**Univerzitet u Banjoj Luci**

**Elektrotehnički fakultet**

**Katedra za automatiku**

**Metodi vještačke inteligencije**

Izvještaj o urađenom projektnom zadatku

**Genetički algoritmi**

Student:

Želimir Maletić, 11125/18

# Opis problema

Problem koji je potrebno riješiti je određivanje globalog minimuma funkcije sa dvije promjenljive čiji je analitički oblik dat relacijom 1.0.

(1.0)

Pored gore navedenog problema, potrebno je odrediti i globalni maksimum krive koja se dobije pri presjeku funkcije z(x,y) i ravni, a čiji se analitički oblik dobije kada u jednačinu (1.0) uvrstimo x=0 čime dobijamo sljedeću krivu datu formulom 1.1.

(1.1)

# Rješenje

Prilikom rješavanja problema korišteno je binarno kodovanje decimalnih brojeva[[1]](#footnote-2). Kod ovog načina kodovanja potrebno je podijeliti interval (u kojem se nalaze sve vrijednosti koje želimo kodovati) na dovoljno male dijelove. Širina dijelova na koje se interval dijeli zavisi od prezciznosti. Ukoliko želimo da postignemo preciznost *p* potrebno je da širina intervala bude jednaka (u daljem tekstu elementarni interval). Svaki broj iz manjeg intervala mijenja se donjom granicom tog intervala. Sada se svaki decimalni broj koduje na način da binarno kodujemo broj elementarnih intervala koji se nalaze unutar intervala [donjaGranica, decimalniBrojZaKodovanje]. Možemo izračunati i koliko je bita potrebno da se decimalni brojevi koduju sa željenom preciznošću što je dato u relaciji 2.1 .

(2.1)

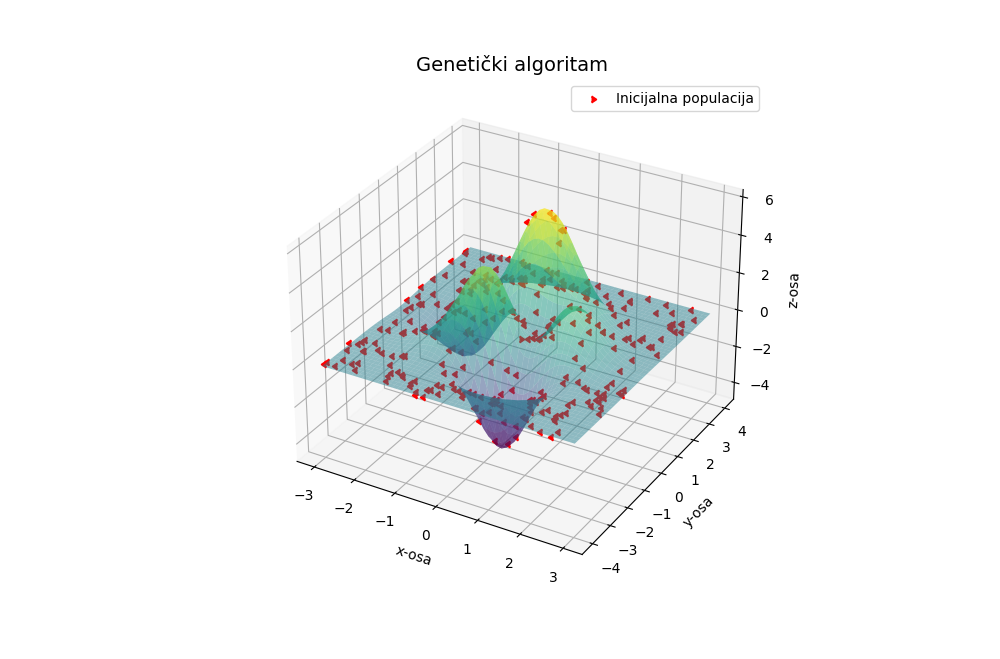
Broj intervala (*bi*) koji se nalaze u intervalu željenog decimalnog broja *x* dobije se kada interval [Gd,x] podijelimo sa širinom elementarnog intervala. Matematički zapis ovog postupka dat je u relaciji 2.2.

(2.2)

Ukoliko znamo kod *bi,* onda možemo da odredimo decimalni broj tako što na donju granicu dodamo *bi* širina elementarnog intervala. Matematički zapis ovog postupka dat je u relaciji 2.3.

(2.3)

Naredni korak u algoritmu jeste biranje početne populacije. Svaka generacija se sastoji od predefinisanog broja hromozoma. Hromozom[[2]](#footnote-3) je logička reprezentacija tačke prostora u kojem tražimo maksimum funkcije. Svaki hromozom karakteriše jedna koordinata sa x-ose i jedna koordinata sa y-ose. Pored ovih podataka u hromozomu se nalaze i binarne reprezentacije ovih koordinata. Prilikom odabira koordinata korišten je pseudo-slučajan generator brojeva koji generiše broj u intervalu od [0,1]. Ovako generisan pseudo-slučajan broj se preslikava[[3]](#footnote-4) u drugi broj iz intervala nad kojim je definisana jedna od osa. Dakle, u jednoj iteraciji kreiranja[[4]](#footnote-5) inicijalne populacije se generišu dva pseudo-slučajna broja. Prvi broj se transformiše u broj iz intervala nad kojim je definisana x-osa. Drugi broj se transformiše u broj iz intervala nad kojim je definisana y-osa. Oba transformisana broja, kao i njihove binarne reprezentacije, se dodijele hromozomu. Prilikom generisanja inicijalne populacije računa se i vrijednost funkcije prilagođenosti za svaki hromozom o čemu će biti više riječi u narednom paragrafu. Ovaj postupak ponavljamo u zavisnosti od željenog broja hromozoma u inicijalnoj populaciji. Nakon ovog koraka imamo inicijalnu populaciju hromozoma koji ulaze u naredne faze genetičkog algoritma. Primjer inicijalne populacije dat je na slici 2.0.



Slika 2.0 Primjer prikaza inicijalne populacije

Kao što je pomenuto u prethodnom paragrafu, za svaki od hromozoma računa se i vrijednost funkcije prilagođenosti. Ta vrijednost se takođe čuva u hromozomu. Funkcija prilagođenosti[[5]](#footnote-6) daje nenegativne vrijednosti za svaki od hromozoma. Potrebno je i da funkcija daje veće vrijednosti za rješenja koja su bolja, u konkretnom problemu rješenja sa manjom vrijdnosti funkcije z(x,y), a manje vrijednosti za lošija rješenja. Funkcija prilagođenosti koja je korištena u ovom rješenju data je relacijom 2.4.

(2.4)

Pri čemu je maxxi maksimalna vrijednost funkcije z(x,y) u generaciji. Relacijom 2.4. se funkcija za koju tražimo minimum rotira oko x-ose i traslira. Na ovaj način funkcija prilagođenosti ff(x) daje maksimalnu nenegativnu ocjenu dobrote za najmanju vrijednost funkcije z(x,y), a najmanju ocjenu dobrote za najveću vrijednost funkcije z(x,y) u generaciji. Pored računanja funkcije prilagođenosti za svaki od hromozoma, moguće je izračunati[[6]](#footnote-7) ocjenu populacije preko relacije 2.5.

(2.5)

Računanje ocjene populacije svodi se na sumiranje pojedinačnih ocjena dobrote svakog od hromozoma u generaciji. Treba napomenuti da veća ocjena populacije ne znači da se u toj populaciji nalazi i optimalnije rješenje. Dakle, moguće je da se desi da u populaciji koja ima manju ocjenu prilagođenosti populacije imamo rješenje koje je optimalnije od rješenja iz populacije koja ima veću vrijednost ocjene populacije. Prilikom iteracija se generacije mijenjaju novim. U ovom procesu se može desiti da generaciju koja ima bolje rješenje zamijenimo generacijom sa lošijim rješenjem. U ovom rješenju je implementirana posebna funkcija koja vodi računa da se zamjena generacija vrši samo pod uslovom da nova generacija ima bolji ekstremum od trenutne generacije. U Listingu 2.0. je prikazana funkcija populationChangeCondition() koja provjerava da li je ekstremum u novonastaloj generaciji bolji od ekstremuma u trenutnoj generaciji. Funkcija je implementirana tako da može da računa uslov za maksimum i minimum. Funkcija može da zahtijeva da se generacija mijenja samo kada u novoj generaciji postoji bolji ekstremum i kada je ocjena dobrote nove populacije veća od ocjene dobrote trenutne populacije. U Listingu 2.0. je drugi dio uslova zakomentarisan (nalazi se iza znaka #). Razlog ovome je sljedeći: prilikom zamjena generacija kroz iteracije primijećeno je da se rijetko javlja nova generacija koja ima i bolji ekstremum i bolju ocjenu populacije. Stoga, pristup u kojem funkcija provjerava oba uslova je pogodan kada imamo veliki broj iteracija, jer je tada i veća vjerovatnoća da se generiše populacija koja zadovoljava oba uslova. U slučaju malog broja iteracija dovoljno je provjeravati samo prvi uslov kao što je urađeno u Listingu 2.0.

Listing 2.0

|  |
| --- |
| #When to change population?  def populationChangeCondition(localExtremum, globalExtremum, subPopulationScore, populationScore):  if(param.EXTREMUM == "max"):  if(localExtremum >= globalExtremum ): #and subPopulationScore >= populationScore):  return True  if(param.EXTREMUM == "min"):  if(localExtremum <= globalExtremum): #and subPopulationScore >= populationScore):  return True  return False |

Prilikom opisa narednih faza algoritma u sljedećim paragrafima treba imati na umu da se ove faze algoritma nalaze unutar petlje. Dakle, kroz sljedeće faze se prolazi u predefinisanom broju iteracija u cilju dobijanja novih populacija koje imaju hromozome sa boljim ocjenama funkcije dobrote. Naredne faze vrše transformacije nad trenutnom populacijom i na kraju ocjenjuju koliko su te transformacije uspješne. Ukoliko se prilikom transformacija dobije bolja populacija, onda se trenutna populacija mijenja novonastalom.

