SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

SEMINAR

Geometrijski algoritam roja čestica primijenjen na rješavanje igre sudoku

Ivana Stokić

Voditelj: doc. dr. sc. Domagoj Jakobović

Zagreb, travanj, 2010.

**Sadržaj**

[1. Uvod 1](#_Toc261042994)

[2. Sudoku 2](#_Toc261042995)

[3. Geometrijski algoritam roja čestica 3](#_Toc261042996)

[3.1 Topologije 4](#_Toc261042997)

[3.1.1 Von Neumann topologija 5](#_Toc261042998)

[3.1.2 Kružna topologija 5](#_Toc261042999)

[3.2 Konveksna kombinacija 5](#_Toc261043000)

[4. Geometrijski algoritam roja čestica primijenjen na igru sudoku 6](#_Toc261043001)

[4.1 Predstavljanje rješenja 6](#_Toc261043002)

[4.2 Funkcija dobrote 6](#_Toc261043003)

[4.3 Rekombinacija 6](#_Toc261043004)

[4.4 Mutacija 7](#_Toc261043005)

[5. Programsko ostvarenje 8](#_Toc261043006)

[5.1 Razred Jedinka 8](#_Toc261043007)

[5.2 Razred GeometricPSO 8](#_Toc261043008)

[5.3 Grafičko sučelje 9](#_Toc261043009)

[6. Testiranje i rezultati 12](#_Toc261043010)

[7. Zaključak 20](#_Toc261043011)

[8. Literatura 21](#_Toc261043012)

[9. Sažetak 22](#_Toc261043013)

# Uvod

Geometrijski algoritam roja čestica formalno je poopćenje algoritma roja čestica. Originalni algoritam prepostavlja kontinuirani prostor pretraživanja i čestice predstavljene nizom realnih brojeva. Postoji mnogo različitih inačica algoritma (proširenja na kombinatorni prostor i probleme). Jedna od njih je i nedavno izložen geometrijski algoritam roja čestica koji će biti opisan u ovom radu.

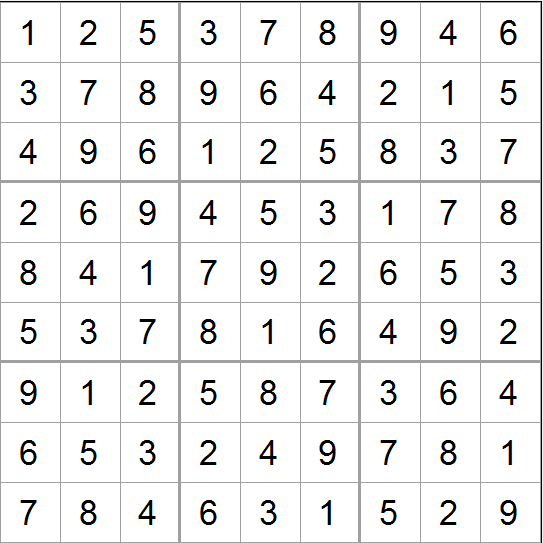
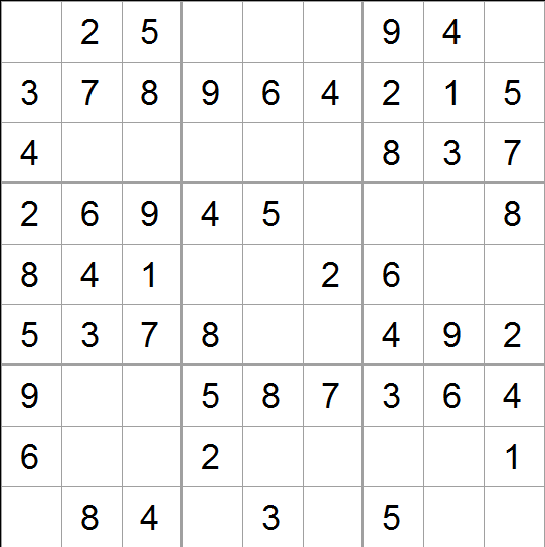
U prvom poglavlju ukratko je općenito opisan geometrijski algoritam roja čestica te je dan pseudokod algoritma. U drugom poglavlju predstavljen je problem, opisana je igra sudoku i način dobivanja zadovoljavajućeg rješenja. U trećem poglavlju objašnjena je primjena algoritma na rješavanje problema. U četvtom poglavlju predstavljeno je programsko ostvarenje, a u petom su prikazani rezultati testiranja uspješnosti implementiranog algoritma.

# Sudoku

Sudoku je popularna logička igra. Na japanskom jeziku sudoku (数独) znači jedan broj (engl. *single number*). Igra je danas vrlo popularna i često je se naziva “ Rubikovom kockom 21. stoljeća ” [6].

