Unaprjeđenja algoritma diferencijalne evolucije podešavanjem parametara i izborom počrtne populacije

Bajer, Dražen

Sveučilište Josipa Jurja Strossmayera u Osijeku

Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek

Dražen Bajer

Unaprjeđenja algoritma diferencijalne evolucije podešavanjem parametara i izborom početne populacije

Doktorska disertacija

Osijek, 2016.

Doktorska disertacija izrađena je na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija Osijek, Sveučilišta Josipa Jurja Strossmayera u Osijeku.

Mentor: dr. sc. Goran Martinović, redoviti profesor, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek, Sveučilište Josipa Jurja Strossmayera u Osijeku

Disertacija ima 145 stranica.

Disertacija broj: 59

Povjerenstvo za ocjenu doktorske disertacije:

1. Dr. sc. Željko Hocenski, redoviti profesor u trajnom zvanju, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek, Sveučilište Josipa Jurja Strossmayera u Osijeku

2. Dr. sc. Goran Martinović, redoviti profesor, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek, Sveučilište Josipa Jurja Strossmayera u Osijeku

3. Dr. sc. Janez Brest, redoviti profesor, Fakulteta za elektrotehniko, računalništvo in informatiko, Univerza v Mariboru, Slovenija

Povjerenstvo za obranu doktorske disertacije:

1. Dr. sc. Željko Hocenski, redoviti profesor u trajnom zvanju, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek, Sveučilište Josipa Jurja Strossmayera u Osijeku

2. Dr. sc. Goran Martinović, redoviti profesor, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek, Sveučilište Josipa Jurja Strossmayera u Osijeku

3. Dr. sc. Janez Brest, redoviti profesor, Fakulteta za elektrotehniko, računalništvo in informatiko, Univerza v Mariboru, Slovenija

4. Dr. sc. Alfonzo Baumgartner, docent, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek, Sveučilište Josipa Jurja Strossmayera u Osijeku

5. Dr. sc. Emmanuel Karlo Nyarko, docent, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek, Sveučilište Josipa Jurja Strossmayera u Osijeku

Datum obrane doktorske disertacije: 8. ožujka 2017. godine.

Zahvale

Zahvaljujem se svom mentoru prof. dr. sc. Goranu Martinoviću na podršci i vođenju te poticanju na istraživanje tijekom poslijediplomskog studija.

Zahvaljujem se svojim roditeljima, Branku i Ivki, te sestri, Ani, na bezuvjetnoj potpori i razumjevanju tijekom mog cjelokupnog školovanja.

Zahvaljujem se „cimeru“ Bruni Zoriću na produktivnoj suradnji kako u istraživanjima tako i u nastavi.

Sadržaj

Popis slika **iv**

Popis tablica **viii**

Popis algoritama **x**

Popis kratica i oznaka **xii**

1 Uvod **1**

1.1 Motivacija za istraživanje 1

1.2 Ciljevi disertacije 3

1.3 Pregled sadržaja disertacije 3

2 Diferencijalna evolucija (DE) **5**

2.1 Algoritam DE 5

2.2 Početna populacija 7

2.3 Mutacija i križanje 8

2.3.1 Mutacija 8

2.3.2 Križanje 11

2.4 Selekcija naredne generacije 13

2.5 Parametri algoritma 14

2.6 Rukovanje granicama prostora pretrage 17

2.7 Često korišteni algoritmi i postavke parametara 18

2.8 Osvrt na diferencijalnu evoluciju 20

3 Podešavanje parametara diferencijalne evolucije **22**

3.1 Motivacija 22

3.2 Pregled literature 23

3.3 Postupak samopodešavanja parametara skaliranja i križanja DE 26

3.3.1 Opis predloženog postupka 26

3.3.2 Detalji ugradnje 28

3.4 Eksperimentalni rezultati i analiza: Predloženi postupak samopodešavanja faktora skaliranja i stope križanja 30
3.4.1 Postavke eksperimenata 30
3.4.2 Usporedba s nepromjenjivim vrijednostima parametara 31
3.4.3 Usporedba s $DE_{(aDE)}$, $DE_{(SLADE)}$ i $DE_{(SspDE)}$ 35
3.4.4 Utjecaj broja pohranjenih vrijednosti parametara 42
3.4.5 Ponašanje tijekom procesa optimizacije 45
3.5 Osvrt na podešavanje parametara i predloženi postupak 47

4 Inicijalizacija populacije diferencijalne evolucije 50
4.1 Motivacija 50
4.2 Pregled literature 51
4.3 Metoda inicijalizacije populacije DE zasnovana na grupiranju podataka i slučajnim varijablama 52
4.3.1 Kratki uvod u grupiranje podataka i algoritam k-means 53
4.3.2 Primjena grupiranja podataka u evolucijskim algoritmima 54
4.3.3 Opis predložene metode 55
4.3.4 Detalji ugradnje 57
4.3.5 Analiza vremenske složenosti 57
4.4 Eksperimentalni rezultati i analiza: Predložena metoda inicijalizacije populacije DE 58
4.4.1 Postavke eksperimenata 58
4.4.2 Ponašanje upotrijebljenih metoda inicijalizacije populacije 59
4.4.3 Učinkovitost na problemima niske i srednje dimenzionalnosti 61
4.4.4 Utjecaj veličine populacije 65
4.4.5 Učinkovitost na problemima viših dimenzionalnosti 68
4.5 Osvrt na inicijalizaciju populacije i predloženu metodu 71

5 Mutacija diferencijalne evolucije 73
5.1 Motivacija 73
5.2 Pregled literature 74
5.3 Mutacija DE zasnovana na prilagodljivoj turnirskoj selekciji 76
5.3.1 Turnirska selekcija u evolucijskim algoritmima 76
5.3.2 Opis predložene mutacije 77
5.3.3 Detalji ugradnje 80
5.4 Eksperimentalni rezultati i analiza: Predložena mutacija DE 80
5.4.1 Postavke eksperimenata 81
5.4.2 Usporedba s nepromjenjivom veličinom turnira 81
5.4.3 Usporedba s DE/*at*best/1, DE/lbest/1 i rank-DE/rand/1 85

5.4.4 Utjecaj najveće dozvoljene veličine turnira 92
5.4.5 Ponašanje tijekom procesa optimizacije 95
5.5 Osvrt na predloženu mutaciju 97

6 Izgradnja radijalnih mreža diferencijalnom evolucijom 100
6.1 Radijalne mreže 100
6.1.1 Kratki uvod u problem klasifikacije 101
6.1.2 Struktura radijalnih mreža 101
6.1.3 Treniranje RBFNs i izgradnja klasifikacijskih modela 103
6.2 Diferencijalna evolucija u izgradnji radijalnih mreža 104
6.3 Eksperimentalni rezultati i analiza: Izgradnja RBFNs pomoću DE 108
6.3.1 Postavke eksperimenata 108
6.3.2 Učinkovitost i ponašanje standardnog algoritma DE 109
6.3.3 Učinkovitost unaprjeđenih inačica algoritma DE 113
6.4 Osvrt na RBFNs i njihovu izgradnju pomoću DE 120

7 Zaključak i budući rad 123
7.1 Zaključci 123
7.2 Budući rad 126

Literatura 128

Sažetak 143

Abstract 144

Životopis 145

A Skupovi testnih funkcija a
A.1 Standradne testne funkcije . . . . . a
A.2 Funkcije CEC 2014 . . . . . j
A.3 Postupak procjene vremenske složenosti . . . . . k

B Podatkovni skupovi za potrebe klasifikacije l
B.1 Opisi skupova l
B.2 Normalizacija podataka . . . . . n

C Korišteni programski jezici i računalno sklopovlje o

Popis slika

2.1 Skica načina rada algoritma DE. 7
2.2 Primjer geometrijskog značenja mutacije DE. 9
2.3 Primjer svih mogućih vektora razlike (perturbacija). 9
2.4 Primjer binomnog križanja DE. 12
2.5 Primjer eksponencijalnog križanja DE. 12
2.6 Često korišteni (a) operatori i (b) postavke parametara. 19
2.7 Često korišteni algoritmi i pripadajuće postavke parametara. 20

3.1 Pridružene i pohranjene vrijednosti parametara pri predloženom postupku samopodešavanja faktora skaliranja i stope križanja. 26
3.2 Međusobna usporedba različitih postavki faktora skaliranja i stope križanja u smislu (a) prosječnih rangova $\overline{R}$ i (b) prosječnih stopa uspješnosti $\overline{SR}$. 31
3.3 Grafovi konvergencije na nekoliko odabranih standardnih testnih funkcija. Usporedba algoritama s fiksnim postavkama faktora skaliranja i stope križanja te s njihovim samopodešavanjem predloženim postupkom. 34
3.4 Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 10$. Usporedba algoritama s fiksnim postavkama faktora skaliranja i stope križanja te s njihovim samopodešavanjem predloženim postupkom. 34
3.5 Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 30$. Usporedba algoritama s fiksnim postavkama faktora skaliranja i stope križanja te s njihovim samopodešavanja predloženim postupkom. 35
3.6 Grafovi konvergencije na nekoliko odabranih standardnih testnih funkcija. Usporedba predloženog i postupaka podešavanja faktora skaliranja i stope križanja iz literature. 39
3.7 Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 10$. Usporedba predloženog i postupaka podešavanja faktora skaliranja i stope križanja iz literature. 40
3.8 Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 30$. Usporedba predloženog i postupaka podešavanja faktora skaliranja i stope križanja iz literature. 41

3.9 Prosječne greške optimizacije na standardnim testnim funkcijama. Usporedba predloženog postupka pri različitim veličinama popisa r. 43
3.10 Prosječne greške optimizacije na funkcijama CEC 2014 za $d=10$. Usporedba predloženog postupka pri različitim veličina popisa r. 44
3.11 Prosječne greške optimizacije na funkcijama CEC 2014 za $d=30$. Usporedba predloženog postupka pri različitim veličina popisa r. 44
3.12 Grafovi konvergencije na nekoliko odabranih standardnih testnih funkcija. Usporedba predloženog postupka pri različitim veličinama popisa r. 45
3.13 Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 10$. Usporedba predloženog postupka pri različitim veličinama popisa r. 46
3.14 Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 30$. Usporedba predloženog postupka pri različitim veličinama popisa r. 47
3.15 Proces samopodešavanja faktora skaliranja i stope križanja na standardnim testnim funkcijama. 48
3.16 Proces samopodešavanja faktora skaliranja i stope križanja na funkcijama CEC 2014. 49

4.1 Vizualizacija populacija generiranih korištenim metodama za inicijalizaciju populacije. 60
4.2 Grafovi konvergencije za nekoliko odabranih funkcija. Usporedba predložene, standardne i metoda iz literature. 64
4.3 Učinkovitost predložene i ostalih metoda inicijalizacije na nekoliko odabranih funkcija. 68
4.4 Grafovi konvergencije na nekoliko odabranih instanci problema viših dimenzionalnosti. Usporedba predložene, standardne i metoda iz literature. 71

5.1 Vjerojatnosti odabira jedinki u ovisnosti o njenom rangu q u populaciji od 100 jedinki. 77
5.2 Vrijednosti za izračun veličine turnira pridružene članovima populacije pri predloženoj mutaciji. 79
5.3 Povećavanje temeljne veličine za sve turnire. Prikaz razlike između linearnog i rasta po logaritamskoj funkciji. 79
5.4 Učinkovitost uporabe jednake fiksne veličine turnira na standardnim i funkcijama CEC 2014 u smislu (a) prosječnih rangova $\overline{R}$ i (b) prosječnih stopa uspješnosti $\overline{SR}$. 81
5.5 Grafovi konvergencije na nekoliko odabranih standardnih testnih funkcija. Usporedba predložene i mutacija s različitim fiksnim selekcijskim pritiscima odabira baznog vektora. 84

5.6 Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 10$. Usporedba predložene i mutacija s različitim fiksnim selekcijskim pritiscima odabira baznog vektora. 84
5.7 Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 30$. Usporedba predložene i mutacija s različitim fiksnim selekcijskim pritiscima odabira baznog vektora. 85
5.8 Grafovi konvergencije na nekoliko odabranih standardnih testnih funkcija. Usporedba predložene i mutacija iz literature. 89
5.9 Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 10$. Usporedba predložene i mutacija iz literature. 90
5.10 Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 30$. Usporedba predložene i mutacija iz literature. 91
5.11 Prosječne greške optimizacije na standardnim testnim funkcijama za različite najveće dozvoljene veličine turnira k_{max}. 94
5.12 Prosječne greške optimizacije na funkcijama CEC 2014 za $d=10$. Usporedba predložene mutacije pri različitim najvećim dozvoljenim veličinama turnira k_{max}. 94
5.13 Prosječne greške optimizacije na funkcijama CEC 2014 za $d=30$. Usporedba predložene mutacije pri različitim najvećim dozvoljenim veličinama turnira k_{max}. 94
5.14 Grafovi konvergencije na nekoliko odabranih standardnih testnih funkcija. Usporedba predložene mutacije pri različitim najvećim dozvoljenim veličinama turnira k_{max}. 95
5.15 Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 10$. Usporedba predložene mutacije pri različitim najvećim dozvoljenim veličinama turnira k_{max}. 96
5.16 Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 30$. Usporedba predložene mutacije pri različitim najvećim dozvoljenim veličinama turnira k_{max}. 97
5.17 Proces prilagodbe veličine turnira na standardnim testnim funkcijama. . . . 98
5.18 Proces prilagodbe veličine turnira na funkcijama CEC 2014. 99

6.1 Skica klasifikacijskog modela. 101
6.2 Struktura uobičajene radijalne mreže. 102
6.3 Predstavljanje rješenja (a) s kodiranjem težina i (b) bez kodiranja težina pri zadanom broju čvorova u skrivenom sloju. 105
6.4 Predstavljanje rješenja pri nepoznatom broju čvorova u skrivenom sloju. . . 105
6.5 Ponašanje populacije algoritma DE. 112

6.6 Dijagrami pravokutnika za standardnu i predložene unaprjeđenje inačice algoritma DE. 114
6.7 Grafovi konvergencije za standardnu i predložene unaprjeđenje inačice algoritma DE. 116
6.8 Grafovi konvergencije za standardni algoritam i isti s ugrađenim različitim postupcima podešavanja faktora skaliranja i stope križanja. 117
6.9 Proces podešavanja faktora skaliranja i stope križanja predloženim postupkom pri izgradnji RBFNs. 119
6.10 Grafovi konvergencije za algoritme s različitim metodama inicijalizacije populacije. 121

Popis tablica

3.1 Korištene dimenzionalnosti standardnih testnih funkcija. 30
3.2 Rezultati na standardnim testnim funkcijama. Usporedba fiksnih postavki faktora skaliranja i stope križanja te njihovog samopodešavanja predloženim postupkom. 32
3.3 Rezultati na funkcijama CEC 2014 za $d = 10$. Usporedba fiksnih postavki faktora skaliranja i stope križanja te njihovog samopodešavanja predloženim postupkom. 33
3.4 Rezultati na funkcijama CEC 2014 za $d = 30$. Usporedba fiksnih postavki faktora skaliranja i stope križanja te njihovog samopodešavanja predloženim postupkom. 33
3.5 Rezultati na standardnim testnim funkcijama. Usporedba predloženog i postupaka podešavanja faktora skaliranja i stope križanja iz literature. 36
3.6 Rezultati na funkcijama CEC 2014 za $d = 10$. Usporedba predloženog i postupaka podešavanja faktora skaliranja i stope križanja iz literature. 37
3.7 Rezultati na funkcijama CEC 2014 za $d = 30$. Usporedba predloženog i postupaka podešavanja faktora skaliranja i stope križanja iz literature. 38
3.8 Procjena složenosti na standardnim testnim funkcijama za predloženi i postupke podešavanja faktora skaliranja i stope križanja iz literature. 42
3.9 Procjena složenosti na funkcijama CEC 2014 za predloženi i postupke podešavanja faktora skaliranja i stope križanja iz literature. 42
3.10 Prosječni rangovi i stope uspješnosti na standardnim funkcijama u ovisnosti o veličini popisa r. 42
3.11 Prosječni rangovi i stope uspješnosti na funkcijama CEC 2014 za $d = 10$ u ovisnosti o veličini popisa r. 43
3.12 Prosječni rangovi i stope uspješnosti na funkcijama CEC 2014 za $d = 30$ u ovisnosti o veličini popisa r. 43

4.1 Vrijednosti τ korištene za pojedine funkcije i dimenzionalnosti problema. . . 59
4.2 Brzina konvergencija u smislu $\mathrm{TER}_{\mathrm{NFEs}}$ i ukupna vremena izvođenja (u sekundama) na funkcijama $f_1 \sim f_{22}$ za $d = 10$. 61

4.3 Brzina konvergencija u smislu TER_{NFEs} i ukupna vremena izvođenja (u sekundama) na funkcijama $f_1 \sim f_{22}$ za $d=30$. 62
4.4 Brzina konvergencija u smislu TER_{NFEs} i ukupna vremena izvođenja (u sekundama) na funkcijama $f_1 \sim f_{22}$ za $d=50$. 63
4.5 Brzina konvergencija u smislu TER_{NFEs} i ukupna vremena izvođenja (u sekundama) na funkcijama $f_{23} \sim f_{30}$. 63
4.6 Učinkovitost korištenih metoda inicijalizacije u usporedbi s predloženom na funkcijama $f_1 \sim f_{22}$ za $d=10$. 65
4.7 Učinkovitost korištenih metoda inicijalizacije u usporedbi s predloženom na funkcijama $f_1 \sim f_{22}$ za $d=30$. 66
4.8 Učinkovitost korištenih metoda inicijalizacije u usporedbi s predloženom na funkcijama $f_1 \sim f_{22}$ za $d=50$. 67
4.9 Učinkovitost korištenih metoda inicijalizacije u usporedbi s predloženom na funkcijama $f_{23} \sim f_{30}$. 67
4.10 Konvergencija u smislu ukupnog NFEs te ukupna vremena izvođenja (u sekundama) na funkcijama $f_1 \sim f_{22}$ za $d=80$. 69
4.11 Konvergencija u smislu ukupnog NFEs te ukupna vremena izvođenja (u sekundama) na funkcijama $f_1 \sim f_{22}$ za $d=120$. 70

5.1 Rezultati na standardnim testnim funkcijama. Usporedba predložene i mutacija s različitim fiksnim selekcijskim pritiscima odabira baznog vektora. . . . 82
5.2 Rezultati na funkcijama CEC 2014 za $d=10$. Usporedba predložene i mutacija s različitim fiksnim selekcijskim pritiscima odabira baznog vektora. 83
5.3 Usporedba na funkcijama CEC 2014 za $d=30$, Usporedba predložene i mutacija s različitim fiksnim selekcijskim pritiscima odabira baznog vektora. . . . 83
5.4 Rezultati na standardnim testnim funkcijama. Usporedba predložene i mutacija iz literature. 86
5.5 Rezultati na funkcijama CEC 2014 za $d=10$. Usporedba predložene i mutacija iz literature. 87
5.6 Rezultati na funkcijama CEC 2014 za $d=30$. Usporedba predložene i mutacija iz literature. 88
5.7 Procjena složenosti na standardnim testnim funkcijama predložene i mutacija iz literature. 92
5.8 Procjena složenosti na funkcijama CEC 2014 predložene i mutacija iz literature. 92
5.9 Prosječni rangovi i stope uspješnosti na standardnim funkcijama u ovisnosti o najvećoj dozvoljenoj veličini turnira k_{max}. 93
5.10 Prosječni rangovi i stope uspješnosti na funkcijama CEC 2014 za $d = 10$ u ovisnosti o najvećoj dozvoljenoj veličini turnira k_{max}. 93

5.11 Prosječni rangovi i stope uspješnosti na funkcijama CEC 2014 za $d = 30$ u ovisnosti o najvećoj dozvoljenoj veličini turnira k_{max}. 93

6.1 Karakteristike korištenih skupova podataka. 108
6.2 Korišteni parametri pristupa u usporedbi. 109
6.3 Rezultati u smislu MSE za prirodnom inspirirane algoritme u usporedbi. . . 110
6.4 Rezultati u smislu MSE za klasične pristupe u usporedbi. 110
6.5 Prosječne vrijednosti MCR za prirodnom inspirirane algoritme u usporedbi. . 111
6.6 Prosječne vrijednosti MCR za klasične pristupe u usporedbi. 111
6.7 Rezultati u smislu složenosti pronađenih mreža za prirodom inspirirane algoritme u usporedbi. 111
6.8 Rezultati u smislu MSE za standardnu i unaprjeđene inačice algoritma DE. . 114
6.9 Rezultati u smislu MCR za standardnu i unaprjeđene inačice algoritma DE. 115
6.10 Rezultati u smislu složenosti mreža za standardnu i unaprjeđene inačice algoritma DE. 115
6.11 Rezultati za standardni DE i algoritam s predloženim postupkom samopodešavanja parametara na nekoliko odabranih standardnih testnih funkcija za d=500. 118
6.12 Rezultati u smislu MSE za standardni i algoritam s predloženim postupkom samopodešavanja parametara. 118
6.13 Rezultati u smislu složenosti mreža za standardni i algoritam s predloženim postupkom samopodešavanja parametara. 118
6.14 Rezultati u smislu MCR za standardni i algoritam s predloženim postupkom samopodešavanja parametara. 119
6.15 Rezultati u smislu MSE za algoritme s različitim metodama inicijalizacije populacije. 120
6.16 Rezultati u smislu složenosti mreža za algoritme s različitim metodama inicijalizacije populacije. 120

A.1 Pregled korištenih standardnih testnih funkcija. . . . . b
A.2 Pregled funkcija CEC 2014. . . . . j

C.1 Karakteristike računala korištenih za potrebe eksperimentalnih analiza. . . . o

Popis algoritama

2.1 Općeniti nacrt rada DE 6
2.2 Nasumična inicijalizacija populacije 8
2.3 Binomno križanje 12
2.4 Eksponencijalno križanje 12

3.1 Nacrt rada DE s ugrađenim predloženim postupkom samopodešavanja faktora skaliranja i stope križanja 29

4.1 k-means 54
4.2 Predložena metoda za inicijalizaciju populacije DE 55

5.1 Odabir vektora u predloženoj mutaciji 78
5.2 Nacrt rada DE s ugrađenom predloženom mutacijom 80

A.1 Pseudo-kod pomoćnog programa za procjenu vremenske složenosti . . . . . k

Popis kratica i oznaka

$\mathcal{A}_j^F$	Popis prethodno uspješnih vrijednosti faktora skaliranja dodijeljen j-tom članu populacije
$\mathcal{A}_j^{CR}$	Popis prethodno uspješnih vrijednosti stope križanja dodijeljen j-tom članu populacije
$\mathbf{a}^j$	j-ti uzorak iz skupa uzoraka
ABC	Umjetna kolonija pčela
ANNs	Umjetne neuronske mreže
$\mathcal{C}(l, s)$	Cauchyjeva slučajna varijabla s parametrom lokacije l i parametrom skaliranja s
c	Broj čvorova u skrivenom sloju radijalne mreže
c_{max}	Najveći dozvoljeni broj čvorova u skrivenom sloju radijalne mreže
c_{min}	Najmanji dozvoljeni broj čvorova u skrivenom sloju radijalne mreže
CR	Stopa križanja
CR_j	Stopa križanja za j-ti član populacije
d	Dimenzionalnost problema
DE	Diferencijalna evolucija
DE/ρ/δ/ν	Oznaka algoritma DE, gdje ρ ukazuju na način određivanja baznog vektora, δ na broj korištenih vektora razlike u mutaciji, a ν na korišteno križanje
DE/kt/1	Algoritam DE koji koristi predloženu mutaciju
$\text{DE}_{(\text{ASP})}$	Algoritam DE s ugrađenim predloženim postupkom samopodešavanja faktora skaliranja i stope križanja
DE_{C}	Algoritma DE s ugrađenom predloženom metodom inicijalizacije populacije
EAs	Evolucijski algoritmi
Δf	Greška optimizacije
ϕ_i	Radijalna funkcija i-tog čvora skrivenog sloja
F	Faktor skaliranja ili mutacije
f	Funkcija cilja
F_j	Faktor skaliranja za j-ti član populacije

FA	Algoritam krijesnice
FEs	Vrednovanja funkcije cilja
g	Trenutna generacija ili iteracija izvođenja algoritma
GAs	Genetski algoritmi
κ_j	Vrijednost dodijeljena j-tom članu populacije koja određuje veličinu k-turnira za odabir baznog vektora
K	Temeljna veličina za sve turnire
k_{max}	Najveća dozvoljena veličina k-turnira
MCR	Stopa pogrešne klasifikacije
MLPs	Višeslojni perceptroni
MSE	Srednja kvadratna greška
$\mathcal{N}(\mu, \sigma)$	Gaussova slučajna varijabla sa srednjom vrijednosti μ i standardnom devijacijom σ
NP	Veličina populacije
NFEs	Broj vrednovanja funkcije cilja
NFEs_{max}	Maksimalni broj vrednovanja funkcije cilja
$\mathcal{P}$	Populacija DE
p	Dimenzionalnost vektora značajki koji opisuju uzorke
PRNG	Generator pseudo-slučajnih brojeva
PSO	Optimizacija rojem čestica
$\mathcal{Q}$	Skup označenih uzoraka
$\mathcal{Q}_T$	Skup označenih uzoraka za potrebe treniranja
$\mathcal{Q}_V$	Skup označenih uzoraka za potrebe nezavisnog vrednovanja
RBFNs	Radijalne mreže
$\mathcal{S}$	Prostor pretrage
$\mathbf{s}^L$	Donja granica prostora pretrage
$\mathbf{s}^U$	Gornja granica prostora pretrage
SR	Stopa uspješnosti
τ	Vrijednost ciljane greške optimizacije
$\mathbf{t}^j$	j-ti pokusni vektor
TER_{NFEs}	Ukupan broj vrednovanja funkcije cilja koji je bio potreban za dosezanje ciljane greške optimizacije
$\mathcal{U}[0, 1]$	Uniformna slučajna varijabla iz $[0, 1]$
$\mathbf{u}^j$	j-ti mutant ili donor vektor
$\mathbf{v}^j$	j-ti vektor ili član populacije DE
$\mathbf{v}^{j,g}$	j-ti vektor populacije DE u generaciji g

POGLAVLJE 1

Uvod

Optimizacija se nalazi u temeljima brojnih disciplina inženjerstva i znanosti. Stalna poboljšanja tehničkih ili sličnih sustava često predstavljaju zahtjev ili potrebu. U tu svrhu se, kao prvi korak, razvijaju modeli kao preslike sustava. Oni uobičajeno uključuju funkciju cilja (sredstvo vrednovanja), parametre ili varijable te ograničenja sustava. Traženje ili podešavanje parametara modela koji će rezultirati očekivanim svojstvima predstavlja problem (globalne) optimizacije. Složenost takvih problema izravno ovisi o modelima koji općenito postaju sve detaljniji. Primjeri složenih problema globalne optimizacije su dizajn i prilagodba antenskih nizova različitih oblika, izgradnja klasifikacijskih i regresijskih modela na temelju poznatih podataka kao i aproksimacija funkcija, dinamičko upravljanje cijenama i drugo. Takvi problemi često posjeduju mnoga svojstva koja ih čine vrlo zahtjevnim za rješavanje. Dodatna otežavajuća okolnost može biti nemogućnost pristupa detaljima modela ili visoka složenost koja ne pruža mogućnost njegovog učinkovitog iskorištavanja. Sve navedeno vodi do potrebe za metodama optimizacije koje se ne oslanjaju na model, nego samo na njegov izlaz za dani ulaz.

1.1 Motivacija za istraživanje

Mnogi problemi optimizacije predstavljaju probleme kontinuirane ili numeričke optimizacije [23, 92], gdje je cilj minimiziranje ili maksimiziranje dane funkcije cilja čije su nezavisne varijable realne. Formalno, takvi problemi mogu se predstaviti parom $(\mathcal{S}, f)$, gdje je $\mathcal{S} \subseteq \mathbb{R}^d$ prostor pretrage, a $f \colon \mathcal{S} \to \mathbb{R}$ je funkcija cilja. Rješavanje problema numeričke optimizacije zahtijeva pronalazak točke ili vektora $\mathbf{x}^\star = (x_1^\star, \ldots, x_d^\star) \in \mathcal{S}$ takvog da

$$\forall \mathbf{x} \in \mathcal{S} \ : \ f(\mathbf{x}^\star) \leq f(\mathbf{x}) \ . \tag{1.1}$$

Treba primijetiti kako se navedeni opis odnosi na problem minimiziranja, ali zbog $\max f(\mathbf{x}) = -\min(-f(\mathbf{x}))$ analogno se može postaviti i za problem maksimiziranja okretanjem nejednakosti. Moguća svojstva funkcije f kao što su nelinearnost u parametrima, nekonveksnost, multimodalnost, isprekidanost, (djelomično) nepostojanje derivacija ili njihova nepouzdanost mogu činiti traženje globalnog minimuma (općenito, globalnog optimuma) iznimno zahtjevnim i nepraktičnim u smislu potrebnog vremena računanja. Uz to, nerijetko informacije ili detalji modela, odnosno analitički oblik funkcije cilja nije dostupan pa je problem nalik crnoj kutiji zbog čega nije moguće iskoristi nikakva eventualno pogodna svojstava (primjerice, postojanje derivacija) koja bi olakšala njegovo rješavanje.

Nepostojanje derivacija, nedostupnost analitičkog oblika funkcije cilja i uz to (razmjerno) veliki broj parametara zahtijevaju metode optimizacije koje se ne oslanjaju ni na što osim na povratnu vrijednost za dani ulaz funkcije. Značajne predstavnike takve skupine metoda čine evolucijski algoritmi (engl. *evolutionary algorithms*, EAs), paradigme evolucijskog računanja (engl. *evolutionary computing*) [34, 155]. Pod nju kao jednu od paradigmi računalne inteligencije (engl. *computational intelligence*) moguće je svrstati i algoritme inteligencije rojeva (engl. *swarm intelligence algorithms*) obuhvaćajući tako mnoštvo različitih prirodom inspiriranih algoritama. U pogledu numeričke optimizacije i EAs, može se istaknuti diferencijalna evolucija (engl. *differential evolution*, DE) kojom se bavi ova disertacija. Ne uzimajući u obzir brojne različite algoritme inspirirane pčelama, kapljicama vode, gravitacijom i sličnim (za kritički osvrt vidi [76, 120]), postoje razne vrste EAs koji su svoju učinkovitost dokazali pri rješavanju mnogih problema globalne optimizacije. Stoga se nameće pitanje razloga odabira DE. Odgovor na ovo pitanje pruža nekoliko činjenica koje mogu opravdati i poduprijeti izbor navedene. Naime, izvorno je predložena za rješavanje problema numeričke optimizacije. Zbog toga ne zahtijeva nikakve prilagodbe ili uporabu posebno dizajniranih varijacijskih operatora. Iako ovo vrijedi i za neke druge EAs, visoka učinkovitost i pouzdanost DE je dokazana u brojim usporedbama s drukčijim metodama optimizacije, ali i drukčijim EAs (vidi primjerice [22, 28, 103, 124, 136]). Na koncu, jednostavnost osnovnog oblika algoritma predstavlja dobru pretpostavku za moguća daljnja proširenja i unaprjeđenja.

Treba istaknuti kako EAs i slični algoritmi optimizacije ne mogu jamčiti pronalazak optimalnih ili čak dobrih rješenja. Međutim, često su sposobni pronaći rješenja bliska optimalnom unutar razumnog vremena. To se prvenstveno odnosi na broj vrednovanja funkcije cilja koja uz to mogu biti računalno zahtjevna (primjerice ako se radi o nekoj simulaciji). Navedeno je u vezi s ranije istaknutom činjenicom da se EAs ne oslanjaju ni na što osim izlaz modela. S obzirom na prethodne pretpostavke o složenosti i nedostatku analitičkog oblika funkcije cilja, što je sve češća pojava pri rješavanju različitih problema, važnost i značaj razmatrane skupine metoda globalne optimizacije je neupitna.

1.2 Ciljevi disertacije

Iako algoritam DE u osnovnom obliku pruža dobru učinkovitost, prostora za proširenja i unaprjeđenja ne nedostaje. To dokazuju i brojna različita unaprjeđenja predložena u literaturi (vidi primjerice [15, 21, 51, 112, 162, 164]). Treba istaknuti kako neka od tih unaprjeđenja predstavljaju vrlo jednostavna proširenja osnovnog algoritma, ali pružaju veću učinkovitost u usporedbi što ide u prilog DE. S druge strane, složenija unaprjeđenja uobičajeno pružaju i proporcionalno veće učinkovitosti.

U ovoj disertaciji predlažu se unaprjeđenja algoritma DE s ciljem postizanja povećanja učinkovitost pri rješavanju problema numeričke optimizacije bez ikakvih pretpostavki o problemima, što ih čini nalik crnim kutijama (engl. *black-box optimisation*). Prvenstveno je cilj povećanje učinkovitosti, ne samo u odnosu na algoritam u koji su ugrađene, nego i u odnosu na isti algoritam u koji su ugrađena srodna unaprjeđenja dostupna u literaturi. Prijedlozi unaprjeđenja odnose se na tri elementa algoritma koji često predstavljaju bitne čimbenike u pogledu učinkovitosti. To su podešavanje parametara algoritma, odabir ili stvaranje početne populacije te mutacija. Dodatni cilj je ispitati utjecaj predloženih unaprjeđenja na učinkovitost standardnog algoritma DE pri izgradnji klasifikacijski modela predstavljenih radijalnim mrežama. Prije toga nužno je ispitati ponašanje i učinkovitost standardnog algoritma pri traženju prikladnih parametara mreža (izgradnji mreža), što predstavlja složeni problem globalne optimizacije posebno s gledišta broja parametara i računalno zahtjevnog vrednovanja.

Navedeni ciljevi disertacije mogu se prikazati i kroz ispunjavanje sljedećih očekivanih izvornih znanstvenih doprinosa:

1. Postupak samopodešavanja parametara križanja i skaliranja diferencijalne evolucije
2. Metoda za inicijalizaciju populacije diferencijalne evolucije zasnovana na grupiranju podataka i slučajnim varijablama
3. Mutacija u diferencijalnoj evoluciji zasnovana na prilagodljivoj turnirskoj selekciji
4. Eksperimentalno vrednovanje predloženih unaprjeđenja diferencijalne evolucije na problemu izgradnje radijalnih mreža

1.3 Pregled sadržaja disertacije

U poglavlju 2 opisana je i razmatrana diferencijalna evolucija. Detaljno su opisani elementi algoritma, a u prvom redu mutacija i križanje te su sažeto opisani često korišteni operatori istih. Isto tako, posebno su razmatrani parametri algoritma kroz njihovo značenje i preporuke za njihovo postavljanje prema literaturi. Na kraju, predstavljen je pregled literature s

analizom u pogledu često korištenih operatora mutacije i križanja, često korištenih postavki parametara te njihovih kombinacija.

Problem odabira i postavljanja prikladnih vrijednosti parametara algoritma DE opisan je u poglavlju 3 kao motivacija za predlaganje postupka za samopodešavanje podskupa parametara. Pregledom literature obuhvaćeni su i sažeto opisani brojni postupci za ostvarivanje podešavanja parametara algoritma DE tijekom izvođenja. Detaljno je opisan predloženi postupak samopodešavanja faktora skaliranja i stope križanja. Prikazani su rezultati provedene eksperimentalne analize. Između ostalog, predloženi postupak ugrađen u standardni algoritam DE uspoređen je s istim algoritmom pri različitim fiksnim postavkama parametara te s nekoliko postupaka podešavanja parametara iz literature.

Poglavlje 4 bavi se značajem početne populacije za učinkovitost DE i EAs općenito. Kao motivacija za prijedlog metode inicijalizacije populacije DE razmatrani su uobičajeni način odabira početne populacije te mogućnost i učinak uključivanja relativno dobrih rješenja u svrhu povećanja učinkovitosti. Dan je pregled literature kroz koji su sažeto opisane metode koje se oslanjaju na znanje o danom problemu za inicijalizaciju populacije te metode koje to ne čine, a u koje spada i predložena metoda. Ona je detaljno opisana i analizirana u smislu vremenske složenosti. Metoda je uspoređena s uobičajenim pristupom inicijalizacije populacije kao i s dvije metode iz literature.

U poglavlju 5, kao motivacija za predlaganje nove mutacije DE, razmatrana je važnost mutacije prvenstveno s gledišta sposobnosti algoritma za obavljanje istraživanje prostora pretrage i iskorištavanje obećavajućijih pronađenih rješenja. Brojni oblici mutacije koji koriste različite pristupe odabiru sudionika u operatoru mutacije opisane su pod pregledom literature. Također, detaljno je opisana predložena mutacija te su prikazani rezultati provedene eksperimentalne analize. Eksperimentalnom analizom, uz ostalo, obuhvaćena je i usporedba predložene mutacije s nekoliko drugih oblika dostupnih u literaturi.

U poglavlju 6 predstavljene su radijalne mreže kao vrsta umjetnih neuronskih mreža. One su razmatrane u pogledu izgradnje klasifikacijskih modela. Sažeto su opisani neki klasični pristupi traženja prikladnih parametara, te je opisana mogućnost i način primjene DE za izgradnju radijalnih mreža. Dani su rezultati provedene eksperimentalne analize, pri čemu je testirano i analizirano ponašanje i učinkovitost standardnog algoritma DE u usporedbi s nekoliko drukčijih prirodom inspiriranih algoritama te klasičnih pristupa traženja parametara mreže. Također, testirana su i analizirana predložena unaprjeđenja algoritma u pogledu prednosti i nedostataka koje pruža njihova ugradnja pri rješavanju navedenog problema.

Zaključci i moguće smjernice za daljnje istraživanje predstavljeni su u poglavlju 7. Sažeto su razmatrani rezultati postignuti ugradnjom predloženih unaprjeđenja u standardni algoritam. Navedeni su neki nedostaci predloženih unaprjeđenja i ponuđene smjernice za moguće dorade. Također, sažeto su predstavljene moguće smjernice za razvoj kombinacija predloženih unaprjeđenja.

POGLAVLJE 2

Diferencijalna evolucija (DE)

DIFERENCIJALNA evolucija jednostavna je metoda globalne optimizacije predložena 1995. godine od K. Price-a i R. Storna [123]. Iako je zasnovana na populaciji mogućih rješenja nad čijim se članovima primjenjuju varijacijski operatori i selekcija, u odnosu na druge uobičajene EAs, DE se ističe za nju karakterističnom mutacijom. Pri tom, mutacija se koristi razlikama članova trenutne populacije za generiranje perturbacija (od čega joj potječe i ime). Time se gubi potreba za posebnim razdiobama vjerojatnosti u svrhu postizanja perturbacija. Osim toga, navedenim se postiže samostalna prilagodba perturbacija, jer su one ovisne o strukturi populacije koja se mijenja. Bez obzira na jednostavnost (barem u osnovnom obliku), DE je od samog začetka pokazala značajan potencijal za rješavanje problema numeričke optimizacije. Dokazana učinkovitost i svestranost čine DE pouzdanim izborom za pristupanje brojim problemima globalne optimizacije. U ovom poglavlju, DE je razmatrana i detaljno opisana.

2.1 Algoritam DE

Struktura algoritma DE, kako je prikazana algoritmom 2.1, odgovara uobičajenoj strukturi EAs. To znači da se koristi varijacijskim operatorima, koje čine mutacija i križanje, za stvaranje novih rješenja, odnosno uzorkovanje prostora pretrage te selekcijom za vođenje istraživanja prema obećavajućim dijelovima ili regijama tog prostora. Navedeni operatori primjenjuju se nad populacijom (potencijalnih) rješenja koja se u kontekstu DE uobičajeno nazivaju vektorima (drugi često upotrebljavani nazivi za članove populacije EAs su, primjerice, jedinke ili kromosomi). Slijedi sažeti i općeniti opis načina rada algoritma DE, a ključni elementi su dalje u nastavku detaljno opisani i razmatrani.

Populaciju DE može se opisati popisom ili slijedom (engl. *sequence*) $\mathcal{P} = \left(\mathbf{v}^{1,g}, \ldots, \mathbf{v}^{NP,g}\right)$, a svaki element $\mathbf{v}^{j,g} = (v_1^j, \ldots, v_d^j) \in \mathbb{R}^d$ trenutne generacije ili iteracije g je d-dimenzionlani

vektor koji predstavlja moguće rješenje za dani problem. Prema tome, pojedine članove populacije moguće je referencirati preko njihovog indeksa, a njihov redoslijed ne ovisi o njihovoj kvaliteti koju se, bez smanjenja općenitosti, izražava u smislu funkcije cilja $f : \mathcal{S} \subseteq \mathbb{R}^d \to \mathbb{R}$. Populaciju se najčešće inicijalizira nasumično generiranim rješenjima unutar cijelog prostora pretrage koji je ograničen rasponima koje pojedini parametri ili varijable koje opisuju problem mogu poprimiti.

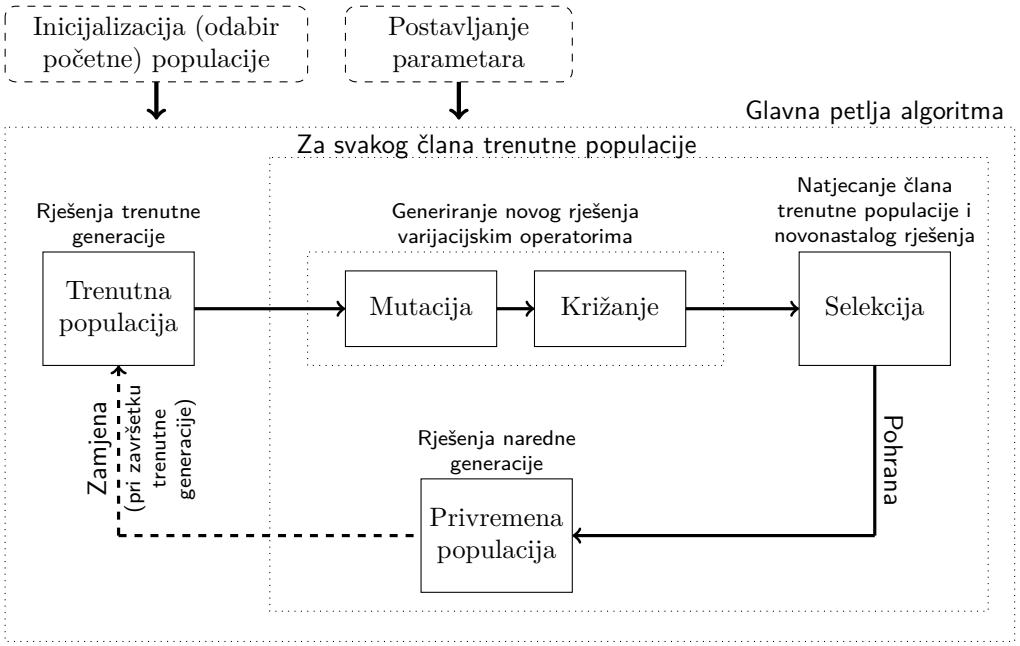
Nakon što je populacija inicijalizirana, prelazi se na generiranje novih rješenja, odnosno vektora. Kroz svaku generaciju, u pravilu se generira NP novih vektora. To se čini pomoću operatora mutacije i križanja. Mutacija je ključna razlika DE i uobičajenih EAs. Naime, osim što je se primjenjuje prije križanja (obrat je uobičajen) ona se ne zasniva na uzrokovanju zadane razdiobe vjerojatnosti. Mutacija se koristi razlikama između vektora populacije za stvaranje perturbacija. Za svaki vektor trenutne populacije $\mathbf{v}^{j,g}$, mutacijom se generira privremeni vektor $\mathbf{u}^{j,g}$ kojeg se potom križa s $\mathbf{v}^{j,g}$ stvarajući novi vektor $\mathbf{t}^{j,g}$. S obzirom da je populacija DE zadane i fiksne veličine te da bi se pretrage vodila prema uvijek boljim rješenjima, nužno je nakon stvaranja novog vektora odrediti hoće li on biti korišten u narednoj generaciji, odnosno iteraciji algoritma. Samo ukoliko je novi vektor $\mathbf{t}^{j,g}$ bolji ili jednak u odnosu na $\mathbf{v}^{j,g}$, on prelazi u narednu generaciju $g+1$, dok se u suprotnom zadržava trenutni član $\mathbf{v}^{j,g}$.

Postupak stvaranja i promicanje novih rješenja ponavlja se do dosezanja postavljenog uvjeta završetka. Često upotrijebljen uvjet završetka je izvršavanje zadanog broja generacija ili slično tome, dosezanje zadanog broja vrednovanja funkcije cilja (engl. *number of function evalutions*, NFEs). Sažeti pregledi uvjeta koje je moguće koristiti u određenim situacijama dani su primjerice u [34, 36, 103].

Algoritam 2.1: Općeniti nacrt rada DE

```
Postavi vrijednosti parametara algoritma;
Inicijaliziraj populaciju P = (v^{1,0}, ..., v^{NP,0});
g := 0;                                          % Trenutna generacija/iteracija
% Glavna petlja
ponavljaj
    % Prolazak kroz sve vektore populacije, redom
    za j := 1, ..., NP čini
        % Stvaranje novih vektora
        Mutacijom stvori mutanta/donora u^{j,g};          % Primjerice prema (2.2)
        Križanjem u^{j,g} i v^{j,g} stvori pokusni vektor t^{j,g};   % Primjerice prema (2.7)
        % Selekcija naredne generacije – usporedba u smislu funkcije cilja
        ako t^{j,g} bolji ili jednak v^{j,g} onda
            v^{j,g+1} := t^{j,g};
        inače
            v^{j,g+1} := v^{j,g};
        kraj
    kraj
    g := g+1;
dok uvjet završetka nije zadovoljen;
```

Prethodno opisani način rada algoritma DE dodatno je ilustriran slikom 2.1. Treba primijetiti kako se algoritam zapravo koristi s dvije populacije. Naime, rješenja, odnosno vektori



Slika 2.1: Skica načina rada algoritma DE.

koji prođu selekciju, pohranjuju se u privremenu populaciju, a na kraju generacije zamjenjuju vektore trenutne populacije. Stoga, nema međudjelovanja vektora trenutne populacije i novonastalih vektora (mutacijom i križanjem).

Diferencijalna evolucija izvorno je predložena za probleme kontinuirane ili numeričke optimizacije [123]. U tom pogledu, pokazala se kao iznimno učinkovita i pouzdana metoda optimizacije. Iako se najčešće upotrebljava za takve vrste problema, njena primjena nije ograničena te je, između ostalog, manje ili više uspješno primijenjena na probleme binarne, diskretne i kombinatorne optimizacije (vidi primjerice [82, 128, 168]). Glavnu prepreku prilikom primjene na nekontinuirane probleme predstavlja učinkovito preslikavanje mutacije u domene različite od realne pa se često zbog jednostavnosti, pribjegava transformacijama iz realne u domenu parametara problema i obratno.

2.2 Početna populacija

Početna populacija određuje gdje se započinje pretraga. Stoga, ona može predstavljati bitan čimbenik u učinkovitosti algoritma. Ipak, populaciju DE se uobičajeno i najčešće inicijalizira potpuno nasumično generiranim rješenjima [3, 9, 110, 164]. Ovo je prikazano algoritmom 2.2. Navedena rješenja, u pravilu se generiraju unutar cijelog prostora pretrage $\mathcal{S} = [\mathbf{s}^L, \mathbf{s}^U] \subset \mathbb{R}^d$, gdje je $\mathbf{s}^L = (s_1^L, \ldots, s_d^L)$ donja granica i $\mathbf{s}^U = (s_1^U, \ldots, s_d^U)$ gornja granica. Mogu se navesti barem dva razloga za odabir nasumične inicijalizacije populacije. Prvi je njena jednostavnost, dok je drugi nedostatak boljeg pristupa. Prvi razlog je očigledan iz navedenog algoritma, a drugi možda isprva nije. Naime, mnogo puta nisu poznate ili dostupne informacije o

unutrašnjosti modela danog problema, te je shodno tome vrlo teško doći do relativno dobrih rješenja. Kvalitetu je moguće izraziti samo u relativnim mjerama, jer se ništa ne zna o krajoliku dobrote (engl. *fitness landscape*). S druge strane, kada postoji znanje o danom problemu, nerijetko ga je moguće iskoristiti za uvođenje dobrih rješenja u početnu populaciju. Uvođenje (relativno) dobrih rješenja može pozitivno utjecati na učinkovitost algoritma, a inicijalizacija populacije DE, u tom pogledu, detaljnije je razmatrana u poglavlju 4.

Algoritam 2.2: Nasumična inicijalizacija populacije

$\mathcal{P} = \left(\mathbf{v}^{1,0}, \ldots, \mathbf{v}^{NP,0}\right)$; % NP je veličina populacije
za $j := 1, \ldots, NP$ **čini**
  za $i := 1, \ldots, d$ **čini**
    $v_i^{j,0} = s_i^L + \mathcal{U}_{j,i}[0,1] \cdot (s_i^U - s_i^L)$; % $\mathcal{U}_{j,i}[0,1]$ je uniformna slučajna varijabla iz $[0,1]$
  kraj
kraj

2.3 Mutacija i križanje

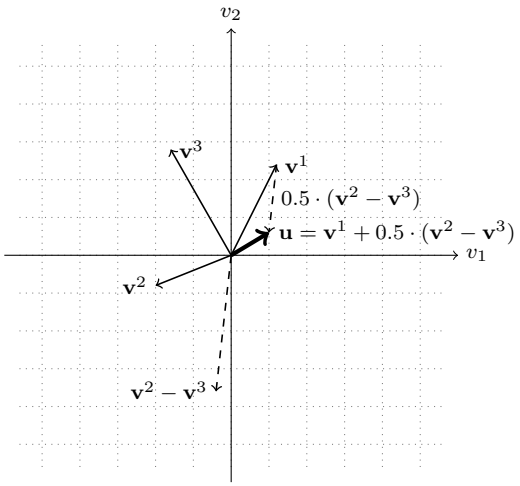
Mutacija i križanje, u pravilu su jedini varijacijski operatori koje se koriste u EAs. Kao što je prethodno navedeno, ni DE nije izuzetak te koristi oba operatora. Mutacijom se stvara privremeno rješenje ili vektor kojeg se potom križa s članom trenutne populacije. Tako se dobiva novo rješenje koje se natječe za prelazak, odnosno za preživljavanje u narednu generaciju.