Prva faza u iteraciji jeste faza selekcije[[7]](#footnote-8). U ovoj fazi se iz trenutne populacije biraju hromozomi koji će ući u naredne faze algoritma. Biranje hromozoma iz populacije je moguće izvesti na nekoliko načina. U ovom rješenju su implementirane tri vrste selekcije opisane u nastavku.

* Ruletska selekcija

U ovom načinu selekcije se za svaki od hromozoma računa vjerovatnoća koja se dobija tako što vrijednost funkcije prilagođenosti svakog od hromozoma u populaciji dijelimo sa ukupnom ocjenom prilagođenosti populacije. Na taj način dobijamo vjerovatnoće koje su proporcionalne vrijednostima funkcije prilagođenosti za svaki hromozom. Dakle, hromozomi koji imaju veće vrijdnosti funkcije prilagođenosti imaju i veće vjerovatnoće izbora. Ovakav način selekcije je pogodan, jer upravo oni hromozomi koji imaju veće vrijednosti funkcije prilagođenosti su poželjni da budu roditelji za narednu generaciju hromozoma. Sada se vjerovatnoće hromozoma posmatraju kao odjeljci na točku ruleta pri čemu je širina odjeljka upravo proporcionalna vjerovatnoći hromozoma. Dodjeljivanje vjrovatnoća hromozomima vrši se na način da se prvom hromozomu dodijeli vjrovatnoća koja mu je izračunata u fazi računanja vjerovatnoća. Narednom hromozomu se dodjeljuje vrijednost koja mu je izračunata plus vrijednost vjerovatnoće prethodnog hromozoma. Ovaj postupak se ponavlja za sve naredne hromozome u populaciji. Nakon određivanja i dodjeljivanja vjerovatnoća hromozomima pristupa se simulaciji okretanja točka ruleta. Ovo je implementirano na način da se ruletski točak okreće onoliko puta kolika je i veličina populacije. Dakle, veličina populacije se ne mijenja u toku izvršavanja algoritma. Okretanje točka se zapravo predstavlja generisanjem pseudo-slučajnog broja u intervalu [0,1]. Na osnovu dobijenog broja se zaključuje kojem intervalu vjerovatnoća generisani broj pripada i hromozom koji odgovara tom intervalu se bira za narednu populaciju. Nakon ove faze imamo međupopulaciju hromozoma koji su izabrani za naredne faze algoritma.

* Turnirska selekcija

Kod turnirske selekcije se hromozomi selektuju na način da se kroz iteracije, kojih ima onoliko koliko želimo da hromozoma bude u narednoj generaciji, slučajnim odabirom selektuje određen broj hromozoma. Iz tog novog podskupa se uzima onaj hromozom koji ima najveću vrijdnost funkcije prilagođenosti. Postupak je veoma jednostavan pa je iz toga razloga najčešće korišten metod selekcije[[8]](#footnote-9). Broj hromozoma K koje biramo iz populacije u podskup može da varira. Taj broj određuje selekcioni pritisak. Najčešće se koristi binarni turnirski postupak (K=2).

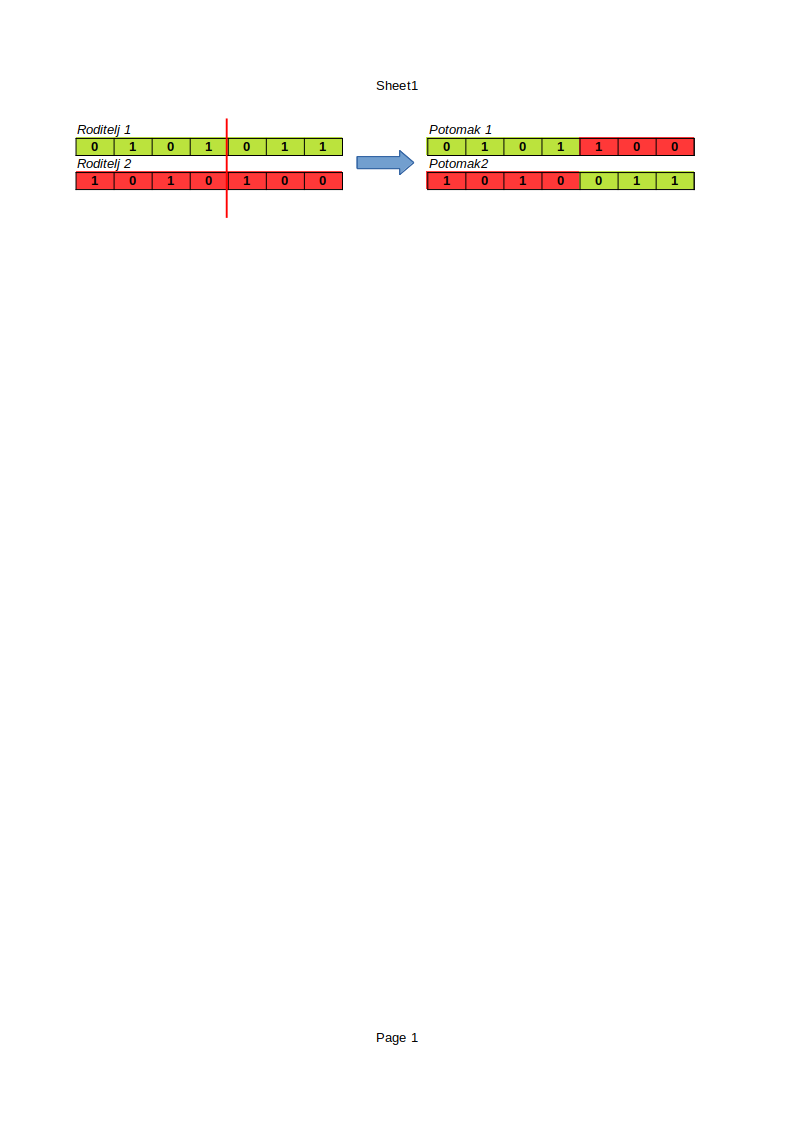
* Elitistička selekcija

Kod elitističke selekcije postupak biranja hromozoma za narednu generaciju je veoma jednostavan. U ovom rješenju je implementiran na način da se populacija hromozoma sortira po vrijednosti funkcije prilagođenosti u opadajućem redoslijedu, nakon čega se prva polovina hromozoma dva puta kopira u međupopulaciju koja ulazi u naredne faze algoritma.

Druga faza u genetičkom algoritmu jeste faza rekombinacije[[9]](#footnote-10). U ovoj fazi će se neki od hromozoma ukrstiti i proizvesti nove hromozome, dok će drugi hromozomi ostati nepromjenjeni. U ovom rješenju proces rekombinacije započinje sa postupkom „miješanja karata“. U svakoj od iteracija se generišu dva slučajna broja iz intervala [0,1]. Nakon toga ova dva broja se trasformišu u cijele brojeve koji pripadaju intervalu indeksa u nizu. Nakon toga dva hromozoma sa indeksima prethodno generisanih brojeva zamijene mjesta. Broj iteracija određen je predefinisanom varijablom i može da se mijenja. Postupak zamjene pozicija proizvoljnog broja hromozoma vrši se u svrhu simulacije stvarnih bioloških procesa. Nakon ovog dijela algoritma dobili smo podgeneraciju hromozoma koji su na slučajan način izmijenili pozicije u nizu. Što više slučajnosti uvedemo, to je simulacija sličnija prirodnim biološkim procesima. Odluku o tome da li će se dva hromozoma rekombinovati donosi generator pseudoslučajnih brojeva na način da generiše broj iz intervala [0,1]. Ukoliko je ovaj broj manji od predefinisane vrijednosti vjerovatnoće rekombinacije tada će se dva hromozoma ukrstiti. U suprotnom ne dolazi do rekombinacije. Sam proces rekombinacije[[10]](#footnote-11) gena može biti izveden na više načina. U ovom rješenju implementirana su četiri načina rekombinacije koji su opisani u nastavku.

* Ukrštanje u jednoj tački

Za binarnu reprezentaciju broja se odrede tačke između svakog od bita i numerišu se na sljedeći način: npr. 0 8 1 7 0 6 1 5 0 4 0 3 1 2 0 1 1 0 0. Tačka ukrštanja bira se na slučajan način. Nakon odabira tačke dijelovi binarne reprezentacije desno od tačke ukrštanja se razmijene između gena što je prikazano na slici 2.1. Binarna reprezentacija koordinata sa x ose prvog hromozoma se na ovaj način ukršta sa binarnom reprezentacijom x koordinate drugog hromozoma. Analogno vrijedi i za y koordinate.



