Kako povezati MaRSovce?

Rafi Irgolič, *Ljubljana, Gimnazija Bežigrad*
Anže Košir, *Dobrova, Polhov Gradec, Elektrotehniško-računalniška strokovna šola in gimnazija Ljubljana*
Živa Urbančič, *Ljubljana, Gimnazija Vič*
Matej Roškarič (mentor), *Maribor, Fakulteta za naravoslovje in matematiko*

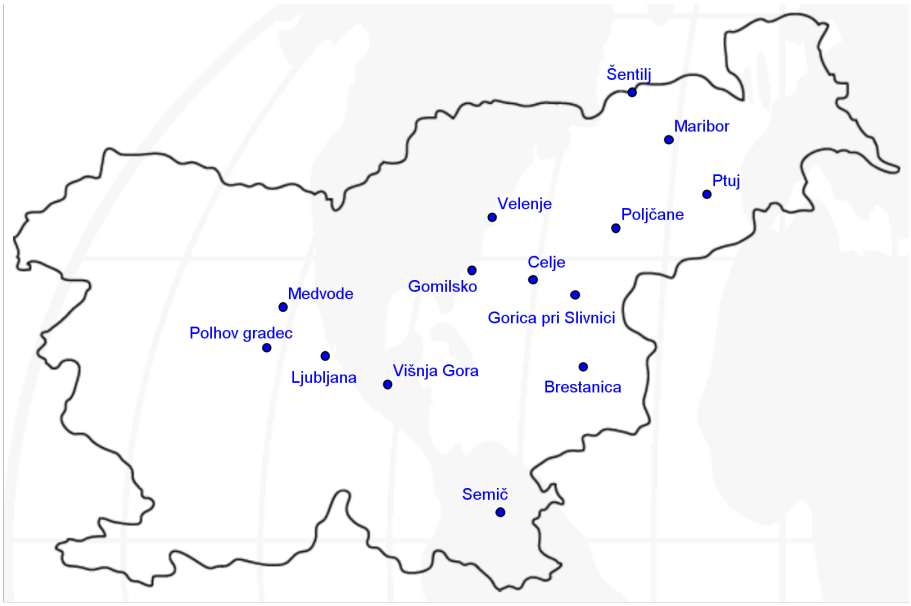
POVZETEK

Matematično raziskovalno srečanje oz. MaRS na žalost traja le en teden na leto, zato smo se spraševali, kako povezati MaRSovce, da bomo po koncu tabora lahko ohranili stike. Kraje iz katerih prihajamo in povezave med njimi si lahko predstavljamo kot graf, zato smo se lotili raziskovanja povezanih grafov in minimalnih vpetih dreves. Pomagali smo si z dvema od že odkritih algoritmov: Primovim in Kruskalovim. Napisali smo tudi program, ki je našel optimalen način, kako povezati MaRSovce.

Uvod

Grafe sestavljata množica točk, imenovanih vozlišča, ter množica povezav med njimi. Raziskovali bomo prav posebno vrsto grafov, ki jih imenujemo povezani grafi. To pomeni, da lahko iz poljubnega vozlišča pridemo v vsako drugo vozlišče v grafu. Povezanemu grafu, v katerem lahko iz poljubnega vozlišča pridemo v vsako drugo, natanko na en način, pravimo drevo. Če povezavam določimo vrednosti, pa lahko govorimo o ceni grafa. Tukaj se nam postavi zanimivo vprašanje. Kako določiti, katere povezave so najbolj ugodne za nas, da bomo povezali vsa vozlišča in ne bomo ustvarili cikla? Ta problem je v prejšnjem stoletju vzbudil veliko zanimanja znanstvenikov, ki so graf najbolj ugodnih povezav poimenovali minimalno vpeto drevo. Rešitev problema je zelo uporabna v vsakdanjem življenju, kot npr. za iskanje najcenejše povezave hiš v vodovodno ali električno omrežje.

