Prirodno-matematički fakultet u Banjoj Luci,  
Matematika i informatika– informatatika



Težinski problem potpune dominacije

Sadržaj:

[1 Opis problema 2](#_Toc100316482)

[3 ILP model za WTDP 3](#_Toc100316483)

[4 Pohlepni algoritam 4](#_Toc100316484)

[5 Genetski algoritam 5](#_Toc100316485)

[6 Instance 6](#_Toc100316486)

[6.1 Primjer instance 6](#_Toc100316487)

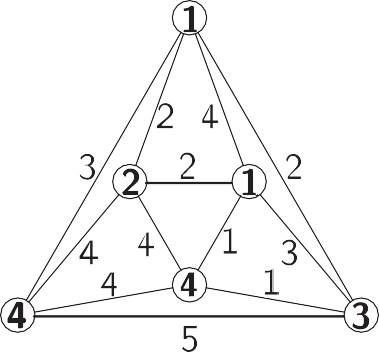
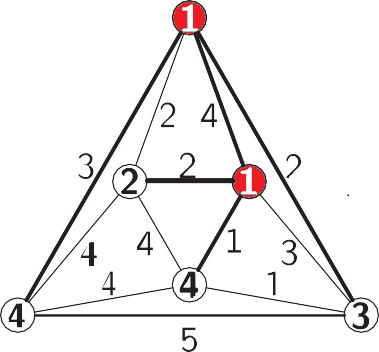
[6.2 Rezultati mjerenja 7](#_Toc100316488)

[7 Zaključak 9](#_Toc100316489)

[8 Literatura 10](#_Toc100316490)

# 1 Opis problema

Težinski problem potpune dominacije je problem iz kombinatorne optimizacije. Dat je neusmjeren graf , gdje predstavlja skup čvorova, a skup ivica grafa . Ivica koja spaja čvorove se označava sa ili sa . Susjedstvo čvorova se definiše kao , a skup ivica identičnih sa čvorom je definisano kao . Za dati grafpodskup čvorovase naziva dominantan skup ako je svaki čvorsusjedan bar jednom čvoru iz D, odnosno, ako za svaki čvor postoji bar jedan čvortakav da je.Drugim riječima, skup koji dobijemo kao rezultat ustvari predstavlja podskup čvorova tako da je svaki čvor u grafu, uključujući i čvorove u , susjedan čvoru u .

**Slika1.**Primjer težinskog problema potpune dominacije. Slika lijevo predstavlja težinski graf. Slika desno predstavlja optimalan totalni skup težinskog grafa

**2 Matematička formulacija**

gdje predstavlja skup ivica unutar skupa , a skup susjeda od .  
Dakle, data funkcija se sastoji od 3 dijela:  
1. Ukupne težine tjemena u   
2. Ukupne težine ivica u podgrafu   
3. Ukupna težina ivica minimalne težine koje povezuju svaki vrh sa vrhovima koji se nalaze u .

# 3 ILP model za WTDP

Neka ja A matrica susjedstva grafa G = (V,E), pri čemu je |V| = n. Koristimo binarnu promjenljivu za označavanje da li je čvor izabran za rješenje ili ne.   
Neka je i . Tada minimalnu veličinu totalnog dominantnog skupa možemo dobiti na sljedeći način:

(1)

s.t.

Data ograničenja obezbijeđuju da svaki čvor u grafu G ima susjeda koji je izabran, i predstavljaju ograničenja ukupnog dominantnog skupa.

Neka su {}, {} i {} skupovi binarnih promjenljivih. Za svaki čvor promjenljiva pokazuje da li je izabrano kao rješenje ili ne. Za svako promjenljiva pokazuje da li je izrabrano kao rješenje ili ne, dok promjenljiva predstavlja minimalnu vrijednost i .

Tada funkciju cilja možemo formulisati na sledeći način:

s.t.

