

# Optimizacija maksimalne k-zadovoljivosti populacionim metahauristikama

Seminarski rad u okviru kursa  
Računarska inteligencija  
Matematički fakultet

David Dimić, Zorana Gajić  
daviddimic@hotmail.com, zokaaa\_gajich@bk.ru

Mart 2019.

## Sadržaj

<b>1</b>	<b>Uvod</b>	<b>2</b>
<b>2</b>	<b>Formulacija problema i reprezentacija rešenja</b>	<b>2</b>
<b>3</b>	<b>Evolucionni algoritmi (EA)</b>	<b>2</b>
3.1	Pseudokod EA	2
3.2	Inicijalizacija rešenja	3
3.3	Fitnes funkcija	3
3.4	Lokalna pretraga	3
3.5	Varijante EA algoritma	3
3.5.1	SAWEA	3
3.5.2	RFEA	3
3.5.3	FlipGA	3
3.5.4	ASAP	3
3.6	Rezultati	3
<b>4</b>	<b>Optimizacija rojem čestica (PSO)</b>	<b>3</b>
4.1	Pseudokod PSO	4
4.2	Inicijalizacija rešenja	5
4.3	Fitnes funkcija	5
4.4	Lokalna pretraga i flip heuristika	5
4.5	Varijante PSO algoritma	6
4.5.1	PSO-LS	6
4.5.2	PSOSAT	6
4.5.3	WPSOSAT	6
4.6	Rezultati	6
<b>5</b>	<b>Zaključak</b>	<b>6</b>

## 1 Uvod

## 2 Formulacija problema i reprezentacija rešenja

Data je formula  $F$  u KNF obliku sa  $n$  promenljivih  $(x_1, x_2, \dots, x_n)$  i  $m$  klauza. Problem maksimalne  $k$ -zadovoljivosti može se definisati na sledeći način:

Klauza  $C_i$  dužine  $k$  je disjunkcija  $k$  literala:  $C_i = (x_1 \vee x_2 \dots \vee x_k)$ , gde je svaki literal promenjiva ili njegova negacija i može se pojavljivati više puta u izrazu. Cilj je pronaći istinitosne vrednosti promenljivih, valuaciju koja je vektor  $\vec{v} = (x_1, x_2, \dots, x_n) \in \{0, 1\}^n$  tako da ova valuacija maksimizuje broj zadovoljenih klauza u formuli  $F$ .

Ako valuacija zadovoljava formulu, onda se ona naziva modelom formule  $F$ . Max  $k$ -SAT problem može se definisati parom  $(\Omega, SC)$ , gde je  $\Omega$  skup svih potencijalnih rešenja iz  $\{0, 1\}^n$ , vektor  $n$  promenljivih, a  $SC : \Omega \rightarrow \mathbb{N}$ , skor valuacije koji je jednak broju zadovoljenih klauza. Shodno ovome, problem max  $k$ -SAT je naći  $\vec{v} \in \Omega$  za koje je  $SC$  maksimalno:

$$\max_{\vec{v} \in \Omega} \{SC(\vec{v})\}$$

Očigledno ima  $2^n$  potencijalnih rešenja koji zadovoljavaju formulu  $F$ . Dokazano je da je problem max  $k$ -SAT je NP-kompletno za svako  $k > 2$ .

Za dobar algoritam je takođe važno dobro predstavljanje rešenja. Postoje više načina reprezentacija, ali je ovde odabran prirodna binarna reprezentacija. Svaka čestica je predstavljena binarnim nizom dužine  $n$ .

## 3 Evolucionarni algoritmi (EA)

Genetski algoritmi su bazirani na Darvinovoj teoriji evolucije, u kojoj unutar jedne populacije najčešće opstaju najbolje prilagođene jedinke. Nova generacija jedinki se dobija ukrštanjem jedinki iz prethodne generacije, pri čemu se s vremena na vreme kod nekih jedinki mogu pojaviti mutacije na nekim genima. Evolutivni algoritam ima populaciju koja se sastoji od jednog ili više pojedinaca. Pojedinaac, koji se naziva hromozom, sastoji se od gena. U našem slučaju gen predstavlja jedan bit. Hromozom predstavlja kodiranje našeg rešenja. Za svaku jedinku se definiše funkcija prilagođenosti. Nakon inicijalizacije početne populacije, nove generacije jedinki se dobijaju uzastopnom primenom genetskih operatora. Smena generacija se vrši sve dok ne bude ispunjen kriterijum zaustavljanja.

