Univerza v Ljubljani
Fakulteta za matematiko in fiziko

Finančni praktikum

Stable roommate problem

Timotej Giacomelli in Nejc Duščak

Mentorja: prof. dr. Sergio Cabello, asist. dr. Janoš Vidali

Ljubljana, 2020

Kazalo

1 Uvod

V projektu pri finančnem praktikumu bova obravnavala *Stable roommate problem*. Problem bova modelirala in poganjala ekspiremente v programskem jeziku Sage.

Stable roommate problem, znan tudi kot kratica **SR**, je eden izmed *stable matching* problemov, katere sta prvič predstavila David Gale in Lloyd Shapely. Problem je dobil ime zaradi svoje praktične uporabe - kako razporediti ljudi v dvoposteljne sobe, glede na njihove preference.

Problem je sestavljen iz $2n$ "udeležencev", kjer ima vsak udeleženec seznam preferenc s $2n-1$ elementi, torej po eno vrednost za vsakega soudeleženca. Vsak udeleženec predstavljen točko v metričnem prostoru, njegov seznam pa so urejene dolžine do ostalih soudeležencev.

Ujemanje je množica n disjunktnih parov udeležencev. Za ujemanje M je par $\{m_1, m_1'\} \notin M$ *blocking pair*, če zadošča naslednjim pogojem:

- $\{m_1, m_1'\}, \{m_2, m_2'\} \in M$,
- m_1 preferira m_2 bolj kot m_1',
- m_2 preferira m_1 bolj kot m_2'.

Oziroma če povemo z besedami, *blockin pair* nastane, če se imata vsaj dva udeleženca, ki nista v paru, pri ujemanju raje, kot s svojim partnerjem. Ujemanje M je nestabilno, če zanj obstaja *blocking pair*. Drugače je ujemanje M stabilno.

Cilj SR je najti stabilno ujemanje ali pokazati, da nobeno ne obstaja. S časoma so uspeli razviti algoritem s časovno zahtevnostjo $O(n^2)$, ki bodisi najde stabilno ujemanje, bodisi ugotovi, da za dani primer ne obstaja nobeno stabilno ujemanje.

Stable roommate problem je v splošnem lahko uporabljen za ujemanje opazovanj in objektov pri nalogi razvrščanja. Na primer v življenjskem primeru iskanje primernega sostanovalca, so lahko le-ti predstavljeni po točkah v nekem prostoru lastnosti: koordinatna os prostora je lahko najprimernejši čas za spanje, želena raven urejenosti prostora, število zabav/piv na semester, itd.. Povsem logično je sklepati, da bo izbran udeleženec tisti, ki bo imel podobne lastnosti.

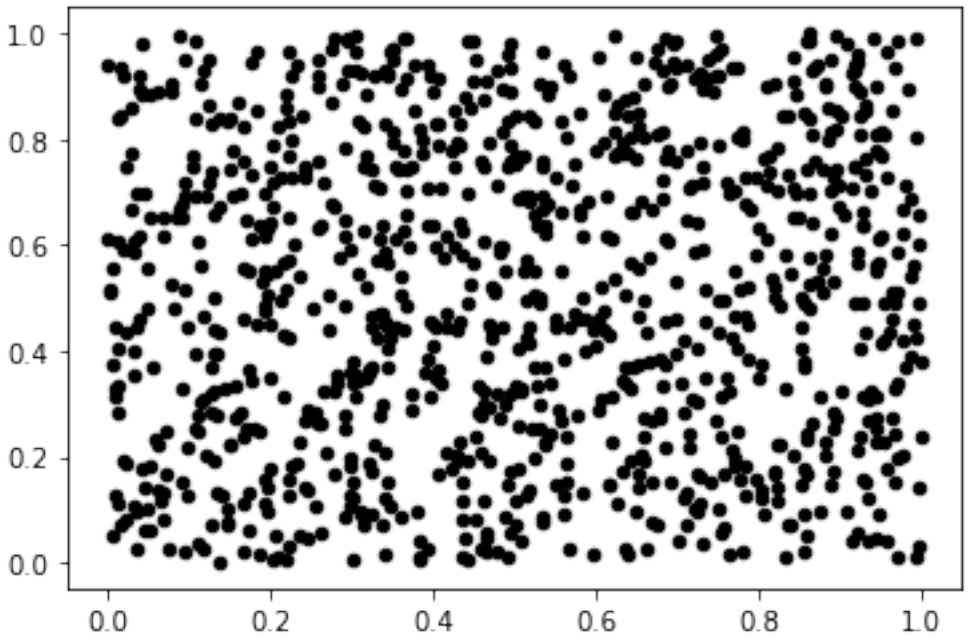
V nadaljevanju projekta bova sledeila sledečim korakom:

