Univerzitet u Beogradu
Elektrotehnički fakultet

SEMINARSKI RAD

REŠAVANJE LAVIRINTA POMOĆU GENETSKOG ALGORITMA

Uroš Pantelić
2021/0073

Beograd, 2023

Sadržaj

1 Uvod

Upotreba genetskih algoritama (GA) za rešavanje problema nije nova stvar. Rad J. H. Hollanda iz 1970-ih pokazao se kao značajan doprinos naučnim i inženjerskim aplikacijama. Od tada, oblast istraživanja u ovoj disciplini je eksponencijalno rasla, iako su doprinosi uglavnom stizali iz akademskih institucija širom sveta. Tek nedavno smo uspeli da prikupimo materijal koji proizlazi iz industrije.

Međutim, koncept ove metode često nije jasno razumljen. Očigledna prepreka koja može odbiti inženjere od korišćenja GA jeste teškoća ubrzanja računskog procesa, kao i intrinzična priroda slučajnosti koja dovodi do izazova u obezbeđivanju performansi. Dodatno, GA se ne smatra matematički vođenim algoritmom. Optimum koji se postiže razvija se iz generacije u generaciju bez strogog matematičkog formulisanja, za razliku od tradicionalnih optimizacionih postupaka zasnovanih na gradijentu. GA je zapravo stohastički, diskretan događaj i nelinearan proces. Dobijeni optimum predstavlja krajnji proizvod koji sadrži najbolje elemente prethodnih generacija, gde atributi snažnijih jedinki imaju tendenciju da se prenesu u naredne generacije. Pravilo igre je "preživljavanje najprilagođenijih će pobediti."

Genetski algoritam se razlikuje od klasičnog, derivatima zasnovanog optimizacionog algoritma na dva glavna načina, kako je sažeto u sledećoj tabeli.

Klasični Algoritam	**Genetski Algoritam**
Generiše jednu tačku u svakoj iteraciji. Niz tačaka se približava optimalnom rešenju.	Generiše populaciju tačaka u svakoj iteraciji. Najbolja tačka u populaciji se približava optimalnom rešenju.
Bira sledeću tačku u nizu determinističkim računanjem.	Bira sledeću populaciju računanjem koje koristi generatora slučajnih brojeva.
Manje napredan	Više napredan
Koristi se u oblastima kao što su programiranje i matematika.	Koristi se u oblastima kao što su istraživanje, mašinsko učenje i veštačka inteligencija.

Tabela 1: Poređenje klasičnog algoritma i genetskog algoritma

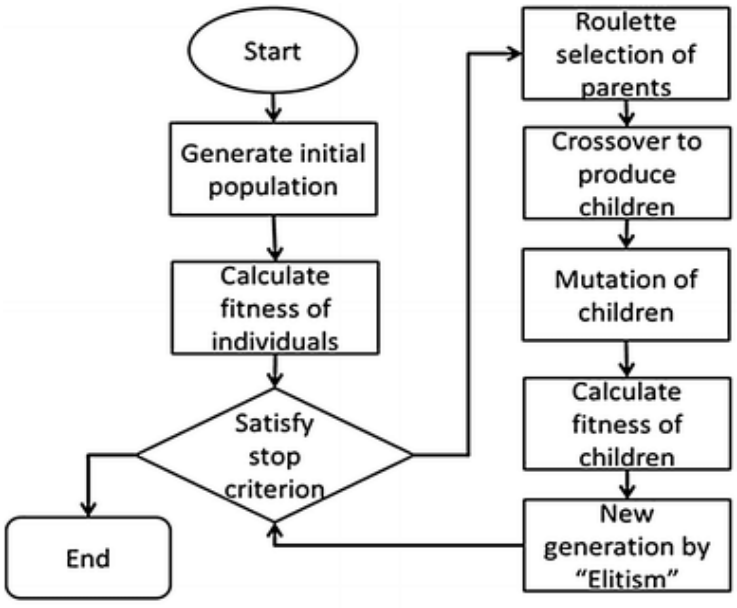
2 Šta su genetski algoritmi

Genetski algoritam je metoda za rešavanje optimizacionih problema, kako sa ograničenjima, tako i bez njih, a zasnovana je na prirodnoj selekciji - procesu koji pokreće biološku evoluciju. Ovaj algoritam ponavlja modifikaciju populacije pojedinačnih rešenja. Na svakom koraku, genetski algoritam bira jedinke iz trenutne populacije da budu roditelji i koristi ih za stvaranje potomstva koje čini sledeću generaciju. Kroz uzastopne generacije, populacija "evoluira"ka optimalnom rešenju.

Genetski algoritam može se primeniti na različite optimizacione probleme koji nisu pogodni za standardne algoritme optimizacije. To uključuje probleme u kojima je ciljna funkcija neprekidna, nediferencijabilna, stohastična ili visoko nelinearna. Takođe, ovaj algoritam može rešavati probleme mešovitog celobrojnog programiranja, gde su neki od parametara ograničeni da budu celobrojne vrednosti.

3 Kako radi genetski algoritam

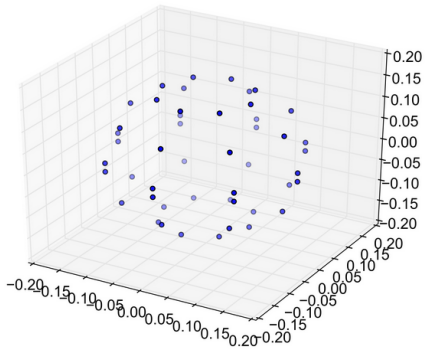
Genetski algoritmi su podskup evolutivnih algoritama, grupe pretrage i optimizacije inspirisane prirodnim procesom evolucije. Evolutivni algoritmi obično koriste evolutivne operacije selekcije, varijacije i zamene kako bi poboljšali ili zamenili populacije generativno, s ciljem unapređenja ukupnog najprilagođenijeg rešenja. Primer ovog ciklusa procesa prikazan je na slici 1.



