Optimizacija rojem čestica

Seminarski rad u okviru kursa
Metodologija stručnog i naučnog rada
Matematički fakultet

Nevena Soldat, Milena Kurtić, Tijana Živković, Ana Miloradović
nevenasoldat@gmail.com, mimikurtic67@gmail.com,
tijanazivkovic6@gmail.com, ana.miloradovic7@gmail.com

17. mart 2020.

Sažetak

Kennedy i Eberhart (2001):
"... gledamo u paradigmu koja je u svom začeću, puna potencijala i novih ideja i novih perspektiva... Istraživači u mnogim zemljama eksperimentišu sa rojevima čestica... Mnoga pitanja koja su postavljena još uvek nisu dobila dobar odgovor."

U ovom radu opisaćemo osnovni algoritam za optimizaciju rojem čestica, kao i neke od postojećih varijacija. Objasnićemo jednostavnije algoritme sa jedinstvenim rešenjem, ali i neke naprednije. Tema će biti i socijalne strukture na kojima se on zasniva, kao i koje su moguće primene.

Sadržaj

1 Uvod

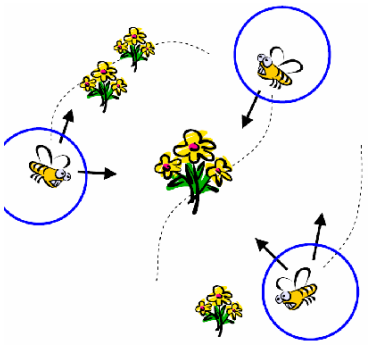
Inteligencija rojeva predstavlja jednu od pet glavnih paradigmi Računarske Inteligencije (Computation Intelligence - CI). Jedinke u okviru grupe (roja) dele prikupljene informacije u zajedničkom cilju da reše neki problem, koje se propagiraju kroz celu grupu tako da se problem rešava mnogo efikasnije nego što bi to mogla pojedinačna jedinka. Prvi i dosta značajan doprinos u polju inteligencije rojeva imao je južnoafrički pesnik Eugene N Marais koji je proučavao socijalno ponašanje kako majmuna, tako i mrava. Posle njega, ranih 1990-ih godina, Marco Dorigo modeluje ponašanje kolonija mrava. Zatim, 1995, Eberhart i Kennedy razvijaju algoritam optimizacije rojem čestica, na osnovu posmatranog jata ptica. Optimizacija rojem čestica (eng. Particle Swarm Optimization - PSO) je stohastička optmizaciona tehnika zasnovana na veoma inteligentnom kolektivnom ponašanju nekih organizama kao što su insekti, ptice i ribe. Algoritam optimizacije rojem čestica otkriven je sasvim slučajno od strane Eberharta i Kenedija, pri pokušaju da se na računaru simulira kretanje jata ptica. Prvobitna namera bila je da se grafički prikaže nepredvidiva koreografija jata ptica, sa ciljem da se otkriju obrasci koji omogućavaju pticama da lete sinhronizovano, i da zadrže optimalnu formaciju pri naglim promenama pravaca. Sada je cilj kreiranje jednostavnog i efikasnog optimizacionog algoritma. Od kada je prvi put predstavljen 1995. doživeo je niz poboljšanja i nastale su brojne varijacije ovog algoritma.

2 Algoritam za optimizaciju rojem čestica

Kako bismo lakše razumeli algoritam možemo zamisliti roj pčela koje lete preko polja sa cvećem. Roj ima urođenu želju da pronađe poziciju gde je cveće najgušće raspoređeno. Takođe, pčele nemaju nikakvo znanje o polju na kom se nalaze. Tako počinju svoju pretragu u različitim smerovima. Svaka pčela pamti mesta na kojima je bila i na kojim je bilo najviše cveća, i tu informaciju može preneti komunikacijom sa ostatkom roja. Kako vreme prolazi, pčele biraju da li će se vratiti na svoje prethodno pronađene najbolje lokacije ili će ići ka lokacijama koje su dobile od ostalih pčela. One koje oklevaju će ići u oba pravca i nalaziće se negde između ciljanih lokacija, u zavisnosti od toga da li će socijalan uticaj biti dominantan ili ne. Povremeno, pčela može da preleti preko dela polja u kom se nalazi više cveća od do sad otkrivenih lokacija. Tada će se čitav roj povlačiti ka novootkrivenoj lokaciji.

Na slici 1, isprekidane linije predstavljaju zamišljene putanje pčela, a strelice prikazuju dve komponente brzine (lokalno najbolju poziciju i globalno najbolju poziciju). Pčela u gornjem delu slike je pronašla globalno najbolju poziciju, dok je pčela sa leve strane pronašla lokalno najbolju poziciju. Pčela u donjem delu slike prikazuje da iako nije pronašla lokalno najbolju poziciju, ide ka globalno najboljoj poziciji.

Na ovaj način pčele pretražuju polje tako što menjaju brzine i pravac kretanja u zavisnosti od toga da li je imala uspeha da pronađe cveće u odnosu na čitav roj. Takođe pčele znaju da izbegavaju mesta koja nisu imala puno cveća. Na kraju će pretražiti celo polje, i nalaziće se iznad mesta na kojem je najveća gustina cveća.



