Algoritmi in podatkovne strukture – 2
(2016/17)
Pisni izpit 13. mali srpan 2017

Pisni izpit morate pisati posamič. Pri reševanju je literatura dovoljena. Pri odgovarjanju bodite natančni in: (i) odgovarjajte *na zastavljena* vprašanja; in (ii) odgovorite na *vsa* zastavljena vprašanja.

Čas pisanja izpita je 90 minut.

Veliko uspeha!

NALOGA	TOČK	OD TOČK	NALOGA	TOČK	OD TOČK
1			3		
2			4		

IME IN PRIIMEK: _____

ŠTUDENTSKA ŠTEVILKA: _____

DATUM: _____

PODPIS: _____

1. naloga: Usmerjevalnik lahko definiramo kot napravo, ki ima k priključkov[1]. IP paket pride na enega od priključkov ter usmerjevalnik se mora odločiti, na kateri priključek mora prispeli paket preposlati. Podatek, na kateri priključek preposlati paket, usmerjevalnik hrani v podatkovni strukturi *usmerjevalna tabela*. Primer majhne usmerjevalne tabele je:

```
Destination        Gateway            Flags      Netif Expire
default            192.168.2.1        UGS         em0
127.0.0.1          link#3             UH          lo0
192.168.2.155/25   link#1             U           em0
192.168.2.101      link#1             UHS         lo0
192.168.2.0/23     link#2             U          igb0
192.168.126.1      link#2             UHS         lo0
```

Usmerjevalno tabelo lahko implementiramo s številskim drevesom z abecedo $\Sigma = \{0, 1\}$ in v njej so ključi naslovi mrež ter podatki priključki (vmesniki), kjer je priključena določena mreža. Na primer, v zgornji tabeli imamo ključa[2]

```
192.168.2.155/25 = 11000000 10101000 00000010 1.......
192.168.2.0/23 = 11000000 10101000 0000001.  ........
```

in podatka `em0` oziroma `igb0`. Ko pride paket, ga usmerjevalnik pošlje na tisti priključek, katerega ključ se najbolj ujema z njegovim IP naslovom. Na primer, če pride paket z naslovom 192.168.2.140, ga usmerjevalnik prepošlje na priključek `em0`, saj se bolje ujema s prvim ključem in če pride za naslov 192.168.2.14, ga prepošlje na naslov `igb0`.

VPRAŠANJA:

A) Iz ključev in podatkov[3]

(136/1, A)	(138/8, B)	(16/7, C)	(162/8, Č)	(92/5, D)
(32/4, E)	(91/2, F)	(10/6, G)	(82/3, H)	(47/3, J)

zgradite številsko drevo, ki je stisnjeno po poti (*PATRICIA tree*).

B) Razmislite, ali bi bilo smiselno stisniti drevo še po plasteh. Utemeljite odgovor.

C) Recimo, da pride paket z naslovom d. Zapišite algoritem, ki bo na podlagi usmerjevalne tabele, implementirane kot številsko drevo, vrnil vmesnik, na katerega naj se paket prepošlje.

[1] k je lahko vse od majhnega števila kot je 4 pa tja do več deset.

[2] Presledki so zgolj zaradi preglednosti, v resnici gre (v prvem primeru) za niz 25 ničel in enic, saj je 25 dolžina maske.

[3] Za lažje delo so ključi največ 8 bitni, medtem ko so podatki enočrkovni.

2. naloga: Imamo naslednja števila

$$59, 29, 93, 40, 40, 21, 82, 37, 36, 82, 15, 38, 6, 8, 5, 86, 68, \quad (1)$$

ki so shranjena v polju `S` pričenši od indeksa 0 naprej. Poleg tega definirajmo `Max(S, i, j)`, ki vrne največji element v polju `S` med indeksoma i in j vključno. Recimo, `Max(S, 7, 12)` vrne 82.

VPRAŠANJA:

A) (i) Predlagajte implementacijo operacije `Max(S,i,j)` za poljubna i in j ter časovno zahtevnostjo $O(1)$. Pred tem lahko izvedete *poljubno predprocesiranje*. (ii) Kakšna je prostorska zahtevnost vaše rešitve? Utemeljite odgovor.

B) Osnovna operacija nad poljem je dostop do i. elementa – na primer `S[4]` pomeni peti element v polju `S`. Običajno je polje implementirano kot implicitna podatkovna struktura. Lahko pa jo implementiramo tudi kot drugače in recimo, da je implementirana kot preskočni seznam – seveda elementi niso urejeni po velikosti v takšnem seznamu. Na primer implementacija polja `S` števil v vrstici (1) ima kot prvi element 59, nato 29 in tako naprej. (i) Opišite postopek, kako dostopiti do i. elementa v preskočnem seznamu. (ii) Kakšna je časovna zahtevnost vaše operacije?

C) Nabor operacij razširimo z operacijo `Insert(S, i, e)`, ki na i. mesto vstavi element e in vse preostale elemente premakne za enega naprej. Recimo, če nad našimi podatki (1) izvedemo operacijo `Insert(S, 1, 77)`, potem bo na 0. mestu 59, na prvem bo vstavljena 77, na drugem 29, in tako naprej do 25. mesta, kjer je 46. (i) Opišite učinkovito implementacijo podatkovne strukture in operacij `Max()` in `Insert()`. (ii) Ali lahko izvedemo operacijo `Max()` v času $o(logn)$, kjer je n število elementov v polju? Utemeljite odgovor.

3. naloga: Peter Zmeda ima za domačo nalogo sedem problemov, pri čemer lahko za vsakega od problemov dobi različno število točk. Poleg tega tudi ocenjuje, da potrebuje za reševanje različnih problemov različno količino časa:

točke	7	9	5	12	14	6	12
čas (v h)	3	4	2	6	7	3	5

Za izdelavo domače naloge ima na voljo 15 ur časa. Očitno ima na voljo premalo časa, da reši vse probleme, in tako se mora odločiti, katere naj rešuje. Opravka imamo z optimizacijskim problemom.

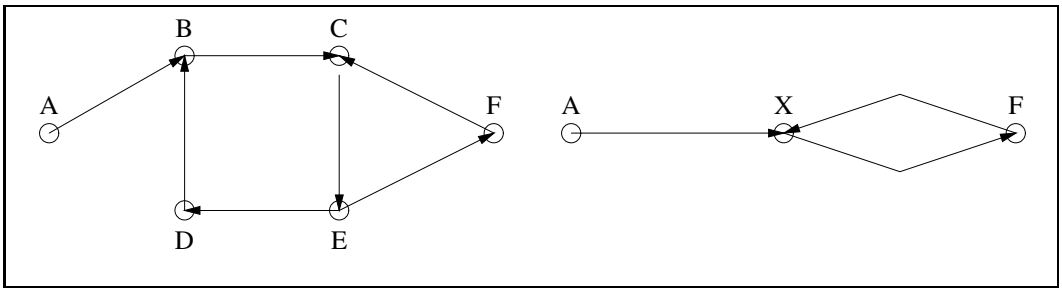
VPRAŠANJA:

A) Najprej predpostavimo, da lahko dobi delne točke. To pomeni, da, če reši pol prvega problema, porabi 1,5h in dobi 3,5 točke. (i.) Predlagajte, katere probleme in kako naj jih rešuje, da bo dobil čim več točk. (ii.) Utemeljite pravilnost vašega odgovora.

B) Recimo, da imamo n problemov s točkami p_i in časi t_i ter časovno omejitvijo T in lahko probleme rešujemo samo delno. (i.) Zapišite algoritem, ki predlaga optimalno strategijo reševanja. (ii.) Utemeljite pravilnost vašega algoritma. (iii.) Kakšna sta časovna in prostorska zahtevnost problema? Utemeljite odgovor.

C) Sedaj predpostavimo, da Peter dobi točke samo, če v celoti reši problem (načelo vse ali nič). (i.) Katere probleme iz zgornje tabele naj sedaj reši? (ii.) Utemeljite pravilnost rešitve tokrat. (iii.) Kaj pa če imamo ponovno n problemov kot pri prejšnjem vprašanju ter iščemo sedaj optimalno rešitev, ko moramo probleme v celoti rešiti. Zapišite algoritem.

4. naloga: Na sl. 1 imamo najprej usmerjen graf, ki vsebuje cikel med točkami



Slika 1: Sesedanje ciklov.

$BCDEB$, ki se v drugem grafu sesede v eno samo točko X. V nalogi bomo imeli opravka z USMERJENIM grafom $G(V, E)$, kjer $|V| = n$ in $|E| = m$.

VPRAŠANJA:

A) Naj bo $A \in V$. (i.) Zapišite algoritem `Cikel(`G`,` A`)`, ki ugotovi, ali obstaja cikel v G, ki vsebuje točko A, in vrne seznam vozlišč c, ki tvorijo cikel. (ii.) Kakšna je časovna zahtevnost vašega algoritma. Utemeljite odgovor. (iii.) Ali obstaja algoritem s časovno zahtevnostjo $o(m)$. Utemeljite odgovor.

B) (i.) Zapišite algoritem `Sesedi(`G`,` c`)`, ki v grafu G sesede cikel c, kjer je c rezultat funkcije `Cikel(`G`,` A`)`, v točko. Ob tem predpostavite, da je graf G podan z matriko sosednosti in da imamo med točkama A in B največ eno usmerjeno povezavo iz A v B (nimamo multigrafa). (ii.) Kakšna je časovna

zahtevnost vašega algoritma? Utemeljite odgovor. (iii.) Funkciji `Cikel()` in `Sesedi()` sta že implementirani. Sestavite algoritem, ki sesede vse točke v grafu G.

C) Kot vemo, je iskanje Hamiltonovega cikla v grafu NP-poln problem. Peter Zmeda je rešil prejšnje vprašanje v tej nalogi in prišel na idejo, da preprosto sesede vse cikle v grafu in, če bi dobil samo eno točko, bi s tem našel Hamiltonov cikel. (i.) Zapišite (narišite) primer grafa, kjer je njegov pristop pravilen. (ii.) Zakaj v splošnem njegov pristop ni pravilen? Utemeljite odgovor. Najbolje s primerom grafa, kjer njegov pristop vrne napačen rezultat.