SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

SEMINAR

Evolucijski algoritmi:
Optimizacija rojem čestica

Marko Pletikosa

Voditelj: *doc.dr.sc. Domagoj Jakobović*

Zagreb, svibanj, 2009.

Sadržaj

1. Uvod

Evolucijski algoritmi (*eng. Evolutionary algorithms, EA*) su područje unutar grane računarske znanosti koju nazivamo umjetnom inteligencijom *(eng. Artificial Intelligence, AI)*. Glavno svojstvo evolucijskog algoritma je poboljšavanje performansi kroz iteracije i učenje skrivenih struktur unutar problema koje vode k rješenju. Valja istaknuti da evolucijski algoritam nema konkretno znanje o problemu.

EA su stohastički algoritmi, koji ne jamče pronalaženje apsolutnog optimuma. Najčešće, uz dovoljno računalnih resursa, pronalaze rješenje koje zadovoljava praktične primjene.

Uglavnom su inspirirani prirodom. Uzmimo za primjer genetske algoritme (GA). Oni za uzor uzimaju biološku evoluciju. Za zadani prikaz jedinki postoje operatori križanja i mutacije. Svaka jedinka predstavlja potencijalo rješenje problema. U pojedinoj iteraciji jedinke se međusobno križaju na način da bolje jedinke imaju veće šanse za križanje i tako predaju svoj povoljan genetski materijal budućim generacijama. Uz to se događa i mutacija na pojedinoj jedinki – kromosomu. Mutacije unose slučajnost i nedeterminizam u populaciju i omogućavaju pronalaženje i drugih rješenja uz ona već pronađena.

Optimizacija rojem čestica *(eng. Particle Swarm Optimization – PSO)* je još jedan od evolucijskih algoritama. PSO je jako sličan evolucijskim metodama poput genetskih algoritama (GA). Kao i kod GA, rješenje problema predstavlja svaka pojedina čestica (jedinka). Čestice se kreću u višedimenzionalnom prostoru problema i traže najbolje rješenje. Za razliku od GA, ovdje nema genetskih operatora mutacije i križanja. Umjesto toga koristi se socijalna mreža susjedstva. Socijalna topologija može biti vrlo različita i tako postoje različite varijacije PSO algoritma. U najčešćem slučaju ponašanje promatrane jedinke ovisi o komunikaciji sa susjedima, tj. njihovom uspjehu, i uspjehu najbolje jedinke u populaciji.

Cilj ovog seminara je dati kratki pregled PSO algoritma i njegove analogije s društvenim mrežama pojedinaca. Sastavni dio seminara je i izvorni kod implementacije tvz. kanonske verzije algoritma. Prije toga se valja upoznati s nekim osnovnim ciljevima umjetne inteligencije.

2. Umjetna inteligencija

AI je grana koja se počela razvijati sredinom prošlog stoljeća. Kao godina utemeljenja se navodi 1956., na konferenciji održanoj u Darmouthu u Sjedinjenim Američkim Državama [1]. Umjetna inteligencija je inspirirana prirodom. Četiri su aspekta u promatralju definicije umjetne inteligencije:

1. Razmišljati ljudski,
2. ponašati se ljudski,
3. razmišljati racionalno,
4. ponašati se racionalno.

Ona je i multidisciplinarno područje koje se može promatrati iz gledišta:

1. Filozofije:
    - Mogu li se formalna pravila iskoristiti kako bi se dobili valjani zaključci?
    - Kako se psihička svijest uzdiže iz fizičkog uma?
    - Odakle dolazi znanje?
    - Kako znanje vodi k akciji?
2. Matematike:
    - Koja su formalna pravila potrebna da se dobiju valjani zaključci?
    - Što se može izračunati?
    - Kako se mi ponašamo kad nemamo pouzdane informacije?
3. Ekonomije:
    - Kakve odluke donositi kako bi maksimizirali dobit?
    - Što učiniti kad se drugi ne slažu s nama?
    - Što učiniti kad je dobit daleko u budućnosti?
4. Neuroznanosti:
    - Kako mozak obrađuje informacije?
5. Psihologije:
    - Kako razmišljaju ljudi i životinje?

