

Konstrukcija i analiza algoritama - pismeni ispit JUN2

27.09.2025.

Ispit traje 3h. Svaki zadatak nosi po 10 poena. Ukupan broj poena na pismenom delu ispita je 60 poena. Prag za prolaz je 17 poena.

Bi-bojenje

Bi-bojenje neusmerenog grafa dodeljuje svakom čvoru u grafu skup od dve boje. Postoje dva tipa bi-bojenja: U slabom bi-bojenju krajevi svake ivice moraju da koriste različite skupove boja; međjutim, ova dva skupa boja mogu da dele jednu boju. U jakom bi-bojenju, krajevi svake ivice moraju da koriste potpuno različite skupove boje, preciznije za svaka dva čvora u grafu skup njihovih boja treba da sadrži četiri različite boje. Svako jako bi-bojenje je takodje i slabo bi-bojenje. Korišćenjem činjenice da je 3-bojenje (bojenje koje svakom čvoru grafa dodeljuje jednu od tri ponudjene boje) validno ako nikoja dva čvora povezana granom nisu obojena istom bojom, dokazati:

- **(5 poena)** Proizvoljan graf G ima validno 3-bojenje ako i samo ako ima validno slabo bi-bojenje sa 3 boje.
- **(5 poena)** Proizvoljan graf G ima validno 3-bojenje ako i samo ako ima validno jako bi-bojenje sa 6 boja.

Rang elemenata u listi

Rang elementa u povezanoj listi definiše se kao rastojanje elementa od kraja liste. Tako, na primer, prvi element ima rang n , drugi $n - 1$, itd, a poslednji element rang 1. Data je povezana lista od n elemenata koji su smešteni u niz A dužine n . Izračunati rangove svih elemenata liste.

- **(2 poena)** Opisati sekvencijalni algoritam koji rešava pomenuti problem.
- **(3 poena)** Svakom elementu niza dodeljuje se po jedan procesor. Na početku svaki procesor zna samo adresu desnog (narednog) suseda svog elementa u listi. Opisati po koracima tok paralelnog algoritma.

- **(3 poena)** Napisati pseudo kod koji odgovara opisanom algoritmu.
- **(2 poena)** Diskutovati složenost i efikasnost paralelnog algoritma.

Generisanje tri broja sa datim verovatnoćama

Dat je funkcija `rand(a, b)` koja generiše jednako verovatne nasumične brojeve u intervalu $[a, b]$, uključujući i krajnje vrednosti.

Potrebno je generisati tri broja x, y, z sa verovatnoćama $P(x), P(y), P(z)$ tako da važi:

$$P(x) + P(y) + P(z) = 1$$

koristeći datu funkciju `rand(a, b)`.

Ideja je da se iskoristi osobina ravnomerne verovatnoće funkcije `rand(a, b)`. Neka su date verovatnoće izražene u procentima. Na primer:

$$P(x) = 40\%, \quad P(y) = 25\%, \quad P(z) = 35\%.$$

- **(4 poena)** Opisati funkciju koja vraća vrednost x sa verovatnoćom $P(x)$ vrednost y sa verovatnoćom $P(y)$ i vrednost z sa verovatnoćom $P(z)$
- **(4 poena)** Napisati pseudokod za osmišljeni algoritam
- **(2 poena)** Diskutovati ispravnost rešenja

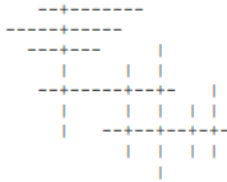
Kravljji preponski galop

Farmer Džon ima brilijantnu ideju za novi spektakularni sport: *Kravljji preponski galop*! Kao što je svima poznato, u regularnom preponskom galopu učestvuje grupa konja koji se trkaju oko staze ispunjene preprekama preko kojih moraju da preskaču. Farmer Džon misli da ista stvar može da funkcioniše i sa posebno istreniranim kravama, pod uslovom da prepreke budu dovoljno niske.

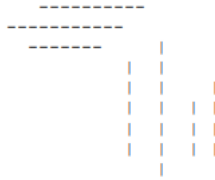
Kako bi dizajnirao stazu, Farmer Džon pravi dijagram svih N mogućih prepreka koje bi potencijalno mogao da izgradi. Svaka prepreka je predstavljena kao linijski segment u ravni, paralelan sa horizontalnom ili vertikalnom osom. Prepreka i ima krajnje tačke (X_{1i}, Y_{1i}) i (X_{2i}, Y_{2i}) , koje su različite, i zadovoljavaju

$$1 \leq X_{1i}, Y_{1i}, X_{2i}, Y_{2i} \leq 1\,000\,000\,000.$$

Farmer Džon želi da izgradi što je moguće više prepreka, uz ograničenje da se nijedne dve ne smeju seći. Dva segmenta se smatraju presečenim ako dele bilo koju tačku, čak i ako je to samo krajnja tačka. Farmer Džon je siguran da se u ulaznim podacima nijedna dva horizontalna segmenta ne seku, kao i da se nijedna dva vertikalna segmenta ne seku.



