Tietorakenteiden harjoitustyö

Toteutusdokumentti

Tehnyt: Taneli Virkkala

Opiskelijanumero:

Ohjaaja: Kristiina Paloheimo

Aihe: Verkon virittävät puut

Kurssi: Tietorakenteiden harjoitustyö: loppukesä 2012

Yliopisto: Helsingin yliopisto, Tietojenkäsittelytieteen laitos

Päivämäärä:

1. **Ohjelman yleisrakenne**

UML-tullaan lisäämään…

Painotettuverkko

Koostuu solmuista (ArrayList<Solmu>). Eli siis sisältää solmuolioita. Metodeihin kuuluu mm. solmujen lisäystä sekä tietyn solmun hakua. Verkko on siis tässä tapauksessa vain lista solmuista. Se on myös suuntaamaton.

Solmu

Solmulla on numero, joka kertoo solmun ”nimen”. Eli siis numerolla viitataan tähän kyseiseen solmuun. Jokaisella solmulla on minimikeko omista kaaristaan (PriorityQueue<Kaari>). Kaaret ovat aina suuruusjärjestyksessä. Solmu tuntee naapurinsa vain kaarien kautta.

Kaari

Kaarella on lähtösolmu (Solmu1) sekä päätesolmu (Solmu2). Sen lisäksi sillä on paino (paino). Kaari on Comparable luokka, joten kaaria voi vertailla toisiinsa painon avulla.

Prim

Algoritmi jolle annetaan parametreina aloitussolmu r sekä Painotettuverkko G. G:stä siis etsitään virittävä puu T. Aluksi r:n pieninkaari poimitaan, jonka jälkeen itse kaari lisätään T:hen. Sekä r että päätesolmu merkitään käydyiksi. Sen jälkeen siirretään kaikki r:n ja päätesolmun kaaret isoon Minimikekoon (PriorityQueue<Kaari> kaikkikaaret). Nyt aina poimitaan kaikistakaarista pienin kaari. Jos pienimmän kaaren päätesolmua (Solmu2) ei ole merkitty käydyksi, se merkitään ja Solmu2:sen kaaret lisätään kaikkiinkaariin. Jos taas Solmu2 on käyty, poimitaan keosta uusi pienin kaari. Algoritmi loppuu kun virittävä puu T on valmis.

Kruskal

Algoritmille annetaan vain verkko, josta virittävä puu luodaan. Aluksi otetaan kaikkien solmujen kaaret minimikekoon kaikkikaaret (PriorityQueue<Kaari>). Sen jälkeen aletaan aina pienin kaari poistaen merkitä kaaren lähtösolmua (Solmu1) ja päätesolmua (Solmu2) läpikäydyiksi. Lisäksi tehdään lisäys poistetulle kaarelle kekoon kaymattomatkaaret(PriorityQueue<Kaari>) Kun kaikki solmut on läpikäyty, yhdistetään metsät (setit) etsimällä erilaisuudet ja lisäämällä muutama kaari. Kaari napataan tässä tapauksessa käymättömistä kaarista. Algoritmi loppuu kun käymättömät kaaret on tyhjä.

Käyttöliittymä

Käyttöliittymässä voi luoda itse oman verkon kaarineen ja solmuineen. Verkossa on oltava väh. 3 solmua ja 2 kaarta. Aika mitataan algoritmeille System.curretTimeMillis()- metodilla. Myös virittävät puut tulostetaan.

1. **Aika- ja tilavaativuudet**

Kruskalin ja Primin algoritmit vievät aikaa O(E log V). E tarkoittaa kaarien lukumäärää ja V solmujen. Ohjelman pitäisi siis tuottaa tulokset ajassa O(2x(E log V)). Tilavaativuus on solmujen lukumäärä O(V). Ohjemassani kuitenkin on myös minimikeko kaarille, joten tilavuus kasvaa. Lisäksi muutama ylimääräinen for-lause.

1. **Suorituskyky- ja O-analyysivertailu**

Tullaan täyttämään…

1. **Työn mahdolliset puutteet ja parannusehdotukset**

TODO.

1. **Lähteet**

Patrick Floreenin Tietorakenteet kurssin materiaali: <http://www.cs.helsinki.fi/u/floreen/tira2012/tira.pdf>

Primin algoritmi: Robert C. Prim (1957)

Kruskalin algoritmi: Joseph Kruskal (1956)

1. **Käyttöohjeet**

Käyttöliittymässä painamalla (a) voi lisätä solmun verkkoon, jonka jälkeen syötetään kokonaisluku (int).

Painamalla (b) voi lisätä kaaren verkossa olevan kahden solmun välille. Painon saa myös itse päättää.

Painamalla (c) voi ajaa algoritmit. Tuloksessa näkyy aika millisekunteina sekä viritetyt puut, jotka ovat syntyneet. Primin algoritmiin on valittava aloitussolmuksi jo verkossa oleva solmu.

Painamalla (d) voi tulostaa tämänhetkisen luoman käyttäjän testiverkon.

Painamalla (q) ohjelman suoritus loppuu.