TKT20001 Tietorakenteet ja algoritmit (kevät 2019)

Kurssikoe 2 (8.5.2019)

Tentissä saa olla mukana käsin kirjoitettu yksi A4-kokoinen "lunttilappu", jonka molemmilla puolilla saa olla tekstiä. Muu lisämateriaali (laskimet, taulukot tms.) ei ole sallittua.

Vastaa kuhunkin tehtävään **erilliselle** konseptipaperille.

Kirjoita jokaisen paperin yläkulmaan **kurssin nimi, kokeen päivämäärä, oma nimesi ja opiskelijanumerosi**. Vaikka jättäisit johonkin tehtävään vastaamatta, palauta silti vastauspaperi kyseiseen tehtävään.

Tehtävissä 3 ja 4, joissa pyydetään ratkaisualgoritmia annettuun ongelmaan, voit esittää algoritmisi käyttämällä luentojen ja kurssikirjan tyyppistä pseudokoodia tai muita ymmärrettäviä pseudokoodityylejä tai oikeaa ohjelmointikieltä, esim. Javaa. Jos käytät oikeaa ohjelmointikieltä, selitä erityisen hyvin, mitä ohjelmassasi tapahtuu, äläkä käytä mitään kielen erikoisia piirteitä. Myös algoritmin sanallinen kuvaus ilman pseudokoodia kelpaa, kunhan se on rittävän selkeä ja yksityiskohtainen, että halutun aikavaativuuden yms. pystyy selvästi toteamaan. Näissä tehtävissä voit käyttää kaikkia kurssilla esitettyjä tietorakenteita ja algoritmeja, niiden tunnettuja aikavaativuuksia jne., kunhan sanot selvästi, mitä milloinkin käytät.

Vastaa kaikkien kysymysten kaikkiin kohtiin. Kokeen maksimipistemäärä on 20.

1. [*4 pistettä*] Selitä seuraavat käsitteet:

   (a) peruuttava haku ja branch-and-bound

   (b) topologinen järjestys

   (c) vahvasti yhtenäiset komponentit

   (d) maksimivirtaus ja minimileikkaus.

   Sopiva vastauksen tarkkuustaso on esim. lyhyt (parin virkkeen mittainen) sanallinen selitys, jota havainnollistetaan yksinkertaisella kuvalla.

2. [*6 pistettä*] Vastaa seuraaviin kohtiin lyhyesti. Kun pyydetään lyhyttä selitystä, se todella tarkoittaa lyhyttä (noin yksi virke); tarkoituksena ei ole selittää asiaa tyhjentävästi vaan osoittaa kokeen tarkastajalle, että tiedät, mistä on kysymys.

   (a) Anna esimerkki ongelmasta, joka voidaan ratkaista soveltamalla syvyyshakua, mutta ei leveyshakua. Anna myös esimerkki ongelmasta, joka voidaan ratkaista soveltamalla leveyshakua, mutta ei syvyyshakua.

   (b) Anna esimerkki algoritmista, jossa verkko on parempi esittää vieruslistana kuin vierusmatriisina. Anna myös esimerkki algoritmista, jossa verkko on parempi esittää vierusmatriisina kuin vieruslistana. Kummassakin tapauksessa selitä lyhyesti, miksi kyseinen esitystapa on parempi.

   (c) Anna esimerkki tilanteesta, jossa suunnatussa painotetussa verkossa yhden kaaren paino on negatiivinen ja sen takia Dijkstran algoritmi ei toimi oikein. Selitä lyhyesti, miksi näin käy.

(d) Anna esimerkki painotetusta suuntaamattomasta verkosta, jonka kaikkien kaarten painot ovat eri suuria ja jonka pienin virittävä puu ei sisällä jonkin syklin keveintä kaarta.

(e) Mainitse jokin verkkoalgoritmi, joka käyttää jonoa. Mainitse myös jokin verkkoalgoritmi, joka käyttää prioriteettijonoa. Selitä lyhyesti, mihin tietorakenteita kyseisissä algoritmeissa käytetään.

(f) Anna esimerkki ongelmasta, jonka ratkaisussa taulukointiin perustuva dynaaminen ohjelmointi on oleellisesti tehokkaampi kuin vastaava rekursiivinen algoritmi. Selitä lyhyesti, mitä taulukkoon tallennetaan.

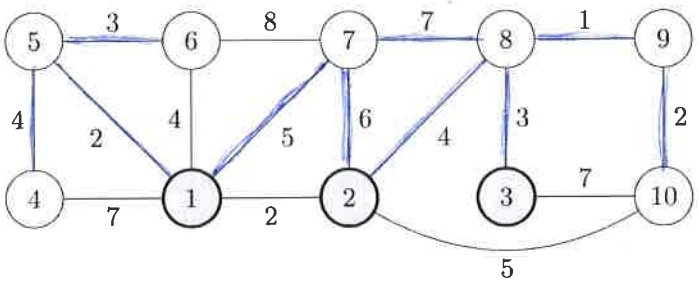
3. [*5 pistettä*] Syötteenä on annettu suunnattu verkko $G = (V, E)$ ja jokaiselle *solmulle* $v \in V$ positiivinen paino $w(v) > 0$. Lisäksi on annettu lähtösolmu s ja kohdesolmu t. Halutaan tietää kustannukseltaan pienin polku solmusta s solmuun t, kun polun kustannus on sillä esiintyvien solmujen painojen summa.

   Esitä ongelmalle tehokas ratkaisualgoritmi. Mikä on algoritmisi pahimman tapauksen aikavaativuusluokka? Perustele lyhyesti.

4. [*5 pistettä*] Syötteenä on annettu suuntaamaton verkko $G = (V, E)$ ja jokaiselle kaarelle (u, v) positiivinen paino $w(u, v) > 0$. Lisäksi on annettu solmujoukko $S \subseteq V$. Halutaan löytää kokonaispainoltaan pienin mahdollinen sellainen kaarijoukko $F \subseteq E$, että jokaisesta verkon solmusta $v \in V$ on johonkin joukon S solmuun $s \in S$ polku, joka käyttää vain joukon F kaaria.

   Esitä ongelmalle tehokas ratkaisualgoritmi. Perustele täsmällisesti, että algoritmisi toimii oikein eli tuottaa halutun lopputuloksen Mikä on algoritmisi pahimman tapauksen aikavaativuusluokka? Perustele lyhyesti.

   Esimerkki: Allaolevassa verkossa $S = \{1, 2, 3\}$, mikä on merkitty varjostamalla solmut 1, 2 ja 3:



   Optimaaliseen ratkaisussa joukkoon F tulee kaaret (5,6), (5,4), (5,1), (1,7), (8,3), (8,9) ja (9,10). Valittujen kaarten kokonaispaino on $3 + 4 + 2 + 5 + 3 + 1 + 2 = 20$. Solmuista 4, 5 ja 6 on polku solmuun 1, solmuista 8, 9 ja 10 on polku solmuun 3 ja solmuun 2 ei tule polkuja muista solmuista: