Genetski algoritmi (GA)

Gordana Popović

(Radni materijal za školsku godinu
2003-2004)

Genetski algoritmi (GA)

Sadržaj:

Evolutivni programi (EP)

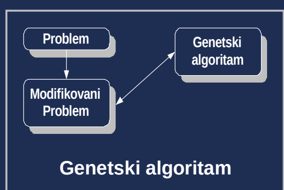
- **Poslednjih 30 godina razvijeni su sistemi zasnovani na principima prirodne evolucije i nasleđivanja.**
  - **Evolutivne strategije (Rechenberg i Schewefel),**
  - **Evolutivno programiranje (Fogel),**
  - **Rasuto (*Scatter)* pretraživanje (Glover),**
  - **Genetski algoritmi (Holland 1975),**
  - **Genetsko programiranje (Koza),**
  - **Itd.**

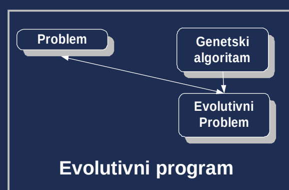
Evolutivni programi (EP)

- **Osnovana karakteristike evolutivnih sistema:**
  - U svakom koraku (iteraciji - *t*) kreira se populacija jedinki – potencijanih rešenja (*P*(*t*)).
  - Svaka jedinka predstavlja potencijalno rešenje problema i implementirana je pomoću neke struktura podataka.
  - Za svaku jedinku se izračunava pogodnost (*fitness*) i selektuju se najpogodnije jedinke za sledeću generaciju, a one loše umiru.
  - Na neke jedinke mogu biti primenjeni genetski operatori ukrštanja i mutacije.

Osnove genetskih algoritama (GA)

- GA su kompjuterski algoritmi za rešavanje optimizacionih problema.
- Za potrebe GA kreiran je mehanizam predstavljanja jedinki pomoću *hromozoma*, koji se sastoje iz *gena* što je potpuno analogno ljudskim hromozomima.
- Osnovna razlika između GA i EP je što se kod klasičnih GA problem modifikuje u odgovarajuću formu, dok EP rešava originalni problem.





5

Osnove pojmovi u GA

Priroda	*Genetski algoritam*
Hromozom Genotip	String
Gen	Karakter
Lokacija	Pozicija karaktera u stringu
Skup hromozoma	Populacija
Fenotip	Dekodirana struktura

6

GA (EP)

begin
    $t \leftarrow 0$
    inicijalizacija *P(t)*
    evaluacija *P(t)*
    while (not ispunjen uslov*) do*
    begin
        $t \leftarrow t + 1$
        selekcija *P(t)* iz *P(t - 1)*
        ukrštanje *P(t)*
        mutacija *P(t)*
        evaluacija *P(t)*
    end
end

7

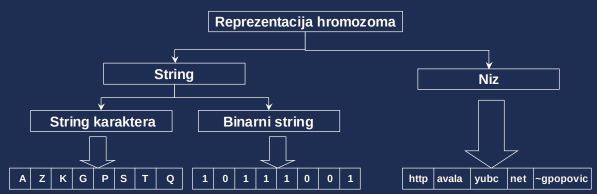
Komponente GA

- Genetska reprezentacija (*kod*) potencijalnih rešenja,
- Način kreiranje inicijalne populacije,
- Evaluaciona funkcija koja igra ulogu okruženja,
- Genetski operatori za izmenu koda dece,
- Parametri optimizacije.

8

Načini kodiranja u GA

- Postoje dva načina kodiranja:



- Binarno kodiranje:

$$x_i = a_i + deciamal(string_2)\frac{b_i - a_i}{2^{m_i} - 1},\quad \begin{array}{l}(b_i - a_i) * 10^6 \leq 2^{m_i} - 1 \\ [a_i, b_i] = D_i \subseteq R \quad i \\ f(x_1, \ldots x_k) > 0, \forall x_i \in D_i.\end{array}$$