Slika 1: Prikaz PSO algoritma na kojem pčele traže cveće

2.1 Originalni PSO

Algoritam za optimizaciju rojem čestica sadrži jedinke (čestice) koje se se kreću kroz višedimenzioni prostor pretrage, a pozicije jedinki se menjaju u skladu sa sopstvenim iskustvom, kao i sa iskustvom susednih jedinki. Svaka od tih čestica predstavlja jedno moguće rešenje. Neka je $x_i(t)$ pozicija čestice i u prostoru pretrage u trenutku t. Pozicija čestice se menja dodavanjem brzine, $v_i(t)$ na trenutnu poziciju.

$$x_i(t+1) = x_i(t) + v_i(t+1)$$

Njihovo kretanje se usmerava imajući u vidu njihovu trenutnu poziciju, njihovu do sada najbolju poziciju, kao i do sada najbolju poziciju čitavog roja. Kognitivna komponenta algoritma predstavlja tendenciju vraćanja u lično najbolje rešenje, dok socijalna komponenta predstavlja tendenciju ka globalno najboljem rešenju. Brzina se računa kao:

$$v_i(t+1) = v_i(t) + c_1 r_1(p_i(t) - x_i(t)) + c_2 r_2(p_g(t) - x_i(t))$$

gde $p_i(t)$ predstavlja najbolju do sada poziciju čestice i u trenutku t, dok je $p_g(t)$ globalno najbolje rešenje (pozicija) do trenutka t. Parametri r_1 i r_2 su nasumične vrednosti izabrane iz uniformne raspodele na intervalu [0,1], dok su parametri c_1 i c_2 konstante koje predstavljaju pozitivna ubrzanja čija je uloga da skaliraju značaj kognitivne, odnosno socijalne komponente brzine.

2.2 Komponente brzine PSO algoritma

2.3 Globalno najbolji PSO

U slučaju globalno najboljeg PSO algoritma, susedi za svaku česticu su zapravo čitav roj. On implementira topologiju zvezde. U topologiji zvezde socijalna komponenta brzine čestice predstavlja informaciju dobijenu od svih čestica u roju. Brzina čestice i se računa kao:

$$v_{ij}(t+1) = v_{ij}(t) + c_1 r_{1j}(t)[y_{ij}(t) - x_{ij}(t)] + c_2 r_{2j}(t)[\hat{y}_j(t) - x_{ij}(t)]$$

gde je $v_{ij}(t)$ brzina čestice i u dimenziji $j = 1,...,n_x$ u trenutku t, $x_{ij}(t)$ je pozicija čestice i u dimenziji j u trenutku t, c_1 i c_2 su konstante koje predstavljaju pozitivna ubrzanja čija je uloga da skaliraju značaj kognitivne, odnosno socijalne komponente brzine, i $r_{1j}(t)$, $r_{2j}(t)$ su nasumične

vrednosti izabrane iz uniformne raspodele na intervalu [0,1]. Lokalno najbolje rešenje, $\mathbf{y}_i$ je najbolja pozicija u kojoj je bila čestica i od početnog trenutka t. U trenutku $t+1$, lokalno najbolje rešenje se računa na sledeći način:

$$y_i(t+1) = \begin{cases} y_i(t) & f(x_i(t+1)) \geq f(y_i(t)) \\ x_i(t+1) & f(x_i(t+1)) < f(y_i(t)) \end{cases} \quad (1)$$

gde je $f : \mathbb{R}^{n_x} \to \mathbb{R}$ fitnes funkcija. Fitnes funkcija računa koliko je dobijeno rešenje blisko optimalnom, odnosno meri kvalitet rešenja. Globalno najbolje rešenje, odnosno pozicija, u trenutku t, je definisano kao

$$\hat{y}(t) \in y_0(t), ..., y_{n_s} | f(\hat{y}(t)) = min f(y_0(t)), ..., f(y_{n_s}(t))$$

gde je n_s ukupan broj čestica. Bitno je napomenutni da je $\hat{y}$ najbolje rešenje koje je pronađeno od strane bilo koje čestice do sada - obično se računa kao najbolje lokalno najbolje rešenje. Globalno najbolje rešenje se takođe može izabrati i od čestica posmatranog roja, u kom slučaju je

$$\hat{y}(t) = min f(x_0(t)), ..., f(x_{n_s}(t))$$

Globalno najbolji PSO je prikazan u narednom algoritmu

Algoritam *gbest* PSO:
Kreiraj i inicijalizuj n_s - dimenzioni roj;
ponavljaj
  za *svaku česticu* $i = 1, ..., n_s$ **uradi**
  // postavi lokalno najbolju poziciju
    ako $f(x_i) < f(y_i)$ **onda**
      $y_i = x_i$;
    kraj
  //postavi globalno najbolju poziciju
    ako $f(y_i) < f(\hat{y})$ **onda**
      $\hat{y} = y_i$;
    kraj
  kraj
  za *svaku česticu* $i = 1, ..., n_s$ **uradi**
    ažuriraj brzinu;
    ažiriraj poziciju;
  kraj
dok nije ispunjen zahtev za zaustavljanje;

2.4 Lokalno najbolji PSO

Lokalno najbolji PSO koristi topologiju prstena, gde svaka čestica i manji broj suseda. Socijalna komponenta predstavlja informaciju koju razmenjuju susedi čestice. Doprinos jedinke brzini je proporcionalna razdaljini između čestice i najbolje pozicije koju su pronašli susedi. Brzina se računa kao:

$$v_{ij}(t+1) = v_{ij}(t) + c_1 r_{1j}(t)[y_{ij}(t) - x_{ij}(t)] + c_2 r_{2j}(t)[\hat{y}_{ij}(t) - x_{ij}(t)]$$