Slika 2.1. Ukrštanje u jednoj tački

* Jednostavna zamjena x koordinata

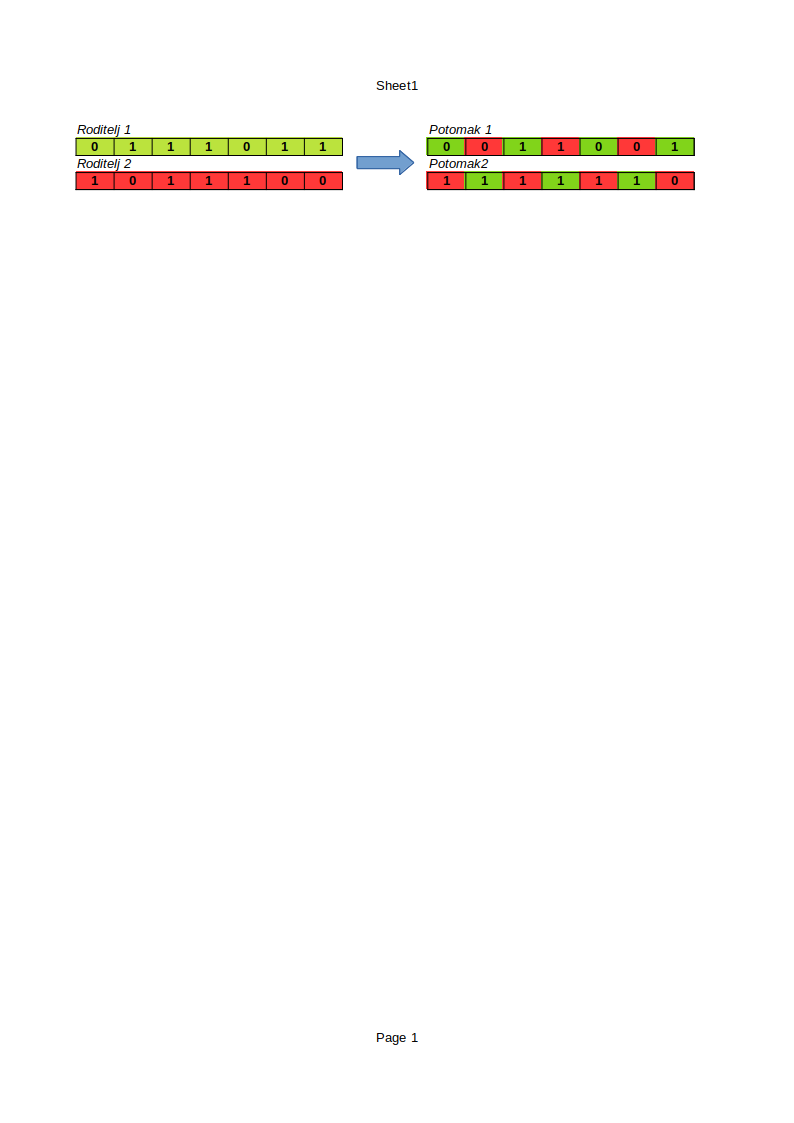
U ovom postupku rekombinacije hromozomi razmijene svoje x koordinate i na taj način kreiraju nove potomke.

* Jednostavna zamjena y koordinata

U ovom postupku rekombinacije hromozomi razmijene svoje y koordinate i na taj način kreiraju nove potomke.

* Uniformno ukrštanje

Kod ove vrste ukrštanja binarna reprezentacija koordinata se dobija tako što se naizmjenično uzima po jedan bit iz prve i jedan bit iz druge binarne reprezentacije kooridate u hromozomu i kopira u binarnu reprezentaciju koordinate prvog potomka. Za drugi potomak postupak je analogan samo što u ovom slučaju počinjemo proces od binarne reprezentacije drugog hromozoma. Ovaj postupak se primjenjuje zasebno prvo na x koordinate pa na y koordinate oba hromozoma. Grafički prikaz uniformnog ukrštanja prikazan je na Slici 2.2.



Slika 2.2 Uniformno ukrštanje

Treća faza algoritma jeste faza mutacije. Odluka o mutaciji hromozoma se vrši generisanjem[[11]](#footnote-12) pseudo-slučajnog broja u intervalu [0, 1]. Ukoliko je generisani broj manji od predefinisane vrijednosti mutacije onda dolazi do mutacije na datom hromozomu. U ovom rješenju je to implementirano na način da se prolazi kroz međugeneraciju hromozoma, koja je proizvod prethodne faze rekombinacij, te se za svaki hromozom generiše pseudoslučajan broj i provjerava da li će doći do mutacije. Sam proces mutacije[[12]](#footnote-13) se vrši na način da se za svaku binarnu reprezentaciju koordinata x i y generište pseudo-slučajan broj. Taj broj se transformiše u cjelobrojni indeks iz intervala indeksa binarne reprezentacije koordinate. Na tom indeksu se bit invertuje i na taj način je izvršena mutacija hromozoma. Ovaj postupak se primjenjuje i na binarnu reprezentaciju x koordinate kao i na binarnu reprezentaciju y koordinate u hromozomu. Nakon ove faze dobija se međugeneracija hromozoma u kojoj su neki od hromozoma mutirani.

Nako ove tri faze algoritma vrši se evaluacija međugeneracije koja je generisana u prethodnim fazama algoritma. Računaju se vrijdnosti funkcije prilagođenosti za svaki od hromozoma u međugeneraciji kao i ukupna ocjena F funkcije prilagođenosti cijele populacije. Sada dolazi prethodno opisan postupak odlučivanja o zamjeni trenutne populacije međugeneracijom nastalom u trenutnoj iteraciji. Ukoliko međugeneracija ima bolje karakteristike od trenutne generacije tada se ta međugeneracija sačuva i proglasi za trenutno najbolju generaciju. U narednu iteraciju algoritma ulazi trenutna međugeneracija gdje se opet ponavljaju sve faze algoritma. Broj iteracija algoritma je unaprijed definisana vrijednost.

U nastavku rada prikazani su rezultati eksperimaneata u vidu tabela, grafičkih prikaza kao i teksta koji pojašnjava koji parametri su odabrani i kakvi rezultati su postignuti istim.

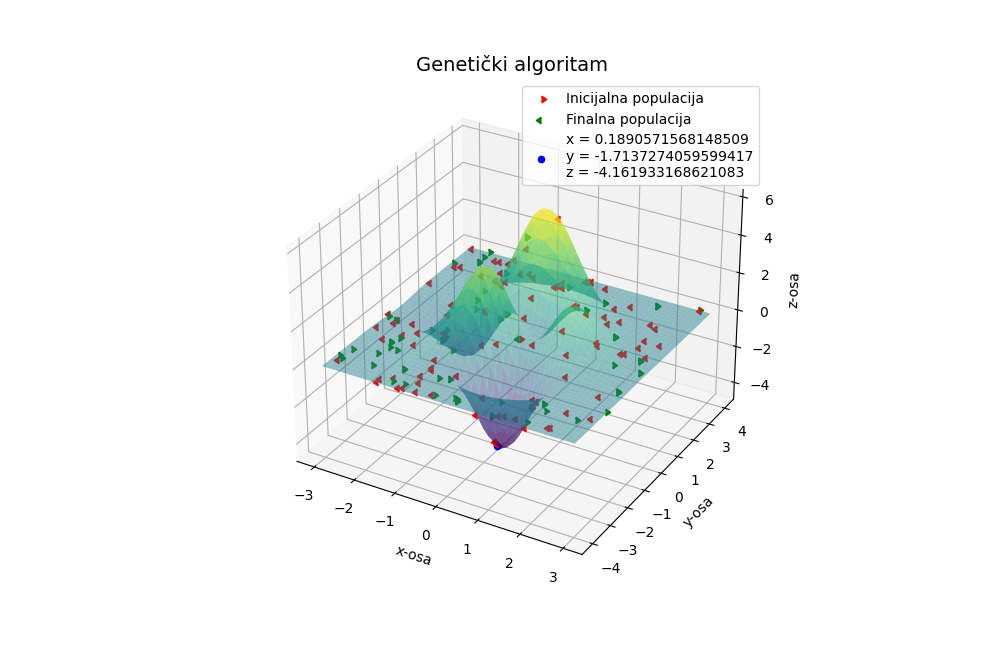
### Eksperiment 1.

U ovom eksperimentu korištena je ruletska selekcija sa rekombinacijom u jednoj tački. Rezultati simulacije sa različitim vrijednostima vjerovatnoće rekombinacije{0.15, 0.3}, vjerovatnoće mutacije{0.05,0.15}, veličine populacije{25,100}, brojem iteracija{25,100} sumirani su tabelarno u Tabeli 2.0.

Tabela 2.0

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Redni broj** | **Verovatnoća rekombinacije** | **Vjerovatnoća mutacije** | **Veličina populacije** | **Broj iteracija** | **Rješenje** |
| **1.** | 0.15 | 0.05 | 50 | 25 | -3.813 |
| **2.** | 0.15 | 0.05 | 50 | 100 | -4.018 |
| **3.** | 0.15 | 0.05 | 100 | 25 | -3.722 |
| **4.** | 0.15 | 0.05 | 100 | 100 | -4.012 |
| **5.** | 0.15 | 0.15 | 50 | 25 | -3.335 |
| **6.** | 0.15 | 0.15 | 50 | 100 | -4.102 |
| **7.** | 0.15 | 0.15 | 100 | 25 | -4.095 |
| **8.** | 0.15 | 0.15 | 100 | 100 | -4.133 |
| **9.** | 0.3 | 0.05 | 50 | 25 | -3.455 |
| **10.** | 0.3 | 0.05 | 50 | 100 | -3.854 |
| **11.** | 0.3 | 0.05 | 100 | 25 | -3.052 |
| **12.** | 0.3 | 0.05 | 100 | 100 | -3.975 |
| **13.** | 0.3 | 0.15 | 50 | 25 | -1.754 |
| **14.** | 0.3 | 0.15 | 50 | 100 | -2.549 |
| **15.** | 0.3 | 0.15 | 100 | 25 | -4.227 |
| **16.** | 0.3 | 0.15 | 100 | 100 | -4.255 |

Grafički prikaz jednog od rješenja u ovom eksperimentu prikazan je na Slici 2.3.



