# Poročilo projekta - Voronoijevi diagrami

Anže Mramor in Peter Tiselj

4.12.2020

## Poročilo projekta - Voronoijevi diagrami

### Uvod - Definicija Vornoijevih diagramov

Voronoijeve diagrame definiramo na ravnini ali na splošnem metričnem prostoru. V danem grafu $G$ in podmnožici vozlišč $U$ ($U \subseteq G$) razdelimo vozlišča grafa $G$, glede na to, katero vozlišče iz $U$ jim je najbližje. Bolj natančno: za $\forall u \in U$, definiramo Voronoijevo celico, $(u, U)$, kot množico vseh vozlišč iz $G$, ki so bližje $u$ kot kateremu koli drugemu vozlišu $U \setminus \{u\}$.\ Oziroma s formalno definicijo:\ Naj bo $X$ metrični prostor z razdaljo $d$, ki je definirana kot razdalja med točko $x \in X$ in podmnožico $A$ : $d(x, A) = inf\{d(x, A)|a \in A\}$ . Naj bo še $K$ množica indeksov in $(P_k)_{k \in K}$ terica (tuple) nepraznih podmnožic v $X$. Potem definiramo Voronoijevo celico kot $R_k$ za $k \in K$ kot

$$R_k = \{x \in X | d(x, P_k) \leq d(x, P_j); \forall j \neq k\}$$

Voronoijev diagram je enostavno terica Voronoijevih celic $(R_k)_{k \in K}$

### Opis problema

Pogledala sva si enostavna omrežja: 1. dvo- in trodeminzionalne mreže različnih velikosti - $1 \times n$, $2 \times n$, $n \times n$, ... 2. različne tipe grafov kot so binarna drevesa in grafi s cikli

Za vsako omrežje sva si izbrala več smiselnih vrednosti $k$, ki so nama predstavljali število naključno izbranih vozlišč, za katere sva izračunala Voronoijeve celice. Nato sva statistično obdelovala velikosti dobljenih Voronoijevih celic. Pogledala sva povprečno število vozlišč v minimalni, maksimalni in povprečni Voronoijevi celici za vsako omrežje in vse izbrane $k$, poskušala iz grafov z manjšim številom vozlišč napovedati kakšno bo število vozlišč v celicah pri grafih z večjim številom vozlišč, ter poskušala ugotoviti kakšna je porazdelitev velikosti celic glede na izbrani $k$ (Peter dopolni primerno kar si delu pls).
Za vsak izbrani $k$ sva program pognala večkrat, saj sva tako lahko opazovala kako se pri različnem izboru začetnih vozlišč, spreminjajo zgornje vrednosti. Rezultate sva shranjevala v *.tsv* datoteke in jih nato obdelala v programu *r*. Ker so tabele za najin problem zelo velike, bova rezultate predstavila predvsem v obliki grafov.

### Glavna ideja programa / psevdokoda

Problema sva se lotila tako, da sva sprva spisala program za generiranje Voronoijevih celic iz grafa in šele nato kodo za generiranje želenih grafov, v nadaljevanju pa bo program predstavljen v bolj logičnem vrstnem redu.

#### Generiranje grafov

Končni cilj tega dela programa je vrniti matriko sosednosti za želen tip grafa. Za vse grafe sva po predlogu prof. Vidalija skonstruirala razred Graf, ter nato še poseben razred za vsak tip grafa posebej. Razredi omogočajo generiranje različnih matrik sosednosti za vsak tip grafa, ter poenostavijo genriranje Voronoijevih celic za vsak tip, saj omogočajo enostavno klicanje za vsak tip, ter različna števila vozlišč grafov.
Poleg tega vsakemu izmed razredov pripada posamezna funkcija, s pomočjo katere se skonstruira primerna matrika sosednosti za dan tip grafa. Te funkcije sva skonstruirala za vsak tip grafa, glede na njihove zakonitosti

npr. v 2D mrežah ima vsako vozlišče lahko največ 4 sosednja vozlišča, v binomskih drevesih pa največ 3. 3D mreže so skonstruirane na podoben način kot 2D mreže, le v treh dimenzijah, cikli pa se generirajo iz binomskih dreves z dodajanjem povezav. Natančnejša razlaga kako sva konstruirala posamezen tip grafa je na voljo v komentarjih ob kodi v programu samem.

