numere întregi **N**, **M** și **K**, reprezentând numărul membrilor rețelei de socializare, numărul relațiilor de prietenie dintre acestia și respectiv dimensiunea grupului de persoane căutat.

Pe următoarele **M** linii se află 2 numere întregi **u** și **v**, semnificând faptul că persoanele **u** și **v** se cunosc si au interese comune.

2.2.2 Date de ieșire

Pe prima linie a fișierului **"sat.cnf"** va conține stringul *"p cnf"* urmat de numerele **V** și **F**.

Următoarele **F** linii conțin variabilele pentru fiecare clauză din formula finală. Variabilele sunt despărțite de un spațiu, iar la final veți adăuga un 0, ce va reprezenta terminarea clauzei.

2.2.3 Restricții și precizări

- Variabilele nu pot conține 0, deci va trebui să începeți numirea variabilelor de la 1.

2.2.4 Exemplu

Exemplu 1	
retele.in	retele_sat.cnf
4 4 1 1 2 2 3 2 4 3 4	p cnf <nr_variables> <nr_clauses> $< var\_1 > < var\_2 >... < var\_n > 0$ ...

2.3 Descifrarea Oracolului

2.3.1 Date de intrare

Pe prima linie a fișierului **"sat.sol"** se va afla răspunsul Oracolului:

- **True**, dacă formula voastră are soluție.
- **False**, dacă formula voastră nu are soluție.

În cazul în care formula are soluție, pe a doua linie va fi numărul **V**, reprezentat de numărul variabilelor formulei.

Pe ultima linie vor fi **V** numere reprezentând numele variabilelor din formula voastră. Aceste numere vor fi pozitive sau negative, cele pozitive reprezentând atribuirea valorii **True** acestui literal, iar cele negative reprezentând atribuirea valorii **False** literalului respectiv.

2.3.2 Date de ieșire

Toate datele se vor afișa la **STDOUT**. În cazul în care răspunsul oracolului este *False*, se va afișa "False". Altfel, pe prima linie va apărea textul *True* vor fi **K** numere cu valori între 1 și **N**, reprezentând persoanele care formează un grup de dimensiune **K**.

2.3.3 Exemplu

Exemplu		
retele_sat.sol	retele.out	Explicație
True 5 -1 2 3 -4 5	True 2 3 5	Persoanele 2, 3 și 5 formează un grup extins.

Observație: Interpretarea soluției oracolului depinde de rezolvarea voastră!

3 RECLAME BUCLUCASE

3.1 Enunț

După campania reușită de expunere de reclame, echipa de dezvoltare de la Metabook, dorește să găsească grupuri și mai bune pentru publicitate. Gigel nu mai are timp să se preocupe de asta, fiind ocupat cu rezolvarea temelor cu Biju de la IOCLA, de aceea vă cere ajutorul. Fiind dată o rețea socială, găsiți grupul esențial de dimensiune minimă pentru rețea. Un grup de persoane se numește esențial dacă prin eliminarea acestora, toți membrii rămași în rețea devin izolați și nu mai au niciun prieten cu interese comune.

3.2 Cererea către Oracol

Pentru a găsi rezultatul acestei problemei, va fi nevoie să formulați cereri multiple spre Oracol, deoarece el poate răspunde doar unor probleme de decizie, nu de optim. Scheletul pentru acest task este adaptat să vă permită să efectuați oricâte cereri aveți nevoie pentru a rezolva problema. Va trebui să reduceți instanța primită la o instanță pentru problema SAT.

3.2.1 Date de intrare

Toate datele se vor citi de la **STDIN**.

```
N M
u1 v1
u2 v2
...
um vm
```

Pe prima linie se află 2 numere întregi **N**, **M**, reprezentând numărul persoanele din rețea și numărul relațiilor de prietenie dintre ele.

Pe următoarele **M** linii se află 2 numere întregi **u** și **v**, semnificând faptul că persoanele **u** și **v** se cunosc si au interese comune.

3.2.2 Date de ieșire

Toate datele se vor afișa la **STDOUT**. Output-ul va conține o singură linie cu numerele persoanelor ce alcătuiesc grupul maxim ce poate fi influențat.

3.2.3 Restricții și precizări

- Variabilele nu pot conține 0, deci va trebui să începeți numirea variabilelor de la 1.

- Sunt posibile mai multe soluții, nu contează care dintre acestea este afișsată.

3.2.4 Exemplu

Exemplu 1		
reclame.in	reclame.out	Explicație
4 4 1 2 2 3 2 4 3 4	2 3	Persoanele 2 și 3 formează grupul de mărime minimă. Fără ele, persoanele 1 și 4 nu ar mai avea prieteni cu interese comune.