Slika 2.3 Grafički prikaz jednog rješenja

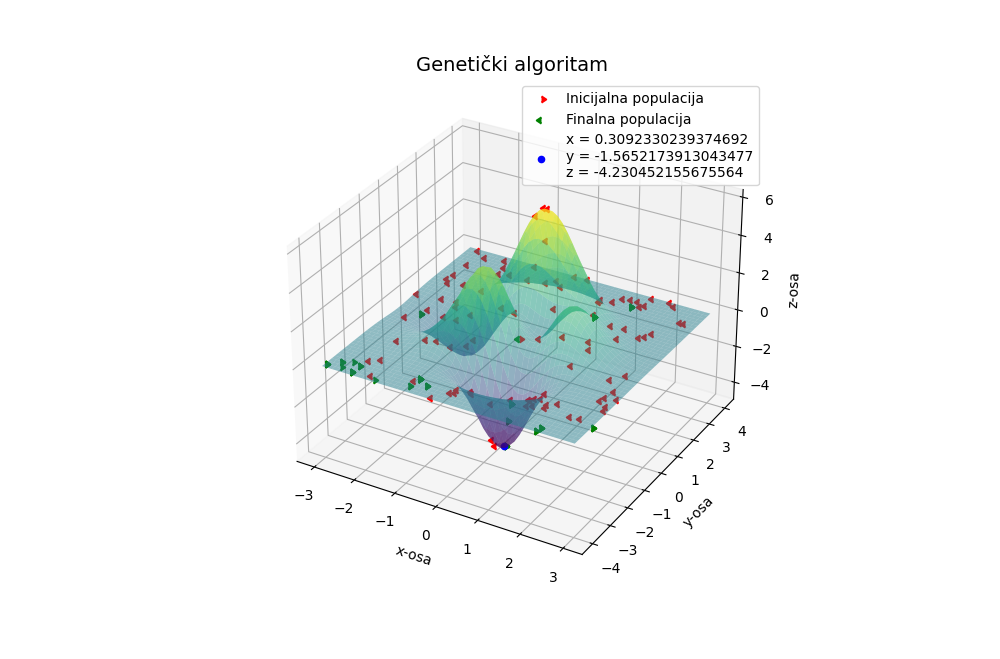
### Eksperiment 2.

U ovom eksperimentu korištena je ruletska selekcija sa rekombinacijom u kojoj se vrši prosta razmjena x koordinata. Rezultati simulacije sa različitim vrijednostima vjerovatnoće rekombinacije{0.15, 0.3}, vjerovatnoće mutacije{0.05,0.15}, veličine populacije{25,100}, brojem iteracija{25,100} sumirani su tabelarno u Tabeli 2.1.

Tabela 2.1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Redni broj** | **Verovatnoća rekombinacije** | **Vjerovatnoća mutacije** | **Veličina populacije** | **Broj iteracija** | **Rješenje** |
| **1.** | 0.15 | 0.05 | 50 | 25 | -4.248 |
| **2.** | 0.15 | 0.05 | 50 | 100 | -3.471 |
| **3.** | 0.15 | 0.05 | 100 | 25 | -4.252 |
| **4.** | 0.15 | 0.05 | 100 | 100 | -3.682 |
| **5.** | 0.15 | 0.15 | 50 | 25 | -3.042 |
| **6.** | 0.15 | 0.15 | 50 | 100 | -3.145 |
| **7.** | 0.15 | 0.15 | 100 | 25 | -3.725 |
| **8.** | 0.15 | 0.15 | 100 | 100 | -4.653 |
| **9.** | 0.3 | 0.05 | 50 | 25 | -2.608 |
| **10.** | 0.3 | 0.05 | 50 | 100 | -2.929 |
| **11.** | 0.3 | 0.05 | 100 | 25 | -3.832 |
| **12.** | 0.3 | 0.05 | 100 | 100 | -4.006 |
| **13.** | 0.3 | 0.15 | 50 | 25 | -4.062 |
| **14.** | 0.3 | 0.15 | 50 | 100 | -2.607 |
| **15.** | 0.3 | 0.15 | 100 | 25 | -4.116 |
| **16.** | 0.3 | 0.15 | 100 | 100 | -4.240 |

Grafički prikaz jednog od rješenja u ovom eksperimentu prikazan je na Slici 2.4.



Slika 2.4 Grafički prikaz jednog rješenja

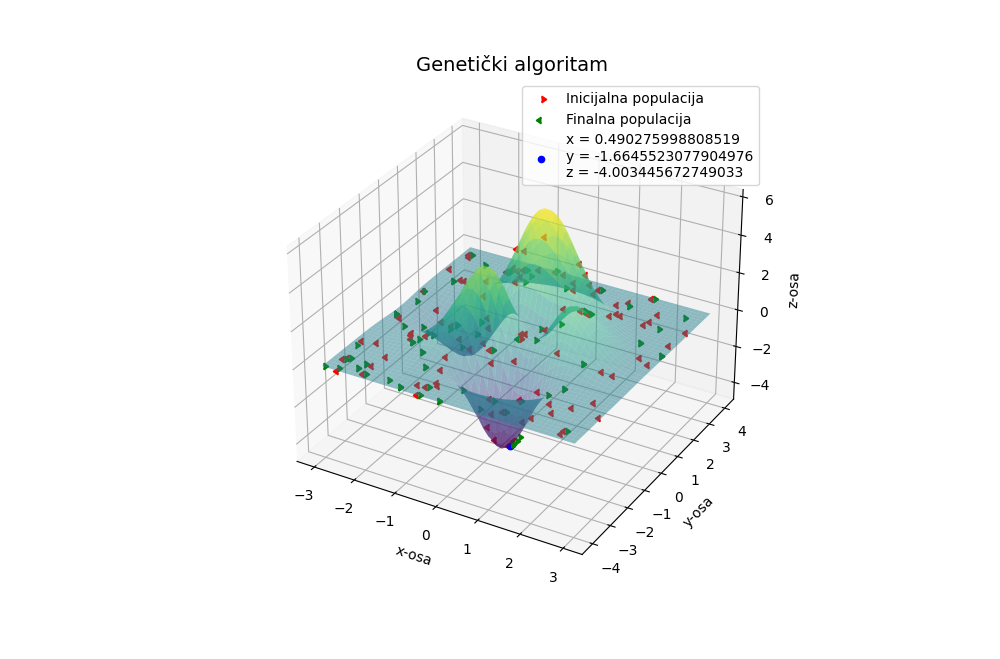
### Eksperiment 3.

U ovom eksperimentu korištena je ruletska selekcija sa rekombinacijom u kojoj se vrši prosta razmjena y koordinata. Rezultati simulacije sa različitim vrijednostima vjerovatnoće rekombinacije{0.15, 0.3}, vjerovatnoće mutacije{0.05,0.15}, veličine populacije{25,100}, brojem iteracija{25,100} sumirani su tabelarno u Tabeli 2.2.

Tabela 2.2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Redni broj** | **Verovatnoća rekombinacije** | **Vjerovatnoća mutacije** | **Veličina populacije** | **Broj iteracija** | **Rješenje** |
| **1.** | 0.15 | 0.05 | 50 | 25 | -1.297 |
| **2.** | 0.15 | 0.05 | 50 | 100 | -2.624 |
| **3.** | 0.15 | 0.05 | 100 | 25 | -3.737 |
| **4.** | 0.15 | 0.05 | 100 | 100 | -3.814 |
| **5.** | 0.15 | 0.15 | 50 | 25 | -2.702 |
| **6.** | 0.15 | 0.15 | 50 | 100 | -3.666 |
| **7.** | 0.15 | 0.15 | 100 | 25 | -4.156 |
| **8.** | 0.15 | 0.15 | 100 | 100 | -4.205 |
| **9.** | 0.3 | 0.05 | 50 | 25 | -3.843 |
| **10.** | 0.3 | 0.05 | 50 | 100 | -2.884 |
| **11.** | 0.3 | 0.05 | 100 | 25 | -3.914 |
| **12.** | 0.3 | 0.05 | 100 | 100 | -4.056 |
| **13.** | 0.3 | 0.15 | 50 | 25 | -3.999 |
| **14.** | 0.3 | 0.15 | 50 | 100 | -3.689 |
| **15.** | 0.3 | 0.15 | 100 | 25 | -4.165 |
| **16.** | 0.3 | 0.15 | 100 | 100 | -3.812 |

Grafički prikaz jednog od rješenja u ovom eksperimentu prikazan je na Slici 2.5.



Slika 2.5 Grafički prikaz jednog rješenja

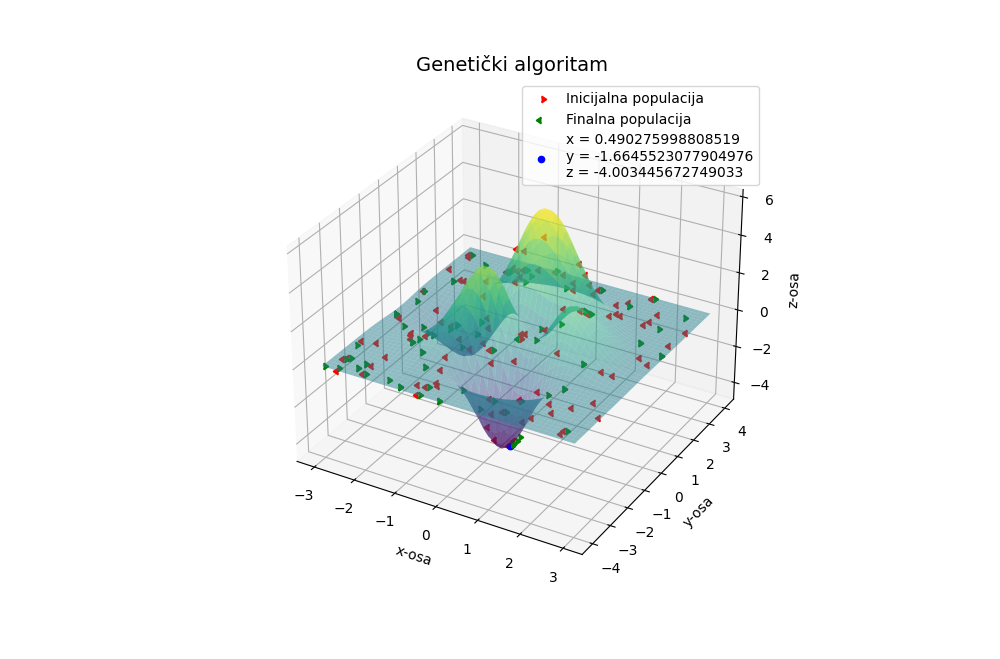
### Eksperiment 4.

