Genetski algoritam za višeciljnu optimizaciju

Aleksandra Bošković, Miloš Miković
aleksandra94@hotmail.rs, milos.mikovicpos@gmail.com

13. april 2020.

Sadržaj

1 Uvod

Problem trgovačkog putnika (Traveling Salesman Problem, TSP) je jedan od najpoznatijih problema iz grupe NP - teških problema diskretne i kombinatorne optimizacije.U osnovi ovaj problem je definisan na sledeći način: trgovački putnik tačno zna koje gradove treba da poseti i koja je njihova međusobna udaljenost, jedini problem je što je u obavezi da poseti svaki grad samo jednom i vrati se u grad iz kog je krenuo. Ovom osnovnom problemu dodaćemo jos jedan podatak koji trgovački putnik ima a to je vreme potrebno da se pređe put između svaka dva grada. Cilj je pronaći tačnu putanju kojom trgovački putnik treba da ide tako da i ukupno provedeno vreme na putu i ukupna pređena dužina puta budu minimalni. Ovu putanju ćemo pokušati da pronađemo korišćenjem genetskog algoritma.

Ideja zasnovana na idejama iz literature [3] , [4] i [1]

2 Algoritam

Ulaz: podaci o udaljenosti između svakog od gradova, kao i potrebno vreme za prelazak puta između svakog od gradova
Izlaz: najbolje pronađeno rešenje genetskim algoritmom

Da bismo mogli da poredimo rešenja definišimo prvo funkciju cilja za ovaj problem:

$$\begin{cases} min \sum_{i,j \in x} d_{ij}, \\ min \sum_{i,j \in x} t_{ij} \end{cases}$$

gde je d_{ij} dužina puta od grada i do grada j, a t_{ij} veme potrebno da se pređe put od grada i do grada j i x jedna permutacija datih gradova.

Sa obzirom na to da je često nemoguće u isto vreme minimizovati obe funkcije, a u cilju podjednakog vrednovanja obe veličine definišimo funkciju cilja na sledeći način

$$f(x) = \frac{D_x}{\sum_{i=1}^{m} D_i} + \frac{T_x}{\sum_{i=1}^{m} T_i}$$

pri čemu je m broj jedinki u generaciji D_x ukupna dužina pređenog puta za rešenje x a T_x ukpno vreme potrebno da se pređe odabrani put x.

Naš cilj je da za odabrano rešenje x važi:

$$f(x) < f(x'), \forall x' \in \Omega$$

pri čemu je Ω skup svih permutacija skupa gradova kroz koje putnik treba da prođe a x izabrana najbolja permutacija.

2.1 Opis implementacije elemenata genetskog algoritma

Kodiranje jedinke:
predstavljena je kao permutacija skupa svih gradova i predstavlja redosled kojim trgovački putnik obilazi gradove.
Inicijalna populacija:
generisana je random iz skupa svih mogućih permutacija obilaska gradova.
Selekcija:
turnirska selekcija odabira jedinki za ukrštanje.
Mutacija:
zamena random odabrana dva indeksa niza koji predstavlja jedinku [2].
Ukrštanje:
Prvi način: ukrštanje prvog reda za kombinatorne probleme
Drugi način: random odabran segment jednog roditelja staviti na početak deteta a ostatak niza dopuniti preostalim elementima onim redosledom kojim se javljaju u drugom roditelju.
Drugi način se pokazao kao bolji jer prvi prebrzo konvergira.
Nova populacija:
n odabranih najboljih jedinki iz prethodne generacije i deca ukrštenih roditelja.
Kriterijum zaustavljanja:
dostignut zadati broj iteracija.