### 3.1 Pseudokod EA

U ovom poglavlju predstavimo osnovni oblik algoritma na kojem se dalje zasnivaju ostale varijante.

**Input** : Formula  $F$  u KNF-u,  $n$  i  $m$

**Output**: Najbolja procenjena valuacija i broj zadovoljenih klauza

inicijalizacija populacije: pozicije i brzine;

$t = 0$ ; // tekuća iteracija

**while** nije zadovoljen uslov zaustavljanja **do**

$t = t + 1$ ;

    Selekcija jedinki za primenu genetskih operatora;

    Ukrštanje za izabrane parove jedinki;

    Mutacija izabranih jedinki;

    Određivanje funkcije prilagođenosti populacije;

**end**

**Algorithm 1:** Osnovni Evolutivni algoritam

### 3.2 Inicijalizacija rešenja

Potrebno je da inicijalizujemo hromosome pseudoslučajnim brojevima  $\{0, 1\}$ . Veoma važnu ulogu igra odabir parametara. Parametri mogu biti fiksni ili im se vrednosti mogu na određen način menjati iz generacije u generaciju.

### 3.3 Fitnes funkcija

Tokom različitih varijanti EA algoritama koristile su se različite funkcije prilagođenosti. Prva fitness funkcija koja se sama nameće jeste broj zadovoljenih klauza, koju ćemo koristiti u FlipGA i ASAP algoritmima. Nešto malo komplikovanije funkcije prilagođenosti se koriste u SAWEA i RFEA, koje ćemo objasniti detaljnije kasnije.

### 3.4 Lokalna pretraga

### 3.5 Varijante EA algoritma

Da bismo dobili što bolje rešenje pokušavamo da kombinujemo različite tehnike operatora selekcije, ukrštanja, mutacije i ostale. U tabeli možemo da vidimo osnovna svojstva varijanti evolutivnog algoritma.

Tabela 1: Varijante EA algoritma

Svojstvo	SAWEA	RFEA	FlipGa	ASAP
zamena	$(1, \lambda^*)$	steady-state	generacijski	$(1 + 1)$
selekcija	-	turnirska	fitnes proporcijalna	-
fitness	$f_{SAW}$	$f_{REF}$	$f_{MAXSAT}$	$f_{MAXSAT}$
ukrštanje	-	-	uniformno	-
mutacija	MutOne	knowledge-based	random	random-adaptive
lokalna pretraga	-	-	flip heuristika	flip heuristika
adaptacija	fitnes	fitnes	-	tabu lista

#### 3.5.1 SAWEA

#### 3.5.2 RFEA

#### 3.5.3 FlipGA

#### 3.5.4 ASAP

### 3.6 Rezultati

## 4 Optimizacija rojem čestica (PSO)

Optimizacija rojem čestica (Particle swarm optimization – PSO) je jedna od tehnika pretraživanja zasnovana na populaciji kao što je genetski algoritam, ali ne koriste evolutivne algoritme kao što su mutacija i ukrštanje. PSO algoritmi su 1995. godine uveli Kenedi i Eberhart kao alternativu standardnim genetskim algoritmima.

Optimizacija rojem čestica je algoritam zasnovan na ponašanju pojedinačnih jedinki unutar određene grupe (na primer, jata ptica ili roja insekata). Ukoliko se, vođeno instinktom, jato prica uputi u određenom smeru u potrazi za hranom, očekivanje je da će čitavo jato slediti upravo onu pticu koja je pronašla izvor hrane. Međutim, i svaka ptica ponaosob može biti vođena sopstvenim instinktom i time na trenutak u potrazi za hranom napustiti jato. Tada se verovatno može desiti da, ukoliko pronađe bolji izvor hrane, čitavo jato upravo krene da sledi tu pticu.

