Vaja 5

Ime in priimek: ..... **Datum:**.....

Model trgovskega potnika

Pri načrtovanju prometnih poti se večkrat srečamo s problemi, kjer moramo izbrati optimalno pot med več mesti. Tipični primeri tovrstnih problemov so načrtovanje voznega reda, dostava dobrin med distribucijskim centrom in prejemniki. Pri tovrstnih problemih imamo omejeno količino prevoznih sredstev, s katerimi moramo dostaviti blago ali ljudi na več krajev, pri čemer stremimo k minimizaciji stroškov (časa, prevožene poti …).
Spoznali bomo, da se vsi tovrstni problemi dajo poenostaviti na t.i. model trgovskega potnika. Potnik mora obiskati neko število krajev in pri tem izbrati takšno pot (zaporedje krajev), da bo prepotoval čim manj kilometrov.

1. Primer: trgovski potnik
Zakaj si tovrstni problemi sploh zaslužijo posebno obravnavo, bomo ugotovili prav na tem primeru. Recimo, da mora trgovski potnik s sedežem v Ljubljani obiskati podružnice v Piranu, Ilirski Bistrici, Sežani, Novi Gorici, Ajdovščini in Postojni. V tem primeru torej nastopa le 6 mest, ki jih mora obiskati.

Koliko različnih razdalj med temi mesti (ne pozabite na sedež) mora potnik poznati, da lahko načrtuje pot?

Na koliko različnih načinov lahko potnik prepotuje pot?

Na koliko različnih načinov bi lahko potnik sestavil pot, če bi moral obiskati 10 mest (n! v Matlabu dobimo s funkcijo `factorial`*)?*

Na koliko različnih načinov bi lahko potnik sestavil pot, če bi moral obiskati 20 mest?

Koliko časa bi potreboval računalnik, da izračuna razdaljo za vse možne kombinacije med 20 mesti, če lahko izračuna razdalje 10^9 kombinacijam na sekundo (hitrost procesiranja 1GHz)?

Metropolisov algoritem

Zaradi hitro naraščajočega števila kombinacij z večanjem števila mest je povsem neuporabno, da bi izračunali razdaljo za vsako izmed teh in nato izbrali tisto z najkrajšo. Obstajajo pa posamezni iterativni algoritmi, ki nam omogočajo, da s statističnim pristopom najdemo t.i. »praktični minimum«. To je kombinacija, ki nima nujno najkrajše razdalje (za slednjo bi morali preizkusiti čisto vse kombinacije), ima pa takšno razdaljo, ki je za praktično uporabo zadosti kratka. Eden od takih statističnih pristopov je Metropolisov algoritem.
Aplikacija tega na problem trgovskega potnika je sledeča. Izhodiščni kombinaciji (zaporedju mest) naredimo majhno spremembo v zaporedju mest. Če ima ta spremenjena kombinacija krajšo razdaljo, potem postane ta nova izhodiščna kombinacija. Če pa je razdalja nove večja, potem naključno izbiramo med staro in novo kombinacijo. Večja kot je razdalja, manjša je verjetnost, da bo nova kombinacija postala novo izhodišče.
Pri žrebanju igra glavno vlogo »referenčna razdalja«. Ta določa, katere spremembe poti so še statistično sprejemljive. Kombinacije, ki bodo daljše od izhodišča za manj kot referenčna razdalja, bodo imele zelo veliko verjetnost, da postanejo nova izhodišča, tiste nad njo pa ne.

2. Primer: Ročno določanje poti

	Ljubljana	Piran	Il. Bistrica	Sežana	N. Gorica	Ajdovščina	Postojna
Ljubljana	-	119	75	80	106	83	52
Piran	119	-	72	58	104	80	72
Il. Bistrica	75	72	-	43	83	59	28
Sežana	80	58	43	-	51	41	32
N. Gorica	106	104	83	51	-	28	60
Ajdovščina	83	80	59	41	28	-	35
Postojna	52	72	28	32	60	35	-

Tabela 1: *Razdalje (v km) med kraji, ki jih mora trgovski potnik prepotovati.*

Na začetku bomo poskušali sami določiti trgovskemu potniku pot potovanja brez pomoči računalnika. S pomočjo tabele 1, poskusite sestaviti najkrajšo pot trgovskemu potniku. Pri tem ne pozabite, da mora začeti in končati svojo pot v Ljubljani.