Slika 1: Nasumična inicijalizacija

3.1 Inicijalizacija

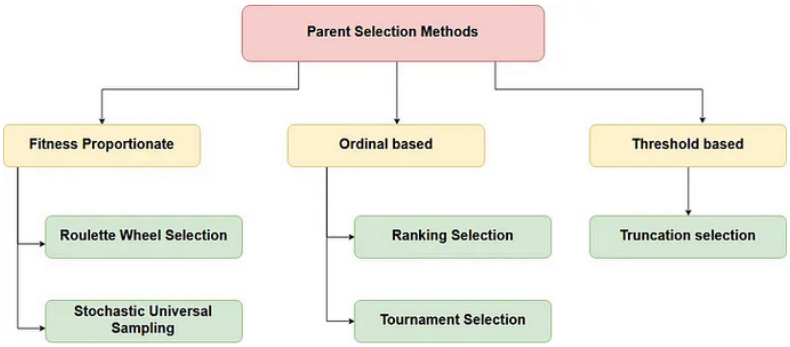
Evolutivni proces počinje inicijalizacijom, gde se generiše početna populacija kandidatskih rešenja. Postoji mnogo različitih metoda inicijalizacije populacija, ali kod genetskih algoritama najpopularnija metoda inicijalizacije je jednostavno stvaranje populacije nasumično inicijalizovanih binarnih nizova. Kada je inicijalna populacija kreirana, počinje evolutivni ciklus generacija.



Slika 2: Nasumična inicijalizacija

3.2 Selekcija

Na svakom koraku generacije, iz populacije roditelja se bira skup roditelja na osnovu vrednosti prilagođenosti svakog pojedinca pomoću mehanizma selekcije, tako da će najprilagođeniji pojedinci imati veću verovatnoću prenosa genetskog materijala na sledeće generacije. Ova izabrana populacija (poznata kao "populacija roditelja") zatim čini osnovu za sve naredne optimizacije tokom koraka generacije.



Slika 3: Tipovi selekcije

3.2.1 Selekcija proporcionalna fitnesu

Selekcija proporcionalna fitnesu je jedan od najpopularnijih načina odabira roditelja. U ovom metodu, pojedinac može postati roditelj sa verovatnoćom koja je proporcionalna njegovom fitnesu. Dakle, jedinke koje su fitnes imaju veću šansu za reprodukciju i prenošenje svojih karakteristika na sledeću generaciju. Ovaj pristup primenjuje pritisak selekcije na fitnije jedinke u populaciji, evoluirajući bolje jedinke tokom vremena.

3.2.2 Selekcija bazirana na rangu

Populacija se sortira prema vrednostima ciljeva. Fitnes koji je dodeljen svakom pojedincu zavisi samo od njegove pozicije u rangu jedinki. Ranking uvodi uniformno skaliranje kroz populaciju i pruža jednostavan i efikasan način kontrole selektivnog pritiska. Verovatnoća da svaki pojedinac bude izabran za reprodukciju zavisi od njegovog fitnesa normalizovanog po ukupnom fitnesu populacije.

3.2.3 Threshold-based Selekcija

Truncation selekcija je veštačka metoda selekcije koja se koristi za roditelje u velikim populacijama ili masovnom odabiru. Ova metoda uključuje sortiranje jedinki prema njihovom fitnesu, pri čemu samo najbolji postaju roditelji. Parametar praga odsječka (Trunc) koristi se za određivanje proporcije populacije koja će biti izabrana kao roditelji. Jedinke ispod praga odsječka ne proizvode potomstvo.

3.3 Ukrštanje

Ukrštanje, takođe poznato kao rekombinacija, predstavlja ključni operator varijacije u genetskim algoritmima. Ovaj proces uključuje uzimanje parova roditelja iz populacije roditelja i kombinovanje njihovih genetskih informacija kako bi se stvorila potomstva. Evo objašnjenja ovog procesa:

1. **Selekcija roditelja:** Parovi roditelja se nasumično biraju iz populacije roditelja, često sa zamjenom. Ovo znači da isti roditelj može biti izabran više puta kako bi stvorio potomke.

2. **Ukrštanje u jednoj tački:** Proces ukrštanja obično se odvija u jednoj tački. Nasumično se bira jedna tačka u oba roditeljska hromozoma.

3. **Zamena genetskih informacija:** Na odabranoj tački ukrštanja, genetske informacije roditelja se razmenjuju. Ova zamena stvara nove jedinke sa kombinacijom osobina oba roditelja.

4. **Stvaranje populacije dece:** Proces ukrštanja se sprovodi sa nasumično odabranim parovima roditelja sve dok nova đečja"populacija ne dostigne istu veličinu kao i originalna populacija roditelja.

Ovaj proces uvodi raznolikost u populaciju kombinovanjem genetskog materijala, simulirajući prirodnu genetsku rekombinaciju koja se dešava tokom reprodukcije. Pomaže istraživanju različitih kombinacija osobina i može dovesti do otkrića novih rešenja u prostoru pretrage.

Parent 1	1, 1, 1, 1, 1, 0, 0,	0, 1, 0
Parent 2	0, 0, 1, 1, 1, 1, 1,	1, 1, 0

Child 1	1, 1, 1, 1, 1, 0, 0,	1, 1, 0
Child 2	0, 0, 1, 1, 1, 1, 1,	0, 1, 0

Slika 4: Ukrštanje u jednoj tački između dva roditeljska GA hromozoma

3.4 Mutacija

Dok ukrštanje jednostavno razmenjuje postojeće informacije između parova kandidatskih rešenja, mutacija u evolutivnim algoritmima obično predstavlja standardni način uvođenja "novog"genetskog materijala u populaciju. Mutacije imitiraju biološke promene i koriste se za održavanje genetske raznovrsnosti između generacija populacije. One uvode nasumične male promene, zadržavajući tako raznolikost.

