Dokumentacija

3. laboratorijska vježba iz Naprednih algoritama i struktura podataka

Student: Lea Suć

Matični broj: 0036444175

Zadatak: Traženje najkraćeg puta između dva vrha grafa pomoću Dijkstrinog algoritma

Službeni zadatak

Modelirati grafom dio nekog naselja i programski odrediti najkraći put između dva mjesta (dva vrha). Nije potrebno graditi komplicirane modele, dovoljni su grafovi s 10...20 vrhova. Program mora riješiti pohranu grafa u kompjutoru, pronalaženje najkraćeg puta i ispis (iscrtavanje) rješenja, pri čemu iscrtavanje grafa i najkraćeg puta nije obavezno, nego samo poželjno, ali prikladan ispis najkraćeg puta je obavezan. Za iscrtavanje je dovoljno iskoristiti osnovnu funkcionalnost programa Graphviz.

1. Dijkstrin algoritam

Dijkstrin algoritam (nazvan po nizozemskom informatičaru E. W. Dijkstri), služi za pronalaženje najkraćeg puta u usmjerenom grafu sa nenegativnim „težinama“ bridova. Ako zamislimo čvorove grafa kao gradove, a vrijednosti, odnosno težine bridova kao udaljenosti između direktno povezanih gradova, Dijkstrin algoritam nalazi najkraći put između dva grada.

Neka je G težinski usmjeren graf i zadan je početni čvor S iz G. Ako skup svih čvorova grafa obilježimo s V, a skup bridova s E, svaki brid iz E predstavljen je parom čvorova (u, v) iz V koje taj brid povezuje. Neka svaki brid ima određenu vrijednost (težinu) označenu s w(u,v). Težina svakog brida može se predstaviti kao udaljenost između dva čvora koje on povezuje. Dužina puta između dva čvora suma je težina bridova na tom putu.

Općenito za svaki vrh v_i na najkraćem putu vrijedi

$$d_{min}(v_0, v_n) = d_{min}(v_0, v_i) + d_{min}(v_i, v_n).$$

Dakle, gledajući unatrag, ako svaki vrh na najkraćem putu ima poznatog barem svojeg neposrednog prethodnika, može se rekonstruirati cijeli put do polaznog vrha.

Česta upotreba Dijkstrinog algoritma je, osim za nalaženje nakraćeg puta između zadana dva čvora iz V, za nalaženje najkraćeg puta od određenog čvora do svih ostalih iz skupa V.

2. Programsko ostvarenje

Rješenje zadatka ostvareno je u programskom jeziku Java. U izradi je korišteno integrirano razvojno okruženje Eclipse IDE za Javu. Iscrtavanje se obavlja pomoću slobodnog (open source) programa Graphviz koji se može preuzeti sa stranice http://www.graphviz.org/. Za uspješno izvršavanje programa potrebno je imati datoteku „graf.txt“ u kojoj je pohranjen graf. Datoteka se učitava s lokacije „C://graf.txt“, no moguće je jednostavno u programu promijeniti put učitavanja.

Dijkstrin algoritam može se prikazati sljedećim pseudokodom:

```
Dijkstra (source, dest)
initialization: for all vertices d(v)= infinity
d(source) = 0;
unvisited = all vertices;
while(unvisited != empty)
    v = vertex in unvisied with the least d(v);
if v == dest
    return ;
remove v from  unvisited;
for all vertices u adjacent to v and in unvisited
    dnew(u)=d(v)+edge(v,u);
    if dnew(u) < d(u)
        d(u) = dnew(u);
    predecessor(u) = v;
```

Slika 1. Pseudokod Dijkstrinog algoritma

Klasa Node

Ova klasa predstavlja čvor, odnosno vrh grafa. Sadrži članske varijable `String label`, koja predstavlja oznaku, odnosno naziv vrha, `int distance`, koja predstavlja udaljenost od početnog vrha, `Map<Node, Integer> connections`, u kojoj su pohranjeni susjedni vrhovi i težine veza između njih i promatranog vrha, te `Node predecessor`, koja predstavlja referencu na najbliži prethodni vrh, što se koristi kod rekonstrukcije najkraćeg puta. U ovoj klasi važna je metoda `connect(Node n, int`

`value)`, koja „povezuje“ promatrani vrh sa susjedima, odnosno pohranjuje u mapu referencu na susjeda i udaljenost do njega.

Klasa Graph

Ova klasa predstavlja ključan dio programa, jer sadrži metodu koja po Dijkstrinom algoritmu traži najkraći put između dva vrha, čije se oznake zadaju prilikom poziva metode `Dijkstra(String n1, String n2)`. Dijkstrin algoritam izvršava se prema pseudokodu prikazanom na slici 1. Na kraju algoritma, pronađeni najkraći put se rekonstruira od traženog vrha, preko njegovih prethodnika, do početnog, i pohranjuje u listu path, koju ova metoda potom vraća.

Kao člansku varijablu ova klasa sadrži listu vrhova u grafu u kojem tražimo najkraći put između određena dva vrha. Osim toga, klasa sadrži metode za dohvaćanje i dodavanje vrhova, te povezivanje oznake vrha s oznakom susjeda i udaljenošću do njega. Nakon izvršavanja algoritma, u main metodi potrebno je pozvati `saveGraph(ArrayList<Node> path)`, koji ispisuje zadani graf i dobiveni najkraći put u datoteku s ekstenzijom ".dot", koja se nakon toga može jednostavno prikazati pomoću Graphviz programa.

Preostale klase

U klasi MainClass poziva se metoda koja učitava graf iz datoteke, metoda koja izvršava Dijkstrin algoritam i metoda koja potom pohranjuje graf s označenim najkraćim putem u datoteku namijenjenu otvaranjem s Graphviz programom. U ovoj klasi se, osim toga, na standardni izlaz ispisuje pronađeni najkraći put kao niz vrhova odvojenih strelicom.

