Genetski algoritam

Verzija implementiranog algoritma je *k*-turnirska eliminacijska selekcija.
Križanje je rađeno sa *tp* - točaka prekida (*k* i *tp* su paramatri algoritma).

U prvom dijelu u svrhu razumjevanja postupka napraviti ću primjer jedne iteracije 3-turnirske eliminacijske selekcije.

Minimizirati ću sljedeću funkciju:

$$f(x) = 10 + x^2 - 10 \cdot \cos(2 \cdot \pi \cdot x), \qquad x \epsilon [-5, 5].$$

Preciznost: *2* decimale (*p=2*).

Neka je trenutačna populacija: $\{-3.58, -3.42, -0.78, -0.15, 2.92, 3.00, 4.16, 4.57, 4.59, 4.71\}$

Računamo potreban broj bitova za reprezentaciju stanja:

$$n \geq \frac{\log[(gg - dg) \cdot 10^p + 1]}{log2} = \frac{\log[(5+5) \cdot 10^2 - 1]}{log2} \approx 9.9672$$

Odabir je n=10, najmanji broj bitova koji zadovoljava gornju relaciju.
Raspolažemo sa 2^{10} stanja kojima interval želimo podijeliti na $2^{10} - 1$ dijelova veličine $i = \frac{gg-dg}{2^{n-1}} = \frac{5+5}{2^{10-1}} = 0.0195$.

Reprezentacija jedinki $\{-3.58, -3.42, -0.78, -0.15, 2.92, 3.00, 4.16, 4.57, 4.59, 4.71\}$ u tom prostoru stanja prikazana je u prva 4 stupca sljedeće tablice.

Reprezantacija jedinki				Fitness	
x	$n10 = round(\frac{x-(-5)}{i})$	***n2***	***n_gray***	***f(x)***	***F(x)= -f(x)+maxf***
-3.58	73	0001001001	0001101101	30.4595	9.4737
-3.42	81	0001010001	0001111001	34.6710	5.2622
-0.78	216	0011011000	0010110100	39.9332	0
-0.15	249	0011111001	0010000101	4.1446	35.7885
2.92	406	0110010110	0101011101	9.0000	30.9332
3.00	410	0110011010	0101010111	31.5795	8.3537
4.16	470	0111010110	0100111101	8.7346	31.1986
4.57	491	0111101011	0100011110	21.9473	17.9858
4.59	492	0111101100	0100011010	9.7633	30.1698
4.99	513	1000000001	1100000001	24.9198	15.0133

maxf=39.9332

Nadalje umjesto binarnog alfabeta koristit ćemo Grayev alfabet. Važna karakteristika Grayevog alfabeta je da se svake dvije susjedne binarne riječi razlikuju u točno jednom bitu. To pak znači da iz bilo kojeg rješenja možemo doći do onog koje je za jedan kvant veće ili manje promjenom samo jednog bita (ne bilo kojeg). Stoga je upotreba Grayevog alfabeta umjesto binarnog često bolje rješenje.

Sada radimo 3 - turnirsku eliminacijsku selekciju.
Na slučajan način izaberemo 3 jedinke neka su to npr. {3.00, -0.15, 4.59}, {0101010111, 0010000101, 0100011010 }. Kako jedinka 0101010111 ima najmanji fitness algoritam će je zamijeniti novom jedinkom koju ćemo dobiti križanjem i mutacijom preostalih dviju jedinki.

Napravimo križanje sa 3 točke prekida: 3-4, 6-7, 8-9. Križanje radimo "kriš-kraš".

Roditelj 1:	001|000|01|01
Roditelj 2:	010|001|10|10
Dijete 1:	001|001|01|10
Dijete 2:	010|000|10|01

Nakon križanja odaberemo jedno dijete te nad njim napravimo operaciju mutacije. Neka je to Dijete 1. Operacija mutacije djeluje tako da svaki bit (alel) invertira uz malu određenu vjerojatnost.

Jedan primjer takve mutacije dan je sljedećom tabelom:

Dijete 1	0010010110
DIjete 1 nakon mutacije	0011010110

Promjena se dogodila na četvrtom bitu (alelu) jedinke.
Radimo pretvorbu nazad u binarni alfabet te dobivamo *n2 = 0010011011, n10 = 155, x = 2.9250, f(x) = 9.6456 i F(x) = 30.2876.*

Genetski algoritam - struktura

Implementacija algoritma je napravljena u MatLabu, algoritam minimizira funkcije ne nužno jedne već i više varijabli na zadanim segmentima. Implementirana je k - turnirska eliminacijska selekcija, križanje je rađeno sa *tp* točaka prekida gdje su *k* i *tp* jedni od parametara algoritma. Mutacija je rađena tako da se svaki alel mijenja sa vjerojatnošću 1/duljina_stringa.