Dva izmed algoritmov, ki rešita problem, smo podrobneje raziskali, pridobljeno znanje pa poskusili uporabiti pri MaRSovsko obarvani nalogi. Sestavili smo povezan graf, kjer vozlišča predstavljajo kraje, iz katerih prihajamo MaRSovci. Želeli smo se povezati v skrivno MaRSovsko omrežje s kabli, zato smo zbrali podatke o razdaljah med posameznimi kraji. Naša naloga je bila, da predlagamo, med katere kraje se splača potegniti kabel.



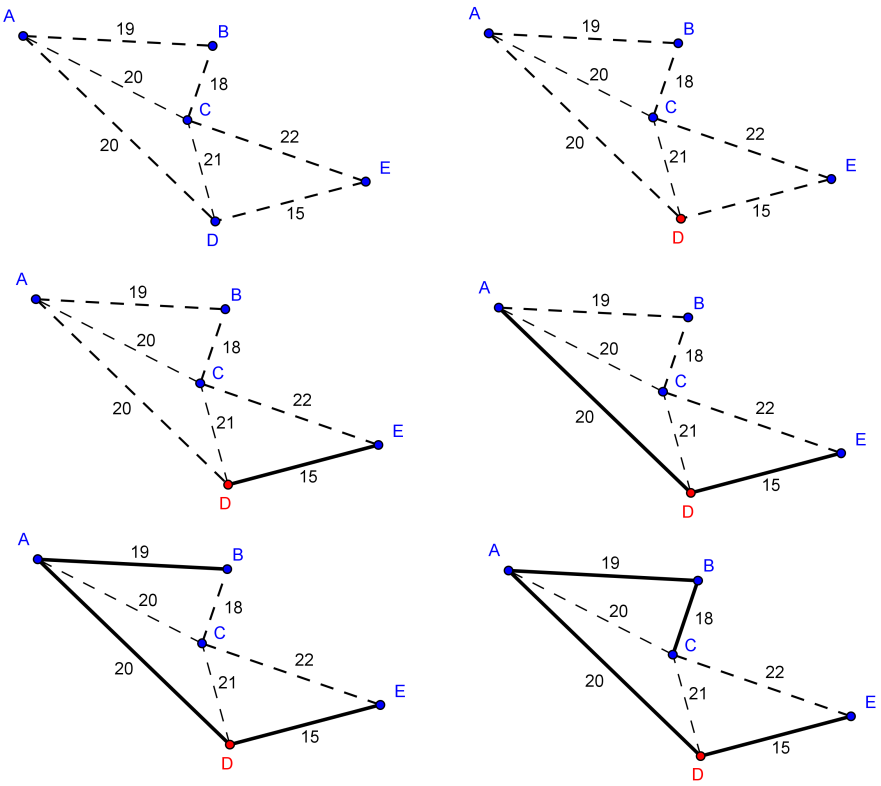
Slika 1: Kraji, iz katerih prihajamo oz. vozlišča grafa.

Primov algoritem

Primov algoritem nam predstavlja zelo zanimiv in enostaven način reševanja problema o minimalnem vpetem drevesu. *Robert C. Prim* je v letih med 1944 ter 1949 delal v laboratoriju *Naval Ordnance*. Tam je skupaj s kolegom *Josephom Kruskalom* razvil dva algoritma za določevanje minimalnega vpetega drevesa. Leta 1959 pa je isti algoritem neodvisno odkril še *Edsger Dijkstra*.

Algoritem deluje po naslednjem principu: Najprej ustvarimo nov graf, v katerega dodamo le eno vozlišče, ta del grafa imenujemo povezan del. Dokler niso vsa vozlišča v povezanem delu grafa, ponavljamo naslednja dva koraka: najdemo najcenejšo povezavo med povezanim in nepovezanim delom, nato pa vozlišče in povezavo iz nepovezanega dela ter pripadajoče vozlišče dodamo v povezan del. Poglejmo si na primeru na sliki 2:

V prvem koraku izberemo vozlišče D. Najcenejša povezava, ki vodi iz njega, je povezava do vozlišča E, zato jo dodamo v povezan del. Nato primerjamo cene povezav iz vozlišč D in E in izberemo najcenejšo med njimi, torej AD. V naslednjih korakih dodamo še povezavi AB in BC. Ker smo povezali vsa vozlišča, je to končna rešitev, ki je zaradi uporabe algoritma tudi minimalna.



