Univerzitet u Beogradu, Matematički fakultet

Projekat iz Računarske inteligencije

Računanje najkraćeg puta dužine k u poligonu

Tijana Nikčević 77/2015
Tijana Živković 148/2015

SADRŽAJ

1. UVOD

Potrebno je pronaći najkracu putanju od tačke *S* do tačke *T* unutar poligona *P*, uz ograničenje da se putanja sastoji od najviše *k* ivica. U nastavku je putanja koja se sastoji od najviše k ivica označena terminom *k*-put.

Ulaz: Poligon P sa n celobrojnih koordinata temena i dve tačke s i t u poligonu P.

Izlaz: *K*-put izmedju tacaka s i t odnosno niz tacaka $p_0, ..., p_h$ koje se nalaze unutar poligona P

gde je h <= k tako da je p0 = s i pk = t, i za svako i za koje vazi 0 <= i < h vazi da je linija izmedju p_i i p_{i+1} unutar poligona P.

Mera: Euklidska duzina puta odnosno suma $\sum d(p_i, p_{i+1}), i=0,1,.., h-1$ gde je d Euklidsko rastojanje.

U radu *"Computing a shortest k-link path in a polygon"* autori J. S. B. Mitchell, C. Piatko i E. M. Arkin razmatraju problem pronalaženja najkraceg k-puta od tačke S do tačke T unutar prostog poligona P, unutar poligona koji može da sadrži rupe, kao i problem gde se minimizuje k i zaokreti na putanji.

Algoritam pronalaženja najkraceg k-puta unutar prostog poligona P do kojeg su došli izračunava k-put koji ima maksimalnu dužinu $(1 + \varepsilon)$ puta dužine najkraceg k-puta, za bilo koju toleranciju greške $\varepsilon > 0$, u vremenu $O(n^3k^3\log(Nk/\varepsilon^{1/k})$ gde je *N* najveća vrednost x koordinate među temenima poligona P. Za opštiji slučaj gde je dozvoljeno da poligoni imaju rupe, napravili su algoritam koji vraća najviše *2k*-put sa duzinom najviše duplo većom od dužine najkraćeg k-puta, u vremenu $O(kE^2)$, gde je *E* broj ivica u grafu vidljivosti formiranom od poligona. U trećoj varijanti problema količina zaokreta se meri kao integral duž putanje. Za ovaj problem su došli do algoritma koji radi u polinomijalnom vremenu.

Kod algoritma pronalaženja najkraceg k-puta unutar prostog poligona P, poligon se deli na manje delove na osnovu prevojnih tačaka krive najkraćeg puta, čime se dobija niz staza manjih konveksnih poligona-staza, od kojih se P sastoji. U okviru ovih manjih problema idalje moze da postoji eksponencijalni broj najkraćih i-putava, i oni se rešavaju daljom dekompozicijom, pri čemu se koristi binarna pretraga, svojsvta poligona kao i dinamičko programiranje.

U mnogim aplikacijama je potrebno izračunati optimalni put ali tako da ukupna zakrivljenost, broj ivica i krivudavost putanje budu što manji. Ukoliko se obrati pažnja samo na broj ivica, put sa minimalnim brojem ivica moze biti proizvoljne dužine u odnosu na (Euklidski) najkraći put. Zbog toga je nekad potrebno izračunati minimalni k-put u poligonu.

2. REŠENJE POMOĆU GENETSKOG ALGORITMA

Algoritam se sastoji iz uobičajenih delova genetskog algoritma. Na pocetku se generiše početni skup rešenja (jedinki), a zatim se u petlji iz skupa rešenja (jedinki) biraju ona koja će učestvovati u reprodukciji. Od jedinki za reprodukciju se kreiraju nove jedinke od kojih se onda kreira nova generacija.

Rešenje, odnosno *k*-put, se kodira kao lista *k-2* tačaka koje čine ivice *k*-puta. Početna tačka A i krajnja tačka B su izostavljene jer su one fiksne, problem se sastoji u pronalaženju ostalih tačaka. Svaka tačka je predstavljena parom x i y koordinate. Dakle rešenje k-puta *A-P1-P2-P3-B* se predstavlja kao [(*P1_x, P1_y*), (*P2_x, P2_y*), (*P3_x, P3_y*)].

Algoritam na ulazu prima broj *k*, početnu tačku A, krajnju tačku B i listu tačaka - temena poligona P. Na početku programa se proverava da li je duž AB unutar poligona. Ukoliko jeste nema potrebe za daljim računom i vraća se k-put u liniji AB. Ukoliko nije nastavlja se s algoritmom.

Generišu se lista sa svim tačkama sa celobrojnim koordinatama koje pripadaju poligonu P. Temena se dodaju više puta jer se često nalaze u najkraćem k putu pa je poželjno da se češće i biraju. Lista je sortirana po koordinatama (prvo x koordinati, zatim y).

Početna populacija odnosno skup početnih rešenja se formirmira dodavanjem pojedinačnih rešenja dok se ne dostigne potrebna veličina generacije. Svako od

pojedinačnih rešenja se pravi izborom *k-2* nasumičnih tačaka, pri čemu se tačke biraju iz liste tako da budu sortirane. Ukoliko algoritam treba da se

prilagodi za poligone sa rupama tačke treba birati bez redosleda. Umesto prilagođenosti kao u većini algoritama, za svaku jedinku (rešenje) se računa njena neprilagođenost. Svaka jedinka se čuva u klasi *Solution* koja se sastoji od njene kodirane reprezentacije i od njene neprilagođenosti.