2.3.1 Mutacija

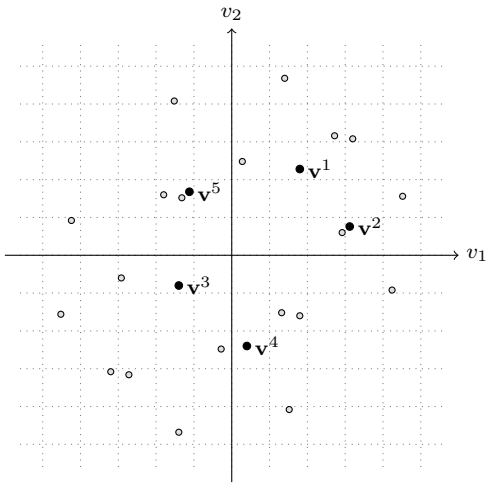
U algoritmu DE, mutacija je prvi, ali i primarni varijacijski operator kojeg se primjenjuje nad vektorima trenutne populacije. Važno svojstvo mutacije DE je što se ne oslanja na unaprijed zadanu razdiobu vjerojatnosti, nego na prostorni raspored vektora trenutne populacije. Tako udaljenosti između pojedinih vektora određuju intenzitet mutacije (engl. *mutation step-size*). Kako populacija konvergira prema jednoj točki, tako se i intenzitet mutacije smanjuje, jer upravo razlike vektora predstavljaju intenzitet mutacije. Za svaki član trenutne populacije ili ciljni vektor (engl. *target vector*) $\mathbf{v}^j$, mutacijom se generira novo rješenje. Točnije, na temelju drugih odabranih vektora trenutne populacije, nastaje novo rješenje, odnosno mutant ili donor vektor kao njihova linearna kombinacija

$$\mathbf{u}^j = \mathbf{v}^b + \sum_{l=1}^{\delta} F_{j,l} \cdot (\mathbf{v}^{q(l)} - \mathbf{v}^{q(l+1)}), \tag{2.1}$$

gdje su $\mathbf{v}^b$ te $\mathbf{v}^{q(l)}$ i $\mathbf{v}^{q(l+1)}$ za $l = 1, \ldots, \delta$ vektori trenutne populacije. Vektor $\mathbf{v}^b$ predstavlja bazni vektor (engl. *base vector*) kojeg se mijenja ili perturbira na temelju vektora razlike (engl. *difference vectors*) $\mathbf{v}^{q(l)} - \mathbf{v}^{q(l+1)}$, $l = 1, \ldots, \delta$. Ove razlike, njih δ, određuju intenzitet



Slika 2.2: Primjer geometrijskog značenja mutacije DE.



Slika 2.3: Primjer svih mogućih vektora razlike (perturbacija).

mutacije, a dodatno su ponderirane parametrom faktora skaliranja (engl. *scale factor*) $F_{j,l} \in [0, +\infty\rangle$. Nije teško zaključiti da odabir vektora koji će sudjelovati u mutaciji određuje gdje će nastati mutant. U tom pogledu, ključan je odabir baznog vektora, jer će mutant biti generiran u njegovoj okolini. Geometrijsko značenje mutacije DE ilustrirano je primjerom u $\mathbb{R}^2$ uz $F = 0.5$ prikazanim slikom 2.2.

S obzirom da se za perturbacije koriste vektori razlike, njihov broj je ograničen i ovisi o veličini populacije. Za razliku od baznog vektora, vektore koji će činiti razlike, najčešće se odabire nasumično. Broj mogućih perturbacija za populaciju veličine NP i uporabu δ vektora razlike, prema Joshi i Sanderson [57], jednak je $\binom{NP}{2\cdot\delta} \cdot 2 \cdot \delta!$. Ograničeni broj mogućih perturbacija uglavnom ne predstavlja nikakav problem u smislu sposobnosti algoritma da istražuje prostor pretrage, osim eventualno u slučaju vrlo malih populacija kada može doći do pojave stagnacije [67] i kada daljnje istraživanje nije moguće. Nadalje, zbroj svih mogućih vektora razlike odnosno perturbacija bit će jednak nuli, jer za jedan odabir jednako je vjerojatan odabir u obrnutom redoslijedu (uz pretpostavku nasumičnog odabira svih vektora) [103]. Iako uporaba većeg broja razlika pri perturbiranju rezultira većom sposobnosti istraživanja, najčešće je $\delta = 1$ ili $\delta = 2$. Primjerom na slici 2.3 ilustrirani su svi mogući vektori razlike ili perturbacije pri nasumičnom odabiru za populaciju od pet vektora.

S obzirom na važnost, u literaturi je predloženo mnoštvo različitih operatora mutacije (za pregled značajnijih, vidi primjerice [24, 26, 91]). Njih je u pravilu moguće poopćiti s (2.1), a razlikuju se u načinu odabira vektora koji sudjeluju u njoj. Različiti operatori mutacije u pravilu se nazivaju s ρ/δ, gdje ρ označava način odabira ili izračuna baznog vektora, a δ broj vektora razlike koji se koriste za njegovo perturbiranje. Uobičajeni operatori mutacije DE su

- rand/1
$$\mathbf{u}^j = \mathbf{v}^{r1} + F_j \cdot (\mathbf{v}^{r2} - \mathbf{v}^{r3}), \tag{2.2}$$
- rand/2
$$\mathbf{u}^j = \mathbf{v}^{r1} + F_j \cdot (\mathbf{v}^{r2} - \mathbf{v}^{r3}) + F_j \cdot (\mathbf{v}^{r4} - \mathbf{v}^{r5}), \tag{2.3}$$
- best/1
$$\mathbf{u}^j = \mathbf{v}^{bst} + F_j \cdot (\mathbf{v}^{r1} - \mathbf{v}^{r2}), \tag{2.4}$$
- best/2
$$\mathbf{u}^j = \mathbf{v}^{bst} + F_j \cdot (\mathbf{v}^{r1} - \mathbf{v}^{r2}) + F_j \cdot (\mathbf{v}^{r3} - \mathbf{v}^{r4}), \tag{2.5}$$
- target-to-best/1 (ili current-to-best/1)
$$\mathbf{u}^j = \mathbf{v}^j + F_j \cdot (\mathbf{v}^{bst} - \mathbf{v}^j) + F_j \cdot (\mathbf{v}^{r1} - \mathbf{v}^{r2}), \tag{2.6}$$

gdje je *bst* indeks najboljeg člana trenutne populacije, a $r1, r2, r3, r4, r5 \in \{1, \ldots, NP\}$ su nasumično odabrani indeksi tako da $j \neq r1 \neq r2 \neq r3 \neq r4 \neq r5$, dok su u pravilu F_j fiksni i jednaki za sve vektore populacije. Mutacija rand/1 jedna je od dviju izvorno predloženih mutacija [123, 124]. Zbog nasumičnog odabira baznog vektora omogućuje opsežno istraživanje prostora pretrage, što za posljedicu ima relativno sporu konvergenciju koja uglavnom ipak ne predstavlja ograničavajući čimbenik. Operator rand/1 pokazao se učinkovitim u rješavanju niza problema globalne optimizacije (vidi primjerice [65, 82, 159, 167]) te je jedan od najčešće upotrebljavanih. Dodatni vektor razlike u mutaciji rand/2 omogućuje opsežnije istraživanje prostora pretrage i pospješuje održavanje raznolikosti populacije, ali uz sporiju konvergenciju. Za razliku od prethodne dvije, mutacije best/1 i best/2 [102, 121] za bazni vektor uvijek odabiru najbolji vektor trenutne populacije. Zbog toga ispoljavaju vrlo brzu konvergenciju, što isto tako može dovesti do brzog gubitka raznolikosti populacije i time do preuranjene konvergencije. Ovo je manje izraženo u slučaju mutacije best/2 zbog uporabe dva vektora razlike, a koja je posebno istaknuta u [124] kao vrlo korisna. Ipak, prema Mezura-Montes et al. [85], best/1 pokazala se u usporedbi s nekoliko drugih operatora mutacije posebno učinkovitom na unimodalnim i razdvojivim problemima. Značajno drukčiji pristup određivanju baznog vektora koristi se u mutaciji target-to-best/1, gdje se baznim vektorom smatra $\mathbf{v}^j + F_j \cdot (\mathbf{v}^{bst} - \mathbf{v}^j)$, a koji za $0 \leq F_j \leq 1$ predstavlja težinsku srednju vrijednost ili konveksnu kombinaciju $\mathbf{v}^j$ i $\mathbf{v}^{bst}$, odnosno točku na liniji koja ih spaja. Slično kao kod prethodna dva operatora, zbog uporabe najboljeg pronađenog rješenja, ova mutacija može ostvariti brzu konvergenciju, ali isto tako može doći do brzog gubitka raznolikosti populacije i zaglavljivanja u lokalnom optimumu [54, 162]. Ovo podupire usporedba i analiza prikazana u [85], gdje je ova mutacija korištena bez naknadnog križanja.

Iako postoje još neki operatori mutacije DE, prema Das et al. [26], navedeni predstavljaju najčešće korištene, ali vrijedi sažeto istaknuti još nekolicinu operatora. Mutacija target-to-rand/1 (ili current-to-rand/1) [85, 104] razlikuje se u odnosu na target-to-best/1 što koristi nasumično odabrani vektor umjesto najboljeg vektora trenutne populacije. Kao što je slučaj s target-to-best/1, nekada se koristi u algoritmu bez naknadne uporabe operatora križanja (vidi primjerice [91, 97, 104]), jer se smatra da je bazni vektor nastao križanjem (aritmetičko križanje – težinska srednja vrijednost). Nadalje, Zhang i Sanderson [162] predložili su izmjenu mutacije target-to-best/1, gdje se umjesto najboljeg vektora trenutne populacije nasumično odabire jedan od zadanog postotka prvih po kvaliteti. Ovim je smanjena opasnost zaglavljivanja u lokalnom optimumu uz zadržavanje relativno brze konvergencije. Vrlo sličnu izmjenu iste mutacije, predložili su Islam et al. [54], gdje se umjesto najboljeg vektora cijele populacije odabire najbolji od zadanog postotka nasumično odabranih vektora (može se promatrati kao turnirska selekcija bez zamjene). U obje varijante, postotak populacije koji se uzima u obzir pri odabiru predstavlja dodatni parametar algoritma. Fan i Lampinen [40] predložili su trigonometrijski operator mutacije koji uključuje i kvalitetu odabranih vektora. Kao pri mutaciji rand/1, nasumično se odabire tri različita (po indeksima) vektora populacije. Bazni vektor čini njihov centar, a perturbira ga se s tri razlike koje su ponderirane odgovarajućim razlikama njihove kvalitete čime se ovaj operator ne oslanja na faktor skaliranja F. Ova mutacija se pokazala korisnom pri traženju parametara jedne vrste umjetne neuronske mreže [100]. U poglavlju 5 je dalje razmatrana važnost mutacije DE te je dan pregled nekih drukčijih pristupa mutaciji, prvenstveno odabiru baznog vektora.

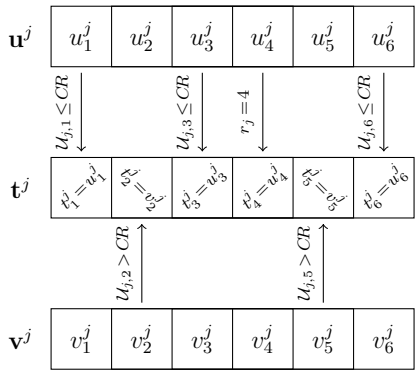
2.3.2 Križanje

Nakon što se mutacijom generira mutant ili donor vektor $\mathbf{u}^j$, dolazi do križanja njega i ciljnog vektora $\mathbf{v}^j$. Rezultat križanja je pokusni vektor (engl. *trial vector*) $\mathbf{t}^j$, a nastaje miješanjem komponenti vektora $\mathbf{u}^j$ i $\mathbf{v}^j$. Mora se napomenuti da pri miješanju ne dolazi do izmjena komponenti ili parametara vektora. Bitna uloga operatora križanje je povećanje raznolikosti novonastalih rješenja, odnosno pokusnih vektora [123].

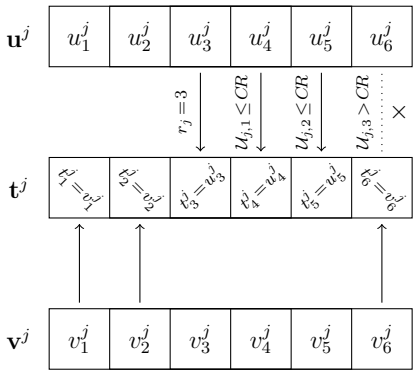
Križanje u DE obavlja se uobičajeno na jedan od dva načina [24, 26, 103]. Najčešće upotrijebljeno križanje DE je uniformno ili binomno (engl. *binomial crossover*) [91, 122] prema kojem je pokusni vektor

$$t_i^j = \begin{cases} u_i^j, & \text{ako } \mathcal{U}_{j,i}[0,1] \leq CR_j \ \vee \ i = r_j, \\ v_i^j, & \text{u suprotnom} \end{cases}, \quad i = 1, \ldots, d, \tag{2.7}$$

gdje je $\mathcal{U}_{j,i}[0,1]$ uniformna slučajna varijabla iz $[0,1]$, r_j je nasumično odabran broj iz skupa $\{1, \ldots, d\}$, a parametar $CR_j \in [0,1]$ je stopa križanja. Vrijednost parametra CR_j određuje koliki dio komponenti će biti preuzet ili naslijeđen iz mutanta, dok r_j osigurava da to bude



Slika 2.4: Primjer binomnog križanja DE.



Slika 2.5: Primjer eksponencijalnog križanja DE.

barem jedna komponenta čime se izbjegava dupliciranje ciljnog vektora $\mathbf{v}^j$. Način generiranja pokusnog vektora $\mathbf{t}^j$ za dani ciljni vektor $\mathbf{v}^j$ i odgovarajući mutant $\mathbf{u}^j$ ovim operatorom, dodatno je prikazan algoritmom 2.3 te ilustriran slikom 2.4 pomoću primjera. Prema navedenom, jasno je da se provode nezavisne usporedbe, slučajnih varijabli i zadane stope križanja, koje mogu imati jedan od dvaju ishoda i o kojima ovisi odakle će potjecati pojedine komponente pokusnog vektora. Tako broj komponenti preuzetih iz mutanta približno odgovara binomnoj razdiobi [24, 34, 103], odakle i potječe ime operatora (nekad ga se naziva i binarnim križanjem, vidi primjerice [41, 91]).

Algoritam 2.3: Binomno križanje

```
Postavi r_j na nasumično odabran broj iz {1, ..., d};
za i := 1, ..., d čini
    ako U_{j,i}[0,1] ≤ CR_j ∨ r_j = i onda
        t_i^j := u_i^j;
    inače
        t_i^j := v_i^j;
    kraj
kraj
```

Algoritam 2.4: Eksponencijalno križanje

```
Postavi r_j na nasumično odabran broj iz {1, ..., d};
i := r_j;
ponavljaj
    t_i^j := u_i^j;
    i := (i + 1) mod d;
dok U_{j,i}[0,1] < CR_j ∧ i ≠ r_j;
dok i ≠ r_j ponavljaj
    t_i^j := v_i^j;
    i := (i + 1) mod d;
kraj
```

Drugo uobičajeno i originalno predloženo [123] križanje DE je eksponencijalno (engl. *exponential crossover*) čiji je način rada, odnosno način generiranja pokusnog vektora $\mathbf{t}^j$ prikazan algoritmom 2.4. Iz istog razloga kao kod binomnog križanja, barem jedna komponenta će biti preuzeta iz mutanta, a r_j određuje koja. Koliko će uzastopnih komponenti koje slijede r_j-tu biti dodatno preuzeto, ovisi o vrijednosti parametra CR_j, a prema vektorima se odnosi kao cikličkim nizovima što se postiže uporabom operacije modulo (označena s mod). Kao što ime ovog operatora ukazuje, broj komponenti preuzetih iz mutanta odgovara eksponencijalnoj razdiobi [103]. Osim toga, prema Qin et al. [104], Qing [107] eksponencijalno križanje ima učinak kao i križanje s dvije točke presjeka [križanje uobičajeno korišteno u genetskim algoritmima (engl. *genetic algorithms*, GAs), vidi primjerice [34]], što je vidljivo i u primjeru sa slike 2.5.

Treba napomenuti kako je u pravilu parametar $CR_j = CR$ za $j = 1, \ldots, NP$ (fiksan i jednak za sve). Osim toga, binomno križanje mnogo je češće korišteno u odnosu na eksponencijalno. Međutim, teško je tvrditi je li je jedno u odnosu na drugo učinkovitije, jer primjerice, rezultati usporedbe u [85] ukazuju na prednost binomnog što se kosi s rezultatima u [107], gdje je pokazana prednost eksponencijalnog. Bez obzira na prethodno navedeno, križanje ima značajan utjecaj na ponašanje i učinkovitost algoritma DE [73] iako mu se za razliku od mutacije ne pridaje prevelika važnost. Štoviše, s obzirom da se njime miješaju komponente ciljnog vektora i mutanta, može ga se promatrati i kao mutaciju ciljnog vektora, gdje je vjerojatnost mutacije izravno povezana s parametrom CR [103, 158]. Bitno je istaknuti da navedena dva operatora predstavljaju samo najčešće korištene, te da algoritam nije ograničen na njih. U literaturi je moguće pronaći i prijedloge izmjena opisanih ili prijedloge uporabe drukčijih (nerijetko složenijih) operatora križanja (vidi primjerice [50, 73, 147]).

2.4 Selekcija naredne generacije

U algoritmu DE, koristi se specifičan oblik selekcije (isti ili sličan oblik moguće je naći i u nekim drugim algoritmima) s ciljem određivanja hoće li novonastali pokusni ili odgovarajući ciljni vektor prijeći u narednu generaciju. Naime, pokusni vektor (potomak) se natječe za opstanak samo s odgovarajućim ciljnim vektorom (roditeljom). To znači kako se j-ti pokusni vektor iz privremene populacije natječe s j-tim ciljnim iz trenutne populacije pa je, bez smanjivanja općenitosti, j-ti vektor naredne generacije ili iteracije $g + 1$

$$\mathbf{v}^{j,g+1} = \begin{cases} \mathbf{t}^{j,g} & \text{ako } f(\mathbf{t}^{j,g}) \leq f(\mathbf{v}^{j,g}), \\ \mathbf{v}^{j,g} & \text{u suprotnom} \end{cases} . \tag{2.8}$$

Ovakav način selekcije ostvaruje blag selekcijski pritisak što pogoduje čuvanju raznolikosti populacije i povećava otpornost na stagnaciju [103]. Nadalje, prema Das et al. [26] mogućnost prelaska pokusnih vektora koji su jednaki (u smislu funkcije cilja) odgovarajućim ciljnim vektorima pomaže pri istraživanju ravnih dijelova krajolika dobrote. Ipak, može se susresti i uporaba stroge nejednakosti u (2.8) (korišteno primjerice u [15, 93, 106, 132, 162]), ali to nije čest slučaj i može se pretpostaviti da su razlike u učinkovitosti algoritma pri jednom ili drugom uglavnom zanemarive.

Kao što je i ranije spomenuto, postoji jasna granica između vektora trenutne i naredne generacije te nema njihovog međudjelovanja. Međutim, postoji i asinkrona ili dinamička inačica algoritma DE [106], gdje ne postoji podjela na generacije. Ovo se postiže izravnom zamjenom ciljnog vektora odgovarajućim pokusnim vektorom odmah po njegovom nastanku ako je bolji ili jednak u smislu funkcije cilja. Stoga se ovakav algoritam koristi samo jednom populacijom vektora što smanjuje prostornu složenost, ali onemogućuje njegove paralelne

izvedbe. Bez obzira na to, u oba slučaja se radi o elitističkom obliku selekcije, gdje najbolje pronađeno rješenje ne može biti izgubljeno iz populacije. Isto tako, održava se kvaliteta rješenja na svim indeksima populacije. Prema tome, trenutna populacija nije nikad lošija (u smislu kvalitete njenih članova) u odnosu na njena prethodna stanja.

Zanimljivo je da se u pravilu uvijek koristi navedeni oblik selekcije. Ovo se može pripisati njenoj jednostavnosti te malobrojnim istraživanjima u pogledu drukčijih oblika selekcije. Neprihvaćenost drukčijih načina selekcije može se obrazložiti i često većim složenostima ugradnje koje obično za sobom povlače i veće vremenske složenosti (vidi primjerice [39, 75]) te nedostatkom opsežnih usporedbi i analiza.

2.5 Parametri algoritma

Standardni oblik algoritma DE, zahtijeva postavljanje tri parametra. Oni redom predstavljaju veličinu populacije NP, faktor skaliranja ili mutacije F te stopu križanja CR. Uobičajeno su EAs i slični algoritmi osjetljivi na postavke svojih parametara, a ni DE ne predstavlja izuzetak u tom pogledu. Stoga, kako bi se postigla zadovoljavajuća učinkovitost algoritma nužno je pronaći i koristiti prikladne vrijednosti parametra. Svaki od navedenih parametara na određeni način utječe na pretragu, odnosno proces optimizacije. Iako se isprva smatralo kako je algoritam DE relativno otporan na različite postavke parametara [102, 124], to se kasnije ispostavilo ne (sasvim) točnim [44].

Općenito, u mutaciji sudjeluje $2 \cdot \delta + 1$ vektora. Osim što su međusobno različiti, oni su u pravilu različiti od danog ciljnog vektora. Ovdje treba istaknuti da se radi o razlikama u poziciji ili indeksima u populaciji, jer se u njoj može nalaziti više jednakih vektora po komponentama. Iz navedenog se može zaključiti kako je najmanja moguća veličina populacije pri primjeni mutacije koja koristi δ vektora razlike jednaka $2 \cdot (\delta + 1)$. S druge strane, gornja granica postoji samo u praktičkom smislu. Algoritam DE oslanja se na razlike vektora koje predstavljaju perturbacije pri mutaciji i čiji je broj ograničen te izravno ovisi o veličini populacije. U tom pogledu veličina populacije ima bitnu ulogu. Međutim, nisu sve perturbacije ili svi mogući vektori razlike korisni i njihov broj ovisi o prostornom rasporedu članova populacije i krajoliku dobrote. Povećavanjem veličine populacije smanjuje se kvocijent broja korisnih perturbacija i onih koji to nisu što rezultira sporijom konvergencijom [57]. Nasuprot tome, male populacije ne raspolažu dovoljnom raznolikošću vektora razlike što može dovesti do preuranjene konvergencije ili pak stagnacije pretrage. Navedeno treba uzeti u obzir pri postavljanju veličine populacije NP. U literaturi se može vidjeti korištenje različitih vrijednosti NP, kao i različite preporuke i rasponi unutar kojih je treba tražiti. Primjerice, Price et al. [103], Storn i Price [124] navode $[5 \cdot d, 10 \cdot d]$ kao raspon unutar kojeg bi se trebale nalaziti dobre vrijednosti NP, gdje je d dimenzionalnost prostora pretrage odnosno problema. Drukčiji raspon navode Gämperle et al. [44], prema kojima se prikladne veličine

populacije nalaze unutar $[3 \cdot d, 8 \cdot d]$. Storn [121, 122] navodi $NP = 10 \cdot d$ kao dobar izbor za različite primjene. Prema Rönkkönen et al. [116], ovisno o vrsti i svojstvima problema prikladna veličina populacija može se nalaziti unutar $[2 \cdot d, 40 \cdot d]$, a dalje navode kako bi za rješavanje složenih funkcija za $d = 10$ i $d = 30$ veličina trebala biti između 200 i 600. Na temelju navedenog može se zaključiti kako treba NP postaviti prema dimenzionalnosti danog problema, ali ipak vrlo često se veličina populacije postavlja na 100 ili 50 bez posebnih obrazloženja, dok su Neri i Tirronen [90] pokazali kako se za $NP < d$ može dobiti dobra učinkovitost algoritma. Opsežan pregled vezan za postavljanje veličine populacije DE kao i analiza njenog utjecaja na nekoliko unaprjeđenih varijanti algoritma dani su u [101].

S obzirom na važnost mutacije u DE, odabir i postavljanje faktora skaliranja može značajno utjecati na ponašanje i učinkovitost algoritma. Nije teško zaključiti kako vrijednost parametra F ima bitnu ulogu u održavanju raznolikosti populacije. Analizu u tom pogledu napravila je Zaharie [157] i došla do najmanje vrijednosti koja je nužna za izbjegavanje preuranjene konvergencije, održavanjem približno jednake varijance populacije. Osim toga, on (uz prostorni raspored vektora) utječe na veličinu okoline unutar koje može biti proizveden mutant u odnosu na odabrani bazni vektor. Općenito, F može poprimiti bilo koju realnu vrijednost. Međutim, $F = 0$ nema smisla, jer će mutant biti preslika baznog vektora. Slično tome, negativne vrijednosti su beskorisne ako se vektore koji će činiti razlike uvijek nasumično odabire iz populacije. Prema Price et al. [103], Neri i Tirronen [91], vrijednosti veće od 1 gotovo nikad nisu potrebne pa se stoga efektivnim rasponom za faktor skaliranja može smatrati $\langle 0, 1]$. Međutim, treba obratiti pozornost na $F = 1$, jer primjerice, u slučaju mutacije rand/1, broj mogućih mutanta je dvostruko manji [67, 103], dok se u istom slučaju mutacija target-to-best/1 svodi na mutaciju best/1. Kao što je slučaj s veličinom populacije, u literaturi postoji mnoštvo preporuka za postavljanje parametra F. Nerijetko se radi o preporukama koje se (uvelike) razlikuju. Među prvima, Price [102] navodi raspon $\langle 0, 1.2]$, dok u gotovo isto vrijeme Storn [121] navodi $[0.5, 1]$ kao raspon unutar kojeg bi se trebalo tražiti prikladnu vrijednost za parametar F. Nedugo zatim Storn i Price [124] navode da je $F < 0.4$ i $F > 1$ rijetko učinkovito, a za prvi izbor preporučuju vrijednost 0.5. Blago veću vrijednost ($F = 0.6$) preporučuju Gämperle et al. [44] i zaključuju kako vrijednosti veće od 1 usporavaju konvergenciju. Prema Zaharie [157], faktor skaliranja, uz zadanu veličinu populacije NP i stopu križanja CR, ne bi trebao biti manji od $\sqrt{(1 - CR/2)/NP}$. Treba napomenuti kako se ovo odnosi na teoriju i izmijenjeni oblik algoritma, ali što ne umanjuje značaj navedenog rezultata. S druge strane, eksperimentalno je došla do zaključka da za $NP = 50$ i $CR = 0.2$, faktor skaliranja treba biti barem 0.3. Nadalje, Rönkkönen et al. [116] tvrde kako su dobre vrijednosti faktora skaliranja uobičajeno unutar $\langle 0.4, 0.95]$ te preporučuju $F = 0.9$. Iz navedenog je teško izvući neki zaključak, osim što faktor skaliranja ne bi trebao biti vrlo mala vrijednost (nije pravilo [103]) kao ni vrijednost veća od 1. Međutim, u literaturi se može vrlo često susresti $F = 0.5$ bez navođenja nekog opravdanja za taj izbor.

Iako se križanje u DE ne smatra jednako bitnim kao mutacija, njegov značaj ne može biti zanemaren. Najčešće se koristi binomno, dok je uporaba eksponencijalnog križanja mnogo manje zastupljena. O stopi križanja ovisi broj mogućih pokusnih vektora i time raznolikost populacije (ne uzimajući u obzir njen prostorni raspored i faktor skaliranja) [67, 103]. Efektivni raspon za vrijednosti CR je $[0, 1]$, a o njoj izravno ovisi razlika novonastalog pokusnog vektora i odgovarajućeg ciljnog. Tako će razlika pri $CR = 0$ biti u jednoj nasumično odabranoj komponenti. S druge strane, pri $CR = 1$ razlika će biti u svih d komponenti pa pokusni vektora predstavlja mutanta čime se zapravo isključuje križanje iz algoritma. Oba granična slučaja mogu se, prema Price et al. [103], opravdati kako slijedi. Vrijednost jednaka ili vrlo bliska donjoj granici korisna je ako je dani problem razdvojiv, jer je u tom slučaju moguća nezavisna optimizacije po parametrima što odgovara vrlo malim vrijednostima CR zbog malih izmjena ciljnog vektora. Naprotiv, kada postoji zavisnost među parametrima samo je vrijednost jednaka 1 ili vrlo bliska njoj prikladna, jer je potrebno mijenjati sve ili većinu parametra istovremeno kako bi se održala odgovarajuća učinkovitost pretrage (samo je pri $CR = 1$ algoritam rotacijski invarijantan). Prema tome, vrijednost parametra CR može utjecati na brzinu konvergencije. Treba napomenuti da navedeno ne isključuje uporabu vrijednosti različitih od graničnih. Primjerice, Lampinen i Zelinka [67] došli su do rezultata prema kojem je broj mogućih pokusnih vektora osjetno manji pri graničnim vrijednostima, nego pri $0 < CR < 1$. U literaturi je moguće pronaći razne preporuke za postavljanje tog parametra. Isprva, i za eksponencijalno križanje, Storn [121] navodi kako vrijednost stope križanja treba biti značajno manja od 1, a u slučaju nezadovoljavajuće učinkovitosti treba je odabrati iz $[0.8, 1]$. Price [102] u isto vrijeme, ali za binomno križanje pokazuje da algoritam jednako dobro radi za $CR = 0$ i $CR = 1$. Kasnije, ponovno za binomno križanje, Storn i Price [124] navode vrijednost 0.1 kao dobar početni izbor, a vrijednosti 0.9 i 1 za postizanje brže konvergencije. Međutim, Gämperle et al. [44] navode kako su prikladne vrijednosti unutar $\langle 0.3, 0.9\rangle$, a dalje tvrde kako vrijednosti iz $[0.9, 1]$ mogu usporiti ili dovesti do preuranjene konvergencije. Rönkkönen et al. [116] tvrde da su najprikladnije vrijednosti iz $\langle 0, 0.2\rangle$ ako je problem razdvojiv, a preporučuju vrijednost 0.9 za slučaj zavisnih parametara te napominju kako pri $CR = 1$ može doći do stagnacije zbog, kao što je prethodno navedeno, manjeg broja mogućih pokusnih vektora. Mezura-Montes et al. [85] navode kako je algoritam izrazito osjetljiv na vrijednosti ovog parametra, te su ga za potrebe usporedbe raznih inačica algoritma DE, za razliku od preostalih, određivali za svaku funkciju i inačicu algoritma zasebno. Usporedbu binomnog i eksponencijalnog križanja i pripadajuću analizu je napravila Zaharie [158] te je došla do ovisnosti matematičkog očekivanja broja komponenti preuzetih iz mutanta i stope križanja. Naime, ta ovisnost je u slučaju binomnog križanja dana kao $CR \cdot (d - 1) + 1$, a u slučaju eksponencijalnog kao $(1 - CR^d)/(1 - CR)$, gdje je $CR \in \langle 0, 1\rangle$. Prema ovom, da bi se postigla slična razina miješanja komponenti, potrebno je koristiti značajno veće vrijednosti CR za eksponencijalno u odnosu na binomno križanje.

Sličnu analizu napravili su Lin et al. [73], a na temelju usporedbe zaključili da su općenito male razlike u učinkovitosti algoritma s jednim ili drugim operatorom križanja. Osim toga, potvrđuju prethodno navodeći da su male vrijednosti prikladne za razdvojive probleme, dok su one bliske 1 nužne u suprotnom. Iz svega navedenog može se primijetiti kako su male vrijednosti parametra prikladne (samo) ukoliko je problem razdvojiv, a inače treba koristiti vrijednosti bliske 1. Kako svojstvo razdvojivosti nije uobičajeno (barem ne u svim parametrima), ne iznenađuje što se u literaturi često koristi $CR = 0.9$.

Važno je naglasiti da su navedeni parametri DE međusobno ovisni (jasno se ogleda i u analizi danoj u [157]). Stoga, promjena vrijednosti jednog u pravilu povlači za sobom potrebu za promjenama preostalih. Nadalje, na temelju svega prethodno navedenog, teško je doći do nekog pravog zaključka kako postaviti sva tri parametra za ostvarivanje zadovoljavajuće učinkovitosti pri rješavanju danog problema. U poglavlju 3 je dalje razmatran ovaj problem te je dan pregled različitih postupaka koji pokušavaju riješiti navedeni samostalnim podešavanjem parametara tijekom izvođenja algoritma.

2.6 Rukovanje granicama prostora pretrage

U pravilu su uvijek poznati rasponi unutar kojih se mogu nalaziti parametri kojima je opisan problem. To znači da se pretraga odvija unutar ograničenog prostora $\mathcal{S} = [\mathbf{s}^L, \mathbf{s}^U] \subset \mathbb{R}^d$, gdje je $\mathbf{s}^L = (s_1^L, \ldots, s_d^L)$ i $\mathbf{s}^U = (s_1^U, \ldots, s_d^U)$ kao i prethodno. S obzirom da $\mathcal{S}$ općenito nije (linearni) podprostor od $\mathbb{R}^d$, a mutacijom se generira mutant kao linearna kombinacija vektora populacije, isti se može nalaziti izvan njega. Ovakva rješenja odnosno vektori nisu valjani i prihvatljivi, jer se križanjem na pokusni vektor mogu prenijeti komponente koje su izvan granica. Prema tome, prilikom računanja mutanta treba uzeti u obzir mogućnost pojave navedenog i rukovati istim na odgovarajući način.

Rukovanje granicama prostora pretrage (engl. *bound constraint handling*) moguće je postići na različite načine. Strogo gledano, radi se o ograničenjima nejednakosti. Jednostavan i intuitivan način ispravljanja mutanta $\mathbf{u}^j$ je postavljanje komponenti na

$$u_r^j := s_r^L + \mathcal{U}_{j,r}[0,1] \cdot (s_r^U - s_r^L), \quad \forall r \colon r \in \{1, \ldots, d\}, u_r^j < s_r^L \vee u_r^j > s_r^U, \tag{2.9}$$

gdje je $\mathcal{U}_{j,r}[0,1]$ uniformna slučajna varijabla iz $[0,1]$. Nasumičnim postavljanjem vrijednosti komponenti vektora (korišteno primjerice u [20, 78, 104]), može doći do njegovih velikih izmjena, što može dovesti do gubljenja usmjerenosti pretrage. To je posebno izraženo ukoliko je ispravljanje učestalo i ukoliko postoji zavisnost među parametrima kojima je problem opisan. S druge strane, ovakav način rukovanja granicama može pospješiti raznolikost populacije. Jednostavniji način ispravka mutanta je postavljanje komponenti na

$$u_r^j := \begin{cases} s_r^L & \text{ako } u_r^j < s_r^L \\ s_r^U & \text{ako } u_r^j > s_r^U \end{cases}, \quad \forall r\colon r \in \{1,\ldots,d\}, u_r^j < s_r^L \vee u_r^j > s_r^U \ . \tag{2.10}$$

Postavljanjem komponenti na vrijednosti granica (korišteno primjerice u [15, 97, 164]) može negativno utjecati na prostorni raspored vektora i raznolikost populacije, jer bi mutanti dobiveni na temelju takvih vektora vjerojatno bili izvan ili blizu istih granica. Međutim, ovakav način ne pravi nasumične izmjene te se može očekivati kako izmjene neće biti velike. Alternativan način je ispravljanje mutanta postavljanjem komponenti na

$$u_r^j := \begin{cases} (s_r^L + v_i^j)/2 & \text{ako } u_r^j < s_r^L \\ (s_r^U + v_i^j)/2 & \text{ako } u_r^j > s_r^U \end{cases}, \quad \forall r\colon r \in \{1,\ldots,d\}, u_r^j < s_r^L \vee u_r^j > s_r^U, \tag{2.11}$$

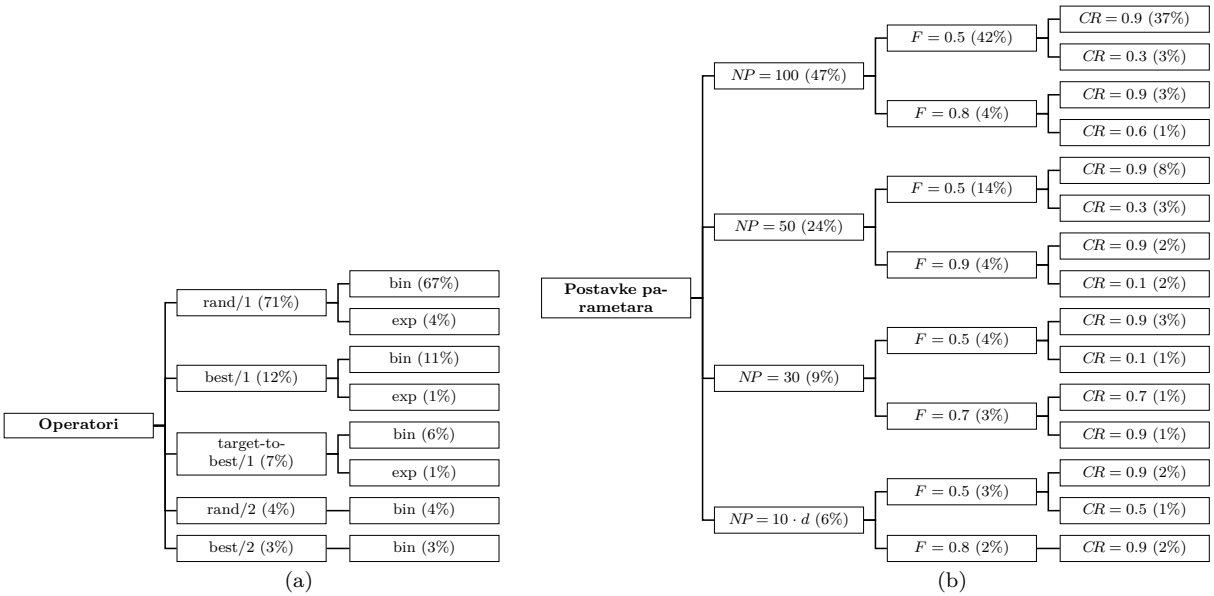
gdje je $\mathbf{v}^j$ dani ciljni vektor. Postavljanjem komponenti na središte između granice i odgovarajuće komponente ciljnog vektora (korišteno primjerice u [125, 162]) postiže se slično kao u prethodnom, ali uz znatno manje narušavanje prostornog rasporeda članova populacije. Umjesto ciljnog vektora moguće je koristiti i bazni vektor [103].

Opisani načini rukovanja ograničenjima granica ne predstavljaju jedine korištene. Primjerice, dvije blago različite i nasumične inačice načina ispravljanja prema (2.11), dane su u [72, 103], gdje se komponenta izvan granice postavlja na nasumičnu vrijednost između odgovarajuće komponente baznog vektora i prekoračene granice. Nekoliko dodatnih načina prikazani su i razmatrani u [41, 103].

2.7 Često korišteni algoritmi i postavke parametara

Opisivanje i označavanje različitih algoritama DE vrši se za nju svojstvenom nomenklaturom. Ona se odnosi na korištene operatore. Iako nije pogodna za opis svih, algoritme se u pravilu označava s DE/ρ/δ/ν, gdje ρ ukazuje na način određivanja ili odabira baznog vektora, δ na broj korištenih vektora razlike u mutaciji, dok ν ukazuje na korišteno križanje (bin i exp za binomno i eksponencijalno križanje, redom).

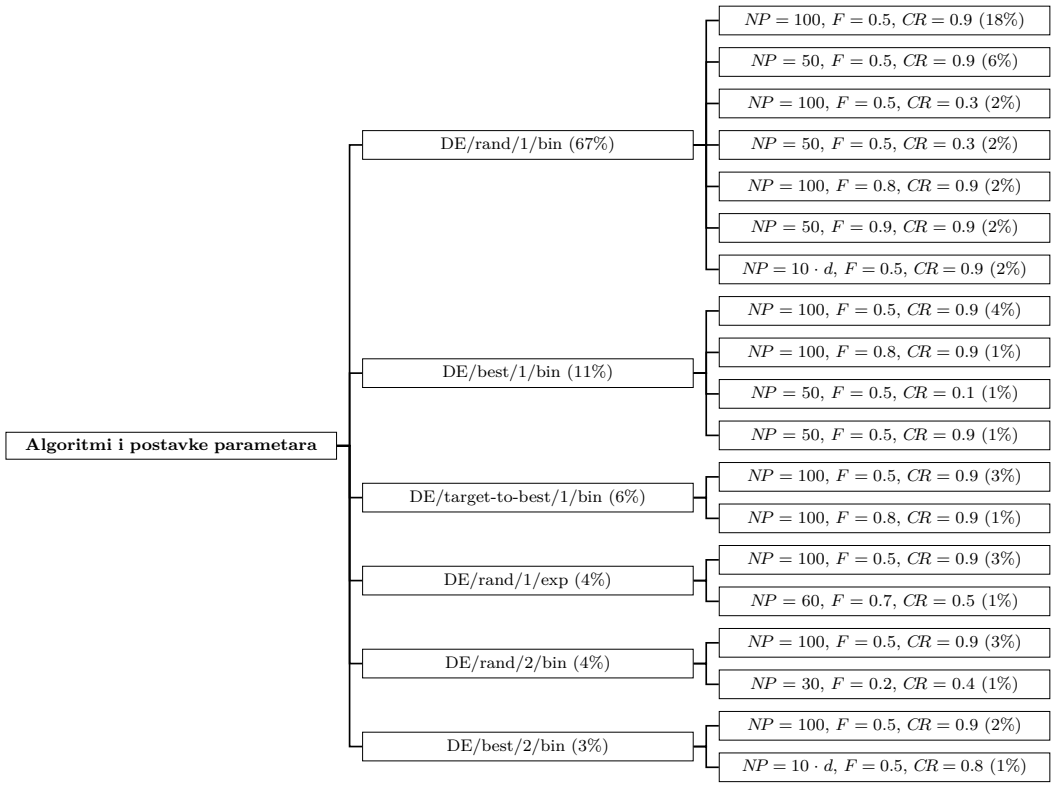
Nadalje, iz svega prethodno navedenog vidljivo je kako na raspolaganju stoje različiti operatori mutacije, barem dva operatora križanja te mnoštvo preporuka za postavljanje i traženje parametara. Takav izbor ne olakšava uporabu algoritma. Kako bi se pružio uvid u često korištene operatore i postavke parametara, napravljen je pregled literature. Točnije, iz 100 odabranih članaka, gdje je jednini kriterij bio da se rješavaju problemi s jednom funkcijom cilja te da nema drugih osim ograničenja prostora, izdvojeni su algoritmi (operatori mutacije i križanja) te pripadajuće postavke parametara. Tako je prikupljeno ukupno 180 algoritama i postavki parametara, a najčešći su zasebno prikazani slikom 2.6.



Slika 2.6: Često korišteni (a) operatori i (b) postavke parametara.

Sa slike je jednostavno uočiljivo da je algoritam DE/rand/1/bin najčešće korišten, a kojeg se smatra standardnim ili kanonskim algoritmom DE. Isto tako postavke parametara $NP = 100$, $F = 0.5$ i $CR = 0.9$ se posebno ističu. Štoviše, i pojedinačno razmatranje vodi do istih operatora i vrijednosti parametara. Osim toga, može se primijetiti rijetka uporaba mutacija s dva vektora razlike što se, u usporedbi s jednim može pripisati sporijoj konvergenciji. Također, $F = 0.5$ najčešće je korištena vrijednost faktora skaliranja te je korištena u kombinaciji sa širokim rasponom vrijednosti za stopu križanja, a posebno s $CR = 0.9$. Te vrijednosti za faktor skaliranja i stopu križanja predstavljaju preporuke Storn i Price [124] te Rönkkönen et al. [116], redom. Slikom 2.7 su prikazani često korišteni algoritmi zajedno s pripadajućim postavkama parametara, jer jedne postavke parametara ne moraju biti prikladne za različite operatore mutacije i križanja. Zanimljivo je da su ranije navedene postavke parametara najčešće korištene u sprezi s različitim operatorima mutacije, što je iznenađujuće zbog različitih načina na koji generiraju mutante. Kao jedan od razloga može se navesti što su često korištene postavke parametara preuzete iz drugih članaka (ne uzimajući u obzir algoritam s kojim su korištene) ili razlog za njihov odabir nije obrazložen.

Važno je istaknuti kako se navedene postavke parametara ne smije smatrati idealnima i pouzdanim izborom. Uzimajući u obzir ranije razmatranje parametara algoritma kao i činjenicu što su različiti algoritmi DE korišteni u sprezi s različitim postavkama parametara (posebno DE/rand/1/bin), jasno je da postavke parametara treba birati prema problemu kojeg se rješava. U tom pogledu, postavke parametara $NP = 100$, $F = 0.5$ i $CR = 0.9$ mogu poslužiti kao prvi izbor ili kao dobra polazna točka za daljnje traženje. To se posebno odnosi na situacije kada je problem nalik crnoj kutiji, dok je u suprotnom i ovisno o svojstvima problema, moguće eventualno napraviti bolji početni izbor za postavljanje parametara algo-



Slika 2.7: Često korišteni algoritmi i pripadajuće postavke parametara.

ritma. Slično se može reći i za odabir operatora, u prvom redu mutacije, koji mogu značajno utjecati na ponašanje algoritma i u konačnici na vjerojatnost pronalaska zadovoljavajućih rješenja. Stoga je s ciljem postizanja što veće učinkovitosti pri rješavanju danog problema nužno odabrati te uskladiti operatore i pripadajuće postavke parametara (operatore se može smatrati i posebnom vrstom parametara algoritma, vidi primjerice [33] i izvore unutar).

Mora se napomenuti da obavljeni pregled literature ni u kom pogledu nije opsežan zbog iznimno velikog broja dostupnih članaka. On predstavlja uzorak koji pruža uvid u uobičajeno korištene algoritme i postavke parametara. Koliko je reprezentativan nije moguće procijeniti, ali s obzirom na njegovu veličinu može se pretpostaviti da odstupanje od stvarne situacije ne može biti veliko. U manjem broju članaka korišteni su samo algoritmi DE u „čistom“ obliku, jer su u većini članaka predložene manje ili veće izmjene ili unaprjeđenja, dok su „čisti“ oblici korišteni za potrebe usporedbe. Također, iako takvi članci nisu bili uključeni u pregled literature, u rijetkim slučajevima nisu navedene postavke parametara ili nisu dane sve pojedine vrijednosti.

2.8 Osvrt na diferencijalnu evoluciju

Diferencijalna evolucija ima mnoge sličnosti s uobičajenim EAs kao što su populacija mogućih rješenja, uporaba varijacijskih operatora i selekcije. Ključni element DE i glavna razlika u

odnosu na druge EAs je mutacija. Osim što prethodni križanju, ona se se koristi razlikama vektora trenutne populacije za stvaranje perturbacija. Te razlike, ponderirane faktorom skaliranja, utvrđuju smjer i intenzitet mutacija, a izmjenom populacije i oni se mijenjaju. Na taj način se postiže samostalna prilagodba krajoliku dobrote [103]. Dodavanjem takve razlike na odabrani vektor trenutne populacije generira se mutant. Dodatna razlika se ogleda u križanju DE koje se može promatrati kao mutacija, jer se njime miješaju komponente člana trenutne populacije i mutanta. Broj komponenti preuzet iz mutanta ovisi od stopi križanja koja u tom smislu približno odgovara vjerojatnosti mutacije.

S obzirom na važnost mutacije, ne nedostaje različitih operatora iste. Oni se prvenstveno razlikuju u načinu odabira ili računanju vektora kojeg se perturbira te u broju razlika korištenih za to. Ovakav izbor nije dostupan kod križanja te se u pravilu svodi na jedan od dva operatora. Različite kombinacije operatora mogu rezultirati različitim algoritmima u pogledu učinkovitosti i ponašanja. Stoga, kod pristupanja danom problemu kojeg se rješava, treba osim parametara uzeti u obzir i dostupne operatore. Parametre predstavljaju veličina populacije, faktor skaliranja ili mutacije te stopa križanja, a o čijim postavkama učinkovitost algoritma značajno ovisi. Zbog njihove međusobne ovisnosti nije ih moguće promatrati zasebno. Međutim, i površnim pregledom literature može se doći do preporuka ili postavki koje mogu poslužiti kao dobra polazna točka za njihovo daljnje podešavanje.

Na koncu, radi se o relativno jednostavnom algoritmu, barem u osnovnom obliku, koji se pokazao i dokazao iznimno učinkovitim. U prvom redu, to vrijedi za probleme numeričke optimizacije za koje je i izvorno predložen. Međutim, uspjesi su zabilježeni i kod rješavanja problema diskretne i kombinatorne optimizacije te drugih vrsta problema. Jednostavnost i učinkovitost čine algoritam DE pouzdanim izborom za rješavanje mnogih problema globalne optimizacije, u prvom redu kada analitički oblik funkcije cilja nije dostupan ili ukoliko ga nije moguće iskoristiti na prikladan način.

POGLAVLJE 3

Podešavanje parametara diferencijalne evolucije

Parametre diferencijalne evolucije nužno je postaviti prije izvođenja algoritma. Međutim, odgovarajuće vrijednosti nije uvijek jednostavno odrediti i općenito zahtijeva posebna dodatna izvođenja algoritma. Osim toga, problem predstavlja i činjenica što se dobre vrijednosti mogu mijenjati tijekom pretrage. Rješavanje ovog problema zahtijeva njihovu stalnu prilagodbu odnosno podešavanje. Podešavanjem vrijednosti parametara tijekom pretrage može se potpomoći učinkovitost algoritma i ujedno osloboditi korisnika od tereta traženja prikladnih vrijednosti. U ovom poglavlju predlaže se postupak samopodešavanja faktora skaliranja i stope križanja. Postupak se ističe pohranom prethodno uspješnih vrijednosti navedenih parametara i njihovim naknadnim iskorištavanjem za potrebe generiranja novih vrijednosti. Eksperimentalnim testiranjem i analizom, iscrpno je vrednovan navedeni prijedlog prvog izvornog znanstvenog doprinosa.

3.1 Motivacija

Općenito, parametre DE predstavljaju veličina populacije NP, faktor skaliranja F i stopa križanja CR. Učinkovitost algoritma je, očekivano, vrlo osjetljiva na vrijednosti tih parametara. Osim toga, svi ti parametri međusobno su ovisni. Dodatnu otežavajuću okolnost pri odabiru dobrih vrijednosti parametara predstavlja i činjenica što različite vrijednosti istih mogu biti potrebne u slučaju različitih problema kako bi se postigla zadovoljavajuća učinkovitost algoritma [35, 154, 162, 166]. Štoviše, slična potreba može se javiti i pri uporabi različitih operatora mutacije i križanja. Prema tome, iako postoje mnoge različite preporuke (vidi primjerice [44, 103, 124]), traženje ili postavljanje vrijednosti parametara (engl. *parameter tuning*) koji će rezultirati zadovoljavajućom učinkovitošću pri rješavanju danog

problema iziskuje značajan napor i vrijeme te se u pravilu svodi na metodu pokušaja i pogrešaka ili na uporabu algoritama za automatsko traženje i postavljanje parametara (vidi primjerice [71, 127]). Bez obzira na pristup, radi se o procedurama koje zahtijevaju dodatno vrijeme računanja.

Nadalje, prikladne vrijednosti parametara mogu se mijenjati tijekom pretrage [59, 162], što znači da u različitim fazama mogu biti prikladne različite vrijednosti. Ovo predstavlja dodatni problem kojeg nije moguće riješiti samo odabirom dobrih vrijednosti parametara prije izvođenja, nego to zahtijeva njihovo mijenjanje, odnosno zahtijeva se podešavanje vrijednosti parametara kako bi se mogla održati učinkovita pretraga. Rješavanje ovog problema implicitno rješava i problem odabira parametara [59]. Potreba za podešavanjem parametara (engl. *parameter control*) tijekom izvođenja EAs uočena je vrlo rano, ali i dalje predstavlja veliki izazov. Pažljivo osmišljeni i izrađeni postupci podešavanja parametara algoritma mogu pospješiti ili unaprijediti njegovu učinkovitost u smislu brzine konvergencije te otpornosti na preuranjenu konvergenciju i stagnaciju pretrage [162].