Igra se sastoji se od 81 polja u tablici veličine 9x9. Tablica je podijeljena na 9 manjih podtablica veličine 3x3. Cilj je popuniti sva polja u tablici, ali tako da se u jednom stupcu, jednom retku i jednoj 3x3 podtablici nalaze svi brojevi od 1 do 9 i svaki od tih brojeva smije se pojaviti samo jednom. Na početku su neka polja popunjena brojevima, a neka su prazna. Igrama se mogu pridijeliti i težine, a to ovisi o broju početnih praznih polja.

Primjer neriješene jednostavnije igre sudoku i njenog rješenja nalazi se na slici 2.1.



Slika 2.1. Neriješeni sudoku (lijevo) i rješenje (desno)

Sudoku obično ima jedno jedinstveno rješenje (može ih imati i više ako je na početku otkriveno malo brojeva) i, iako su pravila rješavanja vrlo jednostavna, najčešće je rješenje “ručno”, a brzo, dosta teško otkriti. Međutim, računala ga danas mogu vrlo brzo otkriti.

Općeniti problem rješavanja igre sudoku, slagalice s podtablicama veličine poznat je kao NP problem [1].



Postoji mnogo različitih algoritama koji se mogu upotrijebiti za rješavanje ove igre. Za rješavanje težih inačica najčešće se koristi *backtracking* algoritam, a mogu se koristiti i prirodom inspirirani optimizacijski algoritmi, kao npr. genetski algoritam, simulirano kaljenje, algoritam roja čestica i sl.

# Geometrijski algoritam roja čestica

Najpoznatiji, običan, algoritam roja čestica predstavili su, u svom radu, 1995. godine R. C. Eberhart i J. Kennedy [4]. Ideja za algoritam nastala je sasvim slučajno, promatrajući jato ptica, otkriveno je da se sve ptice koje u jatu lete ponašaju prema određenim pravilima: svaka ptica izbjegava sudar s nekom drugom pticom, usklađuje brzinu leta s bliskim pticama te pokušava ostati u blizini drugih ptica. Čestice lete kroz prostor pretraživanja i svoju poziciju mijenjaju na temelju vlastitog iskustva i iskustva svojih bliskih susjeda.

Postoji mnogo različitih inačica ovog algoritma, npr. doLPSO (engl. *Division of Labour*), ARPSO (engl. *Attraction and Repulsion PSO*), Meta – PSO. Geometrijski algoritam roja čestica detaljnije je opisan u nastavku.

Geometrijski algoritam roja čestica izmijenjen u odnosu na običan algoritam kako bi se lakše riješili neki kombinatorni optimizacijski problemi.

Za razliku od običnog algoritma roja čestica, geometrijska inačica ne uzima u obzir brzinu čestica, jednadžba računanja pozicije čestice zamijenjena je rekombinacijom koja zadovoljava sva svojstva konveksne kombinacije (engl. *convex combination*), postoji mutacija i parametri moraju biti pozitivni i njihov zbroj uvijek mora biti jednak jedan [1].



Parametar je faktor inercije (najučinkovitije vrijednosti 0 ili 0.2), a faktori ( 0.2, 0.4 ili 0.6) modeliraju jakost privlačne sile između najboljih rješenja i čestice (engl. *sociality and memory*) [1].



Na slici 3.1 prikazan je pseudokod modificiranog algoritma roja čestica (engl. G*PSO*).

Inicijaliziraj početnu populaciju

Dok nije zadovoljen uvjet završetka

Za svaku česticu i

pbest[i] = najbolje rješenje koje je čestica *i* do sad pronašla

gbest[i] = najbolje rješenje cijelog roja

Za svaku česticu *i*

čestica[i] = CX((čestica [i], ), (gbest[i], (pbest[i], ))



Mutiraj česticu[i]

Slika 3.1 Pseudokod algoritma GPSO

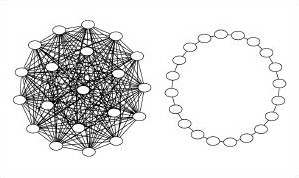
Uvjet završetka algoritma može biti dosegnut određen broj generacija, postignuta određena vrijednost funkcije dobrote i sl. Dok uvjet završetka nije zadovoljen, za svaki česticu iz populacije, roja (koji na početku može biti generiran slučajno) ponavljat će se postupak. Svaka čestica (engl. *particle*) osvježit će svoj *pbest* i *gbest* (*gbest* se može i naknadno odrediti, a svoj *pbest* svaka čestica mora znati u svakom trenutku). *Pbest* je najbolje vlastito rješenje koje je čestica sama do tada pronašla, a *gbest* je globalno najbolje rješenje kojeg zna cijeli roj (engl. *swarm*).

Globalno rješenje može se odrediti na različite načine, već prema tome o kojoj se topologiji susjedstva radi.

CX (engl. *convex combination*) je rekombinacija koja radi s tri elementa, a detaljnije je opisana u poglavlju 4.3.