Neprilagođenost rešenja (jedinke) se računa kao dužina *k*-puta, a ukoliko taj k-put nije unutar poligona računa se dodatna vrednost koja zavisi od broja preseka k-puta sa poligonom.

Biranje jedinki za reprodukciju vrši se turnirskom selekcijom. Svaka jedinka za reprodukciju se dobija kao najbolja od 20 slučajno izabranih iz generacije.

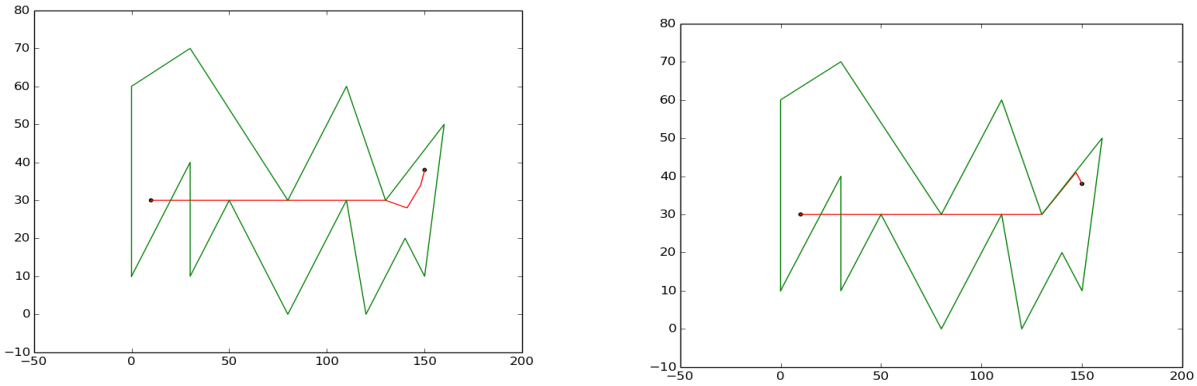
Nova generacija se od stare pravi korišćenjem operatora ukrštanja i mutacije. Ukrštanje je uniformno. Od dva roditelja se formiraju dva deteta. Za svaku od *k-2* tačke se slučajno biraju dve vrednosti. Ukoliko je druga vrednost manja od zadatog parametra *crossover_p_mix*, dete novu tačku dobija ukrštanjem koordinata tačaka roditelja na toj poziciji, a u suprotnom svako dete dobija po jednu celu tačku od jednog roditelja. Od prve slučajne vrednosti zavisi kojem detetu ide tačka (ili koordinata) od kog roditelja.

Mutacija se vrši nad svakom tačkom svakog novog rešenja sa verovatnoćom *mutation_rate*. Ukoliko je došlo do mutacije, tačka se zamenjuje nasumično izabranom tačkom iz liste dozvoljenih tačaka.

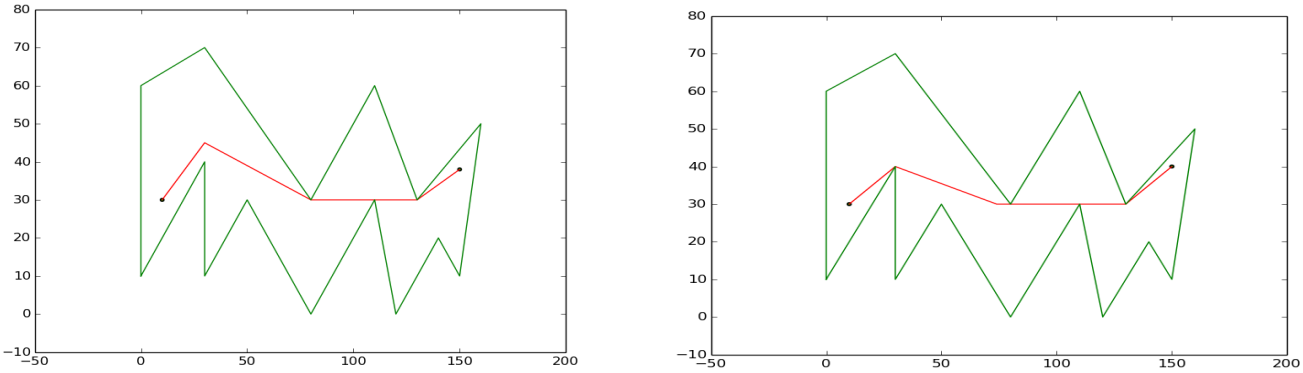
Nakon promene generacije ažurira se broj uzastopnih poslednjih iteracija u kojima su najbolje jedinke iz generacije imale istu neprilagodjenost.

Ukoliko su u poslednjih 5 iteracija najbolje jedinke imale istu neprilagođenost ili ako je dostignut maksimalan broj iteracija, algoritam se zaustavlja i vraća najbolju jedinku (rešenje) iz poslednje generacije.