3 Poređenje sa brute force algoritmom

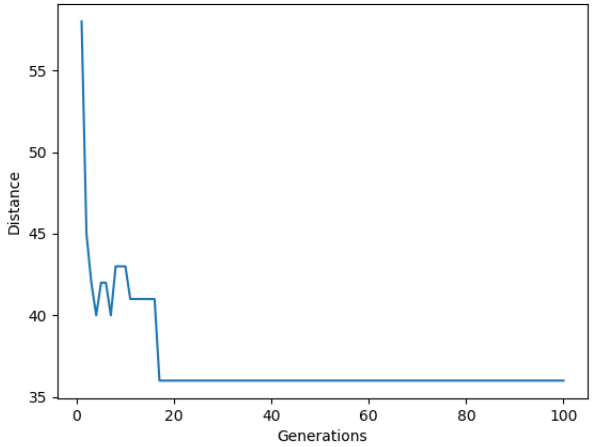
U cilju ocenjivanja dobijenog rešenja ovakvim algoritmom uporedimo sa rešenjem dobijenim brute force algoritmom, koji ispituje sve moguće permutacije.U algoritmu brute force pri odabiru najboljeg rešenja tražimo minimum funkcje:

$$f(x) = \frac{D_x}{\sum_{i=1}^{n!} D_i} + \frac{T_x}{\sum_{i=1}^{n!} T_i}$$

pri čemu su korišćene oznake iste kao i kog rešenja za genetski algoritmom. Uočimo razliku da ovde u imeniocu sumiramo vrednosti svih mogućih permutacija jer su nam one poznate, dok nam kod rešenja genetskim algoritmom ova suma zavisi od populacije u svakoj generaciji.

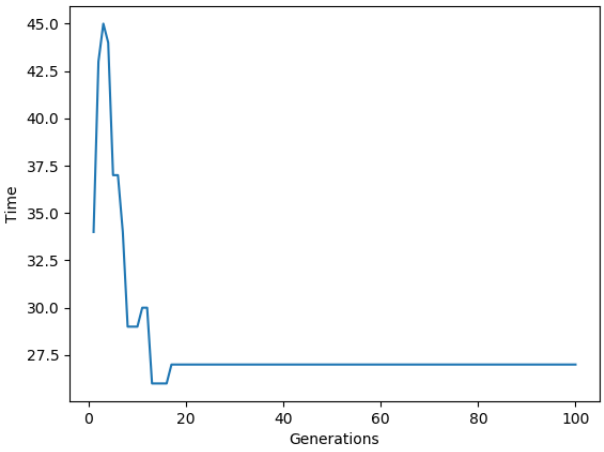
U nastavku na slikama su dati rezultati posmatranog problema za 11 gradova, 100 generacija, verovatnoćom mutacije 0.05, veličinom generacije 20, veličinom turnira 5. Pri čemu najbolje rešenje koje pronalazi algoritam brute force za ovih 11 gradova iznosi $D = 29$ za ukupnu dužinu optimalnog puta i $T = 32$ za vreme potrebno da se pređe optimalan put.

Na slici 1 možemo pratiti na koji način se izabrana vrednost optimalnog rešenja u svakoj generaciji menja i približava vrednosti koji je pronašao algoritam brute force za dužinu puta.



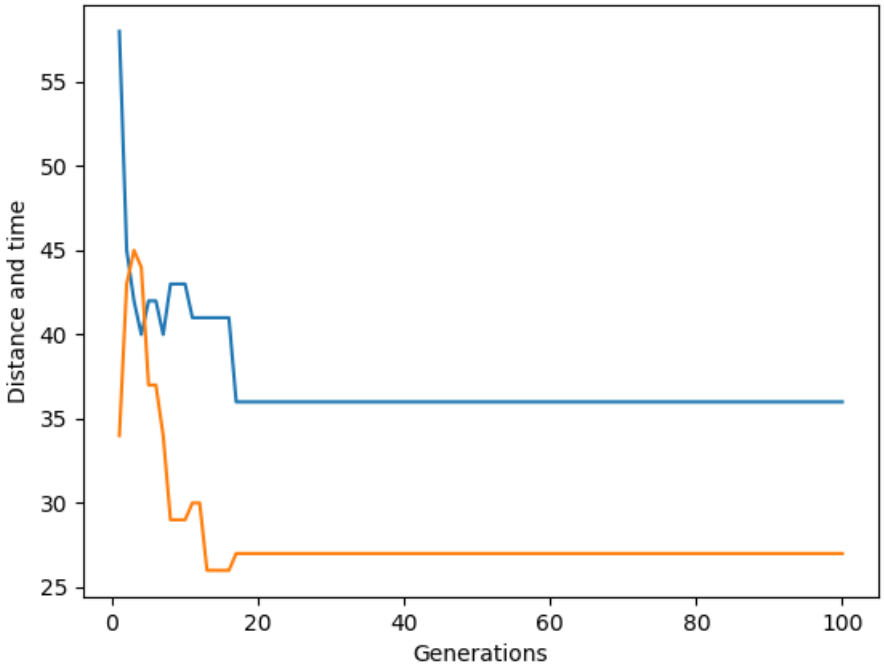
Slika 1: Dužina puta izabranog optimalnog rešenja kroz generacije