U ovom eksperimentu korištena je ruletska selekcija sa unifirmnom rekombinacijom. Rezultati simulacije sa različitim vrijednostima vjerovatnoće rekombinacije{0.15, 0.3}, vjerovatnoće mutacije{0.05,0.15}, veličine populacije{25,100}, brojem iteracija{25,100} sumirani su tabelarno u Tabeli 2.3.

Tabela 2.3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Redni broj** | **Verovatnoća rekombinacije** | **Vjerovatnoća mutacije** | **Veličina populacije** | **Broj iteracija** | **Rješenje** |
| **1.** | 0.15 | 0.05 | 50 | 25 | -3.957 |
| **2.** | 0.15 | 0.05 | 50 | 100 | -3.832 |
| **3.** | 0.15 | 0.05 | 100 | 25 | -4.031 |
| **4.** | 0.15 | 0.05 | 100 | 100 | -3.935 |
| **5.** | 0.15 | 0.15 | 50 | 25 | -3.809 |
| **6.** | 0.15 | 0.15 | 50 | 100 | -2.330 |
| **7.** | 0.15 | 0.15 | 100 | 25 | 3.966 |
| **8.** | 0.15 | 0.15 | 100 | 100 | -4.266 |
| **9.** | 0.3 | 0.05 | 50 | 25 | -2.522 |
| **10.** | 0.3 | 0.05 | 50 | 100 | -2.622 |
| **11.** | 0.3 | 0.05 | 100 | 25 | -4.188 |
| **12.** | 0.3 | 0.05 | 100 | 100 | -3.395 |
| **13.** | 0.3 | 0.15 | 50 | 25 | -3.288 |
| **14.** | 0.3 | 0.15 | 50 | 100 | -3.399 |
| **15.** | 0.3 | 0.15 | 100 | 25 | -4.087 |
| **16.** | 0.3 | 0.15 | 100 | 100 | -4.091 |

Grafički prikaz jednog od rješenja u ovom eksperimentu prikazan je na Slici 2.6.



Slika 2.6 Grafički prikaz jednog rješenja

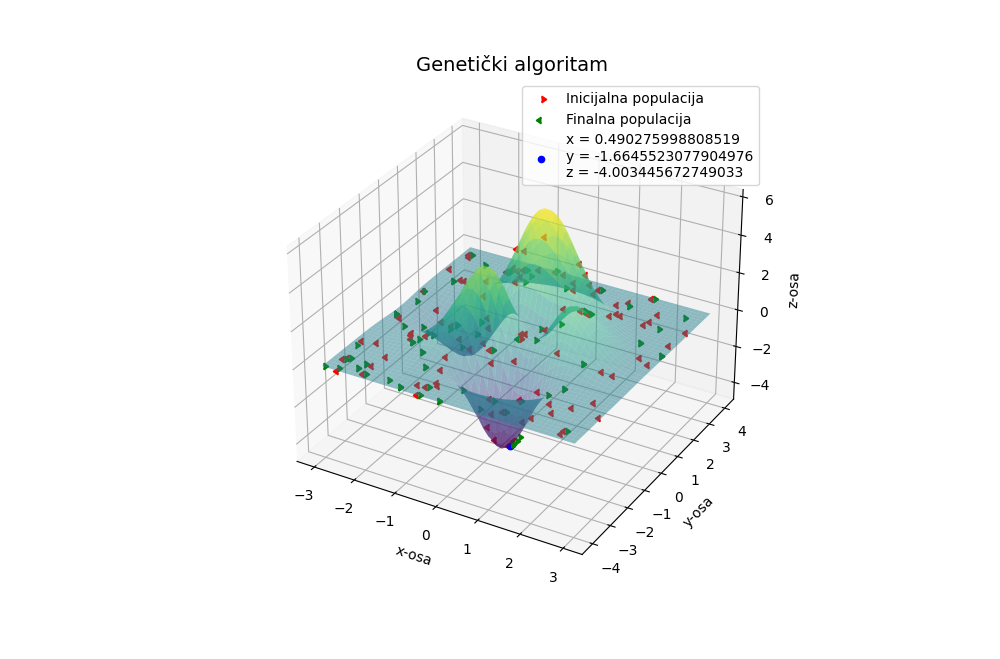
### Eksperiment 5.

U ovom eksperimentu korištena je turnirska selekcija sa rekombinacijom u jednoj tački. Rezultati simulacije sa različitim vrijednostima vjerovatnoće rekombinacije{0.15, 0.3}, vjerovatnoće mutacije{0.05,0.15}, veličine populacije{25,100}, brojem iteracija{25,100} sumirani su tabelarno u Tabeli 2.4.

Tabela 2.4

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Redni broj** | **Verovatnoća rekombinacije** | **Vjerovatnoća mutacije** | **Veličina populacije** | **Broj iteracija** | **Rješenje** |
| **1.** | 0.15 | 0.05 | 50 | 25 | -3.215 |
| **2.** | 0.15 | 0.05 | 50 | 100 | -3.887 |
| **3.** | 0.15 | 0.05 | 100 | 25 | -3.672 |
| **4.** | 0.15 | 0.05 | 100 | 100 | -4.035 |
| **5.** | 0.15 | 0.15 | 50 | 25 | -2.435 |
| **6.** | 0.15 | 0.15 | 50 | 100 | -2.405 |
| **7.** | 0.15 | 0.15 | 100 | 25 | -3.462 |
| **8.** | 0.15 | 0.15 | 100 | 100 | -3.818 |
| **9.** | 0.3 | 0.05 | 50 | 25 | -3.4247 |
| **10.** | 0.3 | 0.05 | 50 | 100 | -2.317 |
| **11.** | 0.3 | 0.05 | 100 | 25 | -4.144 |
| **12.** | 0.3 | 0.05 | 100 | 100 | -3.778 |
| **13.** | 0.3 | 0.15 | 50 | 25 | -4.012 |
| **14.** | 0.3 | 0.15 | 50 | 100 | -2.213 |
| **15.** | 0.3 | 0.15 | 100 | 25 | -2.435 |
| **16.** | 0.3 | 0.15 | 100 | 100 | -4.158 |

Grafički prikaz jednog od rješenja u ovom eksperimentu prikazan je na Slici 2.7.



Slika 2.7 Grafički prikaz jednog rješenja

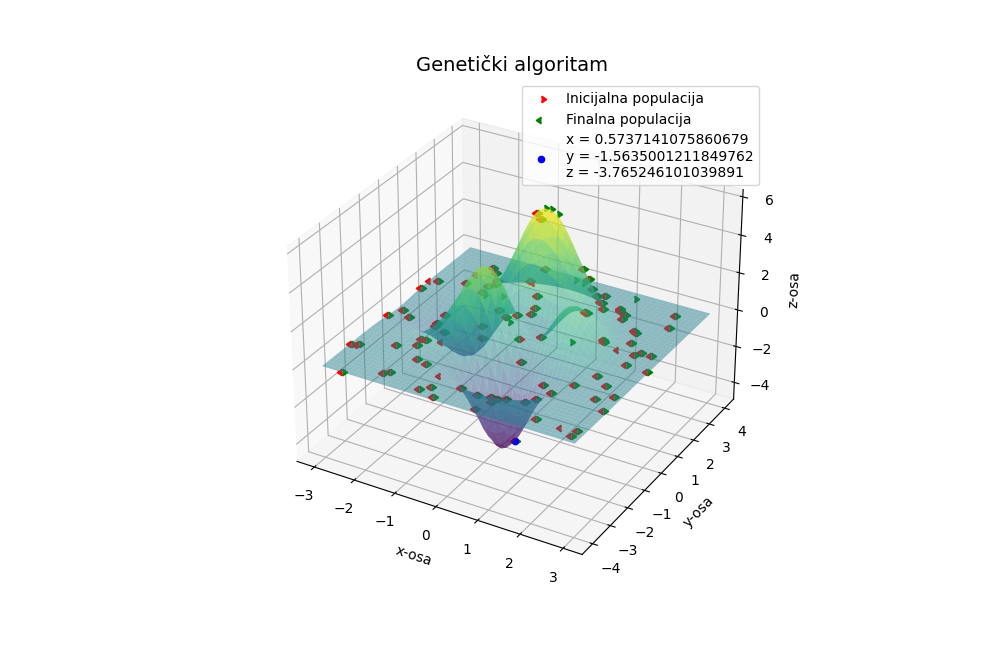
### Eksperiment 6.

U ovom eksperimentu korištena je turnirska selekcija sa rekombinacijom u kojoj se vrši prosta razmjena x koordinata. Rezultati simulacije sa različitim vrijednostima vjerovatnoće rekombinacije{0.15, 0.3}, vjerovatnoće mutacije{0.05,0.15}, veličine populacije{25,100}, brojem iteracija{25,100} sumirani su tabelarno u Tabeli 2.5.