- generirala bova $2n$ naključnih točk v kvadratu velikosti 1x1,
- izračunala bova posamezne razdalje med točkami,
- razdalje bodo predstavljale najine preference (manjša razdalja je večja preferenca), ki jih bova uredila po velikosti,
- napisala bova algoritem, ki bo izračunal ujemanje ali pa ugotovil, da ujemanje ne obstaja,
- analizirala bova, ali se seštevek razdalj med točkami v paru povečuje, ali zmanjšuje, ko povečujeva število točk (n).

2 Opis problema

Stable roomate problem je problem v katerem želimo poiskati stabilno ujemanje udeležencev v problemu. Vsak udeleženec ima seznam preferenc do drugih udeležencev, ki je urejen po velikosti od najvišje preference pa vse do najnižje. Če bi želeli problem predstaviti v realnosti, si lahko mislimo, da imamo posameznika, ki si želi poiskati sostanovalca. Glede na njegove želje bo potencialne sostanovalce uredil glede na svoje preference in nato izbral posameznika z najvišjo preferenco.

V projektu sva obravnavala enostavnejšo verzijo osnovnega problema, in sicer *Geometric stable roomate problem*. V tem problemu so udeleženci predstavljeni kot točke v ravnini, kjer lahko te točke izberemo iz poljubno omejenega območja ali pa jih izbiramo kar iz celotne ravnine. V najinem projektu sva na začetku točke izbirala iz kvadrata velikosti 1x1. Preference so tukaj zelo preproste, saj so podane le kot razdalja do točk, kjer manjša razadlja do točke pomeni večjo preferenco. V tem problemu želiva poiskati stabilne pare, ki ustrezajo seznamu preferenc.



3 Glavne ideje problema in psevdokoda

Točke v kvadratu sva določila tako, da sva generirala x in y koordinato posamezne točke in jih shranila v *slovar točk*, kjer ključ slovarja predstavljajo ime točke (npr. *tocka_0*), vrednost pa koordinate točke. Vrednosti slovarja so oblike (*x,y*). V nadaljevanju projekta bova poskusila točke izbirati tudi iz območij drugačne oblike, če bo le možno tudi točke iz neomejenega območja.

Razdalje med točkami sva izračunala s pomočjo Pitagorovega izreka. S funkcijo *for* sva se sprehodila čez vse ključe v *slovarju točk* in oblikovala novi slovar imenovan *slovar razdalj*, v katerem je ključ slovarja ponovno ime točke, vrednosti pa vsebujejo seznam razdalj do posameznih točk. Prvotno sva računanje Pitagorovega izreka razdelila na 3 podprimere. Kasneje sva našla hitrejši način, ki za izračun razdalje porabi manj časa.
V nadaljevanju sva s pomočjo funkcije *preference* uredila *slovar razdalj* glede na razdaljo. Tako sva dobila seznam tulpov, urejen po velikosti od najkrajše do najdaljše razdalje.