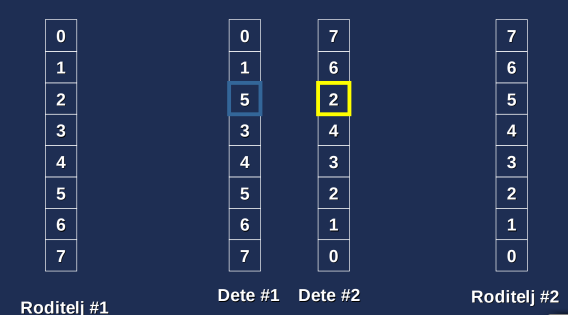
Inicijalizacija i selekcija

- Inicijalizacija se obično vrši izborom *k* jedinki na slučajan način ili se uključuju neka predznanja o raspodeli potencijalnih rešenja.
- Selekcija se može vršiti na osnovu distribucije verovatnoća zasnovane na vrednostima funkcije podobnosti (*fitness*).
- *Točak za rulet* :
  1. *Fitness* za svaku jedinku *eval* (v_i), i=1,...,k.
  2. Ukupan *fitness* populacije $F = \sum_{i=1}^{k} eval(v_i)$
  3. Verovatnoća selekcije za svaki homozom $p_i = eval(v_i)/F$
  4. Kumulativnu verovatnoću $q_i = \sum_{j=1}^{i} p_j$
- Proporcionalna selekcija se vrši oketanjem *točaka k* puta i u svakom krugu se bira 1 hromozom za novu generaciju:
  - Generiše se slučajan broj *r* iz opsega [0…1],
  - Ako je $r<q_1$ selektuje se prvi hromozom, u suprotmon *i*-ti hromozom za koji vazi da je $q_{i-1} \leq r \leq q_i$.

Ukrštanje

- Ukrštanje je genetski koncept seksualne reprodukcije pri kom se kombinuje genetski materijal dva roditelja u cilju dobijanja superiornog naslednika.
- Pri jednostavnom ukrštanju se slično kao i pri selekciji izabere slučajan broj *r* i ako je on manji od verovatnoće ukrštanja p_c vrši se ukrštanje. To znači da ukrštanju podleže p_c*k hromozoma.
- Postoje dve osnovna tipa ukrštanja:
  - Jenopoziciono i
  - Višepoziciono.

Jednopoziciono ukrštanje



Jednopoziciono ukrštanje za binarni kod:

Roditelj #1	Dete #1	Dete #2	Roditelj #2
0	0	1	1
1	1	1	1
0	0	0	0
0	0	1	1
1	0	1	0
0	1	0	1
1	1	1	1
1	0	1	0

13

Višepoziciono ukrštanje

Roditelj #1	A	B	C	D	E	F	G	H
Roditelj #2	I	J	K	L	M	N	O	P

14

Višepoziciono ukrštanje

Roditelj #1	A	B	C	D	E	F	G	H
Dete #1	A	J	C	D	M	F	O	H
Dete #2	I	B	K	L	E	N	G	P
Roditelj #2	I	J	K	L	M	N	O	P

15

Višepoziciono ukrštanje

Za binarni kod:

Roditelj #1	0	1	0	0	1	0	1	1
Dete #1	0	1	0	1	0	1	1	1
Dete #2	1	1	0	1	0	1	1	0
Roditelj #2	1	1	0	1	0	1	1	0

16

Uniformno višepoziciono ukrštanje

Maska	0	1	0	0	1	0	0	1
Roditelj #1	0	1	0	0	1	0	1	1
Roditelj #2	1	1	0	1	0	1	1	0
Dete #1	1	1	0	1	1	1	1	1
Dete #2	0	1	0	0	0	0	1	1

17

Mutacija

- Mutacija vraća divergenciju u populaciju.
- Mutacija se vrši na malom delu populacije da ne bi došlo do nestabilnosti procedure.
- Pri mutaciji se na isti način kao i pri ukrštanju izabere slučajan broj *r* i ako je on manji od verovatnoće mutacije p_m vrši se izmena gena. To znači da mutaciji podleže p_m*k*m gena.