## Topologije

Postoji mnogo različitih topologija, a najpoznatije su one globalne i lokalne. Globalna topologija, još se naziva i zvjezdasta (engl. *star*), mogla bi se prikazati kao potpuno povezan graf . Svaka čestica povezana je sa svakom česticom i globalno najbolje rješenje cijelog roja bit će najbolja čestica u cijelom roju, u cijeloj populaciji. Globalna topologija prikazana je na slici 3.1.



Slika 3.1 Globalna topologija (lijevo) i lokalna topologija (desno)

Lokalnih topologija (engl. *lbest*) ima više (ako se o njima radi, tad ne postoji globalno rješenje za sve čestice, već svaka čestica ima svoje najbolje lokalno rješenje koje uzima iz svog susjedstva): von Neumann (engl. *lattice topology*), zatim kružna (engl. *ring topology*), na slici 3.1, hibridna topologija (kombinacija svih triju prije spomenutih topologija). Prilikom pretraživanja prostora stanja, kad čestici zatreba globalno najbolje rješenje, ona će pogledati svoje najbolje rješenje te najbolje rješenje svojih susjeda. Susjedi neke čestice nisu definirani prema blizini u prostoru rješenja.

Primjena lokalnih topologija je korisna, jer one sprječavaju preranu konvergenciju nekom rješenju koje možda i nije globalno rješenje problema. Korištenje lokalnih topologija omogućava česticama (sporije) istraživanje većeg prostora. Ako čestica pronađe bolje rješenje, ono će izravno utjecati samo na njene susjede, a trebat će puno više vremena kako bi utjecalo na susjede njenih susjeda. Kod globalne topologije samo je jedna najbolja čestica u roju pa tada sve čestice obično konvergiraju u istu točku (sve vide najbolju globalnu česticu). To nije toliko loše, jer čestice mogu pronaći i bolje rješenje dok se kreću prema do sad najboljem, međutim, konvergencija prema lokalnom optimumu kod globalne je topologije vrlo čest slučaj [5].

### Von Neumann topologija

Von Neumann topologiju predstavili su Kennedy i Mendes [5]. Kod ove se topologije sve čestice u roju mogu prikazati u dvodimenzionalnom polju. Susjedstvo svake čestice čine najbolje čestice koje se nalaze desno, lijevo, gore i dolje u dvodimenzionalnom polju i ona sama, tj. njeno do sad pronađeno najbolje rješenje. Primjer topologije von Neumann nalazi se na slici 3.2.

Slika 3.2. Von Neumann topologija

### Kružna topologija

Kružna topologija ima oblik kružnice. To je lokalna topologija koja je prikazana na slici 3.1 (desno). Svaka čestica povezana je sa svojih K susjeda, najčešće je K = 2, tad se radi o lijevom i desnom susjedu. Kod ove topologije potrebno je više vremena za propagaciju novog globalnog rješenja na kraj kružnice.

Postoje još neke topologije koje se rjeđe koriste, npr. piramidalna topologija (3D) i slučajna topologija [8].

## Konveksna kombinacija

Kod geometrijskog algoritma roja čestica koristi se konveksna kombinacija. Ona definira konveksnu ljusku. Zahtjevi kod konveksne kombinacije za točke u metričkom prostoru su [1]:

* Konveksne težine – sve težine moraju biti veće od nule i njihov zbroj mora biti jednak jedan.
* Konveksnost – operator CX kombinira *pbest[i],* *gbest[i]* i česticu *i* te za rezultat (dijete) daje točku u njihovoj konveksnoj ljusci za neku metriku *d*.
* Koherencija između težina i udaljenosti – udaljenosti od točke koja predstavlja rješenje (engl. *equilibrium point*) su padajuće funkcije težina
* Simetrija – ista vrijednost pridružena ima jednaku težinu, tj. u trokutu koji predstavlja rješenje, ako svi koeficijenti imaju jednake vrijednosti, udaljenosti do točke rješenja bit će jednake.



Između više čestica (točnije, između triju njih) provodi se rekombinacija (križanje) koje će biti opisano u idućem poglavlju. Tek se tada pokazuje da to križanje zadovoljava sva četiri gore navedena svojstva te da je zbog toga konveksna kombinacija.

# Geometrijski algoritam roja čestica primijenjen na igru sudoku

U ovom poglavlju bit će detaljnije opisan način na koji je sudoku rješavan uz pomoć algoritma GPSO.

## Predstavljanje rješenja

Rješenje, tj. jedna čestica predstavljena je matricom 9x9. Iz datoteke se učitaju elementi početnog problema i oni se kasnije, u tijeku rješavanja, nikada ne mijenjaju. Svaka jedinka ima informaciju o do sad nađenom svojem najboljem rješenju (pbest) i također informaciju o svom vrijednosti funkcije dobrote.