Do sedaj sva zgenerirala slovar v katerem imava vse točke in urejen seznam preferenc. V tem delu projekta pa nastopi reševanje problema. Ideja reševanja problema je bila, da bi se sprehodila čez vse točke v slovarju in pri vsaki točki preverila njene preference. Ko bi našla najkrajšo razdaljo med dvema točkama, bi ti dve točki povezala v par in posledično na koncu pridobila ujemajoče pare. Algoritem za reševanje te ideje pa sva sestavila iz dveh funkcij:

- najprej sva sestavila funkcijo *Najkrajsa*, ki poišče najkrajšo razdaljo med vsemi točkami. Sprehodimo se po vseh točkah in preverimo njihove prve preference. Ko najdemo najmanjšo oziroma najkrajšo razdaljo, funkcija izbere tisti dve točki, med katerima ta razdalja nastopi in ju poveže v par. Na koncu nam vrne par točk, s pripadajočo najmnajšo razdaljo,
- druga funkcija *Vsi_pari*, pa nam omogoča, da kličemo funkcijo *Najkrajsa* dokler ne povežemo vse točke v paru. Pomembno je tudi omeniti, da ko najdemo povezavo med dvema točkamo, potem ti dve točki zbrišemo tako iz slovarja, kot tudi iz seznama preferenc drugih točk, saj tako dosežemo, da se funkcija hitreje konča.

4 Robni pogoji

V procesu preverjanja pravilnosti algoritma, sva definirala tri "robne primere", v katerihbi lahko prišlo do težav z algoritmom. Pomembno je omeniti, da so vsi primeri narejeni le za točke, ki so generirane v kvadratu. Ti primeri so:

- Točke so razporejene v kvadrat:

  primer_1 = 'tocka_0': (0.1, 0.1), 'tocka_1': (0.2, 0.1), 'tocka_2': (0.1, 0.2), 'tocka_3': (0.2, 0.2)

- Vse točke ležijo na vodoravni premici, kjer je razdalja od levega najbližjega soseda (če obstaja) enaka razdalji do desnega soseda (če obstaja):

  primer_2 = 'tocka_0': (0.1, 0.1), 'tocka_1': (0.2, 0.1), 'tocka_2': (0.3, 0.1), 'tocka_3': (0.4, 0.1), 'tocka_4': (0.5, 0.1), 'tocka_5': (0.6, 0.1), 'tocka_6': (0.7, 0.1), 'tocka_7': (0.8, 0.1), 'tocka_8': (0.9, 0.1), 'tocka_9': (1, 0.1)

- Razširjen primer_1:

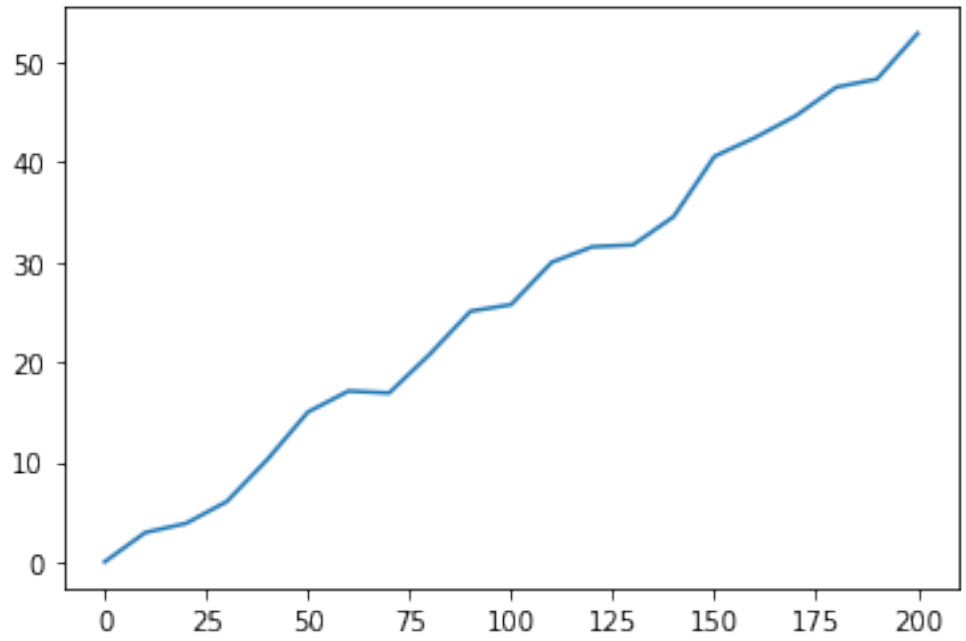
  primer_3 = 'tocka_0': (0, 0), 'tocka_1': (0, 1/3), 'tocka_2': (0, 2/3), 'tocka_3': (0, 1), 'tocka_4': (1/3, 0), 'tocka_5': (1/3, 1/3), 'tocka_6': (1/3, 2/3), 'tocka_7': (1/3, 1), 'tocka_8': (2/3, 0), 'tocka_9': (2/3, 1/3), 'tocka_10': (2/3, 2/3), 'tocka_11': (2/3, 1)