18

Mutacije

Roditelj	1	1	0	1	0	0	0	1
Dete	0	1	0	1	0	1	0	1

19

Fitness – funkcija pogodnosti

- *Fitness* funkcija je evaluaciona funkcija koja određuje da li je neko rešenje bolje od drugih.
- *Fitness* funkcija se računa za svaku jedinku.
- *Fitness* funkcija zavisi od problema koji se rešava.

20

Parametri optimizacije

- ***Potrebno je zadati sledeće parametre:***
  - Veličinu populacije – k
  - Verovatnoću ukrštanja - p_c
  - Verovatnoću mutacije - p_m
  - Kriterijum zaustavljanja.

Shema – teorema

- **Za svaki hromozom dužine m se može napisati 2^m shema odnosno svakoj shemi odgovara 2^r hromozoma ako je r broj (*) simbola sheme.**
- **Jedna shema može imati izgled:**

  S = (* * * 0 0 1 * 1 1 0)

- **<u>Red šeme</u> je broj fiksnih pozicija u shemi o(S)=6**
- **<u>Definiciona dužina sheme</u> je rastojanje između prvog i poslednjeg fiksnog simbola δ(S) = 10 - 4**

Shema – teorema

- **Jednačina rasta sheme:**

$$\xi(S,t+1) \geq \xi(S,t)\,eval(S,t)/\overline{F}\left[1 - p_c \frac{\delta(s)}{m-1} - o(s)\,p_m\right].$$

- **<u>Teorema:</u> Broj hromozoma koji odgovara shemi niskog reda, sa malom definicionom dužinom i nadprosečnom pogodnošću ekponencijalno raste iz generacije u generaciju.**
- **<u>Hipoteza o gradivnim blokovim:</u> Genetski algoritam teži ka rešenju koje je blizu optimalnog pomoću shema niskog reda, sa malom definicionom dužinom i visokom pogodnošću, koje se nazivaju gradivni blokovi.**

Modifikacije GA

- **Razlozi za preranu konvergenciju:**
  - **Kodiranje problema**
  - **Ograničen broj iteracija i veličina populacije**
- **Strategije za sprečavanje prerane konvergencije (*sprečavanje incesta*) mogu biti usmerene u dva prvaca:**
  - **Mehanizam selekcije**
  - **Uključivanje karakteristika funkcija u algoritam.**

Mehanizmi selekcije

- Elitistički model,
- Model očekivane vrednosti,
- Elitistički model očekivane vrednosti,
- Model faktora prepunjenosti,
- Boltzmann-ova selekcija,
- Stohastički, dinamički, statički modeli,
- *Tournament model* (takmičarski model),
- Generacijski modeli, itd.

25

Struktura modGA

begin
  $t \leftarrow 0$
  inicijalizacija *P(t)*
  evaluacija *P(t)*
  while (not ispunjen uslov*) do*
  begin
    $t \leftarrow t + 1$
    selekcija *r roditelja* iz *P(t -1)*
    selekcija (*k-r*) *smrtnika* iz *P(t -1)*
    reprodukcija roditelja iz *P(t)*
    evaluacija *P(t)*
  end
end

26

Ostale modifikacije

- Skaliranje funkcije (linearno skaliranje, sigma transakciono, stepenovanje, itd)
- Model kontraktivnog mapiranja (Banach-ova teorama fiksnih tačaka u metričkom prostoru)
- GA sa promenljivim veličinama populacije

27

Ostale modifikacije

- GA za uslovnu optimizaciju (problem ranca):

$$\text{Max } P(x) = \sum_{i=1}^{n} P_i x_i$$

p.o.

$$\sum_{i=1}^{n} W_i x_i \le C, \quad x_i \in \{0,1\}, \forall x$$

- Algoritmi zasnovani na kaznenim funkcijama,

$$eval(x) = \sum_{i=1}^{n} P_i x_i - Pen(x)$$

- Algoritam zasnovan na metodi oporavka (slučajni oporavak i *greedy* metod),

$$eval(x) = \sum_{i=1}^{n} P_i x_i'$$

- Algoritam zasnovan na dekodiranju (slučajno dekodiranje, i *greedy* metod).