3.2 Pregled literature

Iako je vrlo rano uočena potreba za podešavanjem parametra različitih EAs, ono je i dalje predmet mnogih istraživanja. U tom pogledu ni DE ne predstavlja izuzetak pa u literaturi postoje raznoliki postupci. Uglavnom se radi o postupcima koji služe za podešavanje vrijednosti parametara faktora skaliranja F i stope križanja CR, dok veličina populacije NP ostaje korisnički definiran parametar. Stoga se može izvesti zaključak da dobro izgrađeni postupci mogu prilagoditi vrijednosti navedenih parametara postavljenoj veličini populacije. Općenito, prema Eiben et al. [35], postupke podešavanja parametara se može (ugrubo) razvrstati u deterministčke ili dinamičke, prilagodljive (engl. *adaptive*) i postupke samopodešavanja (engl. *self-adaptive*). Međutim, nije uvijek moguće ili jednostavno povući granicu, odnosno svrstati neki postupak u jednu od navedenih skupina. U nastavku je opisano po nekoliko postupaka iz svake skupine, a ukoliko nije bila očigledna skupina u koju spadaju, svrstani su u skupinu prema deklaraciji autora dotičnih postupaka.

Dva jednostavna deterministička mehanizma za podešavanje faktora skaliranja predstavili su Das et al. [25]. Prvi mehanizam na početku svake generacije nasumično stvara novu vrijednost F unutar intervala $[0.5, 1]$, a vrlo sličan mehanizam su predstavili i Kim et al. [61]. Drugi, linearno smanjuje F tijekom izvođenja algoritma od najveće do najmanje zadane vrijednosti parametra. Huang i Chen [53] predložili su deterministički postupak podešavanja faktora skaliranja i stope križanja zasnovan na sinusnoj i kosinusnoj trigonometrijskoj funkciji, a nove vrijednosti računaju se svake generacije. Međutim, sam postupak zahtijeva postavljanje dva vlastita parametra. Jednostavan postupak, također za podešavanje parametara F i CR, predložili su Zou et al. [169]. U istom se za svaki vektor populacije $\mathbf{v}^j$ stvara

pripadajuća vrijednost parametra $CR_j \in [0.8, 1]$ pomoću uniformnih, dok se vrijednost parametra F_j stvara pomoću Gaussovih slučajnih varijabli uz srednju vrijednost $\mu = 0.75$ i standardnu devijaciju $\sigma = 0.1$. Nove vrijednosti stvaraju se svake generacije prije mutacije i križanja.

Prethodno navedeni mehanizmi ili postupci mijenjaju vrijednosti parametara prema zadanim pravilima ne uzimajući u obzir stanje populacije ili tijek pretrage. Također, ne uzimaju u obzir ni korisnost ili uspješnost pojedinih vrijednosti parametara. Liu i Lampinen [74] predložili su upravljače zasnovane na neizrazitoj logici (engl. *fuzzy logic*) za podešavanje vrijednosti parametara F i CR. Ulaz u upravljače predstavljaju razlike u položajima i kvaliteti populacije u dvije uzastopne generacije, a izlaze vrijednosti parametara. Postupak podešavanja parametara F i CR predložen od Yu i Zhang [154] koristi se kvalitetom vektora populacije kao i njihovom udaljenosti od trenutno najboljeg. Vrijednosti parametara mijenjaju se prema zadanim pravilima koja koriste te podatke. Tatsis i Parsopoulos [129] predložili su postupak podešavanja parametara F i CR pretragom rešetkaste strukture vrijednosti istih. Postupak zahtijeva uvođenje $M = 8$ podpopulacija, uz glavnu, a u kojima se provodi istraživanje prostora pretrage s različitim vrijednostima parametara. Te vrijednosti predstavljaju M susjednih u odnosu na one korištene u trenutno glavnoj populaciji, koju zamjenjuje neka podpopulacija ako je bila dostatno bolja. Prethodna tri postupaka vrše podešavanje na razini populacije, što znači da sve jedinke koriste iste vrijednosti. Međutim, mnogi drugi postupci vrše podešavanje na razini članova populacije tako da dodjeljuju svakom vlastite vrijednosti. Tvrdík [132] je predložio postupak u kojem se za svaki vektor odabire jedan od devet dostupnih parova vrijednosti (F, CR). Odabir je probabilistički, a vjerojatnosti odabira parova su vezane za njihovu uspješnost. Posve drukčiji pristup su predložili Zhang i Sanderson [162], gdje se za svaki $\mathbf{v}^j$ stvaraju vrijednosti CR_j pomoću Gaussovih, a vrijednosti F_j pomoću Cauchyjevih slučajnih varijabli. U oba slučaja stvaraju se nove vrijednosti u okolini sredine prethodno uspješnih vrijednosti parametara cijele populacije, pri čemu se sredina za CR računa kao aritmetička, a za F kao Lehmerova srednja vrijednost. Ovaj postupak podešavanja parametara usvojen je i dorađen, primjerice u [54, 125]. Štoviše, vrlo sličan pristup su ponudili Zhao et al. [164], a on se razlikuje u određivanju sredina u okolini kojih se potom, suprotno od pristupa iz [162], vrijednosti parametara CR_j stvaraju pomoću Cauchyjevih, a vrijednosti F_j pomoću Gaussovih slučajnih varijabli. Yu et al. [152] proširili su svoj postupak podešavanja parametara na razini populacije [154], uvođenjem koraka kojim se naknadno obavlja prilagodba za dio vektor. Kao i u prvom koraku, izmjene se zasnivaju na podacima o rangu i udaljenosti od najboljeg u populaciji. Izmjene se obavljaju samo u slučaju vektora koji su kvalitetni (dobri) i nalaze se relativno blizu najboljeg i obratno za vektore koji su daleko od najboljeg i male su kvalitete. Nadalje, Bajer i Martinović [7] predložili su postupak inspiriran optimizacijom kolonijom mrava (engl. *ant colony optimisation*) u kojem, na konceptualnoj razini, umjetni mravi odabiru F_j i CR_j

za svaki vektor populacije $\mathbf{v}^j$. Za oba parametra dani se skupovi raspoloživih vrijednosti. Umjetni mravi vođeni su modelom feromona koji odražava korisnost dostupnih vrijednosti parametara i kojeg se krajem svake generacije obnavlja.

Većina prethodno navedenih postupaka dodjeljuje svakom članu populacije zasebne vrijednosti, ali one nisu vezana za njih, jer se obavlja njihova stalna zamjena novima. U slučaju postupaka samopodešavanja, svaka jedinka posjeduje vlastite vrijednosti koje može prenijeti na potomke, a osim toga, one u pravilu prolaze neki oblik varijacije. Jedan od prvih postupaka samopodešavanja parametara DE predložili su Brest et al. [15]. U navedenom postupku, svakom vektoru $\mathbf{v}^j$ pridružene su vlastite, odnosno kodirane su unutar njega, vrijednosti F_j i CR_j. Prije mutacije i križanja s vjerojatnostima $\tau_1 = 0.1$ i $\tau_2 = 0.1$, za dani vektor stvaraju se nove nasumične vrijednosti $F_j \in [0.1, 1]$ i $CR_j \in [0, 1]$. Nove vrijednosti se stvaraju za F_j sa zadanom vjerojatnosti neovisno o CR_j i obratno. Ukoliko budu uspješne, one se pridružuju potomku koji zamjenjuje roditelja u novoj generaciji. Na taj način omogućeno je promicanje dobrih vrijednosti parametara. Navedeni postupak proširen je odnosno unaprjeđen u [16], a opsežna i detaljna analiza utjecaja parametara τ_1 i τ_2 provedena je u [160]. Donekle sličan postupak predložili su Noman et al. [93]. U njemu se parametri pridruženi vektorima populacije zamjenjuju novim vrijednostima samo ako su njihovi potomci bili lošiji od prosjeka kvalitete trenutne populacije. Kao i u postupku iz [15], nove vrijednosti se nasumično generiraju unutar $[0.1, 1]$ za F_j te unutar $[0, 1]$ za CR_j. Salman et al. [117] predložili su postupak samopodešavanja faktora skaliranja, dok se stopa križanja generira posebno za svaki vektor pomoću Gaussovih slučajnih varijabli. Navedeni postupak zasnovan je na postupku samopodešavanja stope križanja predloženom u [2]. Prije mutacije se vrijednost F_j koja će biti korištena i dodijeljena potomku, računa kao kombinacija faktora skaliranja pridruženih nasumično odabranim vektorima populacije. Prema načinu samopodešavanja parametara uobičajeno korištenom u evolucijskim strategijama (engl. *evolutionary strategies*), Brest et al. [17] predložili su postupak samopodešavanja parametara F i CR u algoritmu DE. Nadalje, Pan et al. [97] predložili su postupak u kojem je svakom članu populacije $\mathbf{v}^j$, umjesto po jedne, pridružen popis vrijednosti $L_{F_j} = (F^1, \ldots, F^m)$ te popis vrijednosti $L_{CR_j} = (CR^1, \ldots, CR^m)$. Te vrijednosti se redom kroz generacije koriste pri mutaciji i križanju. Dodatno, svakom članu populacije pridruženi su i popisi wL_{F_j} i wL_{CR_j} koji se pune uspješnim vrijednostima F i CR, redom. Nakon što su iskorištene sve vrijednosti parametara iz pridruženih popisa, nakon m generacija, oni se obnavljaju novim vrijednostima koje se stvaraju nasumično (tako da $F \in [0.1, 1]$ i $CR \in [0, 1]$) ili nasumično odabranim vrijednostima iz wL_{F_j} odnosno wL_{CR_j}. Lako se može zaključiti da je navedeni postupak velike prostorne složenosti zbog velikog broja vrijednosti ($m = 50$ korišteno u [97]) pridruženih svakom članu populacije.

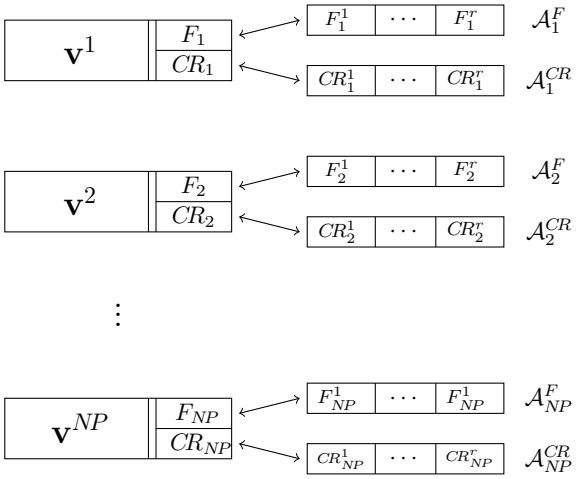
3.3 Postupak samopodešavanja parametara skaliranja i križanja DE

Predloženi postupak samopodešavanja faktora skaliranja i stope križanja zasniva se na pohrani određenog broja njihovih prethodno uspješnih vrijednosti. Traženje pogodnih vrijednosti parametara provodi se u okolini pohranjenih, jer je pretpostavka da se nove pogodne vrijednosti nalaze u okolini nekolicine prethodnih. Pohrana se obavlja za svakog člana populacije zasebno čime se teži očuvanju raznovrsnosti parametara. Održavanje nekolicine prethodno uspješnih vrijednosti parametara za svakog od članova populacije uz njihovo iskorištavanje za generiranje novih ističe predloženi postupak u odnosu na mnoge postupke dostupne u literaturi.

3.3.1 Opis predloženog postupka

Predloženi postupak samopodešavanja faktora skaliranja i stope križanja traži prikladne vrijednosti tih parametara u okolini prethodno uspješnih. Tako su svakom vektoru populacije $\mathbf{v}^j$ pridruženi $F_j \in \langle 0, 1]$ i $CR_j \in [0, 1]$, a dodatno su im dodijeljeni popisi $\mathcal{A}_j^F = \left(F_j^1, \ldots, F_j^r\right)$ i $\mathcal{A}_j^{CR} = \left(CR_j^1, \ldots, CR_j^r\right)$ koji predstavljaju prethodno uspješne vrijednosti tih parametara. Navedeno je ilustrirano slikom 3.1.

Svake generacije g, prilikom mutacije i križanja za dani ciljni vektor (roditelj) $\mathbf{v}^{j,g}$ koriste se vrijednosti F_j^g i CR_j^g. Ukoliko te vrijednosti budu uspješne, odnosno ukoliko nastali pokusni vektor (potomak) $\mathbf{t}^{j,g}$ prijeđe u narednu generaciju $(g+1)$ zamjenjujući roditelja, one se prenose na njega i pohranjuju u $\mathcal{A}_j^F$ odnosno $\mathcal{A}_j^{CR}$ koje mu se također dodjeljuju. S obzirom da su navedeni popisi fiksne veličine r, uspješne vrijednosti zamjenjuje najstarije u



Slika 3.1: Pridružene i pohranjene vrijednosti parametara pri predloženom postupku samopodešavanja faktora skaliranja i stope križanja.

njima. Treba napomenuti da ti popisi mogu sadržavati iste elemente.

S druge strane, ukoliko vrijednosti budu neuspješne, doći će do njihove eventualne zamjene. Naime, trenutno pridružene vrijednosti zadržavaju se ili zamjenjuju novima. Razlog zadržavanja je što i neke dobre vrijednosti, primjenom u mutaciji i križanju ne moraju nužno rezultirati dobrim potomkom. Ovo u prvom redu ovisi o odabiru vektora koji sudjeluju u mutaciji pa je za očekivati kako će nepovoljni odabir rezultirati lošim potomkom bez obzira na korištene vrijednosti parametara. Odluka o zadržavaju ili zamjeni se donosi uz vjerojatnosti 50:50 %. Ipak, ukoliko dođe do zamjene, ona se obavlja jednim od dva pristupa, gdje se odluka o jednom ili drugom donosi uz vjerojatnosti 50:50 %. U prvom pristupu se jednostavno odabiru vrijednosti između pohranjenih, odnosno $F_j^{g+1} = F_j^t \leftrightarrow \mathcal{A}_j^F$ i $CR_j^{g+1} = CR_j^t \leftrightarrow \mathcal{A}_j^{CR}$, gdje je t nasumično odabran iz skupa $\{1, \ldots, r\}$, a $\leftrightarrow$ označava pripadnost popisu. Ovim se daje ponovno prilika nekima od prethodno uspješnih parova (F_j, CR_j). U drugom pristupu se generira nova vrijednost

$$F_j^{g+1} = \mu_j^F + \mathcal{N}_{j,1}(0, 0.1), \quad \mu_j^F = \frac{\sum_{i=1}^{r} w_i \cdot F_j^i \leftrightarrow \mathcal{A}_j^F}{\sum_{i=1}^{r} w_i}, \tag{3.1}$$

i analogno nova vrijednost

$$CR_j^{g+1} = \mu_j^{CR} + \mathcal{N}_{j,2}(0, 0.1), \quad \mu_j^{CR} = \frac{\sum_{i=1}^{r} w_i \cdot CR_j^i \leftrightarrow \mathcal{A}_j^{CR}}{\sum_{i=1}^{r} w_i}, \tag{3.2}$$

gdje je $W = (w_1, w_2, \ldots, w_r) = (r, r-1, \ldots, 1)$ i predstavlja težine za pohranjene vrijednosti parametara, a $\mathcal{N}_{j,i}(0, 0.1)$ za $i = 1, 2$ je Gaussova slučajna varijabla sa srednjom vrijednosti $\mu = 0$ i standardnom devijacijom $\sigma = 0.1$. Prema (3.1) i (3.2), nove vrijednosti se traže u okolini težinske srednje vrijednosti prethodno uspješnih pri čemu se veće težine pridaju novijim vrijednostima iz $\mathcal{A}_j^F$ i $\mathcal{A}_j^{CR}$. Ovo se može obrazložiti pretpostavkom da su pogodne vrijednosti parametara bliže novijima nego starijima. Također, uporabom Gaussovih slučajnih varijabli obavlja se relativno fina pretraga u okolini sredina μ_j^F i μ_j^{CR}.

Nadalje, s ciljem omogućavanja većih promjena pridruženih vrijednosti parametara one se bez obzira na uspjeh mijenjaju s vrlo malom vjerojatnosti. Te zamjene su nezavisne i potpuno nasumične, odnosno

$$F_j^{g+1} = \begin{cases} \mathcal{U}_{j,1}[0,1] & \text{ako } \mathcal{U}_{j,2}[0,1] < p_n \\ F_j^g & \text{u suprotnom} \end{cases}, \tag{3.3}$$

$$CR_j^{g+1} = \begin{cases} \mathcal{U}_{j,3}[0,1] & \text{ako } \mathcal{U}_{j,4}[0,1] < p_n \\ CR_j^g & \text{u suprotnom} \end{cases}, \tag{3.4}$$

gdje je $p_n = 0.01$ (1%), a $\mathcal{U}_{j,i}[0,1]$ za $i = 1, \ldots, 4$ je uniforma slučajna varijabla iz $[0,1]$. Iako su nasumične, one u slučaju uspjeha mogu uvesti raznolikost među pohranjene, dok će u slučaju neuspjeha biti vrlo brzo zamijenjene vrijednostima dobivenima prema prethodno uspješnim vrijednostima. Opisano nasumično generiranje vrijednosti parametara, analogno je pristupu generiranja vrijednosti parametara faktora skaliranja i stope križanja u [15] (kao što je ranije opisano na početku ovog poglavlja), ali uz značajno manje vjerojatnosti generiranja novih vrijednosti. Osim toga, takav način generiranja vrijednosti parametara u predloženom postupku predstavlja samo pomoćni mehanizam.

Prema svemu navedenom, jasno je da ključnu ulogu u predloženom postupku imaju popisi $\mathcal{A}_j^F$ i $\mathcal{A}_j^{CR}$. Oni su pridruženi svakom vektoru zasebno s ciljem što bolje moguće prilagodbe faktora skaliranja i stope križanja svakom vektoru te očuvanja veće raznolikosti tih parametara. Kako se nove vrijednosti parametara generiraju pomoću tih popisa (ne uzimajući u obzir vrlo malo vjerojatno nasumično generiranje navedenih parametara), njihova veličina r ima određeni utjecaj gdje će se nalaziti te nove vrijednosti. Tako manje veličine r omogućuju brže izmjene sredina μ_j^F i μ_j^{CR} i obratno. U tom pogledu, potrebna je uravnoteženost izmjena, jer spore izmjene ne dopuštaju praćenje stanja pretrage, dok brze izmjene mogu dovesti do oscilacija. Imajući to na umu, postavljeno je $r = 3$ čime i prostorna složenost predloženog postupka ostaje umjerena.

3.3.2 Detalji ugradnje

U predloženom postupku samopodešavanja, nove vrijednosti $F_j \in \langle 0,1]$ i $CR_j \in [0,1]$ dobivaju se primarno na temelju pohranjenih i prethodno uspješnih vrijednosti tih parametara. Treba primijetiti kako se, zbog $W = (r, r-1, \ldots, 1)$, nove uspješne vrijednosti pohranjuju na početak popisa $\mathcal{A}_j^F$ odnosno $\mathcal{A}_j^{CR}$ izbacujući zadnji element koji je najstariji. Te vrijednosti i popise $\mathcal{A}_j^F$ i $\mathcal{A}_j^{CR}$ potrebno je prvotno inicijalizirati. U tom smislu, postavlja se $F_j^0 = 0.5$ i $CR_j^0 = 0.9$ za $j = 1, \ldots, NP$. Ove vrijednosti odabrane su kao često korištene u literaturi (vidi potpoglavlje 2.7). Nadalje, početne vrijednosti pohranjene u popisima zadane su kao $\mathcal{A}_j^F = \mathcal{A}_j^{CR} = (1/r, 2/r, \ldots, 1)$ za $j = 1, \ldots, NP$, čime se ravnomjerno pokriva cijeli interval dozvoljenih vrijednosti za parametar F, odnosno parametar CR.

Nove vrijednosti navedenih parametara generiraju se na jedan od dva načina na temelju pohranjenih u popisima. U prvom slučaju, provodi se nasumični odabir između pohranjenih. Važno je primijetiti kako se odabire prethodno uspješni par (vrijednosti s jednakim indeksima u odgovarajućim popisima) zbog međusobne ovisnosti parametara. U drugom slučaju, generiraju se pomoću Gaussovih slučajnih varijabli u okolini težinske srednje vrijednosti pohranjenih. S obzirom da su rasponi vrijednosti parametara ograničeni, postoji mogućnost izlaska iz navedenih granica u slučaju primjene (3.1) odnosno (3.2). Ukoliko do toga dođe, vrši se korekcija [analogno (2.11)]

$$F_j = \begin{cases} \mu_j^F/2 & \text{ako} \quad F_j \leq 0 \\ (\mu_j^F + 1)/2 & \text{ako} \quad F_j > 1 \end{cases}, \tag{3.5}$$

i analogno

$$CR_j = \begin{cases} \mu_j^{CR}/2 & \text{ako} \quad CR_j < 0 \\ (\mu_j^{CR} + 1)/2 & \text{ako} \quad CR_j > 1 \end{cases}. \tag{3.6}$$

Također, u slučaju nasumičnog stvaranja F_j, prema (3.3), moguće je izaći izvan donje granice. Ispravljanje se vrši ponavljanjem generiranja vrijednosti dok god je F_j ispod te granice.

Ugradnja predloženog postupka samopodešavanja faktora skaliranja i stope križanja u algoritam DE zahtijeva odgovarajuća proširenja, u prvom redu u koraku selekcije naredne generacije. Naime, pri selekciji naredne generacije, obavlja se eventualna pohrana i izmjena pridruženih vrijednosti parametara, odnosno njihovo podešavanje (na temelju pohranjenih). Algoritmom 3.1 sažeto je prikazana ugradnja predloženog postupka u standardni algoritam. Strelicama $\Leftarrow$ su naznačene razlike u odnosu na osnovnu strukturu algoritma.

Algoritam 3.1: Nacrt rada DE s ugrađenim predloženim postupkom samopodešavanja faktora skaliranja i stope križanja

```
Postavi veličinu populacije NP; ⇐
Inicijaliziraj populaciju 𝒫 = (v^{1,0}, ..., v^{NP,0});
Postavi F_j^0 := 0.5, CR_j^0 := 0.9, j = 1, ..., NP; ⇐
Postavi 𝒜_j^F = 𝒜_j^CR = (1/r, 2/r, ..., 1), j = 1, ..., NP; ⇐
g := 0;
ponavljaj
    za j := 1, ..., NP čini
        % Stvaranje pokusnih vektora
        Mutacijom, uz F_j^g, stvori mutanta/donora u^{j,g};
        Križanjem u^{j,g+1} i v^{j,g}, uz CR_j^g, stvori pokusni vektor t^{j,g};
        % Selekcija naredne generacije
        ako f(t^{j,g}) ≤ f(v^{j,g}) onda
            v^{j,g+1} := t^{j,g};
            𝒜_j^F ← F_j^g, 𝒜_j^CR ← CR_j^g; ⇐
            F_j^{g+1} := F_j^g, CR_j^{g+1} := CR_j^g; ⇐
        inače
            v^{j,g+1} := v^{j,g};
            ako 𝒰_1[0,1] < 0.5 onda
                ako 𝒰_2[0,1] < 0.5 onda
                    F_j^{g+1} := F_j^t ↔ 𝒜_j^F, CR_j^{g+1} := CR_j^t ↔ 𝒜_j^CR; ⇐
                inače
                    Generiraj F_j^{g+1} i CR_j^{g+1} u okolini prethodno uspješnih; ⇐   % prema (3.1) i (3.2)
                kraj
            kraj
        kraj
        S vjerojatnosti p_n = 0.01 zamijeni F_j^{g+1} i CR_j^{g+1} nasumičnim vrijednostima; ⇐   % prema (3.3) i (3.4)
    kraj
    g := g+1;
dok uvjet završetka nije zadovoljen;
```

3.4 Eksperimentalni rezultati i analiza: Predloženi postupak samopodešavanja faktora skaliranja i stope križanja

Kako bi se pružio uvid u prednosti i nedostatke predloženog postupka samopodešavanja, provedena je odgovarajuća eksperimentalna analiza, a podijeljena je u četiri dijela. Prvi dio predstavlja usporedbu s fiksnim vrijednostima stope križanja i faktora mutacije. Drugi dio se odnosi na usporedbu s postupcima iz literature, a treći predstavlja analizu utjecaja broja pohranjenih prethodno uspješnih vrijednosti parametara. Četvrti dio analize pruža uvid u ponašanje procesa podešavanja parametara predloženim postupkom. Kako bi se pokrio vrlo širok spektar problema, za potrebe testiranja i analize korištena su dva skupa funkcija. Prvi skup predstavlja standardne testne funkcije različitih svojstava, a sastoji se od ukupno 30 funkcija. Funkcije $f_1 \sim f_{22}$ su skalabilne, s time da su $f_1 \sim f_{11}$ unimodalne, a funkcije $f_{12} \sim f_{22}$ su multimodalne kao i $f_{23} \sim f_{30}$ koje nisu skalabilne, nego zadanih dimenzionalnosti. Korištene i zadane dimenzionalnosti za pojedine funkcije prikazane su tablicom 3.1, dok su njihovi opisi dani u dodatku A. Drugi skup predstavlja funkcije pripremljene za CEC (*Congress on Evolutionary Computation*) 2014 natjecanje u numeričkoj optimizaciji s jednom funkcijom cilja [70] (vidi dodatak A). Taj skup se također sastoji od ukupno 30 funkcija, gdje su $F_1 \sim F_3$ unimodalne, $F_4 \sim F_{16}$ su multimodalne, $F_{17} \sim F_{22}$ su hibridne funkcije, dok su $F_{23} \sim F_{30}$ kompozicijske funkcije. Korištene dimenzionalnosti za funkcije CEC 2014 bile su $d = 10$ i $d = 30$. Treba napomenuti da se u slučaju funkcija CEC 2014 radi o vrlo složenom skupu koji predstavlja veliki izazov za sve algoritme globalne optimizacije. Prema navedenom, testiranje i analiza provedeni su na ukupno 90 instanci problema.

3.4.1 Postavke eksperimenata

Kroz cijelu analizu korišten je standardni DE algoritam (DE/rand/1/bin). On je odabran, jer se radi o jednoj od najjednostavnijih, ali i vrlo često korištenoj inačici algoritma. Za svaki algoritam i instancu problema izvršeno je 51 nezavisno izvođenje. Kao jedini korisnički parametar, korištena je veličina populacije $NP = 100$ koja predstavlja čest izbor (vidi potpo-

Tablica 3.1: Korištene dimenzionalnosti standardnih testnih funkcija.

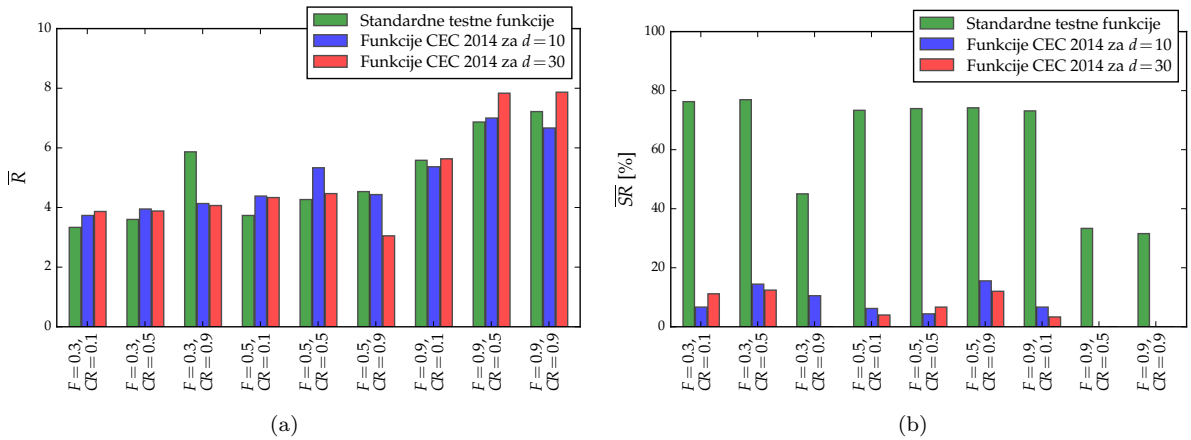
d	f
10	5, 6, 7, 10, 13, 20
30	2, 3, 9, 12, 14, 16, 17, 18, 22
50	1, 4, 8, 11, 15, 19, 21
2	23, 25
3	26
4	24, 28, 29, 30
6	27

glavlje 2.7). Za oba skupa funkcija bilo je dozvoljeno najviše $d \cdot 10^4$ vrednovanja funkcije cilja (engl. *function evaluations*, FEs), s izuzetkom na $f_{23} \sim f_{30}$ za koje je maksimalni broj vrednovanja funkcije cilja (engl. *maximum number of function evaluations*, $\text{NFEs}_{\max}$) bio postavljen na 10^5. U slučaju standardnih testnih funkcija do prekida izvođenja je dolazilo isključivo nakon izvršenog $\text{NFEs}_{\max}$. Pri funkcijama CEC 2014 do prekida je dolazilo kao u prethodnom slučaju ili ranije ako je postignuta zadana greška optimizacije $\Delta f = |f^\star - f| < 10^{-8}$, a koja je uz to uzeta kao nula (prema uputama iz [70]). Međutim, za oba skupa funkcija bilježena je stopa uspješnosti (engl. *success-rate*, *SR*) u postizanju navedene greške optimizacije (izražena u postocima). Nadalje, granicama prostora rukovano je prema (2.11). Populacije su u svim algoritmima inicijalizirane nasumično stvorenim rješenjima unutar cijelog prostora pretrage. Za stvaranje pseudo-slučajnih brojeva je korišten Mersenne Twister [83] generator pseudo-slučajnih brojeva (engl. *pseudo-random number generator*, PRNG).

3.4.2 Usporedba s nepromjenjivim vrijednostima parametara

Kod postavljanja parametara (prvenstveno faktora skaliranja i stope križanja) može se voditi preporukama iz literature, uzeti neke često korištene vrijednost ili pak provesti traženje pogodnih vrijednosti. S obzirom da različite postavke mogu pružati različitu učinkovitost na različitim problemima, provedeni su testiranje i analiza s tog gledišta. Točnije, provedena je usporedba standardnog algoritma DE s ugrađenim predloženim postupkom (označeno s $\text{DE}_{(\text{ASP})}$) i istog algoritma s različitim fiksnim vrijednostima faktora mutacije i stope križanja. U tu svrhu, prvo je provedeno ograničeno, ali vremenski zahtjevno traženje pogodnih vrijednosti navedenih parametara. Također, odabrano je nekoliko postavki iz literature.

Traženje pogodnih parova (F, CR) provedeno je nad skupovima $\Omega_F = \{0.3, 0.5, 0.9\}$ i $\Omega_{CR} = \{0.1, 0.5, 0.9\}$, odnosno nad njihovim Karterzijevim produktom $\Omega_F \times \Omega_{CR}$. Slika 3.2 sažeto prikazuju rezultate pretrage u smislu prosječnih rangova $\overline{R}$ i prosječnih stopa uspješ-



(a) (b)

Slika 3.2: Međusobna usporedba različitih postavki faktora skaliranja i stope križanja u smislu (a) prosječnih rangova $\overline{R}$ i (b) prosječnih stopa uspješnosti $\overline{SR}$.

Tablica 3.2: Rezultati na standardnim testnim funkcijama. Usporedba fiksnih postavki faktora skaliranja i stope križanja te njihovog samopodešavanja predloženim postupkom.

f	$F=0.3$, $CR=0.1$	$F=0.3$, $CR=0.5$	$F=0.5$, $CR=0.4$	$F=0.5$, $CR=0.7$	$F=0.5$, $CR=0.9$	$\mathrm{DE}_{(\mathrm{ASP})}$
1	▽†	△†	▽†	▽†	▽†	¬
2	▽†	△†	▽†	▽†	▽†	¬
3	▽†	▽†	▽†	▽†	△∘	¬
4	△†	▽∘	▽†	▽†	▽†	¬
5	△†	▽†	▽†	△∘	△†	¬
6	▽†	△†	▽†	▽†	▽†	¬
7	▽†	▽†	▽†	▽†	△†	¬
8	○	○	○	○	○	¬
9	▽†	△†	▽†	▽†	▽†	¬
10	▽†	△†	▽†	▽†	▽†	¬
11	▽∘	▽†	▽†	▽†	△†	¬
12	○	▽∘	○	▽†	▽†	¬
13	○	▽∘	○	▽†	▽†	¬
14	○	○	▽†	▽†	▽†	¬
15	○	○	○	○	▽∘	¬
16	▽†	▽†	▽†	▽†	▽†	¬
17	▽†	▽†	▽†	▽†	▽†	¬
18	▽†	▽†	▽†	▽†	▽†	¬
19	○	○	○	▽†	○	¬
20	▽†	○	▽†	▽∘	▽∘	¬
21	○	○	▽†	▽†	▽∘	¬
22	○	○	○	▽†	▽†	¬
23	○	○	○	○	○	¬
24	▽†	▽†	○	○	○	¬
25	○	○	○	○	○	¬
26	○	○	○	○	○	¬
27	○	▽∘	○	○	▽∘	¬
28	▽†	▽∘	○	○	○	¬
29	○	○	○	○	○	¬
30	▽∘	○	○	○	○	¬
—	13	8	16	18	13	▽†
—	2	5	0	1	4	▽∘
—	13	12	14	10	9	○
—	2	5	0	0	3	△†
—	0	0	0	1	1	△∘

nosti $\overline{SR}$. Pojedini parovi (F, CR) međusobno rangirani po svim funkcijama posebno prema prosječnim greškama optimizacije, pri čemu je najmanji rang na danoj funkciji dodijeljen najboljem paru i tako redom. Ukoliko je bilo više istih rangova, odgovarajućim parovima dodijeljena je prosječna vrijednost rangova. Prema slici 3.2, na standardnim testnim funkcijama najmanji $\overline{R}$ ostvareni su s $F = 0.3$ uz $CR \in \{0.1, 0.5\}$, a to se odražava i kroz $\overline{SR}$. Vrlo slična je situacija i u slučaju funkcija CEC 2014 za $d = 10$, dok je za $d = 30$ uvjerljivo najmanji $\overline{R}$ ostvaren s $F = 0.5$ i $CR = 0.9$ pri čemu te postavke, po prosječnim rangovima slijede prethodno spomenute.

Za potrebe usporedbe odabrani su parovi $(F, CR) = (0.3, 0.1)$ i $(0.3, 0.5)$, dobiveni traženjem pogodnih vrijednosti faktora skaliranja i stope križanja. Uz to, iz literature su preuzeti parovi $(F, CR) = (0.5, 0.4)$ [164], $(0.5, 0.7)$ [37, 129] i $(0.5, 0.9)$ [15, 93, 97, 154] koji su korišteni uz različite veličine populacije uključujući $NP = 100$. Tablice 3.2, 3.3 i 3.4 prikazuju odnos algoritama s fiksnim vrijednostima navedenih parametara i algoritma

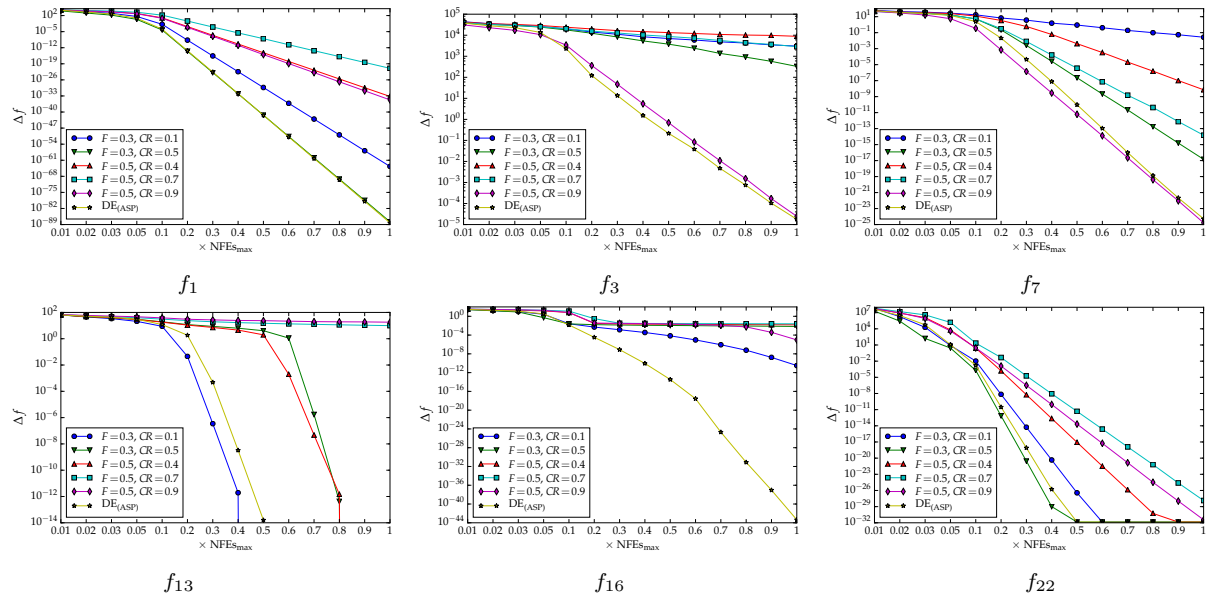
Tablica 3.3: Rezultati na funkcijama CEC 2014 za $d = 10$. Usporedba fiksnih postavki faktora skaliranja i stope križanja te njihovog samopodešavanja predloženim postupkom.

F	$F = 0.3$, $CR = 0.1$	$F = 0.3$, $CR = 0.5$	$F = 0.5$, $CR = 0.4$	$F = 0.5$, $CR = 0.7$	$F = 0.5$, $CR = 0.9$	$\mathrm{DE}_{(\mathrm{ASP})}$
1	▽†	▽†	▽†	▽†	△†	¬
2	▽†	◯	▽†	◯	◯	¬
3	▽†	◯	▽†	◯	◯	¬
4	△†	▽∘	△∘	▽∘	△†	¬
5	△†	▽†	▽†	▽†	▽†	¬
6	▽†	△∘	▽†	△∘	△∘	¬
7	▽†	▽∘	▽†	▽†	▽†	¬
8	◯	◯	◯	▽†	▽†	¬
9	▽†	▽†	▽†	▽†	▽†	¬
10	△†	▽†	▽†	▽†	▽†	¬
11	▽†	▽†	▽†	▽†	▽†	¬
12	△†	▽†	▽†	▽†	▽†	¬
13	▽†	▽†	▽†	▽†	▽†	¬
14	▽†	▽†	▽†	▽†	▽†	¬
15	▽†	▽†	▽†	▽†	▽†	¬
16	▽∘	▽†	▽†	▽†	▽†	¬
17	▽†	▽†	▽†	▽†	△†	¬
18	▽†	▽†	▽†	▽†	△†	¬
19	▽†	▽†	▽†	▽†	▽†	¬
20	▽†	▽†	▽†	▽†	▽∘	¬
21	▽†	▽†	▽†	▽†	▽†	¬
22	△∘	△†	△†	▽†	▽†	¬
23	△∘	◯	◯	◯	◯	¬
24	▽†	▽†	▽†	▽†	▽†	¬
25	▽†	▽†	▽†	▽†	▽†	¬
26	▽†	▽†	▽†	▽†	▽†	¬
27	▽†	▽†	▽†	▽†	▽†	¬
28	△†	△∘	△∘	▽∘	▽∘	¬
29	▽†	▽†	▽†	△∘	△†	¬
30	▽†	▽†	▽†	▽†	△†	¬
—	21	21	25	23	18	▽†
—	1	2	0	2	2	▽∘
—	1	4	2	3	3	◯
—	5	1	1	0	6	△†
—	2	2	2	2	1	△∘

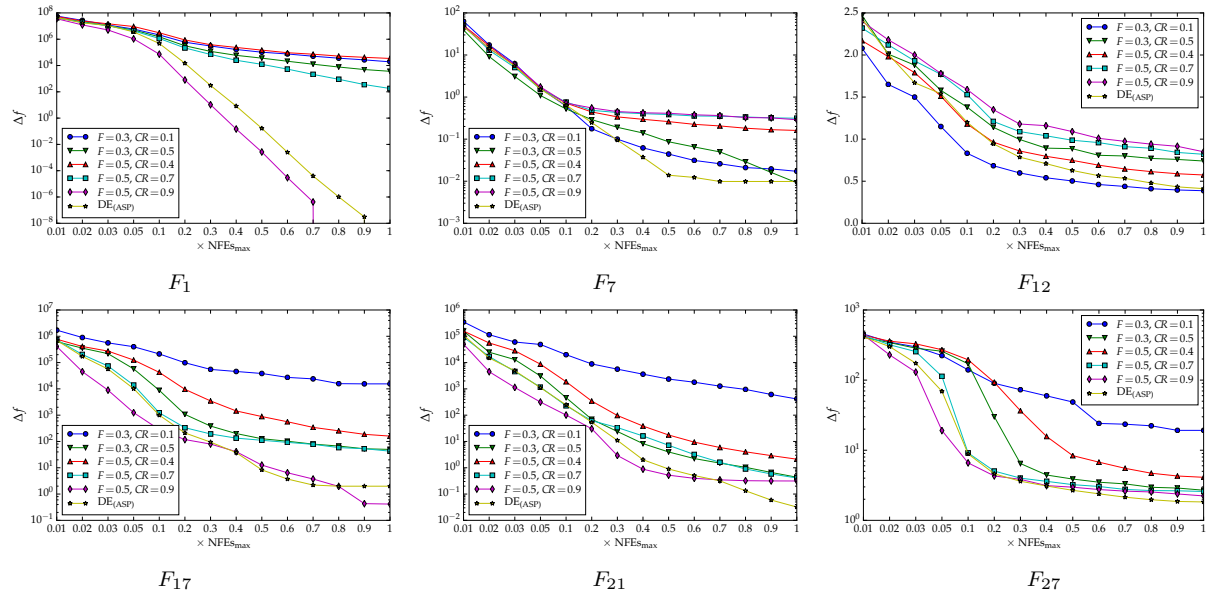
Tablica 3.4: Rezultati na funkcijama CEC 2014 za $d = 30$. Usporedba fiksnih postavki faktora skaliranja i stope križanja te njihovog samopodešavanja predloženim postupkom.

F	$F = 0.3$, $CR = 0.1$	$F = 0.3$, $CR = 0.5$	$F = 0.5$, $CR = 0.4$	$F = 0.5$, $CR = 0.7$	$F = 0.5$, $CR = 0.9$	$\mathrm{DE}_{(\mathrm{ASP})}$
1	▽†	▽†	▽†	▽†	△†	¬
2	▽∘	▽∘	◯	◯	◯	¬
3	▽†	◯	▽†	▽†	◯	¬
4	▽†	▽†	▽†	△†	△†	¬
5	▽†	▽†	▽†	▽†	▽†	¬
6	▽†	△†	▽†	△†	△†	¬
7	△∘	△∘	△∘	△∘	△∘	¬
8	△†	▽†	▽†	▽†	▽†	¬
9	▽†	▽†	▽†	▽†	▽†	¬
10	△†	▽†	▽†	▽†	▽†	¬
11	▽†	▽†	▽†	▽†	▽†	¬
12	▽†	▽†	▽†	▽†	▽†	¬
13	▽†	▽†	▽†	▽†	▽†	¬
14	▽†	▽†	▽†	▽†	▽†	¬
15	▽†	▽†	▽†	▽†	▽†	¬
16	▽†	▽†	▽†	▽†	▽†	¬
17	▽†	▽†	▽†	▽†	△†	¬
18	▽†	▽†	▽†	▽†	△∘	¬
19	▽†	▽†	▽†	▽†	▽†	¬
20	▽†	▽†	▽†	▽†	▽†	¬
21	▽†	▽†	▽†	▽†	▽†	¬
22	▽†	▽†	▽†	▽†	△†	¬
23	◯	▽∘	◯	◯	◯	¬
24	▽†	△∘	△†	△†	△†	¬
25	▽†	▽†	▽†	▽†	△†	¬
26	▽†	▽†	▽†	▽†	▽†	¬
27	▽†	△†	△†	△†	▽∘	¬
28	▽†	△†	△∘	△†	△†	¬
29	▽†	▽†	▽†	▽†	△†	¬
30	▽†	▽†	▽†	▽†	△†	¬
—	25	22	24	22	14	▽†
—	1	2	0	0	1	▽∘
—	1	1	2	2	3	◯
—	2	3	2	5	10	△†
—	1	2	2	1	2	△∘

s ugrađenim predloženim postupkom. Na dnu tablica zbrojno su prikazani dobiveni rezultati. Znak ▽ ukazuje da je prosječna greška optimizacije danog algoritma veća u odnosu na $\mathrm{DE}_{(\mathrm{ASP})}$, na suprotno ukazuje znak △, dok znak ◯ ukazuje da se radi o jednakim prosječnim vrijednostima. Kako bi se utvrdilo radi li se o statistički značajnim razlikama, proveden je Wilcoxonov test ranga s predznakom [30] uz interval pouzdanosti od 95 %. Shodno tome, znakom † je naznačeno da se radi o statistički značajnoj razlici u prosjecima, a znakom ∘ da nema statistički značajne razlike. Prema prikazanim rezultatima jasno se vidi kako je $\mathrm{DE}_{(\mathrm{ASP})}$ u velikoj većini slučajeva ostvario bolje ili barem usporedive rezultate (manje i često statistički značajne ili jednake prosjeke grešaka optimizacije) u odnosu na algoritme s fiksnim postavkama parametara. Ovo vodi do zaključka da predloženi postupak samopodešavanja faktora skaliranja i stope križanja može uglavnom uspješno prilagoditi ili podesiti te parametre danom problemu. Isto sugeriraju i slike 3.3, 3.4 te 3.5 koje za svaki algoritam prikazuju

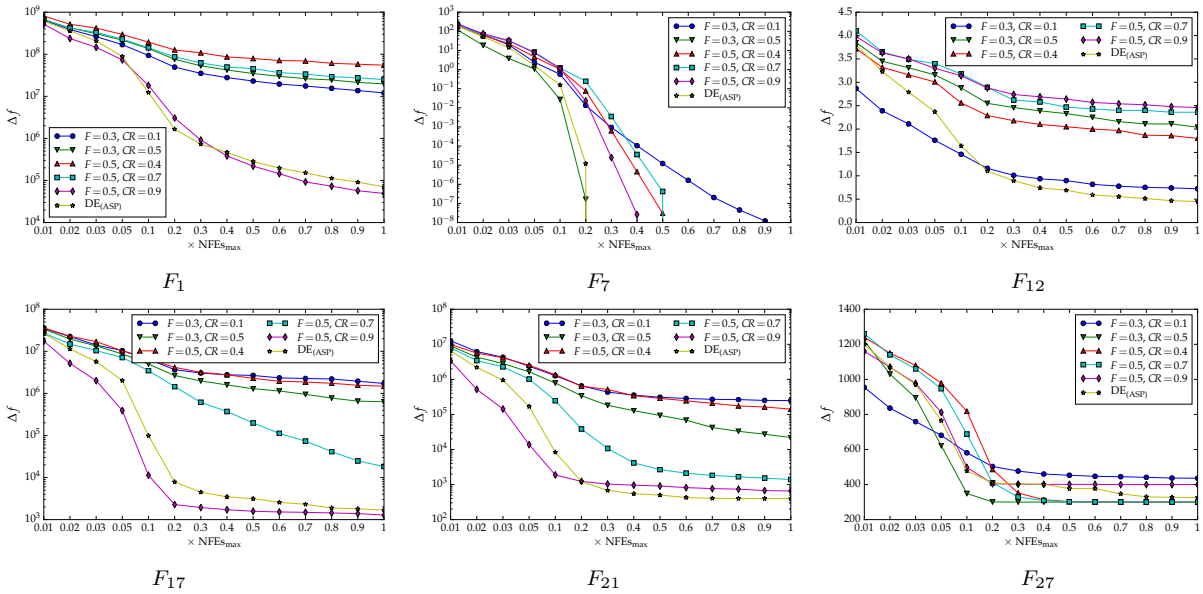


Slika 3.3: Grafovi konvergencije na nekoliko odabranih standardnih testnih funkcija. Usporedba algoritama s fiksnim postavkama faktora skaliranja i stope križanja te s njihovim samopodešavanjem predloženim postupkom.



Slika 3.4: Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d=10$. Usporedba algoritama s fiksnim postavkama faktora skaliranja i stope križanja te s njihovim samopodešavanjem predloženim postupkom.

medijan greške optimizacije svih (51) izvođenja tog algoritma za danu funkciju u odnosu na broj vrednovanja funkcije cilja (NFEs). Prema navedenim slikama, algoritam s ugrađenim predloženim postupkom, u pogledu brzine konvergencije, uspijeva držati korak ili nadmašiti algoritam s najučinkovitijim od upotrijebljenih fiksnih postavki parametara. To ukazuje na sposobnost predloženog postupka da tijekom cijele pretrage, odnosno procesa optimizacije uspijeva pronaći uglavnom prikladne vrijednosti faktora skaliranja i stope križanja.



Slika 3.5: Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 30$. Usporedba algoritama s fiksnim postavkama faktora skaliranja i stope križanja te s njihovim samopodešavanja predloženim postupkom.

3.4.3 Usporedba s DE$_{(aDE)}$, DE$_{(SLADE)}$ i DE$_{(SspDE)}$

U literaturi postoji mnoštvo različitih postupaka za podešavanje parametara DE, a u prvom redu stope križanja i faktora skaliranja, koji se razlikuju manje ili više u načinu rada. Stoga je provedena usporedba s nekoliko odabranih postupaka koji na različite načine obavljaju podešavanje tijekom izvođenja algoritma odnosno pretrage. Za potrebe testiranja i analize, u standardni DE ugrađen je postupak iz [93] (označeno s DE$_{(aDE)}$), postupak iz [164] (označeno s DE$_{(SLADE)}$) te postupak iz [97] (označeno s DE$_{(SspDE)}$). Navedeni postupci podešavanja faktora skaliranja i stope križanja opisani su na početku ovog poglavlja.