Tabela 2.5

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Redni broj** | **Verovatnoća rekombinacije** | **Vjerovatnoća mutacije** | **Veličina populacije** | **Broj iteracija** | **Rješenje** |
| **1.** | 0.15 | 0.05 | 50 | 25 | -3.144 |
| **2.** | 0.15 | 0.05 | 50 | 100 | -1.284 |
| **3.** | 0.15 | 0.05 | 100 | 25 | -4.217 |
| **4.** | 0.15 | 0.05 | 100 | 100 | -2.874 |
| **5.** | 0.15 | 0.15 | 50 | 25 | -3.878 |
| **6.** | 0.15 | 0.15 | 50 | 100 | -4.125 |
| **7.** | 0.15 | 0.15 | 100 | 25 | -3.673 |
| **8.** | 0.15 | 0.15 | 100 | 100 | -3.079 |
| **9.** | 0.3 | 0.05 | 50 | 25 | -3.148 |
| **10.** | 0.3 | 0.05 | 50 | 100 | -3.680 |
| **11.** | 0.3 | 0.05 | 100 | 25 | -2.750 |
| **12.** | 0.3 | 0.05 | 100 | 100 | -4.168 |
| **13.** | 0.3 | 0.15 | 50 | 25 | -3.295 |
| **14.** | 0.3 | 0.15 | 50 | 100 | -4.241 |
| **15.** | 0.3 | 0.15 | 100 | 25 | -3.710 |
| **16.** | 0.3 | 0.15 | 100 | 100 | -4.260 |

Grafički prikaz jednog od rješenja u ovom eksperimentu prikazan je na Slici 2.8.



Slika 2.8 Grafički prikaz jednog rješenja

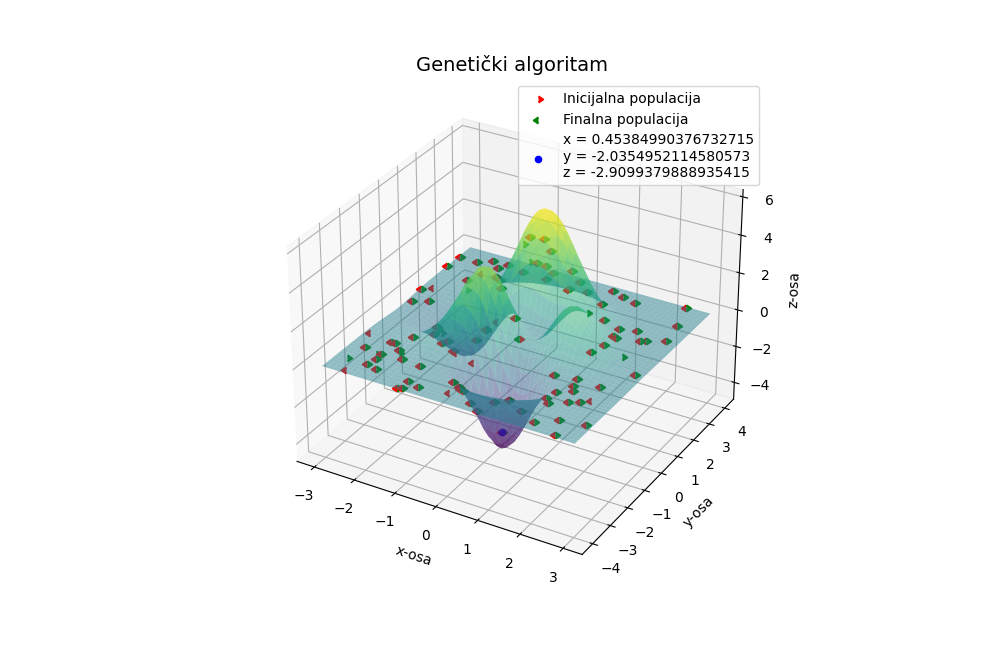
### Eksperiment 7.

U ovom eksperimentu korištena je turnirska selekcija sa rekombinacijom u kojoj se vrši prosta razmjena y koordinata. Rezultati simulacije sa različitim vrijednostima vjerovatnoće rekombinacije{0.15, 0.3}, vjerovatnoće mutacije{0.05,0.15}, veličine populacije{25,100}, brojem iteracija{25,100} sumirani su tabelarno u Tabeli 2.6.

Tabela 2.6

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Redni broj** | **Verovatnoća rekombinacije** | **Vjerovatnoća mutacije** | **Veličina populacije** | **Broj iteracija** | **Rješenje** |
| **1.** | 0.15 | 0.05 | 50 | 25 | -1.528 |
| **2.** | 0.15 | 0.05 | 50 | 100 | -3.391 |
| **3.** | 0.15 | 0.05 | 100 | 25 | -3.797 |
| **4.** | 0.15 | 0.05 | 100 | 100 | -4.014 |
| **5.** | 0.15 | 0.15 | 50 | 25 | -1.180 |
| **6.** | 0.15 | 0.15 | 50 | 100 | -1.381 |
| **7.** | 0.15 | 0.15 | 100 | 25 | -4.207 |
| **8.** | 0.15 | 0.15 | 100 | 100 | -1.072 |
| **9.** | 0.3 | 0.05 | 50 | 25 | -3.868 |
| **10.** | 0.3 | 0.05 | 50 | 100 | -2.624 |
| **11.** | 0.3 | 0.05 | 100 | 25 | -3.512 |
| **12.** | 0.3 | 0.05 | 100 | 100 | -3.344 |
| **13.** | 0.3 | 0.15 | 50 | 25 | -3.901 |
| **14.** | 0.3 | 0.15 | 50 | 100 | -1.041 |
| **15.** | 0.3 | 0.15 | 100 | 25 | -4.050 |
| **16.** | 0.3 | 0.15 | 100 | 100 | -2.914 |

Grafički prikaz jednog od rješenja u ovom eksperimentu prikazan je na Slici 2.9.



Slika 2.9 Grafički prikaz jednog rješenja

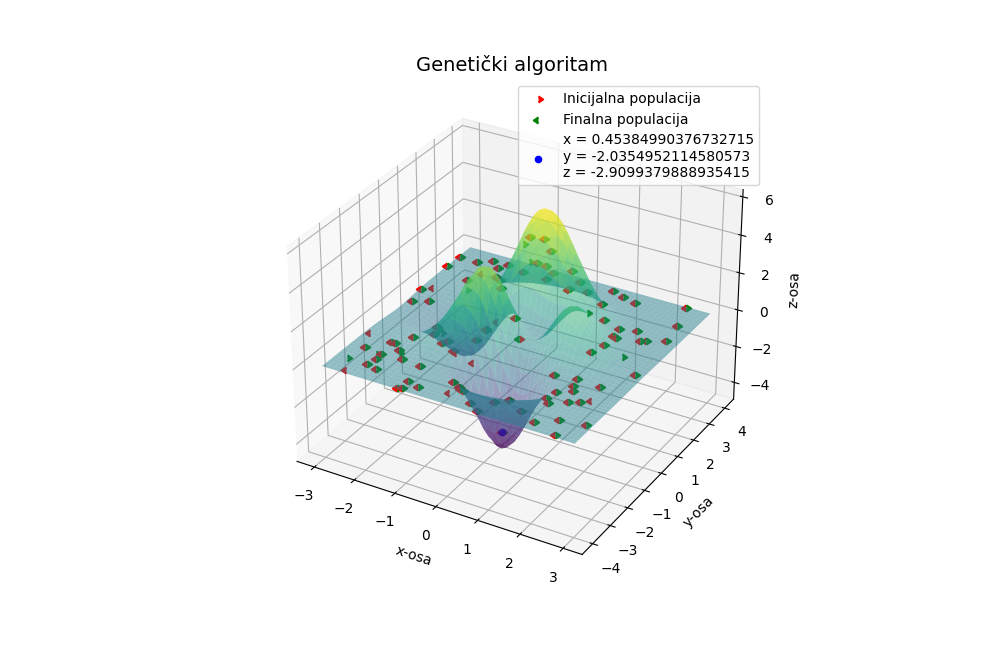
### Eksperiment 8.

U ovom eksperimentu korištena je turnirska selekcija sa uniformnom rekombinacijom. Rezultati simulacije sa različitim vrijednostima vjerovatnoće rekombinacije{0.15, 0.3}, vjerovatnoće mutacije{0.05,0.15}, veličine populacije{25,100}, brojem iteracija{25,100} sumirani su tabelarno u Tabeli 2.7.

Tabela 2.7

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Redni broj** | **Verovatnoća rekombinacije** | **Vjerovatnoća mutacije** | **Veličina populacije** | **Broj iteracija** | **Rješenje** |
| **1.** | 0.15 | 0.05 | 50 | 25 | -3.632 |
| **2.** | 0.15 | 0.05 | 50 | 100 | -0.095 |
| **3.** | 0.15 | 0.05 | 100 | 25 | -3.867 |
| **4.** | 0.15 | 0.05 | 100 | 100 | -3.808 |
| **5.** | 0.15 | 0.15 | 50 | 25 | -2.731 |
| **6.** | 0.15 | 0.15 | 50 | 100 | -2.662 |
| **7.** | 0.15 | 0.15 | 100 | 25 | -4.072 |
| **8.** | 0.15 | 0.15 | 100 | 100 | -4.104 |
| **9.** | 0.3 | 0.05 | 50 | 25 | -2.447 |
| **10.** | 0.3 | 0.05 | 50 | 100 | -4.232 |
| **11.** | 0.3 | 0.05 | 100 | 25 | -2.587 |
| **12.** | 0.3 | 0.05 | 100 | 100 | -1.554 |
| **13.** | 0.3 | 0.15 | 50 | 25 | -3.906 |
| **14.** | 0.3 | 0.15 | 50 | 100 | -3.761 |
| **15.** | 0.3 | 0.15 | 100 | 25 | -3.304 |
| **16.** | 0.3 | 0.15 | 100 | 100 | -2.732 |

Grafički prikaz jednog od rješenja u ovom eksperimentu prikazan je na Slici 2.10.