Prikaz najboljeg rešenja po iteracijama na jednom primeru (k = 5):



Slika 1. Iteracija 0: vrednost funkcije 348.379722 i iteracija 1: vrednost funckije 373.833313133



Slika 2. Iteracija 3: vrednost funkcije 149.562212 i iteracija 14: vrednost funckije 145.711555

3. REŠENJE POMOĆU OPTIMIZACIJE ROJEM ČESTICA

Algoritam je rađen po uzoru na algoritam optimizacije rojem čestica koji je rađen na vežbama. Rešenje, odnosno *k*-put, se kodira kao lista *(k-2)*2* koordinata tačaka koje čine ivice *k*-puta. Početna tačka A i krajnja tačka B su izostavljene kao i kod genetskog algoritma. Za svaku tačku je u listi njena x i y koordinata na odgovarajućem mestu. Dakle rešenje k-puta *A-P1-P2-P3-B* se predstavlja listom: [*P1_x, P1_y*, *P2_x, P2_y*, *P3_x, P3_y*].

Algoritam na ulazu prima broj *k*, početnu tačku A, krajnju tačku B i listu tačaka - temena poligona P. Kao i kod predhodnog algoritma, na početku

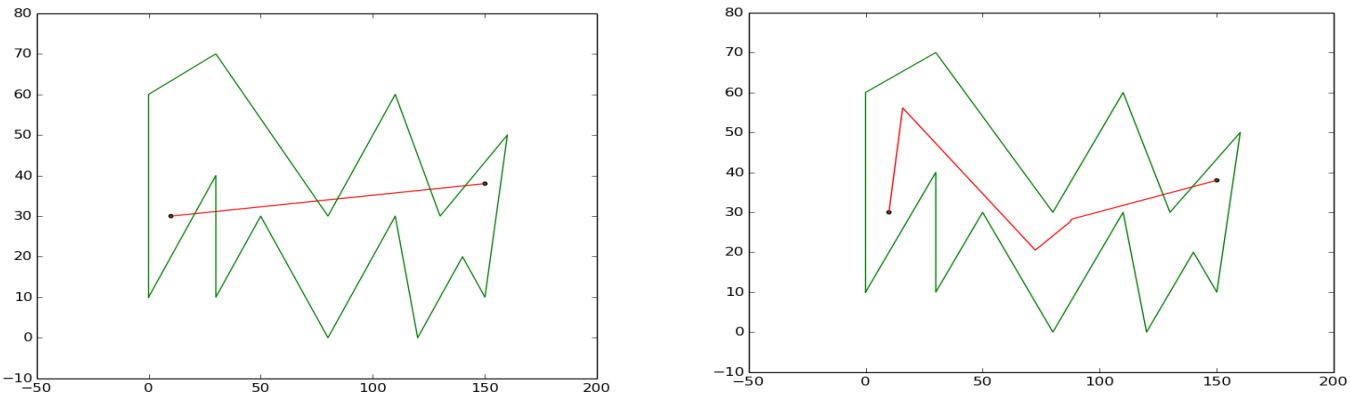
programa se proverava da li je duž AB unutar poligona i ukoliko jeste vraća se k-put u liniji AB. Ukoliko nije nastavlja se s algoritmom.

Početne vrednosti čestica se zadaju kao podjednako udaljene tačke na liniji AB. Funkcija koju algoritam pokušava da minimizuje je funkcija (ne)prilagođenosti iz genetskog algoritma. Početne vrednosti brzina su nasumično dobijene iz intervala dozvoljene brzine.

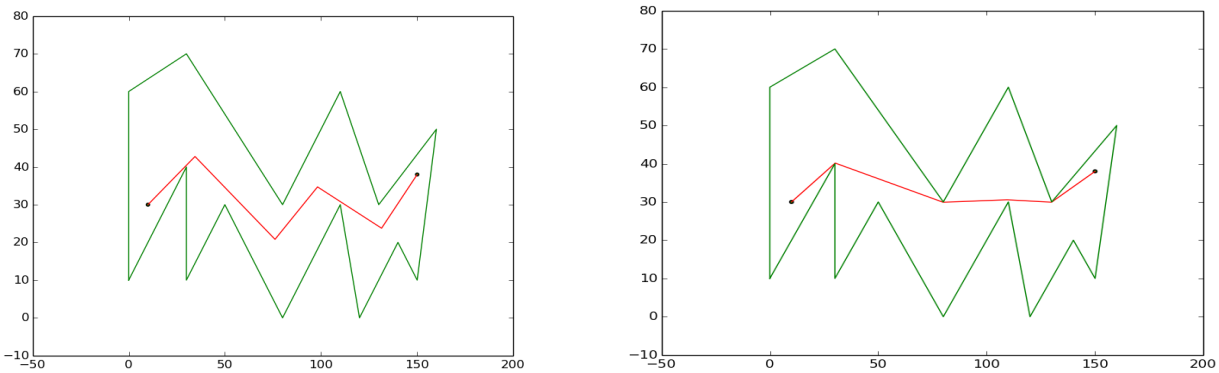
Čestice su predstavljene klasom *Particle* koja sadrži listu pozicije *position_i,* brzine *velocity_i*, i listu *pos_best_i* kojom je označena trenutna najbolja pozicija čestice iz roja. Uokviru klase se nalazi funkcija koja računa prilagodjenost za trenutnu pozicuju i ažurira najbolju poziciju ako je potrebno. Klasa sadrži i funkciju za ažuriranje brzine čestice, kao i funkciju za ažuriranje pozicije čestice na osnovu ažurirane brzine.

Nakon inicijalizacije roja program ulazi u glavnu petlju. U svakoj iteraciji se za svaku česticu u roju ažurira njena prilagođenost, a zatim se ažuriraju i brzine i pozicije čestica. Program se zaustavlja kada se dostigne maksimalan br iteracija ili kada se u poslednjih 20 iteracija ne menja vrednost najmanje greške.