Ostvareni rezultati u smislu prosječnih grešaka optimizacije, njihovih standardnih devijacija te stopa uspješnosti prikazani su tablicama 3.5, 3.6 i 3.7. Kao i ranije, kako bi se uspostavilo radi li se o statistički značajnim razlikama prosjeka, proveden je Wilcoxonov test ranga s predznakom uz interval pouzdanosti od 95 %. Znakom $\downarrow$ je naznačeno da se radi o statistički značajnoj razlici u korist algoritma s ugrađenim predloženim postupkom DE$_{(ASP)}$ u odnosu na dani suparnički, suprotno je naznačeno znakom $\uparrow$, a znakom $\circ$ je označeno da nema statističkog značaja u razlici. Rezultati statističkih testova sažeto su prikazani na dnu tablica kao ukupni brojevi (ukupan broj pojava pojedinih slučajeva statističkog testa u odnosu na pojedine suparničke algoritme). Prema rezultatima prikazanima u tablicama, DE$_{(ASP)}$ je vrlo konkurentan u odnosu na ostale upotrijebljene postupke, a ukupno gledano, uspijeva ostvariti bolje ili usporedive rezultate. Njegova učinkovitost najbolje se ogleda u vrlo rijetko ostvarenim većim prosjecima grešaka optimizacije koji su statistički značajni. Razlike su najviše izražene u slučajevima standardnih testnih funkcija i funckija CEC 2014

Tablica 3.5: Rezultati na standardnim testnim funkcijama. Usporedba predloženog i postupaka podešavanja faktora skaliranja i stope križanja iz literature.

f	$\mathrm{DE}_{(\mathrm{aDE})}$		$\mathrm{DE}_{(\mathrm{SLADE})}$		$\mathrm{DE}_{(\mathrm{SspDE})}$		$\mathrm{DE}_{(\mathrm{ASP})}$	
	prosjek ± std. dev.	SR	prosjek ± std. dev.	SR	prosjek ± std. dev.	SR	prosjek ± std. dev.	SR
1	6.689E−82↓ ± 3.98E−81	100	1.018E−67↓ ± 7.30E−68	100	4.943E−83↓ ± 2.78E−82	100	**6.207E−88** ± 2.12E−87	100
2	2.781E−40↓ ± 2.72E−40	100	5.150E−38↓ ± 2.66E−38	100	8.536E−42↓ ± 1.07E−41	100	**3.147E−43** ± 7.19E−43	100
3	**4.986E−05**↑ ± 1.71E−04	0	1.250E+03↓ ± 5.94E+02	0	2.500E−01↓ ± 7.18E−01	0	5.065E−05 ± 6.85E−05	0
4	8.028E+00↓ ± 2.17E+00	0	5.992E+00↓ ± 3.74E+00	0	**1.755E−03**↑ ± 6.42E−03	0	3.215E−03 ± 6.78E−03	0
5	**1.616E−01**↑ ± 5.93E−01	8	1.005E+00○ ± 1.62E+00	0	2.226E+00↓ ± 1.33E+00	4	9.545E−01 ± 7.34E−01	2
6	3.868E−109↓ ± 2.76E−108	100	**2.222E−120**↑ ± 9.84E−120	100	7.339E−106↓ ± 4.93E−105	100	1.716E−109 ± 9.65E−109	100
7	2.566E−25↑ ± 6.83E−25	100	8.513E−06↓ ± 7.84E−06	0	**9.916E−28**↑ ± 4.59E−27	100	7.097E−24 ± 2.72E−23	100
8	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00 ± 0.00E+00	100
9	2.177E−97↓ ± 1.54E−96	100	1.472E−102↓ ± 3.20E−102	100	9.727E−105↓ ± 6.94E−104	100	**2.198E−118** ± 5.87E−118	100
10	**3.406E−104**↓ ± 1.77E−103	100	3.661E−102↑ ± 2.01E−101	100	1.851E−97↓ ± 1.12E−96	100	3.429E−98 ± 1.50E−97	100
11	1.001E−16↓ ± 3.33E−17	100	**0.000E+00**↑ ± 0.00E+00	100	5.660E−17↓ ± 5.61E−17	100	1.524E−17 ± 3.86E−17	100
12	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00 ± 0.00E+00	100
13	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00 ± 0.00E+00	100
14	5.599E−15↓ ± 1.79E−15	100	**3.997E−15**○ ± 0.00E+00	100	**3.997E−15**○ ± 0.00E+00	100	**3.997E−15** ± 0.00E+00	100
15	8.216E−04↓ ± 2.59E−03	90	**0.000E+00**○ ± 0.00E+00	100	**0.000E+00**○ ± 0.00E+00	100	**0.000E+00** ± 0.00E+00	100
16	7.485E−06↓ ± 2.47E−05	67	2.950E−10↓ ± 6.50E−10	100	1.359E−12↓ ± 1.80E−12	100	**4.904E−36** ± 2.58E−35	100
17	6.003E+00↓ ± 6.09E−01	0	5.698E+00↓ ± 3.39E−01	0	4.810E+00↓ ± 2.12E−01	0	**4.512E+00** ± 2.87E−01	0
18	5.564E+00↓ ± 9.96E−01	0	**1.438E−01**○ ± 3.16E−01	71	2.946E+00↓ ± 7.77E−01	0	1.800E−01 ± 5.23E−01	80
19	9.423E−33○ ± 0.00E+00	100	9.423E−33○ ± 0.00E+00	100	9.423E−33○ ± 0.00E+00	100	9.423E−33 ± 0.00E+00	100
20	**9.987E−02**○ ± 0.00E+00	0	9.987E−02○ ± 2.79E−08	0	**9.987E−02**○ ± 0.00E+00	0	**9.987E−02** ± 0.00E+00	0
21	1.220E−03○ ± 8.71E−03	98	**9.423E−33**○ ± 0.00E+00	100	**9.423E−33**○ ± 0.00E+00	100	**9.423E−33** ± 0.00E+00	100
22	1.725E−32○ ± 2.66E−32	100	**1.350E−32**○ ± 0.00E+00	100	**1.350E−32**○ ± 0.00E+00	100	**1.350E−32** ± 0.00E+00	100
23	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00 ± 0.00E+00	100
24	**5.988E−09**○ ± 0.00E+00	100	1.278E−07↓ ± 3.54E−07	100	**5.988E−09**○ ± 0.00E+00	100	**5.988E−09** ± 0.00E+00	100
25	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00 ± 0.00E+00	100
26	3.997E−15○ ± 0.00E+00	100	3.997E−15○ ± 0.00E+00	100	3.997E−15○ ± 0.00E+00	100	3.997E−15 ± 0.00E+00	100
27	**0.000E+00**○ ± 0.00E+00	100	**0.000E+00**○ ± 0.00E+00	100	4.675E−03○ ± 2.34E−02	96	**0.000E+00** ± 0.00E+00	100
28	3.209E−07○ ± 0.00E+00	100	3.209E−07○ ± 0.00E+00	100	3.209E−07○ ± 0.00E+00	100	3.209E−07 ± 0.00E+00	100
29	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00 ± 0.00E+00	100
30	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00 ± 0.00E+00	100
—	12		9		11		↓	
—	15		18		17		○	
—	3		3		2		↑	

za $d = 10$, dok su za $d = 30$ nešto manje, barem u odnosu na $\mathrm{DE}_{(\mathrm{aDE})}$ i $\mathrm{DE}_{(\mathrm{SspDE})}$. To ne predstavlja veliko iznenađenje, jer se radi o iznimno složenim instancama problema. Nadalje, slike 3.6, 3.7 i 3.8 pružaju uvid u ponašanje korištenih algoritama u smislu konvergencije. Prikazani su, kao i prethodno, medijani izvođenja na danim funkcijama. Slike sugeriraju

Tablica 3.6: Rezultati na funkcijama CEC 2014 za $d=10$. Usporedba predloženog i postupaka podešavanja faktora skaliranja i stope križanja iz literature.

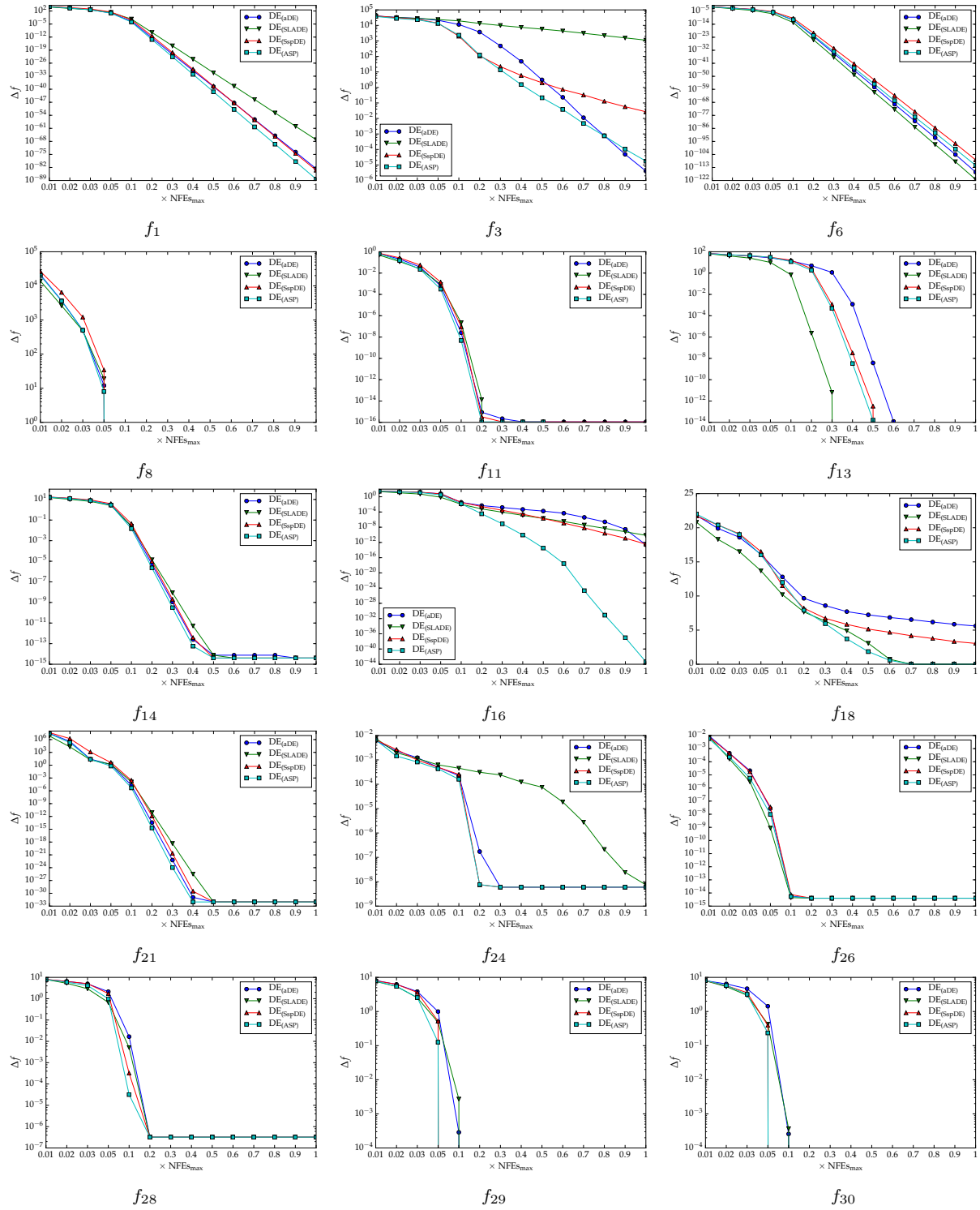
f	$DE_{(aDE)}$		$DE_{(SLADE)}$		$DE_{(SspDE)}$		$DE_{(ASP)}$	
	prosjek ± std. dev.	SR	prosjek ± std. dev.	SR	prosjek ± std. dev.	SR	prosjek ± std. dev.	SR
1	2.999E−05↓ ± 8.52E−05	10	6.584E+03↓ ± 3.72E+03	0	1.598E+00↓ ± 9.82E+00	8	**1.419E−05** ± 7.90E−05	63
2	**0.000E+00**○ ± 0.00E+00	100	5.590E−05↓ ± 1.32E−04	0	**0.000E+00**○ ± 0.00E+00	100	**0.000E+00** ○ ± 0.00E+00	100
3	**0.000E+00**○ ± 0.00E+00	100	4.451E−05↓ ± 1.09E−04	0	**0.000E+00**○ ± 0.00E+00	100	**0.000E+00** ○ ± 0.00E+00	100
4	1.210E+01○ ± 1.63E+01	25	**3.517E+00**↑ ± 9.55E+00	0	1.470E+01○ ± 1.70E+01	22	1.696E+01 ± 1.70E+01	20
5	1.906E+01↑ ± 3.69E+00	0	**1.762E+01**↑ ± 4.88E+00	0	1.895E+01↑ ± 2.90E+00	0	1.965E+01 ± 1.28E+00	0
6	**0.000E+00**○ ± 0.00E+00	100	3.827E−04↓ ± 1.06E−03	12	4.509E−06○ ± 3.21E−05	94	4.874E−08 ± 3.48E−07	98
7	4.002E−02↓ ± 2.43E−02	6	**2.966E−03**↑ ± 5.01E−03	24	1.403E−02↓ ± 9.83E−03	4	9.941E−03 ± 8.71E−03	22
8	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00 ± 0.00E+00	100
9	7.068E+00↓ ± 1.39E+00	0	**3.199E+00**○ ± 1.11E+00	0	5.429E+00↓ ± 1.26E+00	0	3.365E+00 ± 1.46E+00	0
10	2.002E−02○ ± 5.57E−02	27	6.123E−03↑ ± 2.25E−02	92	**0.000E+00**↑ ± 0.00E+00	100	1.959E−02 ± 3.42E−02	73
11	3.462E+02↓ ± 1.05E+02	0	**1.738E+02**↑ ± 7.96E+01	0	2.876E+02↓ ± 9.36E+01	0	2.147E+02 ± 1.16E+02	0
12	4.427E−01○ ± 9.41E−02	0	3.807E−01↑ ± 5.61E−02	0	**3.670E−01**↑ ± 6.87E−02	0	4.314E−01 ± 9.50E−02	0
13	1.426E−01↓ ± 3.16E−02	0	1.524E−01↓ ± 2.67E−02	0	1.322E−01↓ ± 2.86E−02	0	**1.076E−01** ± 2.33E−02	0
14	1.624E−01↓ ± 3.23E−02	0	1.845E−01↓ ± 4.05E−02	0	1.576E−01↓ ± 3.72E−02	0	**1.365E−01** ± 3.11E−02	0
15	1.148E+00↓ ± 2.25E−01	0	9.123E−01↓ ± 1.32E−01	0	9.195E−01↓ ± 1.30E−01	0	**6.533E−01** ± 1.92E−01	0
16	2.207E+00↓ ± 2.25E−01	0	**1.946E+00**○ ± 2.53E−01	0	2.072E+00○ ± 2.50E−01	0	1.969E+00 ± 2.92E−01	0
17	1.183E+01↓ ± 9.09E+00	0	2.189E+02↓ ± 1.69E+02	0	9.726E+00↓ ± 9.81E+00	0	**4.837E+00** ± 7.24E+00	0
18	**8.665E−01**○ ± 8.12E−01	0	3.352E+00↓ ± 1.76E+00	0	1.195E+00↓ ± 6.87E−01	0	9.050E−01 ± 6.41E−01	0
19	3.279E−01↓ ± 1.02E−01	0	1.681E−01↓ ± 8.14E−02	0	2.678E−01↓ ± 8.81E−02	0	**1.159E−01** ± 7.05E−02	0
20	1.061E−01↓ ± 8.43E−02	0	7.301E−02↓ ± 7.24E−02	0	2.212E−01↓ ± 9.80E−02	0	**4.724E−02** ± 4.66E−02	0
21	2.217E−01○ ± 1.93E−01	0	6.980E−01↓ ± 6.28E−01	0	3.042E−01↓ ± 2.15E−01	0	**1.645E−01** ± 2.03E−01	0
22	2.267E−01↓ ± 1.66E−01	0	**5.450E−02**↑ ± 7.67E−02	0	1.315E−01↓ ± 4.29E−02	0	8.524E−02 ± 5.36E−02	0
23	3.295E+02○ ± 0.00E+00	0	3.295E+02○ ± 0.00E+00	0	3.295E+02○ ± 2.87E−13	0	3.295E+02 ± 0.00E+00	0
24	1.143E+02↓ ± 3.40E+00	0	1.102E+02○ ± 3.31E+00	0	1.125E+02↓ ± 1.67E+00	0	**1.093E+02** ± 3.11E+00	0
25	1.284E+02↑ ± 6.64E+00	0	1.261E+02○ ± 9.87E+00	0	**1.205E+02**↑ ± 2.00E+01	0	1.284E+02 ± 2.49E+01	0
26	1.002E+02↓ ± 2.69E−02	0	1.002E+02↓ ± 3.03E−02	0	1.001E+02↓ ± 2.51E−02	0	**1.001E+02** ± 1.98E−02	0
27	**3.766E+01**↑ ± 9.79E+01	0	4.969E+01↑ ± 1.09E+02	0	5.106E+01↓ ± 1.15E+02	0	5.047E+01 ± 1.15E+02	0
28	3.695E+02○ ± 2.25E+01	0	**3.571E+02**↑ ± 7.69E−01	0	3.740E+02○ ± 3.34E+01	0	3.671E+02 ± 1.79E+01	0
29	**2.216E+02**↑ ± 3.66E+00	0	2.675E+02↓ ± 2.80E+01	0	2.221E+02↑ ± 8.83E−01	0	2.226E+02 ± 8.72E−01	0
30	**4.661E+02**↑ ± 8.66E+00	0	5.048E+02↓ ± 1.24E+01	0	4.685E+02↑ ± 1.12E+01	0	4.735E+02 ± 1.63E+01	0
—	14		15		15		↓	
—	11		6		9		○	
—	5		9		6		↑	

da je brzina konvergencije za $DE_{(ASP)}$ u pravilu bliska ili pak veća u odnosu na najbolji suparnički algoritam. Zanimljivo je primijetiti da je $DE_{(SLADE)}$ bio često osjetno manje učinkovit na unimodalnim funkcijama i to kako na standardnim tako i na funkcijama CEC 2014. Također, može se primijetiti da je $DE_{(SLADE)}$ na nekim instancama problema pružao naj-

Tablica 3.7: Rezultati na funkcijama CEC 2014 za $d=30$. Usporedba predloženog i postupaka podešavanja faktora skaliranja i stope križanja iz literature.

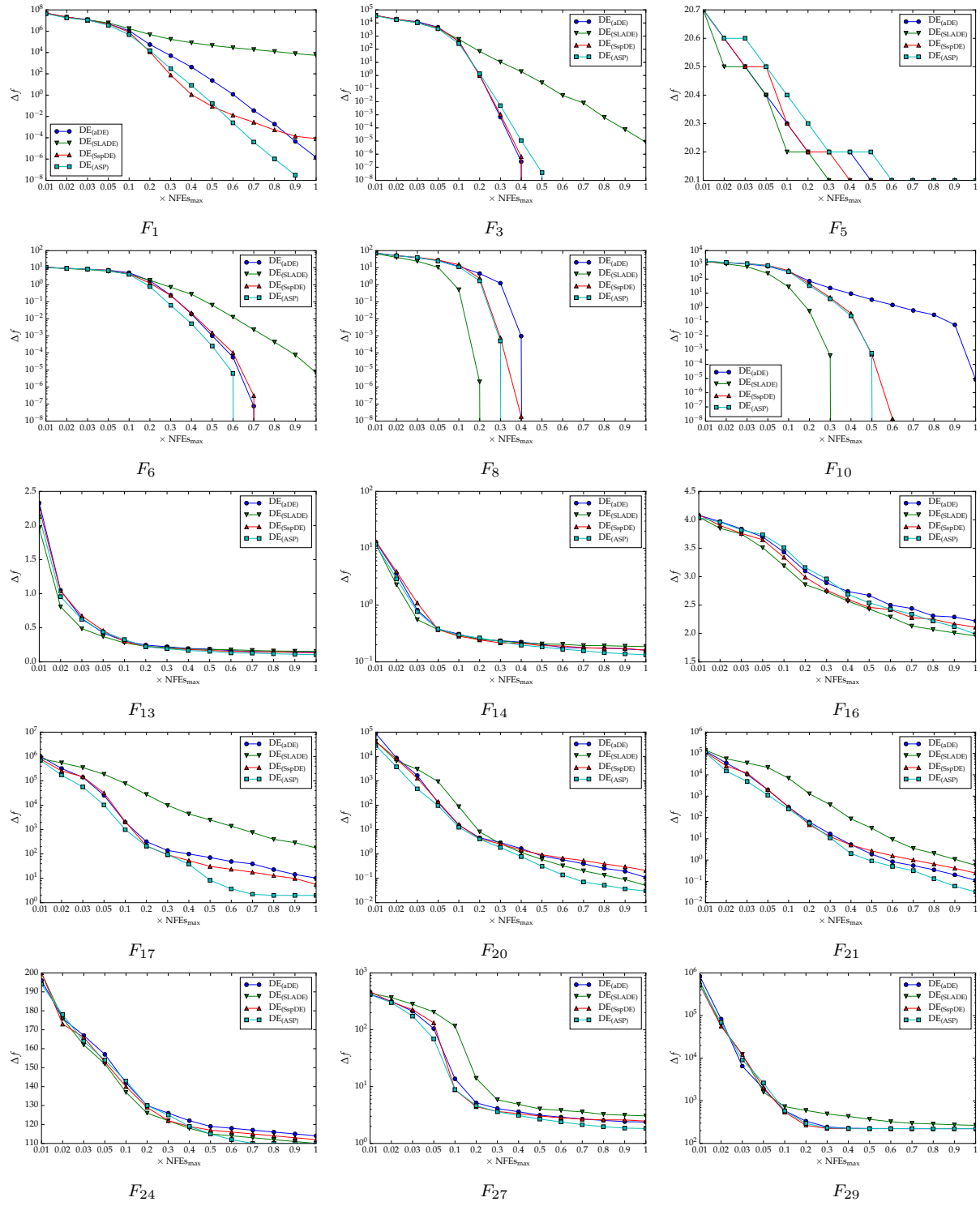
f	$\mathrm{DE}_{(\mathrm{aDE})}$		$\mathrm{DE}_{(\mathrm{SLADE})}$		$\mathrm{DE}_{(\mathrm{SspDE})}$		$\mathrm{DE}_{(\mathrm{ASP})}$	
	prosjek ± std. dev.	SR	prosjek ± std. dev.	SR	prosjek ± std. dev.	SR	prosjek ± std. dev.	SR
1	2.134E+05↓ ± 1.42E+05	0	4.298E+06↓ ± 1.83E+06	0	5.062E+05↓ ± 3.24E+05	0	**8.368E+04** ± 5.39E+04	0
2	0.000E+00∘ ± 0.00E+00	100	0.000E+00∘ ± 0.00E+00	100	0.000E+00∘ ± 0.00E+00	100	0.000E+00 ± 0.00E+00	100
3	0.000E+00∘ ± 0.00E+00	100	0.000E+00∘ ± 0.00E+00	100	0.000E+00∘ ± 0.00E+00	100	0.000E+00 ± 0.00E+00	100
4	**4.335E+01**∘ ± 3.46E+01	0	8.283E+01↓ ± 1.82E+01	0	7.575E+01↓ ± 1.86E+01	0	4.736E+01 ± 2.95E+01	0
5	2.041E+01↓ ± 4.50E−02	0	2.040E+01↓ ± 3.07E−02	0	**2.034E+01**↑ ± 3.51E−02	0	2.036E+01 ± 4.50E−02	0
6	**2.060E+00**↑ ± 3.89E+00	12	7.532E+00↓ ± 3.16E+00	0	8.797E+00↓ ± 3.52E+00	0	3.085E+00 ± 2.49E+00	0
7	3.384E−04∘ ± 1.71E−03	96	**0.000E+00**∘ ± 0.00E+00	100	**0.000E+00**∘ ± 0.00E+00	100	3.383E−04 ± 1.71E−03	96
8	**0.000E+00**↑ ± 0.00E+00	100	**0.000E+00**↑ ± 0.00E+00	100	**0.000E+00**↑ ± 0.00E+00	100	1.951E−01 ± 4.46E−01	82
9	5.720E+01↓ ± 1.07E+01	0	5.248E+01↓ ± 7.11E+00	0	4.473E+01↓ ± 5.31E+00	0	**4.109E+01** ± 8.26E+00	0
10	2.520E+01↓ ± 1.87E+01	0	3.143E−02↑ ± 2.18E−02	14	**8.164E−04**↑ ± 4.08E−03	96	4.641E−01 ± 6.72E−01	0
11	3.016E+03↓ ± 3.22E+02	0	3.008E+03↓ ± 2.09E+02	0	2.469E+03↓ ± 2.66E+02	0	**1.894E+03** ± 3.42E+02	0
12	5.671E−01↓ ± 8.88E−02	0	5.748E−01↓ ± 6.92E−02	0	4.707E−01∘ ± 5.34E−02	0	**4.486E−01** ± 7.21E−02	0
13	3.024E−01↓ ± 4.21E−02	0	3.076E−01↓ ± 3.79E−02	0	2.991E−01↓ ± 3.33E−02	0	**2.203E−01** ± 3.69E−02	0
14	2.681E−01↓ ± 2.67E−02	0	2.721E−01↓ ± 2.82E−02	0	2.756E−01↓ ± 2.63E−02	0	**2.429E−01** ± 2.53E−02	0
15	7.290E+00↓ ± 1.17E+00	0	7.448E+00↓ ± 7.15E−01	0	6.020E+00↓ ± 7.58E−01	0	**3.622E+00** ± 8.70E−01	0
16	1.033E+01↓ ± 4.72E−01	0	1.029E+01↓ ± 2.95E−01	0	9.913E+00↓ ± 3.06E−01	0	**9.704E+00** ± 3.42E−01	0
17	**1.835E+03**∘ ± 1.53E+03	0	2.183E+05↓ ± 1.48E+05	0	4.545E+03↓ ± 7.11E+03	0	1.906E+03 ± 1.00E+03	0
18	**1.850E+01**↑ ± 9.17E+00	0	6.392E+02↓ ± 7.28E+02	0	2.603E+01↑ ± 1.55E+01	0	5.706E+01 ± 2.14E+01	0
19	4.589E+00↓ ± 6.00E−01	0	5.258E+00↓ ± 5.78E−01	0	4.662E+00↓ ± 5.56E−01	0	**3.698E+00** ± 7.80E−01	0
20	**1.417E+01**↑ ± 6.48E+00	0	5.603E+01↓ ± 2.13E+01	0	1.814E+01↑ ± 4.22E+00	0	2.152E+01 ± 1.03E+01	0
21	4.534E+02∘ ± 2.80E+02	0	1.537E+04↓ ± 1.37E+04	0	5.475E+02↓ ± 1.81E+02	0	**3.922E+02** ± 1.98E+02	0
22	**3.201E+01**↑ ± 1.04E+01	0	1.168E+02↓ ± 6.20E+01	0	1.542E+02↓ ± 5.75E+01	0	7.876E+01 ± 6.11E+01	0
23	3.152E+02∘ ± 0.00E+00	0	3.152E+02∘ ± 0.00E+00	0	3.152E+02∘ ± 5.74E−14	0	3.152E+02 ± 0.00E+00	0
24	2.252E+02∘ ± 2.95E+00	0	2.250E+02∘ ± 1.26E+00	0	**2.245E+02**∘ ± 2.21E+00	0	2.247E+02 ± 1.47E+00	0
25	**2.030E+02**↑ ± 3.50E−01	0	2.055E+02↓ ± 8.79E−01	0	2.036E+02↑ ± 6.63E−01	0	2.040E+02 ± 1.02E+00	0
26	1.003E+02↓ ± 4.02E−02	0	1.003E+02↓ ± 3.33E−02	0	1.003E+02↓ ± 3.89E−02	0	**1.002E+02** ± 3.83E−02	0
27	3.582E+02∘ ± 5.00E+01	0	3.780E+02↓ ± 4.66E+01	0	**3.437E+02**∘ ± 4.98E+01	0	3.468E+02 ± 4.70E+01	0
28	8.114E+02↑ ± 2.53E+01	0	**7.803E+02**↑ ± 1.87E+01	0	7.963E+02↑ ± 2.30E+01	0	8.260E+02 ± 2.70E+01	0
29	8.808E+02↑ ± 1.17E+02	0	1.301E+03↓ ± 1.77E+02	0	**8.190E+02**↑ ± 1.03E+02	0	9.558E+02 ± 1.21E+02	0
30	**1.161E+03**∘ ± 4.73E+02	0	1.795E+03↓ ± 6.69E+02	0	1.182E+03∘ ± 3.42E+02	0	1.379E+03 ± 5.34E+02	0
—	12		22		14		↓	
—	10		5		8		∘	
—	8		3		8		↑	

veću brzinu konvergencije, ali na nekim drugima uvjerljivo najmanju. Navedeno sugerira da stalne zamjene parametara (faktora mutacije i stope križanja) pa čak kada se radi o vrijednostima u okolini prethodno dobrih, nisu na određenim vrstama problema učinkovite (prema dobivenim rezultatima, posebno u slučaju unimodalnih funkcija).



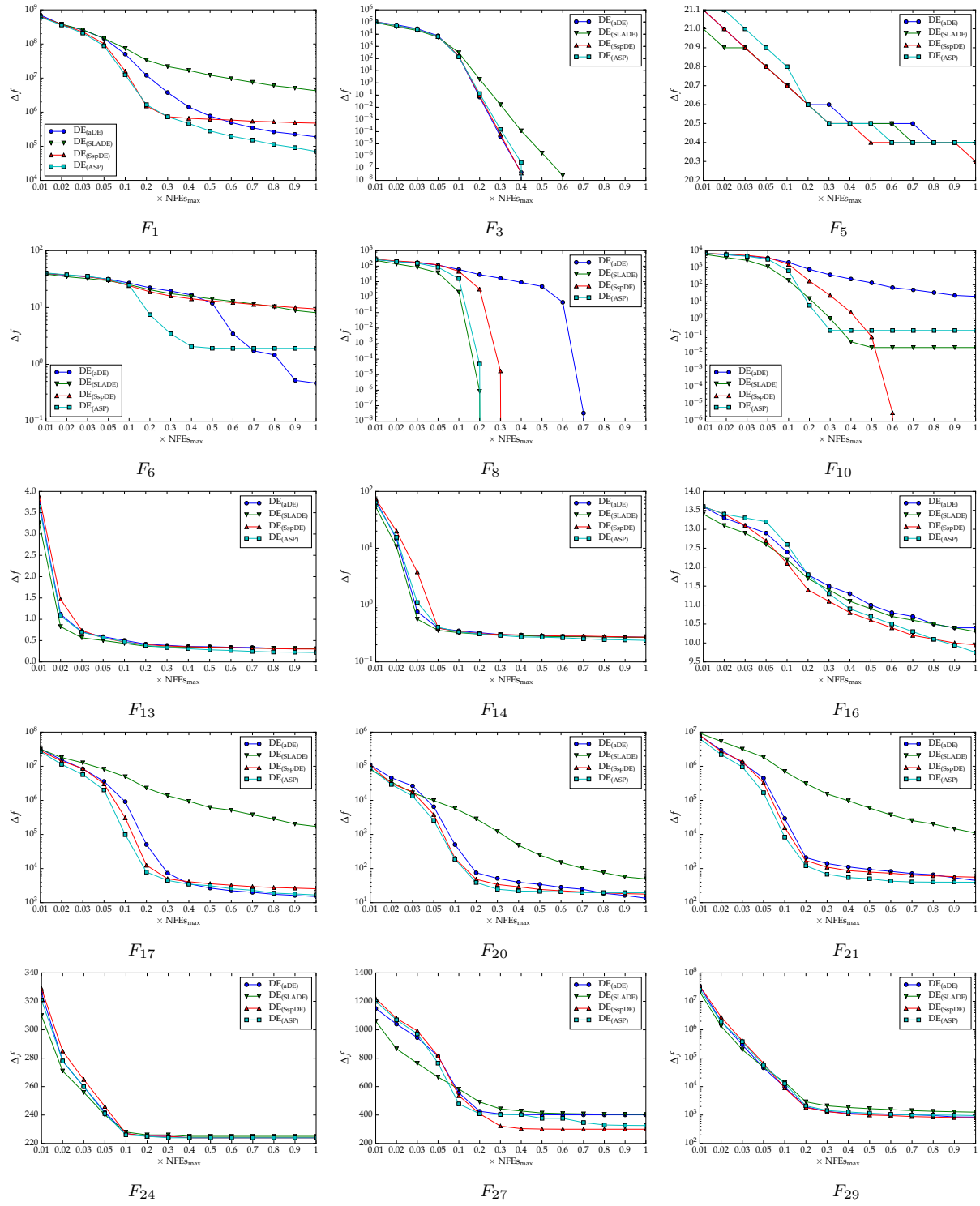
Slika 3.6: Grafovi konvergencije na nekoliko odabranih standardnih testnih funkcija. Usporedba predloženog i postupaka podešavanja faktora skaliranja i stope križanja iz literature.

S obzirom da svaki postupak podešavanja parametara podiže vremensku složenost algoritma u koji je ugrađen, može se postaviti pitanje koliko postupci upotrijebljeni u usporedbi utječu na navedenu složenost. Odgovor na to pitanje pruža provedena procjena složenosti algoritama prema [70]. Za razliku od uputa danih u [70], procjena je izvršena na četiri



Slika 3.7: Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 10$. Usporedba predloženog i postupaka podešavanja faktora skaliranja i stope križanja iz literature.

funkcije umjesto na jednoj kako bi se pružio na što višoj razini neovisan uvid u vremensku složenost. Također, cijeli postupak procjene na svakoj funkciji ponavljan je 25 puta umjesto jednom, a na kraju je uzet medijan. Na taj način znatno je smanjen utjecaj varijacija u vremenima izvođenja. Cijeli postupak procjene vremenske složenosti algoritama opisan je u



Slika 3.8: Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 30$. Usporedba predloženog i postupaka podešavanja faktora skaliranja i stope križanja iz literature.

dodatku A. Dobiveni rezultati prikazani su u tablicama 3.8 i 3.9, a pružaju uvid u relativne (u odnosu na cijenu vrednovanja odgovarajuće funkcije cilja) složenosti i njihovu ovisnost od dimenzionalnosti problema.

Prema prikazanim rezultatima, jasno se vidi da ugradnja nekog od upotrijebljenih postu-

Tablica 3.8: Procjena složenosti na standardnim testnim funkcijama za predloženi i postupke podešavanja faktora skaliranja i stope križanja iz literature.

Alg.	$d=10$				$d=30$				$d=50$			
	f_3	f_8	f_{12}	f_{17}	f_3	f_8	f_{12}	f_{17}	f_3	f_8	f_{12}	f_{17}
$\text{DE}_{(0.5,\,0.9)}$	2.50	2.59	3.06	2.96	6.15	6.34	7.96	7.65	9.84	10.15	12.75	12.42
$\text{DE}_{(\text{aDE})}$	2.59	2.82	3.21	3.08	6.38	6.54	8.07	7.91	10.28	10.45	13.04	12.73
$\text{DE}_{(\text{SLADE})}$	3.82	3.65	4.18	4.26	7.48	7.38	8.94	8.89	11.13	11.13	13.67	13.48
$\text{DE}_{(\text{SspDE})}$	3.18	3.26	3.73	3.21	6.89	6.98	8.33	8.16	10.42	10.62	13.12	12.48
$\text{DE}_{(\text{ASP})}$	3.48	3.24	3.89	4.03	7.24	7.00	8.66	8.55	10.81	10.79	13.53	13.21

Tablica 3.9: Procjena složenosti na funkcijama CEC 2014 za predloženi i postupke podešavanja faktora skaliranja i stope križanja iz literature.

Alg.	$d=10$				$d=30$			
	F_5	F_{11}	F_{14}	F_{18}	F_5	F_{11}	F_{14}	F_{18}
$\text{DE}_{(0.5,\,0.9)}$	2.52	3.07	2.51	2.53	6.27	6.43	6.16	6.18
$\text{DE}_{(\text{aDE})}$	2.86	3.12	2.65	3.08	6.57	7.30	6.58	6.37
$\text{DE}_{(\text{SLADE})}$	3.85	4.48	3.96	3.88	7.66	8.20	7.69	7.46
$\text{DE}_{(\text{SspDE})}$	2.72	3.36	2.76	3.01	6.37	6.75	6.27	6.55
$\text{DE}_{(\text{ASP})}$	3.56	4.05	3.62	3.53	7.40	7.44	7.81	7.08

paka podešavanja parametara ne podiže osjetno vremensku složenost u odnosu na algoritam u koji su ugrađeni. Također, s obzirom na linearni porast složenosti povećanjem dimenzionalnosti, jasno je da niti jedan od njih ne ovisi o dimenzionalnosti. Zanimljivo je primijetiti da velika prostorna složenost postupka ugrađenog u $\text{DE}_{(\text{SspDE})}$ ne utječe negativno na njegovu vremensku složenost. Na koncu, najveće vremenske složenosti mogu se pripisati postupcima korištenim u $\text{DE}_{(\text{SLADE})}$ i $\text{DE}_{(\text{ASP})}$, redom. To se može pripisati češćom uporabom PRNG u odnosu na druga dva upotrijebljena postupka. U tom pogledu, dodatno vrijeme računanja troše i stvaranja slučajnih brojeva po Gaussovoj i Cauchyjevoj razdiobi.

3.4.4 Utjecaj broja pohranjenih vrijednosti parametara

Kao što je prethodno navedeno, ključnu ulogu u predloženom postupku samopodešavanja stope križanja i faktora skaliranja igraju upravo popisi u koje se pohranjuju uspješne vrijednosti tih parametara. Kako se u prvom redu nove vrijednosti parametara dobivaju na temelju pohranjenih, veličina popisa r može imati znatan utjecaj na ponašanje postupka.

Tablica 3.10: Prosječni rangovi i stope uspješnosti na standardnim funkcijama u ovisnosti o veličini popisa r.

r	$f_1 \sim f_{11}$	$f_{12} \sim f_{22}$	$f_{23} \sim f_{30}$
	$\overline{R}$ ($\overline{SR}$)	$\overline{R}$ ($\overline{SR}$)	$\overline{R}$ ($\overline{SR}$)
2	3.9 (72.9)	3.4 (75.4)	3.2 (99.8)
3	3.4 (72.9)	2.9 (80.0)	**2.9** (100)
5	2.5 (72.7)	**2.7** (80.0)	3.2 (99.8)
7	**2.4** (72.7)	2.9 (78.1)	**2.9** (100)
10	2.8 (72.7)	3.0 (76.1)	**2.9** (100)

Tablica 3.11: Prosječni rangovi i stope uspješnosti na funkcijama CEC 2014 za $d=10$ u ovisnosti o veličini popisa r.

r	$F_1 \sim F_3$	$F_4 \sim F_{16}$	$F_{17} \sim F_{22}$	$F_{23} \sim F_{30}$
	$\overline{R}$ ($\overline{SR}$)	$\overline{R}$ ($\overline{SR}$)	$\overline{R}$ ($\overline{SR}$)	$\overline{R}$ ($\overline{SR}$)
2	**2.0** (96.1)	**2.3** (26.1)	3.0 (0.3)	3.2 (0.0)
3	2.3 (87.6)	3.2 (24.0)	2.5 (0.0)	2.7 (0.0)
5	2.7 (73.2)	2.7 (25.5)	**1.8** (0.0)	**2.2** (0.0)
7	3.0 (66.7)	3.1 (25.0)	3.3 (0.0)	2.8 (0.2)
10	5.0 (58.2)	3.8 (23.2)	4.3 (0.0)	4.2 (0.0)

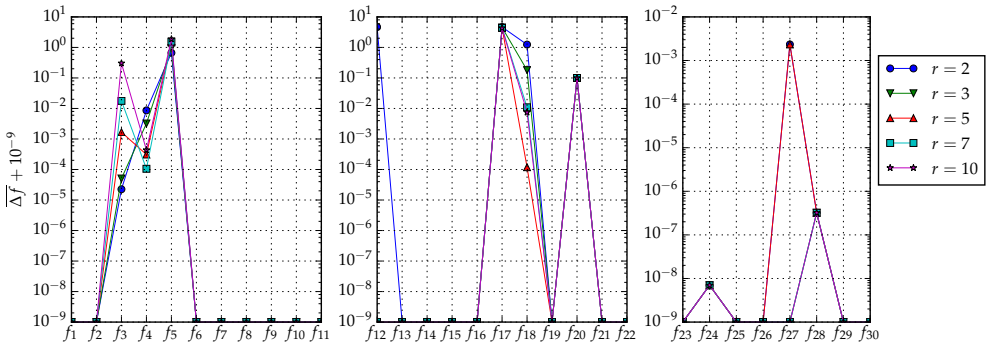
Tablica 3.12: Prosječni rangovi i stope uspješnosti na funkcijama CEC 2014 za $d=30$ u ovisnosti o veličini popisa r.

r	$F_1 \sim F_3$	$F_4 \sim F_{16}$	$F_{17} \sim F_{22}$	$F_{23} \sim F_{30}$
	$\overline{R}$ ($\overline{SR}$)	$\overline{R}$ ($\overline{SR}$)	$\overline{R}$ ($\overline{SR}$)	$\overline{R}$ ($\overline{SR}$)
2	**2.0** (66.7)	**2.2** (12.5)	**2.2** (0.0)	2.9 (0.0)
3	2.3 (66.7)	2.8 (13.7)	2.5 (0.0)	3.6 (0.0)
5	3.3 (66.7)	3.0 (14.9)	2.7 (0.0)	2.9 (0.0)
7	3.7 (58.2)	3.1 (15.4)	3.2 (0.0)	3.0 (0.0)
10	3.7 (34.0)	3.9 (15.5)	4.5 (0.0)	**2.6** (0.0)

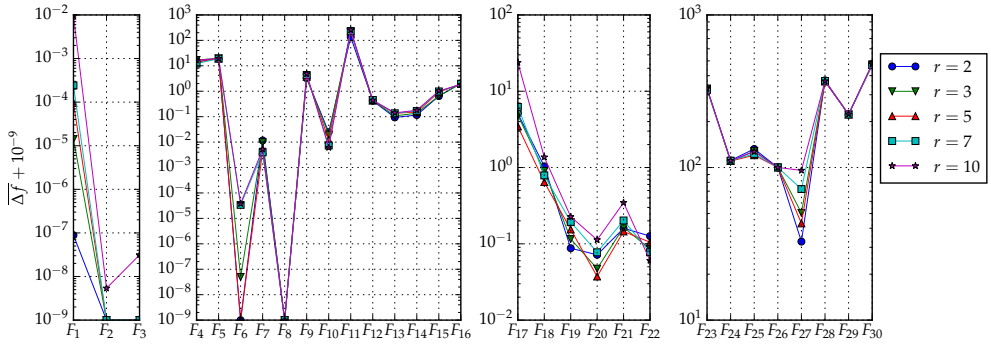
Stoga su provedeni testiranje i analiza s gledišta razine utjecaja broja pohranjenih vrijednosti parametara, odnosno veličina popisa $\mathcal{A}_j^F$ i $\mathcal{A}_j^{CR}$ na ponašanje predloženog postupka.

Tablice 3.10, 3.11 i 3.12 prikazuju razlike u učinkovitosti predloženog postupka u ovisnosti o veličini popisa $\mathcal{A}_j^{CR}$ i $\mathcal{A}_j^F$ u smislu postignutih rangova i stopa uspješnosti. U tablicama su prikazane prosječne vrijednosti po ranije navedenim razredima funkcija. Inačice postupka s različitim vrijednostima za veličinu r međusobno su rangirane (prema prosječnim ostvarenim greškama optimizacije) po svakoj funkciji pri čemu je najmanji rang dodijeljen najboljoj inačici i tako redom. Ako je bilo više istih rangova, odgovarajućim inačicama dodijeljena je srednja vrijednost tih rangova.

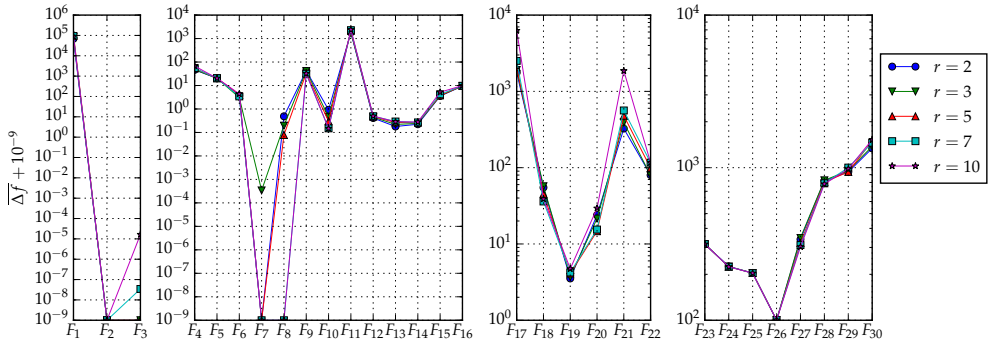
Kao što se može primijetiti iz prikazanih rezultata, za različite razrede funkcija najmanji prosječni rangovi su postignuti s različitim veličinama r. U slučaju standardnih testnih funkcija, cjelokupno gledano, bolji rezultati postignuti su za $r > 3$. Ovo se posebno odnosi na skalabilne, uni- i multimodalne funkcije $f_1 \sim f_{22}$, dok su razlike u prosječnim rangovima $\overline{R}$ i prosječnim stopama uspješnosti $\overline{SR}$ beznačajne na multimodalnim funkcijama male dimenzionalnosti $f_{23} \sim f_{30}$. Nasuprot tome, na funkcijama CEC 2014 za $d = 10$ i $d = 30$ većinom su najmanji $\overline{R}$ ostvareni pri vrijednosti $r = 2$, iako se to ne ogleda uvijek i u $\overline{SR}$. Međutim, promatranjem slika 3.9, 3.10 i 3.11, koje prikazuju prosječne ostvarene greške optimizacije $\overline{\Delta f}$ po funkcijama, može se vidjeti kako se najčešće ne radi o značajnim ili pak vidljivim razlikama u kvaliteti. Zbog zornijeg prikaza, na sve $\overline{\Delta f}$ dodana je vrijednost 10^{-9}. Nadalje,



Slika 3.9: Prosječne greške optimizacije na standardnim testnim funkcijama. Usporedba predloženog postupka pri različitim veličinama popisa r.



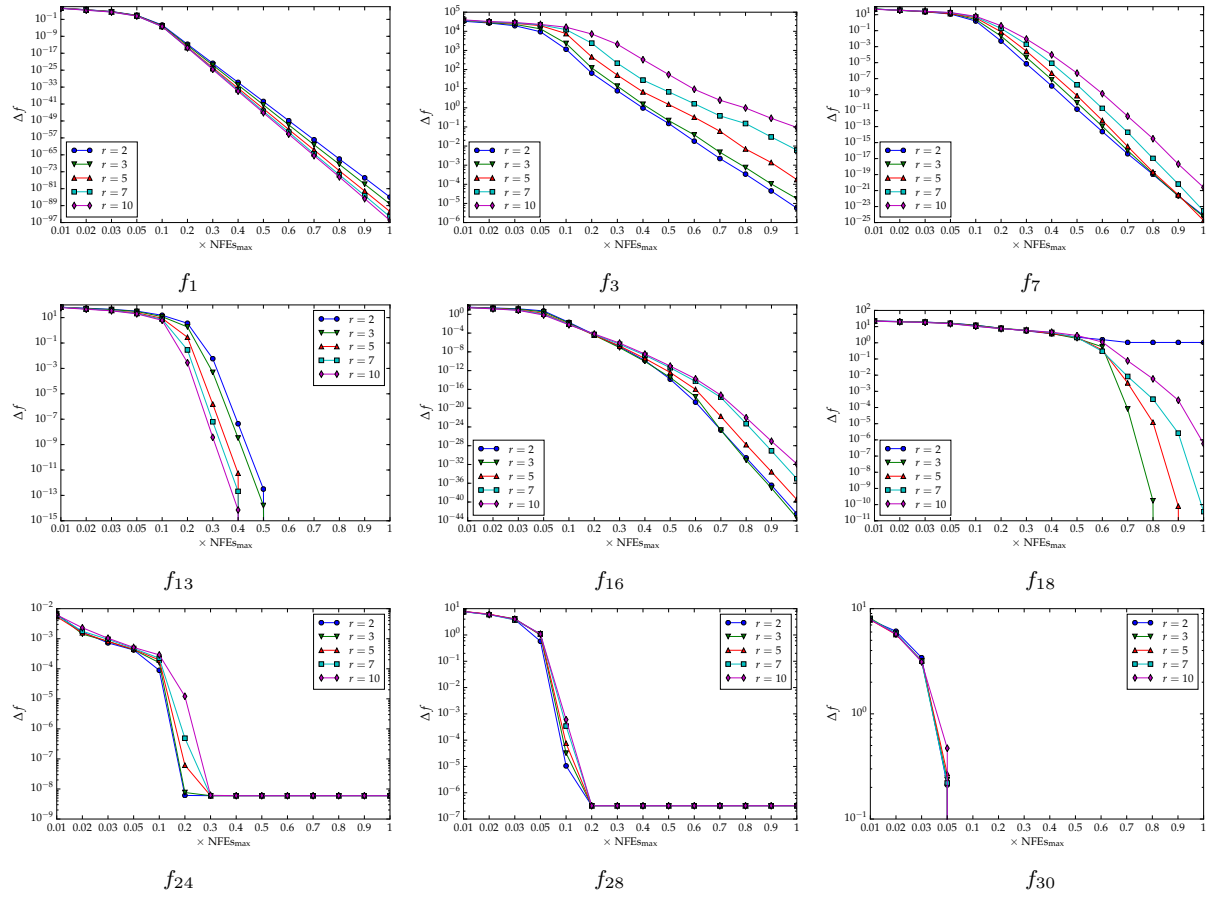
Slika 3.10: Prosječne greške optimizacije na funkcijama CEC 2014 za $d = 10$. Usporedba predloženog postupka pri različitim veličina popisa r.



Slika 3.11: Prosječne greške optimizacije na funkcijama CEC 2014 za $d = 30$. Usporedba predloženog postupka pri različitim veličina popisa r.

slike 3.12, 3.13 i 3.14 pružaju uvid u ponašanje postupka s različitim vrijednostima veličine r. One prikazuju medijan greške optimizacije svih izvođenja algoritma za danu funkciju u odnosu na NFEs. Slike ne sugeriraju značajan utjecaj broja prethodno uspješnih vrijednosti parametara na brzinu konvergencije, osim u nekoliko slučajeva, gdje je razlika vidljiva i gdje su veće brzine konvergencije postignute malim vrijednostima za r i obratno. Također, slike potvrđuje da se većinom radi o malim ili beznačajnim razlikama konačno postignutih grešaka optimizacije.

Analiza prikazanih rezultata dovodi do zaključka da predloženi postupak samopodešavanja faktora skaliranja i stope križanja nije posebno osjetljiv na veličine popisa $\mathcal{A}_j^F$ i $\mathcal{A}_j^{CR}$, odnosno na broj prethodno uspješnih vrijednosti koje se koriste za generiranje novih. Međutim, oni sugeriraju kako u određenim situacijama veći popisi mogu biti pogodniji u odnosu na manje i obratno. Ipak, općenito dvije najmanje upotrijebljene veličine $r = 2$ i $r = 3$ pokazale su se najkorisnijima, a najveća upotrijebljena $r = 10$ najmanje korisnom. Ovo podupire prethodnu pretpostavku da manje vrijednosti omogućuju brže promjene, a veće vrijednosti nedovoljno brze za praćenje stanja pretrage. Manje veličine popisa $\mathcal{A}_j^F$ i $\mathcal{A}_j^{CR}$ također idu u prilog i manjoj prostornoj složenosti predloženog postupka.