Na slici 2 možemo pratiti na koji način se izabrana vrednost optimalnog rešenja u svakoj generaciji menja i približava vrednosti koji je pronašao algoritam brute force za vreme potrebno da se pređe optimalan put.



Slika 2: Vreme potrebno da se pređe optimalan put kroz generacije

Slika 3 objedinjuje prethodne slike i prikazuje vrednosti obe dimenzije problema kroz generacije



Slika 3: Optimalno rešenje kroz generacije

Na sledećim tabelama prikazaćemo na primeru 11(tabela 1) i 10(tabela 2) gradova procenat da predloženi genetski algoritam za optimalno rešenje dobije neki od najboljih 5 rešenja dobijenih algoritmom brute force.

Tabela 1: Vrednosti u tabeli važe za posmatran problem 11 gradova, veličinom populacije 20, verovatnoćom mutacije 0.05,veličinom turnira 5 i brojem generacija 3000

Distanca	Vreme	Odabran za optimalno
29	32	27%
33	29	0%
32	30	46%
36	27	0%
33	30	0%

Tabela 2: Vrednosti u tabeli važe za posmatran problem 10 gradova, veličinom populacije 20, verovatnoćom mutacije 0.05,veličinom turnira 5 i brojem generacija 2500

Distanca	Vreme	Odabran za optimalno
30	29	17%
23	35	18%
26	33	32%
32	28	0%
27	33	0%

Tabela 3: Vreme izvršavanja programa

Broj gradova	Brute Force	Genetski algoritam
10	3.14 sec	1.5070 sec%
11	31.3 sec	1.81 sec%

Svi testovi(3)rađeni su na računaru sledećih karakteristika:
Operativni sistem: GNU/Linux
Procesor: Intel i7-6498DU 2.50ghz
Ram memorija: 8GB

4 Zaključak

Prikazani genetski algoritam na problemu trgovačkog putnika sa dodatim zahtevima za vreme može pronaći globalno optimalno rešenje sa velikom verovatnoćom. U cilju poboljšanja ovog algoritma moguće je inicijalnu populaciju generisati pažljivim odabirom permutacija, tako da pri konstrukciji permutacije gradove navodimo tako da njihova međusobna udaljenost ili vreme budu što je moguće manji [4].Ova poboljšanja u vidu inicijalne populacije mogu povećati vreme izvršavanja algoritma, ali su tačniji.

Literatura

[1] RajatKumar Palc Chiranjit Changdara, G.S.Mahapatrab. An efficient genetic algorithm for multi-objective solid travelling salesman problem under fuzziness, 2014. on-line at: https://www.sciencedirect.com/science/article/abs/pii/S2210650213000679?fbclid=IwAR3bJlMb-0O51Up-kRFKaJQ3brZ9Z4cMRAaqPsdNxqmj8XkXrBuVipnF5L0.

[2] Andries P. Engelbrecht. *Computational intelligence, second edition.* 2007.

[3] Ibrahim A. Hameed. Multi-objective Solution of Traveling Salesman Problem with Time, 2020. on-line at: https://www.researchgate.net/publication/331824516_Multi-objective_Solution_of_Traveling_Salesman_Problem_with_Time?fbclid=IwAR2vnsmFLYhjphtIPyrfIbbMHMgbp4Wuc4n3jmF00jPC5raP1VGfYchJtCA.

[4] Mei Ma and Hecheng Li. A hybrid genetic algorithm for solving bi-objectivetraveling salesman problems, 2017. on-line at: https://iopscience.iop.org/article/10.1088/1742-6596/887/1/012065/pdf?fbclid=IwAR2o9DPMsVBKjzJBgRS_ez-rwYbGRKSGOJZ8eOfRWtsgQkzL4qilV75-CiM.