Zapišite svoje zaporedje mest.

Koliko km bo trgovski potnik prepotoval po vaši poti?

3. Primer: Računalniško določanje poti

Poberite s spleta datoteki `trgovski.m` *in* `metropolis.m`*. ter ju shranite v isti imenik (mapo). Poženite prvo skripto. S tem se bodo naložile nekatere spremenljivke, v katerih se bodo shranile vrednosti razdalj med kraji. Vsakemu kraju bomo priredili po en indeks. Tako bo na primer Ljubljana imela indeks 1, Piran 2, Il. Bistrica 3 … Vsak indeks bo torej predstavljal svoje mesto. Razdalje med temi mesti so podane v spremenljivki* `razdalje`*. Če na primer želimo izvedeti razdaljo med Sežano (indeks 4) in Ajdovščino (indeks 6) vpišemo v Matlabovo konzolo*

```
razdalje(4,6)
```

Če pa vpišemo samo

```
razdalje
```

brez indeksov, nam se bodo izpisale vse razdalje iz tabele 1 v matrični obliki.

Najprej definirajte izhodiščno pot in jo v vektorski obliki zapišite in shranite v vektor `a`*. Ni nujno, da je ta pot blizu optimalne; lahko je povsem naključna.*

Primer:
```
a=[1 7 4 5 3 2 6]
```

Kateri poti ustreza vektor iz zgornjega primera? Navedite zaporedje krajev.

Zapišite, katero izhodiščno pot ste si vi izbrali.

Ko enkrat imamo izhodiščno pot, jo lahko optimiziramo s funkcijo `metropolis`*, ki ima tri argumente. Prvi je izhodiščna pot, drugi je matrika razdalj in tretji referenčna pot. Ker ima naš primer zelo preprosto topologijo, ni nevarnosti, da bi se algoritem zapletel v razne lokalne minimume, zato lahko damo poljubno majhno od 0 večje število (recimo 0.1).*

Vnesite ukaz

```
[a,l]=metropolis(a,razdalje,0.1)
```

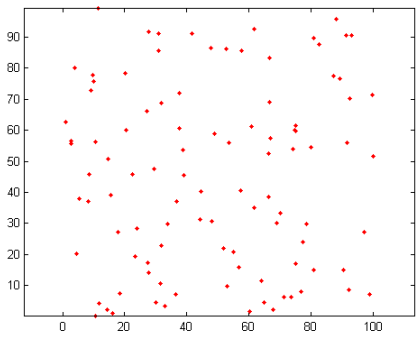
in navedite katero pot je izbral računalnik s pomočjo tega algoritma. V spremenljivko `a` *se bo shranila izračunana pot, v spremenljivko* `l` *pa izračunana razdalja.*

Kakšna je optimalna pot, ki jo je računalnik izbral? Pazite na izhodiščno in končno mesto, ki mora biti v Ljubljani.

Kakšno razdaljo (v km) bi v tem primeru moral prepotovati trgovski potnik?

4. Primer: bolj kompleksna topologija

V našem prejšnjem primeru smo določali optimalno pot za primer zelo preproste topologije (geometrije sosedov). Zato ni bilo potrebe po uvedbi referenčne poti. V tem primeru pa si bomo ogledali malo bolj kompleksen primer na sistemu 100 naključno razpršenih mest. Poberite s spleta datoteki `kompleksna.m` *in* `komp_plot.m`*. ter ju shranite v isti imenik kot prejšnji datoteki. Poženite prvo skripto. Matlab nam bo generiral 100 naključno razpršenih mest na območju 100km×100km in jih vrisal kot točke v graf. Zaradi poenostavitve, bomo privzeli, da so razdalje med mesti kar zračne razdalje (zanemarili bomo strukturo cest).*



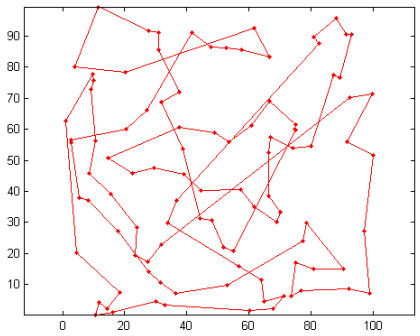
Slika 1: Primer naključno izbranih 100 mest.