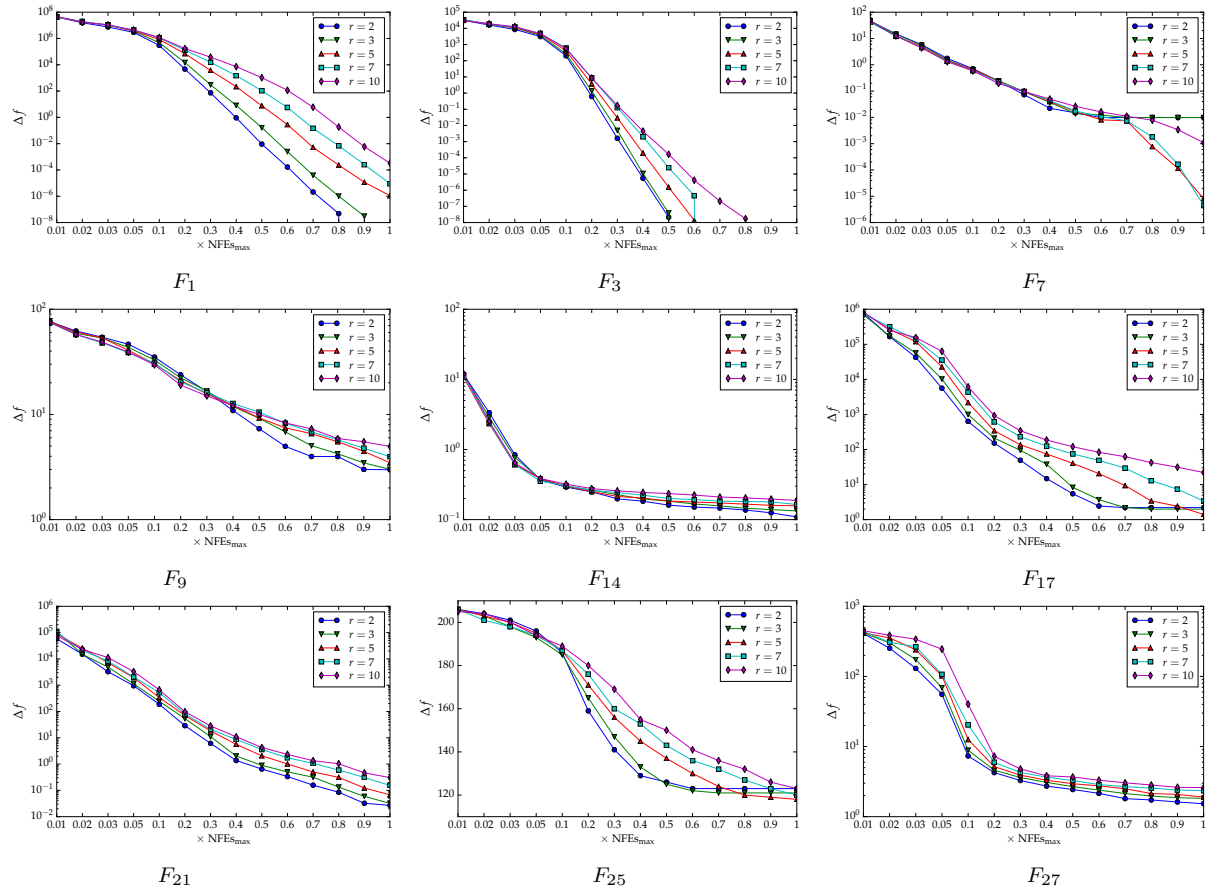
Slika 3.12: Grafovi konvergencije na nekoliko odabranih standardnih testnih funkcija. Usporedba predloženog postupka pri različitim veličinama popisa r.

3.4.5 Ponašanje tijekom procesa optimizacije

Predloženi postupak samopodešavanje faktora skaliranja i stope križanja vrši stalne izmjene tih parametara s ciljem pronalaska pogodnih vrijednosti. Shodno tome, kako bi se omogućio uvid u njegovo ponašanje, bilježen je niz podataka tijekom jednog izvođenja. U prvom redu bilježene su vrijednosti parametara koje su rezultirale unapređenjem do tada najboljeg rješenja. Uz navedeno, bilježeni su prosjeci uspješnih (potomak prešao u narednu generaciju) i neuspješnih vrijednosti parametara. Nadalje, praćen je i njihov kumulativni broj. S obzirom da ključnu ulogu u predloženom postupku imaju prethodno uspješne vrijednosti, odnosno popisi $\mathcal{A}_j^F$ i $\mathcal{A}_j^{CR}$, bilježena je njihova raznolikost (slično standardnoj devijaciji)

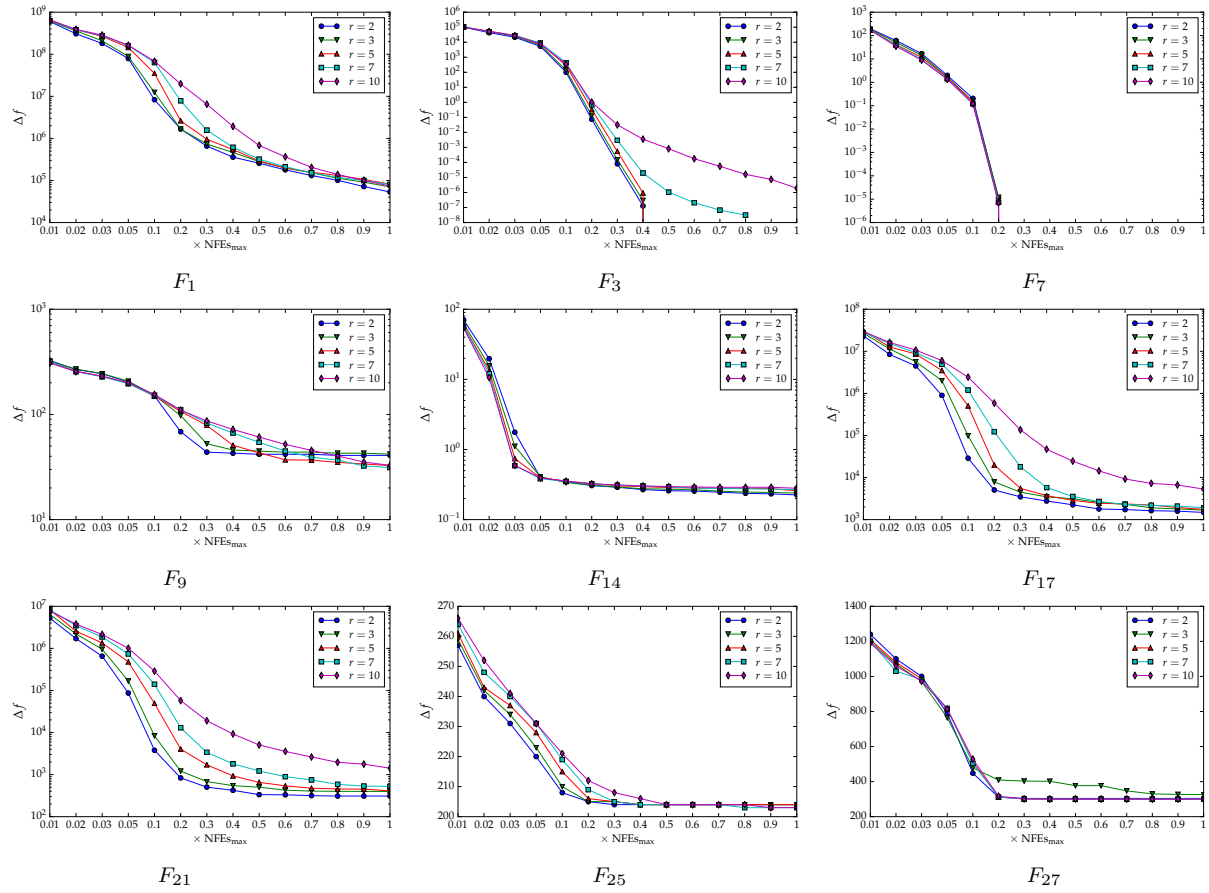
$$D(X) = \frac{1}{NP} \sum_{x \in X} d(x, \overline{x}), \quad \overline{x} = \frac{\sum\limits_{x \in X} x}{NP}, \tag{3.7}$$

gdje X odgovara skupu $\{\mathcal{A}_j^F : j = 1, \ldots, NP\}$ odnosno skupu $\{\mathcal{A}_j^{CR} : j = 1, \ldots, NP\}$, a $d(\cdot, \cdot)$ je Euklidska udaljenost. Slike 3.15 i 3.16 prikazuju navedene podatke kroz generacije za nekoliko odabranih funkcija.



Slika 3.13: Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 10$. Usporedba predloženog postupka pri različitim veličinama popisa r.

Prema prikazanom, može se vidjeti da su na različitim funkcijama različiti parovi vrijednosti parametara (označeno s $F^{(\uparrow)}$ i $CR^{(\uparrow)}$) dovodili do unaprjeđenja do tada najboljeg pronađenog rješenja. Zanimljivo je da je raspon vrijednosti jednog parametra često vrlo ograničen, a ujedno drugog nije. Također, može se primijetiti da se u slučaju nekih funkcija kroz proces pretrage ili optimizacije mijenjaju te vrijednosti (vidi grafove za f_{18} i F_9), a što upućuje na potrebu stalnog podešavanja parametara. Nadalje, može se primijetiti i da prosjeci uspješnih i neuspješnih vrijednosti (označeno s $\overline{F^{(+)}}$ i $\overline{CR^{(+)}}$ te $\overline{F^{(-)}}$ i $\overline{CR^{(-)}}$, redom), dobro prate vrijednosti $F^{(\uparrow)}$ i $CR^{(\uparrow)}$. Raznolikost popisa $\mathcal{A}_j^F$ i $\mathcal{A}_j^{CR}$ održava se tijekom izvođenja i ne ovisi značajno o broju uspješnih odnosno neuspješnih vrijednosti parametara. Padovi u raznolikosti mogu se primijetiti kada dolazi do povećanja broja uspješnih, nakon čega obično opet dolazi do porasta.

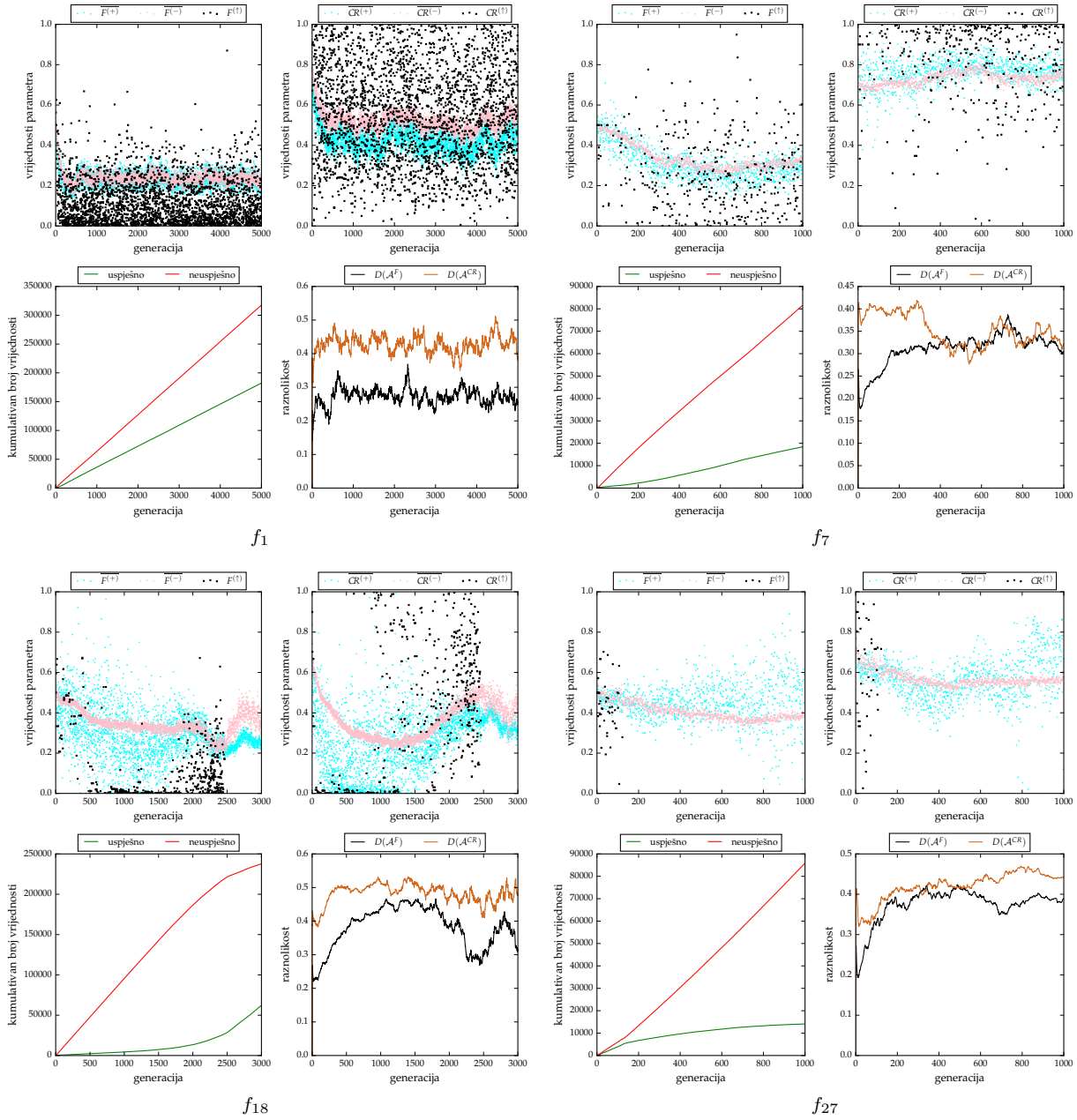


Slika 3.14: Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 30$. Usporedba predloženog postupka pri različitim veličinama popisa r.

3.5 Osvrt na podešavanje parametara i predloženi postupak

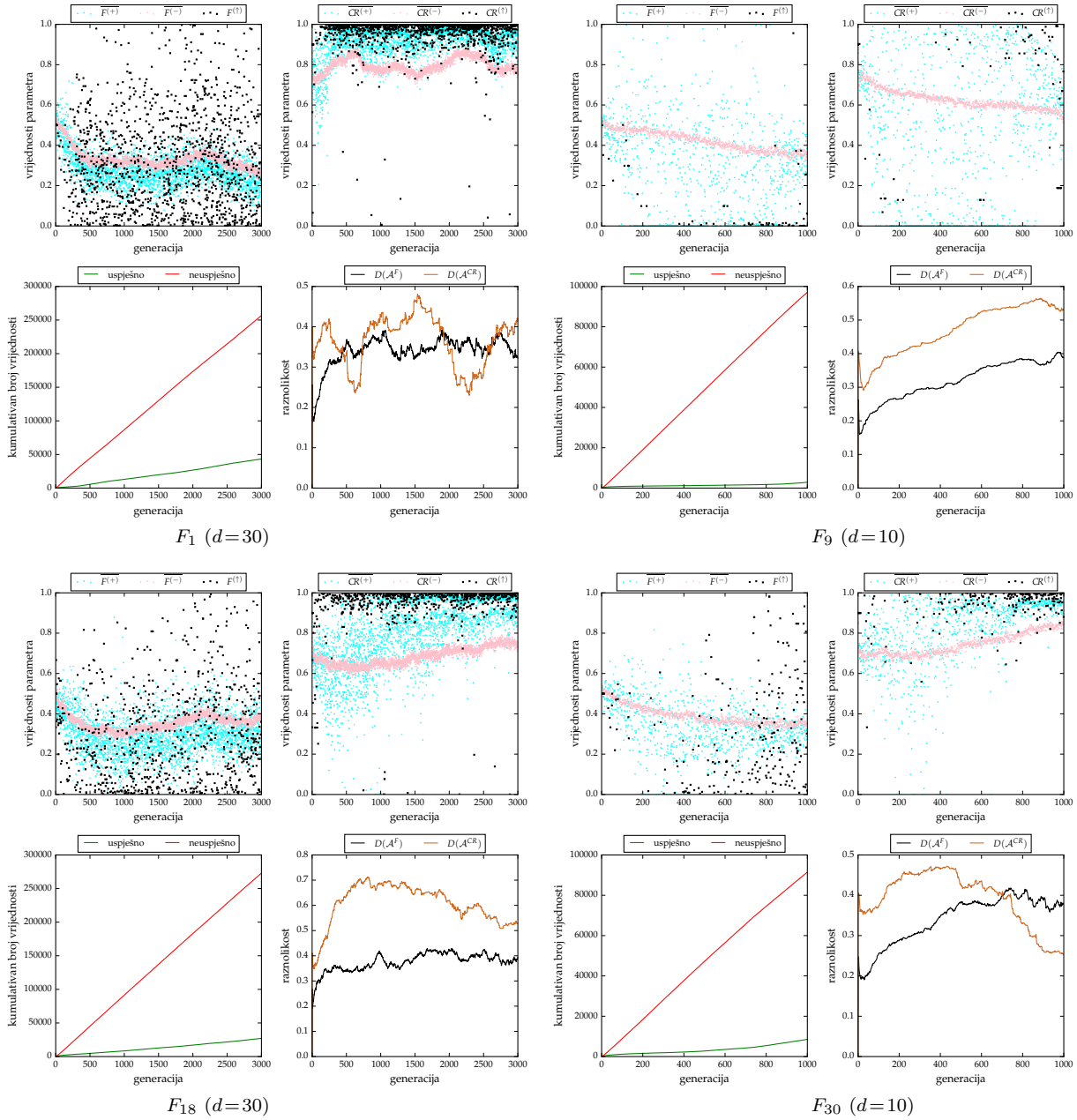
Pri postavljanju parametara DE, ali isto tako parametara drugih sličnih algoritama, nije se uvijek moguće osloniti na iste postavke ili preporuke iz literature. Drugim riječima, potrebno je tražiti pogodne vrijednosti, jer za različite probleme mogu biti nužne različite postavke parametara za ostvarivanje zadovoljavajuće učinkovitosti korištenog algoritma. Rezultati provedene eksperimentalne analize ukazuju kako se predloženim postupkom samopodešavanja faktora skaliranja i stope križanja može djelotvorno izbjeći računalno zahtjevno tražnje dobrih vrijednosti tih parametara. Nadalje, rezultati sugeriraju da se predloženim postupkom mogu postići bolja rješenja u odnosu na fiksne postavke parametara (barem u opsegu korištenih), za kojima on ne zaostaje ni u pogledu brzine konvergencije. Isto tako, oni upućuju na zaključak da je navedeni postupak, ostvarivanjem uglavnom boljih ili usporedivih rješenja, konkurentan upotrijebljenim postupcima iz literature koji na drukčije načine podešavaju spomenute parametre.

Kao jedan od nedostataka predloženog postupka samopodešavanja može se navesti broj



Slika 3.15: Proces samopodešavanja faktora skaliranja i stope križanja na standardnim testnim funkcijama.

pohranjenih prethodno uspješnih ($r = 3$) na temelju kojih se generiraju nove vrijednosti faktora skaliranja i stope križanja. Međutim, navedeni broj, odnosno veličina odgovarajućih popisa $\mathcal{A}_j^F$ i $\mathcal{A}_j^{CR}$ zadana je i fiksna, te sukladno tome ne zahtijeva izmjene ili prilagodbu. Štoviše, provedeno testiranje i analiza upućuju na otpornost predloženog postupka na male izmjene veličine r, te da značajno veće vrijednosti, u usporedbi, uglavnom usporavaju konvergenciju. Slično tome, vjerojatnost zamjene vrijednosti spomenutih parametara nasumično generiranim vrijednostima $p_n = 0.01$ također je zadana i fiksna. Osim toga, veće vrijednosti p_n mogle bi narušiti podešavanje pomoću prethodno uspješnih vrijednosti parametara.



Slika 3.16: Proces samopodešavanja faktora skaliranja i stope križanja na funkcijama CEC 2014.

Postupak samopodešavanja parametara faktora skaliranja i stope križanja DE opisan u ovom poglavlju, s obzirom na provedeno eksperimentalno testiranje, analizu te ostvarene povoljne rezultate, predstavlja ispunjenje prijedloga prvog izvornog znanstvenog doprinosa ove disertacije.

POGLAVLJE 4

Inicijalizacija populacije diferencijalne evolucije

POČETNA populacija kao bitan čimbenik evolucijskih algoritama može utjecati na brzinu ili stopu konvergencije i u konačnici na sposobnost algoritma u pronalaženju zadovoljavajućih rješenja. Ukoliko se početna populacija sastoji od dobrih rješenja, ona može odmah od početka usmjeriti pretragu prema obećavajućim dijelovima prostora. Ipak, ona se najčešće inicijalizira potpuno nasumično stvorenim rješenjima koja su u pravilu niske kvalitete. Stoga navedeno ponašanje nije za očekivati. U ovom poglavlju predlaže se metoda inicijalizacije populacije DE koja pokušava otkriti obećavajuće dijelove prostora pretrage i stvoriti (relativno) dobra rješenja u njihovoj blizini. U tu svrhu, predložena metoda koristi se grupiranjem podataka i Cauchyjevim slučajnim varijablama. Navedeni prijedlog drugog izvornog znanstvenog doprinosa, sveobuhvatno je vrednovan opsežnim testiranjem i analizom.

4.1 Motivacija

Kao i u drugim EAs te sličnim prirodom inspiriranim algoritmima, početna populacija u DE igra važnu ulogu. Ona može utjecati na brzinu konvergencije, ali i na uspješnost algoritma u pronalasku dobrih ili pak zadovoljavajućih rješenja. Najčešće se početna populacija popunjava potpuno nasumično stvorenim rješenjima [60, 77] (uporabom uniformnih slučajnih varijabli, vidi Alg. 2.2). Tako stvorene populacije su u pravilu vrlo raznolike. To predstavlja poželjno svojstvo, jer omogućuje opsežno istraživanje prostora pretrage te pomaže u izbjegavanju preuranjene konvergencije kao i zaglavljivanja u lokalnom optimumu. Međutim, takve početne populacije, općenito, sadrže isključivo rješenja male kvalitete. Također, iako takve populacije mogu obuhvatiti gotovo cijeli prostor pretrage, one postaju izuzetno raspršene s povećanjem dimenzionalnosti prostora [109]. Posljedica toga može biti dugo

vrijeme potrebno za dosezanje obećavajućih dijelova odnosno regija prostora pretrage što u konačnici dovodi do potrebe za većom količinom vremena za konvergenciju prema dobrim ili obećavajućim rješenjima.

Kada je vrednovanje funkcije cilja računalno zahtjevno ili kada je broj njenih vrednovanja ograničen, poželjene su veće brzine konvergencije prema obećavajućim dijelovima prostora pretrage, jer one smanjuju vrijeme potrebno za pronalazak zadovoljavajućih ili dobrih rješenja. Povećanje brzine konvergencije moguće je postići uvođenjem dobrih rješenja u početnu populaciju [81]. Međutim, to često nije jednostavno ostvariti.

4.2 Pregled literature

Jedan od razloga zbog kojeg se najčešće početna populacija stvara nasumično je i jednostavnost ugradnje takvog pristupa. Dodatno, s obzirom da joj se uobičajeno ne pridaje posebna pažnja [3], može se izvesti zaključak kako su takve početne populacije dostatne za postizanje zadovoljavajuće učinkovitosti algoritama. Ipak, u literaturi se nude različite unaprijeđene metode ili pristupi inicijalizaciji populacije koje mogu pružiti povećanje učinkovitosti, a u prvom redu brzine konvergencije. Treba istaknuti da su metode inicijalizacije opisane u nastavku korištene u različitim algoritmima, ali one nisu vezane za njih i primjenjive su u drugim algoritmima.

Kada je znanje o problemu koji se rješava dostupno, često je moguće iskoristiti jednostavne heuristike za stvaranje dobrih rješenja koja se potom mogu uvesti u početnu populaciju. Zbog iskorištavanja spomenutog znanja o problemu kojeg se rješava, takvim metodama inicijalizacije populacije je u pravilu ograničena primjena na isti ili eventualno srodan i vrlo sličan problem. Problem trgovačkog putnika (engl. *travelling salesman problem*, TSP) predstavlja izvanredan primjer za koji postoji mnoštvo heuristika. Tako su primjerice, Wang et al. [140] predložili izmijenjeni pohlepni algoritam, dok su Martinović i Bajer [81] predložili izmijenjeni algoritam najbližeg susjeda za potrebe stvaranja početne populacije. Opsežan pregled i usporedbu različitih pristupa inicijalizaciji populacije za slučaj TSP može se pronaći, primjerice u [99]. Razne metode inicijalizacije populacije predložene su i za mnoge druge probleme osim TSP. Primjerice, Guerrero et al. [49] predstavili su dvije različite metode za potrebe rješavanja problema segmentacije krivulje. Bajer et al. [8] predložili su uporabu algoritma k-means za inicijalizaciju dijela populacije DE za izgradnju radijalnih mreža. Za potrebe rješavanja jedne vrste problema raspoređivanja poslova, Zhang et al. [161] predložili su novu metodu inicijalizacije populacije, a Modiri-Delshad i Rahim [86] predložili su metodu za potrebe rješavanja problema ekonomične raspodjele.

Mnogi (stvarni) problemi nalik su crnim kutijama pa ne postoji znanje (informacije) o njihovoj unutrašnjosti ili je njihov model vrlo složen. Stvaranje početne populacije koja sadrži dobra rješenja, u takvim slučajevima, predstavlja značajan problem. Stoga, nije iz-

nenađujuće što se najčešće populacija inicijalizira potpuno nasumično stvorenim rješenjima, jer je vrlo teško doći do dobrih rješenja na jednostavan i učinkovit način. Ipak, postoje pokušaji svladavanja navedenog problema. Metodu inicijalizacije populacije koja se koristi učenjem zasnovanim na opoziciji (engl. *opposition-based learning*, OBL) [113, 145] predložili su Rahnamayan et al. [110, 112]. Metoda se u osnovi koristi definicijom suprotnih točaka (engl. *opposite points*), gdje se za svaku danu (nasumično stvorenu) točku odnosno rješenje $\mathbf{v} \in [\mathbf{s}^L, \mathbf{s}^U] \subset \mathbb{R}^d$, računa suprotno $\mathbf{o} = \mathbf{s}^L + \mathbf{s}^U - \mathbf{v}$. Navedena metoda inicijalizacije populacije uspješno je upotrijebljena u nekoliko različitih algoritama (vidi primjerice [31, 45, 55]). Unaprijeđena varijanta koja se koristi definicijom kvazi-suprotnih točka (engl. *quasi-opposite points*) predložena je u [111], a predstavlja proširenje prethodno navedene metode. Kvazi-suprotna točka stvara se nasumično između suprotne i točke koja je središte prostora pretrage. Nadalje, potpuno drukčiji pristup inicijalizaciji populacije EAs dali su Ali et al. [3]. Predložene su dvije inačice, gdje se jedna koristi blago izmijenjenom metodom kvadratne interpolacije, dok se druga koristi, također blago izmijenjenim, Nelder-Mead algoritmom. Prethodno navedene tri metode stvaraju novu populaciju rješenja na temelju zadane nasumično stvorene, a za konačnu početnu odabiru najbolja rješenja iz unije te dvije populacije.

Treba spomenuti i kvazi-nasumične nizove (engl. *quasi-random sequences*), kao što su Sobol ili Halton nizovi, koji omogućuju ravnomjernije popunjavanje prostora u odnosu na nasumično stvorene točke. Ovi nizovi u pravilu gube svoja svojstva u prostorima većih dimenzija što predstavlja glavnu prepreku njihove uporabe. Ipak, mogu se pronaći pokušaji primjene navedenih nizova u inicijalizaciji populacija (vidi primjerice [98, 133]). Međutim, u [95] je pokazano da početne populacije stvorene pomoću kvazi-nasumičnih nizova ne donose prednosti u odnosu na nasumično stvorene populacije. Metodu inicijalizacije populacije koja pokušava postići značajno ravnomjernije popunjavanje prostora u usporedbi sa standardnom nasumičnom, bez obzira na njegovu dimenzionalnost, predstavili su Richards i Ventura [115]. Metoda je zasnovana na stvaranju približne Voronoi teselacije pomoću algoritma Ju-Du-Gunzburger, ali metoda zahtijeva nasumično generiranje velikog broja točaka u svakoj iteraciji tog algoritma.

4.3 Metoda inicijalizacije populacije DE zasnovana na grupiranju podataka i slučajnim varijablama

Predložena metoda za inicijalizaciju populacije [9] ne pretpostavlja nikakvo znanje o problemu koji se rješava, a zasnovana je na grupiranju podataka (engl. *data clustering*) [131, 146] i Cauchyjevim slučajnim varijablama. Grupiranje predstavlja jedan od ključnih koraka u predloženoj metodi pa je u nastavku prvo dan kratak uvod u grupiranje podataka kao pro-

blema globalne optimizacije. Također, dan je sažeti pregled primjene grupiranja podataka u evolucijskim algoritmima.

4.3.1 Kratki uvod u grupiranje podataka i algoritam k-means

Za dani skup $\mathcal{A} = \{\mathbf{a}^j : j = 1, \ldots, n\} \subset [\alpha, \beta] \subset \mathbb{R}^d$, gdje je $\alpha = (\alpha_1, \ldots, \alpha_d) \in \mathbb{R}^d$, $\beta = (\beta_1, \ldots, \beta_d) \in \mathbb{R}^d$, grupiranje podataka zahtijeva raspodjelu tog skupa u $1 < k < n$ podskupova (grupa) $\pi_1, \ldots, \pi_k$ takvih da

$$\bigcup_{j=1}^{k} \pi_j = \mathcal{A}, \quad \pi_r \bigcap \pi_s = \emptyset, \ r \neq s, \quad |\pi_j| \geq 1, \ j = 1, \ldots, k \ . \tag{4.1}$$

Raspodjela skupa $\mathcal{A}$ u k podskupova koji zadovoljavaju (4.1) čini čvrstu particiju ili k-particiju od $\mathcal{A}$, te se može predstaviti skupom $\Pi(\mathcal{A}) = \{\pi_1, \ldots, \pi_k\}$.

Uvođenjem mjere udaljenosti (engl. *proximity measure*) $d \colon \mathbb{R}^d \times \mathbb{R}^d \to [0, +\infty\rangle$ [130, 131] između elemenata skupa podataka, moguće je svakoj grupi π_j dodijeliti predstavnika u obliku centra $\mathbf{z}^j$. Uobičajeno se koristi Euklidska udaljenost $d(\mathbf{a}^r, \mathbf{a}^s) = \sqrt{\sum_i (\mathbf{a}_i^r - \mathbf{a}_i^s)^2}$. Prema tome, svaku grupu je moguće predstaviti centrom

$$\mathbf{z}^j = \frac{1}{|\pi_j|} \sum_{\mathbf{a} \in \pi_j} \mathbf{a} \ . \tag{4.2}$$

Obratno, na temelju zadanih centara $\mathcal{Z} = \{\mathbf{z}^1, \ldots, \mathbf{z}^k\}$ može se odrediti particija prema principu najmanje udaljenosti (engl. *minimum distance principle*)

$$\pi_j = \{\mathbf{a} \in \mathcal{A} \colon d(\mathbf{a}, \mathbf{z}^j) \leq d(\mathbf{a}, \mathbf{z}^r), \ \forall r = 1, \ldots, k\}, \quad j = 1, \ldots, k \ . \tag{4.3}$$

Broj mogućih particija skupa $\mathcal{A}$ je vrlo velik. Zbog toga je nužno uvesti kriteriji za vrednovanje pojedinih particija. Traženje optimalne particije moguće je definirati kao problem globalne optimizacije [130, 146]

$$\min F(\mathcal{Z}, \Pi) = \sum_{j=1}^{k} \sum_{\mathbf{a} \in \pi_j} d^2(\mathbf{a} - \mathbf{z}^j) \ . \tag{4.4}$$

Nadalje, svakom podatku $\mathbf{a}^j \in \mathcal{A}$ može se pridružiti različita težina $w_j > 0$, kako bi se naglasila njihova važnost. Pri izračunu centara težine određuju njihove doprinose. Sukladno tome, (4.2) postaje

$$\mathbf{z}^j = \frac{1}{W_j} \sum_{\mathbf{a}^r \in \pi_j} \mathbf{a}^r, \quad W_j = \sum_{\mathbf{a}^r \in \pi_j} w_r \ . \tag{4.5}$$

Jedan od najpoznatijih i često korištenih algoritama koji traži lokalno optimalnu particiju od $\mathcal{A}$, u smislu (4.4), je algoritam k-means [131, 142, 146]. On predstavlja jednostavan

algoritam alternirajuće optimizacije koji određuje particiju prema zadanim centrima, a zatim ih ispravlja na temelju te particije. Postupak se ponavlja do dostignuća danog kriterija ili uvjeta zaustavljanja. Iako ima određene nedostatke, zbog svoje jednostavnosti i brzine (prema Duda et al. [32], u pravilu je potreban relativno mali broj iteracija) predstavlja učinkovit i koristan alat za rudarenje podataka (vidi primjerice [5, 96]). Algoritam k-means za grupiranje podataka kojima su pridružene težine prikazan je algoritmom 4.1.

Algoritam 4.1: k-means

postavi k i odaberi početne centre;
ponavljaj
  pridruži svaki $\mathbf{a} \in \mathcal{A}$ u odgovarajuću grupu; % prema (4.3)
  za svaku grupu $\pi \in \Pi$ ispravi dodijeljeni centar; % prema (4.5)
dok *uvjet zaustavljanja nije dosegnut*;

4.3.2 Primjena grupiranja podataka u evolucijskim algoritmima

Grupiranje podataka je pronašlo široku primjenu koja seže od astronomije do obrade signala. Raspodjela u grupe međusobno sličnih podataka može otkriti nove i korisne informacije o danom skupu podataka. Stoga ne iznenađuje što je grupiranje podataka, u raznim oblicima, pronašlo i mjesto u području EAs. U tom smislu, ono je upotrijebljeno na različite načine u svrhu pospješivanja pretrage EAs.

Algoritam grupiranja podataka korišten je u hibridnom EA predloženom od Martinez-Estudillo et al. [79] za izgradnju jedne vrste umjetne neuronske mreže. Točnije, korišten je algoritam k-means za grupiranje određenog broja najboljih rješenja pronađenih od strane korištenog EA kako bi se odredilo koja rješenja će proći lokalnu pretragu (engl. *local search*). Primjenom lokalne pretrage na međusobno različita rješenja, odnosno na najbolja iz pojedinih grupa može se smanjiti računalna zahtjevnost. Zhang et al. [163] su iskoristili grupiranje podataka u drugu svrhu. Naime, kako bi procijenili stanje optimizacije, oni su koristili algoritam k-means za grupiranje populacije u zadani broj grupa. Procjena je potom obavljana na temelju relativnih veličina grupe koja sadrži najbolje i one koja sadrži najlošije rješenje. Ti podaci su zatim predani neizrazitom sustavu za podešavanje vrijednosti parametara, odnosno vjerojatnosti križanja i mutacije. S ciljem povećanja brzine konvergencije i postizanja ravnoteže između istraživanja i iskorištavanja, Gong et al. [48] predložili su uporabu algoritma fuzzy c-means (FCM) [118, 131] unutar DE. U tom pristupu se periodno provodi jedan korak FCM algoritma za raspodjelu populacije u nasumično odabrani broj grupa. Dobiveni centri se potom natječu s odabranim članovima populacije za prelazak u narednu generaciju. Isti autori su koristili algoritam k-means umjesto FCM u [21]. U oba slučaja se grupiranje rabi u svrhu povećanja iskorištavanja prikupljanjem informacija sadržanih u međusobno sličnim vektorima odnosno rješenjima.

4.3.3 Opis predložene metode

Predložena metoda inicijalizacije populacije zasnovana je na ideji stvaranja (relativno) dobrih rješenja u različitim obećavajućim regijama prostora pretrage. U tu svrhu koristi se grupiranjem podataka i Cauchyjevim slučajnim varijablama. Za otkrivanje, odnosno za približavanje obećavajućim regijama služi grupiranje podataka, dok Cauchyjeve slučajne varijable, u obliku jednostavne mutacije, služi za stvaranje novih rješenja čime se vrši daljnje istraživanje tih regija. Predložena metoda inicijalizacije populacije DE prikazana je algoritmom 4.2.

Algoritam 4.2: Predložena metoda za inicijalizaciju populacije DE

```
stvori nasumičnu populaciju P_R, postavi k = √NP = √|P_R|, P_C = ∅;
raspodijeli P_R u k grupa kako bi se dobili centri Z_R, Z_T = Z_R;
za j := 1 → NP−k čini
    odaberi centar z^p ∈ Z_T;                          % jednostavna selekcija
    stvori novo rješenje v̂^j, P_C ← v̂^j;               % prema (4.7)
    ako f(v̂^j) < f(z^p) onda
        z^p := v̂^j;
    kraj
kraj
odaberi NP/2 najboljih rješenja iz {Z_R ∪ P_C} i NP/2 najboljih iz P_R za početnu populaciju;
```

Početnu populaciju može se predstaviti skupom $\mathcal{P}_R = \{\mathbf{v}^j : j = 1, \ldots, NP\} \subset [\alpha, \beta] \subset \mathbb{R}^d$. Nju se najčešće, kao što je ranije spomenuto, inicijalizira potpuno nasumično, odnosno $\mathbf{v}_i^j = \mathcal{U}_i[\alpha_i, \beta_i]$, $i = 1, \ldots, d$, gdje je $\mathcal{U}_i[\alpha_i, \beta_i]$ uniformna slučajna varijabla iz $[\alpha_i, \beta_i]$. Također, s obzirom da je populacija ograničene veličine, moguće je uzorkovati samo mali dio prostora pretrage. Stoga je za očekivati kako će početna populacija sadržavati isključivo rješenja male kvalitete. Predložena metoda započinje s takvom populacijom i koristi dostupne informacije o krajoliku dobrote u pokušaju otkrivanja obećavajućih dijelova prostora pretrage. Navedeno čini grupiranjem cijele populacije $\mathcal{P}_R$ u $k = \sqrt{NP}$ grupa pri čemu koristi i kvalitetu pojedinih rješenja koja je čine. Na taj način, bolja rješenja imaju veći utjecaj na formiranje grupa, što znači da će im centri biti bliže. Prema Cai et al. [21], Gong et al. [48], grupiranje populacije se može smatrati oblikom višeroditeljskog križanja. U predloženoj metodi, kao što je već navedeno, provodi se grupiranje, ali s ponderiranim podacima. Svakom rješenju se prvotno pridružuje težina vezana za njegovu kvalitetu u smislu funkcije cilja. Težine su normalizirane kako bi im se ograničio raspon. Shodno navedenom, svakom rješenju $\mathbf{v}^j \in \mathcal{P}_R$ se pridružuje težina

$$w_j = a + \frac{(f_{max} - f_j) \cdot (b - a)}{f_{max} - f_{min}}, \tag{4.6}$$

gdje je f_j kvaliteta danog rješenja, f_{max} i f_{min} su kvaliteta najlošijeg i najboljeg rješenja, redom, dok je $a = 0.1$ i $b = 0.9$.

Nakon grupiranja populacije $\mathcal{P}_R$, dobiva se skup centara $\mathcal{Z}_R = \{\mathbf{z}^1, \ldots, \mathbf{z}^k\}$ koji predstavljaju pojedine grupe. Njih se može smatrati potomcima, gdje su roditelji centra $\mathbf{z}^j$ svi koji

pripadaju toj grupi. U nadi da su centri blizu obećavajućih dijelova prostora pretrage, obavlja se daljnje istraživanje u njihovoj okolini. Uporabom jednostavne Cauchyjeve mutacije, u blizini odabranog centra $\mathbf{z}_p$ generira se novo rješenje

$$\hat{\mathbf{v}}_i^r = \mathbf{z}_i^p + \mathcal{C}_{r,i}(0,1), \quad i = 1, \ldots, d, \tag{4.7}$$

gdje je $\mathcal{C}_{r,i}(0,1)$ Cauchyjeva slučajna varijabla s parametrom lokacije $l = 0$ i parametrom skaliranja $s = 1$ (parametri odgovaraju standardnoj Cauchyjevoj razdiobi). Funkcija gustoće vjerojatnosti Cauchyjeve razdiobe [29]

$$f_{pd}(x; l, s) = \frac{1}{s\pi(1 + (\frac{x-l}{s})^2)}, \tag{4.8}$$

nalikuje Gaussovoj (normalnoj) razdiobi, ali prilazi osi toliko sporo da matematičko očekivanje ne postoji, a varijanca nije definirana. Stoga je stvaranje rješenja na većoj udaljenosti od centara vjerojatnije zbog čega je moguće obuhvatiti veći dio prostora pretrage oko tih centara. Navedeno čini Cauchyjevu razdiobu prikladnijom za istraživanje većih susjedstva u odnosu na Gaussovu razdiobu.

S obzirom na teškoću određivanja pravog broja grupa u nekom skupu podataka (vidi primjerice [135]), odabrana je vrijednost $k = \sqrt{NP} = \sqrt{|\mathcal{P}_R|}$ koju se u pravilu smatra gornjom granicom [89, 119]. Općenito, neće svi centri biti jednako udaljeni od obećavajućih regija, te je za očekivati kako su im bolji centri, u smislu funkcije cilja, bliže. Prema tome, opsežnije istraživanje oko takvih centara vjerojatnije će rezultirati daljnjim unaprjeđenjima. U predloženoj metodi, bolje centre se odabire s većom vjerojatnosti. Točnije, bira ih se pomoću jednostavne selekcije (engl. *roulette wheel selection*). Time se utjecaj postavljenog broja grupa djelotvorno smanjuje.

Centri dobiveni grupiranjem, ovisno o populaciji $\mathcal{P}_R$ i konačnoj particiji, mogu biti bliže ili dalje od obećavajućih dijelova prostora pretrage. Stoga, kako bi im se što više približilo, novo rješenje zamjenjuje centar na temelju kojeg je dobiveno ako je bolje, a daljnje istraživanje se onda obavlja oko njega. Bitno je istaknuti da se pri tom zadržavaju originalni centri $\mathcal{Z}_R$.

Prema svemu navedenom, generira se nova populacija $\mathcal{P}_C$ veličine $NP–k$. Međutim, treba napomenuti da se takva populacija sastoji od rješenja koja su relativno blizu originalnim centrima. Štoviše, zbog bliskosti grupiranih rješenja, daljnje istraživanje prostora pretrage je usmjereno prema regijama u kojima se nalaze ti centri, posebno prema najboljima, jer je općenito više rješenja generirano oko njih. To može biti poželjno kada se jedan ili više centara nalazi blizu globalnog optimuma, ali zbog nedostatka raznolikosti to može dovesti do preuranjene konvergencije. Stoga, kako bi se osigurala dovoljna raznolikost, konačna početna populacija sastoji se od dva dijela. Prvi dio predstavlja $NP/2$ najboljih rješenja iz skupa $\{\mathcal{Z}_R \cup \mathcal{P}_C\}$, dok drugi dio predstavlja $NP/2$ najboljih rješenja iz $\mathcal{P}_R$.

4.3.4 Detalji ugradnje

Predložena metoda izvodi dva ključna koraka. Prvi korak je grupiranje početno zadane populacije $\mathcal{P}_R$, a drugi korak je generiranje novih rješenja pomoću jednostavne Cauchyjeve mutacije. Grupiranje se provodi jednom pri čemu se koristi algoritam k-means (Alg. 4.1) te se na taj način stvara k rješenja. Preostalih $NP-k$ rješenja stvara se pomoću Cauchyjeve mutacije.

Centre za algoritam k-means inicijalizira se pomoću k različitih i nasumično odabranih rješenja iz $\mathcal{P}_R$. Time se izbjegava mogućnost pojave praznih grupa. Nadalje, korištena su dva uvjeta zaustavljanja. Do prekida dolazi ukoliko je izvršeno 50 iteracija ili ranije ako udaljenost između bilo kojeg para centara iz dvije uzastopne iteracije nije veća od $\epsilon = 0.01$ (pretpostavka konvergencije). Nakon grupiranja, dobiveni centri koriste se za stvaranje novih rješenja. Za svako novo rješenje odabire se centar pomoću jednostavne selekcije. Treba napomenuti kako se originalni centri zadržavaju pa se koriste njihove kopije. Također, kvaliteta centara normalizira se analogno pridruživanju težina, odnosno prema (4.6). To se obavlja prije odabira. S obzirom da novo rješenje zamjenjuje odgovarajući centar ako je bolje, potrebno je obaviti normalizaciju nakon svake zamjene.

4.3.5 Analiza vremenske složenosti

Predložena metoda generira ukupno $2 \cdot NP$ rješenja, gdje se NP stvara potpuno nasumično, njih $k = \sqrt{NP}$ dobiva se grupiranjem, dok se preostalih $NP - k$ generira Cauchyjevom mutacijom. Prema tome, $2 \cdot NP - k$ rješenja stvara se pomoću slučajnih varijabli. Složenost stvaranja NP rješenja pomoću uniformnih slučajnih varijabli može se procijeniti s $O(NP \cdot d)$. Stvaranje $NP-k$ rješenja pomoću Cauchyjeve mutacije zahtijeva prethodni odabir centra za svako rješenje pa se složenost može procijeni s $O((NP-k) \cdot (k+d))$. Zbog relativno male vrijednosti k i male složenosti jednostavne selekcije, složenost stvaranja navedenih $2 \cdot NP-k$ rješenja može se grubo procijeniti s $O((2 \cdot NP-k) \cdot d)$. Nadalje, k rješenja predstavljaju centre grupa dobivenih algoritmom k-means čija se složenost, prema Naldi et al. [89], Theodoridis i Koutroumbas [131], može procijeniti s $O(t_{max} \cdot NP \cdot d \cdot k)$, gdje je t_{max} najveći dozvoljeni broj iteracija (50 u konkretnom slučaju). Ovo se odnosi na granični slučaj, jer se koristi i dodatni, prethodno navedeni, uvjet zaustavljanja. Osim toga, treba napomenuti da se prije grupiranja svakom rješenju pridružuje težina pa se složenost cijelog postupka grupiranja može procijeniti s $O(NP \cdot (t_{max} \cdot k \cdot d + 1))$.

4.4 Eksperimentalni rezultati i analiza: Predložena metoda inicijalizacije populacije DE

S ciljem utvrđivanja prednosti i nedostataka predložene metode inicijalizacije populacije, provedena je eksperimentalna analiza, a podijeljena je u četiri dijela. Prvi dio pruža uvid u ponašanje predložene metode inicijalizacije. Drugi dio odnosi se na analizu učinkovitosti u smislu brzine konvergencije te vremena izvođenja na problemima niske i srednje dimenzionlnosti. Treći dio predstavlja analizu utjecaja veličine populacije. Četvrti dio je analiza učinkovitosti u prostorima viših dimenzionalnosti. Cjelokupno testiranje i analiza provedeni su na skupu standardnih testnih funkcija uz uporabu različitih dimenzionalnosti za svaku (osim ako su zadane dimenzionalnosti). Točnije, provedeni su na ukupno 118 instanci problema. Navedeni skup funkcija prikazan je i opisan u dodatku A. Treba istaknuti da su problemi za $d < 10$ smatrani problemima niske dimenzionalnosti, problemi za $10 \leq d \leq 50$ srednje, a oni za $d > 50$ viših dimenzionalnosti.

Predložena metoda ugrađena je u standardni DE (označeno s $\mathrm{DE_C}$) kako bi se ustanovio njen utjecaj na učinkovitost algoritma. Za potrebe usporedbe, u standardni algoritam DE su ugrađeni standardni, odnosno nasumični pristup inicijalizaciji (naznačeno s $\mathrm{DE_R}$), te metode koje se koriste suprotnim [110] (označeno s $\mathrm{DE_O}$) odnosno kvazi-suprotnim točkama [111] (označeno s $\mathrm{DE_{QO}}$). Treba napomenuti kako metode korištene u $\mathrm{DE_C}$, $\mathrm{DE_O}$ i $\mathrm{DE_{QO}}$ zahtijevaju $2 \cdot NP$ vrednovanja funkcije cilja, dok ona korištena u $\mathrm{DE_R}$ zahtijeva samo NP vrednovanja.

4.4.1 Postavke eksperimenata

Za svaki algoritam i instancu problema izvršeno je 51 nezavisno izvođenje algoritma. Parametri algoritama bili su jednaki za sve i odgovaraju onima korištenima u [110]. Točnije, korišteni su $NP = 100$, $CR = 0.9$ i $F = 0.5$. Također, bilo je dozvoljeno najviše 10^6 vrednovanja funkcije cilja. Do prekida izvođenja dolazilo je ukoliko je dosegnuta ciljana greška optimizacije $\Delta f = |f^* - f^{best}| < \tau$ ili ukoliko je izvršen dozvoljeni broj vrednovanja. Vrijednost τ za pojedine funkcije postavljena je relativno visoko, jer je cilj bio ispitati učinkovitost upotrijebljenih metoda inicijalizacije populacije. Korištene vrijednosti za τ prikazane su tablicom 4.1. Vrijednosti greške optimizacije manje od τ su uzete kao nula. Početne nasumične populacije koje se koriste u svim navedenim metodama generirane su unutar cijelog prostora pretrage, a granicama istog rukovano je prema (2.11). Za stvaranje pseudo-slučajnih brojeva korišten je, kao i prethodno, Mersenne Twister PRNG.