Original	1, 1, 1, 1, 1, 0, 0, 0, 1, 0
Mutated	1, 1, 0, 1, 1, 0, 0, 1, 1, 0

Slika 5: Mutacija bita na GA hromozomu

Ova genetska varijacija imitira biološke mutacije i koristi se za očuvanje genetske raznovrsnosti između generacija populacije, uvodeći nasumične male promene.

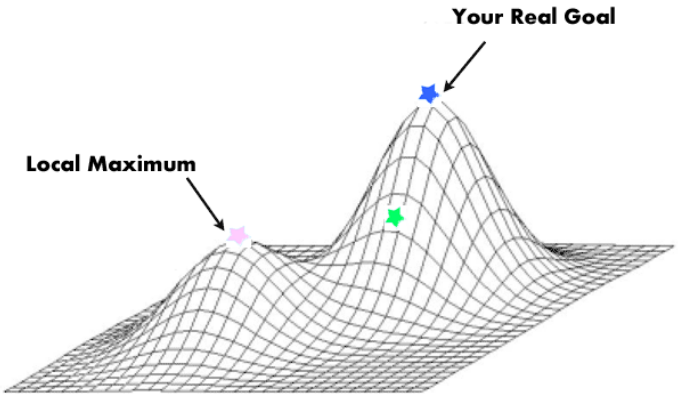
3.5 Evaluacija

Nakon stvaranja populacije dece, svi potomci moraju biti ocenjeni kako bi im se dodelila vrednost prilagođenosti, na osnovu koje mogu biti rangirani u odnosu na svoje vršnjake. Ova prilagođenost se koristi za sortiranje populacije i postavljanje verovatnoća za faze selekcije i zamene tokom procesa pretrage, omogućavajući najprilagođenijim jedinkama veću šansu za prenos genetskog materijala.

Funkcija prilagođenosti, jednostavno definisana, predstavlja funkciju koja uzima kandidatsko rešenje problema kao ulaz i kao izlaz daje ocenu o tome koliko je to rešenje "prilagođeno"ili đobro"u odnosu na razmatrani problem. Izračunavanje vrednosti prilagođenosti se vrši više puta u genetskom algoritmu, stoga bi trebalo da bude dovoljno brzo. Sporo izračunavanje vrednosti prilagođenosti može negativno uticati na performanse genetskog algoritma, čineći ga izuzetno sporim. U većini slučajeva, funkcija prilagođenosti i ciljna funkcija su iste, s obzirom da je cilj ili maksimizovati ili minimizovati zadatu ciljnu funkciju. Međutim, kod složenijih problema sa više ciljeva i ograničenjima, Dizajner Algoritma može odabrati različitu funkciju prilagođenosti.

Funkcija prilagođenosti trebalo bi da poseduje sledeće karakteristike:

- Funkcija prilagođenosti treba biti dovoljno brza za izračunavanje.
- Mora kvantitativno meriti koliko je dato rešenje prilagođeno ili koliko se prilagođeni pojedinci mogu proizvesti iz datog rešenja.



Slika 6: Globalni i lokalni maksimum

Lokalni maksimumi predstavljaju značajan izazov ne samo za genetske algoritme, već i za svaku tehniku optimizacije koja teži pronalaženju globalnog optimuma. Genetski algoritam se dobro snalazi u fazi istraživanja, gde pojedinci otkrivaju delove rešenja i kombinuju ih. Međutim, kada određeni pojedinac postigne lokalno optimalnu tačku, uspeva da zadrži prednost tokom nekoliko iteracija, čime svi pojedinci postaju slični. Dominantni pojedinac opstaje kroz generacije, doprinosi u razmeni genetskog materijala i širi svoje gene drugim

kandidatima. Udaljeni istraživači gube efikasnost i postepeno nestaju. Napredak se zaustavlja kada diverzitet opadne. Početni pristup uključuje uvođenje diverziteta u funkciju prilagođenosti [Vinston, 1992; Juret, 1992]. Ova metoda naseljava lokalne maksimume kako bi ih izbegla, kažnjavajući jedinke blizu naseljenih maksimuma i nagrađujući udaljene istraživače. Izazovi uključuju nejasno integrisanje prilagođenosti i diverziteta, kao i povećanje veličine populacije sa porastom otkrivenih maksimuma.

3.6 Zamena

Završni čin koraka generacije je zamena stare "N"populacije novom "N+1"populacijom dece. Iako postoji mnogo metoda, najtipičnija metoda zamene je Generacijska zamena, gde se cela prethodna generacija zamenjuje novom generacijom dece.

3.7 Završetak

Genetski algoritam obično završava nakon unapred definisanog broja generacija ili ako je ispunjen određeni kriterijum zaustavljanja (npr. prilagođenost je iznad nekog praga, stopa greške je ispod nekog praga itd). Najprilagođenije rešenje u populaciji zatim se vraća kao ukupno najbolje rešenje.