Uz navedene aspekte, tu su svakako i računalno inženjerstvo (Kako izgraditi učinkovito računalo?), Teorija kontrole i kibernetika (Kako mogu i mogu li ikako umjetno stvoreni entiteti samostalno djelovati?) i Lingvistike (Kako jezik utječe na misao?).

3. Optimizacija rojem čestica

Optimizacija rojem čestica (PSO) je stohastička metoda optimizacije temeljena na populaciji jedinki. Izvorni algoritam su razvili Dr.Eberhart i Dr. Kennedy 1995. godine inspirirani načinima interakcije između ptica, jata riba i ljudi.

PSO je predstavnik takozvane inteligencije roja *(eng. Swarm intelligence)* koja prioizilazi iz interakcija više jednostavnih procesnih jedinica. Sama riječ roj ukazuje na količinu jedinica, slučajnost i neurednost, dok inteligencija sugerira da se iz ponašanja tih jedinica može riješiti problem. Te jedinke u kontekstu PSO algoritma nazivamo česticama. Čestice možemo zamišljati kao ptice, ribe, kukce, ljude, programske agente, ili bilo kakav entitet koji je sposoban komunicirati.

Ljudi uče o svijetu oko sebe istraživajući i dijeljeći svoje znanje s drugima. Na sličan način i čestice napreduju: Svaka čestica predlaže svoje rješenje problema, diskutira ga s ostalima, prihvaća poneke sugestije i onda nastavlja istraživati u smjeru koji se čini ispravnim. Čak i kad je problem jako složen (kad je cilj višeobjektivan, problem multimodalan – ima više lokalnih optimuma, ali samo jedan je globalno optimalan, nelinearan ili je omjer snage signala naspram šuma malen - SNR), čestice na iznenađujuće brz način pronalaze rješenje.

3.1 Karakteriskike algoritama

Svaki PSO algoritam ima početnu populaciju čestica. Populacije su redovito puno manje nego kod genetskih algoritama i najčešće variraju od 20 do 50 čestica. Isti algoritam se primjenjuje na različite probleme, kao način inicijalizacije parametara čestica se nameće odabir slučajnih vrijednosti unutar dozvoljenog skupa stanja.

Glavne razlike između različitih inačica algoritma su u primjenjenoj socijalnoj topologiji. Tradicionalne topologije su *lbest (eng. Locally best)* i *gbest (eng. Globally best).* U gbest topologiji sve su čestice povezane są svim ostalim česticama. U praksi to znači da na svaku česticu djeluje čestica koja je do sada uspjela pronaći najbolje rješenje. Iako gbest topologija ima najveći broj veza između čestica, ona znači samo praćenje trenutno najbolje čestice (pohlepan način – *eng. greedy*). Lbest topologija je prstenasta rešetka u kojoj je svaka čestica povezana s česticama koje se nalaze lijevo ili desno od nje u polju čestica. Prednosti ovakve strukture su u tome što različite skupine čestica mogu konvergirati prema različitim rješenjima i time izbjeći zamke lokalnih optimuma. Cijena za povećanu sigurnost u pronalaženju boljih rješenja je sporija konvergencija.

Uz populaciju i topologiju, nužno je odrediti pravilo po kojem čestice mijenjaju svoje privatne vrijednosti. Vrijeme je diskretizirano i promatra se kroz iteracije algoritma. U ovisnosti o primjenjenoj topologiji, na promatranu česticu u pojedinoj iteraciji utječe određeni skup populacije. U ovisnosti o tim vezama, trenutno najbolje poznatoj poziciji i komunikaciji između čestica, promatrana čestica odabire smjer u kojem će se kretati do slijedeće iteracije i brzinu kojom će ići.

