Problem maksimalnog zajedničkog podstabla

Seminarski rad u okviru kursa
Računarska inteligencija
Matematički fakultet, Univerzitet u Beogradu

Velimir Bićanin, Ana Jakovljević
velimirbicanin@gmail.com, ana.jakovljev98@gmail.com

9. septembar 2020.

Sažetak

U ovom seminarskom radu biće obrađen problem maksimalnog zajedničkog podstabla za čije rešavanje ćemo predstaviti različite heuristike prilagođene problemu, kao i metaheuristike opšte namene. Biće upoređeni rezultati izvršavanja različitih metoda.

Ključne reči:

Sadržaj

1 Uvod

Problem maksimalnog zajedničkog podgrafa se javlja u različitim oblastima današnjice ... Problem maksimalnog zajedničkog stabla je specijalan slučaj tog problema, ali i pored toga postoji potreba za izučavanjem. Jedan je od razloga je velika primena u poređenju hemiskih struktura. Informacije koje se dobijaju poređenjima se smeštaju u bazu i koriste za dalja istraživanja. Ovi problemi su NP-teški, pa za njih trenutno ne postoji polinomijalno rešenje. Zato ćemo predstaviti algoritme koji pružaju suboptimalna rešenja u razumnom vremenu.

Definicija 1.1 *Ulaz za ovaj problem predstavlja kolekcija korenskih, neoznačenih stabala proizvoljnog stepena* $\{T1, T2, ..., Tm\}$*. Kao rezultat primene algoritama dobija se maksimalno stablo koje je izomorfno podstablu svakom od ulaznih stabala.*

2 Heuristike

Za rešavanje ovog problema predstavićemo aproksimativne algoritme koji nam daju dobre performanse izvrašavanja i zadovoljavajuće rešenje. Razlike između algoritama su bazirane na način izbora čvorova pomoću kojih generišu rešenja. Za početak uvodimo oznaku S za najmanje među ulaznim stablima i n za broj čvorova koje on sadrži.

2.1 Algoritam aproksimacije opšte namene

Podskup čvorova X stabla određuje jedinstveno podstablo koje sadrži čvorove iz X, svaki čvor na putu između čvorova u X i sve grane koje ih povezuju u stablo, tj. postoji jedinstveno minimalno podstablo koje sadrži X. Algoritam se zasniva na podeli čvorova najmanjeg stabla. Čvorovi se dele u proizvoljnih n/log(n) skupova veličine log(n). Svaki poskup svakog skupa jedinstveno određuje podstablo za koje je potrebno proveriti da li predstavlja zajedničko podstablo ulaza. Pamtiti maksimalno nađeno podstablo i na kraju ga ispisati. Broj stabala kandidata je n/log(n) * 2log(n) < n2.

```
Ulaz: T = {t1,t2, ..., tm}
Izlaz: maksimalno zajednicko podstablo M

S = min_stablo(C)
skupovi = podeliti_cvorove(S)
for skup in skupovi:
    for podskup in skup:
        G = generisano_stablo(podskup)
        if zajednicko_podstablo(G,T):
            M = max_stablo(M,G)
```

2.2 Metoda slučajnosti

Uvodimo nov parametar k = n/opt, n je veličina minimalnog stabla, a opt pretpostavljena veličina maksimalnog podstabla. Sa preciznom procenom parametra k moguće je razviti unapređen algoritam za stabla manjeg stepena. Uvodni algoritam predstavlja metod slučajnosti koji nasumično bira čvorove datog stabla i na osnovu njih generiše jedinstveno minimalno podstablo za koje proverava da li je izomorfno podstablo. Algoritam se vodi time da verovatnoća da su izabrani odgovarajući čvorovi raste ukoliko se uzorkovanja izvrši O(n2) puta.

2.3 Deterministički algoritam

Neka je D jednak maksimalnom stepenu čvorova stabla. Dva čvora stablo su udaljena ukoliko je distanca između njih najmanje 1/2 *logD(n), u suprotnom oni su bliski. Skup čvorova X je raspršen ako je svaki par čvorova u X udaljen. Ideja je da se naprave takvi skupovi za koje važi da su ima svi čvorovi udaljeni.