Prikaz najboljeg rešenja po iteracijama na jednom primeru (k = 6):



Slika 3. Iteracija 0: vrednost funkcije 740.228385144 i iteracija 1: vrednost funckije 373.833313133



Slika 4. Iteracija 10: vrednost funkcije 159.18612095 i iteracija 30: vrednost funckije 145.032633377

4. ALGORITAM NALAŽNJA k-PUTA SA NAJMANJOM DUŽINOM U KONVEKSNOM POLIGONU

Programski jezik korišćen za implementaciju algoritma je python, uz odgovarajuće biblioteke za potrebe računanja i grafičkog prikaza, a to su: math, random, numpy, matplotlib i shapely.

Na samom početku, generiše se nasumicni poligon (preuzetom funkcijom generatePolygon), u našim primerima i kodu, on ima trideset temena. Zatim se takođe nasumično biraju dve tacke koje pripadaju ivici ili unutrašnjosti poligona. Cilj je da se pronađe minimialno rastojanje izmedju te dve tačke (nazvaćemo ih tačkom A i B), a da se to postigne povlačenjem najviše k duži unutar poligona. Maksimalna dužina k je unapred zadata.

U algoritmu ćemo početnom tačkom nazivati onu koja ima manju x-koordinatu, da bismo uvek pokretali algoritam s desna ulevo, radi veće jednostavnosti. Početna duž je baš ona između tačke A i B. Ukoliko ta duž ne preseca nijednu ivicu poligona, stajemo sa izvršavanjem algoritma i minimalna dužina je upravo dužina te duži od tačke A do tačke B. Ovo rešenje će svakako biti prihvatljivo, jer je minimalna dužina k baš jedan.

U prvom koraku, ukoliko početna duž seče neku od ivica poligona, tražimo tačke preseka sa svim ivicama, biramo prvu od njih. U odnosu na poziciju te presečne tačke, biramo najbliže teme poligona, kako bismo napravili putanju do njega I na taj način se kretali najkraćim mogućim putem.

U odnosu na to da li je nađeno teme bliže početnoj ili krajnjoj tački, odlučuje se da li će to teme biti novi početak ili kraj za nastavak puta i izvršavanje algoritma. Tačke seređaju u listu, u redosledu trenutne putanje. Ovaj deo algoritma se nastavlja sve dok se ne stigne do tačke B. U svakom koraku nalaženja najbližeg temena, povecavamo velicinu trenutnog puta, tj. broj duži putanje, kao i trenutnu njenu dužinu.

Kao sto je već rečeno, cilj je naći najkraći put, ali koristeći što manji broj duži putanje zbog ograničenja. Naredni korak je nalaženje puta između svake dve nesusedne tačke prvobitne putanje, kako bi se smanjio broj duži i pronašao kraći put. Ovo se postiže funkcijom algorytm2. Proverava se da li duž između svake dve nesusende tačke - line, seče neku od ivica poligona. Ako je to slučaj, duž ne može da bude prihvaćena, a ako važi suprotno i uz to je dužina ove nove putanje manja, ažuriramo skup tačaka krajnje putanje, kao I ukupne dužine.

U trećem koraku se proveravava da li je ovim zadovoljeno ograničenje za k duži, on je opisan funkijom k_algorytm. Najpre se pravi lista gde će se čuvati sve liste mogućih koordinata nove putanje, kao i lista njima odgovarajućih dužina. Od parova tačaka dobijenih iz prethodnog koraka, gledamo presek pravih. Ukoliko je presečna tačka baš teme početnog poligona, neće biti prihvaćena. Slično, ako presečna tačka ne pripada poligonu, takođe neće biti prihvaćena. Ako nije u pitanju nijedan od ova dva slučaja, a nova dužina je manja, presečna tačka će postati nova tačka putanje, a biće izbačene odgovarajuće dve koje pripadaju pravama uz pomoć kojih je tražen presek. Ova tačka preseka će biti prihvaćena i u slučaju da zbog ograničenja k mora da se smanji broj duži k-puta.

Sve moguće novonastale putanje, kao i njihove dužine, biće smeštene u prethodno pomenute liste. Nove tacke biće one čiji je k-put najmanje dužine. U svakom koraku izbacivanja dve i ubacivanjem nove ogovarajuće tačke preseka, smanjujemo ukupan broj duži k-puta.

Nakon ovog koraka, trebalo bi ponavljati iznova drugi i treći korak da bi se dobio još kraći put.

Uz pomoć funkcije generatePolygon generiše se nasumični poligon i funkcijom random_point_within dve nasumične tačke unutar njega.

Korišćene su i pomoćne funkcije, intersect – koja vraća broj presečnih tačaka poligona i prosleđene duži, a ako ih ima, takođe vraća i koordinate presečne tačke, kao i redni broj ivice na kojoj je nadjen taj presek.

Funkcijom nearest_vertice dobijaju se koordinate najbližeg temena od presečne tačke, ali onog koji je bliži centru poligona.

Na samom kraju, kao izlaz se dobija broj duži, tj. k, koordinate tačaka puta, kao i ukupna dužina k-puta.

Ako nije moguće ispuniti uslov maksimalnog k, ispisaće se odgovarajuća poruka.

Grafički su prikazane putanje u koracima kao i krajnji put.