Pretraživanje se odvija u dvije faze koje zovemo istraživanje i iskorištavanje. Početno čestice traže dobru regiju unutar prostora rješenja i tu fazu nazivamo

istraživanjem. Nakon pronalaska dobre regije, čestice traže najbolju točku unutar regije. Fazu kad su susjedi unutar iste regije i traže optimum unutar nje, nazivamo iskorištavanjem.

Ukoliko čestica i njeni susjedi imaju uspjeha u regiji, nastavit će je pretraživati. Međutim, ukoliko neki od susjeda pretraživaju druge regije, čestice će pretraživati prostor u velikom radijusu tj. u širinu i tako se vraćati fazi istraživanja. Sustav je fleksibilan te dopušta da susjedi simultano pretražuju različite regije i jednostavno se prebacuju iz regije u regiju i izbjegavaju lokalne optimume. Putanje (trajektorije) čestica su oscilatorne i možemo reći da jedinka istražuje kadje amplituda oscilacije velika, a iskorištava kad je amplituda mala, kao i kod njenih susjeda.

Čestica se evaluira funkcijom koju nazivamo funkcijom dobrote *(eng. Fitness function)*. Ta funkcija procjenjuje uspješnost pojedine čestice i na temelju nje se čestice rangiraju – određuju lokalno i globalno najbolje koje utječu na ostale. Ona je različita od problema do problema. Da bi primjenili PSO na različite probleme, potrebno je samo promijeniti funkciju dobrote.

3.2 Kanonski algoritam

Svaka čestica je opisana svojim varijablama stanja:

- Položaj u prostoru **x[]**
- Brzinom kretanja **v[]**
- Listom susjeda
- Do sada najboljim položajem **p[]**

U kanonskom algoritmu, na svaku česticu utječe najuspješniji susjed iz liste susjedstva. Susjedstvo može biti cijela populacija ili njen dio. Osnovne formule po kojima se mijenja stanje čestice *i* u dimenziji *j* su slijedeće [4]:

1. v[i][j] = chi × (v[i][j] + rand1()* phi1 × (p[i][j] – x[i][j]) + rand2()× phi2 × (p[g][j] – x[i][j])) (1)
2. x[i][j] = x[i][j] + v[i][j] (2)

U danim formulama, chi, phi1 i phi2 su korisnički definirane konstante, koje najčešće iznose 0.792 za chi, a phi1 i phi2 u zbroju daju 4.1. Zašto su konstante tako odabrane? U literaturi je ponuđen jednostavan odgovor: "Zato što tako sustav funkcionira". Međutim, treba biti oprezan jer za krivi odabir konstanti sustav eskplodira u smislu da brzine čestica postanu prevelike te se čestice često nađu na rubovima prostora pretraživanja. Početno se **x** inicijalizira na slučajne vrijednosti unutar **Xmin** i **Xmax** granica, a **v** unutar -**Vmax** i **Vmax**. Funkcija rand() vraća slučajnu vrijednost između 0.0 i 1.0.

Prema danim formulama je vidljivo da čestica oscilira između vlastite prethodne najbolje vrijednosti i najbolje vrijednosti susjedstva iz prethodnog koraka.

Formula se primjenjuje u svakoj iteraciji. Vidljivo je da, čim se čestica udalji od opisane sredine, njena brzina udaljavanja se sve više smanjuje te se ona počinje vraćati.

Međutim, ponašanje nije periodičko iz više razloga. Prvi od njih je što je težina dodjeljena najboljim vrijednostima stohastička. Povratne vrijednosti rand() funkcija variraju između 0.0 i 1.0. Stoga je moguće da neki od faktora uopće ne igraju ulogu u određivanju smjera i brzine za sljedeći korak. Amplitudu određuju chi, phi1 i phi2 pomnoženi s slučajnim brojevima.

Ipak, to nije najvažniji razlog aperiodičnosti ponašanja čestica. Moguće je da je čestica u trenutnom koraku pronašla točku u kojoj postiže najbolju funkciju dobrote pa će se prema njoj orijentirati. Uz to je moguće da je netko iz susjedstva pronašao novu najbolju lokaciji pa će se index "g" promijeniti, ili pak da će čestica koja je i prije imala najbolju vrijednost u susjedstvu pronaći još bolji položaj pa će se p[g][j] promijeniti, iako se "g" nije promijenio.

