Proiectarea Algoritmilor
Tema 2
Dată publicare: 05.04.2015
Deadline soft: 23.04.2015
Dată Actualizare: 13.04.2015

1. Problema 1 - **K-Colorare**

Prima problemă este o reformulare a unei probleme clasice, cu importante aplicații in practică: problema colorării grafurilor [0], [1].

1.1 Enunț

Fie un graf cu **N** noduri si **M** muchii. Considerând că avem la dispoziție **C** culori, dorim să asociem fiecărui nod o culoare astfel incât pentru oricare două noduri vecine să atribuim o culoare diferită. În plus, atașăm un cost pentru orice combinație între culorile a două noduri adiacente. Se cere determinarea unei colorări valide având costul total minim, respectiv afișarea unui mesaj de eroare in cazul care colorarea nu este posibilă.

1.2 Aplicații practice

Multe probleme intâlnite in practică se pot reduce la o instanță a problemei colorării grafurilor.
Un exemplu generic este problema planificării în timp a unor task-uri. Adesea, alegerea algoritmului trebuie să țină cont de o serie de restricții: o penalizare pentru utilizarea simultană a unei resurse (de exemplu, numărul de task-uri care pot fi planificate in același timp), costuri pentru o anumită ordine aleasă între task-uri, etc.
De exemplu, o optimizare importantă realizată de compilator este alocarea valorilor utilizate des pe regiștrii procesorului. [2] Această problemă poate fi modelată ca o instanță a problemei colorării unui graf: considerăm ca fiecare nod reprezintă o valoare, respectiv fiecare registru o culoare. Avem muchie intre doua valori daca nu putem sa le salvăm simultan in același registru.
Deși pe cazul general problema este NP-completă, solutiile intalnite in practică exploatează natura specifică a datelor problemei (e.g.: [3]) pentru a deriva algoritmi polinomiali sau algoritmi care să aproximeze eficient răspunsul optim. La evaluarea temei, vom ține cont de implementările deosebite, pentru acordarea unui punctaj suplimentar.

[0] http://en.wikipedia.org/wiki/Graph_coloring
[1] http://cgm.cs.mcgill.ca/~godfried/teaching/dm-reading-assignments/Map-Graph-Coloring.pdf
[2] http://www.lighterra.com/papers/graphcoloring/
[3] http://en.wikipedia.org/wiki/Interval_graph

1.3 Format date intrare/ieșire

Soluția va citi datele de intrare din fișierul **kcol.in** și va scrie rezultatul în fișierul **kcol.out**.

1.3.1 Format intrare:

- Pe prima linie se află 3 numere separate prin spațiu **N** (numărul de noduri în graf), **M** (numărul de muchii în graf), **C** (numărul maxim de culori disponibile)
- Pe următoarele **M** linii se află perechi de numere (**x1, x2**) ce reprezintă nodurile între care există o muchie.
- Urmează o linie goală.
- Pe următoarele **C * (C - 1) / 2** linii se află triplete de numere (**C1, C2, cost**) ce reprezinta costul asociat utilizarii culorilor **C1** și **C2** în 2 noduri adiacente.

Restricții intrare:
0 < N <= 30
0 <= M <= N * (N - 1) / 2
0 < C <= 10
0 <= x1, x2 < N

1.3.2 Format ieșire:

- Pe prima linie se află un număr **MinCost** ce reprezinta costul minim obținut.
- Pe următoarele **N** linii se află perechi de numere (**Xi, Ci)** ce reprezintă indicele culorii (**Ci**) asignată nodului **Xi**.
- **Daca sunt prea puține culori disponibile, fișierul de ieșire va conține doar o singură linie, cu valoarea -1.**
- **Dacă există mai multe soluții, se va afișa una dintre ele.**

Intrare	**Ieșire**
5 5 3 0 1 0 4 1 2 2 3 3 4 0 1 1 0 2 2 1 2 3	8 0 0 1 1 2 0 3 1 4 2

Graful are 5 noduri si 5 muchii, există maxim 3 culori disponibile. Costul utilizarii culorii 0 alături de culoarea 1 este 1. Costul utilizarii culorii 0 alături de culoarea 2 este 2. Costul utilizarii culorii 1 alături de culoarea 2 este 3.	Soluția are un cost minim posibil de 8. Nodul 0 are culoarea 0. Nodul 1 are culoarea 1. Nodul 2 are culoarea 0. Nodul 3 are culoarea 1. Nodul 4 are culoarea 2. Costul este obtinut astfel: Intre nodurile 0 si 1, costul este 1. Intre nodurile 1 si 2, costul este 1. Intre nodurile 2 si 3, costul este 1. Intre nodurile 3 si 4, costul este 3. Intre nodurile 0 si 4, costul este 2. 2 + 3 + 1 + 1 + 1 = 8