(2)

(8)

(6)

(7)

(9)

(10)

(3)

(4)

(5)

Ograničenje (2) je drugi oblik ograničenja totalnog dominantnog skupa. Ograničenja (3)-(7) obezbjeđuju pravilno postavljanje varijabli i pokazuju koje ivice su izabrane. Primjetimo da su i i koji se tiču ivice postavljeni na nula, ograničenje (3) forsira varijablu da uzme vrijednost nula, što znači da ivica koja povezuje dva neizabrana čvora ne može biti dio rješenja. U ograničenju (8) skup predstavlja skup svih ivica čvora .Kako ograničenja (4)-(6) daju linearizaciju za , možemo da vidimo da, ukoliko se i i postave na 0, tada mora biti 1. Stoga je varijabla jedanka onoj iz uslova (10) , koji ukazuje da ivica koja spaja dva izabrana čvora takođe mora biti izabrana. Ograničenje (11) obezbjeđuje da je neizabrani čvor povezan sa najmanje jednom granom koja je u rješenju, a funkcija cilja obezvjeđuje da je ona minimalne težine.

# 4 Pohlepni algoritam

**Algoritam1:** Pohlepni algoritam

1:***ulaz:*** *neusmjeren povezan težinski graf*

2:

3:*U D dodajemo čvor za koji ima*

4: ***dok*** *postoji čvor iz koji nije susjed ni jednom čvoru iz* ***radi***

5:*čvor v i granu (u,v) sa*

*dodajemo u podgraf D*

6:**kraj**

7:**izlaz:**Totalan dominantan skup grafa

Za ulazni graf , algoritam počinje sa praznim skupom , odnosno grafom koji ne sadrži ni jedan čvor. Prvo biramo prvi čvor koji dodajemo u graf S i to radimo tako što tražimo minimalan odnos stepena čvora i težine tog čvora za svaki čvor grafa. Ukoliko ovo nije totalni dominirajuć skup, nastavljamo da dodajemo čvorove dok se to ne desi. U svakom narednom koraku biramo čvor sa najmanjim zbirom odnosa stepena čvora i težine čvora i težine grane . Čvor biramo iz susjednih čvorova čvorevima koji su već u , tako da se taj čvor ne nalazi u . Novoizabrani čvor i granu dodajemo u naš graf . Poslije izvijesnog broja koraka dobijamo totalan dominantan skup početnog grafa .

# 5 Genetski algoritam

**Algoritam2:** Genetski algoritam

1:***ulaz:*** *neusmjeren povezan težinski graf*

2:*inicijalizacija populacije P*

3:*evaluacija populacije P*

4:***dok*** *nije zadovoljen uslov* ***radi***

5: *selekcija*

6: *ukrštanja*

7: *mutacija*

8: *evaluacija populacije*

*9:****kraj***

10:***izlaz:****Totalan dominantan skup grafa*

Algoritam počinje sa radom za ulazni graf G. Zatim se nasumičnim izborom generiše populacija. Veličina populacije zavisi od veličine instance, tj. određuje se na osnovu broja čvorova. Za svaku jedinku se, takođe nasumično, generiše binarni niz kojim se predstavljaju geni jedinke. Dužina niza jednaka je broju čvorova u grafu.  
Neka je i{ 0, 1, …, n-1} i neka je n broj čvorova u grafu. Element 0 označava da čvor na poziciji i nije označen, a 1 da čvor na poziciji i jeste označen.  
Svakoj jedinki dodijeljena je brojevna vrijednost tj. *fitness*, koja se odnosi na kvalitet jedinke, a računa se u 3. koraku – evaluaciji.  
U procesu selekcije biramo jedinke koje će da prežive u procesu evolucije. Tehnika koju koristimo je rulet selekcija u kojoj 20 najboljih jedinki preživljava u narednu generaciju, od tih 20 biramo 15 koje će se ukrštati, a preostale se odbacuju.   
U narednom koraku se vrši ukrštanje gena roditeljskih jedinki. Koristi se jednopoziciono ukrštanje, pri čemu se tačka prekida generiše na slučajan način. Jedinke dobijene u procesu ukrštanja dalje mutiraju. Mutacija gena se vrši sa vjerovatnoćom . Nakon mutacije ponovo se vrši evaluacija.   
Proces selekcije, ukrštanja, mutacije i evaluacije se ponavlja sve dok se neko rješenje ne ponovi određen broj puta tj. ako se u određenom broju ponavljanja iteracije ne desi nikakvo poboljšanje.  
Na kraju, iz liste jedinki za rješenje uzimamo onu sa najmanjim izračunatim fitnesom.

