Univerza v Ljubljani
Fakulteta za matematiko in fiziko

THE POWER OF ADJACENT CHOICES

Finančni praktikum

Avtorja:
Justin Raišp, Maša Orelj

Ljubljana, 2022

1 Navodilo

Imamo n žog in n košev $b_1, \ldots, b_n$, ki so prazni. Koši so postavljeni v krogu: koš b_i ima soseda b_{i-1} in b_{i+1}, kjer zaradi krožnosti velja, da sta b_1 in b_n soseda. Opazujemo naslednji naključen proces: Za vsako žogo naključno izberemo koš b_i, pogledamo še oba soseda in damo žogo v koš z najmanjšim številom žog v tistem trenutku izmed teh treh. Zanima nas število žog v košu z največ žogami na koncu procesa. Ta proces potem razširimo na več načinov:

- za sosede štejemo koše, ki so na razdaljah največ $2, 3, \ldots,$
- za n košev vzamemo $2n, 3n, 4n, \ldots$ žog,
- iščemo koš z najmanjšim številom žog.

Opazujemo lahko tudi dvodimenzionalno mrežo košev s topologijo torusa. Torej imamo npr. n^2 košev $b_{i,j}$, kjer sta $i, j \in [n]$, kjer za soseda $b_{i,j}$ in $b_{k,l}$ velja $|i-k| + |j-l| = 1$. Soseda sta tudi $b_{1,i}$ in $b_{n,i}$ ter $b_{i,1}$ in $b_{i,n}$. Podobno lahko gledamo tudi tridimenzionalno verzijo.

2 Program za reševanje problema

Za reševanje opisanega problema sva uporabila programski jezik *Python*. Začela sva s pisanjem ustreznega programa za reševanje naloge v eni dimenziji, pri katerem je možno spreminjanje števila sosedov(razdalje), količine žog in količine košev. Pri tem sva privzela, da v primeru, ko je košev z najmanjšim številom žog med pregledanimi koši več, položimo žogo v koš z najmanjšim indeksom.

```
#ISKANJE SOSEDOV
def najdi_sosede_1d(kosi,izbrana_kosarica, razdalja=1):
sosedi = {}
st_kosev = len(kosi)
for j in range((-razdalja),(razdalja+1)):
    najdi indeks soseda v listu s pomocjo modula, da velja kroznost
    v slovar sosedov dodaj njegov indeks kot kljuc in njegovo stevilo zog kot vre
izmed sosedov poisi tistega z minimalno vrednost
return sosedi, minimalna_vrednost
```

```
#ISKANJE MAKSIMALNEGA STEVILA ZOG
def maksimalno_stevilo_zogic(st_zogic, st_kosev, razdalja=1):
zacetek merjenja casa
kosi = [0]*st_kosev
for i in range(st_zogic):
    nakljucno izberemo kosarico
    najdi_sosede_1d(kosi, izbrana_kosarica, razdalja) #s pomozno funkcijo doloci
    poisci vse kandidate, ki so sosedi in imajo minimalno vrednost
    izmed kandidatov izberi tistega z najnizjim indeksom
    kosi[minimalni_sosed] += 1
poisci maksimalno stevilo zog v kosu
prestej stevilo kosev z maksimalnim stevilom zog
izracunaj delez kosev z maksimalnim stevilom zog
konec merjenja casa
izracunaj casovno zahtevnost algoritma
return maksimalno_stevilo_zog, casovna_zahtevnost, delez_kosev
```

Funkcija *maksimalno_stevilo_zogic* torej sprejme izbrano število žog, število košar in razdalja, vrne pa vrednost maksimalnega število žog v košari, potreben čas za izvedbo funkcije in delež košar z maksimalnim številom žog.

Program sva prilagodila tudi na dvodimenzionalno mrežo košev, ki deluje na podoben način.

3 Analiza rezultatov