4 Primena genetskih algoritama

- **Trgovanje:** Genetski algoritmi se često koriste u kvantitativnom trgovanju na institucionalnom nivou. Iako su prvobitno razvijeni za rešavanje optimizacionih problema, genetski algoritmi su prilagođeni i primenjeni za analizu finansijskih tržišta.
- **VLSI (Very Large Scale Integration):** Genetski algoritam se koristi za fizički dizajn VLSI čipova. Algoritam istovremeno optimizuje postavku ćelija i ukupno rutiranje, doprinoseći poboljšanju performansi čipova.
- **Dizajn filtera:** Morfološki filteri su važna klasa nelinearnih filtera za digitalnu obradu i analizu signala. Genetski algoritmi se koriste za traženje optimalnih morfoloških filtera za specifične zadatke obrade signala.
- **Robotika:** Primena genetskog algoritma u robotici ogleda se u navigacionim sistemima robota. GA pomaže u dizajniranju navigacionih sistema koji omogućavaju robotima da bezbedno i efikasno putuju kroz svoje okoline.
- **Igranje igara:** Genetski algoritmi se koriste za rešavanje problema teorije igara. Predstavljen je novi metod za rešavanje teorije igara i pronalaženje optimalne strategije za igrača A ili igrača B.
- **MATLAB:** Genetski algoritam se može primeniti za rešavanje problema koji nisu pogodni za standardne optimizacione algoritme. Postoji poseban

GA Toolbox razvijen za MATLAB, koji pruža praktično i efikasno rešenje za modeliranje, dizajn i simulaciju.

- **Detekcija lica:** Na Univerzitetu u Novom Meksiku dizajniran je sistem za prepoznavanje lica. Binarni hromozom se koristi za kodiranje pet karakteristika lica, a GA se koristi za optimizaciju kako bi se dobio optimalni rezultat.

- **Kontrola:** U inženjeringu sistema za kontrolu, GA se primenjuje na brojne metodologije kontrole. Optimizacija parametara kontrolera i konfiguracija poboljšava ukupne performanse sistema.

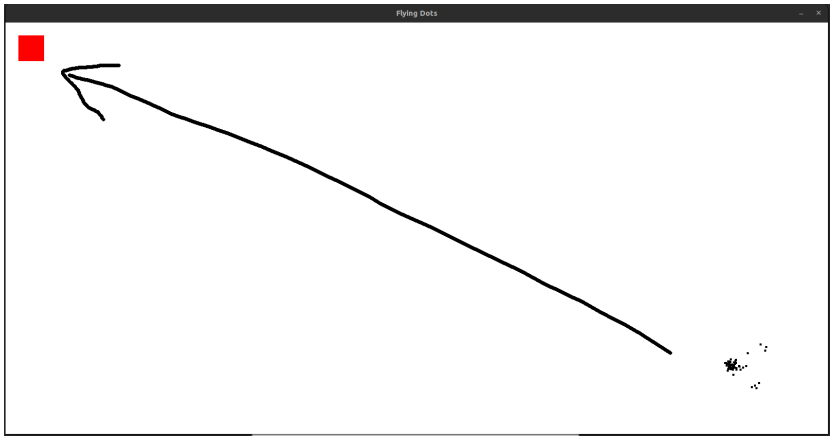
- **Prepoznavanje govora:** U automatskom sistemu prepoznavanja govora, govorni obrasci (testni uzorci) obično se identifikuju sa unapred pohranjenim govnim obrascima. GA se koristi za prevazilaženje izazova u registraciji vremena testnih i referentnih uzoraka.

5 Simulacija genetskog algoritma za rešavanje lavirinta

U ovoj sekciji ću predstaviti simulaciju genetskog algoritma za rešavanje lavirinta. Implementacija je u Python-u, koristeći Pygame biblioteku. Da bismo započeli razvoj algoritma za rešavanje lavirinta, počećemo sa simulacijom jednostavnog traganja za putem do cilja, bez ikakvih prepreka.

5.1 Simulacija 1: Jednostavno Traženje Puta do Cilja

U ovoj simulaciji, tačke će pokušati pronaći put do cilja u lavirintu bez prisustva prepreka. Svaka tačka koristi genetski algoritam sa svojim mozgom koji generiše pravce kretanja. Prikaz simulacije možete videti na slici 7.



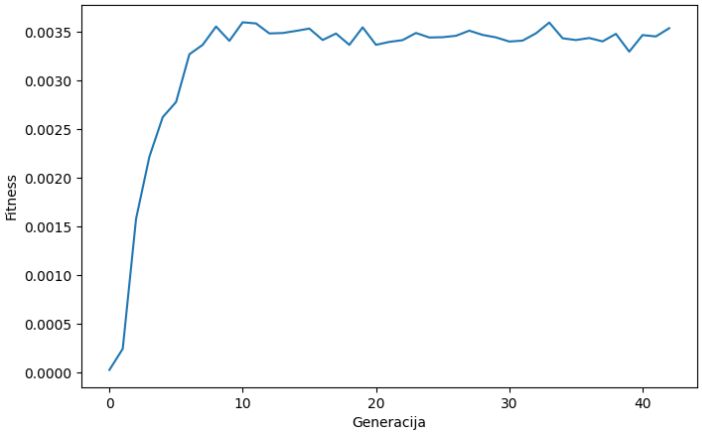
Slika 7: Prva simulacija

5.1.1 Postavke

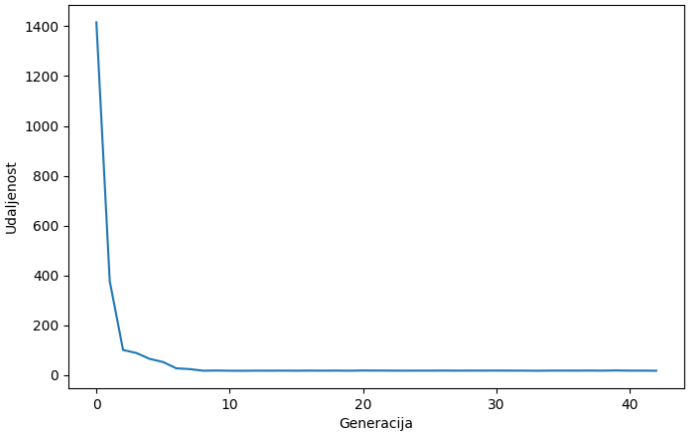
- Veličina populacije: 400
- Broj koraka: 100
- Brzina kretanja tačaka: Random (3-11)
- Interval uglova: 0.1 do 1.9 * 3.14159
- Interval promene uglova: -0.1 do 0.1
- Broj dodatnih koraka: 0

5.1.2 Rezultati

Nakon izvršavanja ove simulacije, analiziraćemo performanse tačaka u pronalaženju puta do cilja. Prikazaćemo rezultate u vidu grafika na slikama 8 i 9.