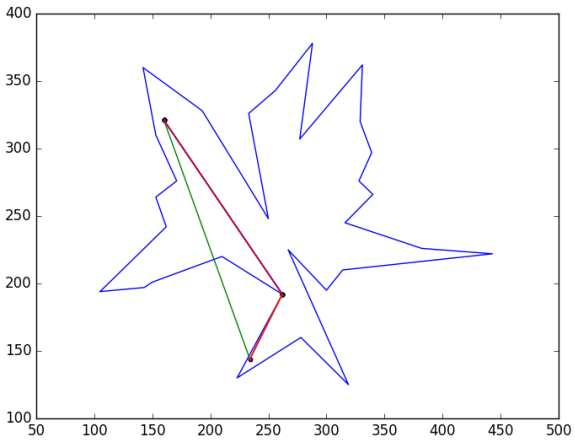
5. TESTIRANJA

U test primerima su uzete navedene tačke kao početne i krajnje i prikazan odgovarajući izlaz nakon izvrsavanja svakog od tri algoritma. Korišćena su ista temena poligona za sva tri.

Prvi test primer:

Koordinate početne tačke: x = 179, y = 327

Koordinate krajnje tačke: x = 325, y = 343



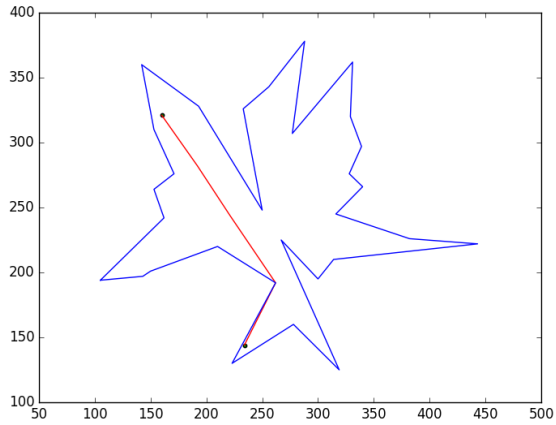

```
Pocetna tacka:
[ 160.  321.]
Krajnja tacka:
[ 234.  144.]
Broj k duzi:
2
Duzina linije:
220.023416842
Konacna temena:
[160.0, 262.0, 234.0]
[321.0, 192.0, 144.0]
```

Slika 5. Algoritam konstrukcijom

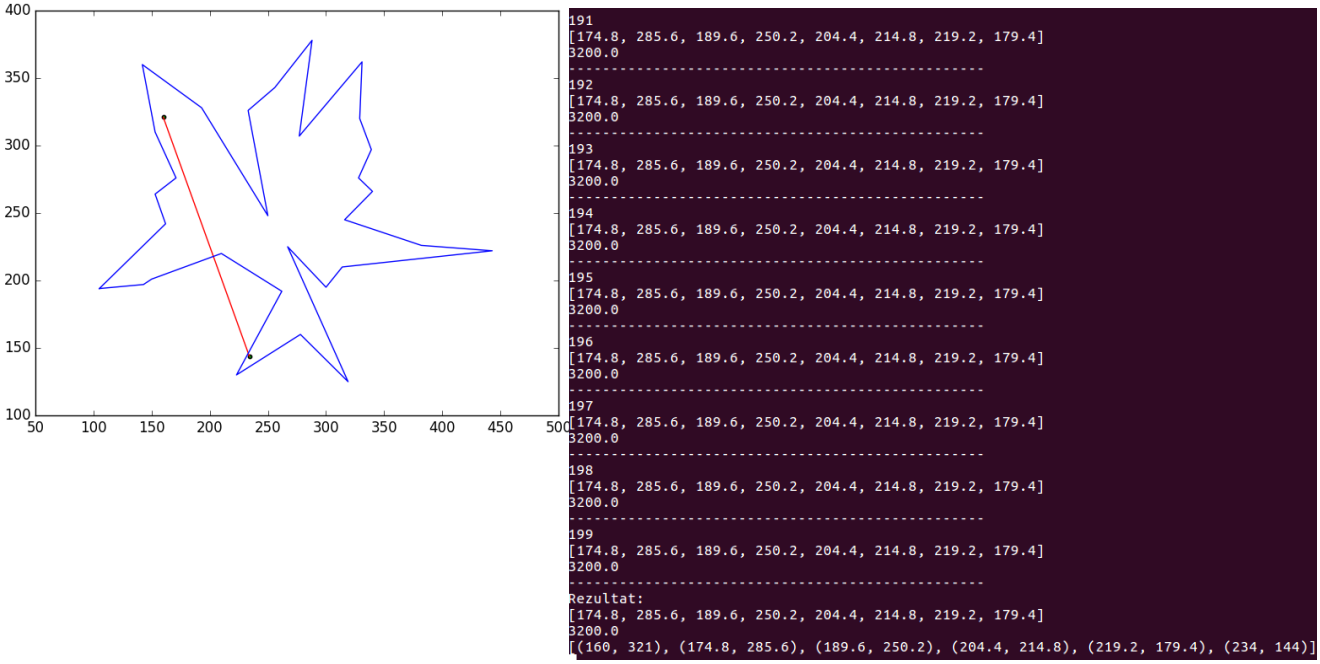
```
Iteration: 0
Reproduction chromos sum unfitness: 1047628
Top solution: 257.648901
------------------------------------------------------------
------------------------------------------------------------
Iteration: 1
Reproduction chromos sum unfitness: 726228
Top solution: 222.528072
------------------------------------------------------------
------------------------------------------------------------
Iteration: 2
Reproduction chromos sum unfitness: 537884
Top solution: 221.047935
------------------------------------------------------------
------------------------------------------------------------
Iteration: 3
Reproduction chromos sum unfitness: 419864
Top solution: 220.125935
------------------------------------------------------------
------------------------------------------------------------
Iteration: 4
Reproduction chromos sum unfitness: 284499
Top solution: 220.107662
------------------------------------------------------------
------------------------------------------------------------
Iteration: 5
Reproduction chromos sum unfitness: 238278
Top solution: 220.046992
------------------------------------------------------------
------------------------------------------------------------
Iteration: 6
Reproduction chromos sum unfitness: 229707
Top solution: 220.038775
------------------------------------------------------------
```

```
Iteration: 7
Reproduction chromos sum unfitness: 232139
Top solution: 220.038775
------------------------------------------------------------
------------------------------------------------------------
Iteration: 8
Reproduction chromos sum unfitness: 232138
Top solution: 220.038775
------------------------------------------------------------
------------------------------------------------------------
Iteration: 9
Reproduction chromos sum unfitness: 232003
Top solution: 220.038775
------------------------------------------------------------
------------------------------------------------------------
Iteration: 10
Reproduction chromos sum unfitness: 232796
Top solution: 220.038775
------------------------------------------------------------
------------------------------------------------------------
Iteration: 11
Reproduction chromos sum unfitness: 228722
Top solution: 220.038775
------------------------------------------------------------
------------------------------------------------------------
Iteration: 12
Reproduction chromos sum unfitness: 226484
Top solution: 220.038775
------------------------------------------------------------
------------------------------------------------------------
Iteration: 13
Reproduction chromos sum unfitness: 226215
Top solution: 220.038775
------------------------------------------------------------
------------------------------------------------------------
Iteration: 14
Reproduction chromos sum unfitness: 228652
Top solution: 220.038775
------------------------------------------------------------
Broj preseka:
0
Solution: [(192, 282), (221, 244), (262, 192)] unfitness: 220
```