## Funkcija dobrote

Funkcija dobrote predstavljena je zbrojem broja različitih elemenata u svakom retku, svakom stupcu i svakoj 3x3 podtablici. Najbolja je ona jedinka koja ima najveću vrijednost funkcije dobrote, jer to znači da u pojedinom retku, stupcu i podtablici ima sve različite elemente. Najveća vrijednost funkcije dobrote je 243 (), a najmanja vrijednost jednaka je 27 ().



## Rekombinacija

Nad česticama, tj. nad njihovim dijelovima koji nisu fiksni, provodi se rekombinacija. Za nju je potrebna maska. Duljina maske jednaka je duljini pojedinog retka (maska se sastoji od 9 znamenaka). Znamenke se uzimaju iz skupa {1, 2, 3}; broj jedinica, dvojki i trojki proporcionalan je vrijednostima težina ( koje se izračunaju iz početnih parametara ( koje unosi korisnik; svaki se parametar pomnoži s 9 i dobiju se vrijednosti težina. Mora vrijediti , zbroj težina jednak je 9.

Na slici 4.1 prikazan je primjer rekombinacije nad jedinkama duljine 7.

Maska: 1 2 2 3 1 3 2

Roditelj1: 1 2 3 4 5 6 7

Roditelj2: 3 5 1 4 2 7 6

Roditelj3: 3 2 1 4 5 7 6

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Dijete: 1 5 3 4 2 7 6

Slika 4.1 Rekombinacija

Kreće se od maske, slijeva na desno. Njen prvi element je broj 1. To znači da prvi element u roditelju2 i roditelju3 mora postati jednak elementu na tom istom mjestu u roditelju1. U drugom i trećem roditelju tražit će se broj 1 (to je element iz prvog roditelja) i zamijenit će se brojevima 3 iz drugog i trećeg roditelja. Nakon toga će svi roditelji na prvom mjestu u nizu imati element s vrijednošću 1. Idući element u maski je 2, to znači da će element na drugom mjestu (prvom ako se počinje od nule) u prvom i drugom roditelju morati postati jednak elementu na drugom mjestu u roditelju1, a to je broj 2. Ponovo će se tražiti isti elementi u oba roditelja i na kraju će se petice zamijeniti dvojkama. Postupak se ponavlja dok se ne dođe do kraja maske. Niz kojem će biti jednaka sva tri roditelja predstavljat će dijete, u ovom slučaju je to niz 1 5 3 4 2 7 6. Rekominacija se obavlja po svim retcima čestice.

## Mutacija

Mutacija također mijenja samo elemente čestice koji nisu fiksni. Slučajno se zamijene dva elementa u retku. Mijenjaju se samo elementi po stupcima, u svakom retku, jer fiksni elementi zajedno s permutacijama redaka čine čvrsta ograničenja (engl. *hard constraints*). Slaba ograničenja (engl. *soft constraints*) su permutacije stupaca.

# Programsko ostvarenje

Program je napisan u programskom jeziku C#.

## Razred Jedinka

Razred *Jedinka* predstavlja jednu jedinku, česticu u roju. Svaka čestica zna svoje do tad pronađeno najbolje rješenje, zatim vrijednost funkcije dobrote (pozove se metoda *IzracunajFitness()* koja izračuna vrijednost funkcije dobrote za jedinku), zatim zadane elemente učitane iz datoteke (*zadaniElementi* tipa *Matrica*) te stanje koje predstavlja stanje slagalice.

## Razred GeometricPSO

U razredu *GeometricPSO* nalazi se metoda *gPSO* koja za parametre prima float vrijednosti *omega, fi1* i *fi2* koje unosi korisnik, zatim *brGeneracija* koja predstavlja broj generacija, *velicinaPop* što je veličina populacije, *Matricu zadaniElementi* i *zastavicu* tipa string koja određuje o kojoj se topologiji radi, ako je njena vrijednost “*von\_Neumann*”, tada se radi o topologiji von Neumann, ako je vrijednost “*Global*”, tad se radi o globalnoj topologiji, a ako je vrijednost zastavice “*Ring*”, tad se radi o kružnoj topologiji. U nastavku je naveden prototip metode *gPSO*:

Jedinka gPSO(float omega, float fi1, float fi2, int brGeneracija, int velicinaPop, Matrica zadaniElementi, string zastavica);

Na početku se inicijalizira populacija (uz pomoć konstruktora koji postavlja stanje nepopunjenih elemenata matrice (dvodimenzionalnog polja) na neku slučajnu vrijednost koja se već ne pojavljuje u *zadaniElementi* u tom retku). Nakon toga se za svaku generaciju određuje najbolja globalna čestica (ako je zastavica “*Global*”, tada se pronalazi čestica s najvećom vrijednošću funkcije dobrote). Ako je zastavica “*von\_Neumann*” ili “*Ring*”, pronalazi se najbolja lokalna čestica (pronalaze se *pbest* čestice u susjedstvu trenutne jedinke, uključujući i *pbest* trenutne čestice) u metodi *OdrediGlobalnuJedinku*.