Rezultat je aperiodična i nepredvidljiva trajektorija čestica koju je teško shvatiti i predstavlja srž neuredne, složene i evoluirajuće inteligencije roja.

Amplituda titranja je, kao što je već rečeno, slučajna, ali uvelike ovisi o razlici između p[i][j] i p[g][j]. Ta razlika se mijenja iz iteracije u iteraciju. Možemo reći da je položaj čestice u sljedećem koraku određen hiperpravokutnikom definiranim razlikama između p[i][j] i p[g][j] za svaki j. Opseg pretrage je omeđen razlikom između čestičinih prijašnjih najboljih točaka (vlastite i one najboljeg susjeda). Ovu činjenicu nazivamo konsenzusom. Kad su te točke dovoljno blizu, čestice se "dogovore" nastaviti fazu iskorištavanja (eksploatacije) na tom području. Stupanj konsenzusa definira okvire pretrage. Kao funkciju dobrote (uspješnosti) svake čestice, jednostavno uzmemo funkciju kojoj tražimo ekstrem.

Možemo reći da čestice opisuju oscilatorne putanje koje ih pomiču naprijed i natrag oko stohastičke sredine vlastite najbolje točke i najbolje točke koju su susjedi pronašli.

3.3 Socijalna analogija

AI je usko povezana s mnogim znanostima, kao što je navedeno u drugom poglavlju. PSO je povezana s kognitivnom psihologijom. Mnoge studije pokazuju da su pojedinčeva uvjerenja, stavovi, sjećanja i procesi razmišljanja pod velikim utjecajem okoline. Ljudi često nisu svjesni vlastitog kognitivnog procesa i ne mogu verbalno opisati tok misli koji je doveo do određenog zaključka: Mnogi samostalno objašnjeni procesi se mogu bolje opisati kao racionalizacije, a ne kao opisi. Pošto su početne faze AI-a uvelike temeljene na takvim opisima jedinki, jasno je da je ostalo dovoljno prostora za napredak naspram tradicionalnih heurističkih algoritama.

Jedan od osnovnih principa na kojima se temelji algoritam roja čestica je pogled na kognitivni proces kao socijalni proces. Interni monolog misli se vrlo lako može promatrati kao zamišljeni razgovor mislioca i nekog drugog – druge osobe, promatrača, ili promatrajući sebe istovremeno kao govornika i slušača. U bilo kojem slučaju, istraživanja su utvrdila da je kognitivni proces izvan škole ili laboratorija gotovo uvijek temeljen na suradnji pojedinaca. Prema tim istraživanjima, zajedničko

poimanje problema i konteksta je nužno za koordinirani kognitivni proces. Predloženo je i novo područje – sociokognicija. Jedna od poznatijih rečenica utemeljitelja je: "Iako neki tvrde da mozak kao fizička lokacija mentalnog procesiranja zahtjeva da kognitivne procese smatramo inidividualnim pa čak i privatnim aktivnostima – od perceptivnog opažanja i memorije do rješavanja problema – ono uključuje ili reprezentacije drugih ljudi ili uporabu kulturnih tvorevina i norma koje imaju socijalnu prošlost" (J. M. Levine, L. B. Resnick, E. T. Higgins).

Ljudska inteligencija kao takva djeluje kroz socijalnu interakciju među pojedincima. Naše svjesno iskustvo razmišljanja nije dostatno znanstveno objašnjenje kognicije. Čini se da bi se bolji opis manje usmjerio na individualna i privatna razmišljanja, a više na interakciju među ljudima.