5 Eksperimentiranje z algoritmom

Za eksperimentiranje z najinim algoritmom sva si izbrala računanje vsote razdalj med točkami v izbranem območju in preverila kako se vsota obnaša glede na število izbranih točk in glede na izbrano območje. Za ta namen sva sestavila funkciji *vsota_razdalj* in *razlicne_vsote*. Funkcija *vsota_razdalj* sešteje razdalje med točkami, ki so povezane v par. Funkcijo *razlicne_vsote* pa sva sestavila z namenom, da opazujeva kako se obnašajo vsote, ko povečujemo število točk.

5.1 Eksperimentiranje v kvadratu velikosti 1x1

Pri opazovanju gibanja števila vsote sva dobila spodnjo funkcijo.

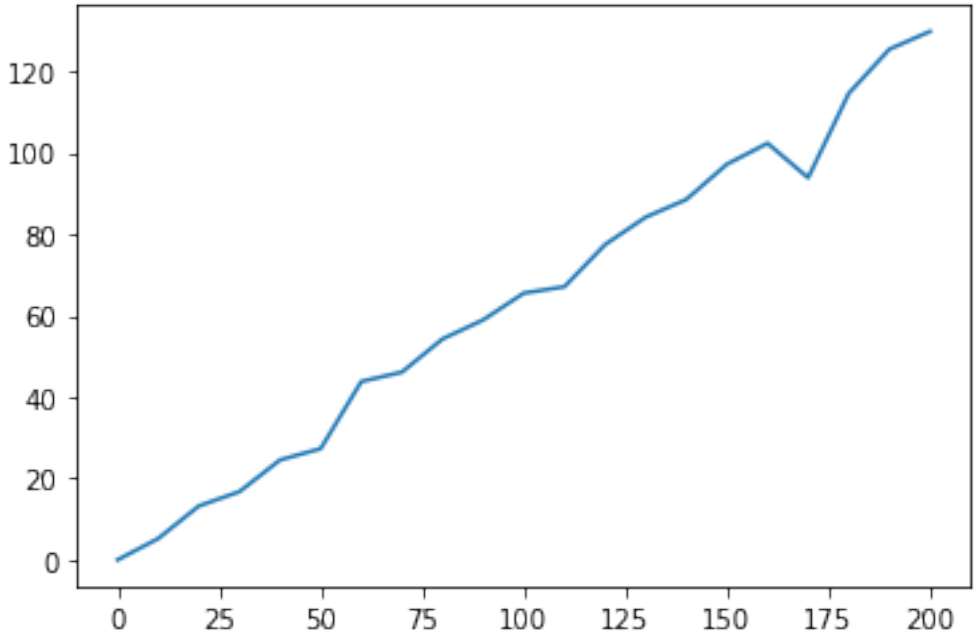


Opazimo lahko, da vsote naraščajo skupaj s številom točk. Seveda v nekaterih primerih prihaja do manjših odstopanj, vendar je opazno, da vsote rastejo.

