6. domaća zadaća – višekriterijska optimizacija uporabom algoritma NSGA-II

Uvod

U ovoj zadaći implementirat ćete algoritam NSGA-II koji obavlja višekriterijsku optimizaciju funkcije.

U okviru ove domaće zadaće rješavat ćete optimizacijski problem prikazan u nastavku.

$$Problem:\begin{cases} minimiziraj & f_1(\vec{x}) = x_1, \\ minimiziraj & f_2(\vec{x}) = \frac{1+x_2}{x_1} \\ uzograničenje & 0.1 \leqslant x_1 \leqslant 1, \\ uzograničenje & 0 \leqslant x_2 \leqslant 5 \end{cases}$$

I prije samog postupka rješavanja, s obzirom na jednostavnost kriterijskih funkcija, možemo nešto zaključiti o njihovom odnosu. Vrijedi:

$$f_1 = x_1$$
$$f_2 = \frac{1+x_2}{f_1}$$

Stoga je lako zaključiti da će se uz fiksirani f_1 minimum od f_2 postići postavljanjem x_2 na vrijednost 0 što je njegova donja granica. Tada će vrijediti:

$$f_2 = \frac{1}{f_1}$$

što je izraz iz kojega možemo zaključiti kakav oblik Pareto fronte očekujemo.

Zadatak.

Napisati općenitu implementaciju algoritma NSGA-II. Pod pojmom općenitu podrazumijevamo implementaciju koja će moći raditi s proizvoljnim brojem kriterijskih funkcija. Prijedlog je problem višekriterijske optimizacije modelirati zasebnim sučeljem koje nudi barem dvije metode ("barem" jer će nedostajati još neke informacije o problemu koje možete po potrebi dodati):

```
interface MOOPProblem {
  int getNumberOfObjectives();
  void evaluateSolution(double[] solution, double[] objectives);
}
```

Metoda `getNumberOfObjectives()` vraća broj kriterija koji su definirani u okviru problema. Metoda `evaluateSolution` je zadužena za vrednovanje rješenja: prima točku iz prostora rješenja, nad njom računa vrijednosti svih kriterija i rezultat pohranjuje u polje koje je predano kao drugi argument. Umjesto predloženog oblika metode `evaluateSolution` možete koristiti i metodu oblika:

```
double[] evaluateSolution(double[] solution);
```

Jedina razlika je što prvi oblik dozvoljava da se polje u koje će biti pohranjen rezultat unaprijed stvori izvana i potom koristi svaki puta dok drugi oblik traži metodu `evaluateSolution` da svaki puta alocira novo polje u koje će pohraniti rezultat vrednovanja.

Također, pretpostavite da se po svim kriterijima **uvijek radi minimizacija**, odnosno da će, ako je problem drugačiji, on najprije biti pretvoren u minimizacijski problem.
Rad programa isprobajte na dva problema.

Problem 1.

Ovaj problem definiran je na sljedeći način. Potrebno je minimizirati:

$$minimiziraj \quad f_1(\vec{x}) = x_1^2$$
$$minimiziraj \quad f_2(\vec{x}) = x_2^2$$
$$minimiziraj \quad f_3(\vec{x}) = x_3^2$$
$$minimiziraj \quad f_4(\vec{x}) = x_4^2$$

pri čemu je $\vec{x} = (x_1, x_2, x_3, x_4) \in R^4$ uz ogradu:

$$-5 \leqslant x_i \leqslant 5, i \in \{1,2,3,4\}.$$

Razmislite li, brzo ćete uvidjeti da ovo nije uistinu višekriterijski problem (razumijete li zašto?). Poslužit će nam, međutim, za provjeru rada implementacije.

Problem 2.

Fokus ove domaće zadaće je primjena na problem opisan u uvodnom dijelu.

Rješenje

Napišite program `hr.fer.zemris.optjava.dz6.MOOP` koji preko naredbenog retka prima sljedeće argumente:

- *fja:* 1 za prvi problem, 2 za drugi problem,
- *n*: željena veličina populacije,
- *maxiter*: maksimalni broj generacija/epoha.

Napisati implementaciju koja *crowding* računa u prostoru ciljnih funkcija kako je i predviđeno izvornom verzijom algoritma.

Po završetku rada optimizacijskog procesa populaciju podijelite u fronte te ispišite koliko rješenja imate u svakoj fronti. U datoteku "izlaz-dec.txt" ispišite sva pronađena rješenja a u datoteku "izlaz-obj.txt" ispišite njihove dobrote; za problem 2 u ovoj posljednjoj datoteci ispisat ćete uređene parove (f_1, f_2). Sadržaj datoteke s dobrotama prikažite grafički (koristite neki gotov alat za vizualizaciju; npr. LibreOffice Draw može generirati graf s ovim podatcima). Fiksirajte osi prikaza na 0.1 do 1 za f_1 te na 0 do 10 za f_2.

Ako Vam za pokretanje programa treba još koja vrijednost koja je specifična za problem koji rješavate, predajte je kao argument naredbenog retka i to dokumentirajte u datoteci README.txt. Predana zadaća mora u vršnom direktoriju projekta imati i jednu PNG datoteku: sliku koja prikazuje dobiveni rezultat optimizacije problema 2 kada se koristi dijeljenje u prostoru kriterijskih funkcija.