Slika 2.10 Grafički prikaz jednog rješenja

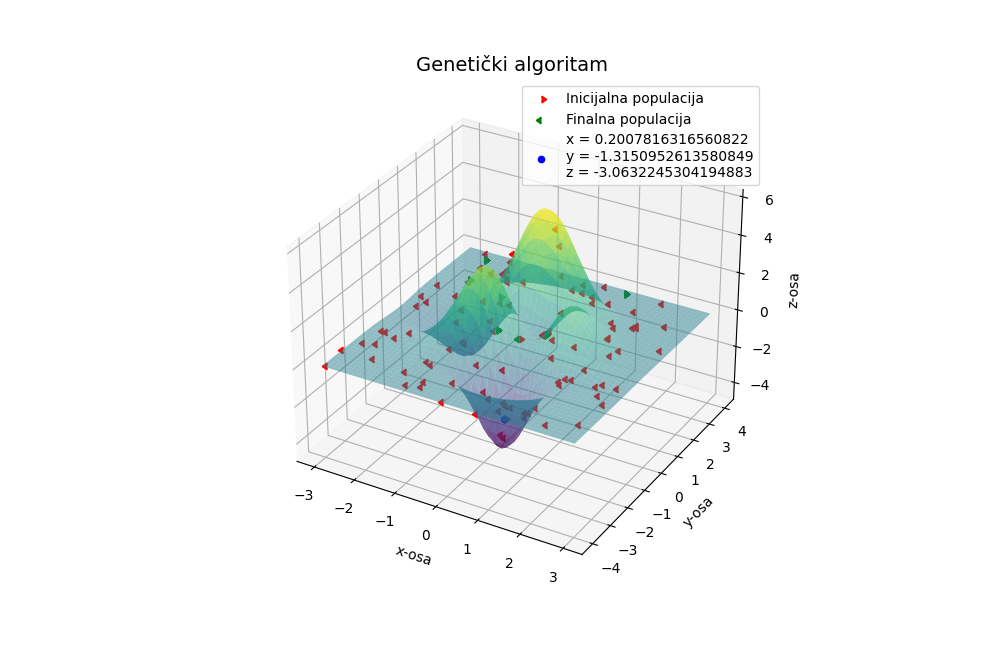
### Eksperiment 9.

U ovom eksperimentu korištena je elitistička selekcija sa rekombinacijom u jednoj tački. Rezultati simulacije sa različitim vrijednostima vjerovatnoće rekombinacije{0.15, 0.3}, vjerovatnoće mutacije{0.05,0.15}, veličine populacije{25,100}, brojem iteracija{25,100} sumirani su tabelarno u Tabeli 2.8.

Tabela 2.8

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Redni broj** | **Verovatnoća rekombinacije** | **Vjerovatnoća mutacije** | **Veličina populacije** | **Broj iteracija** | **Rješenje** |
| **1.** | 0.15 | 0.05 | 50 | 25 | -0.589 |
| **2.** | 0.15 | 0.05 | 50 | 100 | -1.398 |
| **3.** | 0.15 | 0.05 | 100 | 25 | -2.998 |
| **4.** | 0.15 | 0.05 | 100 | 100 | -0.818 |
| **5.** | 0.15 | 0.15 | 50 | 25 | -0.820 |
| **6.** | 0.15 | 0.15 | 50 | 100 | -1.894 |
| **7.** | 0.15 | 0.15 | 100 | 25 | -1.973 |
| **8.** | 0.15 | 0.15 | 100 | 100 | 0.065 |
| **9.** | 0.3 | 0.05 | 50 | 25 | -0.139 |
| **10.** | 0.3 | 0.05 | 50 | 100 | -0.210 |
| **11.** | 0.3 | 0.05 | 100 | 25 | -1.809 |
| **12.** | 0.3 | 0.05 | 100 | 100 | -0.686 |
| **13.** | 0.3 | 0.15 | 50 | 25 | -0.841 |
| **14.** | 0.3 | 0.15 | 50 | 100 | -1318 |
| **15.** | 0.3 | 0.15 | 100 | 25 | -1.736 |
| **16.** | 0.3 | 0.15 | 100 | 100 | -4.061 |

Grafički prikaz jednog od rješenja u ovom eksperimentu prikazan je na Slici 2.11.



Slika 2.11 Grafički prikaz jednog rješenja

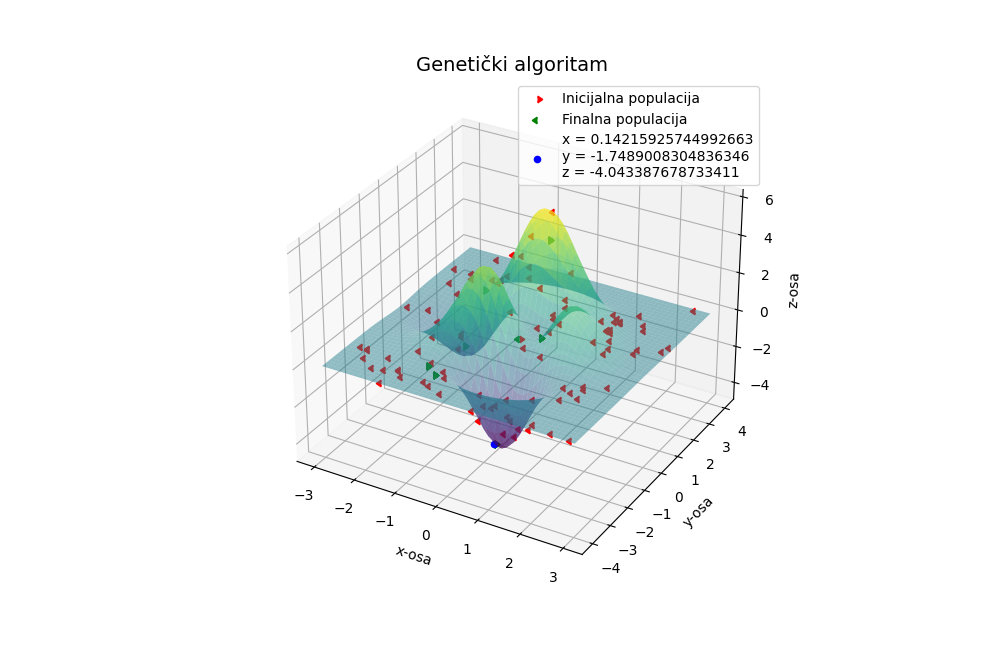
### Eksperiment 10.

U ovom eksperimentu korištena je elitistička selekcija sa uniformnom rekombinacijom. Rezultati simulacije sa različitim vrijednostima vjerovatnoće rekombinacije{0.15, 0.3}, vjerovatnoće mutacije{0.05,0.15}, veličine populacije{25,100}, brojem iteracija{25,100} sumirani su tabelarno u Tabeli 2.9.

Tabela 2.9

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Redni broj** | **Verovatnoća rekombinacije** | **Vjerovatnoća mutacije** | **Veličina populacije** | **Broj iteracija** | **Rješenje** |
| **1.** | 0.15 | 0.05 | 50 | 25 | -3.294 |
| **2.** | 0.15 | 0.05 | 50 | 100 | 1.412 |
| **3.** | 0.15 | 0.05 | 100 | 25 | -3.012 |
| **4.** | 0.15 | 0.05 | 100 | 100 | -0.799 |
| **5.** | 0.15 | 0.15 | 50 | 25 | -0.911 |
| **6.** | 0.15 | 0.15 | 50 | 100 | -0.366 |
| **7.** | 0.15 | 0.15 | 100 | 25 | -4.072 |
| **8.** | 0.15 | 0.15 | 100 | 100 | -0.949 |
| **9.** | 0.3 | 0.05 | 50 | 25 | -3.688 |
| **10.** | 0.3 | 0.05 | 50 | 100 | -3.205 |
| **11.** | 0.3 | 0.05 | 100 | 25 | -0.827 |
| **12.** | 0.3 | 0.05 | 100 | 100 | -3.126 |
| **13.** | 0.3 | 0.15 | 50 | 25 | -0.559 |
| **14.** | 0.3 | 0.15 | 50 | 100 | -1.984 |
| **15.** | 0.3 | 0.15 | 100 | 25 | -2.037 |
| **16.** | 0.3 | 0.15 | 100 | 100 | -3.771 |

Grafički prikaz jednog od rješenja u ovom eksperimentu prikazan je na Slici 2.12.



Slika 2.12 Grafički prikaz jednog rješenja

Prilikom eksperimenta u kojem se fiksiraju parametri vjerovatnoće rekombinacije na vrijednost 0.3, vjerovatnoće mutacije na vrijednost 0.15, veličine populacije na vrijednost 300, broj iteracija na vrijednost 25, vrste rekombinacije u jednoj tački, mogu se izvući sljedeći zaključci vezani za različite metode selekcije. Najbolje rezultate od tri implementirane metode selekcije(ruletska, turnirska i elitistička) daje turnirska selekcija. Aritmetička sredina dobijenih rješenja kroz pet pokretanja algoritma iznosi -4.218. Ovaj rezultat je dobijen za vrijednost selekcionog pritiska K=2. Prilikom povećavanja tog parametra na 10 dobijamo lošije prosječno rješenje u pet pokretanja algoritma koje iznosi -3.965. Elitistička selekcija u ovoj simulaciji je dala najlošije prosječno rješenje koje iznosi -3.403. Ruletska selekcija daje prosječno rješenje -3.944.

Prilikom eksperimenta u kojem se fiksiraju parametri vjerovatnoće rekombinacije na vrijednost 0.3, vjerovatnoće mutacije na vrijednost 0.15, veličine populacije na vrijednost 300, broj iteracija na vrijednost 25, turnirske vrste selekcije (K=2), mogu se izvući sljedeći zaključci vezani za različite vrste rekombinacije hromozoma. Primijećeno je da najbolje prosječno rješenje pri datim parametrima daje metoda rekombinacije u jednoj tački. Ostale tri metode u ovom eksperimentu proizvele su približno jednaka prosječna rješenja.