Klasa DataLoading služi za učitavanje grafa iz tekstualne datoteke. Graf je pohranjen u obliku oznaka početnog, sljedećeg vrha i udaljenosti među njima, odvojenih razmakom.

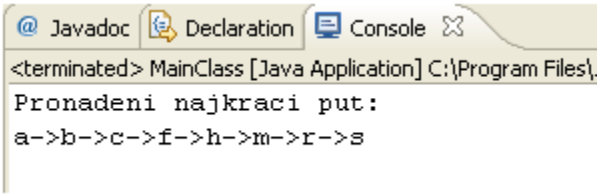
Klasa NodeComparator uspoređuje vrhove po varijabli koja predstavlja udaljenost od početnog vrha, a služi za jednostavno sortiranje liste vrhova prema udaljenosti.

3. Primjer rada programa

Željeni graf zadaje se u datoteci „graf.txt“, dakle nije hardkodiran i moguće ga je proizvoljno zadati. Pritom su svaka dva spojena vrha u datoteci zapisana kao oznaka prvog, oznaka drugog vrha i udaljenost među njima, koja mora imati nenegativnu vrijednost, a između moraju biti razmaci. Primjer zadavanja grafa prikazan je na slici 2.

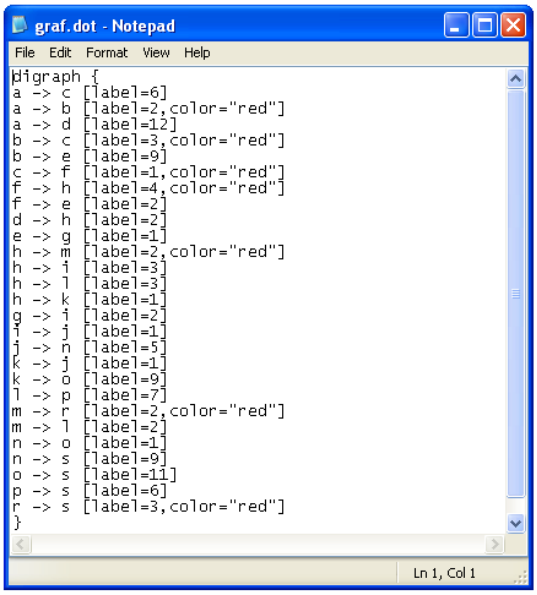
Slika 2. Zadani vrhovi grafa i udaljenosti između spojenih vrhova

Nakon izvršavanja programa na standardni izlaz ispisuje se pronađeni najkraći put. U ovom primjeru traži se put između vrhova „a“ i „s“, a izlaz programa prikazan je na slici 3.



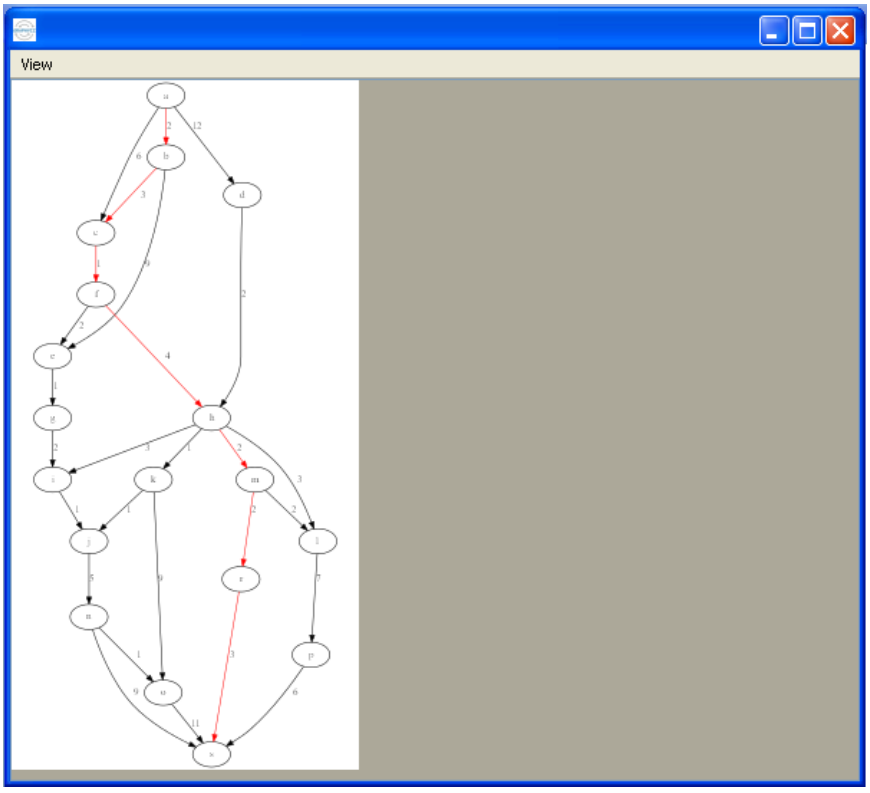
Slika 3. Izlaz programa za graf zadan u tekstualnoj datoteci, u kojem se traži najkraći put između vrhova „a“ i „s“

Nakon izvršavanja programa u datoteku „graf.dot“ ispisuju se se vrhovi i veze među njima u posebnom obliku kako bi se graf automatski mogao otvoriti Graphviz programom. Primjer je prikazan na slici 4.



Slika 4. Graf kako je ispisan u datoteci „graf.dot"

Primjer iscrtanog grafa s označenim najkraćim putem prikazan je na slici 5.



Slika 5. Graf u datoteci „graf.dot", vizualiziran Graphviz programom