Jedinka OdrediGlobalnuJedinku(int velicinaPop, Jedinka trenutna, List<Jedinka> sveJedinke, string zastavica);

Za svaku česticu određuje se najbolja od svih u njenom susjedstvu i najbolja od tih najboljih bit će povratna vrijednost metode *gPSO*, tj. to je najbolja čestica u trenutnoj generaciji.

Metoda *jeLiNajbolja* prima dva argumenta, dvije čestice tipa *Jedinka* i vraća najbolju od tih dviju te se ona sprema u *pbest* trenutne čestice (do sad najbolja pronađena čestica trenutne čestice). Ta se metoda poziva nakon rekombinacije i mutacije, jer se tad mijenja *pbest* – rekombinacijom ili mutacijom može nastati i bolja čestica od trenutne.

Jedinka jeLiNajbolja(Jedinka prethodna, Jedinka trenutna);

Metoda *convexCombination* za parametre prima jednodimenzionalno polje u kojem se nalazi maska, zatim trenutnu česticu, globalno najbolje rješenje (*globalBest*), najbolje rješenje trenutne čestice (*pBest*) i *Matricu zadaniElementi*.

Jedinka convexCombination(int[] maska, Jedinka trenutnaJedinka, Jedinka globalBest, Jedinka pBest, Matrica zadaniElementi);

U toj se metodi odvija rekombinacija (konveksna kombinacija) između triju roditelja (*trenutna čestica, pbest* i *gbest*). Pravi roditelji se ne mijenjaju, već se napravi kopija roditelja i promjene se odvijaju u kopijama roditelja. Metoda vraća novu česticu – dijete. Za svaku rekombinaciju ponovo se generira maska (ne koristi se ista svaki put).

Maska se generira u metodi *generirajMasku*, čiji je prototip prikazan u nastavku:

int[] generirajMasku(float omega, float fi1, float fi2);

U toj se metodi generira maska potrebna za rekombinaciju, računaju se tri nove težine koje se dobiju množenjem unesenih parametara (*omega, fi1*, *fi2*) s 9. Prema veličinama novih težina odredi se broj jedinica, dvojki i trojki koje će sadržavati maska. Vrijednost varijable *omega1* odredit će broj jedinica u maski, vrijednost varijable *omega2* broj dvojki itd. Metoda vraća polje u kojem se nalazi maska.

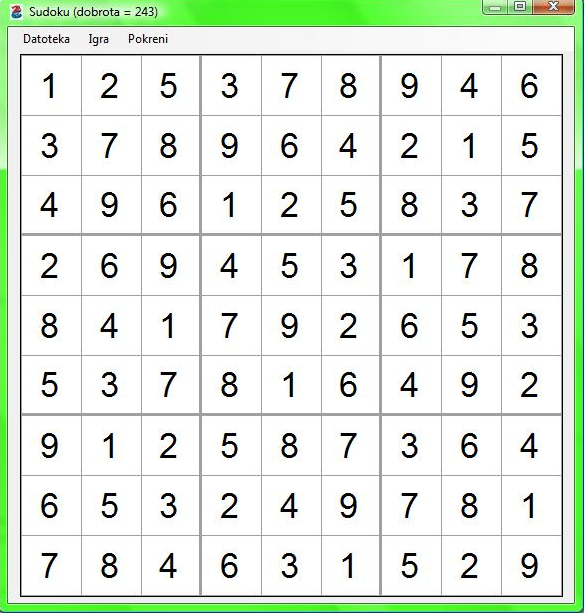
Metoda *mutiraj* uzima jednu česticu i mutira je. Slučajno se odrede dva indeksa (ne smiju biti indeksi na kojima se nalaze fiksni elementi) i ti se elementi zamijene.

Jedinka mutiraj(Jedinka trenutnaJedinka, Matrica zadaniElementi);

Metoda vraća jedinku. Za zamjenu elemenata koristi se pomoćna metoda *zamijeniDvaElementa*.

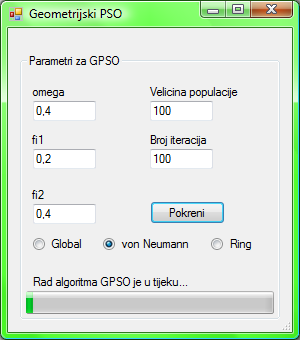
## Grafičko sučelje

Grafičko sučelje sastoji se od dviju formi. Jedna (u *DataGridView-u*) prestavlja slagalicu sudoku, a u drugoj se unose parametri (i može se vidjeti napredovanje algoritma). Prva je prikazana na slici 5.3, a druga na slici 5.4.