#### Konstruiranje Voronoijevih celic za posamezen graf

Ko enkrat imamo program, ki nam skonstruira matriko sosednosti za poljuben tip grafa določen z dimenzijami ali številom vzolšič, lahko zgeneriramo Voronoijev diagram za ta graf. Za to najprej potrebujemo razdalje med vsemi vozlišči znotraj grafa. Za to sva uporabila znan Floyd-Warshallov algoritem, ki iz matrike sosednosti vrne najkrajše poti med vsemi pari vozlišč. Nato pa te vrednosti uporabi glavna funkcija programa, Voronoi, ki kot podatke vzame matriko sosednosti želenega grafa, množico U vseh središč Voronoijevih celic, ter tip grafa (da vemo za kateri tip grafa iščemo diagram), ter vrne seznam seznamov. V vsakem od notranjih seznamov je prvi element središče Voronoijeve celice, sledijo pa vsa ustrezna vozlišča.
Funkcija deluje tako, da za graf s pomočjo Floyd-Warshalla določi najkrajše razdalje vseh vozlišč grafa do izbranih središč celic. Nato po stolpcih dobljene matrike primerja vse razdalje med seboj in poišče najkrajšo, ki jo potem doda v ustrezen seznam (k pripadajočemu središču). Če je neko vozlišče enako oddaljeno do več središč, ga doda na seznam k vsem središčem. Funkcija nato zapiše končni seznam seznamov v obliki *data frame*-a.

#### Generiranje rezultatov

Na koncu sva sestavila funkcijo, ki nam generira Voronoijeve diagrame za določen tip in število vozlišč, ter jih zapisuje v tekstovno datoteko. Odločila sva se, da želiva preveriti diagrame za čim več različih vrednosti $k$, saj bi tako dobila najboljše primerjave za grafe. Vendar se je hitro izkazalo, da bo program precej časovno zahteven, zato sva se odločila, da si bova ogledala le vrednosti $k$ med 5 in 60 odstotki vseh vozlišč grafa, saj sva tam pričakovala najbolj zanimivo dogajanje. Za recimo grafe s 100 vozlišči sva torej vzela $k = 5, 6, 7, ..., 60$. Odločila sva se tudi, da bi bilo smiselno generiranje za vsak graf večkrat ponoviti. Ker je bilo spet očitno, da bo za velike grafe to lahko časnovno zelo dolgotrajno, sva se odločila, da bova za grafe z največ 100 vozlišči generirala 50 vzorcev, za večje pa le 20. Za primerjavo - za konstruiranje diagrama s 100 vozlišči je program potreboval povprečno 22 sekund, za 200 vozlišč povprečno 36 sekund, za 500 vozlišč pa že približno 25 minut, iz česar lahko razberemo, da je časovna zahtevnost programa zelo eksponentna.
Vse rezultate za en tip grafa s stalnim številom vozlišč sva shranila v isto tekstovno datoteko, ter na koncu programa dodala še funkcijo main, s katero sva lahko generirala diagrame za več različnih tipov in števil vozčišč hkrati.