28

EP za numeričku optimizaciju

- Realno ili decimalno kodiranje je blize realnosti pošto je prostor dopustivih rešenja sličan kao prostor originalnog problema.
- Svaki hromozom predstavlja vektor realnih brojeva koji je iste dužine kao i vektor rešenja.
- Smanjuje se vreme evaluacije i zahtevani memorijski prostor računara.
- Preciznost je veća nego kod binarnog kodiranja i lakše se primenjuje na probleme sa netrivijenim ograničenjima.

29

Specijalizovani operatori

- Uniformna mutacija, $\langle v_1, \ldots, v_i', \ldots, v_k \rangle, \quad v_i' \in D_i$
- Neuniformna mutacija, $v_i' = \begin{cases} v_i + \Delta(t, u_k - v_k), & za \quad 0 \\ v_i - \Delta(t, v_k - l_k), & za \quad 1 \end{cases}$
- Jednostavno ukrštanje,
- Aritmetičko ukrštanje:

Ako se ukrštaju vektori s_v^t i s_w^t rezultijući vektori su :

$s_v^{t+1} = a s_w^t + (1 - a) s_v^t$ **i**

$s_w^{t+1} = a s_v^t + (1 - a) s_w^t$

30

Problem trgovačkog putnika (TSP)

- Zadatak trgovačkog putnika je da tačno jednom obiđe svaki od *n* gradova na svojoj teritoriji i da se vrati na startnu tačku pri čemu je cilj da troškovi putovanja budu minimalni.
- Dopustivi skup rešenja predstavljaju permutacije od *n* gradova. Ima ih ukupno *n-1*!.
- Svaka permutacija predstavlja potencijalno rešenje problema, a optimalno rešenje je permutacija sa minimalnim troškovima.

31

TSP-Reprezentacija putanje

- Ovo je najprirodniji način kodiranja.

Putanja → Reprezentacija

5 – 1 – 7 –8 – 9 – 4 – 6 – 2 - 3 → (5 1 7 8 9 4 6 2 3)

- Ukrštanje

R1=(1 2 3 4 5 6 7 8 9) → D1=(1 2 3 1 5 6 7 8 9)

R2=(4 5 2 1 8 7 6 9 3) → D2=(4 5 2 4 8 7 6 9 3)

32

TSP-Reprezentacija putanje

– Delimično mapirano ukrštanje - PMX

R1=(1 2 3 | 4 5 6 7 | 8 9) → D1=(x x x 1 8 7 6 x x) → D1=(4 2 3 1 8 7 6 5 9)

R2=(4 5 2 | 1 8 7 6 | 9 3) → D2=(x x x 4 5 6 7 x x) → D2=(1 8 2 4 5 6 7 9 3)

Mapiranje: 1 Û 4, 8 Û 5, 7 Û 6 i 6 Û 7

– Redno ukrštanje - OX

R1=(1 2 3 | 4 5 6 7 | 8 9) → D1=(x x x 4 5 6 7 x x) → D1=(2 1 8 4 5 6 7 9 3)

R2=(4 5 2 | 1 8 7 6 | 9 3) → D2=(x x x 1 8 7 6 x x) → D2=(3 4 5 1 8 7 6 9 2)

Mapiranje: 9 – 3 – ~~4~~ – ~~5~~ – 2 –1 – 8 - ~~7~~ - ~~6~~

– Ciklično ukrštanje - CX

R1=(1 2 3 4 5 6 7 8 9) → D1=(1 2 3 4 x x x 8 x) → D1=(1 2 3 4 7 6 9 8 5)

R2=(4 1 2 8 7 6 9 3 5) → D2=(4 1 2 8 x x x x x) → D2=(4 1 2 8 5 6 7 3 9)

TSP-Ordinalna reprezentacija

- Ako se kodira lista od *n* gradova *i*-ti element liste je iz opsega od 1 do *n*-*i*-1.