Slika 5.3 Prikaz slagalice sudoku

U naslovu forme ispisuje se, u svakoj generaciji, trenutna vrijednost funkcije dobrote. Klikom na “*Igra*” može se učitati lagana, srednja ili teška inačica igre, a klikom na “*Pokreni*” pa na “*Geometrijski PSO*” otvara se druga forma. U njoj postoje mjesta za unos parametara algoritma GPSO, a to su veličina populacije, broj generacija, parametri *omega, fi1* i *fi2* potrebni za određivanje maske. Korisnik mora odabrati jednu od topologija, topologiju von Neumann, globalnu ili kružnu topologiju (Ring).

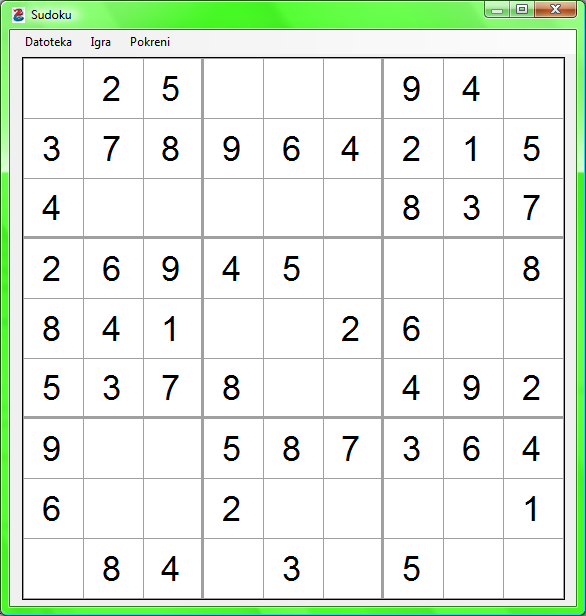


Slika 5.4. Učitavanje parametara za GPSO

# Testiranje i rezultati

Testiranje je izvršeno za različite težine igre sudoku. Ponovljeno je trideset puta za svaku od triju implementiranih topologija s najučinkovitijim parametrima [1]. Za trideset puta izbrojano je koliko je puta, za svaku topologiju posebno, vrijednost funkcije dobrote dosegla 243 (najveća moguća vrijednost).

Prvi test izvršen je na sudoku slagalici koja je prikazana na slici 6.1. To je jednostavnija sudoku slagalica. Prva tri testa provedena su uz parametre omega = 0.4, fi1 = 0.2 i fi2 = 0.2. Veličina populacije jednaka je 100, a broj generacija jednak je 250.



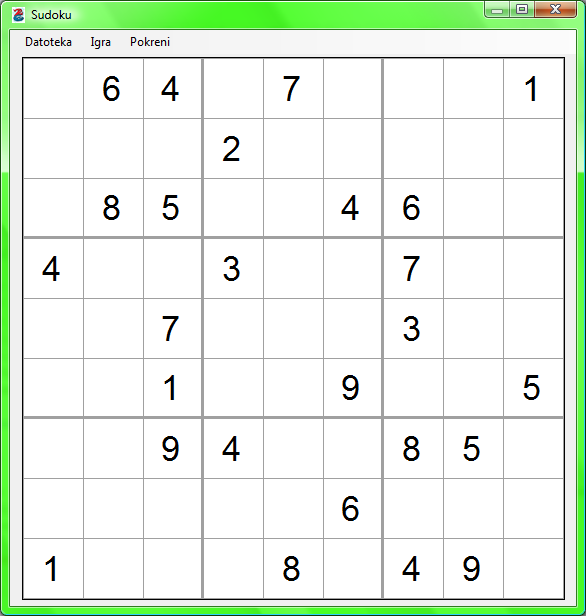
Slika 6.1 Početno stanje sudoku slagalice za prvi test

U tablici 6.1 prikazani su rezultati u 30 mjerenja za svaku od topologija. Broj u zadnjem retku pokazuje koliko je puta dobrota bila jednaka 243. U prvom stupcu nalaze se vrijednosti funkcije dobrote najbolje čestice nakon 250. generacije. Iz rezultata može se vidjeti da je vrijednost funkcije dobrote kod topologije von Neumann 22 puta od njih 30 bila jednaka 243. Kod topologije ring radi se o 14 puta, a za zvjezdastu (globalnu) topologiju o 9 puta (30%).