## Statistično modeliranje v programu r

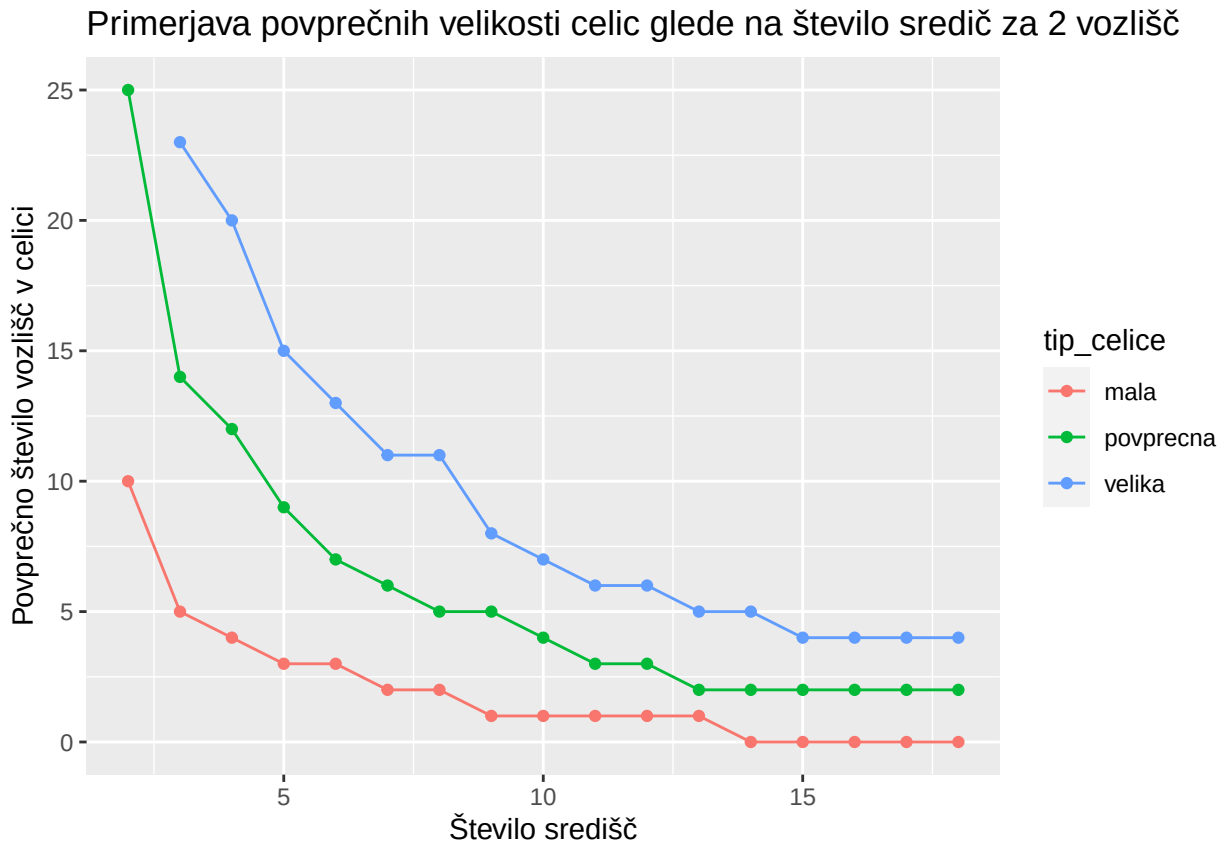
#### Obdelava podatkov v programu r

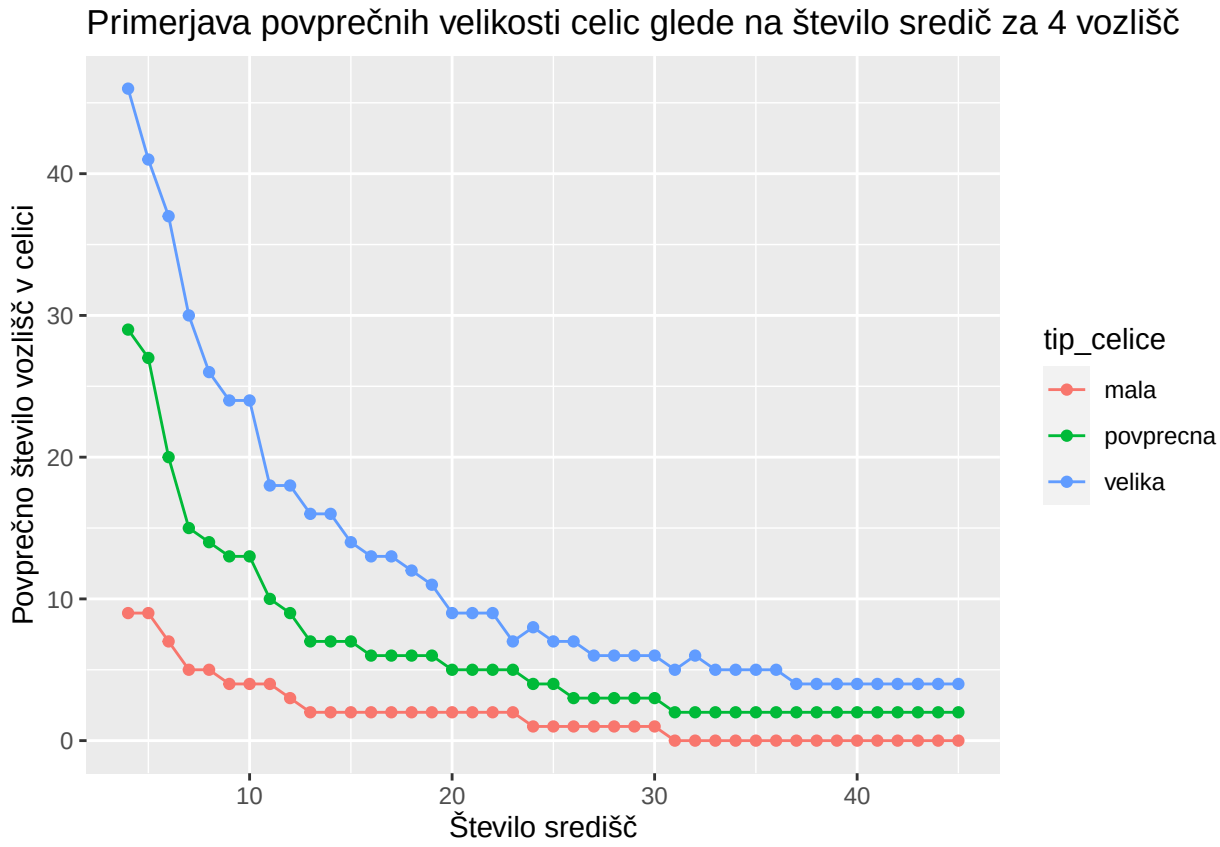
Podakte sva uvozila v program r in jih statistično obdelala. Najprej sva spisala funkcijo, ki je vsaki celici priredila status velike, povprečne ali male, glede na število elementov v njej. Najprej sva preračunala povprečno velikost vseh celic, glede na število središč. Kot velike celice sva nato nastavila tiste, ki so bile od