5.2 Eksperimentiranje v krogu s polmerom r

V drugem primeru sva definirala točke znotraj kroga s polmerom r, kjer sva polmer r izbrala na začetku. Tu lahko eksperimentiramo na več različnih

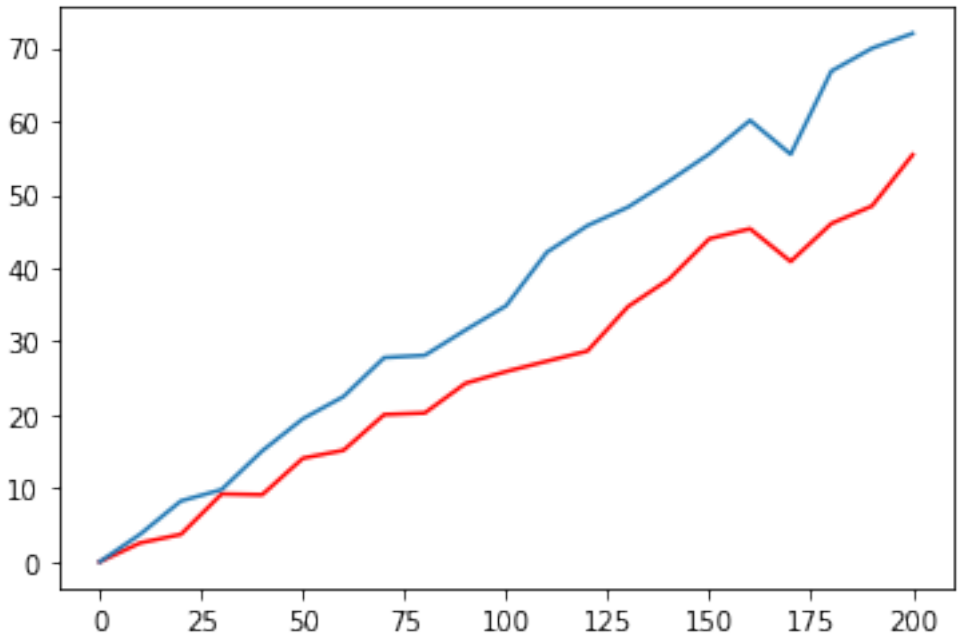
načinov. Na primer povečujemo število izbranih točk pri dani izbiri r=1.



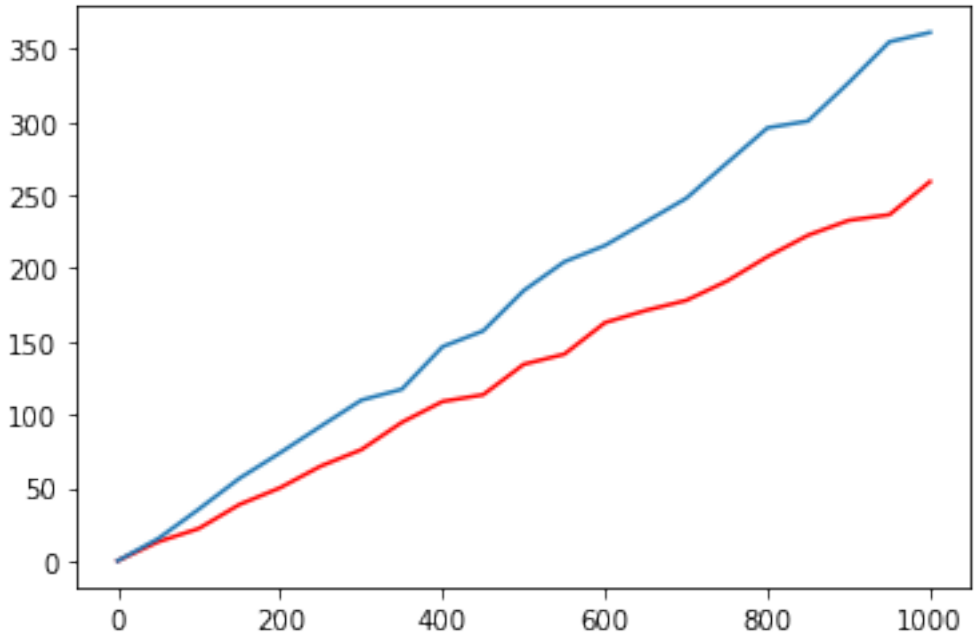
Opazimo, da tudi v tem primeru vsote rastejo enakomerno z višanjem števila izbranih točk.