Osnovna struktura koju sam koristio pri implementaciji je polje ćelija (cell arrey), tj polja koja mogu sadržavati podatke različitih tipova i veličina.

U polju ćelija držim jedinke populacije tj. sve njihove podatke potrebne za realizaciju algoritma.
Dakle glavno polje ćelija mi izgleda ovako:

$\{jedinka_1, jedinka_2, jedinka_3, \ldots jedinka_m\}$,

gdje je *m* veličina populacije. Svaka $jedinka_i$ je oblika:

$jedinka_i = \{\{x_1,x_2,\ldots x_n\}, \{n10_{x1,}\, n10_{x2,\ \ldots}\, n10_{xn}\}, \{n2_{x1,}\, n2_{x2,\ \ldots}\, n2_{xn}\}, \{n\_gray_{x1}, n\_gray_{x2}, \ldots n\_gray_{xn}\}, f(x_1,x_2,\ldots x_n), F(x_1,x_2,\ldots x_n)\}$,

gdje je *n* broj varijabli funkcije f, $n2_{xi}$ je prikaz u prostoru stanja dobiven međurezultatom $n10_{xi}$, n_gray_{xi} je gray zapis $n2_{xi}$.
Vrijednost funkcije u varijablama $(x_1,x_2,\ldots x_n)$ dana je s $f(x_1,x_2,\ldots x_n)$, dobrota je dana s $F(x_1,x_2,\ldots x_n)$.
Ćelije $\{n2_{x1,}\, n2_{x2,\ \ldots}\, n2_{xn}\}$ i $\{n\_gray_{x1}, n\_gray_{x2}, \ldots n\_gray_{xn}\}$ su ćelije stringova.

S tako definiranim poljem ćelija algoritam prolazi kroz iteracije, radi k-turnirsku eliminacijsku selekciju, obavi operacije križanja i mutacije, te kada prođe kroz sve iteracije vraća jedinku sa najvećom vrijednošću funkcije dobrote F. Funkcija dobrote je oblika F=-f+K, gdje je za K uzet upravo maxf.

Goran Brajdić, mag.inf. et math.

Gray alfatet vs Binarni alfabet

Sljedećom tablicom prikazana su uprosjećena vremena i reziduali genetskog algoritma
koji radi sa binarnim i gray-evim alfabetom za iste vrijednosti parametara.
Seed je isti za obje verzije (gray i bin) a mijenjao se u svakom od 20 ponovljenih pokusa.
Reziduali su računati kao $\left|f(x\_min) - f(\hat{x}\_min)\right|$ gdje je x_min pravi minimum f-je, a $\hat{x}\_min$ minimum dobiven genetskom algoritmom.

Funkcija	**Gray alfabet**		**Binarni alfabet**	
	Prosječno vrijeme (sec)	*Prosječni rezidual*	*Prosječno vrijeme (sec)*	*Prosječni rezidual*
$\boldsymbol{f(x) = (x-1)^2}$	*0.046637*	*7.1074e-007*	*0.031462*	*3.2307e-007*
$\boldsymbol{f(x) = 10 + x^2 - 10 \cdot \cos(2 \cdot \pi \cdot x)}$	*0.037241*	*0.0396*	*0.033081*	*0.0314*
$\boldsymbol{f(x,y) = x^2 + (y-1)^2}$	*0.100009*	*1.1720e-004*	*0.085484*	*8.0684e-006*
$\boldsymbol{f(x,y) = \cos(x) + 3 \cdot y^2}$	*0.142330*	*2.1679e-006*	*0.124833*	*1.0175e-004*
$\boldsymbol{f(x,y) = 10 + x^2 + y^2 - 10 \cdot \cos(2 \cdot \pi \cdot x)}$	*0.217269*	*6.3024e-004*	*0.181384*	*0.0094*

Dakle iz gornje tablice vidimo da je algoritam koji koristi gray alfabet malo sporiji od algoritma
s binarnim alfabetom.
Za prve tri funkcije reziduali su relativno blizu za oba alfabeta, no minimizacijom kompliciranijih
funkcija vidimo da gray alfabet verzija daje bolje rezultate od verzije sa binarnim alfabetom.

Za poziv algoritma koji optimizira i crta sve u tablici gore navedene funkcije pokrenite m-fajl "test" u matlabu.
Detaljno pojašanenje ulaznih i izlaznih parametara, kao i pomoćnih funkcija koje glavni program
koristi nalazi se na početku u m-fajlu *genetski_algoritam*.