Slika 1: Primer rasporeda prepreka



Slika 2: Rešenje za sliku 1 - prepreke koje farmer Džon može da izgradi

Ulaz

Prva linija sadrži ceo broj C ($C \leq 10$), broj test primera.

Svaki test primer počinje sa brojem N , brojem prepreka. Sledećih N redova sadrži po četiri cela broja:

$$X_{1i}, Y_{1i}, X_{2i}, Y_{2i},$$

koji predstavljaju krajnje tačke i -te prepreke.

Izlaz

Za svaki test primer ispisati jedan ceo broj u posebnom redu: maksimalan broj prepreka koje je moguće postaviti tako da se nijedne dve ne seku.

Primer

Ulaz:

```
2
3
4 5 10 5
6 2 6 12
8 3 8 5
2
1 2 1 10
4 5 10 5
```

Izlaz:

2
2

Pokrivač skupova

Neka je dat univerzum $U = \{1, 2, \dots, n\}$ i kolekcija skupova $\mathcal{S} = \{S_1, S_2, \dots, S_m\}$ pri čemu je $S_i \subseteq U$. *Set Cover problem* traži najmanji podskup $\mathcal{C} \subseteq \mathcal{S}$ takav da $\bigcup_{S \in \mathcal{C}} S = U$, tj. da svi elementi univerzuma budu pokriveni izabranim skupovima.

Ovaj problem je NP-težak, pa se u praksi često koriste *aproksimativni algoritmi* koji daju dovoljno dobro rešenje u razumnom vremenu.

- **(10 poena)** Napisati implementaciju funkcije *set_cover_greedy_maxgain*, pohlepnog algoritma koji u svakom koraku bira skup koji pokriva najveći broj još nepokrivenih elemenata.

Primer:

Univerzum:

$$U = \{1, 2, 3, 4, 5, 6\}$$

Kolekcija podskupova:

$$S_1 = \{1, 2, 3\}, \quad S_2 = \{2, 4\}, \quad S_3 = \{3, 4, 5\}, \quad S_4 = \{4, 5\}, \quad S_5 = \{5, 6\}, \quad S_6 = \{1, 6\}$$

Greedy algoritam (uvek bira skup koji pokriva najveći broj nepokrivenih elemenata)

- **Početno stanje:** nepokriveno = $\{1, 2, 3, 4, 5, 6\}$. S_1 pokriva 3 nova elementa $\{1, 2, 3\}$, pa biramo S_1 . Nepokriveno $\rightarrow \{4, 5, 6\}$.
- Među preostalim skupovima: S_3 pokriva $\{4, 5\}$ (2 nova), S_4 pokriva $\{4, 5\}$ (2 nova), S_5 pokriva $\{5, 6\}$ (2 nova), S_6 pokriva $\{6\}$ (1), S_2 pokriva $\{4\}$ (1). Biramo S_3 (pokriva $\{4, 5\}$). Nepokriveno $\rightarrow \{6\}$.
- Ostao je element 6. $S_5 = \{5, 6\}$ ili $S_6 = \{1, 6\}$ oba pokrivaju 6. Biramo S_5 . Nepokriveno $\emptyset$.

Rešenje

Greedy rešenje je:

$$\{S_1, S_3, S_5\}$$

što daje ukupno 3 skupa.

Skupovne operacija nad AVL stablima

Data je implementacija AVL stabla sa sledećim operacijama: umetanje, brisanje, pretraga, rang, k -ti najmanji, brojenje u opsegu, prethodnik/sledbenik, itd. Potrebno je **dodatno** implementirati *skupovne operacije* nad dvema AVL strukturama bez duplikata (svako stablo ima jedinstvene ključeve). Implementirati sledeće funkcije:

- **(3 poena)** `Node* set_intersection(Node* A, Node* B)` – vraća novo AVL stablo sa $A \cap B$.
- **(3 poena)** `Node* set_difference(Node* A, Node* B)` – vraća novo AVL stablo sa $A \setminus B$.
- **(4 poena)** `Node* set_symmetric_difference(Node* A, Node* B)` – vraća novo AVL stablo sa $(A \triangle B) = (A \cup B) \setminus (A \cap B)$.

Primer ulaza:

```
14
I1 1
I1 3
I1 5
I2 3
I2 4
I2 6
UNION
INTER
DIFF12
DIFF21
SYMDIFF
UNION->1
PRINT1
T1
```

Izlaz:

```
1 3 4 5 6
3
1 5
4 6
1 4 5 6
1 3 4 5 6
1 3 4 5 6
4
```