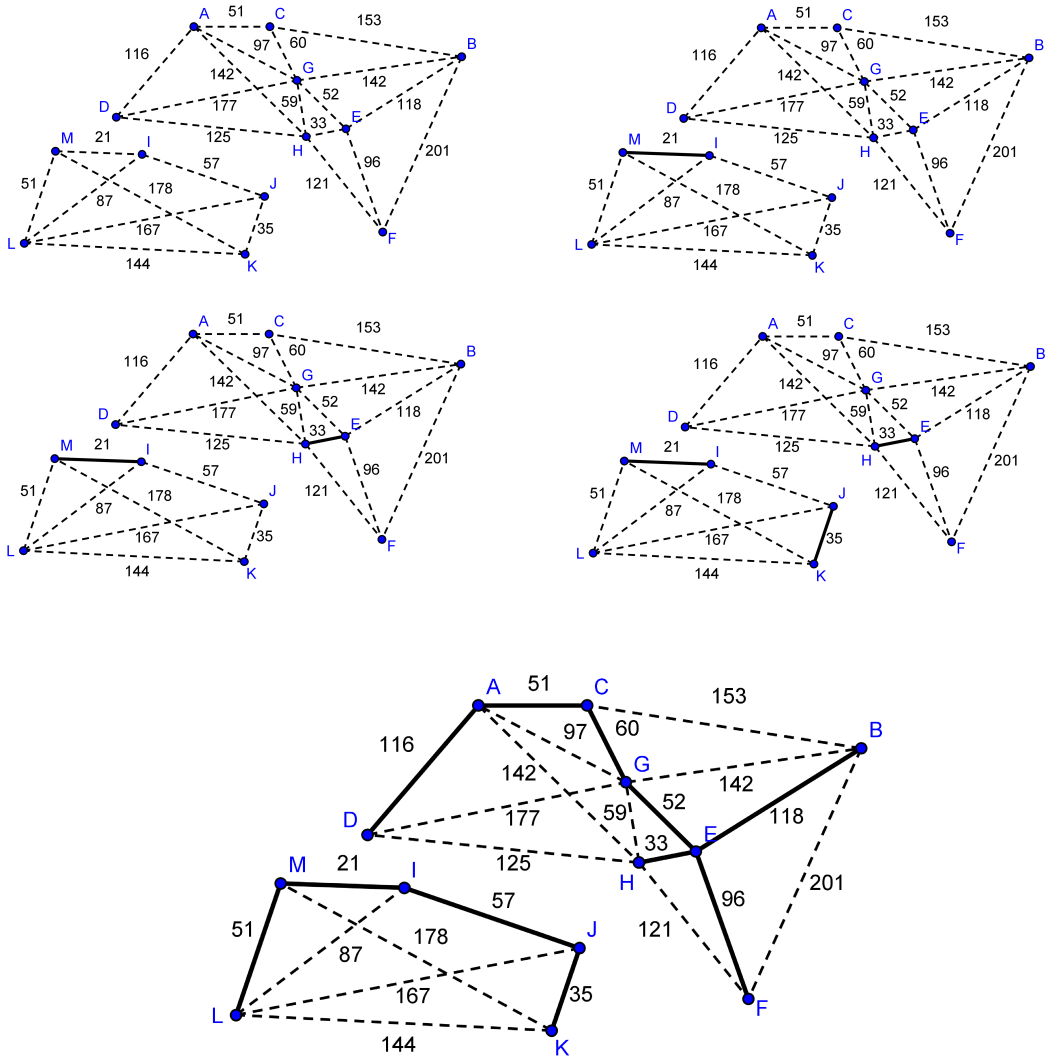
Slika 2: Zaporedje korakov pri reševanju problema s Primovim algoritmom.

Zanimivo je, da za končno rešitev ni pomembno, v katerem vozlišču začnemo. Pri primerih, kjer je veliko enako dragih povezav, se lahko zgodi, da je rešitev več, a so vse najcenejše.