povprečnih večje za faktor 1.9, kot male pa tiste, ki so bile manjše za faktor 0.1. Nato sva za vsak tip celice preračunala povprečno velikost in jih prikazala na grafu, za vsako število vozlišč posebej. V tem sklopu sva naredila še primerjavo povprečnih velikosti celic glede na število središč in vozlišč.
(za tu spiši kar si ti svojega obdeloval)

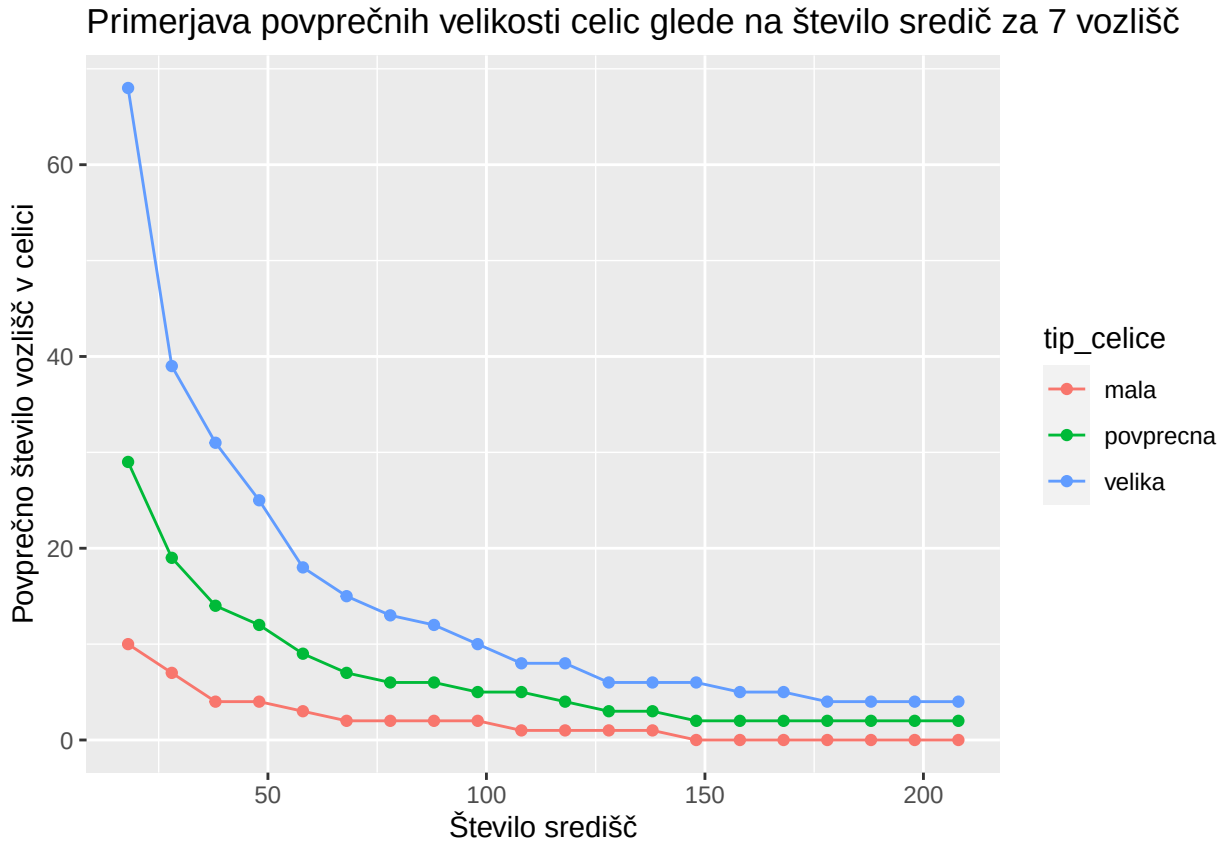
Za konec sva iz rezultatov poskušala napovedati, kako bi izgledali rezultati pri grafih z več vozlišči.