Algoritam se zasniva na poznavanju parametar k sa određenom preciznošću. Stablo S se particioniše u kolekcije od najviše k*m šuma gde svaka sadrži između n/km i 2n/km elemenata. Koreni stabala u šumama moraju imati istog roditelja u S. Zatim za svako stablo svake šume iz daljeg razmatranja odbaciti čvorove koji su bliski korenu stabla. Podeliti preostale čvorove u 2n/km grupa gde svaka grupa sadrži najviše jedan čvor iz svake šume, a zatim pokušati sve podskupove veličine m/3 (m = logkn) i proveriti da li indukuju zajedničko podstablo. Za podelu stabla moguće je primeniti pohlepni algoritam.

```
Ulaz: T = {t1,t2, ..., tn}
Izlaz: maksimalno zajednicko podstablo M

S = min_stablo(C)
sume = particionisati(S)
for suma in sume:
    for stablo in suma:
        eliminisati_bliske(stablo)
grupe = grupisati(sume)
for grupa in grupe:
    M = max_stablo(M, generisi(grupa))
```

Svi koraci algoritma mogu biti izvršeni u linearnom vremenu osim poslednjeg koji zavisi od broj podskupova koji se generiše, kao i od složenosti testiranja izomorfizma podstabla.

3 Metaheuristike

Metahueristike su metode koje imaju širi opseg primene i postoji mogućnost njihovog prilagođavanja različitim konkretnim problemima. Na dati problem smo primenili genetski algoritam i metodu simuliranog kaljenja.

3.1 Genetski algoritam

Genetski algoritam je populacioni algoritam kojim kroz niz iteracija populacija teži optimalnoj vrednosti. Za konkretan problem pre svega je porebno prilagoditi način predstavljanja jedinke populacije i odrediti početnu populaciju. U ovoj implementaciji je kao kod jedinke korišćena struktura koja opisuje stablo. Pokušano je predstavljanje stabla pomoću binarnog niza, ali ispitivanje izomorfnosti takvih stabala nije bilo moguće. Početna populacija treba da bude takva da je iz nje moguć brz razvoj kvalitetne populacije, pa je izabran skup jedinki koje sadrže jedan čvor. Radi raznovrsnosti razmatrano je inicijalizovanje jedinkama koje sadrže različit broj čvorova u rasponu od 0 do maksimalnog stepena najmanjeg stabla, međutim rezultati pokazuju da su obe početne populacije jednako dobre. Za dalji rad, potrebno je definisati funkcije prilagođenosti. Kriterijum za rešenje je da mora da bude izomorfno podstablo svakog stabla ulazne kolekcije, pa se kvalitet jedinke postavlja na nula ukoliko to ne ispunjava. Ukoliko je taj kriterijum ispunjen, kao prilagođenost se koristi veličina stabla jedinke. U iteracijama genetskog algoritma dolazi do selekcije, ukrštanja i mutacije jedinki i na taj način se kreira nova populacija. Razmatrane selekcije su turnirska i ruletska koje su pokazale slične performanse. Ukrštanje se vrši kombinovanjem dece između slučajno izabranih čvorova stabala, a mutacija slučajnim izborom čvora koji sa određenom verovatnoćom biva izbačen iz stabla.Ubačen je elimitizam, jer je uočeno poboljšanje performansi, kao i prekid iteracija nakon određenog broja ponavljanja najbolje jedinke populacije.

3.2 Simulirano kaljenje

Simulirano kaljenje je algoritam koji se zasniva na prilagođavanju jedne jedinke uz pomoć funkcije prilagođenosti. Kroz niz iteracija, ta jedinka doseže do optimalne vrednosti. Prilagođavanje jedinke se vrši eliminacijom ili dodavanjem čvorova na stablo koje je čini. Funkcija prilagođenosti je ista kao kod genetskog algoritma. Kako ne bi došlo do lokalne pretrage uvodi se mogućnost prihvatanja i lošijeg rešenja, čija verovatnoća opada sa povećanjem broja iteracija.

4 Rezultati istraživanja