5.3 Eksperimentiranje s kvadratom in krogom

V tem razdelku sva opazovala, kako različna območja (v najinem primeru kvadrat in krog), ki imajo enako površino, vplivajo na vsoto razdalj vseh točk v paru. Za ta primer sva si izbrala kvadrat velikosti 1x1, na drugi strani pa sva izbrala krog s polmerom r=1/sqrt(pi). Tako sva dobila območji, ki imata enako ploščino.
Na sliki so z rdečo barvo označene vsote točk, ki ležijo v kvadratu, z modro pa točke, ki ležijo v krogu.

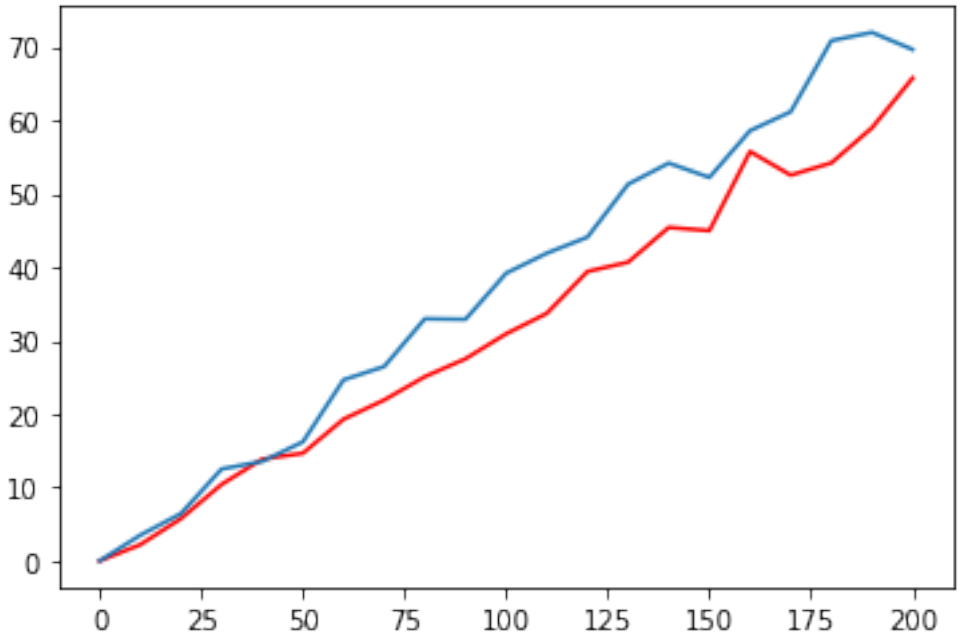


Opazimo lahko, da so vsote večje v krogih kot v kvadratih.
Če pa pogledamo še spodnjo sliko pa lahko še bolj nazorno vidimo, da vsote točk v krogu tudi hitreje naraščajo.



5.4 Eksperimentiranje s kocko in sfero

Za konec pa sva pogledala še območja oblike v treh dimentzijah (3D). Najprej sva se lotila s kocko z robom dolžine 1. Videla sva, da se tudi v tem primeru vsote povečujejo skupaj z večanjem števila točk.
Enako sva ponovila na sferi z enakim volumnom kot pri kocki. Gibanje vsot za oba primera je prikazano na spodnji sliki, kjer rdeča črta označuje vsote razdalj v kocki, modra pa vsote v sferi.



Vidimo, da so tudi v treh dimenzijah vsote razdalj, med točkami v paru, v sferi večje od vsot razdalj v kocke.