Intrare 2	**Ieșire 2**
3 3 2 0 1 0 2 1 2 0 1 100	-1
Acest exemplu contine un graf complet cu 3 noduri, având la dispozitie 2 culori distincte.	Numărul de culori disponibile este prea mic pentru a putea satisface restrictiile problemei, prin urmare outputul este -1.

2.4 Limite de timp

C++: 0.5 secunde
Java: 3.0 secunde

2. Problema 2 - **Expoziție de flori**

2.1 Enunț

Ștefania organizează o expoziție de flori în toate parcurile din București. Cum numărul așteptat de persoane este foarte mare, ea vrea ca fiecare alee din fiecare parc să poată fi traversată doar într-un singur sens pe durata expoziției. Mai mult, din fiecare intersecție a aleilor sau punct de acces al parcului trebuie să existe un număr par de alei pe care să poți să le urmezi.

Cum Ștefania este ocupată cu aspectul cromatic, vă roagă pe voi, studenți pasionați la Calculatoare, să găsiți orientarea corectă a aleilor.

2.2 Cerință:

Se dă un graf neorientat asupra căruia voi trebuie să modificați toate muchiile în arce și în plus la final gradul extern al fiecărui nod trebuie să fie par. Realizați această orientare dacă este posibilă.

2.3 Format date intrare/ieșire

Soluția va citi datele de intrare din fișierul **expozitie.in** și va scrie rezultatul în fișierul **expozitie.out**.

2.3.1 Format intrare:

- Pe prima linie se află un numar **N** (numărul de noduri în graf)
- Pe următoarele linii se află perechi de numere naturale **x y** cu semnificația că există muchie de la **x** la **y**.

2.3.2 Format ieșire:

- Se vor scrie pe câte o linie perechi de numere de forma **i j** semnificând faptul că muchia **[i, j]** a devenit arcul **(i, j)**, orientat de la **i** la **j**.
- Dacă operația de orientare nu este posibilă se va scrie mesajul **Imposibil**.

Restricții si precizări:
0 < N <= 5000
0 <= x, y < N
Numărul de muchii ale grafului <= 100000
Graful nu este neapărat conex.
Ordinea de scriere a arcelor nu contează.
Dacă există mai multe soluții, se va afișa una dintre ele.

Intrare	Ieșire
4 0 1 1 2 2 3 3 0	0 1 0 3 2 1 2 3
Graful are 4 noduri și 4 muchii.	Aceasta este o posibilă reprezentare. Exista 2 arce de la nodul 0 si 2 arce de la nodul 2.

2.4 Limite de timp

C++: 0.15 secunde
Java: 0.9 secunde

Format arhivă și testare

Temele pot fi testate automat pe vmchecker - acesta suportă temele rezolvate in C/C++ și Java. Dacă doriți să realizați tema în alt limbaj trebuie să trimiteți un e-mail lui Traian Rebedea (traian.rebedea@cs.pub.ro) în care să îi cereți explicit acest lucru.

Arhiva cu rezolvarea temei trebuie sa fie format zip, cu extensia **.zip** și să conțină în rădăcina acesteia:

- Fișierul/fișierele sursă
- Fișierul Makefile
- Fișierul README

Fișierul pentru make trebuie denumit obligatoriu **Makefile** și trebuie să conțină următoarele reguli:

- **build-p1**, care va compila sursele coresponzatoare primei probleme
- **build-p2**, care va compila sursele coresponzatoare celei de a doua probleme
- **run-p1**, care va rula executabilul pentru problema 1
- **run-p2**, care va rula executabilul pentru problema 2
- **clean**, care va șterge executabilele generate

Atentie! Numele regulilor trebuie să fie exact cele de mai sus, în special cele de run. Absența sau denumirea diferită a acestora va avea drept consecință obținerea a 0 puncte pe testele echivalente problemei rezolvate de regula respectivă.
Atenție! Pentru cei care folosesc C/C++ este *permisa* compilarea cu optiuni de optimizare a codului (O1,O2, etc.). Versiunea compilatorului de g++ de pe vmchecker este 4.7.2.