Teorija kognitivne disonance tvrdi da je um skup kognitivnih elemenata koji su povezani složenim vezama. Osoba pokušava, na bilo koji način, smanjiti nesklad među elementima, npr. kad je potrebno istovremeno razmišljati o kontradiktornim logičkim pravilima, kad se osoba slaže s mišljenjima druge osobe koja joj nije draga i sl. Možemo reći da je zadovoljstvo rezultat složene funkcije velikog broja kognitivnih elemenata. U analogiji s PSO algoritmom vidimo da je zadovoljstvo par funkciji dobrote, a sklad kognitivnih elemenata problem pronalaženja najbolje ravnoteže tj. Najbolje točke u prostoru. Način optimizacije ovakvog problema je često kroz interakciju s drugim ljudima, poimanju njihovih shvaćanja i pogleda na problem.

Vidimo da PSO objedunjuje inženjerske probleme i sociokognitivne procese kako bi pronašli bolje načine modeliranja i rješavanja problema iz oba područja.

3.4 Usporedba s genetskim algoritmima

Većina evolucijskih tehnika ima slijedeće značajke:

1. Slučajno generiranu početnu populaciju,
2. funkciju dobrote *(eng. Fitness function)* koja ovisi o udaljenosti od traženog optimuma,
3. reprodukciju temeljenu na vrijednosti funkcije dobrote,
4. uvjet zaustavljanja: broj iteracija, srednju pogrešku.

Kao i GA, PSO ima slučajnu početnu populaciju, funkciju dobrote, te obje tehnike napreduju na temelju slučajnih vrijednosti te ne jamči uspjeh. Glavna razlika između GA i PSO je u selekciji. GA mijenjaju populaciju iz koraka u korak, eleminirajući jedinke s lošom dobrotom, a one bolje se međusobno križaju i/ili mutiraju – svaka populacija u iteracji se smatra drugom generacijom u vremenu. Za razliku od njih, PSO i ostale tehnike temeljene na socijalnoj interakciji, zadržavaju jedinke i one se poboljšavaju međusobnom komunikacijom iz iteracije u iteraciju.

Temeljni princip evolucije je borba za preživljavanje. Jedinke koje prežive imaju šanse za reprodukciju i/ili mutaciju i prenošenje svojih svojstava na buduće generacije. U takvim uvjetima populacija najčešće napreduje.

Glavni princip algoritma roja čestica je suradnja i dijeljenje znanja. Svaka čestica je i učitelj i učenik. S vremenom ukupna se učinkovitost povećava, a čestice koje međusobno komuniciraju teže istom području u prostoru problema. Bolje čestice uspjevaju utjecati na one manje uspješne.

3.5 PSO i Umjetne neuronske mreže

Umjetna neuronska mreža *(NM ili eng. Artificial neural networks – ANN)* je paradigma koja za uzor uzima funkcioniranje ljudskog mozga, tj. njegovih sastavnih jedinica – neurona. Najpopularniji algoritam za treniranje NM-a je tzv. Algoritam s propagacijom greške unatrag *(eng. Back-propagation algorithm).* Istraživanja provedena proteklih godina pokazuju veliku uspješnost primjene evolucijskih tehnika u unapređenju paradigme.

Nekoliko je glavnih aspekta NM-a: podešavanje težina poveznica između neurona, tzv. sinapsi, rasporeda neurona – topologije mreže, prijenosna funkcija i algoritam učenja.

Većina primjena EA na NM uključuje podešavanje težina sinapsi i topologije mreže. U tu svrhu se jako često koriste genetski algoritmi (GA). U problemima klasifikacije, postotak pogrešno klasificiranih objekata može poslužiti kao funkcija dobrote. EA uspješno rješavaju problem kad je funkcija prijenosa nediferencijabilna tj. kad nema nikakvih informacija o gradijentu. Nedostatci su slabije performanse kod nekih problema i vrlo složen odabir operatora mutacije i križanja u slučaju GA.

PSO algoritam se pokazao kao obećavajuća zamjena algoritma propagacije greške unatrag. Radovi pokazuju da je moguće PSO podesiti da bude brži i učinkovitiji od GA u velikom broju slučajeva.