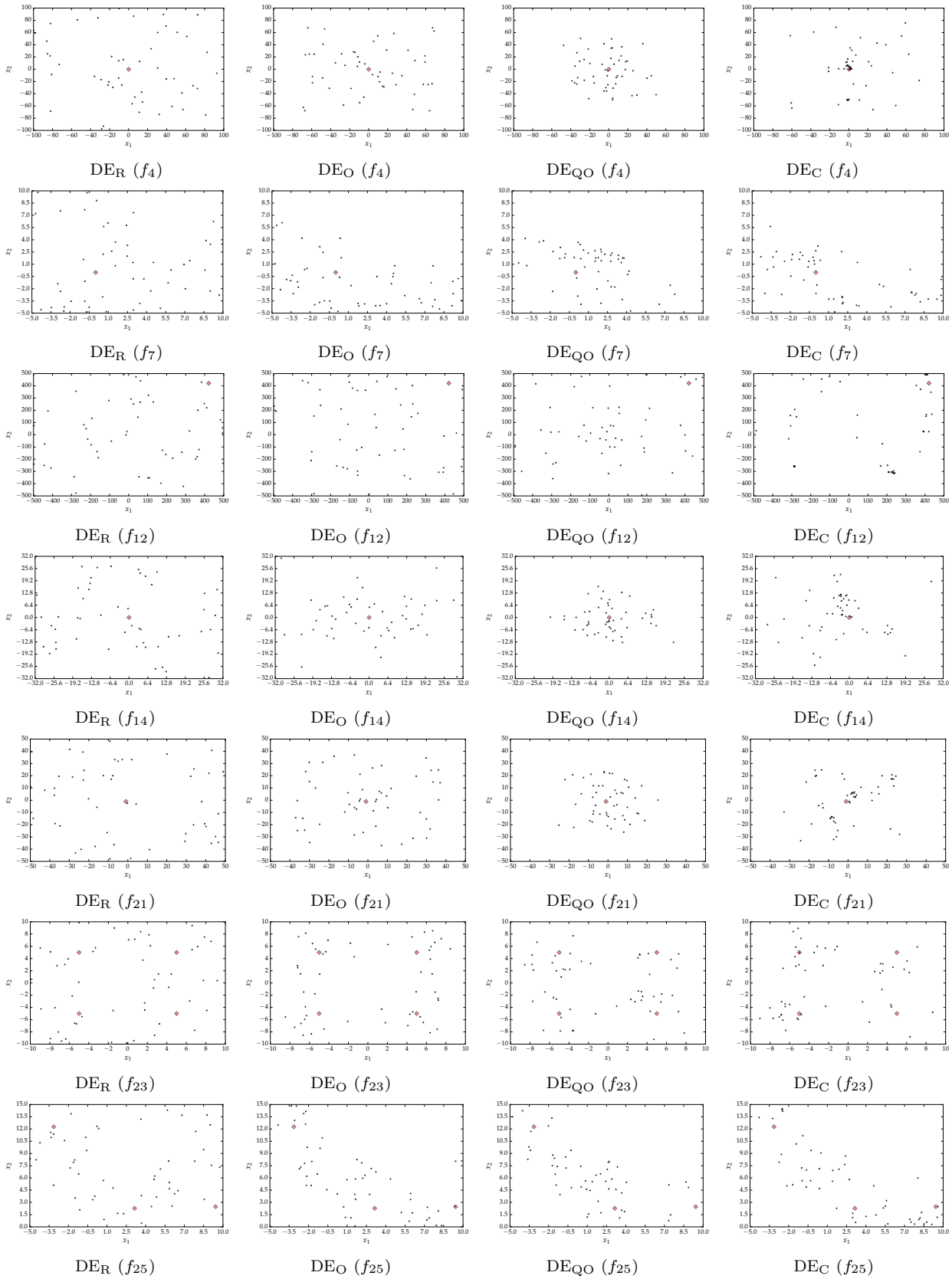
Tablica 4.1: Vrijednosti τ korištene za pojedine funkcije i dimenzionalnosti problema.

f	τ
$\{1,\ldots,22\}\backslash\{4,5,12,13,15,17,18,20\}$	10^{-1}
4	10
5	50
12	10^3
13	10^2 za $d=10$, $2{\cdot}10^2$ za $d=30$, $3{\cdot}10^2$ za $d\geq 50$
15	1
17	3 za $d=10$, 10 za $d=30$, 20 za $d\geq 50$
18	5 za $d=10$, 20 za $d=30$, 40 za $d\geq 50$
20	1
$\{23,\ldots,30\}\backslash\{27\}$	10^{-3}
27	1

4.4.2 Ponašanje upotrijebljenih metoda inicijalizacije populacije

Metode inicijalizacije upotrijebljene u $\mathrm{DE_R}$, $\mathrm{DE_O}$, $\mathrm{DE_{QO}}$ i $\mathrm{DE_C}$ stvaraju početne populacije na različite načine, ali svima je zajednička nasumično stvorena populacija. Kako bi se pružio uvid u ponašanje navedenih metoda, barem na konceptualnoj razini, one su korištene za generiranje početnih populacija pri nekoliko odabranih funkcija za $d = 2$. Za svaku funkciju, generirano je 50 rješenja danom metodom unutar cijelog prostora pretrage. Dobivene populacije prikazane su slikom 4.1, gdje je lokacija globalnog optimuma (minimuma) označena rombom ($\Diamond$).

Kao što se može vidjeti sa slike 4.1, populacije generirane pojedinim metodama se razlikuju u strukturi. Primjerice, populacije generirane uobičajenim ili standardnim pristupom ne iskazuju smislenu strukturu, što je bilo i za očekivati. Vidljivo je da populacije generirane metodom koja se koristi OBL (korištena u $\mathrm{DE_O}$) ispoljavaju simetriju koja i odgovara definiciji suprotnih točaka. Zanimljivo je primijetiti kakvu razliku čini uporaba kvazi-suprotnih točaka u odnosu na suprotne točke. Tako se u slučaju populacija generiranih metodom inicijalizacije upotrijebljenom u $\mathrm{DE_{QO}}$ ne primjećuje simetrija, ali se može primijetiti sklonost generiranja rješenja u široj okolini centra prostora (čak i kada je globalni optimum daleko od njega ili kada prostor nije simetričan u odnosu na položaj optimuma). Štoviše, te populacije su često mnogo više zbijene pa je, jasno, njihova raznolikost mnogo manja. U tom pogledu, slično se može opaziti i u slučaju predložene metode inicijalizacije populacije (korištena u $\mathrm{DE_C}$), gdje se često mogu vidjeti male grupe rješenja koje okružuju optimum ili su u njegovoj blizini. Također, mogu se vidjeti i rješenja koja su daleko od optimuma, a ona uglavnom potječu od nasumično stvorenog dijela populacije. S obzirom na brojna rješenja u blizini optimuma, njegovo dostizanje kroz manji broj vrednovanja funkcije cilja je vjerojatnije u usporedbi, jer ta rješenja efektivno usmjeravaju pretragu prema tim regijama odmah od početka.



Slika 4.1: Vizualizacija populacija generiranih korištenim metodama za inicijalizaciju populacije.

4.4.3 Učinkovitost na problemima niske i srednje dimenzionalnosti

Tablice 4.2, 4.3, 4.4 i 4.5 prikazuju ostvarene rezultate u smislu brzine konvergencije i vremena izvođenja. Brzina konvergencije izražena je kao ukupan (zbroj svakog izvođenja algoritma na danoj funkciji) broj vrednovanja funkcije cilja potrebnih za dostizanje ciljane greške optimizacije ($\mathrm{TER}_{\mathrm{NFEs}}$). Kada stopa uspješnosti (SR) nije bila 100 % za sve algoritme na danoj instanci problema, u zagradama je prikazana ostvarena SR. Vremena izvođenja dana su također kao ukupna vremena u sekundama. Osim toga, navedene tablice prikazuju ubrzanja,

$$\text{ubrzanje} = \left(1 - \frac{\sum_{i=1}^{n} \mathrm{TER}_{\mathrm{NFEs}}(f_i) \text{ za } \mathrm{DE_C}}{\sum_{i=1}^{n} \mathrm{TER}_{\mathrm{NFEs}}(f_i) \text{ za } \mathrm{DE_{R/O/QO}}}\right) \cdot 100\,\%, \tag{4.9}$$

postignuta algoritmom koji uključuje predloženu metodu inicijalizacije u usporedbi s drugim upotrijebljenim algoritmima.

Kao što se može vidjeti iz prikazanih rezultata, predloženom metodom postignuta su značajna ubrzanja u usporedbi s onima korištenima u $\mathrm{DE_R}$, $\mathrm{DE_O}$, dok su razlike u odnosu na metodu korištenu u $\mathrm{DE_{QO}}$ osjetno manje. Štoviše, $\mathrm{DE_{QO}}$ je ostvario u usporedbi najveće

Tablica 4.2: Brzina konvergencija u smislu $\mathrm{TER}_{\mathrm{NFEs}}$ i ukupna vremena izvođenja (u sekundama) na funkcijama $f_1 \sim f_{22}$ za $d = 10$.

f	$\mathrm{DE_R}$		$\mathrm{DE_O}$		$\mathrm{DE_{QO}}$		$\mathrm{DE_C}$	
	$\mathrm{TER}_{\mathrm{NFEs}}$	Uk. t	$\mathrm{TER}_{\mathrm{NFEs}}$	Uk. t	$\mathrm{TER}_{\mathrm{NFEs}}$	Uk. t	$\mathrm{TER}_{\mathrm{NFEs}}$	Uk. t
1	617728	0.23	626512	0.20	570513	0.19	580195	0.19
2	642270	0.17	640918	0.20	565708	0.19	613419	0.22
3	979986	0.30	998723	0.31	929016	0.32	923442	0.30
4	240740	0.06	245788	0.09	167093	0.14	167201	0.08
5	490019	0.13	508366	0.16	453949	0.15	456150	0.16
6	10319	0.01	13214	0.03	4862	0	6459	0.06
7	611106	0.16	619811	0.21	600940	0.20	598037	0.20
8	524559	0.31	519918	0.35	476663	0.33	477474	0.29
9	82899	0.02	84101	0.04	23113	0.03	82321	0.06
10	560490	0.14	564436	0.17	504812	0.17	511923	0.16
11	71116	0.03	74624	0.06	16687	0.02	71128	0.08
Uk.	4831232	**1.57**	4896411	1.82	**4313356**	1.75	4487749	1.80
—	7.11 %		8.35 %		−4.04 %		ubrzanje	
12	970415	0.87	1141733	1.07	1005599	1.03	1083334	1.04
13	9828	0.01	12097	0.02	4560	<0.01	4937	0.04
14	696609	0.51	703647	0.52	645080	0.50	663382	0.59
15	380276	0.34	385059	0.42	340094	0.35	339850	0.33
16	711600	0.52	723023	0.62	573002	0.50	656177	0.48
17	907643	0.83	865443	0.90	838664	0.76	768161	0.72
18	213183	0.24	232729	0.29	121390	0.15	194630	0.23
19	278520	0.23	285890	0.27	217639	0.21	246714	0.20
20	387834	0.12	399706	0.17	336955	0.16	363253	0.14
21	516259	0.55	514363	0.61	449571	0.53	460071	0.45
22	517017	0.63	520929	0.60	461519	0.54	474177	0.62
Uk.	5589184	4.85	5784619	5.49	**4994073**	**4.72**	5254686	4.86
—	5.98 %		9.16 %		−5.22 %		ubrzanje	

Tablica 4.3: Brzina konvergencija u smislu TER_{NFEs} i ukupna vremena izvođenja (u sekundama) na funkcijama $f_1 \sim f_{22}$ za $d = 30$.

f	DE_{R}		DE_{O}		DE_{QO}		DE_{C}	
	TER_{NFEs}	Uk. t	TER_{NFEs}	Uk. t	TER_{NFEs}	Uk. t	TER_{NFEs}	Uk. t
1	2341313	1.51	2372658	1.78	2245142	1.67	2055707	1.28
2	2801960	1.97	2766368	1.9	2574340	1.77	2522163	1.62
3	9129637	10.84	9054396	10.41	8627519	10.96	8112064	9.24
4	1441821	0.98	1453525	0.96	1193713	0.81	812738	0.58
5	2252161	1.49	2246176	1.45	2103547	1.36	1985879	1.31
6	28086	0.08	31224	0.11	5308	0.02	5219	0.08
7	7234623	4.88	7222430	5.16	7281945	4.82	7224522	4.79
8	1980927	2.87	1969818	3.04	1857692	2.79	1672228	2.42
9	612121	0.43	618811	0.44	480919	0.38	526347	0.41
10	2130015	1.34	2133048	1.35	2014821	1.3	1847444	1.19
11	521814	0.36	514718	0.35	404334	0.3	443416	0.33
Uk.	30474478	26.76	30383172	26.93	28789280	26.18	**27207727**	**23.23**
—	10.72 %		10.45 %		5.49 %		ubrzanje	
12	12041057	28.09	13600343	31.76	11769591	27.53	11915720	27.28
13	2787244	5.38	2926064	5.57	2261783	4.33	2433444	4.73
14	2435920	4.58	2446262	4.67	2288153	4.31	2180927	4.04
15	1798557	4.47	1804734	4.82	1667547	4.14	1468525	3.70
16	2431825	4.69	2414761	4.7	2155459	4.49	2086164	4.01
17	– (45%)	117.35	– (47%)	117.83	– (37%)	120.09	– (57%)	113.44
18	6098507	20.08	5670511	18.3	4846279	16.29	4948131	15.54
19	1036539	2.71	1055645	2.35	878322	1.86	834114	1.75
20	2265937	1.48	2262842	1.73	2167595	1.51	1962892	1.34
21	1977881	5.19	1960306	5.47	1795496	4.65	1623503	3.98
22	2257844	6.55	2262682	6.3	2115359	5.73	1943764	5.04
Uk.	35131311	200.56	36404150	203.49	31945584	194.93	**31397184**	**184.85**
—	10.63 %		13.75 %		1.72 %		ubrzanje	

brzine konvergencije na funkcijama manjih dimenzionalnosti ($f_1 \sim f_{22}$ za $d = 10$ i $f_{23} \sim f_{30}$), ali je na funkcijama većih dimenzionalnosti ($f_1 \sim f_{22}$ za $d = 30$ i $d = 50$) potpuno izgubio tu prednost. Treba napomenuti da velika većina upotrijebljenih instanci problema ima globalni optimum u centru prostora ili blisko njemu. Stoga se navedena prednost može povezati s tendencijom metode zasnovane na kvazi-suprotnim točkama (korištena u DE_{QO}) da generira rješenja u okolini centra prostora pretrage pa porastom dimenzionalnosti opada vjerojatnost stvaranja rješenja relativno bliskih centru po svakoj dimenziji. Ipak, promatranjem TER_{NFEs} vrijednosti, može se primijetiti kako je DE_{C} na većini instanci problema trebao osjetno manje vrednovanja funkcije cilja u odnosu na DE_{R}, DE_{O}, dok se DE_{QO} pokazao puno konkurentnijim u usporedbi s potonja dva.

Predložena metoda inicijalizacije populacije je, kao što je jasno iz algoritma 4.2, složenija i računalno zahtjevnija u odnosu na druge korištene u usporedbi. Međutim, s gledišta ukupnih vremena izvođenja, to je nadoknađeno za vrijeme izvođenja algoritma. Kao što se može primijetiti, na većini instanci problema vremena izvođenja bila su manja za DE_{C} u usporedbi s DE_{R}, DE_{O}, ali i DE_{QO} u slučaju funkcija za $d = 30$ i $d = 50$. Izuzetak čine funkcije manjih dimenzionalnosti ($d \leq 10$) na kojima se ubrzanja nisu vidljivo odrazila u vremenima izvođenja. To se može povezati s malom računalom zahtjevnosti vrednovanja tih funkcija. Ipak, iz ostvarenih rezultata, može se zaključiti da inicijalizacija populacija igra beznačajnu

Tablica 4.4: Brzina konvergencija u smislu TER_{NFEs} i ukupna vremena izvođenja (u sekundama) na funkcijama $f_1 \sim f_{22}$ za $d = 50$.

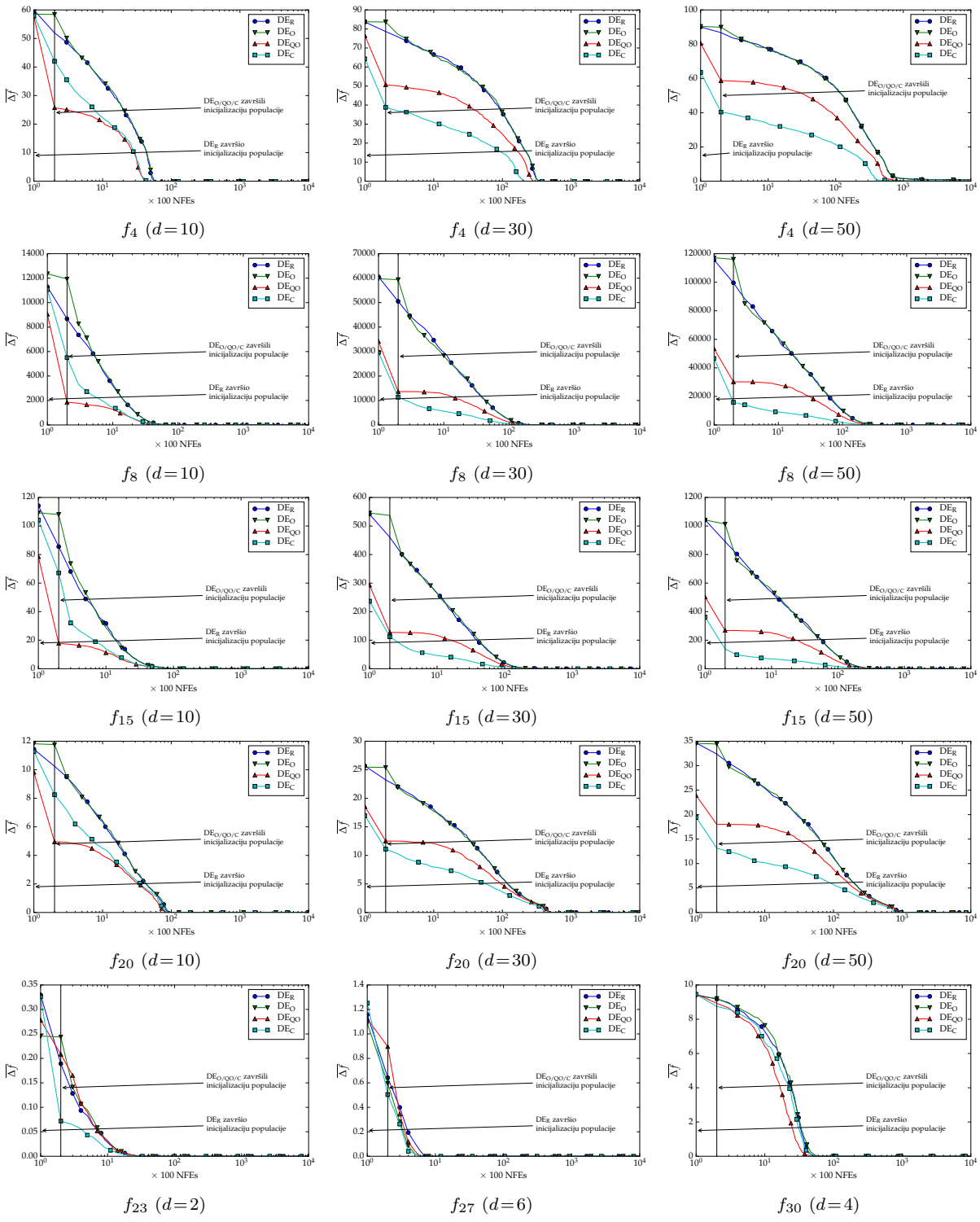
f	DE_{R}		DE_{O}		DE_{QO}		DE_{C}	
	TER_{NFEs}	Uk. t	TER_{NFEs}	Uk. t	TER_{NFEs}	Uk. t	TER_{NFEs}	Uk. t
1	3747678	4.08	3732347	4.03	3568660	3.91	3262655	3.17
2	4447097	4.64	4447392	4.70	4262676	4.53	3945189	3.96
3	31512594	82.77	31857476	84.28	31007437	81.61	28631014	73.73
4	– (94%)	6.39	– (94%)	6.34	– (2611396)	2.73	– (1842185)	1.89
5	4505208	4.63	6434605	6.66	4289155	4.50	5093887	5.15
6	43637	0.20	41053	0.22	5346	0.03	5225	0.13
7	28236112	29.59	28791900	30.28	28634007	36.92	28766255	29.47
8	3085865	7.14	3105232	7.37	2909504	10.9	2591162	5.87
9	1143488	1.29	1149650	1.28	969238	1.47	928915	1.06
10	3410249	3.37	3415374	5.49	3252139	3.27	2938894	2.92
11	944623	1.00	937442	1.67	763950	0.90	770666	0.87
Uk.	81076551	145.10	83912471	152.31	79662112	150.76	**110228451**	**128.20**
—	5.11 %		8.32 %		3.42 % (4.25 %)		ubrzanje	
12	– (94%)	115.26	– (23289728)	97.84	– (94%)	98.62	– (92%)	100.85
13	15834218	50.16	15532367	49.84	14650086	46.25	13845465	43.03
14	3735093	11.14	3682517	11.07	3500913	10.53	3282662	9.52
15	2791856	11.63	2813413	11.83	2638028	11.09	2263139	9.13
16	3479366	11.17	3427597	11.01	3112569	9.98	2887836	8.94
17	– (24%)	211.94	– (35%)	203.27	– (25%)	206.44	– (45%)	188.81
18	811406	4.42	766308	4.16	336750	1.88	542536	2.92
19	1519315	5.17	1512859	5.15	1316884	4.51	1156274	3.87
20	4140687	4.25	4128327	4.31	4061752	4.31	3674154	3.74
21	3162194	13.72	3125047	13.64	2952860	12.66	2461825	9.52
22	3833405	17.77	3845044	17.08	3644477	15.78	3180698	12.75
Uk.	39307540	456.62	38833479	429.19	36214319	422.05	**33294589**	**393.06**
—	15.30 %		14.26 %		8.06 %		ubrzanje	

Tablica 4.5: Brzina konvergencija u smislu TER_{NFEs} i ukupna vremena izvođenja (u sekundama) na funkcijama $f_{23} \sim f_{30}$.

f	DE_{R}		DE_{O}		DE_{QO}		DE_{C}	
	TER_{NFEs}	Uk. t	TER_{NFEs}	Uk. t	TER_{NFEs}	Uk. t	TER_{NFEs}	Uk. t
23	131470	0.01	134248	0.03	138805	0.03	99273	0.03
24	109055	0.03	108984	0.05	95368	0.06	110462	0.05
25	112655	0.02	102435	0.03	115216	0.04	101330	0.03
26	75891	0.02	72438	0.03	75852	0.03	71256	0.05
27	11585	<0.01	10130	0.01	11569	0.02	9158	0.05
28	341875	0.07	344988	0.09	290638	0.08	339023	0.08
29	317425	0.08	328474	0.14	274318	0.08	309869	0.09
30	318152	0.08	326353	0.16	279730	0.09	308461	0.09
Uk.	1418108	**0.31**	1428050	0.53	**1281496**	0.44	1348832	0.47
—	4.89 %		5.55 %		−5.25 %		ubrzanje	

ulogu u cjelokupnom vremenu izvođenja, barem u slučaju upotrijebljenih metoda.

Uvid u ponašanje algoritama s upotrijebljenim metodama inicijalizacije pruža slika 4.2. Prikazana je prosječna greška optimizacije $\overline{\Delta f}$ u odnosu na broj vrednovanja funkcije cilja (NFEs) za nekoliko odabranih funkcija. Greška optimizacije je bilježena svakih 100 vrednovanja. U slučaju funkcija za $d = 30, 50$, značajna prednost DE_{C} u usporedbi s DE_{R}, DE_{O} i DE_{QO} jasno je vidljiva na početku (nakon završetka inicijalizacije). To ukazuju i na sposobnost predložene metode u stvaranju početnih populacija koje sadrže relativno dobra rješenja. Iako prednost opada s NFEs, DE_{C} je uspijeva u pravilu zadržati uzimajući u ob-



f_4 $(d=10)$ f_4 $(d=30)$ f_4 $(d=50)$

f_8 $(d=10)$ f_8 $(d=30)$ f_8 $(d=50)$

f_{15} $(d=10)$ f_{15} $(d=30)$ f_{15} $(d=50)$

f_{20} $(d=10)$ f_{20} $(d=30)$ f_{20} $(d=50)$

f_{23} $(d=2)$ f_{27} $(d=6)$ f_{30} $(d=4)$

Slika 4.2: Grafovi konvergencije za nekoliko odabranih funkcija. Usporedba predložene, standardne i metoda iz literature.

zir prethodno navedena postignuta ubrzanja. Grafovi na slici 4.2 pružaju uvid kako $\mathrm{DE_{QO}}$ gubi na učinkovitosti s povećanjem dimenzionalnosti problema, a i na blagu stagnaciju koja nastupa odmah nakon inicijalizacije.

Ostvareni rezultati, cjelokupno gledano, ukazuju na veću učinkovitost algoritma s ugra-

đenom predloženom metodom inicijalizacije u odnosu na isti s ugrađenim drugim metodama korištenim u usporedbi. Navedena veća učinkovitost se ispoljila u osjetno manjem broju vrednovanja potrebnih za dostizanje ciljane greške optimizacije, koji je shodno tome doveo do manjih vremena izvođenja na većini instanci problema. Ovo upućuje na zaključak da bi predložena metoda bila posebno pogodna za situacije u kojima je NFEs ograničen ili u kojima je vrednovanje funkcije cilja računalno zahtjevno.

4.4.4 Utjecaj veličine populacije

Veličina populacije igra važnu ulogu u učinkovitosti EAs. Veće populacije omogućuju opsežnije istraživanje prostora pretrage, ali zahtijevaju veći broj vrednovanja funkcije cilja, u odnosu na manje, kako bi konvergirale. Osim toga, veće populacije mogu pokriti veći dio prostora pretrage, što može dovesti do bržeg otkrivanja obećavajućih regija. Imajući navedeno na umu, dodatno je ispitan utjecaj veličine populacije na učinkovitost upotrijebljenih metoda inicijalizacije. U tu svrhu, korištene metode inicijalizacije testirane su s različitim budžetima za vrednovanje funkcije cilja. Točnije, testirane su s budžetima od NFEs = 100, 200, 400 i 600 samo za stvaranje početne populacije. U slučaju standardne metode inicijalizacije (ugrađena u $\mathrm{DE_R}$), veličina populacije odgovara dozvoljenom NFEs ($NP \Leftrightarrow$ NFEs). Za ostale tri metode, veličina populacije odgovara polovici dozvoljenih NFEs ($NP \Leftrightarrow$ NFEs/2).

Tablica 4.6: Učinkovitost korištenih metoda inicijalizacije u usporedbi s predloženom na funkcijama $f_1 \sim f_{22}$ za d=10.

f	NFEs = 100			NFEs = 200			NFEs = 400			NFEs = 600			$\mathrm{DE_C}$
	$\mathrm{DE_R}$	$\mathrm{DE_O}$	$\mathrm{DE_{QO}}$	$\mathrm{DE_R}$	$\mathrm{DE_O}$	$\mathrm{DE_{QO}}$	$\mathrm{DE_R}$	$\mathrm{DE_O}$	$\mathrm{DE_{QO}}$	$\mathrm{DE_R}$	$\mathrm{DE_O}$	$\mathrm{DE_{QO}}$	
1	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
2	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
3	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
4	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
5	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
6	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
7	▽∘	▽∘	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
8	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
9	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
10	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
11	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
12	△†	△†	△†	△†	△†	△†	△†	△†	▽†	△†	△†	△†	¬
13	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
14	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
15	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
16	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
17	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
18	▽†	▽†	△∘	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
19	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽∘	▽∘	△†	¬
20	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
21	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
22	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
—	20	20	2	21	21	2	21	21	3	20	20	2	▽†
—	1	1	0	0	0	0	0	0	0	1	1	0	▽∘
—	1	1	19	1	1	20	1	1	19	1	1	20	△†
—	0	0	1	0	0	0	0	0	0	0	0	0	△∘

Tablica 4.7: Učinkovitost korištenih metoda inicijalizacije u usporedbi s predloženom na funkcijama $f_1 \sim f_{22}$ za d=30.

f	NFEs = 100			NFEs = 200			NFEs = 400			NFEs = 600			DE_C
	DE_R	DE_O	DE_{QO}	DE_R	DE_O	DE_{QO}	DE_R	DE_O	DE_{QO}	DE_R	DE_O	DE_{QO}	
1	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
2	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
3	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
4	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
5	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
6	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
7	△○	△○	▽○	▽†	▽○	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
8	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
9	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽○	▽†	▽†	△†	¬
10	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
11	▽†	▽†	△○	▽†	▽†	▽○	▽†	▽†	△○	▽†	▽†	△†	¬
12	△†	△†	▽○	△○	△†	▽○	△†	△†	▽○	△†	△○	▽○	¬
13	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
14	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	△†	△†	△†	¬
15	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽○	▽†	▽†	▽○	¬
16	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
17	▽†	▽†	▽○	▽†	▽†	△○	▽†	▽†	△○	▽†	▽†	▽○	¬
18	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
19	▽†	▽†	△○	▽†	▽†	△○	▽†	▽†	△○	▽†	▽†	△†	¬
20	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
21	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
22	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
—	20	20	12	21	20	13	21	21	11	20	20	10	▽†
—	0	0	3	0	1	0	2	0	3	0	0	3	▽○
—	1	1	5	0	1	5	1	1	5	2	1	9	△†
—	1	1	2	1	0	2	0	0	3	0	1	0	△○

Tablice 4.6, 4.7, 4.8 i 4.9 prikazuju relativnu kvalitetu, u smislu greške optimizacije, početnih populacija stvorenih metodama korištenim u DE_R, DE_O i DE_{QO} u odnosu na predloženu metodu, korištenu u DE_C. Treba napomenuti da se prikazani rezultati odnose na najbolje rješenje u početnim populacijama. Na dnu tablica prikazani su ostvareni rezultati kao ukupni brojevi. U tablicama znak ▽ ukazuje da je prosječna kvaliteta lošija (veći prosjek greške optimizacije) u odnosu na onu ostvarenu predloženom metodom, dok je suprotno naznačeno znakom △. Nadalje, znakom † označeno je da se radi o statistički značajnoj razlici, dok je znakom ○ označeno suprotno. Kao statistički test korišten je Wilcoxonov test ranga s predznakom uz interval pouzdanosti od 95 %.

Iz prikazanih rezultata, očigledna je veća učinkovitost predložene metode inicijalizacije populacije u odnosu na ostale u usporedbi. Predložena metoda uspijeva stvoriti početne populacije koje sadrže bolja rješenja, bez obzira na dozvoljeni budžet. Ova prednost postojana je na velikoj većini instanci problema. Samo je nešto manje izražena u slučaju funkcija male dimenzionalnosti, što se može pripisati malom prostoru pretrage za te funkcije. Također, metoda korištena u DE_{QO} je samo u slučaju funkcija za $d = 10$ bila učinkovitija u odnosu na predloženu, što je bilo za očekivati s obzirom na prethodne rezultate u smislu postignutih ubrzanja. Slika 4.3 pruža uvid o kolikim se zapravo razlikama radi. Kao što je vidljivo, razlike su uglavnom značajnije izražene osim u slučaju metoda korištenih u DE_R i DE_O, što se može obrazložiti većinom simetričnim krajolicima dobrote (što je nepogodno za generiranje

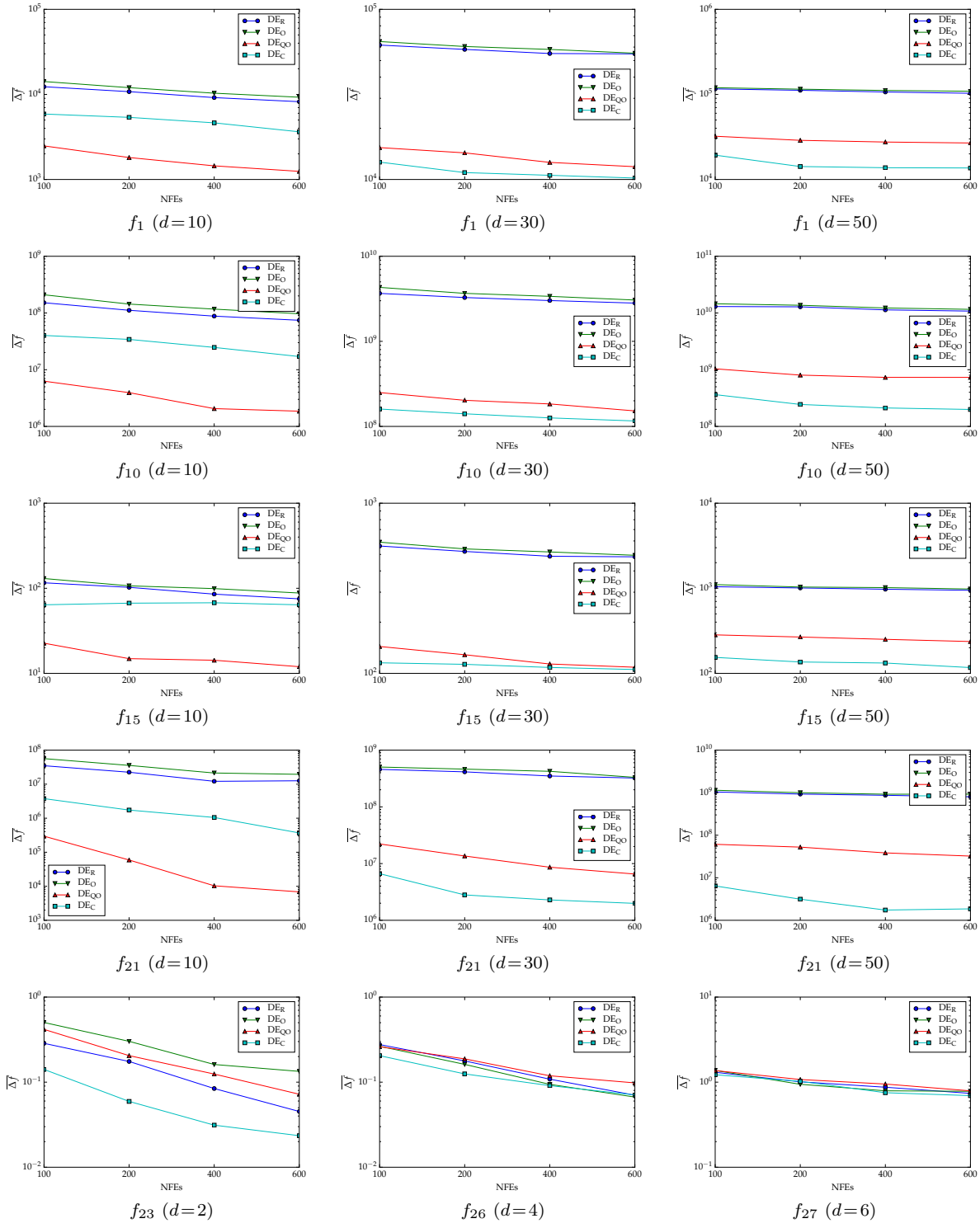
Tablica 4.8: Učinkovitost korištenih metoda inicijalizacije u usporedbi s predloženom na funkcijama $f_1 \sim f_{22}$ za d=50.

f	NFEs = 100			NFEs = 200			NFEs = 400			NFEs = 600			DE_C
	DE_R	DE_O	DE_{QO}	DE_R	DE_O	DE_{QO}	DE_R	DE_O	DE_{QO}	DE_R	DE_O	DE_{QO}	
1	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
2	▽†	▽†	△∘	▽†	▽†	△†	▽†	▽†	▽†	▽†	▽†	▽†	¬
3	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
4	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
5	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
6	▽†	▽†	△∘	▽†	▽†	△∘	▽†	▽†	△†	▽†	▽†	△†	¬
7	△∘	▽∘	▽†	▽†	▽∘	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
8	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
9	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
10	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
11	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
12	△†	△†	▽∘	△∘	△∘	▽∘	△†	△∘	▽∘	△∘	△∘	▽∘	¬
13	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
14	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
15	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
16	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
17	▽†	▽†	△∘	▽†	▽†	▽∘	▽†	▽†	▽∘	▽†	▽†	▽∘	¬
18	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	▽†	▽†	△†	¬
19	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
20	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
21	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
22	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
—	20	20	16	21	20	16	21	21	17	21	21	17	▽†
—	0	1	1	0	1	2	0	0	2	0	0	2	▽∘
—	1	1	2	0	0	3	1	0	3	0	0	3	△†
—	1	0	3	1	1	1	0	1	0	1	1	0	△∘

Tablica 4.9: Učinkovitost korištenih metoda inicijalizacije u usporedbi s predloženom na funkcijama $f_{23} \sim f_{30}$.

f	NFEs = 100			NFEs = 200			NFEs = 400			NFEs = 600			DE_C
	DE_R	DE_O	DE_{QO}	DE_R	DE_O	DE_{QO}	DE_R	DE_O	DE_{QO}	DE_R	DE_O	DE_{QO}	
23	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
24	▽†	▽∘	△∘	▽†	▽∘	△∘	▽†	▽†	△∘	▽†	▽†	△∘	¬
25	▽∘	▽∘	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	▽†	¬
26	▽†	▽†	▽∘	▽†	▽∘	▽†	▽∘	▽∘	▽†	△∘	△∘	▽†	¬
27	▽∘	▽∘	▽∘	△∘	△∘	▽∘	▽∘	▽∘	▽†	▽∘	▽∘	▽∘	¬
28	▽†	▽†	▽∘	▽†	▽†	▽†	▽†	▽†	▽∘	▽†	▽†	▽†	¬
29	▽†	▽†	△∘	▽†	▽†	▽∘	▽†	▽†	△∘	▽†	▽†	▽†	¬
30	▽†	▽†	△∘	▽†	▽†	△∘	▽†	▽†	△∘	▽†	▽†	▽∘	¬
—	6	5	2	7	5	4	6	6	4	6	6	5	▽†
—	2	3	3	0	2	2	2	2	1	1	1	2	▽∘
—	0	0	0	0	0	0	0	0	0	0	0	0	△†
—	0	0	3	1	1	2	0	0	3	1	1	1	△∘

suprotnih točaka). Zanimljivo je primijetiti kako povećanje budžeta za stvaranje početne populacije, ne rezultira uvijek značajno ili pak vidljivo boljim rješenjima. To upućuje na zaključak da je u slučaju nekih funkcija (kako unimodalnih, tako i multimodalnih) potrebno opsežnije i usmjerenije istraživanje za pronalazak boljih rješenja, nego što su to sposobne obaviti korištene metode inicijalizacije sa svojim ograničenim brojem vrednovanja funkcije cilja.



Slika 4.3: Učinkovitost predložene i ostalih metoda inicijalizacije na nekoliko odabranih funkcija.

4.4.5 Učinkovitost na problemima viših dimenzionalnosti

Povećanje dimenzionalnosti povlači za sobom puno veće povećanje složenosti problema [141]. Međutim, mnoga su testiranja naprednih metoda inicijalizacije ograničena na probleme niskih ili srednjih dimenzionalnosti [60]. Primjerice, najveća dimenzionalnost problema u [110]

Tablica 4.10: Konvergencija u smislu ukupnog NFEs te ukupna vremena izvođenja (u sekundama) na funkcijama $f_1 \sim f_{22}$ za $d = 80$.

f	$\mathrm{DE_R}$		$\mathrm{DE_O}$		$\mathrm{DE_{QO}}$		$\mathrm{DE_C}$	
	Uk. NFEs	Uk. t	Uk. NFEs	Uk. t	Uk. NFEs	Uk. t	Uk. NFEs	Uk. t
1	5561090 (100%)	8.39	5562752 (100%)	8.68	5368238 (100%)	8.22	4864154 (100%)	7.64
2	6322668 (100%)	9.75	6307536 (100%)	9.97	6363755 (100%)	9.93	5618624 (100%)	9.11
3	51000000 (0%)↓	289.06	51000000 (0%)↓	309.4	51000000 (0%)↓	288.89	51000000 (0%)	288.90
4	51000000 (0%)↓	77.24	51000000 (0%)↓	79.43	40230352 (24%)↓	60.94	29045830 (47%)	44.15
5	50649028 (16%)↓	78.76	49783322 (16%)∘	79	49398444 (27%)∘	76.89	48991921 (29%)	77.56
6	58211 (100%)	0.44	55379 (100%)	0.47	5354 (100%)	0.03	5187 (100%)	0.19
7	51000000 (0%)∘	82	51000000 (0%)∘	87.53	51000000 (0%)∘	82.1	51000000 (0%)	82.51
8	5821168 (100%)	20.96	5216860 (100%)	18.76	5252823 (100%)	18.09	4245495 (100%)	14.79
9	1975364 (100%)	3.23	1936458 (100%)	3.71	1650398 (100%)	2.77	1456585 (100%)	2.51
10	5131207 (100%)	7.58	5152321 (100%)	8.27	4900216 (100%)	7.34	4427503 (100%)	6.66
11	1518466 (100%)	2.3	1519186 (100%)	2.38	1295765 (100%)	2.1	1185018 (100%)	1.94
Uk.	228518736	579.71	227014628	607.60	215169580	557.30	**200655299**	**535.96**
—	12.19 %		11.61 %		6.75 %		poboljšanje	
12	51000000 (0%)∘	302.03	50348924 (4%)∘	314.31	49729202 (8%)∘	294.5	50460092 (4%)	298.48
13	33794979 (100%)	166.77	33207938 (98%)∘	164.82	33762280 (98%)∘	166.17	33191362 (100%)	163.66
14	5286946 (100%)	24.24	5280364 (100%)	24.37	5062685 (100%)	24.54	4683014 (100%)	21.33
15	4082746 (100%)	26.10	4087976 (100%)	26.16	3870982 (100%)	24.84	3305625 (100%)	21.18
16	5582724 (100%)	27.37	5635421 (100%)	27.68	5286067 (100%)	25.92	4917952 (100%)	24.17
17	51000000 (0%)∘	346.18	51000000 (0%)∘	346.15	51000000 (0%)∘	345.85	51000000 (0%)	344.48
18	50126984 (14%)∘	420.04	50677866 (6%)∘	423.64	50274737 (12%)∘	420.32	50570779 (14%)	422.76
19	2007807 (100%)	10.51	2007754 (100%)	10.55	1743332 (100%)	9.19	1472313 (100%)	7.77
20	7383510 (100%)	11.28	7487588 (100%)	11.48	6973308 (100%)	10.75	6572080 (100%)	10.14
21	5664766 (100%)	36.58	8467427 (94%)∘	52.13	5796420 (98%)∘	36.01	4728001 (98%)	27.89
22	8767714 (98%)∘	55.47	9073743 (98%)∘	57.04	7387796 (98%)∘	46.19	5759299 (100%)	35.07
Uk.	226216642	1426.57	228794187	1458.35	222182574	1404.27	**217845535**	**1376.92**
—	3.70 %		4.79 %		1.95 %		poboljšanje	

bila je $d = 30$, a u [27] je bila $d = 60$, dok u je [115] bila $d = 50$. Stoga, pitanje o korisnosti različitih metoda inicijalizacije populacije u prostorima viših dimenzionalnosti uglavnom ostaje neodgovoreno. Kako bi se pružio, barem donekle, odgovor na ovo pitanje, provedeni su dodatno testiranje i analiza na instancama problema viših dimenzionalnosti.

Tablice 4.10 i 4.11 prikazuju ostvarene rezultate u smislu ukupnih brojeva vrednovanja funkcije cilja i ukupnih vremena izvođenja u sekundama. Također, u zagradama su dane stope uspješnosti u dosezanju ciljane greške optimizacije. Kako su *SR* na približno polovici instanci problema manje od 100 %, upotrijebljen je Wilcoxonov test ranga s predznakom da bi se procijenila razlika u kvaliteti rješenja ostvarenih s $\mathrm{DE_C}$ u odnosu na ostale algoritme u usporedbi. Pri tom, znak ↓ označava statistički značajnu razliku prosjeka u korist $\mathrm{DE_C}$, dok znak ∘ označava odsutnost iste. Također, u navedenim tablicama su prikazana i poboljšanja,

$$\text{poboljšanje} = \left(1 - \frac{\sum_{i=1}^{n} \text{Uk. NFEs}(f_i) \text{ za } \mathrm{DE_C}}{\sum_{i=1}^{n} \text{Uk. NFEs}(f_i) \text{ za } \mathrm{DE_{R/O/QO}}}\right) \cdot 100\,\%, \tag{4.10}$$

u smislu ukupnih NFEs, koja su računata analogno (4.9).

Prema prikazanim rezultatima, algoritam s ugrađenom predloženom metodom inicijalizacije ($\mathrm{DE_C}$) postigao je značajna poboljšanja u smislu ukupnih NFEs u usporedbi s algoritmima u koje su ugrađene druge upotrijebljene metode inicijalizacije ($\mathrm{DE_R}$, $\mathrm{DE_O}$ i $\mathrm{DE_{QO}}$).

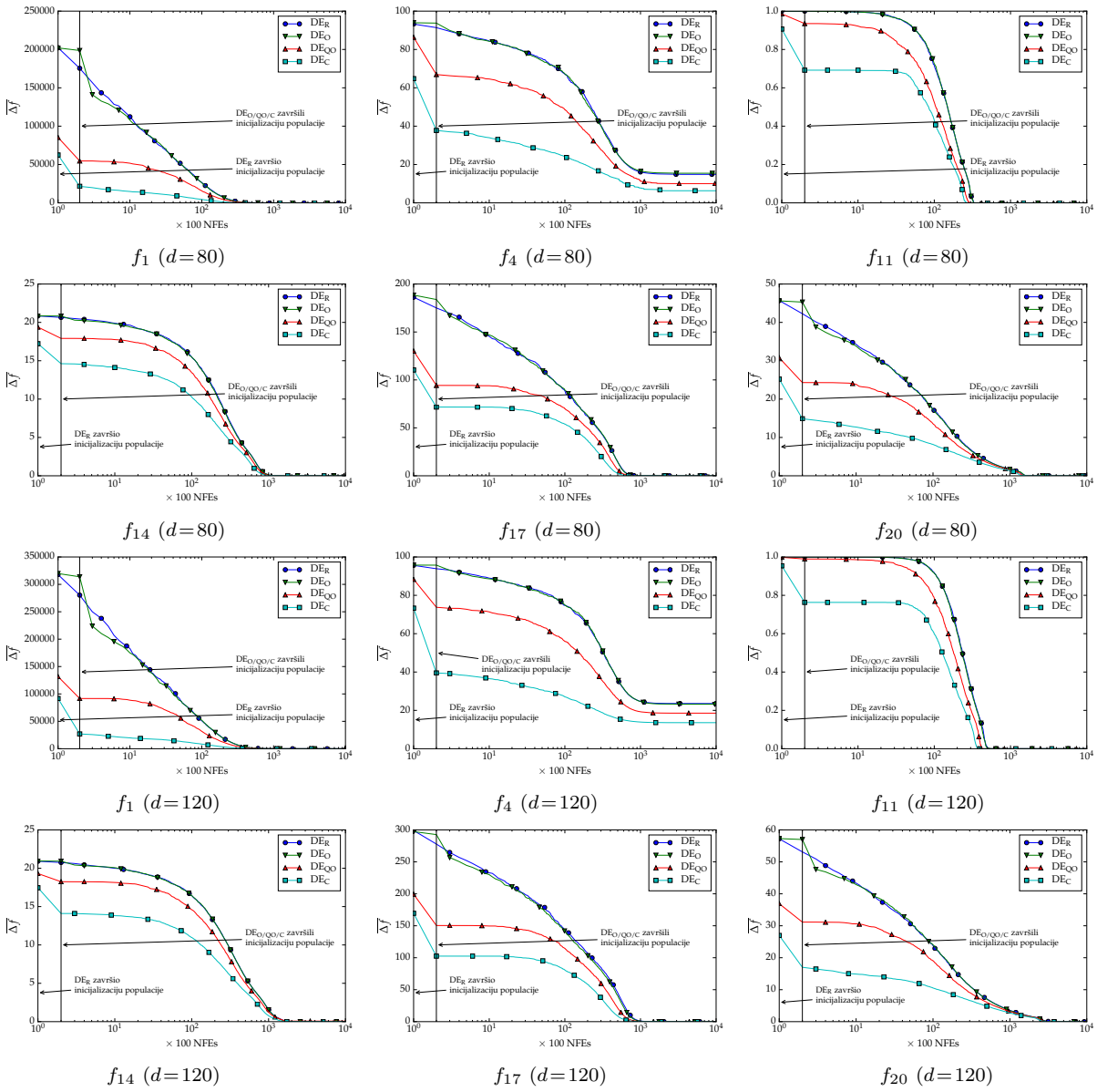
Tablica 4.11: Konvergencija u smislu ukupnog NFEs te ukupna vremena izvođenja (u sekundama) na funkcijama $f_1 \sim f_{22}$ za $d = 120$.

f	DE_R		DE_O		DE_{QO}		DE_C	
	Uk. NFEs	Uk. t	Uk. NFEs	Uk. t	Uk. NFEs	Uk. t	Uk. NFEs	Uk. t
1	8320417 (100%)	18.28	8365408 (100%)	20.54	8041823 (100%)	18.04	7250460 (100%)	15.97
2	8935497 (100%)	20.88	8865383 (100%)	22.97	9069939 (100%)	21.37	8014898 (100%)	18.99
3	51000000 (0%)↓	580.76	51000000 (0%)↓	624.73	51000000 (0%)↓	594.21	51000000 (0%)	581.13
4	51000000 (0%)↓	112.34	51000000 (0%)↓	116.74	51000000 (0%)↓	112.92	51000000 (0%)	113.19
5	51000000 (0%)↓	116.24	51000000 (0%)↓	123.82	51000000 (0%)∘	117.11	51000000 (0%)	117.41
6	70072 (100%)	0.77	67857 (100%)	0.91	5411 (100%)	0.06	5192 (100%)	0.25
7	51000000 (0%)∘	121.88	51000000 (0%)∘	123.26	51000000 (0%)∘	122.19	51000000 (0%)	122.73
8	13917571 (100%)	66.21	14461952 (100%)	76.06	13372541 (100%)	70.1	13294543 (100%)	69.16
9	3192968 (100%)	7.69	3231835 (100%)	7.92	2696247 (100%)	6.6	2237126 (100%)	5.59
10	7622356 (100%)	16.55	7606594 (100%)	16.78	7322335 (100%)	16.05	6637848 (100%)	14.57
11	2321475 (100%)	5.15	2273935 (100%)	5.24	1971820 (100%)	4.6	1712211 (100%)	4.01
Uk.	246058881	1066.74	246599029	1138.97	244508296	1083.27	**241440067**	**1062.99**
—	1.88 %		2.09 %		1.25 %		poboljšanje	
12	51000000 (0%)∘	449.64	51000000 (0%)∘	453.43	51000000 (0%)∘	469.28	51000000 (0%)	450.12
13	35748731 (96%)∘	262.69	33990638 (98%)∘	252.46	34914843 (90%)∘	260.82	33562618 (88%)	247.79
14	9034857 (98%)∘	62.51	8851869 (98%)∘	60.65	11269165 (92%)∘	79.33	8001322 (98%)	53.87
15	5829035 (100%)	55.6	5906013 (100%)	56.56	5658965 (100%)∘	54.36	4826398 (100%)	46.19
16	8452034 (100%)	61.62	8460913 (100%)	61.76	7772171 (100%)	57.03	7399674 (100%)	54.21
17	51000000 (0%)∘	518.42	51000000 (0%)∘	542.31	51000000 (0%)∘	574.45	51000000 (0%)	535.83
18	51000000 (0%)∘	653.76	51000000 (0%)↓	641.81	51000000 (0%)∘	676.13	51000000 (0%)	641.47
19	2575682 (100%)	20.04	2579520 (100%)	20.86	2251244 (100%)	18.46	1908043 (100%)	15.55
20	13632534 (100%)	30.39	13485363 (100%)	30.04	13176305 (100%)	29.58	12562284 (100%)	28.10
21	17003012 (90%)∘	151.9	14533313 (96%)∘	130.59	9995095 (98%)∘	90.38	6442072 (100%)	56.05
22	25465924 (92%)∘	230.74	28781319 (84%)∘	258.86	24007558 (88%)∘	215.75	14645207 (96%)	131.42
Uk.	273063284	2497.29	271862883	2509.32	264017166	2525.57	**244059829**	**2260.60**
—	10.62 %		10.23 %		7.56 %		poboljšanje	

Ovo se odnosi kako na unimodalne, tako i na multimodalne funkcije. Navedena poboljšanja ogledaju se i u ukupnim vremenima izvođenja, a koja su za DE_C na većini instanci problema manja u usporedbi. Nadalje, slično ponašanje kao prethodno (Sl. 4.2), može se vidjeti na slici 4.4. Može se primijetiti opet značajna prednost DE_C u odnosu na druge algoritme. Navedena prednost opada s NFEs, ali s obzirom na rezultate u smislu ostvarenih poboljšanja uglavnom je uspijeva zadržati. Treba primijetiti kako su u slučajevima gdje su *SR* bile manje od 100 % često postignuta bolja rješenja (statistički značajne razlike prosjeka grešaka optimizacije) na unimodalnim funkcijama. Osim toga, slika 4.4 sugerira da predložena metoda uspijeva i u slučaju prostora viših dimenzionalnosti stvoriti početne populacije koje sadrže relativno dobra rješenja.