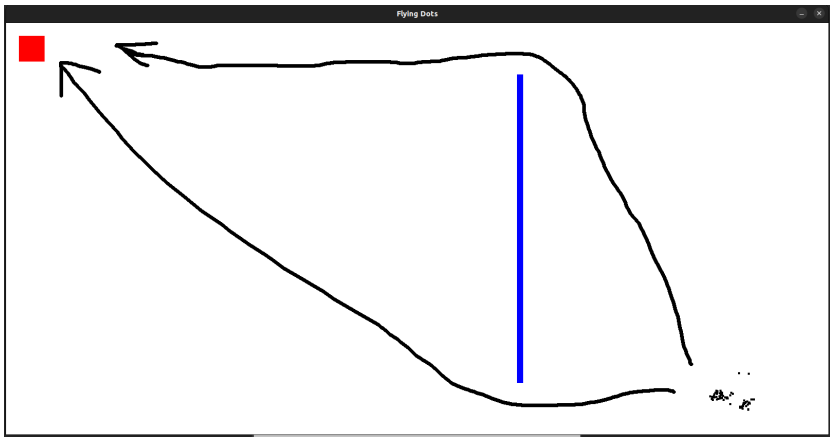
Slika 8: Prosečna vrednost funkcije prilagođenosti tačaka



Slika 9: Prosečna udaljenost tačaka od cilja

5.2 Simulacija 2: Traženje Puta s Preprekom

U ovoj simulaciji, tačke će se suočiti s izazovom pronalaženja puta do cilja koji je otežan prisustvom prepreke na sredini puta. Svaka tačka i dalje koristi genetski algoritam sa svojim mozgom za generisanje pravaca kretanja. Prikaz ove simulacije možete videti na slici 10.



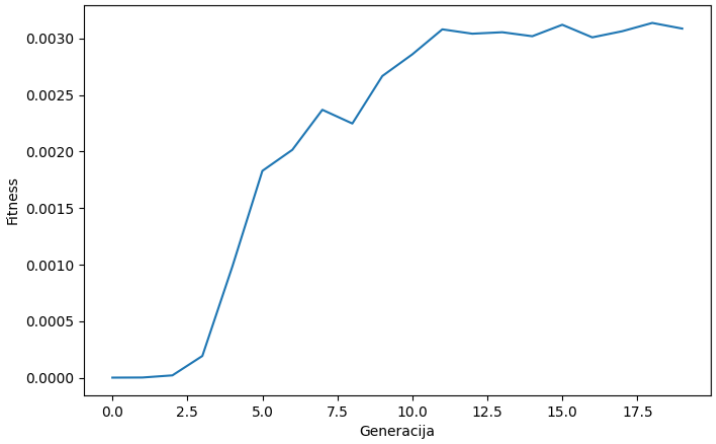
Slika 10: Druga simulacija

5.2.1 Postavke

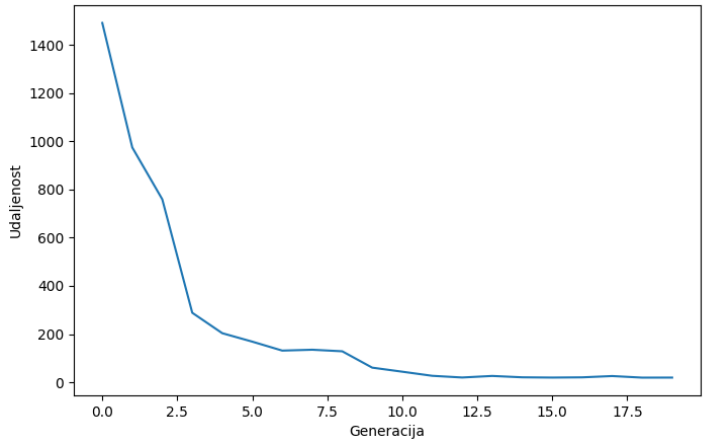
- Veličina populacije: 400
- Broj koraka: 100
- Brzina kretanja tačaka: Random (3-11)
- Interval uglova: 0.1 do 1.9 * 3.14159
- Interval promene uglova: -0.1 do 0.1
- Broj dodatnih koraka: 0

5.2.2 Rezultati

Nakon izvršavanja ove simulacije, analiziraćemo performanse tačaka u suočavanju s preprekom na putu do cilja. Prikazaćemo rezultate na slikama 11 i 12.