3.6 Binarni problemi

Genetski algoritmi rješavaju probleme pomoću binarno kodirane jedinke. Binarno kodiranje je korisno pri rješavanju problema čija su rješenja diskretna. Da bi PSO uspješno rješavao ove probleme, potrebno je neznatno modificirati algoritam. Brzina se računa kao u kanonskom algoritmu, dok je promjena u računanju položaja čestice. U svrhu računanja položaja koristi se tzv. sigmoidalna funckija. Njena formula glasi:

$$S(v) = 1/ (1+ \exp(-v)).$$

Iz same formule vidimo da je njena kodomena skup realnih brojeva između 0.0 i 1.0. Vrijednost položaja čestice u danoj dimenziji postavljamo na 1 ako je vrijednost slučajno generiranog broja manja od vrijednosti sigmoidalne funkcije za argument brzine čestice u toj dimenziji. Inače 0. Možemo napisati:

1. `v[i][j] = chi × (v[i][ j] + rand1()* phi1 × (p[i][j] – x[i][j]) + rand2()× phi2 × (p[g][j] – x[i][j]))` (1)
2. `if (rand() < S(v[i][j])) x[i][j] = 1; else x[i][j] = 0;`

4. Opis implementacije

S ciljem boljeg shvaćanja rada algoritma i testiranja njegove uspješnosti pri rješavanju jednostavnih problema, u ovom seminaru sam implementirao kanonsku verziju algoritma. Pseudokod glasi:

```
Xmax <- prostor pretraživanja
Vmax <- maksimalna brzina čestice
eval() <- funkcija dobrote specifična za dani problem
chi = 0.792
phi1, phi2 <- korisničke konstante
Inicijaliziraj:
     za i = 1 do broja čestica
          za j = 1 do dimenzije prostora
               x[i][j] = jednolika razdioba u [-Xmax, Xmax]
               v[i][j] = jednolika razdioba u [-Vmax, Vmax]
               p[i][j] = x[i][j] // najbolji položaj
     nDobrota = eval(p[i]); //najbolja vrijednost dobrote
     ako (rand() < 0.5) tada listaSusjeda = čestice s indeksima
                                            između (1, i);
     inače listaSusjeda = čestice s indeksima (i, brojČestica);

Iteriraj:
     za svaki i = 1 do broja čestica
          sn = indeks susjeda s najboljom nDobrota
          za j = 1 do dimenzije prostora
v[i][j]=chi×(v[i][j] + rand1()*phi1×(p[i][j]−x[i][j])+
rand2()×phi2×(p[sn][j]−x[i][j]));
               ako (v[i][j] > Vmax ili v[i][j] < -Vmax)
                    v[i][j] = Vmax tj. v[i][j] = -Vmax

               x[i][j] = x[i][j] + v[i][j];
               ako (x[i][j] > Xmax ili v[i][j] < -Xmax)
                    v[i][j] = Xmax tj. v[i][j] = -Xmax
```

```
            trenutnaDobrota = eval(x[i]);
            ako (trenutnaDobrota < nDobrota)
                 nDobrota = trenutnaDobrota;
                 za j = 1 do dimenzija prostora
                      p[i][j] = x[i][j];
dok nije zadovoljen uvjet zaustavljanja.
```

Kao što je već napomenuto, za rad algoritma je vrlo važna topologija susjedstva. U ovom primjeru korištena je tzv. *lbest* topologija. Na česticu utječe:

1. Njen položaj,
2. njen najbolji poznati položaj i
3. najbolji poznati položaj najuspješnijeg susjeda (susjed s indeksom **sn** u pseudokodu).

Ako čestice promatramo kao jednodimenzionalno polje čiji su indeksi od 1 do **n**, za promatranu česticu **i** slučajnim odabirom za susjede te čestice odabiremo one lijevo od nje (indeksi od 1 do **i**), ili one desno od nje (indeksi od **i** do **n**), naravno, pazeći na rubne uvjete (svaka čestica bi trebala imati barem jednog susjeda). Odabrana je upravo *lbest* topologija jer, iako u pravilu, uz više iteracija, daje bolje rezultate od *gbest* topologije i otpornija je na zamke lokalnih optimuma.