Primer tako naključno izbranih mest vidimo na sliki 1. Pri taki razporeditvi mest bi težko ročno izbrali optimalno pot. Zato se bomo iskanja slednje lotili kar z Matlabom. Najprej poskusimo podobno kot v prejšnjem primeru in damo za referenčno razdaljo kar 0,1km.

```
[a,l]=metropolis(a,razdalje,0.1)
```

Vpišite, kakšno optimalno razdaljo dobite.

Dobljeno pot si lahko narišete tako, da v konzolo vtipkate ukaz `komp_plot`.



Slika 2: Primer izračunane optimalne poti z direktno nastavljeno referenčno razdaljo (0,1km).

Poskusite določiti optimalno pot tako, da začnete pri večji referenčni razdalji in jo postopoma zmanjšujete. Vnesite postopoma sledeče referenčne razdalje: 20km, 15km, 10km, 7km, 5km, 3km, 2km, 1,5km, 1km, 700m, 500m, 300m, 200m in 100m. Sproti si zapisujte kakšno pot dobite na posameznem koraku.

S(20km)=

S(15km)=

S(10km)=

S(7km)=

S(5km)=

S(3km)=

S(2km)=

S(1500m)=

S(1000m)=

S(700m)=

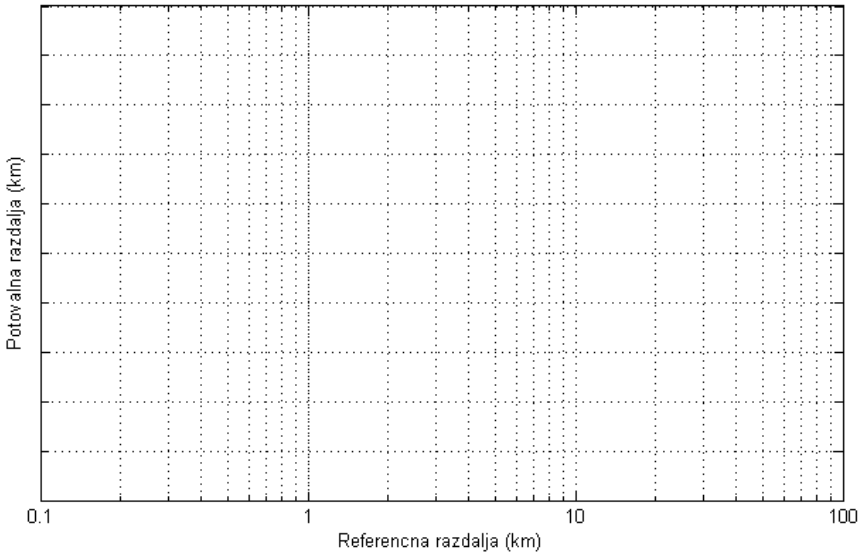
S(500m)=

S(300m)=

S(200m)=

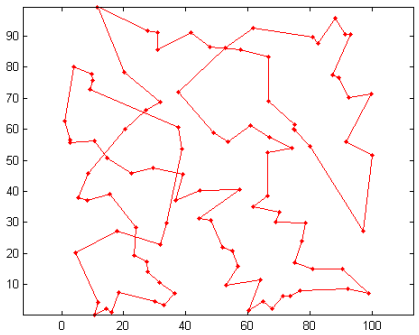
S(100m)=

V spodnji graf narišite odvisnost potovalne razdalje od referenčne razdalje.



Izračunajte, za kolikšen delež smo zmanjšali potovalno pot, s tem ko smo postopoma zmanjševali referenčno razdaljo (v primerjavi z direktnim izračunom).

Izrišite si potovalno pot (ukaz `komp_plot`*).*



Slika 3: Izračunana optimalna pot s postopnim zmanjševanjem referenčne razdalje.

Nekaj inteligentnega za na konec

V praksi malo kdaj iščemo najkrajšo pot. Namesto tega pot optimiziramo tako, da jo prepotujemo v najkrajšem času, ali s čim manj stroški. Opišite, kaj bi morali spremeniti v našem algoritmu, da bi namesto najkrajše poti izračunali najcenejšo.

Reference:

http://en.wikipedia.org/wiki/Travelling_salesman_problem

Zanimivost:
Klub računalniški kompleksnosti, so človeški možgani presenetljivo dobri pri določanju najkrajše poti. Preizkusite se lahko na sledeči strani.

http://www.tsp.gatech.edu/games/tspOnePlayer.html