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. Seznam 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 pa 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 točke izbiramo kar iz celotne ravnine. V najinem projektu sva točke izbrala iz kvadrata velikosti 1x1. Preference so tukaj zelo preproste, saj so podane le kot razdalja do točk. Tako manjša razadlja do točke pomeni večjo preferenco. V tem problemu želiva poiskati stabilne pare, ki ustrezajo seznamu preferenc.

3 Glavne ideje problema in psevdokoda

Za iskanje rešitev problema sva si izbrala poljubne točke znotraj kvadrata velikosti 1x1. Generirala sva x in y koordinato posamezne točke in jih shranila v *slovar točk*, kjer ključ slovarja predstavljajo ime točke (naprimer *tocka_0*), vrednost pa koordinata točke. Vrednost slovarja je oblike (*x,y*) kar imenujemo tupli. V nadaljevanju projekta bova poskusila točke izbirati tudi iz območij drugačne oblike, če bo le možno tudi točke iz neomejenega območja.

Nato sva izračunala razdalje med točkami 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, vrednost pa vsebuje seznam razdalj do posamezne točke, torej dobimo seznam tulpov. 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:

- prvo 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 pa se imenuje *Vsi_pari*, ki 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 izbrišemo iz seznama preferenc drugih točk, saj tako dosežemo, da se funkcija hitreje konča.

4 Reševanje problema izenačenja

Pri pregledu pravilnosti reševanja problema sva ob koncu naletela na problem izenačenja. Nastopi, ko pride do pojava, ko ima poljubna točka enako razdaljo do dveh drugih točk v območju. To pomeni, da ima enako preferenco do dveh točk. Najin algoritem pri uporabi funkcije *preference* uredi razdaljo po velikosti, če pa se pojavi problem izenačenja pa kot prvo točko navede tisto, ki je prva ko se funkcija sprehodi skozi ključe *slovarja razdalj*. Zato kasneje, ko pridemo do reševanja problema s funkcijama *Najkrajsa* in *Vsi_pari* naletimo na problem, ko funkciji pogledata le prvo preferenco, ne upoštevata pa možnost izenačenja. Tako sva bila primorana algoritem prilagoditi tudi za primer izenačenja.

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 točk in glede na območje, ki sva ga izbrala.