#### Rezultati

#### Binomska drevesa in cikli


Primerjava povprečnih velikosti celic glede na število sredič za 2 vozlišč
Povprečno število vozlišč v celici
25
20
15
10
5
0
5
10
15
Število središč
tip_celice
mala
povprecna
velika

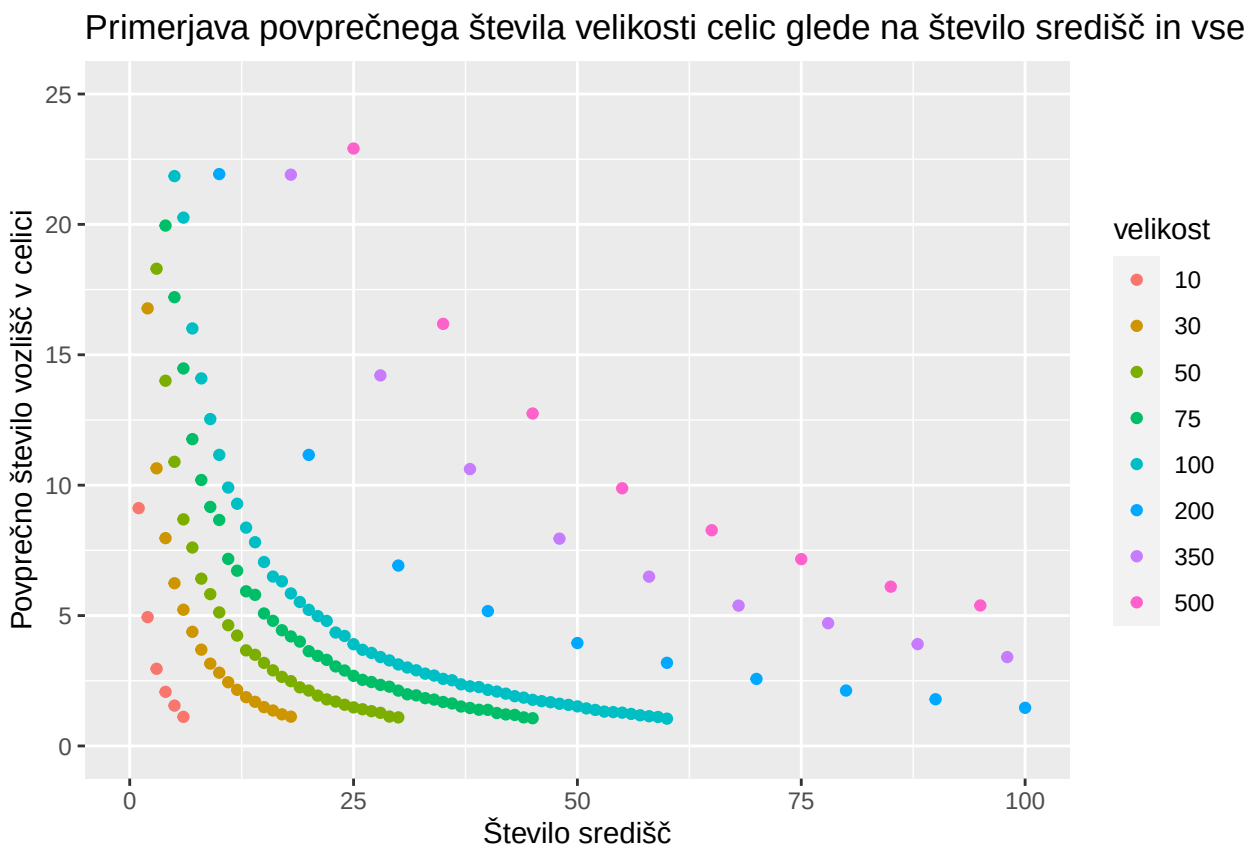

Primerjava povprečnih velikosti celic glede na število sredič za 4 vozlišč
Povprečno število vozlišč v celici
40
30
20
10
0
10
20
30
40
Število središč
tip_celice
mala
povprecna
velika




Kot je zelo lepo razvidno iz zgornjih grafov, se za vse 3 (in tudi vse neprikazane grafe) pojavlja isti vzorec. Ko je število središč majhno, so povprečne velikosti celice občutno večje kot ob velikih številih središč. To je seveda logično, saj je jasno, da je povprečno število celic sorazmerno padajoče z večanjem števila središč. Morda se na grafih nekoliko zavajajoče zdi, da imajo grafi s 100 in 500 vozlišči ob majhnem številu središč enako velikost celic, vendar je treba opomniti, da se graf za 500 vozlišč začne z 20 središči, graf za 100 vozlišči pa s 5, ko primerjamo pri enakem številu središč se vrednosti občutno razlikujejo - pri grafu s 100 vozlišči je pri dvajsetih središčih povprečna velikost povprečne celice nekaj manj kot 10, medtem ko je pri grafu z 500 vozlišči ta velikost 30. Na vseh grafih je očitno eksponentno padajoče sorazmerje, kar lahko predvidevamo za vse grafe z več vozlišči.

Zelo podobne rezultate dobimo tudi pri analiziranju ciklov, le da so bile najvišje povprečne velikosti celic nekaj višje (za približno 10). Še vedno pa velikosti eksponentno padajo z višanjem števila središč.

