Fakultet elektrotehnike i računarstva

Zavod za primjenjeno računarstvo

**Napredni algoritmi i strukture podataka**

3. laboratorijska vježba

Vinko Kodžoman 0036470625

Zagreb, 19.1.2016

**Zadatak**

2) Modelirati grafom dio nekog naselja i programski odrediti najkraći put između dva mjesta (dva vrha). Tko želi, može modelirati i nešto drugo, gdje bi bilo čak i bridova negativnih težina.

− za kolokviranje vježbe važno je pregledno i jasno opisati model te organizaciju podataka u programu

− nije potrebno graditi komplicirane modele, dovoljni su grafovi s 10...20 vrhova. Naravno, kompliciraniji modeli će vjerojatno biti i izazovniji te kao takvi zanimljivija i „plodonosnija“ laboratorijska vježba.

− program mora riješiti pohranu grafa u kompjutoru, pronalaženje najkraćeg puta i ispis (iscrtavanje) rješenja

− iscrtavanje grafa i najkraćeg puta nije obavezno, nego samo poželjno, ali prikladan ispis najkraćeg puta je obavezan

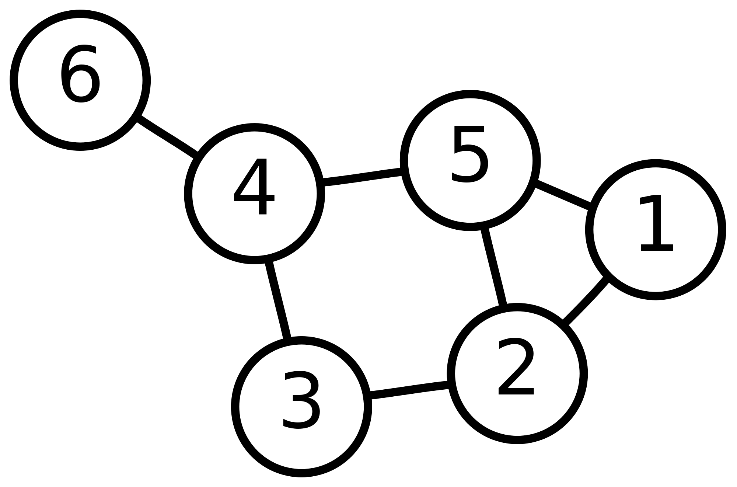
− iscrtavanje se brzo i relativno lako može postići prepuštanjem tog posla slobodnom (open source) programu Graphviz koji možete preuzeti sa stranice http://www.graphviz.org/, gdje su i podrobne upute za njegovo korištenje. Dovoljno je iskoristiti samo njegovu osnovnu funkcionalnost, bez posebnog dotjerivanja rješenja, a ni njegovo pozivanje ne mora biti automatsko, nego je dovoljno programski pripremiti podatke za Graphviz, a pozivati ga možete i „ručno“ iz komandne linije.

− tko ne zna što modelirati, može raditi s grafom na slici

**Rješenje zadatka**

**Teorijski uvod**

Kako bismo mogli rješiti problem iz stvarnog svijeta na računalu moramo ih modelirati. Naći odgovarajući model nije lako, predmeti iz stvarnog svijeta su prepuni karakteristika koju su nama sa računarske strane nepotrebni. Modeliranje protoka, mreža, puteva, mapa i drugi stvari je lagano sa grafovima. Graf je uređeni par nepraznog konačnog skupa vrhova V (vertex) i skupa (moguće praznog) bridova E (edge); G=(V,E). Grafovi nam omogućuju da modeliramo putove (skupove bridova) između nama bitnih odredišta (vrhova) [Slika 1].



Slika Ne usmjereni graf sa 6 vrhova

**Brid** (edge) je svaki dvočlani podskup skupa V (skup vrhova).

**Red** (order) grafa je broj vrhova u njemu.

Gornji graf ne daje puno informacija on može pretstavljati, ceste u gradu i odredišta koje poštar mora dostaviti. Dodavanjem težina na grafove možemo modelirati udaljenost između pojedinih lokacija (vrhova).

**Petlja** (loop) je brid koji počinje i završava u istom vrhu.

**Supanj** (degree) vrha je broj bridova koji su priležeći (incident) tom vrhu (spojeni s njim).

Jedan od velikih problema unutar grafa je pronalaženje najkraćeg puta između dva vrha, za ovu laboratorijsku vježbu sam odabrao primjenu BFS i heurističke pretrage A\*.

**Implementacija**

Program je napisan u programskom jeziku Java u IntelliJ IDEA 15 (Jet Brains). Niti jedna vanjska biblioteka nije korištena, osim onih ugrađenih u JE-08 (kolekcije, rad sa ulazom, izlazom, ...)

**CanvasMaker**

Parser učitava točke tj. lokacije koje predstavljaju sustave u svemiru iz datoteke (input.txt) u obliku matrice. Učitani podaci se modeliraju u obliku matrice sa enumeracijamak koje predstavljaju polje. Nakon što su svi podaci učiani korisnik bira vrstu rada algoritma (BFS ili A\*).

Primjer matrice ulaza:

1 2 7 T1 8 4 6 12

4 1 8 2 14 9 12 8

P T2 9 4 7 T2 1 15

7 3 T1 8 4 6 7 SL

1 2 3 4 5 T1 7 9

8 2 5 9 T2 4 6 10

SS 1 T2 8 1 8 5 C

2 3 4 5 6 T1 8 10

Gornja matrica modelira svemir. Igrač kreće sa lokacije P i završava na lokaciji C. Cilj je doći od P do C plaćajući što manju cijenu i što efikansije (sa algoritamske strane). Igrač se može kretati gore, dole, lijevo i desno, ali ne može proći sa lijeve polovice matrice na desnu. Razlog tome je jer lijeva polovica matrice predstavlja jednu galaksiju, a desna drugu. Jedini naćin da se pređe je putem teleportera (T) ili space shuttla (S). Koštanje micanja sa jednog polja na drugo je apsolutna razlika njihovih visina. Teleportirati se može samo na teleportere istih indeksa i cijena je jednaka manhatanovoj udaljenosti između lokacija teleportera. Cijena shuttla je 3 \* manhattanova udaljenost između shuttle lokacija.

Heuristika primjenjena u A\* algoritmu je ona koja navodni pretraživanje prema cijelu tako da dodaje vrijednost kao manhattan udaljenost od trenutne lokacije do cilja.

**Zaključak**

Algoritam kao BFS sigurno pronalazi riješenje, ali za velike probleme je algoritam ne primjenjiv. Kako bismo mogli riješiti velike probleme koriste se greedy algoritmi i heuristike. A\* algoritam je BFS sa dodatkom heurističke funkcije koja navodi algoritam k točnom riješenju, i samim time smanjuje prostor pretrage.

**Literatura**

|  |  |
| --- | --- |
| **Vrsta** | **Format** |
| Knjiga | Domagoj Kusalić: Napredno programiranje i algoritmi u C-u i C++-u 5. nepromijenjeno izdanje, Element, Zagreb, 2014. |
| Prezentacije | UI 1-3 Pretraživanje prostora |