Kratko poročilo

2. skupina - An Optimal Bound for the MST Algorithm

Sara Korat Anja Leskovšek

7. november 2017

Kratek opis problema

Računanje energije učinkovitega oddajnega omrežja je ena izmed pomembnejših nalog pri brezžičnemu internetu. Problem predstavimo z grafom, kjer vozlišča predstavljajo postaje, ki so med sabo povezane z radijskimi oddajniki in sprejemniki. Če želimo poslati sporočilo iz postaje a do postaje b, potrebuje postaja a oddati sporočilo z dovolj energije, da doseže postajo b. Vsaka postaja ima dodeljeno vrednost moči, ki nam pove obseg, znotraj katerega lahko vse postaje sprejmejo njeno sporočilo. Iz teh podatkov lahko sestavimo transmisijski graf $G = (S, A)$, kjer množica točk S predstavlja skupino povezav, množica A pa vsebuje le take usmerjene povezave od vozlišča a do vozlišča b, če je b znotraj obsega postaje a. Celotna moč, ki jo potrebujemo, da v grafu G vzpostavimo vse povezave je enaka:

$$power(G) = \sum_{i \in V} = \gamma \cdot r_G(i)^\alpha$$

Parameter $\gamma \geq 1$ predstavlja kvaliteto prenosa in $\alpha \geq 1$ dolžinski smerni koeficient moči. V idealnem okolju je $\alpha = 2$, vendar lahko doseže tudi vrednost 6, odvisno od stanja lokacije omrežja. V našem primeru kvaliteta ne vpliva na končni rezultat, zato lahko brez škode za splošnost predpostavimo $\gamma = 1$. $r_G(i)$ pa predstavlja minimalni potrebni obseg izvora i, da vzpostavi vse svoje izhodne povezave v grafu G. Izračunamo ga z naslednjo formulo:

$$r_G(i) := \underset{j \in \Gamma_G(i)}{max}\, dist(i, j)$$

Oznaka $\Gamma_G(i)$ predstavlja vse sosede vozlišča i v grafu G.

Problem EEBT (energy efficient broadcasting tree problem) predstavi postaje kot točke v Evklidski ravnini in ceno povezav predstavlja dolžina med njimi. Cilj problema je poiskati transmisijski graf G, ki minimizira $power(G)$ in vsebuje usmerjeno vpeto drevo z izvorom. EEBT je NP-težek, kar pomeni, da ni algoritma, ki bi problem rešil v polinomskem času. Najboljša aproksimacija rešitve je poiskati minimalno vpeto drevo oz. najcenejše vpeto drevo omrežja z izvorom in usmeriti povezave.

Najin cilj v projektni nalogi je eksperimentalno prikazati naslednji izrek.

IZREK 1: Naj bo S množica točk v enotski krožnici s središčem v koordinatnem izhodišču. Središče krožnice je tudi element množice S. Naj bodo $e_1, e_2, \ldots, e_{|S|-1}$ povezave v Evklidskem najmanjšem vpetem drevesu. Potem velja:

$$\mu(S) = \sum_{i=1}^{|S|-1} |e_i|^2 \leq 6$$

Načrt dela

Naloga bo predstavljena v okolju Python, vendar bodo nekateri grafi izrisani s programskim jezikom R. Na začetku bo definirana funkcija, ki naključno izbere točke v enotskem krogu vključno s središčem, ki je v koordinatnem izhodišču. (Pri generiranju uporabimo enakomerno porazdelitev števil na intervalu $[-1, 1]$ za vsako koordinato posebej. Ker s tem dobimo točke znotraj kvadrata s stranico 2, preden točko shranimo v slovar naključnih točk, preverimo še, če se nahaja znotraj kroga.)

Potem se bo določila cena povezav med točkami, kar pomeni, da bo program izračunal vse dolžine. S tem bomo dobili bomo poln graf, kar pomeni, da so vse točke med sabo povezane. V nadaljevanju bo potrebno uporabiti ustrezen algoritem, ki vrne najcenejše

vpeto drevo. Možnosti je več, kot na primer Kruskalov algoritem, Primov algoritem, Boruvkov algoritem,... Vsi uporabljajo požrešno metodo, vendar so med njimi razlike v časovni zahtevnosti. Primov algoritem ima vedno časovno zahtevnost $O(n^2)$, medtem ko imata Boruvkov in Kruskalov algoritem časovno zahtevnost $O(Elog(V))$, kjer je E število povezav in V število vozlišč. Ker imamo poln graf, je $E = \frac{n(n-1)}{2}$, kar pomeni, da je časovna zahtevnost približno $O(n^2log(n))$. Iz tega sledi, da je v našem primeru Primov postopek najbolj primeren.

Naslednja funkcija v Pythonu bo uporabila Primov algoritem in vrnila vsoto kvadratov dolžin dobljenega najcenejšega drevesa. Ideja eksperimenta je, da celoten program večkrat zaženemo in pogledamo maksimalno vrednost. Če je maksimum manjši od 6, potem smo eksperimentalno pokazali veljavnost izreka.

V nadaljevanju projekta bomo primerjali enotski krog z drugimi liki, kot so elipsa, kvadrat, pravokotnik in trikotnik. Za bolj natančno primerjavo bomo vzeli take like, ki bodo imeli enako ploščino kot enotska krožnica. Enako kot prej, bo program v vsak lik naključno dal željeno število točk, med njimi izračunal dolžine in izvedel Primov algoritem. Program bomo večkrat zagnali, da dobimo čim bolj natančen maksimum. Vse izračunane maksimume bomo med sabo primerjali in primerjali z oceno enotskega kroga.

Zanimivo bi bilo tudi primerjati ocene vsot pri krogih z različnim polmerom. Polmer bi počasi povečevali in opazovali kaj se dogaja z vsotami kvadratov dolžin oz. z njihovimi zgornjimi mejami. Predvidevava, da bodo vsote enkrat šle proti neskončnosti (ko bo polmer šel proti neskončnosti), zato naju zanima, s kakšno hitrostjo se bo vsota večala.

Poleg kroga bomo pogledali tudi kak drug lik, pri katerem bomo večali njegovo ploščino sorazmerno s polmerom kroga. Rezultati bodo prikazani z grafom v R-u.

Hipoteze

Pri večanju pločine kroga, se bo meja, ki je v izreku za enotski krog postavljena na 6, večala skupaj s ploščino. Prav tako se bo večala meja tudi pri ostalih likih (kvadratih, pravokotnikih, elipsah, trikotnikih), pričakujeva pa, da bodo vsaj pri ploščini π meje podobne.

Predvidevava, da bodo zgornje meje vsot kvadratov dolžin varirale med liki z istimi ploščinami - verjetno bodo pomembne tudi maksimalne razdalje, ki jih lahko dosežemo znotraj lika (torej tudi način, kako bomo izbrali dolžine stranic npr. pravokotnikov). Večje kot bodo ploščine, večja bo razlika maksimalnih razdalj med liki (npr. med krogom in kvadratom. Pri ploščini π je maksimalna dolžina znotraj kroga 2, diagonala kvadrata pa približno $2,507$. Pri ploščini 4π je maksimalna dolžina znotraj kroga 4, diagonala kvadrata pa je dolga približno $5,0133$). Z dovoljšnim številom poskusov s čim več točkami želiva ugotoviti, kako maksimalne razdalje znotraj likov vplivajo na zgornje meje vsot.