Kruskalov algoritem

Kruskalov algoritem je drugi algoritem za iskanje minimalnega vpetega drevesa. Leta 1956 ga je iznašel in zapisal *Joseph Kruskal*. Ta algoritem za razliko od Primovega, za začetek ne potrebuje naključnega vozlišča in se ne zanaša na prej postavljene povezave, kar pa pomeni da se lahko uporablja tudi na nepovezanih grafih.

Deluje pa takole: najprej povezave razporedimo od najcenejše do najdražje in najcenejši izberemo. Nadaljujemo po vrssti od najcenejše do najdražje povezave in pazimo, da ne ustvarimo cikla. V primeru, da le-ta ne bo nastal, zgradimo povezavo. To ponavljamo, dokler ne dobimo vpetega podgrafa.



Slika 3: Zaporedje prvih treh korakov pri reševanju problema s Kruskalovim algoritmom in končna rešitev.

Opis našega programa in rezultat

Za reševanje MaRSovske naloge smo si pomagali s programom v jeziku C++, v katerem smo naprogramirali Primov algoritem. Vse razdalje smo vnesli v matriko povezav, velikosti $n \times n$, kjer n predstavlja število krajev. V prvo vrstico smo vpisali dolžine povezav od prvega vozlišča do vseh ostalih, v vsaki drugi dolžine povezav z drugim itd. Matrika je simetrična, saj bi kabel med dvema krajema v vsako smer meril (in stal) enako. V mesta, kjer bi se kraj sicer povezal sam s sabo, smo vpisali število, ki je večje od največje razdalje med krajema, da bi program tako povezavo spoznal za neustrezno.

Zaradi enostavnosti program začne v prvi vrstici matrike (ki predstavlja prvi kraj), saj smo ugotovili, da ni pomembno, v katerem kraju začnemo. Med povezavami iz izbere najcenejšo in si zapomni,

v kateri kraj gre. V naslednjem koraku pregleda vrstici obeh krajev in izbere najcenejšo povezavo, ki vodi do novega vozlišča. Nadaljuje s tremi vrsticami, nato štirimi, petimi, šestimi ... vse dokler ne poveže vseh krajev. Kot rezultat izpiše matriko, v katero je vpisoval dolžine ustreznih povezav.

Slika 4: Najugodnejša povezava MaRSovcev, ki smo jo dobili s pomočjo programa.

Zaključek

Pri raziskovanju tega problema nismo omejeni samo na ta dva algoritma, saj poznamo še precej drugih. Seveda pa vsi dajo enako dober rezultat. Tudi pri problemih, ki jih lahko z njimi rešujemo pa se ne konča pri iskanju najcenejšega drevesa, saj lahko grafom dodamo še dodatne zahteve, a jih vseeno lahko pretvorimo na problem minimalnega vpetega drevesa.

Bralcu prepuščamo zanimiv primer: V mestu z n hišami potrebuje vsaka hiša vodo. Izgradnja vodnjaka ob hiši stane V_i evrov, cev med hišama i in j pa stane C_{ij} evrov. Hiša je preskrbljena z vodo, če je ob njej vodnjak ali pa je povezana s hišo ob kateri je vodnjak. Opši algoritem, ki nam pove, kako lahko najceneje preskrbimo vse hiše z vodo.

Viri

[1] *Povezan graf*, `wiki.fmf.uni-lj.si/wiki/Povezan_graf`, citirano dne 22. 8. 2013.

[2] *Minimalno vpeto drevo*, `http://sl.wikipedia.org/wiki/Minimalno_vpeto_drevo`, citirano dne 22. 8. 2013.

[3] *Primov algoritem*, `http://sl.wikipedia.org/wiki/Primov_algoritem`, citirano dne 22. 8. 2013.

[4] *Kruskal's algorithm*, `http://en.wikipedia.org/wiki/Kruskal\%27s_algorithm`, citirano dne 22. 8. 2013.

[5] *Točka (teorija grafov)*, `http://sl.wikipedia.org/wiki/ToÄDka_(teorija_grafov)`, citirano dne 22. 8. 2013.