Slika 11: Prosečna vrednost funkcije prilagođenosti tačaka



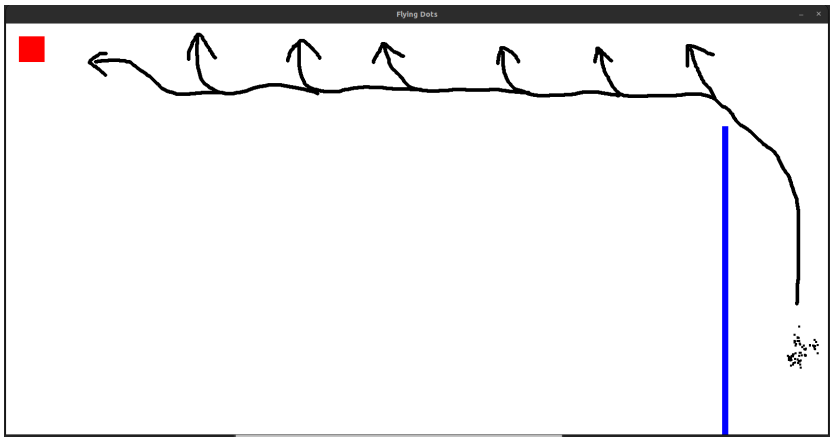
Slika 12: Prosečna udaljenost tačaka od cilja

5.2.3 Zaključak

Simulacije 1 i 2 pokazale su impresivne rezultate u brzom pronalaženju puta do cilja u veoma malom broju generacija. Međutim, postavlja se pitanje kako će se algoritam ponašati u uslovima kada se zid približi tačkama, ograničavajući njihovu slobodu kretanja.

5.3 Simulacija 3: Prepreka koja Ograničava Kretanje

U ovoj simulaciji postavljamo prepreku blizu tačaka kako bismo im ograničili slobodu kretanja. Primećujemo da tačke i dalje uspevaju da stignu do cilja, ali u nekim slučajevima to se odvija sporije zbog ograničenog istraživanja prostora.



Slika 13: Treća simulacija

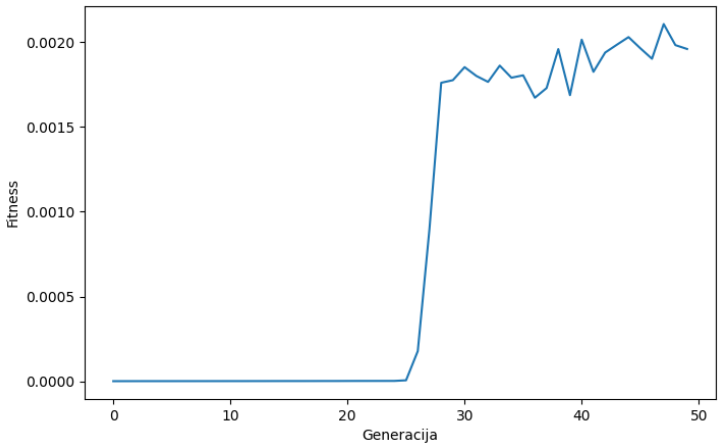
5.3.1 Postavke

Postavke za ovu simulaciju verovatno su slične prethodnim:

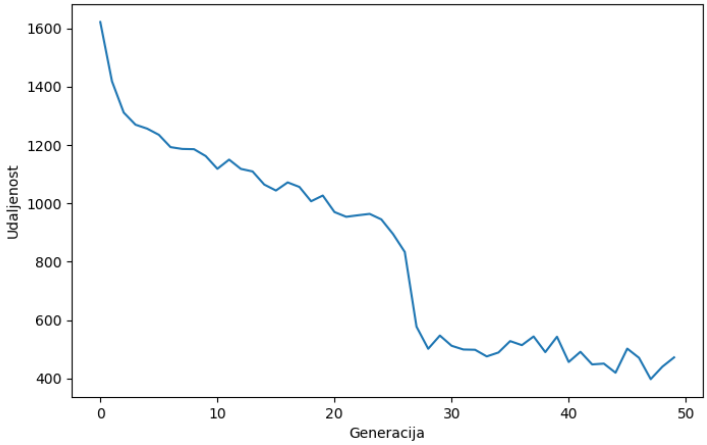
- Veličina populacije: 400
- Broj koraka: 100
- Brzina kretanja tačaka: Random (3-11)
- Interval uglova: 0.1 do 1.9 * 3.14159
- Interval promene uglova: -0.1 do 0.1
- Broj dodatnih koraka: 0

5.3.2 Rezultati

Analiziramo rezultate simulacije kako bismo sagledali kako se algoritam ponaša u uslovima kada je kretanje tačaka ograničeno preprekom. Prikazaćemo rezultate u vidu grafika na slikama 14 i 15.



Slika 14: Prosečna vrednost funkcije prilagođenosti tačaka



Slika 15: Prosečna udaljenost tačaka od cilja

5.3.3 Zaključak

Iako algoritam uspeva da pronađe put do cilja čak i uz prisustvo prepreke koja ograničava kretanje tačaka, primetno je da se to odvija sporije u odnosu na prethodne simulacije. Sledeće pitanje koje se postavlja jeste kako će algoritam reagovati ukoliko se cilj pomeri na donji levi ugao ekrana.

5.4 Simulacija 4: Cilj je još više udaljen od tačaka

U ovoj simulaciji sam pomakao cilj još dalje od tačaka, čime sam dodatno otežao pronalaženje puta. Na slici 16 možete videti kako algoritam reaguje na ovu promenu.

Slika 16: Četvrta simulacija

5.4.1 Postavke

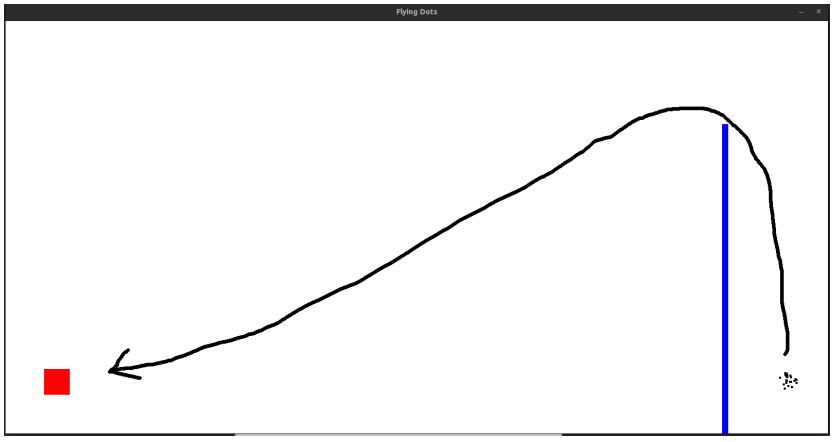
- Veličina populacije: 400
- Broj koraka: 100
- Brzina kretanja tačaka: Random (3-11)
- Interval uglova: 0.1 do 1.9 * 3.14159
- Interval promene uglova: -0.1 do 0.1
- Broj dodatnih koraka: 0