Rezultati testiranja na jednostavnim funkcijama su ohrabrujući (optimum je **0**):

f(**x**)= π(x -a), u 100 iteracija i 40 jedinki:
        i i

Najmanje pronađene vrijednosti funkcije su reda veličine 1e-6, prosjek vrijednosti čestica iznosi 1e-3. Potrebno je oko 55 iteracija da pogreška najbolje čestice padne ispod 1e-3.

Ukoliko testiramo s složenijim funkcijama, poput:

$$f6 = 0.5 - \frac{\sin^2\sqrt{\sum x_i^2} - 0.5}{[1.0 + 0.001 \cdot \sum x_i^2]^2}$$

Rezultati su vrlo dobri: Roj od 40 čestica za 100 iteracija i dimenziju prostora 5 pronađe minimum oko točke reda veličine 1e-3, uz srednju vrijednost čestica oko 1e-2. Navedena funkcija je primjer funkcije koja jako brzo oscilira i za koju je teško pronaći točan minimum evolucijskim algoritmima.

S ciljem što veće dostupnosti algoritma na različitim platformama, implementacija je napisana u programskom jeziku Java. Da bi se pokušao riješiti novi problem, potrebno je samo definirati funkciju dobrote koju jednostavno napišemo koristeći definirano sučelje *(eng. Interface).* Planiran je daljnji rad na programu, uz testiranje različitih topologija i parametara, te vizualizaciju rješavanja problema korištenjem *appleta.*

5. Zaključak

Iako je PSO relativno novo područje, staro tek 15ak godina, pokazalo se učinkovitim u rješavanju određenih problema. Primjenjuje se na različite klase problema, koje uključuju: binarne probleme, multiobjektne probleme, dinamične probleme, održavanje električnih mreža ili rano dijagnosticiranje bolesti. PSO se brzo razvija i često se pojavljuju novije i poboljšane verzije algoritama.

Posljednjih nekoliko godina, algoritam se ne koristi samo kao alat za rješavanje inženjerskih problema korištenjem socijalnih uzoraka, već se pomoću njega istraživaju i simuliraju novi modeli socijalnih ponašanja.

6. Literatura

[1] Stuart russell and Peter Norwig: Artificial intelligence: A modern approach 2nd edition, Prentice Hall.

[2] Xiaohui Hu, Ph.D., 15. travnja 2009., http://www.swarmintelligence.org/

[3] Igor Kononenko i Matjaž Kukar: Machine learning and data mining: Introduction to principles and algorithms, Horwood Publishing, ISBN – 10: 1-904275-21-4, ISBN-13: 978-1-904275-21-3, 2007.g.

[4] James Kennedy: Swarm Intelligence, Springer US, ISBN 978-0-387-40532-2 (Print) 978-0-387-27705-9 (Online)
http://www.springerlink.com/content/q528q1743224q233/

7. Sažetak

Algoritam optimizacije rojem čestica možemo svrstati u evolucijske algoritme umjetne inteligencije. Temelji se na populaciji čestica koje možemo zamisliti kao ptice. Svaka čestica predstavlja jedno rješenje problema. Čestice "lete" višedimenzionalnim prostorom problema i komunicirajući sa susjedima i izmjenjujući iskustva usmjeravaju se prema boljim rješenjima.

Ključan aspekt je primjenjena socijalna topologija. Ona određuje što i tko sve utječe na ponašanje čestice u sljedećem koraku. Najčešće, na česticu utječe trenutni položaj u prostoru, skup njenih susjeda s kojima komunicira te čestica koja predstavlja trenutni optimum. Ti parametri pomažu u određivanju novog smjera kretanja čestice. Uz smjer, i brzinu kretanja ovisi o spomenutim parametrima. Iako se opisani način prvotno čini jednostavan, iznimno je složen i nepredvidljiv kad se pogledaju trajektorije čestica, ali je istovremeno učinkovit. Unutar sebe krije modele ponašanja jedinki kakvi se mogu naći i među ljudskom populacijom.