# 6 Instance

Generisano je ukupno 45 instanci, koje su podijeljene u 3 grupe po 15 instanci. Podijela je vršena na osnovu veličine instanci na:

* Male instance : sadrže 20 čvorova
* Srednje instance: sadrže 30 čvorova
* Velike instance: sadrže 50 čvorova

Model, operacije učitavanja instanci i ispitivanje rezultata napisani su u programskom jeziku Java. Mjerenje instance vršeno je u IBM ILOG CPLEX – u. Specifikacije laptopa na kojim su vršena sva mjerenja:

* Procesor: Intel(R) Core(TM) i3-6006U CPU @ 2.00GHz 1.99 GHz
* RAM memorija: 4.00 GB
* Operativni sistem: Windows 10

## 6.1 Primjer instance



Slika 1: primjer instance

U prvom redu se nalaze parametri |V| i |E|. Zatim u sledećih |V| redova se nalaze čvor i , i težina čvora n. Nakon toga za svaku granu naveden je redni broj grane, zatim oznaka čvora od koga polazi grana, pa oznaka čvora do koga ta grana vodi i na kraju parameter e tj. težina navedene grane.

## 6.2 Rezultati mjerenja

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instanca | | | Pohlepni algoritam | | Genetski algoritam | | | | CPLEX | |
| Id | |V| |  | rezultat | t[ms] | rezultat | maksimalni rezultat | t[ms] | S10[[1]](#footnote-2) | rezultat | t[ms] |
| 1 | 20 | 38 | 66 | 2 | 66 | 64 | 189 | 1.673 | 63 | 377 |
| 2 | 20 | 40 | 87 | 2 | 76 | 67 | 153 | 3.769 | 58 | 286 |
| 3 | 20 | 33 | 77 | 1 | 76 | 68 | 87 | 3.225 | 58 | 226 |
| 4 | 20 | 43 | 70 | 1 | 62 | 59 | 157 | 1.685 | 51 | 533 |
| 5 | 20 | 36 | 83 | 2 | 64 | 60 | 64 | 2.343 | 55 | 197 |
| 6 | 20 | 93 | 62 | 2 | 48 | 45 | 330 | 1.019 | 44 | 702 |
| 7 | 20 | 93 | 53 | 2 | 53 | 50 | 281 | 1.939 | 47 | 1302 |
| 8 | 20 | 91 | 51 | 2 | 49 | 46 | 303 | 1.300 | 46 | 1058 |
| 9 | 20 | 98 | 54 | 1 | 45 | 41 | 323 | 1.758 | 40 | 1149 |
| 10 | 20 | 87 | 55 | 1 | 45 | 42 | 294 | 1.281 | 41 | 1033 |
| 11 | 20 | 154 | 55 | 1 | 38 | 37 | 316 | 1.000 | 37 | 1363 |
| 12 | 20 | 152 | 39 | 1 | 39 | 35 | 527 | 2.107 | 35 | 1393 |
| 13 | 20 | 145 | 50 | 1 | 44 | 42 | 462 | 1.044 | 40 | 1394 |
| **14** | **20** | **142** | **39** | **1** | **34** | **34** | **420** | **0.806** | **34** | **1267** |
| 15 | 20 | 146 | 40 | 1 | 37 | 34 | 508 | 0.831 | 34 | 1263 |