Uređena lista gradova → Putanja → Ordinalna reprezentacija

S= (1 2 3 4 5 6 7 8 9) → 1 – 2 – 4 –3 – 8 – 5 –9 –6 - 7 → *L*=(1 1 2 1 4 1 3 1 1)

– Klasično ukrštanje

R1=(1 1 2 1 | 4 1 3 1 1) → D1=(1 1 2 1 5 3 3 2 1)

R2=(5 1 5 5 | 5 3 3 2 1) → D2=(5 1 5 5 4 1 3 1 1)

P1=(1 – 2 – 4 – 3 - 8 - 5 - 9 – 6 - 7) → P1=(1 – 2 – 4 – 3 - 9 - 7 - 8 – 6 - 5)

P2=(5 – 1 – 7 – 8 - 9 - 4 - 6 – 3 - 2) → P2=(5 – 1 – 7 – 8 - 6 - 2 - 9 – 3 - 4)

TSP-Matrična reprezentacija (Mc-Mahon)

- Matrični element m_{ij} sadrži 1 ako i samo ako se grad *i* pojavljuje pre grada *j* u putanji. Npr. putanja (3 1 2 8 7 4 6 9 5) se predstavlja matricom:

	1	2	3	4	5	6	7	8	9
1	0	1	0	1	1	1	1	1	1
2	0	0	0	1	1	1	1	1	1
3	1	1	0	1	1	1	1	1	1
4	0	0	0	0	1	1	1	1	1
5	0	0	0	0	0	0	0	0	0
6	0	0	0	0	1	0	0	0	1
7	0	0	0	1	1	1	0	0	1
8	0	0	0	1	1	1	1	0	1
9	0	0	0	0	1	0	0	0	0

- Genetski operatori:

1. Presek – u prvoj fazi se pronalazi presek matrica dva roditelja, a u sledećoj se dobijenoj matrici dodaju 1 jednog od roditelja i kompletira se putanja analizom sume redova i kolona.
2. Unija – kreiraju se dva disjunktivna skupa gradova pri čemu se prepisuju delovi matrice prvog roditelja za prvu grupu i matrice drugog roditelja za drugu grupu. Matrica se kompletira analizom suma redova i kolona.

TSP-Binarna matrična reprezentacija (Seniew)

- Matrični element m_{ij} sadrži 1 ako i samo ako se iz grad *i* ide direktno u grad *j*. Npr. putanja (1 2 4 3 8 6 5 7 9) se predstavlja matricom:

	1	2	3	4	5	6	7	8	9
1	0	1	0	0	0	0	0	0	0
2	0	0	0	1	0	0	0	0	0
3	0	0	0	0	0	0	0	1	0
4	0	0	1	0	0	0	0	0	0
5	0	0	0	0	0	0	1	0	0
6	0	0	0	0	1	0	0	0	1
7	0	0	0	0	0	0	0	0	1
8	0	0	0	0	0	1	0	0	0
9	1	0	0	0	0	0	0	0	0

- Genetski operatori:

1. Mutacije – slučajno se u hromozomu biraju redovi i kolone, a zatim se brišu bitovi u njihovom preseku i zamenjuju novim konfiguracijama.
2. Ukrštanje– sva polja hromozoma deteta se postavljaju na 0. Zatim se upoređuju polja matrica roditelja i prepisuju ako su ista, nakog čega se naizmenično prepisuju vrednosti polja, i na kraju se analiziraju vrednosti kolona i redova i dopunjavaju na slučajan način.

Zaključak

- Evolutivni programi su zasnovani na prirodnim zakonima i zato su lako razumljivi.
- Eksterimentalno je dokazano da EP daju dobra rešenja i da su vremenski efikasni.
- Do sada su objavljeni brojni radovi sa primenama GA i EP na raznim optimizacionim problemima uključujući raspoređivanje, bikonektivene računarske mreže pa čak i transportni problem. Takođe se primenjuju u *data mining* sistemima, sistemima rutiranja i pretraživanja interneta.
- GA i EP pružaju široke mogućnosti istraživanja novih načina reprezentacije netrivijelnih problema, rešavanja problema skaliranja, rešavanja problema uslovne optimizacije, itd.

37