Observație: Interpretarea soluției oracolului depinde de rezolvarea voastră!

4 ALOCAREA REGIȘTRILOR

4.1 Enunț

În cadrul cursului de IOCLA am văzut că procesorul nu lucrează direct cu memoria RAM, ci dispune de regiștrii pentru a face operații.

Programarea în acest mod este foarte anevoioasă, de aceea, au apărut limbaje de nivel mai înalt, care sa permită programatorilor să gestioneze și să aloce ușor resursele. De exemplu în limbaje precum C, Python, Java etc. este foarte ușor sa declarăm și să lucrăm cu un număr nelimitat de variabile precum vectori, structuri, tipuri de date primitive, etc., chiar dacă numărul de regiștrii este limitat de capacitatea fizică a sistemului de calcul utilizat.

De exemplu, pentru următoarele operații de atribuire dint-un limbaj utilizat:

$$
\begin{aligned}
a &:= b + c \\
d &:= a + 1 \\
e &:= d * 2
\end{aligned}
$$

ar trebui să fie alocați cel puțin trei regiștrii:

$$
\begin{aligned}
a &:= b + c & r_1 &:= r_2 + r_3 \\
d &:= a + 1 & r_1 &:= r_1 + 1 \\
e &:= d * 2 & r_1 &:= r_1 * 2
\end{aligned}
$$

Deși această problemă este de o importanță deosebită în dezvoltatea compilatoarelor moderne, nu se cunoaște o soluție optimă și fezabilă până acum (Problema P vs NP).

Cu toate acestea putem apela la un oracol care să ne rezolve (parțial) altă problemă la fel de grea și să creăm o reducere de la problema alocării regiștrilor la cea a unui oracol ce rezolva problema SAT.

4.2 Cererea către Oracol

Fiind date un număr de variabile, relațiile dintre acestea și un număr fix de regiștrii, determinați dacă există o alocare a regiștrilor pentru variabilele date. Pentru aceasta va trebui să reduceți instanța primită la o instanță pentru problema SAT.

Există mai multe modalități în care puteți să faceți acest lucru, iar implementarea checker-ului vă oferă libertatea de a vă alege voi ce reducere doriți să folosiți.

4.2.1 Date de intrare

Toate datele se vor citi de la **STDIN**.

```
N M K
u1 v1
u2 v2
...
um vm
```

Pe prima linie se află 3 numere întregi **N**, **M** și **K**, reprezentând numărul de variabile, numărul relațiilor dintre ele și numărul de registre disponibile. O relație între 2 variabile reprezintă faptul că acestea sunt independente și nu pot fi puse în același registru.

Pe următoarele **M** linii se află 2 numere întregi **u** și **v**, semnificând faptul că exista relație între variabilele **u** și **v**.

4.2.2 Date de ieșire

Prima linie a fișierului **"sat.cnf"** va conține stringul *"p cnf"* urmat de numerele **V** și **F**.

Următoarele **F** linii conțin variabilele pentru fiecare clauză din formula finală. Variabilele sunt despărțite de un spațiu, iar la final veți adăuga un 0, ce va reprezenta terminarea clauzei.

4.2.3 Restricții și precizări

- Variabilele nu pot conține 0, deci va trebui să începeți numirea variabilelor de la 1.
- Două variabile care nu au legătură pot avea ori același registru, ori regiștrii diferiți.
- Un registru poate să țină mai multe variabile, atât timp cât se respectă condițiile anterioare.
- Se pot repartiza și mai puțin de K registre, este esențial să nu fie mai mult de K registre.

4.2.4 Exemplu

Exemplu 1	
registre.in	registre_sat.cnf
4 4 1 1 2 2 3 2 4 3 4	p cnf <nr_variables> <nr_clauses> $< var\_1 > < var\_2 >... < var\_n > 0$ ...

4.3 Descifrarea Oracolului

Răspunsul pe care îl veți primi de la Oracol va conține doar alegerea variabilelor. Va trebui să faceți și o interpretare a acestui răspuns pe baza reducerii pe care ați ales-o.

4.3.1 Date de intrare

Pe prima linie a fișierului **"sat.sol"** se va afla răspunsul Oracolului:

- **True**, dacă formula voastră are soluție.
- **False**, dacă formula voastră nu are soluție.

În cazul în care formula are soluție, pe a doua linie va fi numărul **V**, reprezentat de numărul variabilelor formulei.

Pe ultima linie vor fi **V** numere reprezentând numele variabilelor din formula voastră. Aceste numere vor fi pozitive sau negative, cele pozitive reprezentând atribuirea valorii **True** acestui literal, iar cele negative reprezentând atribuirea valorii **False** literalului respectiv.