PSO pripada skupu algoritama koji se zasnivaju na inteligenciji roja (swarm intelligence). Algoritam radi nad skupom jedinki, koji se naziva rojem. Elementi ovog skupa se nazivaju česticama. Svaka čestica predstavlja kandidatsko rešenje optimizacionog problema. Čestice se na unapred definisan način kreću po prostoru pretraživanja. Njihovo kretanje se usmerava imajući u vidu njihovu trenutnu poziciju, njihovu do sada najbolju poziciju, kao i do sada najbolju poziciju čitavog roja. Pod najboljom pozicijom čitavog roja se podrazumeva do sada najbolja pozicija, uzimajući u obzir sva njegova rešenja. Proces se ponavlja dok ne bude zadovoljen kriterijum zaustavljanja, a u svakoj iteraciji se ažurira najbolja vrednost rešenja za svaku česticu, kao i za roj u celini.

Neka je dat roj sa  $\vec{S}$  čestica. Svaka čestica se sastoji od tri elementa:

- Pozicija u prostoru za pretragu  $\vec{x}_i$
- Brzina, vektor  $\vec{v}_i$
- Sećanje, koje se koristi za skladištenje elitnih čestica globalne pretrage  $\vec{P}_g$ , kao i najboljih individualnih rešenja  $\vec{P}_i$  koja su do sada pronašle zasebne čestice

Nije neophodno da se u budućim populacijama nalazi bilo koji elitni pojedinac, iako svaka čestica u populaciji pokušava da bude blizu svog najboljeg rešenja i globalnog najboljeg rešenja.

Osnovni oblik PSO algoritma dat je sledećim samoažurirajućim jednačinama:

$$\vec{v}_i^{t+1} = w \cdot \vec{v}_i^t + c_1 \cdot \vec{r}_1 \times (\vec{P}_i^t - \vec{x}_i^t) + c_2 \cdot \vec{r}_2 \times (\vec{P}_g^t - \vec{x}_i^t) \quad (1)$$

$$\vec{x}_i^{t+1} = \vec{x}_i^t + \vec{v}_i^{t+1} \quad (2)$$

Jednačina 1 opisuje kako se ažurira brzina  $i$ -te čestice, a 2 koja je sledeća pozicija  $i$ -te čestice, pri čemu je:

- $w$  - faktor inercije
- $c_1, c_2$  - faktori učenja: kognitivna i socijalna
- $\vec{v}_{id}^t$  - brzina  $i$ -te čestice u iteraciji  $t$
- $\vec{x}_{id}^t$  - pozicija  $i$ -te čestice u iteraciji  $t$
- $\vec{r}_1, \vec{r}_2$  - pseudoslučajni brojevi iz uniformnog intervala  $[0, 1]$
- $\vec{P}_i$  - najbolje individualno rešenje čestice  $i$
- $\vec{P}_g$  - trenutno najbolje globalno rešenje

Kako je max k-SAT problem diskretan potrebno je prilagoditi jednačinu 2. Izračunata brzina  $\vec{v}_i$  je iz  $\mathbb{R}^n$ , pa je potrebno da se svede na  $\{0, 1\}^n$ . Jedan predlog za ažuriranje položaja čestice, izložen u radu [1], dat je sigmoidnom transformacijom. Sada  $v_i^t$  predstavlja verovatnoću da bit  $x_i^t$  uzme vrednost 1.

$$x_i^t = \begin{cases} 1, & \text{rand}(0, 1) < \text{sigmoid}(v_i^t) \\ 0, & \text{inace} \end{cases} \quad (3)$$

$$\text{sigmoid}(v_i^t) = \frac{1}{1 + e^{-v_i^t}} \quad (4)$$

## 4.1 Pseudokod PSO

U ovom poglavlju dat je osnovni oblik algoritma na kojem se zasnivaju ostale varijante i izmene koje će biti detaljnije izložene. Jednu populaciju, odnosno roj, čini unapred određen broj čestica, lista potencijalnih rešenja kao i dodeljene brzine za svaku od njih. Kroz iteracije računa se fitnes, ažuriraju se brzine i pozicije čestica dok se ne zadovolji kriterijum zaustavljanja. Jedna varijanta (WPSOSAT) uvodi i lokalnu pretragu koja se izvodi umesto ažuriranja pozicija, odnosno jednačine 3. Kriterijumi zaustavljanja koji se mogu koristiti u opštem slučaju su: da li je dostignut maksimalan broj unapred zadatih iteracija ili, da li u poslednjih nekoliko iteracija nema značajnog napretka. U test primerima za koje unapred znamo da je formula zadovoljiva, ili koliko je klauza zadovoljivo,