Ko primerjamo grafe za različna števila vozlišč, je to eksponentno padanje zelo lepo vidno.




**2D in 3D mreže** Naslednja podatkovna struktura, ki sva si jo v luči analiziranja Voronoijevih diagramov pobliže ogledala, so dvo in tridimenzionalne mreže. Pri mrežah sva pričakovala, da se bodo povprečne velikosti Voronoijevih celic razlikovale od "strukture" mreže, saj lahko npr. mrežo s 24 vozlišči konstruiramo kot mrežo $3 \times 8$ ali kot $4 \times 6$. \ Za analizo sva tako izbrala vse možne mreže z 256 vozlišči ($2^8$). Vse možne mreže so torej, dvodimenzionalne: $1 \times 256$, $2 \times 128$, $4 \times 64$, $8 \times 32$ in $16 \times 16$; tridimenzionalne $2 \times 2 \times 64$, $2 \times 4 \times 32$, $2 \times 8 \times 16$, $4 \times 4 \times 16$ ter $4 \times 8 \times 8$. \ S korakom 5 sva dvajsetkrat generirala naključno izbrana vozlišča med 5 ter 125 vozlišči. Z grafa sva hitro ugotovila, da so dvodimenzionalne mreže precej podobne in da omejenost zgolj na pot dolžine 256 ne vpliva drastično na povprečne velikosti Voronoijevih celic. Tudi dodatna dimenzija na drugem grafu ne pomeni drastičnega povečanja povprečnih velikosti celic, npr. na mreži $1 \times 256$ nam 40 izbranih vozlišč generira Voronoijeve celice s povprečno močjo oz. velikostjo 5.85, medtem ko nam 40 izbranih vozlišč na najbolj "svobodni" mreži $4 \times 8 \times 8$ generira Voronoijeve celice povprečne moči 9.73.