4.3.2 Date de ieșire

Toate datele se vor afișa la **STDOUT**. În cazul în care răspunsul oracolului este **False**, se va afișa o singură linie cu textul *False*.

Altfel, pe prima linie va apărea textul *True*, iar pe a doua linie vor fi **N** numere cu valori între 1 și **K**, reprezentând numărul registrului asignat fiecărei variabile.

4.3.3 Exemplu

Exemplu 1		
registre_sat.sol	registre.out	Explicație
True 5 -1 2 3 -4 5	True 1 2 3 2	Prima variabilă va fi repartizată registrului numărul 1. A doua variabilă va fi repartizată registrului numărul 2. A treia variabilă va fi repartizată registrului numărul 3. A patra variabilă va fi repartizată registrului numărul 2.

Observație: Interpretarea soluției oracolului depinde de rezolvarea voastră!

5 CLARIFICĂRI PENTRU FOLOSIREA ORACOLULUI

Să presupunem că în urma reducerii pe care ați făcut-o pentru o problemă aveți o formulă de tipul:

$$(x \vee y \vee \neg z \vee \neg w) \wedge (\neg x \vee z) \wedge (\neg y \vee w) \wedge (x \vee y \vee w) \wedge (\neg y \wedge \neg z)$$

Pentru a codifica această formulă într-un fișier de tipul DIMACS care să poată fi folosit de Oracol (SAT solver), va trebui să oferim fiecărei variabile câte un număr pozitiv de la 1 până la 4 (numărul total de variabile). Să spunem că alegem $x \to 1$, $y \to 2$, $z \to 3$, $w \to 4$. Atunci fișierul .cnf corespunzător acestei formule va fi

```
p cnf 4 5
1 2 -3 -4 0
-1 3 0
-2 4 0
1 2 4 0
-2 -3 0
```

Prima linie conține antetul obligatoriu *p cnf* urmat de numărul de variabile si numărul de clauze. Următoarele linii vor conține codificările pentru fiecare clauză. Acestea sunt formate dintr-un șir de numere semnificând variabilele din clauză, terminându-se cu cifra 0. Fiecare din numerele pentru clauze pot fi negative sau pozitive, un număr negativ reprezentând că variabila corespunzătoare numărului este negată în clauză, iar unul pozitiv că variabila nu este negată.

Fișierul pe care oracolul o să îl creeze va avea forma

```
True
4
1 -2 3 4
```

Acest fișier arată că formula poate fi satisfăcută și după oferă o alegere a variabilelor din formulă. Pentru cazul nostru, se observa ca alegând y ca *False*, și restul ca *True*, formula este satisfăcută.

6 PUNCTARE

- Punctajul temei este de 100 puncte, distribuit astfel:
  - **Rețele sociale**: 20p
  - **Reclame buclucașe**: 40p
  - **Alocarea registrelor**: 30p
  - Comentarii și README: 10p
- Punctajul pe README și comentarii este condiționat de obținerea a unui punctaj strict pozitiv pe cel puțin un test.
- Vor exista mai multe teste pentru fiecare problemă în parte. Punctele pe teste sunt independente, punctajul pe un anumit test nefiind condiționat de alte teste.
- În fișierul README va trebui să descrieți soluția pe care ați ales-o pentru fiecare problemă, să precizați complexitatea pentru fiecare și alte lucruri pe care le considerați utile de menționat.
- Pentru a primi punctajul complet pe o problemă este nevoie să implementați atât cererea spre oracol, cât și descifrarea răspunsului, deoarece checker-ul verifică doar validitatea răspunsului final, permițându-vă astfel să aveți libertate deplină în alegerea reducerilor pe care doriți să le folosiți.

6.1 Checker

- Atât checkerul cât și scheletul sunt disponibile TODO.
- Arhiva se va trimite **OBLIGATORIU** pe vmchecker, unde tema se va testa folosind un set de teste private.
- Pentru citirea in Java se recomanda folosirea **BufferedReader**.

7 FORMAT ARHIVĂ

- Temele vor fi testate automat pe vmchecker. Acesta suportă temele rezolvate în C/C++ sau Java.
- Arhiva cu rezolvarea temei trebuie să fie **.zip**, având un nume de forma **Grupa_NumePrenume_Tema2.zip** (ex: 399CX_BijuGigel_Tema2.zip).

- **ATENȚIE!** Tema va fi compilată și testată **DOAR pe Linux**.

- **ATENȚIE!** Pentru cei ce folosesc C/C++ **NU** este permisă compilarea cu opțiuni de optimizare a codului (O1, O2, etc.).

- **ATENȚIE!** Orice nerespectare a restricțiilor duce la pierderea punctajului (după regulile de mai sus).