možemo koristiti kriterijum da se dostigao ukupan broj zadovoljivih klauza.

```

Input : Formula  $F$  u KNF-u,  $n$  i  $m$ 
Output: Najbolja procenjena valuacija i broj zadovoljenih klauza
inicijalizacija populacije: pozicije i brzine;
 $t = 0$ ; // tekuća iteracija
while nije zadovoljen uslov zaustavljanja do
     $t = t + 1$ ;
    for  $i \leftarrow 0$  to broj cestica u roju do
        Izračunaj  $Fitness(\vec{P}_i^t)$ ;
        Sačuvaj individualni najbolji rezultat kao globalni  $\vec{P}_g$ ;
        Ažuriraj brzine na osnovu  $\vec{P}_i$  i  $\vec{P}_g$ ;
        Ažuriraj pozicije  $\vec{v}_i^t$ ;
        Ažuriraj individualni najbolji rezultat  $\vec{P}_i$ ;
        Ažuriraj globalni najbolji rezultat  $\vec{P}_g$ ;
    end
end

```

**Algorithm 2:** Osnovni PSO algoritam

## 4.2 Inicijalizacija rešenja

Potrebno je inicijalizovati pozicije čestica i vektora brzine. Pozicije su inicijalizovane pseudo-slučajnim brojevima  $\{0,1\}$ , a vektor brzine realnim brojevima iz intervala  $[-V_{min}, V_{max}]$ , gde su granice intervala jedan od parametara PSO algoritma.

## 4.3 Fitnes funkcija

Fitnes funkcija je veoma važna za performanse algoritma. Prva fitnes funkcija koja se sama nameće jeste broj zadovoljenih klauza, kakva je data u samoj formulaciji problema, ali se takva funkcija nije pokazala kao dovoljno dobra. Bolji mehanizam je stepenasto ažuriranje težina (SAW - stepwise adaptation weights) uvedena od strane Eiben-a [2]. Ona je data sledećim formulama:

$$F_{SAW}(x) = \sum_{i=1}^m W_i C_i(x) \quad (5)$$

$$W_{i+1} = W_i + 1 - C_i(x^*) \quad (6)$$

Svakoј klauzi  $C_i$  dodeljuje se težina  $W_i$ . Ova funkcija ima za cilj identifikovanje težih klauza u procesu učenja koja je predstavljena većom vrednošću  $W_i$ . Na početku su težine inicijalizovane na 1, pa se potom ažuriraju jednačinom 6.  $x^*$  je tekuće najbolje rešenje.

## 4.4 Lokalna pretraga i flip heuristika

Da bi se unapredio standardni PSO algoritam uvedena je, umesto ažuriranja pozicija čestice jednačinom 3, stohastička lokalna pretraga heuristikom okretanja bitova (eng. *flip heuristic*). Ova heuristika zasniva se na izmeni pojedinačnih bitova u tekućem rešenju. Za svaki bit u čestici pokušava se njegovo okretanje. Ako sa tom izmenom ima dobiti (u smislu ne pogoršavanja funkcije cilja) čuva se nova vrednosti okrenutog bita, dok se u suprotnom vraća na njegovu staru vrednost. Čitav proces se ponavlja dokle god ima napretka, ako je bar jedna izmena poboljšala tekuće rešenje.

**Input** : pozicija čestice  $p$ , Formula  $F$  u KNF-u, maxFlip  
**Output**: nova pozicija čestice  $p$

```

improvement = 1;
numFlip = 0;
while improvement > 0 and numFlip < maxFlip do
    improvement = 0;
    for  $i \leftarrow 0$  to  $n$  do
        flip  $p[i]$ ;
        numFlip += 1;
        Izračunaj dobit: gain;
        if gain >= 0 then
            prihvati flip;
            improvement += gain;
        end
        else
            odbaci flip, vrati na staru vrednost  $p[i]$ ;
        end
    end
end

```

**Algorithm 3:** Funkcija lokalne pretrage

## 4.5 Varijante PSO algoritma

Da bismo uporedili kombinaciju lokalne pretrage, SAW fitnes funkcije i klasičnog PSO algoritma implementirani su i testirane sledeće tri verzije:

### 4.5.1 PSO-LS

### 4.5.2 PSOSAT

### 4.5.3 WPSOSAT

PSO-LS - Osnovna varijanta algoritma koji ne koristi lokalnu pretragu, već sigmoidnu transformaciju, jednačine 1 i 3 za ažuriranje brzina i pozicije čestica. Fitnes funkcija je  $F_{SAW}$ , sa korišćenjem težina nad klauzama. Karakteriše ga sporija konvergencija do globalnog optimuma, ali pojedinačne iteracije se izvršavaju brže.

PSOSAT - Koristi lokalnu pretragu, ali ne i  $F_{SAW}$ , pa je funkcija cilja broj zadovoljenih klauza. Mana ovog algoritma je teško izlaženje iz lokalnih optimuma zbog korišćenja fitnes funkcije koja ne raznanaže težinu klauza.

WPSOSAT - Modifikovan PSO algoritam sa korišćenjem flip heuristike i  $F_{SAW}$  fitnes funkcije. Značaj lokalne pretrage ogleda se u poređenju sa PSO-LS, a korišćenje  $F_{SAW}$  u poređenju sa PSOSAT.

## 4.6 Rezultati

Svi algoritmi pokretani su pet puta sa istim parametrima i beležen je prosečan broj zadovoljenih klauza, pri čemu podebljan rezultat označava da se do rešenja dolazilo u prvoj iteraciji. U tabeli 3 skoro svi algoritmi su uspeli da brzo nađu rešenje. Za sada između PSOSAT i WPSOSAT nema razlike. Već za poslednji test vidi se slabost ne korišćenja lokalne pretrage. Već u tabeli 4 PSO-LS nije mogao da se uporedi sa ostala dva algoritma. Za neke instance do globalnog optimuma došao je jedino WPSOSAT odakle se vidi značaj SAW funkcije.

## 5 Zaključak

Tabela 2: Parametri

Parametar	Vrednost
Broj iteracija	1000
w	1
c1	1.7
c2	2.1
Broj čestica	20
max flip	30000
$v_{min}$	-1
$v_{max}$	1

Tabela 3: AIM nezadovoljivi testovi

Instanca	Broj literala	Broj klauza	PSO-LS	PSOSAT	WPSOSAT
aim-50-1.6-no	50	80	79	<b>79</b>	<b>79</b>
aim-50-2.0-no	50	100	99	<b>99</b>	<b>99</b>
aim-100-1.6-no	100	160	159	<b>159</b>	<b>159</b>
aim-100-2.0-no	100	200	199	<b>199</b>	<b>199</b>
aim-200-2.0-yes	200	400	398.6	<b>399</b>	<b>399</b>

Tabela 4: AIM zadovoljivi testovi

Instanca	Broj literala	Broj klauza	PSO-LS	PSOSAT	WPSOSAT
aim-50-1.6-yes	50	80	79	79.2	80
aim-50-2.0-yes	50	100	99	99	100
aim-50-3.4-yes	50	170	168.6	170	170
aim-50-6.0-yes	50	300	300	300	300
aim-100-1.6-yes	100	160	158.6	159	159
aim-100-2.0-yes	100	200	198	199	200
aim-100-3.4-yes	100	340	328	339.4	340
aim-100-6.0-yes	100	600	580.2	<b>600</b>	<b>600</b>
aim-200-2.0-yes	200	400	395.4	399	399
aim-200-6.0-yes	200	1200	1135.6	<b>1200</b>	<b>1200</b>

## Literatura

- [1] Kennedy J, Eberhart RC. "A discrete binary version of the particle swarm algorithm," In: Proceedings of IEEE conference on systems, man and cybernetics , p. 4104–4109, 1997.
- [2] Gottlieb J, and Voss N. "Adaptive fitness functions for the satisfiability problem". In: Parallel Problem Solving from Nature - PPSN VI 6th International Conference, Paris, France. Springer Verlag. LNCS 1917, 2000.
- [3] Abdesslem Layeb, "A particle swarm algorithm for solving the maximum satisfiability problem", MISC Lab., Computer science department, University mentouri of Constantine, Algeria