Slika 6.1.



Slika 6.2. Genetski algoritam



Slika 7. Algoritam rojem čestica

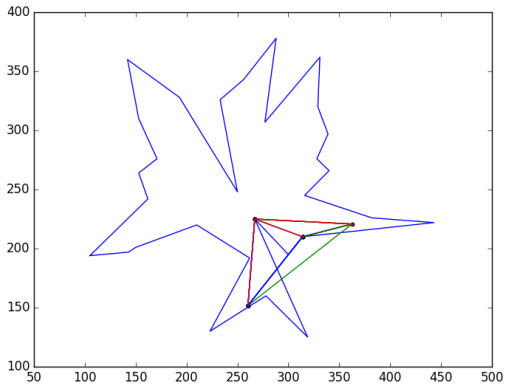
U prvom test primeru su prikazani poligoni i njihove krajnje najoptimalnije putanje. Prvi algoritam daje dobar rezultat nalaženjem presečnih tačaka i najbližeg temena poligona i time najkraći put od početne do krajnje tačke. Isto tako, genetski algoritam je našao najkraći put, a za njega su prikazane i sve

itracije kroz koje je prolazio. Prikazan je i rezultat dobijen algoritmom primenjivanj optimizacije rojem čestica.

Drugi test primer:

Koordinate početne tačke: x = 362.55631868, y = 220.7947158

Koordinate krajnje tačke: x = 260.32511082, y = 151.84525141

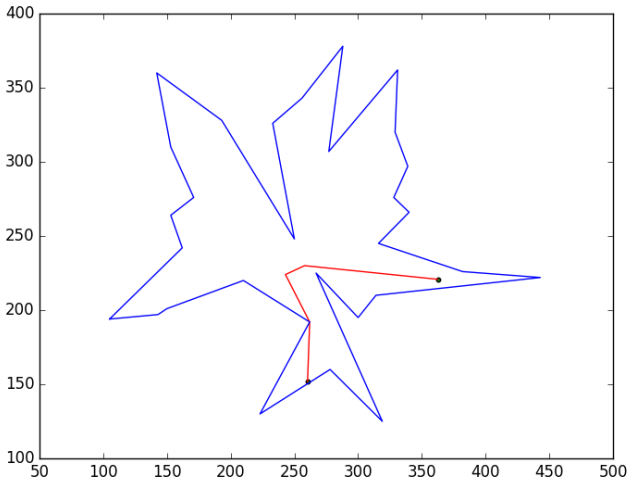


Slika 8. Algoritam konstrukcijom

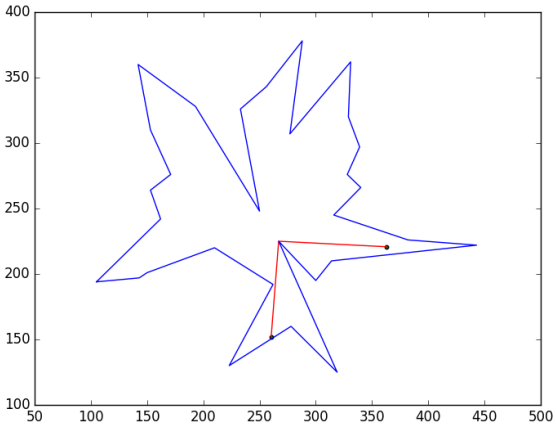
```
------------------------------------------------------------
Iteration: 0
Reproduction chromos sum unfitness: 1026485
Top solution: 325.090505
------------------------------------------------------------
------------------------------------------------------------
Iteration: 1
Reproduction chromos sum unfitness: 665867
Top solution: 263.904217
------------------------------------------------------------
------------------------------------------------------------
Iteration: 2
Reproduction chromos sum unfitness: 557505
Top solution: 225.127763
------------------------------------------------------------
------------------------------------------------------------
Iteration: 3
Reproduction chromos sum unfitness: 539567
Top solution: 225.127763
------------------------------------------------------------
------------------------------------------------------------
Iteration: 4
Reproduction chromos sum unfitness: 336826
Top solution: 215.530283
------------------------------------------------------------
------------------------------------------------------------
Iteration: 5
Reproduction chromos sum unfitness: 252212
Top solution: 213.860231
------------------------------------------------------------
------------------------------------------------------------
Iteration: 6
Reproduction chromos sum unfitness: 238825
Top solution: 212.586995
------------------------------------------------------------
```

```
------------------------------------------------------------
Iteration: 7
Reproduction chromos sum unfitness: 238048
Top solution: 206.638132
------------------------------------------------------------
------------------------------------------------------------
Iteration: 8
Reproduction chromos sum unfitness: 237161
Top solution: 206.638132
------------------------------------------------------------
------------------------------------------------------------
Iteration: 9
Reproduction chromos sum unfitness: 283065
Top solution: 206.638132
------------------------------------------------------------