Prilikom eksperimenta u kojem se fiksiraju parametri vjerovatnoće mutacije na vrijednost 0.05, veličine populacije na vrijednost 300, broj iteracija na vrijednost 25, ruletske vrste selekcije, metode rekombinacije u jednoj tački, mogu se izvući sljedeći zaključci vezani za zavisnost kvaliteta rješenja od vrijednosti vjerovatnoće rekombinacije. Uticaj vrijednosti vjerovatnoće rekombinacije, pri istim početnim populacijama, na krajnje rješenje sumirano je u Tabeli 2.10. Uočava se da pri datim parametrima najoptimalnije rješenje se dobija kada je vrijednost vjerovatnoće rekombinacije u intervalu [0.15, 0.3].

Tabela 2.10

|  |  |
| --- | --- |
| **Verovatnoća rekombinacije** | **Rješenje** |
| 0.01 | -3.429 |
| 0.15 | -3.548 |
| 0.3 | -3.917 |
| 0.5 | -3.430 |
| 0.8 | -3.413 |
| 0.99 | -3.102 |

Prilikom eksperimenta u kojem se fiksiraju parametri vjerovatnoće rekombinacije na vrijednost 0.3, veličine populacije na vrijednost 300, broj iteracija na vrijednost 25, ruletske vrste selekcije, metode rekombinacije u jednoj tački, mogu se izvući sljedeći zaključci vezani za zavisnost kvaliteta rješenja od vrijednosti vjerovatnoće mutacije. Uočava se da pri datim parametrima najoptimalnije rješenje se dobija kada je vrijednost vjerovatnoće mutacije u intervalu [0.05, 0.20].

Kroz seriju pokretanja simulacije utvrđeno je da sa porastom populacije raste i tačnost rješenja. Razlog ovome je što pri većim populacijama je i veća vjerovatnoća da će se određen broj slučajno generisanih hromozoma naći u okolini stvarnog rješenja. Kroz iteracije će se ta rješenja birati za roditelje dati optimalno rješenje problema.

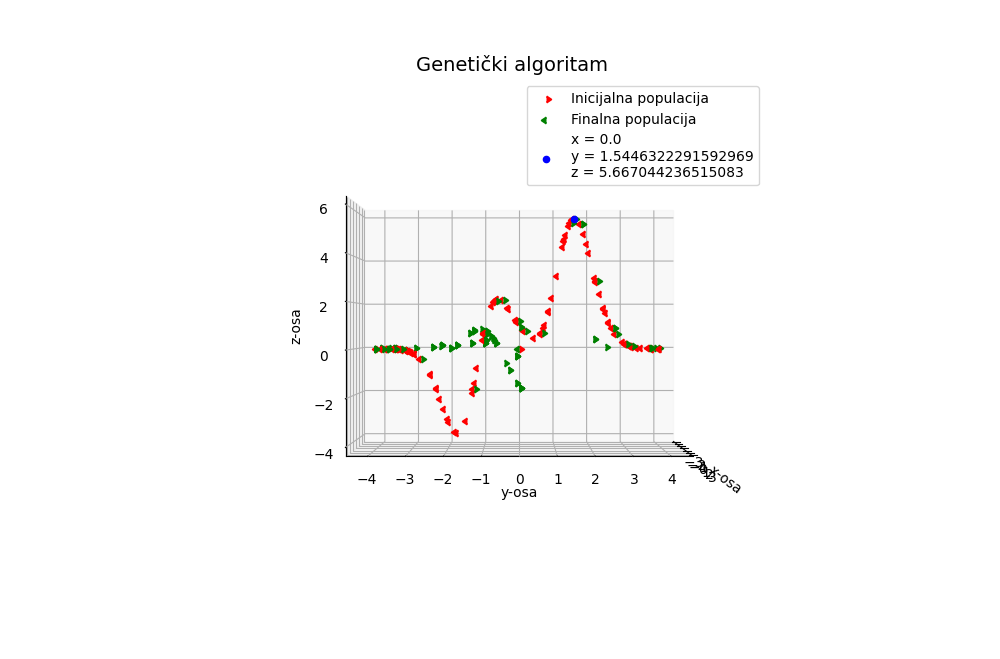
Kroz seriju pokretanja, pri fiksnoj veličini populacije od 500 jedinki , utvrđeno je da algoritam daje dobra rješenja ukoliko je broj iteracija iz intervala (25,300). Ukoliko se nastavi povećavati broj iteracija, kvalitet rješenja opada.

Pored problema pronalaženja minimuma krive date relacijom 1.0. postavljen je i problem pronalaženja maksimuma funkcije date relacijom 1.1. Simulacija ove funkcije je u ovom rješenju implementirana da se svakoj od x koordinata proslijedi vrijednost 0.0, dok se vrijednosti y kooridnata generišu na pseudo-slučajan način. U eksperimentu je korištena elitistička selekcija sa rekombinacijom u jednoj tački. Rezultati simulacije sa različitim vrijednostima vjerovatnoće rekombinacije{0.15, 0.3}, vjerovatnoće mutacije{0.05,0.15}, veličine populacije{25,100}, brojem iteracija{25,100} sumirani su tabelarno u Tabeli 2.11.

Tabela 2.11

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Redni broj** | **Verovatnoća rekombinacije** | **Vjerovatnoća mutacije** | **Veličina populacije** | **Broj iteracija** | **Rješenje** |
| **1.** | 0.15 | 0.05 | 50 | 25 | 5.458 |
| **2.** | 0.15 | 0.05 | 50 | 100 | 5.347 |
| **3.** | 0.15 | 0.05 | 100 | 25 | 5.623 |
| **4.** | 0.15 | 0.15 | 50 | 25 | 4.772 |
| **5.** | 0.15 | 0.15 | 50 | 100 | 5.622 |
| **6.** | 0.15 | 0.15 | 100 | 25 | 5.645 |
| **7.** | 0.3 | 0.05 | 50 | 25 | 5.579 |
| **8.** | 0.3 | 0.05 | 50 | 100 | 4.948 |
| **9.** | 0.3 | 0.05 | 100 | 25 | 5.592 |
| **10.** | 0.3 | 0.15 | 50 | 25 | 5.523 |
| **11.** | 0.3 | 0.15 | 50 | 100 | 5.681 |
| **12.** | 0.3 | 0.15 | 100 | 25 | 5.675 |

Grafički prikaz jednog od rješenja u ovom eksperimentu prikazan je na Slici 2.13.



Slika2.13.Grafički prikaz jednog rješenja

# Zaključak

Prilikom izrade ovog projekta primijećeno je da postoji jaka veza između izbora inicijalne populacije i kvaliteta finalnog rješenja. Kako se inicijalna populacija bira na slučajan način, postoji mogućnost da se generiše inicijalna populacija čije su sve jedinke grupisane daleko od stvarnog rješenja problema. U tom slučaju je potreban veliki broj iteracija da bi se kroz faze algoritma približili stvarnom rješenju (s tim da povećanje broja iteracija ne garantuje da će bolje rješenje stvarno i biti pronađeno). Ovaj problem možemo riješiti na način da ravan koju generišu x i y osa podijelimo u četiri kvadranta pa da onda u svkom od njih na slučajan način postavimo četvrtinu broja jedinki u populaciji. Na ovaj način bi se obezbijedila ravnomjernija raspodjela inicijalnih jedinki.

Praktična korist ovog algoritma ogleda se u tome da je ovo programsko rješenje moguće primijeniti na bilo koju funkciju sa jednom ili dvije promjenljive. Takođe je moguće tražiti kako minimum tako i maksimum date funkcije što može biti korisno u slučajevima kada je analitički zahtijevno računati ekstremume funkcije.

1. Binarno kodovanje i dekodovanje decimalnih brojeva implementirano je u fajlu *BinaryCoding.py*  u funkcijama codeBinary() i decodeBinary(). [↑](#footnote-ref-2)
2. Koncept hromozoma implementiran je klasom *Chromosome* u fajlu *Chromosome.py* [↑](#footnote-ref-3)
3. Preslikavanje se vrši pomoću funkcije *transformToInterval()* u fajlu *BinaryCoding.py* [↑](#footnote-ref-4)
4. Kreiranje inicijalne populacije vrši se u funkciji *initializePopulation()* u fajlu *AlgorithmPhaseFunctions.py* [↑](#footnote-ref-5)
5. Implementirana u funkciji *fitnessFunction()* u fajlu *BinaryCoding.py* [↑](#footnote-ref-6)
6. Računanje ocjene populacije vrši se pomoću funkcije calculatePopulationFitnessScore() u fajlu *BinaryCoding.py* [↑](#footnote-ref-7)
7. Sve vrste selekcije implementirane su u funkciji *selection()* u fajlu *AlgorithmPhaseFunctions.py* [↑](#footnote-ref-8)
8. Izvor: Prezentacija sa predavanja [↑](#footnote-ref-9)
9. Faza rekombinacije je implementirana u funkciji *crossover()* u fajlu *AlgorithmPhaseFunctions.py* [↑](#footnote-ref-10)
10. Rekombinacija je implementirana kao metoda *recombine()* klase Chromosome u fajlu *Chromosome.py* [↑](#footnote-ref-11)
11. Faza u kojoj se odlučuje o mutaciji se odvija u funkciji *mutation()*  u fajlu *AlgorithmPhaseFunctions.py* [↑](#footnote-ref-12)
12. Sama mutacija odvija se u metodi *mutate()* kalse Chromosome u fajlu *Chromosome.py* [↑](#footnote-ref-13)