**Tabela 1: Rezultati za male instance**

## 

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instanca | | | Pohlepni algoritam | | Genetski algoritam | | | | CPLEX | |
| Id | |V| | |E| | rezultat | t[ms] | rezultat | maksimalni rezultat | t[ms] | S10 | rezultat | t[ms] |
| 1 | 50 | 227 | 131 | 2 | 138 | 126 | 1765 | 3.627 | 111 | 264302 |
| 2 | 50 | 241 | 120 | 3 | 139 | 131 | 1856 | 3.208 | 106 | 543235 |
| 3 | 50 | 249 | 133 | 4 | 142 | 127 | 1489 | 4.051 | 111 | 600065 |
| 4 | 50 | 256 | 110 | 3 | 128 | 118 | 1968 | 4.001 | 101 | 363798 |
| 5 | 50 | 260 | 135 | 4 | 135 | 130 | 1939 | 2.727 | 108 | 601741 |
| 6 | 50 | 587 | 98 | 3 | 102 | 88 | 6302 | 3.195 | 82 | 601543 |
| 7 | 50 | 606 | 87 | 4 | 110 | 97 | 4700 | 2.750 | 87 | 601062 |
| 8 | 50 | 624 | 88 | 4 | 109 | 95 | 5013 | 3.429 | 85 | 602212 |
| 9 | 50 | 600 | 116 | 6 | 105 | 90 | 5701 | 2.685 | 82 | 602667 |
| 10 | 50 | 634 | 100 | 3 | 100 | 87 | 5675 | 1.745 | 82 | 602101 |
| 11 | 50 | 968 | 113 | 4 | 89 | 82 | 8230 | 3.124 | 77 | 568591 |
| 12 | 50 | 975 | 109 | 3 | 97 | 89 | 7655 | 2.571 | 72 | 275787 |
| 13 | 50 | 970 | 131 | 4 | 99 | 83 | 8229 | 3.822 | 74 | 380330 |
| 14 | 50 | 950 | 101 | 3 | 93 | 86 | 7211 | 1.897 | 76 | 600551 |
| 15  **Tabela 2: Rezultati za srednje instance** | 50 | 999 | 111 | 4 | 92 | 80 | 7385 | 2.576 | 79 | 600542 |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instanca | | | Pohlepni algoritam | | Genetski algoritam | | | | CPLEX | |
| Id | |V| | |E| | rezultat | t[ms] | rezultat | maksimalni rezultat | t[ms] | S10 | rezultat | t[ms] |
| 1 | 100 | 990 | 207 | 193 | 234 | 225 | 19291 | 4.959 | 186 | 601075 |
| 2 | 100 | 960 | 211 | 39 | 245 | 229 | 19293 | 6.090 | 187 | 600357 |
| 3 | 100 | 961 | 190 | 41 | 239 | 225 | 17622 | 5.473 | 189 | 600761 |
| 4 | 100 | 985 | 190 | 23 | 229 | 214 | 14971 | 5.929 | 196 | 600926 |
| 5 | 100 | 979 | 201 | 24 | 225 | 211 | 13397 | 4.076 | 183 | 601128 |
| 6 | 100 | 2466 | 183 | 25 | 204 | 189 | 41383 | 6.133 | 184 | 602311 |
| 7 | 100 | 2464 | 203 | 21 | 211 | 195 | 43158 | 6.974 | 182 | 603549 |
| 8 | 100 | 2523 | 175 | 34 | 210 | 189 | 39621 | 6.201 | 179 | 601118 |
| 9 | 100 | 2474 | 196 | 25 | 192 | 183 | 41580 | 5.748 | 169 | 601131 |
| 10 | 100 | 2493 | 157 | 36 | 193 | 182 | 45857 | 6.794 | 172 | 601041 |
| 11 | 100 | 3916 | 191 | 26 | 199 | 184 | 76478 | 4.742 | 159 | 601760 |
| 12 | 100 | 3971 | 209 | 23 | 205 | 193 | 68970 | 4.247 | 164 | 601214 |
| 13 | 100 | 3942 | 187 | 20 | 199 | 181 | 64232 | 7.472 | 164 | 601394 |
| 14 | 100 | 3932 | 180 | 21 | 191 | 181 | 62377 | 3.874 | 151 | 601834 |
| 15 | 100 | 3997 | 202 | 27 | 186 | 175 | 66852 | 3.534 | 150 | 601329 |

## 

**Tabela 3: Rezultati za velike instance**

# 7 Zaključak

Na osnovu tabele rezultata možemo zaključiti da prosječan rezultat genetskog algoritma u svim slučajevima daje bolje rezultate od pohlepnog algoritma za male instance, dok je za srednje instance genetski algoritam bolji od pohlepnog u 60% slučajeva.  
Na velikim instancama genetski algoritam daje najlošije rezultate, i samo u 20% slučajeva je bolji od pohlepnog algoritma.  
  
Cplex daje bolje rezultate i od pohlepnog i od genetskog algoritma za instance svih veličina. Takođe Cplex je pronašao optimalno rješenje za sve instance. Prosječno vrijeme izvršavanja na malim instanacama je 13543 ms, na srednjim 520569ms, i na velikim 601395ms.

Pohlepni algoritam se izvršava brže od genetskog algoritma na instancama svih veličina.

Možemo zaključiti da genetski algoritam ima bolje performance na malim i srednjim instancama. Na velikim instancama bolje performance ima pohlepni algoritam.

# 8 Literatura

1. S10 – standardna devijacija [↑](#footnote-ref-2)