Prikazani rezultati pružaju (barem donekle, odnosno za slučajeve razmatranih instanci problema) odgovor na pitanje korisnosti naprednih metoda inicijalizacije populacije u prostorima pretrage viših dimenzionalnosti. Prema navedenom, odgovor ide u korist naprednih metoda inicijalizacije. Dobiveni rezultati jasno sugeriraju da te metode mogu biti korisne u prostorima viših dimenzionalnosti te da su sposobne postići prednost nad standardnim, potpuno nasumičnim pristupom inicijalizaciji populacije.

Treba istaknuti da su dodatni rezultati na nekoliko funkcija za $d = 500$ i $d = 1000$ dostupni u [9]. Testiranje i analiza pokazali su da predložena metoda može i na tim problemima, vrlo visoke dimenzionalnosti (engl. *large-scale optimization problems*), pružiti dobru polaznu



Slika 4.4: Grafovi konvergencije na nekoliko odabranih instanci problema viših dimenzionalnosti. Usporedba predložene, standardne i metoda iz literature.

poziciju za daljnju pretragu. Međutim, s obzirom da povećanje dimenzionalnosti problema nije moguće pratiti povećanjem računalnih resursa, za postizanje dobrih rješenja dodatno su nužna unaprjeđenja posebno prilagođena takvim vrstama problema.

4.5 Osvrt na inicijalizaciju populacije i predloženu metodu

Početna populacije je važan čimbenik svih prirodom inspiriranih algoritama optimizacije pa tako i algoritma DE. Međutim, često joj se ne posvećuje posebna pažnja te je se naj-

češće stvara potpuno nasumično. Rezultati provedene eksperimentalne analize ukazuju da upotrijebljene napredne metode inicijalizacije, a posebno predložena, ipak mogu potpomoći učinkovitost algoritma koji ih koristi. Predložena metoda, u pogledu različitih veličina populacije, uspijeva stvoriti početne populacije koje sadrže relativno dobra rješenja. Njena korisnost se u prvom redu odrazila na brzinu konvergencije, a što se općenito odrazilo i na smanjenje vremena izvođenja. Navedeno je od posebnog značaja kada je vrednovanje funkcije cilja računalno zahtjevno ili kada je broj istih ograničen, jer može pružiti dostizanje zadovoljavajućih rješenja problema u kraćem vremenu.

Treba primijetiti kako su prostori pretrage korištenih funkcija relativno veliki, odnosno veliki su rasponi vrijednosti koje mogu poprimiti pojedine nezavisne varijable. Veličina prostora, u navedenom smislu, bitna je zbog korištene mutacije, točnije, parametra skaliranja s Cauchyjeve razdiobe koji predstavlja intenzitet mutacije. Prema tome, u slučaju značajno većih ili značajno manjih prostora, nužno je prilagoditi navedeni parametara razdiobe. Naime, pri vrlo velikim prostorima male vrijednosti s, ne mogu pružiti dovoljno istraživanje, dok pri malim prostorima veće vrijednosti s dovode do ponašanja sličnog uporabi uniformne razdiobe. Slično tome, ako se rasponi vrijednosti pojedinih nezavisnih varijabli uvelike razlikuju, potrebno je za svaku posebno prilagoditi parametar skaliranja ili napraviti transformaciju prostora pretrage.

Kao glavni nedostatak predložene metode se jasno ističe njena računalna zahtjevnost, što se uglavnom može pripisati koraku grupiranja početno zadane nasumične populacije. Međutim, ostvareni rezultati sugeriraju da se vrijeme potrošeno na inicijalizaciju i više nego nadoknađuje tijekom izvođenja algoritma. Još jedan nedostatak predložene metode je postojanje mogućnosti stvaranja grupa rješenja koja su isključivo u blizini lokalnih optimuma što može potencijalno dovesti do preuranjene konvergencije ili stagnacije. Ipak, rezultati ostvareni u analizi ne ukazuju na pojave takvih situacija. Štoviše, vjerojatnost pojave navedenog može se jednostavno smanjiti povećanjem omjera broja nasumično stvorenih i rješenja stvorenih grupiranjem i Cauchyjevom mutacijom. Povećanjem broja nasumično stvorenih rješenja izravno se povećava raznolikost početne populacije što može biti korisno za neke vrste problema.

Metoda inicijalizacije populacije DE zasnovana na grupiranju podataka i Cauchyjevim slučajnim varijablama zajedno s provedenom eksperimentalnom analizom i s obzirom na postignute rezultate koji joj idu u prilog, predstavlja ispunjenje prijedloga drugog izvornog znanstvenog doprinosa ove disertacije.

POGLAVLJE 5

Mutacija diferencijalne evolucije

Mutacija diferencijalne evolucije od izuzetnog je značaja za njenu učinkovitost. Ona izravno utječe na mogućnost otkrivanje novih dijelova prostora pretrage te na mogućnost istraživanja okoline obećavajućih točaka tog prostora. S obzirom da se mutacijom obavlja istraživanje u okolini baznog vektora, njegov odabir ima istaknutu ulogu u usmjeravanju pretrage. U tom smislu, mali selekcijski pritisak omogućuje opsežnije istraživanje cijelog prostora, dok veliki pritisak omogućuje opsežnije istraživanje okoline nekolicine točaka u istom. Međutim, za postizanje učinkovite pretrage neophodno je traženje ravnoteže između ta dva pristupa, jer oba imaju pozitivna i negativna svojstva. U ovom poglavlju predlaže se mutacija DE koja uvodi selekcijski pritisak u odabir baznog vektora pomoću k-turnirske selekcije. Veličine k-turnira zadanih za svakog člana populacije posebno, prilagođavaju se, a time i selekcijski pritisak, s ciljem upravljanja istraživanjem i iskorištavanjem. Opsežnim testiranjem i naknadnom analizom, ispitan je i vrednovan ovaj prijedlog trećeg izvornog znanstvenog doprinosa.

5.1 Motivacija

Kao i drugi uobičajeni EAs, tako i DE koristi varijacijske operatore, odnosno križanje i mutaciju, te selekciju za pokretanje procesa pretrage. Međutim, ključnu ulogu u učinkovitosti DE nedvojbeno igra mutacija [47, 72, 80]. Općenito, ona se može smatrati oblikom lokalne pretrage oko baznog vektora [72], s obzirom da ga se perturbira s jednim ili više vektora razlike koji određuju intenzitet iste. Prema tome, odabir baznog vektora ima bitan utjecaj na sposobnost algoritma da obavlja istraživanje (engl. *exploration*) različitih neotkrivenih dijelova prostora i iskorištavanje (engl. *exploitation*) u smislu pretrage okoline ili susjedstva pronađenih obećavajućih točaka tog prostora. Tako dvije često korištene mutacije, rand/δ i best/δ, gdje je uglavnom $\delta \in \{1, 2\}$, predstavljaju dva granična slučaja u odabiru baznog

vektora.

Prema navedenom, lako se može zaključiti da manji selekcijski pritisak pri odabiru baznog vektora pogoduje istraživanju i potpomaže očuvanju raznolikosti populacije te koristi pri izbjegavanju zaglavljivanja u lokalnim optimumima. Naprotiv, veći selekcijski pritisak pogoduje iskorištavanju i vodi prema bržoj konvergenciji koja, međutim, može dovesti do zaglavljivanja u lokalnom optimumu. Stoga, da bi se postigla učinkovita pretrage, iznimno je važno uspostaviti ravnotežu između istraživanja i iskorištavanja [11, 134, 152]. Pretjerano istraživanje može zahtijevati veliki broj vrednovanja funkcije cilja kako bi se pronašla zadovoljavajuća ili dobra rješenja. S druge strane, pretjerano iskorištavanje može dovesti do preuranjene konvergencije ili stagnacije pretrage zbog brzog gubitka raznolikosti populacije.

5.2 Pregled literature

Postizanje ravnoteže između istraživanja i iskorištavanja je ključ postizanja učinkovite pretrage ne samo u DE, nego i svim stohastičkim algoritmima optimizacije. Mutacija je u DE od izuzetnog značaja za njegovu učinkovitost, jer može izravno utjecati na istraživanje i iskorištavanje. Također, slično kao što je slučaj s parametrima algoritma, različiti oblici mutacije mogu biti pogodni za različite probleme [104], a slična je situacija i tijekom pretrage [164]. Stoga je u literaturi moguće naći mnoštvo prijedloga novih ili izmjena postojećih mutacija te prvenstveno načina određivanja vektora koji sudjeluju u njoj kao što je opisano u nastavku.

Kaelo i Ali [58] predložili su izmjenu mutacije rand/1, gdje se najbolji od nasumično odabranih vektora postavlja za baznog. Isto su kasnije predložili i Lin et al. [72], dok su Martinović i Bajer [80] dalje ispitali učinak pri postavljanju redoslijeda i preostala dva vektora koji tvore vektor razlike (perturbaciju). Navedeni pristup u osnovi predstavlja 3-turnirsku selekciju bez zamjene za odabir baznog vektora pa dva najlošija u populaciji ne mogu biti nikad odabrana. Na neki način, korak dalje napravili su Qiu et al. [108], prijedlogom uporabe 3-turnirske selekcije za odabir baznog, ali i terminalnog vektora (prvi od dva koji čine vektor razlike). Jednostavnu izmjenu rand/1 predložili su Li et al. [68] u kojoj se za terminalni vektor umjesto nasumično odabranog, postavlja drugi najbolji u trenutnoj populaciji. Pristup mutaciji, odnosno odabiru baznog ili terminalnog vektora, koji su predložili Yu et al. [152] pridružuje svakom vektoru populacije $\mathbf{v}^j$ parametar $t_j \in \{1, \ldots, NP\}$, gdje je NP veličina populacije. On određuje koliki dio najboljih iz populacije sudjeluje u odabiru baznog ili terminalnog vektora. Pristup se može rabiti za odabir terminalnog ukoliko je u nekoj mutaciji ciljni vektor sudjeluje u izračunu baznog (primjerice, target-to-best/δ). Vrijednost t_j pridružena nekom vektoru se u slučaju neuspjeha [pokusni vektor (potomak) nije prešao u novu generaciju] nanovo i nasumično stvara, dok se u suprotnom prenosi na potomka. Odabir između t_j najboljih iz populacije je nasumičan pa prema tome manje vrijednosti parametra predstavljaju veći selekcijski pritisak i obratno. Znatno drukčiju izmjenu

mutacije DE predložili su Bansal i Sharma [11] prema kojoj je uveden parametar $C \in [0, 1]$, a koji određuje težinu baznog vektora. Navedeni parametar sličan je faktoru skaliranja te množi samo bazni vektor, a linearno ga se povećava (od 0.1 do 1) čime se vremenom sve veći značaj daje tom vektoru. Nadalje, Yu et al. [153] predložili su novi oblik mutacije, lbest/1 koji se može smatrati lokalnom inačicom best/1, a u kojoj se populacija dijeli u zadani broj grupa. Za dani ciljni vektor, bazni vektor se odabire kao najbolji iz iste grupe, dok se vektori koji čine razliku biraju iz cijele populacije. Podjela u grupe vrši se, jednostavno, prema položajima, odnosno indeksima vektora u populaciji. Izmjenu mutacije rand-to-best/1 koja koristi u osnovi dvije razlike (kao target-to-best/1, ali uz nasumično odabrani vektor umjesto ciljnog) su predložili Abbas et al. [1]. U navedenoj izmjeni se umjesto najboljeg vektora trenutne populacije koji je terminalni vektor u prvoj razlici i nasumično odabranog koji je terminalni u drugoj razlici, odabiru dva vektora pomoću 3-turnirskih selekcija. U prvoj turnirskoj selekciji se odabire pobjednik, dok se u drugoj odabire drugi po kvaliteti koji sudjeluje u danom turniru. Nadalje, Zhou et al. [165] predložili su uvođenje arhive iz koje se rješenja koriste kao alternativni izvor baznih vektora. Osim toga, za svaki vektor populacije $\mathbf{v}^j$ je vezan podatak s_j koliko uzastopnih generacija nije bio zamijenjen boljim rješenjem. Shodno tome, ako taj broj za dani ciljni vektor premaši postavljeni prag S (pretpostavka da je došlo do stagnacije), bazni vektor se odabire iz arhive umjesto trenutne populacije. Odabir se vrši pomoću jednostavne selekcije pri čemu su vjerojatnosti odabira ovisne o kvaliteti pohranjenih vektora. Štoviše, ukoliko ni bazni vektor odabran iz arhive ne rezultira kroz određeni broj generacija unapređenjem, odabir se ponovo provodi. Arhiva, zadane veličine, puni se dobrim, ali istovremeno raznolikim rješenjima. Opisani pristup uvodi niz dodatnih parametara koje je potrebno postaviti te posebnu proceduru za popunjavanje arhive.

Prethodno opisane izmjene mutacije utječu samo na izbor baznog ili terminalnog (ili terminalnih) vektora. Osim toga, one ne uzimaju u obzir cjelokupno stanje populacije. Epitropakis et al. [38] predložili su shemu koja uvodi vjerojatnosti odabira za sve vektore koji mogu sudjelovati u mutaciji. Vjerojatnosti odabira se računaju prema međusobnim udaljenostima između svih vektora populacije, pri čemu su vjerojatnosti obrnuto proporcionalne udaljenosti. Vektori koji će sudjelovati u mutaciji odabiru se jednostavnom selekcijom prema dodijeljenim vjerojatnostima. Vrlo sličan pristup odabira svih vektora koji će sudjelovati u mutaciji, predložili su Cai i Wang [20]. Tako se za dani ciljni vektor računaju vjerojatnosti odabira prema njegovoj udaljenosti od ostalih u populaciji. Što su mu vektori bliži veća je vjerojatnost njihovog sudjelovanja u mutaciji. Osim toga, za bazni vektor se postavlja najbolji između odabranih (turnirska selekcija). U oba navedena pristupa se vjerojatnosti odabira za dani ciljni vektor $\mathbf{v}^j$ računaju kao $p_{j,k} = 1 - \frac{d(\mathbf{v}^j, \mathbf{v}^k)}{\sum_{k=1}^{NP} d(\mathbf{v}^j, \mathbf{v}^k)}$, $k = 1, \ldots, NP$, $j \neq k$, gdje je $d(\cdot, \cdot)$ korištena mjera udaljenosti (uobičajeno Euklidska udaljenost), a NP veličina populacije. Gong i Cai [47] predložili su postupak koji izmjenjuje odabir baznog i terminalnog vektora, a koji je općenit i time primjenjiv u mnogim operatorima mutacije DE kao i

prethodno dva opisana. U navedenom postupku računaju se i pridružuju vjerojatnosti odabira p_j za svaki vektor populacije. Vjerojatnosti se računaju prema rangovima r_j koji su izravno proporcionalni kvaliteti pojedinih vektora (za uzlazno sortiranu populaciju veličine NP, $r_j = NP - j$, $j = 1, \ldots, NP$). Što je vektor bolji, to mu je rang veći, a samim tim i pridružena vjerojatnost odabira, odnosno $p_j = r_j/NP$, $j = 1, \ldots, NP$. Potpuno drukčiji pristup odabiru svih vektora koji će sudjelovati u mutaciji predložili su Guo et al. [51], a koji utječe i na odabir vektora koji će biti korišten pri križanju. Naime, za svakog člana populacije vezan je podatak s_j koji pokazuje koliko generacija član nije zamijenjen boljim rješenjem. Ukoliko taj podatak za dani ciljni vektor prelazi postavljeni prag, $s_j > S$, vektori koji će sudjelovati u mutaciji i vektor s kojim će biti križan dobiveni mutant odabiru se iz arhive, dok se u suprotnom odabiru iz trenutne populacije. Arhiva, jednake veličine kao i populacija, puni se uspješnim pokusnim vektorima, odnosno onima koji su zamijenili roditelja. Odabir rješenja iz arhive je nasumičan. Opisani pristup, iako se koristi arhivom slično kao onaj predložen u [165], značajno je manje složen u pogledu dodatnih parametara, popunjavanja arhive i odabira rješenja iz iste.

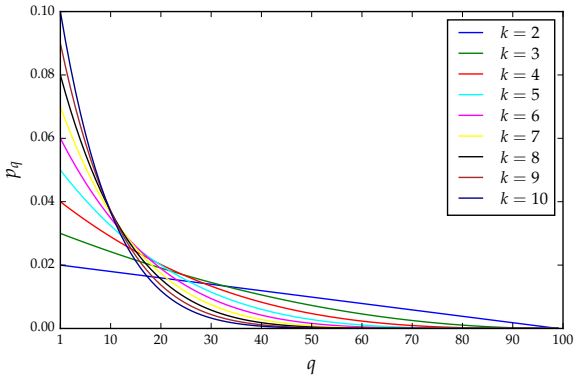
5.3 Mutacija DE zasnovana na prilagodljivoj turnirskoj selekciji

Predložena mutacija zasniva se na uvođenju selekcijskog pritiska pri odabiru baznog vektora uporabom k-turnirske selekcije. Veličine turnira zadane su za svakog člana populacije posebno, a one se podešavaju odnosno prilagođavaju tijekom pretrage. S obzirom da turnirska selekcija ima ključnu ulogu u predloženoj mutaciji, ona je u nastavku prvo sažeto opisana. Iako u literaturi postoje mutacije koje se koriste turnirskom selekcijom, veličine turnira su fiksne (često turniri s tri sudionika) i u pravilu jednake za sve članove populacije.

5.3.1 Turnirska selekcija u evolucijskim algoritmima

Selekcija u EAs u pravilu se provodi u dva navrata. Prvi puta se provodi pri odabiru roditelja, odnosno jedinki na temelju kojih će se stvoriti nova generacija, a u slučaju DE pod tim se može smatrati odabir vektora koji će sudjelovati u mutaciji. Drugi puta se provodi pri odabiru nove ili naredne generacije. Općenito, postoji mnoštvo različitih oblika za provođenje selekcije u EAs (vidi primjerice [34, 155]). Jedan od oblika koji omoguće jednostavno postavljanje selekcijskog pritiska je k-turnirska selekcija [43, 144] koja uz to ima niz povoljnih svojstava.

Jedno od povoljnih svojstava je njezina jednostavnost. Tako se u standardnoj inačici prvo nasumično odabire k jedinki iz populacije, a potom se pobjednikom proglašava najbolja između njih. Sudionike turnira može se birati sa ili bez zamjene, čime se dopušta ili



Slika 5.1: Vjerojatnosti odabira jedinki u ovisnosti o njenom rangu q u populaciji od 100 jedinki.

onemogućuje sudjelovanje iste jedinke više puta u istom turniru. Prema tome, u k-turnirskoj selekciji bez zamjene, $k-1$ najlošijih jedinki nikada ne može biti odabrano. Lako je zaključiti kako se povećanjem turnira k izravno povećava selekcijski pritisak. Vjerojatnost odabira neke jedinke ovisi o njenom rangu unutar populacije. Nije teško doći do vjerojatnosti odabira neke jedinke uz pretpostavku da je populacija sortirana i potpuno raznolika. Tako je vjerojatnost odabira q-te jedinke

$$p_q = \frac{\binom{NP-q}{k-1}}{\binom{NP}{k}}, \quad q = 1, \ldots, NP, \tag{5.1}$$

gdje je NP veličina populacije koja je sortirana uzlazno u smislu funkcije cilja, odnosno vrijedi $f_1 < f_2 < \cdots < f_{NP}$ (slučaj problema minimiziranja). Slikom 5.1 ilustrirane su za različite veličine turnira vjerojatnosti odabira jedinki populacije ovisno o njihovim rangovima u populaciji.

Uz jednostavnost k-turnirske selekcije može se povezati i mala vremenska složenost koja se za provođenje NP turnira može procijeniti s $O(NP \cdot k)$. S obzirom da je prilikom odabira važan samo međusobni odnos sudionika turnira, ona ne zahtijeva informacije o cijeloj populaciji do kojih, prema Eiben i Smith [34], u određenim situacijama nije jednostavno ili pak nije moguće doći. Navedeno čini k-turnirsku selekciju vrlo pogodnom za ugradnju u paralelne izvedbe EAs (vidi primjerice [46]).

5.3.2 Opis predložene mutacije

Predloženom mutacijom stvara se mutant ili donor

$$\mathbf{u}^j = \mathbf{v}^t + F_j \cdot (\mathbf{v}^{r1} - \mathbf{v}^{r2}), \tag{5.2}$$

gdje je $\mathbf{v}^t$ bazni vektor, a vektor razlike čine $\mathbf{v}^{r1}$ i $\mathbf{v}^{r2}$, redom. Odabir $\mathbf{v}^t$ provodi se turnirskom selekcijom, a veličina turnira i time selekcijski pritisak prilagođavaju se tijekom pretrage.

Svakom vektoru populacije $\mathbf{v}^j$ dodijeljena je vrijednost $\kappa_j \in [0, 1\rangle$ koja određuje veličinu k_j-turnira pri odabiru odgovarajućeg baznog vektora, što je ilustrirano slikom 5.2. Prilagodba je vođena temeljnom veličinom za sve turnire, a čija vrijednost utječe na tendenciju populacije da obavlja istraživanje odnosno iskorištavanje. Ostali vektori (koji tvore vektore razlike) odabiru se nasumično pa se u tom smislu predložena mutacija može promatrati kao proširenje mutacije rand/1 ili best/1. Treba istaknuti kako predložena mutacija nije ograničena na jedan vektor razlike, a proširenja za $\delta > 1$ su trivijalna i slijede prema (2.1).

Svake generacije g, prilikom odabira vektora koji će sudjelovati u mutaciji, za dani ciljni vektor $\mathbf{v}^{j,g}$, prvo se odabire bazni $\mathbf{v}^{t,g}$. Njegov odabir se provodi turnirskom selekcijom čija je veličina zadana dodijeljenom vrijednosti κ_j^g. S obzirom da je $\kappa_j^g \in [0, 1\rangle$, kako bi se dobila veličina turnira, potrebno je obaviti transformaciju

$$k_j = T(\kappa_j^g; k_{max}) = \lfloor \kappa_j \cdot k_{max} \rfloor + 1, \tag{5.3}$$

gdje $k_{max} = 10$ predstavlja najveću dozvoljenu veličinu turnira. Ova vrijednost predstavlja uobičajenu veličinu turnira u evolucijskom programiranju (engl. *evolutionary programming*) [103]. Mora se napomenuti da se provodi k_j-turnirska selekcija bez zamjene. Ostale vektore koji će sudjelovati u mutaciji, odnosno one koji tvore vektor razlike, $\mathbf{v}^{r1,g}$ i $\mathbf{v}^{r2,g}$ odabire se nasumično tako da $j \models t \neq r1 \neq r2$, $j, t, r1, r2 \in \{1, \ldots, NP\}$, gdje $\models$ označava da ciljni i bazni vektor mogu biti isti član populacije. Prema (5.3), veličina turnira je u rasponu $[1, k_{max}]$ pa se ne provodi uvijek turnir, jer u slučaju $k_j = 1$ vrši se zapravo nasumični odabir baznog vektora. Postupak odabira vektora koje će sudjelovati u mutaciji prikazan je algoritmom 5.1.

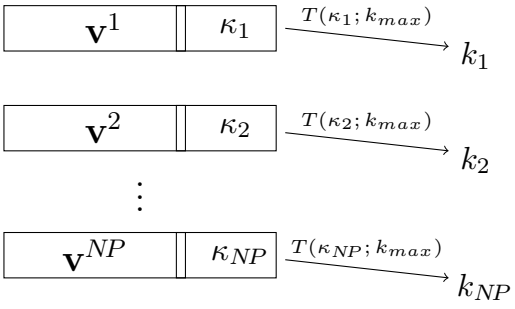
Algoritam 5.1: Odabir vektora u predloženoj mutaciji

Za $\mathbf{v}^{j,g}$ izračunaj veličinu turnira k_j; % prema (5.3)
$\mathcal{I}_T = \emptyset$, $s = 0$;
% Određivanje vektora koji će sudjelovati u k_j-turniru
ponavljaj
  nasumično odaberi $r \in \{1, \ldots, NP\}$, $r \notin \mathcal{I}_T$;
  $\mathcal{I}_T \leftarrow r$;
  $s := s+1$;
dok $s < k_j$;
Odredi bazni vektor $\mathbf{v}^{t,g}$, $t \in \mathcal{I}_T$, $f(\mathbf{v}^t) \leq f(\mathbf{v}^r)\ \forall r \in \mathcal{I}_T$;
Nasumično odaberi $\mathbf{v}^{r1,g}, \mathbf{v}^{r2,g}$, tako da $j \models t \neq r1 \neq r2 \in \{1, \ldots, NP\}$;

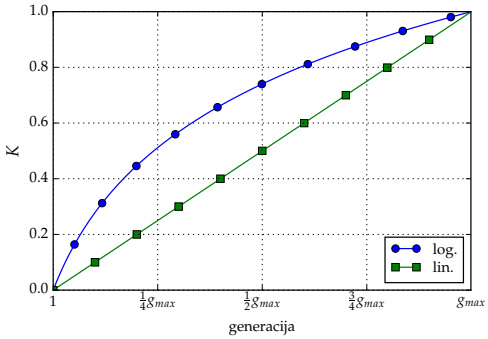
U cilju prilagodbe selekcijskog pritiska pri odabiru baznog vektora, tijekom pretrage, vrijednosti κ_j^g dodijeljene svakom vektoru $\mathbf{v}^{j,g}$ se mijenjaju. Do izmjena dolazi prilikom selekcije nove generacije. Ukoliko pokusni vektor (potomak) prijeđe u narednu generaciju $(g+1)$ zamjenjujući roditelja, vrši se samo blaga izmjena

$$\kappa_j^{g+1} = \mu_j + \mathcal{N}_j(0, 0.1), \quad \mu_j = K \cdot (1 - \alpha) + \alpha \cdot \kappa_j^g, \tag{5.4}$$

dok se u suprotnom generira nova vrijednost



Slika 5.2: Vrijednosti za izračun veličine turnira pridružene članovima populacije pri predloženoj mutaciji.



Slika 5.3: Povećavanje temeljne veličine za sve turnire. Prikaz razlike između linearnog i rasta po logaritamskoj funkciji.

$$\kappa_j^{g+1} = K + \mathcal{C}_j(0, 0.1), \tag{5.5}$$

gdje je $\mathcal{N}_j(0, 0.1)$ Gaussova slučajna varijabla sa srednjom vrijednosti $\mu = 0$ i standardnom devijacijom $\sigma = 0.1$, $\mathcal{C}_j(0, 0.1)$ je Cauchyjeva slučajna varijabla s parametrom lokacije $l = 0$ i parametrom skaliranja $s = 0.1$, a $K \in [0, 1\rangle$ je temeljna veličina za sve turnire. Prema (5.4), nova vrijednost se generira u okolini težinske srednje vrijednosti prethodne i temeljne K, gdje je $\alpha = 0.9$. Kako se koristi Gaussova slučajna varijabla sa $\sigma = 0.1$, nova vrijednost κ_j^{g+1} bit će blizu sredine koja je zbog $\alpha = 0.9$ samo blago pomaknuta prema temeljnoj K u odnosu na prethodnu κ_j^g. Razlog izmjene čak i uspješnih vrijednosti je postojanje opasnosti da u ranim fazama pretrage dođe do prevladavanja vektora za koje se provode veliki turniri. Naime, zbog većeg selekcijskog pritiska, veća je vjerojatnost dobivanje dobrih mutanta i time potomaka, odnosno pokusnih vektora. Navedeno bi moglo dovesti do osjetnog opadanja istraživanja vrlo rano tijekom pretrage. Nadalje, prema (5.5), nova vrijednost generira se u okolini temeljne K, ali zbog uporabe Cauchyjevih slučajnih varijabli, ta okolina je znatno veća u odnosu za slučaj Gaussovih slučajnih varijabli.

Treba primijetiti da se nove vrijednosti κ_j^{g+1} pridružene vektorima populacije, računaju u manjoj ili većoj mjeri pomoću temeljne veličine K. Stoga, na razini cijele populacije, manje vrijednosti K promiču istraživanje, dok veće promiču iskorištavanje. Kako bi se promoviralo opsežnije istraživanje tijekom ranih faza pretrage, a kasnije veće iskorištavanje dobrih rješenja, obavlja se dinamička ili deterministička prilagodba temeljne veličine

$$K = \log\left(1 + \frac{9}{g_{max}} \cdot g\right), \tag{5.6}$$

gdje je g trenutna generacija od maksimalno dozvoljenih g_{max}. Povećavanjem K po logaritamskoj funkciji (baza 10) postiže se brži rast u odnosu na primjerice, linearno povećanje (Sl. 5.3).

Algoritam 5.2: Nacrt rada DE s ugrađenom predloženom mutacijom

```
Postavi vrijednosti parametara algoritma;
Inicijaliziraj populaciju P = (v^{1,0}, ..., v^{NP,0});
Postavi κ_j^0 = U_j[0,1⟩, j = 1, ..., NP i K = 0; ⇐
g := 0;
ponavljaj
    Izračunaj vrijednost temeljne veličine za sve turnire K; ⇐          % prema (5.6)
    za j := 1, ..., NP čini
        % Stvaranje pokusnih vektora
        Odaberi vektore za sudjelovanje u mutaciji; ⇐                   % prema Alg. 5.1
        Stvori mutanta u^{j,g};                                         % prema (5.2)
        Stvori pokusni vektor t_i^{j,g}, križanjem v^{j,g} i u^{j,g};
        % Selekcija naredne generacije
        ako f(t^{j,g}) ≤ f(v^{j,g}) onda
            v^{j,g+1} := t^{j,g};
            κ_j^{g+1} := μ_j + N_j(0,0.1); ⇐                            % prema (5.4)
        inače
            v^{j,g+1} := v^{j,g};
            κ_j^{g+1} := K + C_j(0,0.1); ⇐                              % prema (5.5)
        kraj
    kraj
    g := g+1;
dok uvjet završetka nije zadovoljen;
```

5.3.3 Detalji ugradnje

U predloženoj mutaciji DE, stalno se provodi turnirska selekcija baznog vektora pri čemu su veličine turnira za svaki pojedini vektor populacije $\mathbf{v}^j$ zadane pridruženim im vrijednostima κ_j. S obzirom da se one mijenjaju prilikom odabire nove generacije, potrebno ih je prvotno inicijalizirati, a najjednostavniji način obavljanje navedenog je $\kappa_j = \mathcal{U}_j[0,1\rangle$ za $j = 1, \ldots, NP$, gdje je $\mathcal{U}_j[0,1\rangle$ uniformna slučajna varijabla iz $[0,1\rangle$.

Nadalje, prema (5.4) i (5.5), nove vrijednosti dobivaju se na temelju Gaussovih odnosno Cauchyjevih slučajnih varijabli. Stoga se može dogoditi da te nove vrijednosti budu izvan dozvoljenih granica, odnosno $0 > \kappa_j \geq 1$. Kako bi se navedeno izbjeglo, u oba slučaja se ponavlja generiranje κ_j sve dok je vrijednost izvan granica.

Ugradnja predložene mutacija zahtijeva odgovarajuće izmjene ili preinake algoritma DE. Algoritmom 5.2 je prikazan standardni oblik s odgovarajućim proširenjima. Strelicama $\Leftarrow$ su naznačene razlike u odnosu na uobičajenu osnovnu strukturu algoritma. Kao što se može primijetiti iz prikazanog, manje izmjene se pojavljuju na nekoliko mjesta u glavnoj petlji algoritma.

5.4 Eksperimentalni rezultati i analiza: Predložena mutacija DE

S ciljem ispitivanja učinkovitosti predložene mutacije DE, odnosno da bi se dobio uvid u ponašanje algoritma u koji je ugrađena, provedena je eksperimentalna analiza. Testiranje i analiza podijeljeni su u četiri dijela. Prvi dio predstavlja usporedbu s fiksnim veličina

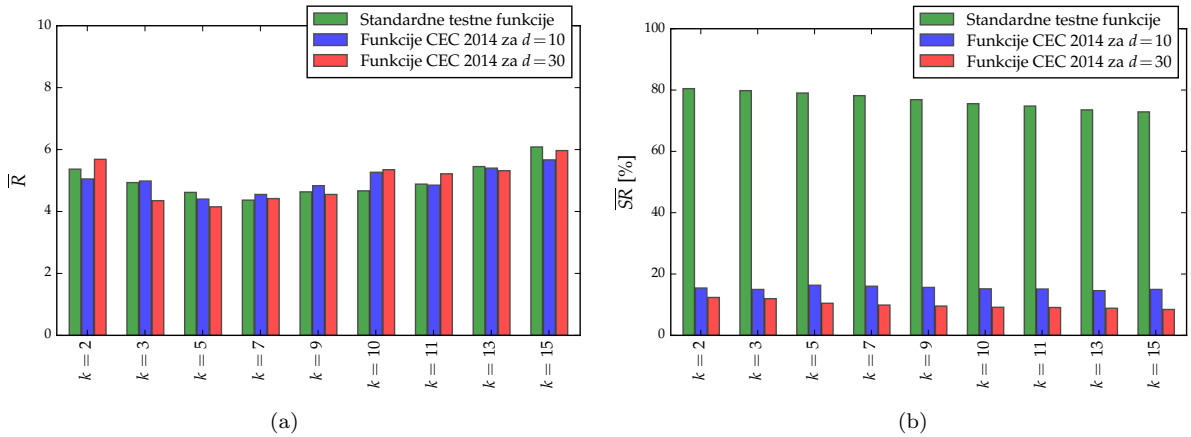
k-turnirske selekcije, odnosno usporedbu s oblikom predložene mutacije u kojem se ne vrši prilagodba veličina turnira. U drugom dijelu je provedena usporedba s nekoliko mutacija DE iz literature. Treći dio se odnosi na ispitivanje utjecaja najveće dozvoljene veličine turnira k_{max}, dok četvrti dio pruža uvid u ponašanje procesa prilagodbe veličina turnira. Kao i u poglavlju 3, cjelokupna analiza provedena je na dva skupa testnih funkcija. Podsjećanja radi, testiranje i analiza provedeni su na 30 standardnih testnih funkcija uz dimenzionalnosti zadane tablicom 3.1 te na skupu funkcija CEC 2014 koji se sastoji također od 30 funkcija, uz uporabu dimenzionalnosti $d = 10$ i $d = 30$ (sveukupno 90 instanci problema).

5.4.1 Postavke eksperimenata

Kroz cijelu analizu upotrijebljen je standardni DE uz izmjenu odnosno zamjenu mutacije rand/1. To znači da je u svim algoritmima korišteno binomno križanje. Postavke testiranja i analize odgovaraju onima opisanim u potpoglavlju 3.4.1. Parametri algoritama bili su jednaki za sve algoritme. Točnije, za veličinu populacije postavljeno je $NP = 100$, za faktor skaliranja $F = 0.5$, a za stopu križanja $CR = 0.9$. Ove vrijednosti, kao što je već bilo navedeno, predstavljaju često korištene postavke parametara.

5.4.2 Usporedba s nepromjenjivom veličinom turnira

U predloženoj mutaciji, provodi se turnirska selekcija baznog vektora. Selekcijski pritisci, odnosno veličine turnira zadane su vrijednostima pridruženim svakom vektoru populacije. Tijekom pretrage vrši se prilagodba tih vrijednosti s ciljem postizanja ravnoteže između istraživanja i iskorištavanja. Kako bi se pružio uvid u važnost, odnosno uvid u prednosti i nedostatke prilagodbe, provedeno je ispitivanje u odnosu na jednaku fiksnu veličinu turnira za svaki vektor. U tu svrhu prvo je provedeno testiranje različitih fiksnih veličina turnira k.



Slika 5.4: Učinkovitost uporabe jednake fiksne veličine turnira na standardnim i funkcijama CEC 2014 u smislu (a) prosječnih rangova $\overline{R}$ i (b) prosječnih stopa uspješnosti $\overline{SR}$.

Tablica 5.1: Rezultati na standardnim testnim funkcijama. Usporedba predložene i mutacija s različitim fiksnim selekcijskim pritiscima odabira baznog vektora.

f	DE/rand/1	DE/BoR/1	$k=5$	$k=7$	$k=10$	DE/best/1	DE/kt/1
1	▽†	▽†	▽†	△†	△†	▽†	¬
2	▽†	▽†	▽†	△†	△†	▽†	¬
3	▽†	▽†	▽†	△†	△†	▽†	¬
4	▽†	▽†	▽†	▽†	▽†	▽†	¬
5	▽†	▽†	▽∘	▽∘	▽†	▽†	¬
6	▽†	▽†	▽†	△†	△†	▽†	¬
7	▽†	▽†	▽†	△†	△†	▽†	¬
8	◯	◯	◯	◯	◯	▽†	¬
9	▽†	▽†	▽†	△†	△†	▽†	¬
10	▽†	▽†	▽†	△†	△†	▽†	¬
11	◯	◯	◯	◯	◯	▽†	¬
12	▽∘	△†	▽∘	▽†	▽†	▽†	¬
13	▽†	▽†	△∘	△∘	△∘	▽†	¬
14	▽†	△∘	◯	◯	◯	▽†	¬
15	△†	△∘	▽∘	▽∘	▽†	▽†	¬
16	▽†	▽†	▽†	▽∘	▽∘	▽†	¬
17	▽†	▽†	▽†	△∘	△†	△†	¬
18	▽†	▽†	▽∘	▽∘	▽†	▽†	¬
19	△†	△∘	△∘	▽∘	△†	▽†	¬
20	▽∘	△∘	◯	△∘	△∘	▽†	¬
21	△†	▽∘	▽∘	▽∘	▽†	▽†	¬
22	△†	◯	◯	▽∘	▽∘	▽†	¬
23	◯	◯	◯	◯	◯	◯	¬
24	◯	▽∘	◯	◯	◯	▽†	¬
25	◯	◯	◯	◯	◯	◯	¬
26	◯	◯	◯	◯	◯	▽∘	¬
27	△∘	▽∘	△∘	▽∘	▽∘	▽†	¬
28	△∘	△∘	▽∘	△∘	▽†	▽†	¬
29	◯	◯	◯	◯	▽∘	▽†	¬
30	◯	◯	◯	◯	▽∘	▽†	¬
—	14	13	10	2	7	26	▽†
—	2	3	6	8	5	1	▽∘
—	8	8	11	9	7	2	◯
—	4	1	0	7	9	1	△†
—	2	5	3	4	2	0	△∘

Pogodne fiksne veličine turnira tražene su među vrijednostima koje obuhvaćen značajan prostor oko najveće dozvoljene veličine turnira k_{max} u predloženoj mutaciji. Pretraga je provedeno nad skupom $\Omega_k = \{2, 3, 5, 7, 9, 10, 11, 13, 15\}$, a rezultati pretrage su sažeto prikazani slikom 5.4. Algoritmi s različitim veličinama turnira su međusobno rangirani, u smislu prosječne greške optimizacije, na svakoj funkciji posebno pri čemu je na danoj funkciji najmanji rang dodijeljen najboljem i tako redom. Ukoliko je bilo više jednakih rangova, odgovarajućim algoritmima dodijeljena je prosječna vrijednost rangova. Prema rezultatima prikazanim na slici 5.4, u slučaju standardnih testnih funkcija najučinkovitije su bile veličine $k = 5,\ 7$ i 10, slično u slučaju funkcija CEC 2014 za $d = 10$ s $k = 5,\ 7$ i 9, dok su za $d = 30$ najučinkovitije bile $k = 3,\ 5$ i 7. Za usporedbu s algoritmom u koji je ugrađena predložena mutacija (označeno s DE/kt/1), odabrana je fiksna veličina turnira $k = 10$ (isto kao k_{max}) te dodatno dvije koje su se pokazale u najučinkovitijima (najmanji prosječni rangovi). Kako bi se ostvario potpuniji uvid u učinkovitost DE/kt/1, u usporedbu su uključeni i standardni DE

Tablica 5.2: Rezultati na funkcijama CEC 2014 za $d = 10$. Usporedba predložene i mutacija s različitim fiksnim selekcijskim pritiscima odabira baznog vektora.

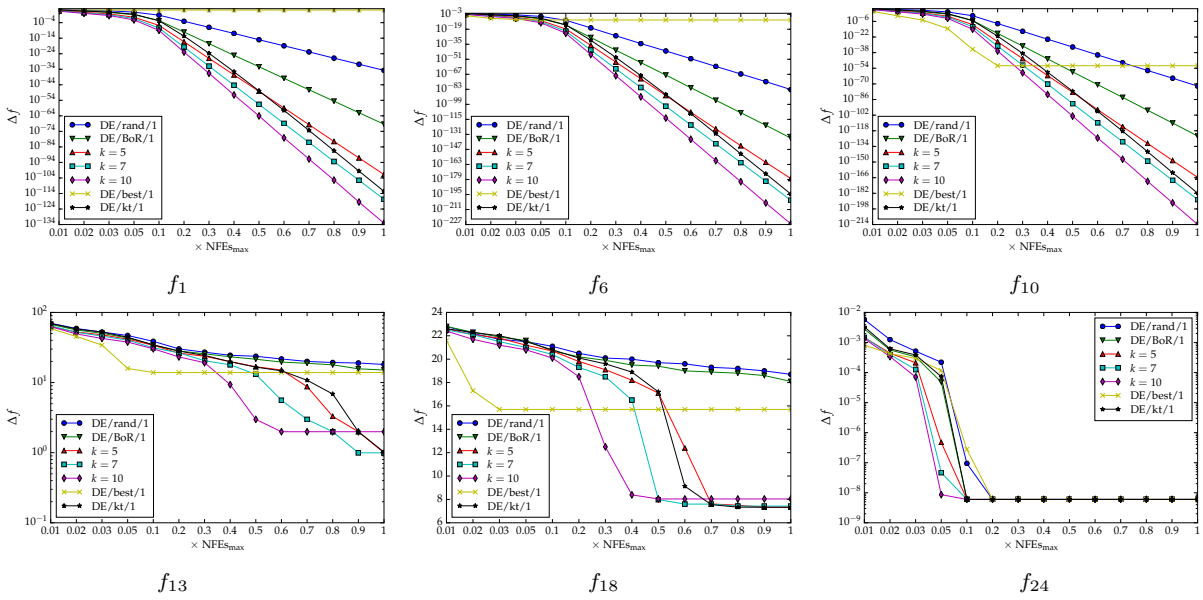
F	DE/rand/1	DE/BoR/1	$k = 5$	$k = 7$	$k = 10$	DE/best/1	DE/kt/1
1	◯	◯	◯	◯	◯	▽†	¬
2	◯	◯	◯	◯	◯	▽†	¬
3	◯	◯	◯	◯	◯	▽†	¬
4	△†	▽∘	▽∘	▽∘	▽∘	▽†	¬
5	▽†	▽†	△∘	△∘	▽∘	▽∘	¬
6	◯	◯	▽∘	▽∘	▽†	▽†	¬
7	▽†	△∘	▽∘	▽∘	△∘	▽†	¬
8	▽†	▽†	▽∘	▽∘	▽†	▽†	¬
9	▽†	▽†	▽∘	△†	△∘	▽†	¬
10	▽†	▽†	△∘	▽∘	▽∘	▽†	¬
11	▽†	▽†	▽∘	△∘	△∘	▽†	¬
12	▽†	▽†	▽∘	△∘	△†	△†	¬
13	▽†	▽†	▽∘	△†	△†	▽†	¬
14	▽†	▽†	△∘	△∘	△†	▽†	¬
15	▽†	▽†	▽∘	△∘	△†	△†	¬
16	▽†	▽†	△∘	△∘	△†	▽†	¬
17	△∘	△∘	▽∘	▽†	▽†	▽†	¬
18	△†	△†	△∘	▽∘	▽†	▽†	¬
19	▽†	▽†	▽∘	△∘	▽∘	▽†	¬
20	△†	△†	▽∘	▽†	▽∘	▽†	¬
21	△∘	△∘	▽∘	▽∘	▽∘	▽†	¬
22	△∘	△∘	△∘	▽∘	▽†	▽†	¬
23	◯	◯	◯	◯	◯	▽†	¬
24	▽†	▽†	△∘	△∘	▽∘	▽†	¬
25	△†	△†	△∘	△∘	△∘	▽†	¬
26	▽†	▽†	△∘	△∘	△∘	▽†	¬
27	▽†	▽†	▽∘	▽∘	▽†	▽†	¬
28	▽∘	△∘	▽∘	▽†	▽∘	▽†	¬
29	▽†	▽∘	▽∘	▽∘	▽†	▽†	¬
30	△∘	▽∘	▽†	▽∘	▽†	▽†	¬
—	16	14	1	3	8	27	▽†
—	1	3	16	11	8	1	▽∘
—	5	5	4	4	4	0	◯
—	4	3	0	2	5	2	△†
—	4	5	9	10	5	0	△∘

Tablica 5.3: Usporedba na funkcijama CEC 2014 za $d = 30$, Usporedba predložene i mutacija s različitim fiksnim selekcijskim pritiscima odabira baznog vektora.

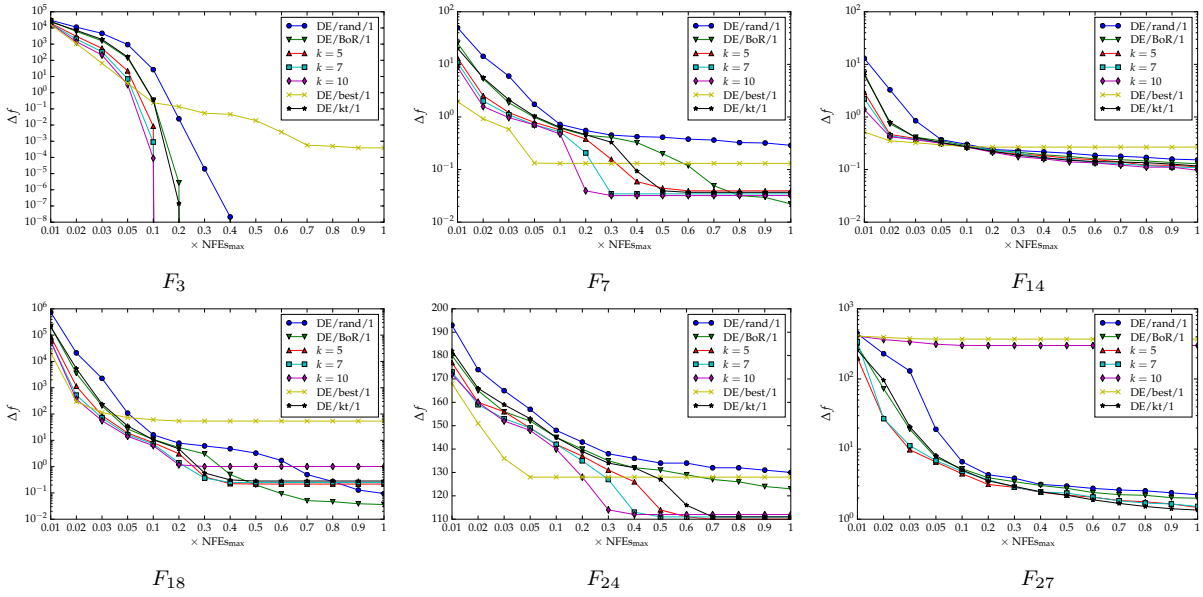
F	DE/rand/1	DE/BoR/1	$k = 3$	$k = 5$	$k = 10$	DE/best/1	DE/kt/1
1	▽†	▽†	▽†	▽∘	△∘	▽†	¬
2	◯	◯	◯	◯	◯	▽†	¬
3	◯	◯	◯	◯	◯	▽†	¬
4	△†	▽†	▽†	▽†	▽∘	▽†	¬
5	▽∘	△∘	△∘	△∘	△∘	△†	¬
6	△†	△∘	△∘	▽∘	▽†	▽†	¬
7	△†	△∘	△∘	▽∘	▽†	▽†	¬
8	▽†	▽†	△†	△†	▽†	▽†	¬
9	▽†	▽†	▽†	▽∘	△†	▽†	¬
10	▽†	▽†	▽†	△∘	▽∘	▽†	¬
11	▽†	▽†	▽†	▽∘	△†	△†	¬
12	▽∘	▽∘	△∘	▽∘	△†	△†	¬
13	▽†	▽†	▽†	△∘	△†	▽†	¬
14	▽†	▽∘	△∘	▽∘	▽∘	▽†	¬
15	▽†	▽†	▽∘	△∘	△†	▽†	¬
16	▽†	▽†	▽†	▽∘	△†	△†	¬
17	▽†	▽†	▽†	▽†	▽†	▽†	¬
18	▽†	▽†	▽†	▽∘	▽†	▽†	¬
19	▽†	▽†	▽†	▽†	▽†	▽†	¬
20	▽†	▽†	▽†	▽†	▽†	▽†	¬
21	▽†	▽∘	△∘	▽∘	▽†	▽†	¬
22	△∘	△∘	▽∘	▽∘	▽†	▽†	¬
23	◯	◯	◯	◯	◯	▽†	¬
24	△†	▽∘	▽†	▽†	▽†	▽†	¬
25	△†	△†	△∘	▽∘	▽†	▽†	¬
26	▽†	▽†	▽†	△†	▽†	▽†	¬
27	▽∘	▽†	▽†	▽∘	▽†	▽†	¬
28	△†	△∘	▽∘	▽†	▽†	▽†	¬
29	△†	▽†	▽∘	▽∘	▽†	▽†	¬
30	△†	▽∘	▽∘	▽†	▽†	▽†	¬
—	15	16	14	7	16	26	▽†
—	3	5	5	14	3	0	▽∘
—	3	3	3	3	3	0	◯
—	8	1	1	2	6	4	△†
—	1	5	7	4	2	0	△∘

(DE/rand/1), DE/best/1 te standardni algoritam s prethodno opisanom izmjenom mutacije rand/1 predloženoj u [72] (označeno s DE/BoR/1). Time su obuhvaćene i dodatne mutacije s različitim selekcijskim pritiscima pri odabiru baznog vektora.