Tablica 6.1 Vrijednosti funkcije dobrote najbolje jedinke

|  |  |  |
| --- | --- | --- |
| Von Neumann | Ring | Global |
| 243 | 239 | 239 |
| 239 | 243 | 235 |
| 243 | 243 | 239 |
| 243 | 243 | 243 |
| 239 | 239 | 239 |
| 243 | 243 | 224 |
| 235 | 243 | 235 |
| 239 | 239 | 231 |
| 243 | 239 | 243 |
| 243 | 239 | 235 |
| 243 | 243 | 243 |
| 243 | 243 | 232 |
| 235 | 235 | 239 |
| 243 | 239 | 243 |
| 243 | 243 | 235 |
| 243 | 235 | 243 |
| 235 | 243 | 231 |
| 235 | 239 | 243 |
| 243 | 243 | 235 |
| 243 | 239 | 235 |
| 243 | 239 | 243 |
| 243 | 243 | 235 |
| 243 | 243 | 231 |
| 243 | 243 | 231 |
| 243 | 235 | 235 |
| 243 | 231 | 235 |
| 243 | 243 | 239 |
| 243 | 239 | 235 |
| 243 | 239 | 243 |
| 239 | 235 | 243 |
|  |  |  |
| 22 | 14 | 9 |

Na slici 6.2 prikazana je slagalica sudoku za drugi test (teža inačica, s više nepopunjenih polja), a u tablici 6.2 nalaze se rezultati. Testiranje je izvršeno za iste parametre kao prethodno.

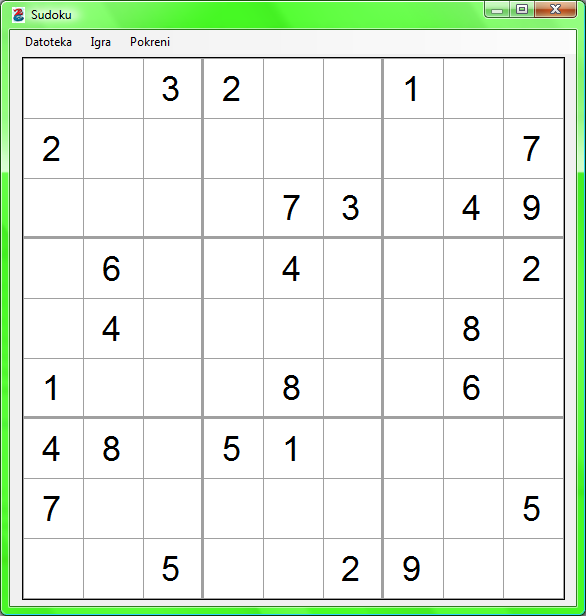


Slika 6.2 Početno stanje slagalice sudoku za drugi test

Tablica .2 Vrijednosti funkcije dobrote najbolje jedinke

|  |  |  |
| --- | --- | --- |
| Von Neumann | Ring | Global |
| 239 | 243 | 243 |
| 243 | 239 | 235 |
| 239 | 235 | 235 |
| 239 | 239 | 235 |
| 243 | 243 | 239 |
| 243 | 243 | 243 |
| 243 | 243 | 243 |
| 243 | 235 | 239 |
| 243 | 235 | 230 |
| 243 | 243 | 235 |
| 243 | 239 | 233 |
| 239 | 243 | 233 |
| 243 | 239 | 243 |
| 243 | 235 | 233 |
| 243 | 235 | 243 |
| 233 | 243 | 243 |
| 243 | 231 | 235 |
| 243 | 239 | 231 |
| 243 | 243 | 239 |
| 243 | 243 | 224 |
| 239 | 239 | 243 |
| 243 | 243 | 243 |
| 243 | 243 | 243 |
| 243 | 243 | 243 |
| 239 | 239 | 243 |
| 243 | 243 | 243 |
| 243 | 243 | 239 |
| 243 | 239 | 243 |
| 243 | 243 | 239 |
| 243 | 243 | 239 |
|  |  |  |
| 23 | 16 | 13 |

Sljedeće testiranje provedeno je nad istim parametrima algoritma, ali na još težoj inačici igre sudoku sa slike 6.3. Rezultati su prikazani u tablici 6.3. Za topologiju von Neumann uspješnost je 19/30, za globalnu 5/30, a za kružnu 18/30.



Slika 6.3 Početno stanje slagalice sudoku za treći test

Tablica 6.3 Vrijednosti funkcije dobrote najbolje jedinke

|  |  |  |
| --- | --- | --- |
| Von Neumann | Ring | Global |
| 243 | 239 | 233 |
| 239 | 243 | 243 |
| 243 | 243 | 235 |
| 235 | 243 | 239 |
| 243 | 243 | 243 |
| 239 | 243 | 229 |
| 243 | 243 | 231 |
| 239 | 243 | 227 |
| 243 | 239 | 243 |
| 239 | 243 | 229 |
| 239 | 243 | 239 |
| 243 | 239 | 235 |
| 239 | 239 | 235 |
| 243 | 233 | 243 |
| 235 | 243 | 239 |
| 243 | 243 | 235 |
| 243 | 243 | 243 |
| 243 | 243 | 231 |
| 243 | 239 | 235 |
| 243 | 243 | 235 |
| 243 | 239 | 231 |
| 243 | 239 | 235 |
| 239 | 235 | 239 |
| 243 | 243 | 235 |
| 235 | 243 | 239 |
| 243 | 243 | 235 |
| 239 | 243 | 239 |
| 243 | 239 | 231 |
| 243 | 239 | 239 |
| 243 | 239 | 231 |
|  |  |  |
| 19 | 18 | 5 |