------------------------------------------------------------
Iteration: 10
Reproduction chromos sum unfitness: 246815
Top solution: 205.157667
------------------------------------------------------------
------------------------------------------------------------
Iteration: 11
Reproduction chromos sum unfitness: 226362
Top solution: 201.285619
------------------------------------------------------------
------------------------------------------------------------
Iteration: 12
Reproduction chromos sum unfitness: 239732
Top solution: 198.521506
------------------------------------------------------------
------------------------------------------------------------
Iteration: 13
Reproduction chromos sum unfitness: 233514
Top solution: 198.521506
------------------------------------------------------------
------------------------------------------------------------
Iteration: 14
Reproduction chromos sum unfitness: 228731
Top solution: 198.521506
------------------------------------------------------------
Broj preseka:
0
Solution: [(258, 230), (243, 224), (262, 192)] unfitness: 198
```

Slika 9.1.



Slika 9.2. Genetski algoritam




```
195
[359.4331038687045, 220.93216337711888, 267.0, 224.99999999999994, 261.65392693417334,
73126015, 260.98863727710886, 159.11729228721705]
169.107445163
-------------------------------------------------
196
[359.4331038687045, 220.93216337711888, 267.0, 224.99999999999994, 261.65392693417334,
73126015, 260.98863727710886, 159.11729228721705]
169.107445163
-------------------------------------------------
197
[359.4331038687045, 220.93216337711888, 267.0, 224.99999999999994, 261.65392693417334,
73126015, 260.98863727710886, 159.11729228721705]
169.107445163
-------------------------------------------------
198
[359.4331038687045, 220.93216337711888, 267.0, 224.99999999999994, 261.65392693417334,
73126015, 260.98863727710886, 159.11729228721705]
169.107445163
-------------------------------------------------
199
[359.4331038687045, 220.93216337711888, 267.0, 224.99999999999994, 261.65392693417334,
73126015, 260.98863727710886, 159.11729228721705]
169.107445163
-------------------------------------------------
Rezultat:
[359.4331038687045, 220.93216337711888, 267.0, 224.99999999999994, 261.65392693417334,
73126015, 260.98863727710886, 159.11729228721705]
169.107445163
[(362.55631868, 220.7947158), (359.4331038687045, 220.93216337711888), (267.0, 224.9999
), (261.65392693417334, 166.40866673126015), (260.98863727710886, 159.11729228721705),
982, 151.84525141)]
```

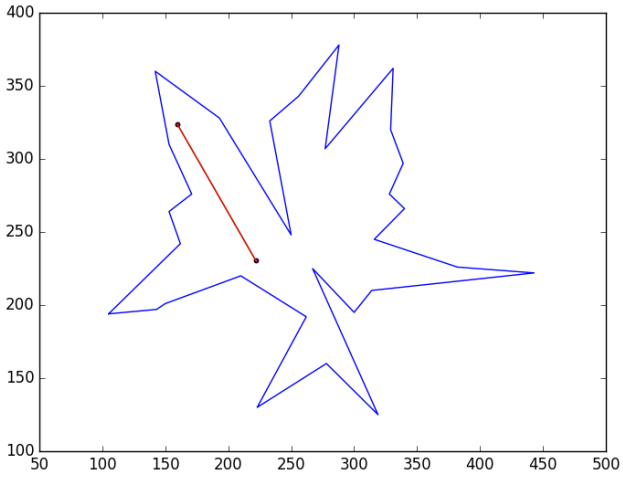
Slika 10. Algoritam rojem čestica

U drugom test primeru u su korišćene nove koordinate početne i krajnje tačke. Primecujemo da je naš algoritam i algoritam optimizacijom roja čestica dao isti rezultat, dok je genetski dao duži put, koristeći uz to i više duži za konstruisanje krajnje putanje.