5.4.2 Zaključak

U ovoj simulaciji se pojavio problem lokalnog maksimuma. Algoritam ne istražuje dalji deo puta jer, prilikom pokušaja istraživanja, prvo mora proći kroz područje s manjim fitness rezultatom nego kada udari u prepreku. Kao rezultat toga, tačke se stalno sudaraju u prepreku i ne pokušavaju pronaći put. Ovaj problem ukazuje na potrebu za promenom fitness funkcije, što će biti tema naredne simulacije.

5.5 Simulacija 5: Nadogradnja simulacije 4

U ovoj nadogradnji simulacije 4, promeniću osnovne postavke problema, uključujući fitness funkciju, ukupan broj koraka i broj dodatnih koraka dodanih posle svake generacije. Ove promene omogućavaju algoritmu da pronađe put u malom broju generacija. Na slici 17 možete videti kako algoritam reaguje na ove promene.



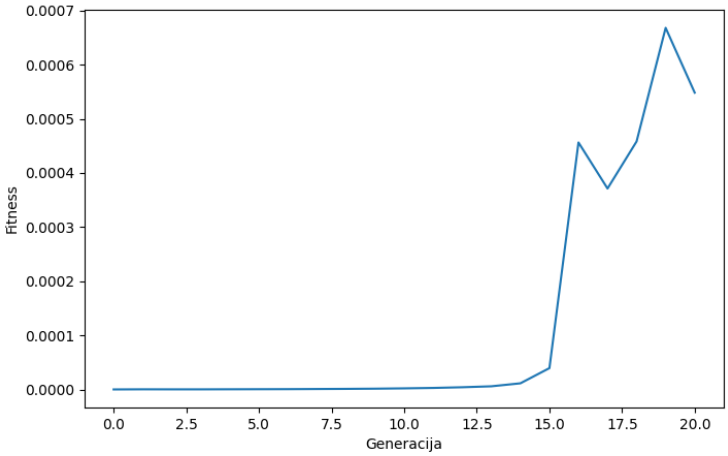
Slika 17: Peta simulacija

5.5.1 Postavke

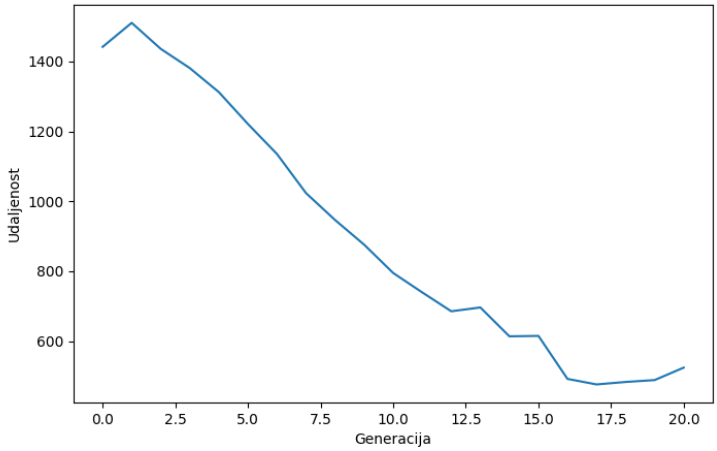
- Veličina populacije: 400
- Broj koraka: 200
- Brzina kretanja tačaka: Random (3-11)
- Interval uglova: 0.1 do 1.9 * 3.14159
- Interval promene uglova: -0.1 do 0.1
- Broj dodatnih koraka: 10

5.5.2 Rezultati

Algoritam uspeva da pronađe putanju do cilja za vrlo kratko vreme. Rezultati su prikazani na slikama 18 i 19.



Slika 18: Prosečna vrednost funkcije prilagođenosti tačaka



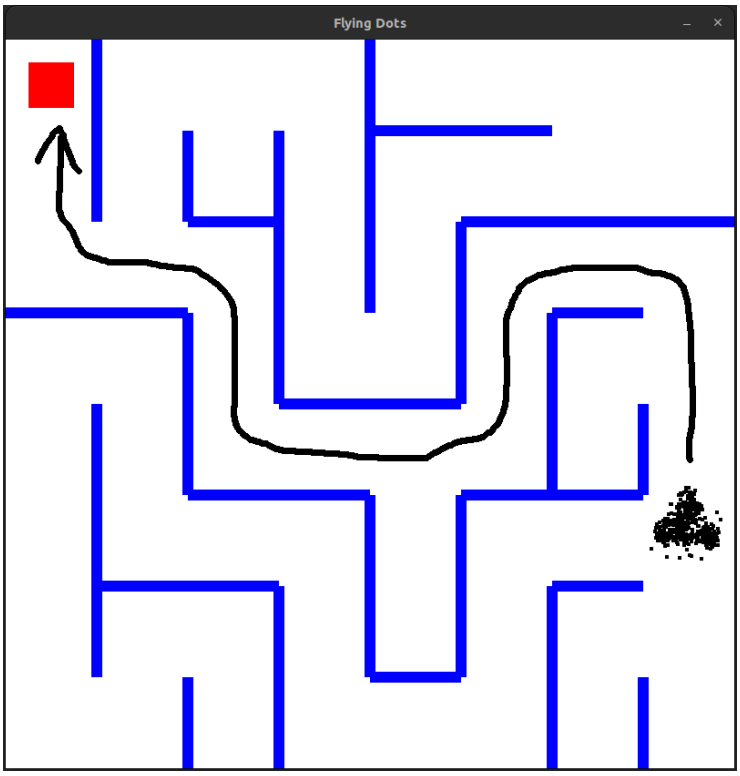
Slika 19: Prosečna udaljenost tačaka od cilja