U sva tri slučaja može se vidjeti da globalna topologija najmanje puta (prilikom 30 pokretanja) dolazi do globalno najboljeg rješenja, dok su preostale dvije topologije bolje. Topologija von Neumann u sva tri slučaja daje najbolje rezultate (u više od 60% slučajeva dobrota dostiže vrijednost 243). Takvi rezultati bili su i očekivani; jasno je da će obje lokalne topologije više puta dati bolja rješenja, jer zbog traženja najbolje lokalne čestice u susjedstvu trenutne (a ne samo jedne globalne ka kojoj će sve konvergirati), čestice dobiju priliku istraživati veći prostor rješenja te je zbog toga manja vjerojatnost da će algoritam zaglaviti u lokalnom optimumu. To se vrlo lako može dogoditi u globalnoj topologiji i zato je ona nešto lošija.

U nastavku je prikazano još jedno testiranje (tablica 6.4) s istom veličinom populacije i brojem generacija i na istoj slagalici sa slike 6.3, ali s novim vrijednostima ostalih parametara (omega = 0, fi1 = 0.4, fi2 = 0.6). Ovog puta bolja je kružna topologija, dok je globalna opet najlošija.

Tablica .4 Vrijednosti funkcije dobrote najbolje jedinke

|  |  |  |
| --- | --- | --- |
| Von Neumann | Ring | Global |
| 243 | 243 | 225 |
| 243 | 243 | 235 |
| 243 | 243 | 227 |
| 235 | 243 | 235 |
| 243 | 239 | 235 |
| 235 | 243 | 233 |
| 239 | 243 | 233 |
| 243 | 239 | 235 |
| 239 | 239 | 235 |
| 243 | 243 | 231 |
| 235 | 243 | 227 |
| 243 | 243 | 231 |
| 243 | 243 | 235 |
| 239 | 243 | 235 |
| 243 | 243 | 235 |
| 243 | 243 | 239 |
| 243 | 243 | 235 |
| 235 | 243 | 235 |
| 239 | 239 | 233 |
| 235 | 243 | 231 |
| 243 | 243 | 243 |
| 239 | 243 | 235 |
| 239 | 239 | 235 |
| 239 | 243 | 229 |
| 243 | 243 | 229 |
| 243 | 243 | 235 |
| 231 | 239 | 235 |
| 243 | 243 | 235 |
| 243 | 239 | 239 |
| 243 | 243 | 227 |
|  |  |  |
| 17 | 23 | 1 |

# Zaključak

Algoritam roja čestica izmijenjen je i primijenjen na igru sudoku. Za razliku od običnog algoritma roja čestica, kod geometrijskog nema brzine, dodana je mutacija i ne računa se pozicija nove čestice, već se nad česticom vrši rekombinacija (koja je zapravo konveksna kombinacija, njome se pronalazi točka u konveksnoj ljusci koju čine tri roditelja i ta točka je rezultat rekombinacije). Na temelju testiranja može se zaključiti kako je bolje koristiti lokalne topologije, pogotovo topologiju von Neumann koja daje najbolje rezultate. Najjednostavnija (globalna) topologija nije toliko poželjna, jer je veća vjerojatnost ostanka u lokalnom optimumu i ne pronalaženja globalnog rješenja.

# Literatura

1. Moraglio A., Togelius J., Geometric Particle Swarm Optimization for the Sudoku Puzzle, U niversity of Essex, UK, 2007.
2. Perez M., Marwala T., Stochastic optimization approaches for solving sudoku, University of the Witwatersrand, Johannesburg, South Africa
3. McGerty S., Solving Sudoku Puzzles with Particle Swarm Optimisation, Macquarie University, Sydney, Australia
4. Čupić M., Prirodom inspirirani optimizacijski algoritmi, Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva, Zagreb, 2009.
5. Mendes R., Kennedy J., Neves J., Avoiding the pitfalls of local optima: how topologies can save the day
6. Lynce I., Ouaknine J., Sudoku as a SAT Problem, Technical University of Lisbon, Portugal
7. Hamdan, S. A., Hybrid Particle Swarm Optimiser using multi – neighborhood topologies, Arab Open University, Department of Computing and Information Technology, Kuwait
8. PSO: Neighborhood Topologies, http://tracer.uc3m.es/tws/pso/neighborhood.html

# Sažetak

U ovom seminarskom radu opisan je geometrijski algoritam roja čestica (GPSO). Opisana je primjena tog algoritma na rješavanje igre sudoku i implementacija. Izvršena su testiranja nad trima različitim topologijama i ustanovljeno je da je algoritam s topologijom von Neumann najuspješniji (u većini slučajeva dostiže se najveća moguća vrijednost funkcije dobrote), nakon toga slijedi kružna topologija (K = 2), a najlošija je globalna (zvjezdasta topologija).