Treći test primer:

Koordinate početne tačke: x = 221.70344421, y = 230.73104356

Koordinate krajnje tačke: x = 159.60986738, y = 323.70420134




```
cetna tacka:
221.70344421  230.73104356]
ajnja tacka:
159.60986738  323.70420134]
oj k duzi:

zina linije:
1.801701021
nacna temena:
59.60986738, 221.70344420999999]
23.70420134, 230.73104355999999]
```

Slika 11. Algoritam konstrukcijom

```
----------------------------------------------------------
Iteration: 0
Reproduction chromos sum unfitness: 1008969
Top solution: 152.906427
----------------------------------------------------------
----------------------------------------------------------
Iteration: 1
Reproduction chromos sum unfitness: 639899
Top solution: 138.297880
----------------------------------------------------------
----------------------------------------------------------
Iteration: 2
Reproduction chromos sum unfitness: 319886
Top solution: 126.724137
----------------------------------------------------------
----------------------------------------------------------
Iteration: 3
Reproduction chromos sum unfitness: 172490
Top solution: 118.918228
----------------------------------------------------------
----------------------------------------------------------
Iteration: 4
Reproduction chromos sum unfitness: 155606
Top solution: 116.614361
----------------------------------------------------------
----------------------------------------------------------
Iteration: 5
Reproduction chromos sum unfitness: 136804
Top solution: 115.943838
----------------------------------------------------------
----------------------------------------------------------
Iteration: 6
Reproduction chromos sum unfitness: 134612
Top solution: 113.573004
----------------------------------------------------------
```

```
----------------------------------------------------------
Iteration: 7
Reproduction chromos sum unfitness: 130367
Top solution: 112.510590
----------------------------------------------------------
----------------------------------------------------------
Iteration: 8
Reproduction chromos sum unfitness: 127685
Top solution: 112.510590
----------------------------------------------------------
----------------------------------------------------------
Iteration: 9
Reproduction chromos sum unfitness: 122196
Top solution: 112.510590
----------------------------------------------------------
----------------------------------------------------------
Iteration: 10
Reproduction chromos sum unfitness: 124725
Top solution: 112.510590
----------------------------------------------------------
----------------------------------------------------------
Iteration: 11
Reproduction chromos sum unfitness: 123498
Top solution: 112.510590
----------------------------------------------------------
----------------------------------------------------------
Iteration: 12
Reproduction chromos sum unfitness: 124677
Top solution: 111.859519
----------------------------------------------------------
----------------------------------------------------------
Iteration: 13
Reproduction chromos sum unfitness: 123508
Top solution: 111.859519
----------------------------------------------------------
----------------------------------------------------------
Iteration: 14
Reproduction chromos sum unfitness: 124388
Top solution: 111.859519
----------------------------------------------------------
Broj preseka:
0
Solution: [(209, 248), (194, 273), (194, 273)] unfitness: 111
```

Slika 12. Genetski algoritam

```
[(221.70344421, 230.73104356), (209.284728844, 249.32567511599999), (196.86601347799999, 267.9203
06672), (184.447298112, 286.514938228), (172.02858274599998, 305.109569784), (159.60986738, 323.7
0420134)]
```

Slika 13. Algoritam rojem čestica

Treći test primer je bazni slučaj, kada nema presečnih tačaka početne prave sa poligonom i tada sva tri daju dobre rezultate.

U narednoj tabeli prikazani su rezultati sva tri algoritma na 50 nasumično izabranih poligona, sa nasumično zadatim tačkama unutar njih. Prikazano je vreme trajanja programa u milisekundama, dužina puta kao i da li je pronađen k-put unutar poligona.

41	180.283000	163.696609149	√	819.284000	190.313693	√	481.419000	163.696612	X
42	182.148000	123.293190885	√	229.599000	126.836728	√	885.373000	300.052853	√
43	17.114000	141.670553394	√	707.525000	141.670553	√	2.517000	141.670553	√
44	18.866000	255.718227681	√	572.600000	255.718228	√	2.618000	255.718228	√
45	181.363000	218.732526374	√	134.317000	229.851122	√	496.593000	218.733512	√
46	17.180000	102.846812333	√	883.283000	102.846812	√	2.659000	102.846812	√
47	18.590000	108.126506728	√	336.799000	108.126507	√	2.568000	108.126507	√
48	402.399000	111.653501944	√	522.091000	131.005016	√	699.017000	111.663140	√
49	180.233000	111.177771865	√	250.235000	168.174953	√	739.615000	111.177780	√
50	17.098000	102.312872003	√	150.738000	102.312872	√	2.524000	102.312872	√

	Algoritam konstrukcijom	Genetski	PSO
Prosecno vreme (sec)	0.12615298	11.68	6.1
Broj neuspesnih	0	0	1

5. ZAKLJUČAK

Na prostim poligonima, kod kojih postoji centar poligona, najbolje se ponaša algoritam

koji konstruiše put. Genetski algoritam i algoritam optimizacije rojem čestica su sporiji, alioptimizacije rojem čestica, ali za razliku od njega uspešno radi i kada da bi se došlo do rešenja put treba da prođe kroz tačno određenu tačku ili veoma uzak deo na nekom mestu. Oba algoritma bi mogla da se poboljšaju menjanjem funkcija cilja da umesto broja preseka k- puta sa poligonom u obzir uzima tačnu dužinu puta van poligona.

6. LITERATURA

1. https://www.computer.org/csdl/proceedings/focs/1992/2900/00/0267794.pdf