5.5.3 Zaključak

Na osnovu naših istraživanja, primetili smo da podešavanje funkcije prilagođenosti, zajedno sa malim modifikacijama genetskog algoritma, posebno dodavanjem koraka nakon svake generacije, dovodi do uspešnog pronalaženja cilja sa smanjenim brojem generacija. Sledeći korak u istraživanju biće primena genetskog algoritma u lavirintu, gde se nadamo da će pokazati sličnu efikasnost.

5.6 Simulacija 6 - Nalaženje cilja u lavirintu

U ovoj simulaciji imamo lavirint kroz koji tačke moraju proći do cilja, a želimo da vidimo kako se sada ponaša naš algoritam. Uz malo promena postavki, dobijamo algoritam koji uspešno pronalazi cilj.



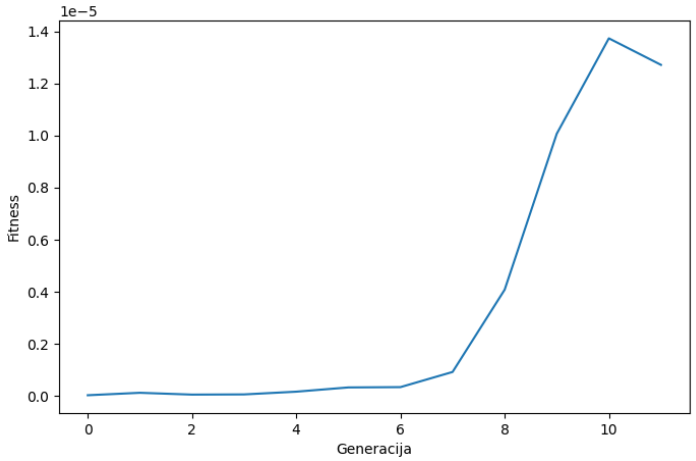
Slika 20: Slika 20 - Prikaz lavirinta

5.6.1 Postavke

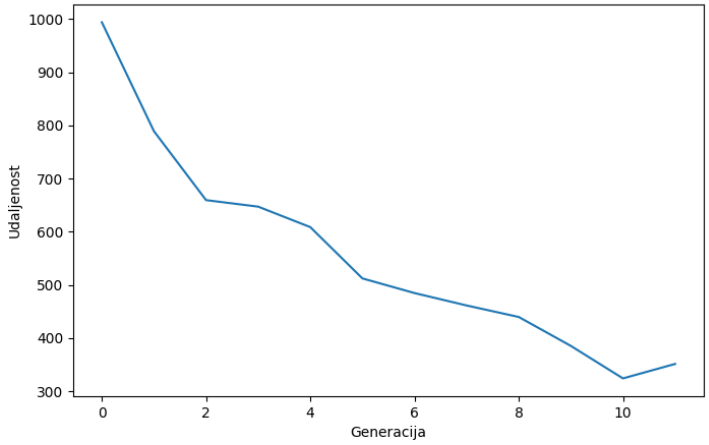
- Veličina populacije: 1000
- Broj koraka: 100
- Brzina kretanja tačaka: Random (3-11)
- Interval uglova: 0.1 do 1.9 * 3.14159
- Interval promene uglova: -0.1 do 0.1
- Broj dodatnih koraka: 50

5.6.2 Rezultati

Kao što možemo primetiti na slikama 21 i 22, algoritam veoma brzo nalazi cilj. Pritom se ne zaustavlja na lokalnim maksimumima, jer je broj koraka koji se dodaju nakon svake generacije povećan. Time je povećano istraživanje celog lavirinta.



Slika 21: Prosečna vrednost funkcije prilagođenosti tačaka



Slika 22: Prosečna udaljenost tačaka od cilja

6 Zaključak

Genetski algoritmi predstavljaju svestran alat sa primenom u raznim oblastima, kako pokazuje i ova simulacija. Vidimo da se genetski algoritam uspešno prilagođava rešavanju problema pronalaženja cilja u lavirintu. Važno je napomenuti da, sa povećanjem kompleksnosti problema, može biti potrebno prilagoditi sam problem, uključujući i promenu funkcije prilagođenosti (fitness function). S obzirom na troškove izračunavanja, gde je skuplja funkcija manje koristan i sporiji je algoritam, važno je pažljivo odabrati prilagođeni pristup za optimalne rezultate.

Dodatno, tokom ove simulacije, primećeno je da povećanje broja koraka nakon svake generacije doprinosi bržem pronalaženju cilja, omogućavajući algoritmu da izbegne lokalne maksimume i istražuje celokupni lavirint.

7 Literatura

[1] S.N.Sivanandam · S.N.Deepa. "Introduction to Genetic Algorithms". Springer-Verlag Berlin Heidelberg 2008.

[2] Topias Blickle, Loather Thiele. "A Comparison of Selection Schemes used in Genetic Algorithm", Swiss Federal Institute of Technology (ETH) Gloriastresse 35, 8092 Zurich,Nr. 11, December 1995.

[3] www.wikipedia.com

[4] www.google.com

[5] www.mathworks.com

[6] www.sciencedirect.com

[7] www.ieee.org

[8] https://www.pico.net/kb/how-does-a-genetic-algorithm-work

[9] https://www.tutorialspoint.com/genetic-algorithms