Ostvareni rezultati usporedbe prikazani su tablicama 5.1, 5.2 i 5.3, gdje je znakom ▽ naznačeno da se radi o prosječnoj grešci optimizacije koja je veća u odnosu na postignutu s DE/kt/1, suprotno je naznačeno znakom △, a znakom ◯ je naznačeno da se radi o jednakim prosjecima. Nadalje, kako bi se utvrdilo radi li se o statistički značajnim razlikama, proveden je Wilcoxonov test ranga s predznakom uz interval pouzdanosti od 95 %. Tako znak † ukazuje da se radi o statistički značajnoj razlici u prosjecima, dok znak ∘ ukazuje na odsustvo statističkog značaja. Na dnu tablica dan je sažet prikaz rezultata u smislu navedenih pokazatelja. Promatranjem rezultata može se dobiti dojam da se fiksnom i jednakom

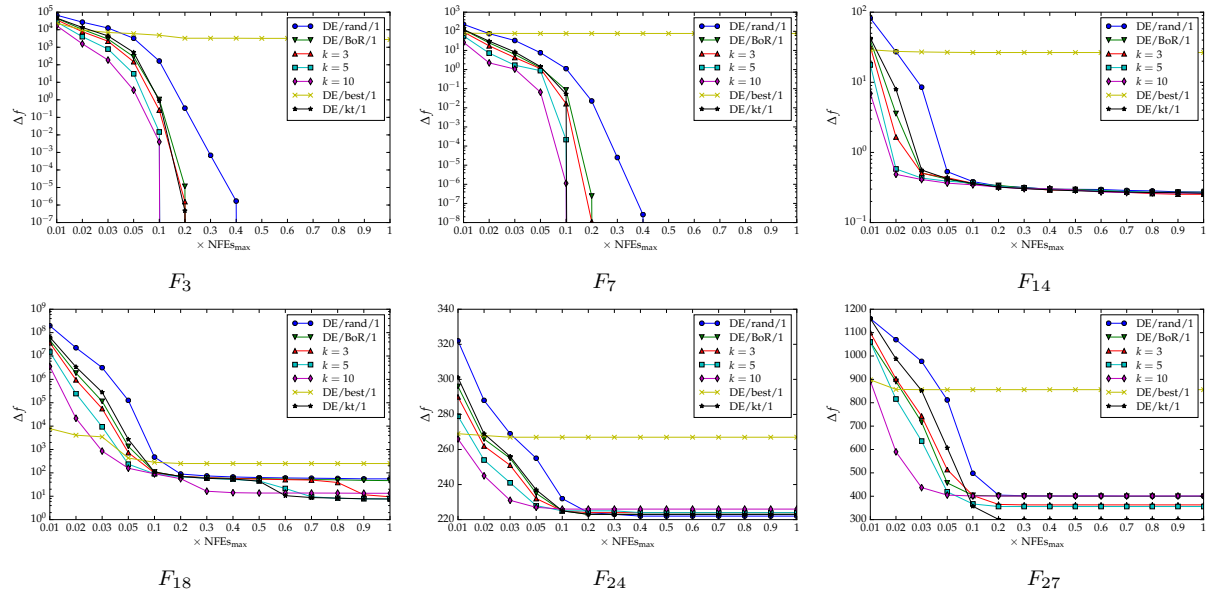


Slika 5.5: Grafovi konvergencije na nekoliko odabranih standardnih testnih funkcija. Usporedba predložene i mutacija s različitim fiksnim selekcijskim pritiscima odabira baznog vektora.



Slika 5.6: Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 10$. Usporedba predložene i mutacija s različitim fiksnim selekcijskim pritiscima odabira baznog vektora.

veličinom turnira mogu dobiti podjednako dobri ili čak u neki slučajevima bolji rezultati u odnosu na prilagodbu veličina turnira. Međutim, mora se primijetiti da najbolji rezultati u pojedinim tablicama nisu postignuti istom veličinom turnira k, te da su na funkcijama CEC 2014 za $d = 30$ ipak osjetno bolji rezultati postignuti s DE/kt/1 bez obzira na upotrijebljenu vrijednost za k. Također, kao što se može vidjeti sa slika 5.5, 5.6 i 5.7, često se radi o razlikama koje, iako mogu biti statistički značajne, nisu od praktičnog značaja te da se radi o manjim razlikama u brzini konvergencije. Nadalje, među prikazanim rezultatima, posebno



Slika 5.7: Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 30$. Usporedba predložene i mutacija s različitim fiksnim selekcijskim pritiscima odabira baznog vektora.

se ističe mutacija best/1 kao najmanje učinkovita, što može dovesti do krivog zaključka da se radi o vrlo neučinkovitom pristupu. Naime, ovo ponašanje je posljedica korištenih vrijednosti parametara (iako se relativno često koriste i pri mutaciji best/1, vidi potpoglavlje 2.7), a posebno stope križanja. Drukčijim postavkama faktora skaliranja i stope mutacije sigurno je moguće postići značajno bolje ponašanje (vidi primjerice [85]). Međutim, parametri su bili jednaki za sve, jer je cilj bio prikazati i utjecaj selekcijskog pritiska pri odabiru baznog vektora, a koji je u slučaju mutacije best/1 najveći mogući.

Na temelju prikazanih rezultata, može se zaključiti da prilagodba veličine turnira za svaki vektor posebno ima bitan učinak na ponašanje i učinkovitost predložene mutacije. Njom se može djelotvorno izbjeći skupo, u smislu vremena računanja, traženje pogodne fiksne veličine turnira. Prema dobivenim rezultatima, pogodne fiksne veličine mogu pružiti samo u određenim situacijama veću brzinu konvergencije koja se u pravilu ne ogleda u (praktički) značajnim razlikama u kvaliteti konačno pronađenih rješenja. Još jedan nedostatak uporabe jednake i fiksne veličine turnira je što različite veličine mogu biti pogodne za različite probleme, a što je također moguće riješiti prilagodbom veličina turnira.

5.4.3 Usporedba s DE/*at*best/1, DE/lbest/1 i rank-DE/rand/1

S obzirom na važnost mutacije u algoritmu DE, ne iznenađuje što je u literaturi moguće pronaći različite prijedloge izmjena postojećih oblika, prijedloge novih ili prijedloge proširenja koji utječu na odabir vektora koje će sudjelovati u mutaciji. Prema tome, provedeni su testiranje i analiza u obliku usporedbe predložene i nekoliko različitih proširenja mutacije iz literature. Za potrebe usporedbe, u standardni algoritam su ugrađeni mutacija predložena

Tablica 5.4: Rezultati na standardnim testnim funkcijama. Usporedba predložene i mutacija iz literature.

f	DE/*at*best/1		DE/lbest/1		rank-DE/rand/1		DE/kt/1	
	prosjek ± std. dev.	SR	prosjek ± std. dev.	SR	prosjek ± std. dev.	SR	prosjek ± std. dev.	SR
1	5.882E−100↓ ± 1.87E−99	100	5.950E−101↓ ± 2.11E−100	100	2.157E−68↓ ± 4.77E−68	100	**6.289E−112** ± 2.59E−111	100
2	2.368E−46↓ ± 5.01E−46	100	**1.021E−56**↑ ± 1.32E−56	100	6.075E−31↓ ± 5.87E−31	100	3.419E−54 ± 3.21E−54	100
3	1.013E−17↓ ± 3.48E−17	100	**2.017E−24**○ ± 3.97E−24	100	2.171E−13↓ ± 2.43E−13	100	2.935E−24 ± 9.36E−24	100
4	2.403E+00↓ ± 1.70E+00	0	1.490E+00↓ ± 6.62E−01	0	1.163E+00↓ ± 1.28E+00	0	**3.727E−01** ± 2.01E−01	0
5	**0.000E+00**○ ± 0.00E+00	100	1.789E−21○ ± 1.28E−20	100	3.894E−28↓ ± 1.68E−27	100	**0.000E+00** ± 0.00E+00	100
6	1.790E−166↓ ± 0.00E+00	100	1.140E−163↓ ± 0.00E+00	100	2.149E−130↓ ± 9.26E−130	100	**3.042E−189** ± 0.00E+00	100
7	6.921E−56↓ ± 1.37E−55	100	**3.582E−70**↑ ± 5.16E−70	100	4.954E−43↓ ± 7.99E−43	100	1.199E−67 ± 2.53E−67	100
8	**0.000E+00**○ ± 0.00E+00	100	1.961E−01○ ± 8.00E−01	92	**0.000E+00**○ ± 0.00E+00	100	**0.000E+00** ± 0.00E+00	100
9	2.395E−149↓ ± 1.57E−148	100	3.365E−132↓ ± 1.65E−131	100	2.874E−108↓ ± 1.56E−107	100	**5.576E−166** ± 0.00E+00	100
10	2.423E−165↓ ± 0.00E+00	100	**8.058E−186**↑ ± 0.00E+00	100	1.933E−117↓ ± 9.98E−117	100	1.138E−179 ± 0.00E+00	100
11	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00 ± 0.00E+00	100
12	4.412E+02○ ± 2.31E+02	0	2.831E+03↓ ± 5.23E+02	0	**2.694E+02**↑ ± 1.57E+02	6	4.575E+02 ± 2.53E+02	2
13	7.072E+00↓ ± 6.16E+00	8	3.137E+00↓ ± 1.38E+00	0	8.533E+00↓ ± 6.78E+00	6	**2.888E+00** ± 3.73E+00	14
14	**3.997E−15**○ ± 0.00E+00	100	1.002E−01○ ± 3.52E−01	92	**3.997E−15**○ ± 0.00E+00	100	**3.997E−15** ± 0.00E+00	100
15	2.895E−03○ ± 6.67E−03	78	4.803E−03○ ± 1.33E−02	80	1.788E−03○ ± 4.04E−03	82	**1.304E−03** ± 3.82E−03	86
16	1.313E−17↓ ± 6.90E−17	100	4.772E−03↓ ± 2.17E−02	45	4.354E−18↓ ± 3.11E−17	100	**4.354E−18** ± 3.11E−17	100
17	1.113E+01↓ ± 7.64E−01	0	**9.856E+00**○ ± 1.00E+00	0	1.128E+01↓ ± 6.26E−01	0	9.921E+00 ± 1.52E+00	0
18	9.151E+00○ ± 4.16E+00	0	9.976E+00↓ ± 1.06E+00	0	1.087E+01↓ ± 4.55E+00	0	**7.243E+00** ± 8.20E−01	0
19	**9.636E−33**○ ± 6.59E−34	100	1.220E−03↓ ± 8.71E−03	98	1.220E−03○ ± 8.71E−03	98	2.208E−32 ± 8.95E−32	100
20	9.987E−02○ ± 1.40E−09	0	**9.987E−02**○ ± 0.00E+00	0	**9.987E−02**○ ± 0.00E+00	0	9.987E−02 ± 1.40E−09	0
21	2.342E−02○ ± 1.11E−01	90	3.313E−01↓ ± 5.82E−01	57	7.318E−03○ ± 2.38E−02	90	**3.659E−03** ± 1.48E−02	94
22	4.309E−04○ ± 2.15E−03	96	3.878E−03↓ ± 2.46E−02	94	**1.350E−32**○ ± 0.00E+00	100	**2.154E−04** ± 1.54E−03	98
23	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00 ± 0.00E+00	100
24	**5.988E−09**○ ± 0.00E+00	100	8.668E−06○ ± 6.19E−05	100	**5.988E−09**○ ± 0.00E+00	100	**5.988E−09** ± 0.00E+00	100
25	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00 ± 0.00E+00	100
26	3.997E−15○ ± 0.00E+00	100	3.997E−15○ ± 0.00E+00	100	3.997E−15○ ± 0.00E+00	100	3.997E−15 ± 0.00E+00	100
27	2.805E−02○ ± 5.11E−02	76	**2.330E−04**↑ ± 1.66E−03	98	1.169E−02○ ± 3.58E−02	90	1.169E−02 ± 3.58E−02	90
28	1.981E−01○ ± 9.90E−01	96	**3.209E−07**○ ± 0.00E+00	100	**3.209E−07**○ ± 0.00E+00	100	9.907E−02 ± 7.07E−01	98
29	1.034E−01○ ± 7.39E−01	98	**0.000E+00**○ ± 0.00E+00	100	**0.000E+00**○ ± 0.00E+00	100	**0.000E+00** ± 0.00E+00	100
30	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00○ ± 0.00E+00	100	0.000E+00 ± 0.00E+00	100
—	11		11		13		↓	
—	19		15		16		○	
—	0		4		1		↑	

u [152] kao izmjena rand/1 (označeno s DE/*at*best/1), mutacija predložena u [153] uz preporučenu podjelu populacije u 10 grupa (označeno s DE/lbest/1) te proširenje predloženo u [47] povezano s mutacijom rand/1 (označeno s rank-DE/rand/1). Navedeni oblici mutacije opisani su na početku ovog poglavlja.

Tablica 5.5: Rezultati na funkcijama CEC 2014 za $d = 10$. Usporedba predložene i mutacija iz literature.

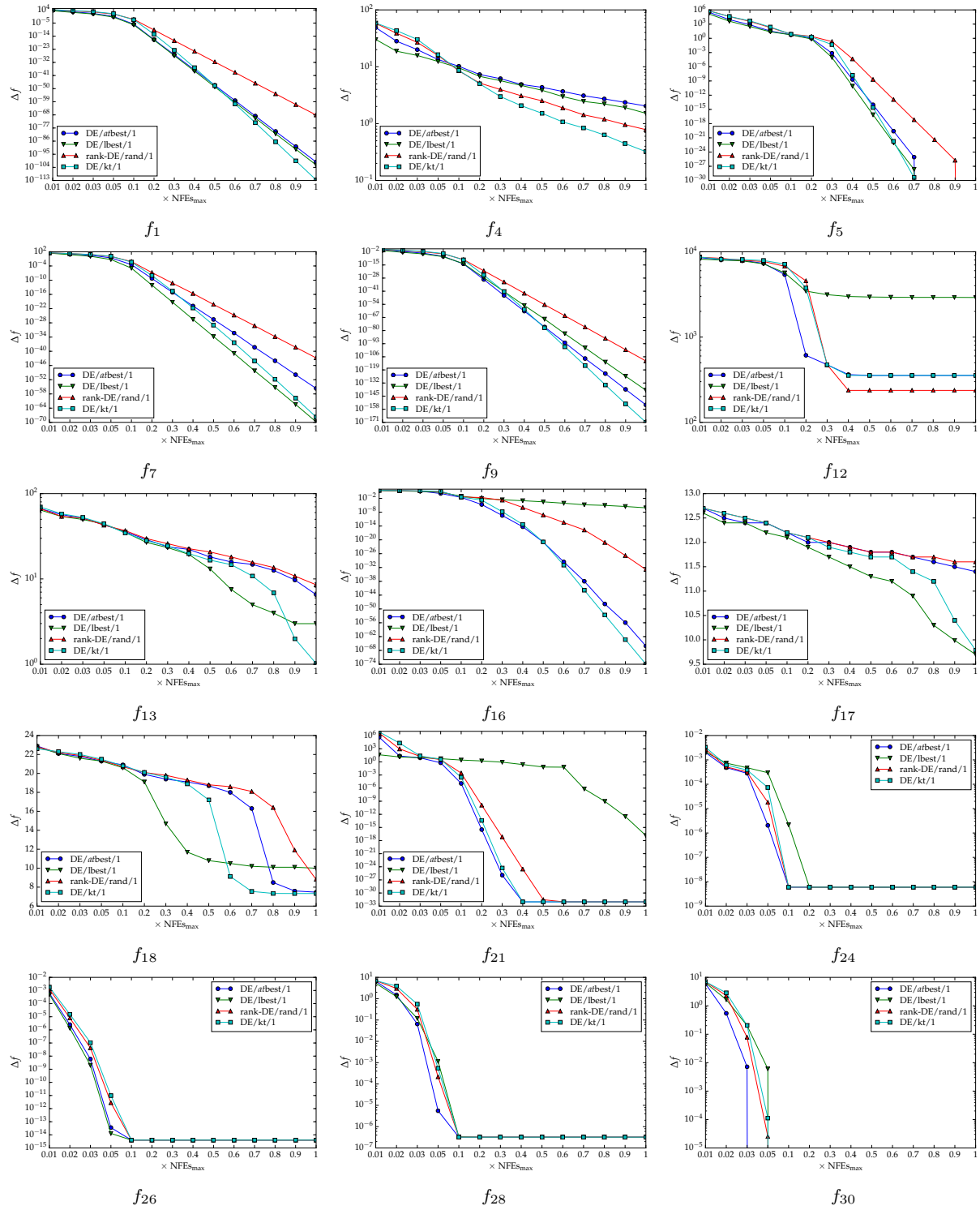
F	DE/*a*tbest/1		DE/1best/1		rank-DE/rand/1		DE/kt/1	
	prosjek ± std. dev.	SR	prosjek ± std. dev.	SR	prosjek ± std. dev.	SR	prosjek ± std. dev.	SR
1	0.000E+00∘ ± 0.00E+00	100	0.000E+00∘ ± 0.00E+00	100	0.000E+00∘ ± 0.00E+00	100	0.000E+00 ± 0.00E+00	100
2	0.000E+00∘ ± 0.00E+00	100	0.000E+00∘ ± 0.00E+00	100	0.000E+00∘ ± 0.00E+00	100	0.000E+00 ± 0.00E+00	100
3	0.000E+00∘ ± 0.00E+00	100	0.000E+00∘ ± 0.00E+00	100	0.000E+00∘ ± 0.00E+00	100	0.000E+00 ± 0.00E+00	100
4	1.901E+01∘ ± 1.70E+01	33	**1.078E+01**∘ ± 1.57E+01	33	1.449E+01∘ ± 1.65E+01	41	1.602E+01 ± 1.66E+01	33
5	**1.677E+01**∘ ± 7.57E+00	14	1.979E+01↓ ± 2.17E+00	0	1.904E+01∘ ± 4.82E+00	2	1.790E+01 ± 6.52E+00	8
6	8.770E−02↓ ± 2.69E−01	90	5.058E−01↓ ± 7.88E−01	18	**0.000E+00**∘ ± 0.00E+00	100	**0.000E+00** ± 0.00E+00	100
7	**3.311E−02**∘ ± 2.33E−02	4	3.776E−02∘ ± 2.25E−02	2	3.832E−02∘ ± 2.48E−02	6	4.030E−02 ± 3.06E−02	8
8	3.609E+00↓ ± 4.03E+00	12	3.282E+00↓ ± 1.85E+00	0	6.520E+00↓ ± 5.95E+00	2	**1.291E+00** ± 1.29E+00	29
9	1.111E+01↓ ± 8.54E+00	2	**5.547E+00**∘ ± 2.43E+00	0	1.536E+01↓ ± 7.77E+00	0	7.479E+00 ± 4.87E+00	0
10	1.002E+02↓ ± 1.52E+02	0	9.323E+01↓ ± 7.93E+01	0	2.468E+02↓ ± 1.97E+02	0	**1.151E+01** ± 2.31E+01	0
11	6.625E+02↓ ± 3.81E+02	0	5.099E+02↓ ± 3.57E+02	0	7.493E+02↓ ± 3.29E+02	0	**2.444E+02** ± 2.97E+02	0
12	6.889E−01↓ ± 1.90E−01	0	5.970E−01∘ ± 2.21E−01	0	7.060E−01↓ ± 2.24E−01	0	**5.474E−01** ± 2.43E−01	0
13	1.143E−01∘ ± 2.71E−02	0	1.552E−01↓ ± 4.65E−02	0	1.186E−01∘ ± 2.66E−02	0	**1.106E−01** ± 2.36E−02	0
14	1.218E−01∘ ± 3.48E−02	0	1.440E−01↓ ± 4.40E−02	0	1.239E−01∘ ± 3.17E−02	0	**1.144E−01** ± 2.67E−02	0
15	1.889E+00↓ ± 4.65E−01	0	**1.444E+00**∘ ± 5.23E−01	0	1.801E+00↓ ± 4.45E−01	0	1.540E+00 ± 5.64E−01	0
16	2.110E+00↓ ± 3.47E−01	0	2.645E+00↓ ± 3.10E−01	0	2.298E+00↓ ± 3.03E−01	0	**1.949E+00** ± 3.34E−01	0
17	3.773E+00∘ ± 7.24E+00	6	1.294E+01↓ ± 2.07E+01	0	**2.042E+00**∘ ± 6.04E+00	4	2.168E+00 ± 4.66E+00	8
18	4.826E−01∘ ± 5.28E−01	0	3.962E−01∘ ± 4.48E−01	0	**3.318E−01**∘ ± 4.10E−01	0	4.311E−01 ± 4.86E−01	0
19	2.045E−01↓ ± 1.07E−01	0	8.589E−01↓ ± 5.63E−01	0	2.411E−01↓ ± 1.10E−01	0	**6.130E−02** ± 5.54E−02	2
20	2.872E−01∘ ± 2.37E−01	0	4.276E−01↓ ± 1.64E−01	0	**1.459E−01**∘ ± 1.85E−01	0	1.912E−01 ± 1.79E−01	0
21	1.137E+00∘ ± 3.28E+00	0	9.847E−01↓ ± 2.35E+00	0	**4.058E−01**∘ ± 2.78E−01	0	7.286E−01 ± 2.36E+00	0
22	2.927E−01∘ ± 2.48E−01	0	4.391E+00↓ ± 6.51E+00	0	**2.070E−01**∘ ± 2.13E−01	0	6.663E−01 ± 2.79E+00	0
23	3.295E+02∘ ± 0.00E+00	0	3.295E+02∘ ± 0.00E+00	0	3.295E+02∘ ± 2.87E−13	0	3.295E+02 ± 0.00E+00	0
24	**1.099E+02**↑ ± 3.31E+00	0	1.109E+02∘ ± 2.14E+00	0	1.116E+02∘ ± 5.78E+00	0	1.112E+02 ± 3.24E+00	0
25	1.453E+02↑ ± 4.46E+01	0	**1.319E+02**↑ ± 2.34E+01	0	1.471E+02↑ ± 4.39E+01	0	1.627E+02 ± 4.23E+01	0
26	1.001E+02∘ ± 2.92E−02	0	1.002E+02↓ ± 4.47E−02	0	1.001E+02∘ ± 3.08E−02	0	**1.001E+02** ± 2.75E−02	0
27	1.265E+02↓ ± 1.74E+02	0	1.315E+02↓ ± 1.55E+02	0	8.752E+01↓ ± 1.50E+02	0	**7.743E+01** ± 1.49E+02	0
28	3.928E+02∘ ± 5.03E+01	0	3.864E+02∘ ± 2.87E+01	0	4.032E+02↓ ± 4.96E+01	0	**3.838E+02** ± 3.93E+01	0
29	2.221E+02∘ ± 5.63E−01	0	2.246E+02↓ ± 1.55E+01	0	**2.182E+02**∘ ± 1.81E+01	0	2.191E+02 ± 1.49E+01	0
30	4.792E+02↓ ± 2.60E+01	0	4.857E+02↓ ± 5.40E+01	0	4.747E+02∘ ± 2.55E+01	0	**4.709E+02** ± 2.01E+01	0
—	11		17		10		↓	
—	17		12		19		∘	
—	2		1		1		↑	

Tablice 5.4, 5.5 i 5.6 prikazuju ostvarene rezultate u smislu prosječnih grešaka optimizacije, njihovih standardnih devijacija i stopa uspješnosti. Također, kao i u prethodnim testiranjima, uz prosjeke su prikazani i rezultati provedenog Wilcoxonovog testa ranga s predznakom uz interval pouzdanosti od 95 %. Sukladno tome, znak ↓ označava da se radi o statistički značajnoj razlici u korist DE/kt/1, suprotno je naznačeno znakom ↑, dok znak ∘

Tablica 5.6: Rezultati na funkcijama CEC 2014 za $d=30$. Usporedba predložene i mutacija iz literature.

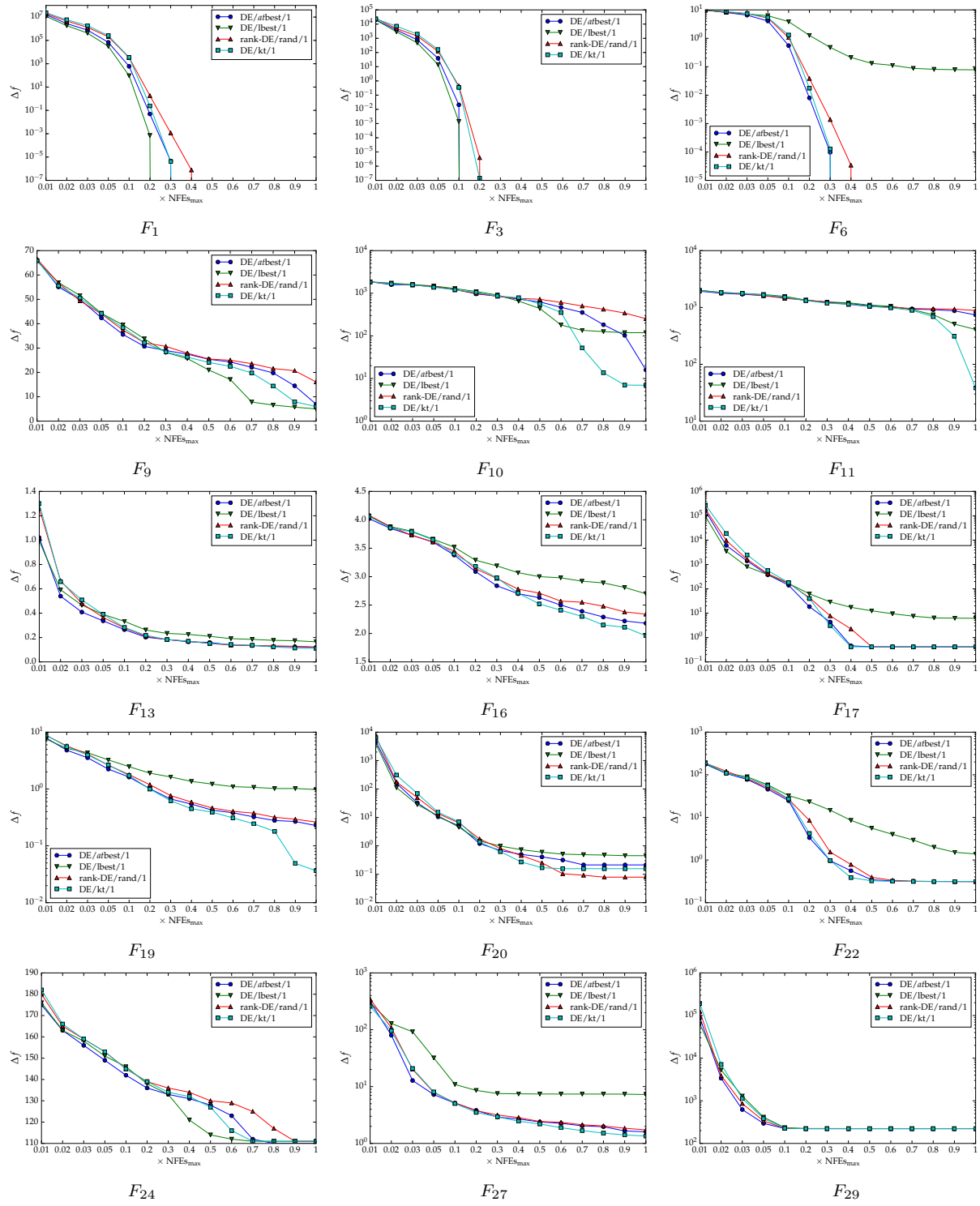
F	DE/$a$tbest/1		DE/lbest/1		rank-DE/rand/1		DE/kt/1	
	prosjek ± std. dev.	SR	prosjek ± std. dev.	SR	prosjek ± std. dev.	SR	prosjek ± std. dev.	SR
1	1.539E+04↓ ± 9.57E+03	0	9.694E+03∘ ± 6.58E+03	0	2.623E+04↓ ± 2.11E+04	0	**8.857E+03** ± 7.10E+03	0
2	0.000E+00∘ ± 0.00E+00	100	0.000E+00∘ ± 0.00E+00	100	0.000E+00∘ ± 0.00E+00	100	0.000E+00 ± 0.00E+00	100
3	0.000E+00∘ ± 0.00E+00	100	0.000E+00∘ ± 0.00E+00	100	0.000E+00∘ ± 0.00E+00	100	0.000E+00 ± 0.00E+00	100
4	2.697E+00∘ ± 1.24E+01	0	1.277E+00↓ ± 9.10E+00	84	1.267E+00↓ ± 8.87E+00	0	**3.013E−01** ± 1.63E−01	2
5	2.094E+01∘ ± 6.01E−02	0	2.094E+01∘ ± 5.45E−02	0	**2.094E+01**∘ ± 5.48E−02	0	2.095E+01 ± 4.36E−02	0
6	4.780E−01∘ ± 6.73E−01	41	1.940E+01↓ ± 5.81E+00	0	**2.949E−01**∘ ± 6.50E−01	78	3.571E−01 ± 6.61E−01	57
7	2.318E−03∘ ± 5.73E−03	82	8.317E−04∘ ± 2.62E−03	82	**5.316E−04**∘ ± 2.22E−03	94	1.160E−03 ± 3.01E−03	86
8	**1.551E+01**∘ ± 4.59E+00	0	3.570E+01↓ ± 7.61E+00	0	2.227E+01∘ ± 1.87E+01	0	1.748E+01 ± 4.44E+00	0
9	1.195E+02↓ ± 6.18E+01	0	**3.498E+01**∘ ± 9.54E+00	0	1.499E+02↓ ± 4.76E+01	0	6.820E+01 ± 6.33E+01	0
10	6.317E+02∘ ± 1.25E+03	0	2.105E+03↓ ± 3.88E+02	0	2.189E+03↓ ± 1.63E+03	0	**2.450E+02** ± 2.13E+02	0
11	6.615E+03↓ ± 3.20E+02	0	**6.071E+03**∘ ± 1.35E+03	0	6.532E+03∘ ± 2.93E+02	0	6.209E+03 ± 1.06E+03	0
12	2.379E+00∘ ± 2.74E−01	0	**2.317E+00**∘ ± 3.63E−01	0	2.409E+00∘ ± 2.79E−01	0	2.380E+00 ± 2.75E−01	0
13	2.715E−01↓ ± 3.19E−02	0	3.396E−01↓ ± 5.88E−02	0	2.968E−01↓ ± 3.68E−02	0	**2.492E−01** ± 3.86E−02	0
14	2.606E−01∘ ± 3.05E−02	0	**2.597E−01**∘ ± 3.24E−02	0	2.664E−01∘ ± 3.05E−02	0	2.605E−01 ± 2.95E−02	0
15	1.456E+01∘ ± 9.46E−01	0	**1.317E+01**∘ ± 2.84E+00	0	1.477E+01↓ ± 1.08E+00	0	1.397E+01 ± 1.74E+00	0
16	1.196E+01↓ ± 2.68E−01	0	1.231E+01↓ ± 2.75E−01	0	1.203E+01↓ ± 2.47E−01	0	**1.172E+01** ± 4.11E−01	0
17	4.043E+02↓ ± 3.24E+02	0	9.405E+02↓ ± 7.69E+02	0	5.146E+02↓ ± 4.56E+02	0	**1.722E+02** ± 1.50E+02	0
18	1.249E+01↓ ± 1.19E+01	0	1.883E+01↓ ± 9.33E+00	0	2.335E+01↓ ± 1.89E+01	0	**7.974E+00** ± 3.32E+00	0
19	3.501E+00↓ ± 1.08E+00	0	5.365E+00↓ ± 1.18E+00	0	3.442E+00↓ ± 1.13E+00	0	**2.311E+00** ± 1.09E+00	0
20	8.202E+00↓ ± 3.13E+00	0	1.921E+01↓ ± 6.27E+00	0	9.507E+00↓ ± 2.85E+00	0	**5.879E+00** ± 2.83E+00	0
21	**7.665E+01**∘ ± 9.08E+01	0	2.118E+02↓ ± 1.13E+02	0	9.867E+01∘ ± 1.69E+02	0	7.990E+01 ± 1.02E+02	0
22	6.916E+01∘ ± 6.38E+01	0	1.463E+02↓ ± 7.95E+01	0	6.485E+01∘ ± 6.29E+01	0	**5.808E+01** ± 5.75E+01	0
23	3.152E+02∘ ± 0.00E+00	0	3.152E+02∘ ± 0.00E+00	0	3.152E+02∘ ± 5.74E−14	0	3.152E+02 ± 0.00E+00	0
24	2.245E+02↓ ± 5.17E+00	0	2.275E+02↓ ± 4.41E+00	0	2.234E+02∘ ± 2.45E+00	0	**2.222E+02** ± 4.77E+00	0
25	2.030E+02∘ ± 4.53E−01	0	2.031E+02↓ ± 4.53E−01	0	**2.029E+02**∘ ± 2.90E−01	0	2.030E+02 ± 3.07E−01	0
26	1.003E+02∘ ± 4.17E−02	0	1.023E+02↓ ± 1.40E+01	0	1.003E+02↓ ± 4.23E−02	0	**1.003E+02** ± 3.84E−02	0
27	3.440E+02∘ ± 4.46E+01	0	4.076E+02↓ ± 2.89E+01	0	**3.249E+02**∘ ± 4.34E+01	0	3.367E+02 ± 4.53E+01	0
28	8.341E+02∘ ± 8.15E+01	0	1.047E+03↓ ± 1.11E+02	0	8.157E+02∘ ± 4.16E+01	0	**8.107E+02** ± 4.34E+01	0
29	7.100E+02∘ ± 9.59E+01	0	2.503E+04↓ ± 1.70E+05	0	**6.801E+02**↑ ± 1.27E+02	0	7.033E+02 ± 1.12E+02	0
30	8.887E+02↓ ± 6.83E+02	0	1.013E+03↓ ± 4.13E+02	0	7.134E+02∘ ± 3.55E+02	0	**6.587E+02** ± 2.92E+02	0
—	11		19		12		↓	
—	19		11		17		∘	
—	0		0		1		↑	

ukazuje na odsustvo statističkog značaja. Na dnu navedenih tablica dani su ukupni brojevi pojave pojedinih rezultata statističkih testova. Kao što se može primijetiti iz prikazanih rezultata, predložena mutacija je cjelokupno gledano, znatno učinkovitija u odnosu na druge korištene u usporedbi, što se najbolje ogleda u neznatnom broju slučajeva, gdje su dobiveni veći prosjeci grešaka optimizacije koji su statistički značajni. Prednosti DE/kt/1 primjetne



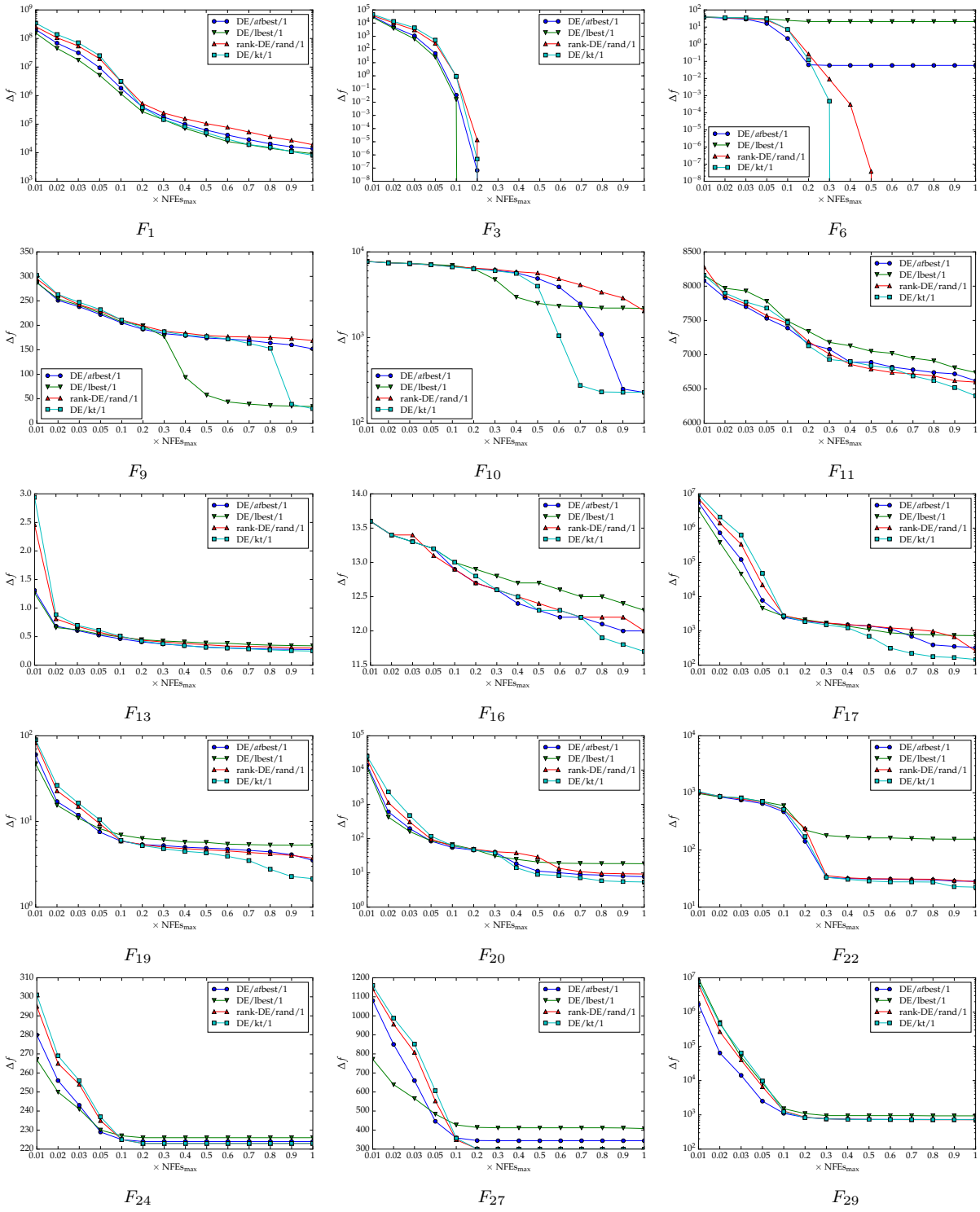
Slika 5.8: Grafovi konvergencije na nekoliko odabranih standardnih testnih funkcija. Usporedba predložene i mutacija iz literature.

su kako na standardnim testnim funkcijama, tako i na funkcijama CEC 2014 za $d = 10$ i $d = 30$. Kao što je vidljivo na slikama 5.8, 5.9 i 5.10 koje prikazuju medijane izvođenja, brzine konvergencije u ranim fazama pretrage bliske su za sve upotrijebljene algoritme, a razlike uobičajeno postaju očigledne tek kasnije. Stoga je zanimljivo primijetiti da je brzina



Slika 5.9: Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 10$. Usporedba predložene i mutacija iz literature.

konvergencije algoritma s ugrađenom predloženom mutacijom često nešto manja u odnosu na ostale na samom početku pretrage, ali i da osjetno raste vremenom. Ovo ponašanje je u skladu s povećanjem temeljne veličine za sve turnire K. Mutacija korištena u DE/*at*best/1 također uključuje prilagodbu selekcijskog pritiska odabira baznog vektora. Usporedbom se



Slika 5.10: Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 30$. Usporedba predložene i mutacija iz literature.

može zaključiti da je prilagodba u predloženoj mutaciji učinkovitija. Štoviše, iako mutacija korištena u DE/*at*best/1 može ostvariti znatno veći selekcijski pritisak, on se ne ogleda u većim brzinama konvergencije, osim u vrlo ranim fazama pretrage, kao što je prethodno opaženo.

Tablica 5.7: Procjena složenosti na standardnim testnim funkcijama predložene i mutacija iz literature.

Alg.	$d=10$				$d=30$				$d=50$			
	f_3	f_8	f_{12}	f_{17}	f_3	f_8	f_{12}	f_{17}	f_3	f_8	f_{12}	f_{17}
DE/rand/1	2.50	2.59	3.06	2.96	6.15	6.34	7.96	7.65	9.84	10.15	12.75	12.42
DE/*at*best/1	4.12	3.02	3.62	3.48	7.16	6.81	8.64	8.12	10.71	10.53	13.63	12.76
DE/lbest/1	2.42	2.46	2.97	2.86	6.09	6.14	7.86	7.53	9.66	9.84	12.78	12.22
rank-DE/rand/1	4.77	3.92	4.54	4.30	7.86	7.61	9.54	8.84	11.39	11.44	14.29	13.63
DE/kt/1	4.00	3.99	4.49	4.35	7.71	7.66	9.32	8.95	11.20	11.41	13.90	13.44

Tablica 5.8: Procjena složenosti na funkcijama CEC 2014 predložene i mutacija iz literature.

Alg.	$d=10$				$d=30$			
	F_5	F_{11}	F_{14}	F_{18}	F_5	F_{11}	F_{14}	F_{18}
DE/rand/1	2.52	3.07	2.51	2.53	6.27	6.43	6.16	6.18
DE/*at*best/1	2.98	3.63	2.99	3.30	6.86	6.95	6.77	6.81
DE/lbest/1	2.51	3.04	2.41	2.44	6.19	6.35	5.82	6.32
rank-DE/rand/1	3.83	4.55	3.88	4.02	7.56	7.81	7.55	7.76
DE/kt/1	3.73	4.43	3.93	4.08	7.83	7.97	7.58	7.63

Na koncu, s ciljem davanja uvida u vremenske složenosti, provedeni su testiranje i procjena kao u potpoglavlju 3.4.3, a dobiveni rezultati prikazani su u tablicama 5.7 i 5.8. Lako se može ustanoviti da korištene izmjene ili proširenja mutacije nemaju značajan utjecaj na vremensku složenost algoritma u koji su ugrađene. Također, s obzirom na linearni porast, niti jedna od navedenih mutacija nije ovisna o dimenzionalnosti problema. Prema prikazanim rezultatima, najveće vremenske složenosti mogu se pripisati DE/kt/1 i rank-DE/rand/1, redom. Zanimljivo je primijetiti kako je procijenjena vremenska složenost za DE/lbest/1 i DE/rand/1 približno jednaka, odnosno čak blago manja za DE/lbest/1. Ovo se može obrazložiti manjom uporabom PRNG u slučaju DE/lbest/1, gdje se svake generacije traže bazni vektori za sve grupe, te su oni tako unaprijed poznati za svaki ciljni vektor. Također, opsežna uporaba PRNG (nasumični odabir sudionika turnira te generiranje novih vrijednosti κ_j pomoću Gaussovih i Cauchyjevih slučajnih varijabli) ima osjetan utjecaj na procijenjenu vremensku složenost predložene mutacije. Mutacija u preostala dva algoritma, DE/*at*best/1 i rank-DE/rand/1, zahtijeva sortiranje cijele populacije na početku svake generacije pa se porast složenosti u odnosu na standardni algoritam (DE/rand/1) može u prvom redu tome pripisati.

5.4.4 Utjecaj najveće dozvoljene veličine turnira

Podsjećanja radi, u predloženoj mutaciji veličine turnira prilagođavaju se unutar granica $[1, k_{max}]$, gdje donja granica predstavlja odsustvo selekcijskog pritiska, dok gornja granica predstavlja najveći dozvoljeni pritisak. Prema tome, veličina ili vrijednost k_{max} može značajno utjecati na ponašanje i učinkovitost predložene mutacije. Kako bi se ispitao utjecaj navedenog, odnosno kako bi se utvrdila osjetljivost predložene mutacije na najveću dozvo-

Tablica 5.9: Prosječni rangovi i stope uspješnosti na standardnim funkcijama u ovisnosti o najvećoj dozvoljenoj veličini turnira k_{max}.

k_{max}	$f_1 \sim f_{11}$	$f_{12} \sim f_{22}$	$f_{23} \sim f_{30}$
	$\overline{R}$ ($\overline{SR}$)	$\overline{R}$ ($\overline{SR}$)	$\overline{R}$ ($\overline{SR}$)
2	7.8 (89.8)	5.1 (57.6)	5.2 (98.5)
3	7.0 (90.9)	5.0 (57.0)	4.8 (98.8)
5	6.2 (90.9)	4.5 (55.8)	**4.3** (99.5)
7	5.1 (90.9)	**4.1** (55.8)	4.8 (98.8)
9	4.8 (90.7)	4.8 (55.1)	5.2 (98.5)
10	3.7 (90.9)	5.6 (54)	5.2 (98.5)
11	3.9 (90.4)	5.7 (54.5)	4.8 (98.8)
13	3.4 (90.6)	4.6 (55.1)	5.0 (98.8)
15	**3.1** (90.7)	5.6 (54)	5.9 (98.3)

Tablica 5.10: Prosječni rangovi i stope uspješnosti na funkcijama CEC 2014 za $d = 10$ u ovisnosti o najvećoj dozvoljenoj veličini turnira k_{max}.

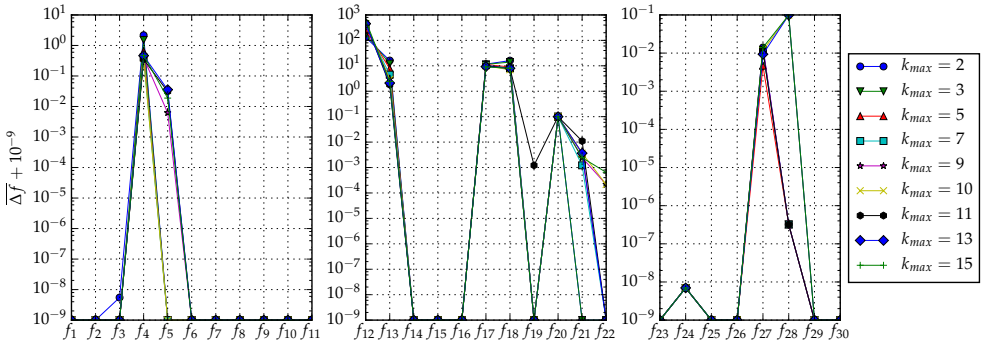
k_{max}	$F_1 \sim F_3$	$F_4 \sim F_{16}$	$F_{17} \sim F_{22}$	$F_{23} \sim F_{30}$
	$\overline{R}$ ($\overline{SR}$)	$\overline{R}$ ($\overline{SR}$)	$\overline{R}$ ($\overline{SR}$)	$\overline{R}$ ($\overline{SR}$)
2	5.0 (100)	7.9 (12.8)	**2.8** (0.7)	3.9 (0.0)
3	5.0 (100)	7.5 (12.4)	3.2 (1.3)	4.0 (0.0)
5	5.0 (100)	6.1 (12.8)	3.8 (1.0)	**3.5** (0.0)
7	5.0 (100)	5.9 (14.0)	4.8 (1.3)	5.9 (0.0)
9	5.0 (100)	4.5 (13.7)	4.5 (2.9)	5.1 (0.0)
10	5.0 (100)	4.4 (13.7)	6.2 (1.6)	4.6 (0.0)
11	5.0 (100)	4.3 (13.1)	5.0 (2.0)	5.6 (0.0)
13	5.0 (100)	**2.2** (13.6)	7.8 (1.3)	6.0 (0.0)
15	5.0 (100)	2.3 (14.6)	6.8 (1.6)	6.4 (0.0)

Tablica 5.11: Prosječni rangovi i stope uspješnosti na funkcijama CEC 2014 za $d = 30$ u ovisnosti o najvećoj dozvoljenoj veličini turnira k_{max}.

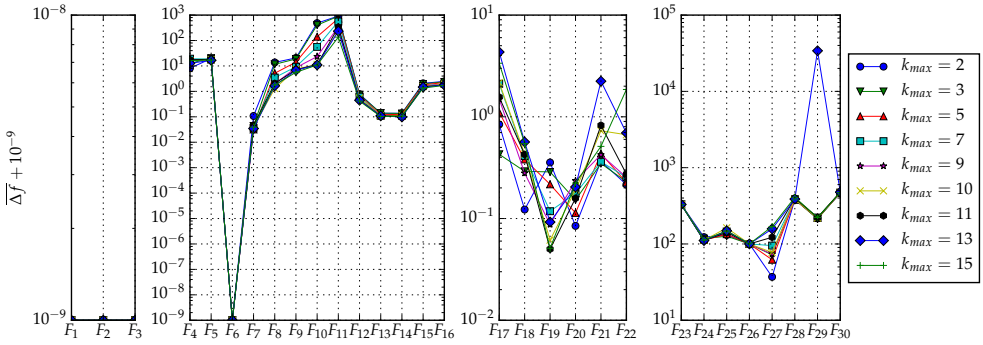
k_{max}	$F_1 \sim F_3$	$F_4 \sim F_{16}$	$F_{17} \sim F_{22}$	$F_{23} \sim F_{30}$
	$\overline{R}$ ($\overline{SR}$)	$\overline{R}$ ($\overline{SR}$)	$\overline{R}$ ($\overline{SR}$)	$\overline{R}$ ($\overline{SR}$)
2	6.3 (66.7)	7.0 (14.9)	8.0 (0.0)	4.0 (0.0)
3	6.0 (66.7)	7.0 (14.6)	6.8 (0.0)	**3.8** (0.0)
5	5.7 (66.7)	5.0 (13.9)	5.8 (0.0)	3.9 (0.0)
7	5.3 (66.7)	4.8 (13.4)	4.2 (0.0)	4.8 (0.0)
9	4.0 (66.7)	4.3 (11.0)	**3.0** (0.0)	5.6 (0.0)
10	5.0 (66.7)	4.5 (11.2)	3.2 (0.0)	5.5 (0.0)
11	4.7 (66.7)	**3.8** (12.1)	3.8 (0.0)	5.5 (0.0)
13	4.3 (66.7)	4.0 (10.0)	5.2 (0.0)	6.1 (0.0)
15	**3.7** (66.7)	4.7 (8.1)	5.0 (0.0)	5.9 (0.0)

ljenu veličinu turnira, provedena je analiza s tog gledišta. Za potrebe testiranja korištene su vrijednosti iz skupa $\Omega_{k_{max}} = \{2, 3, 5, 7, 9, 10, 11, 13, 15\}$ čime je obuhvaćena široka okolina postavljene vrijednosti u predloženoj mutaciji ($k_{max} = 10$).

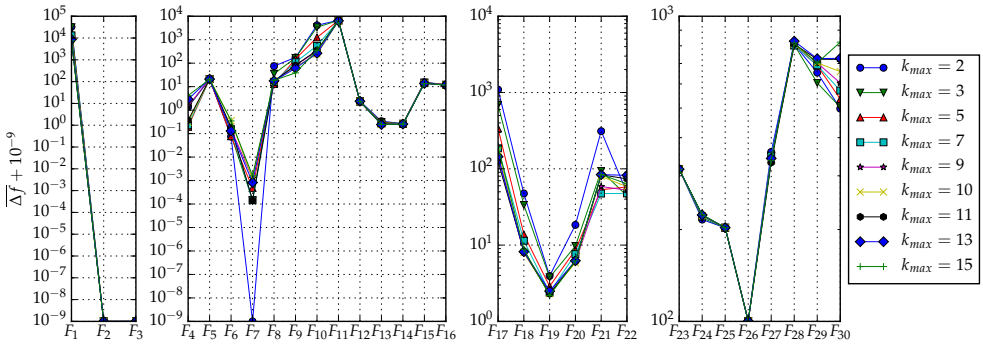
Tablice 5.9, 5.10 i 5.11 prikazuju razlike u učinkovitosti predložene mutacije u ovisnosti o najvećoj dozvoljenoj veličini turnira k_{max}. U tablicama su prikazani prosjeci postignutih rangova ($\overline{R}$) i prosjeci stopa uspješnosti ($\overline{SR}$). Međusobno rangiranje (u smislu prosječnih ostvarenih grešaka optimizacije) obavljeno je po svakoj funkciji pri čemu je najmanji rang dodijeljen najboljoj inačici mutacije i tako redom. Ukoliko je bilo više istih rangova, odgovarajućim inačicama dodijeljena je srednja vrijednost tih rangova. Kao što je vidljivo iz prikazanih rezultata, za različite razrede funkcija postignuti su najmanji prosječni rangovi (najbolji prosječni rezultati) s različitim vrijednostima za k_{max}. Može se primijetiti da su na unimodalnim funkcijama očekivano bolji rezultati ostvareni najvećim korištenim vrijednostima za k_{max}, dok su se u slučaju ostalih razreda funkcija različite vrijednosti istakle kao najpogodnije. Ipak, promatranjem slika 5.11, 5.12 i 5.13, vidljivo je da se često ne radi o velikim razlikama, a posebno između bliskih vrijednosti za k_{max}. Navedene slike prikazuju prosječne ostvarene greške optimizacije po funkcijama, gdje je zbog zornijeg prikaza na svaku dodana vrijednost 10^{-9}. Nadalje, slike 5.14, 5.15 i 5.16 prikazuju medijane izvođenja,



Slika 5.11: Prosječne greške optimizacije na standardnim testnim funkcijama za različite najveće dozvoljene veličine turnira k_{max}.

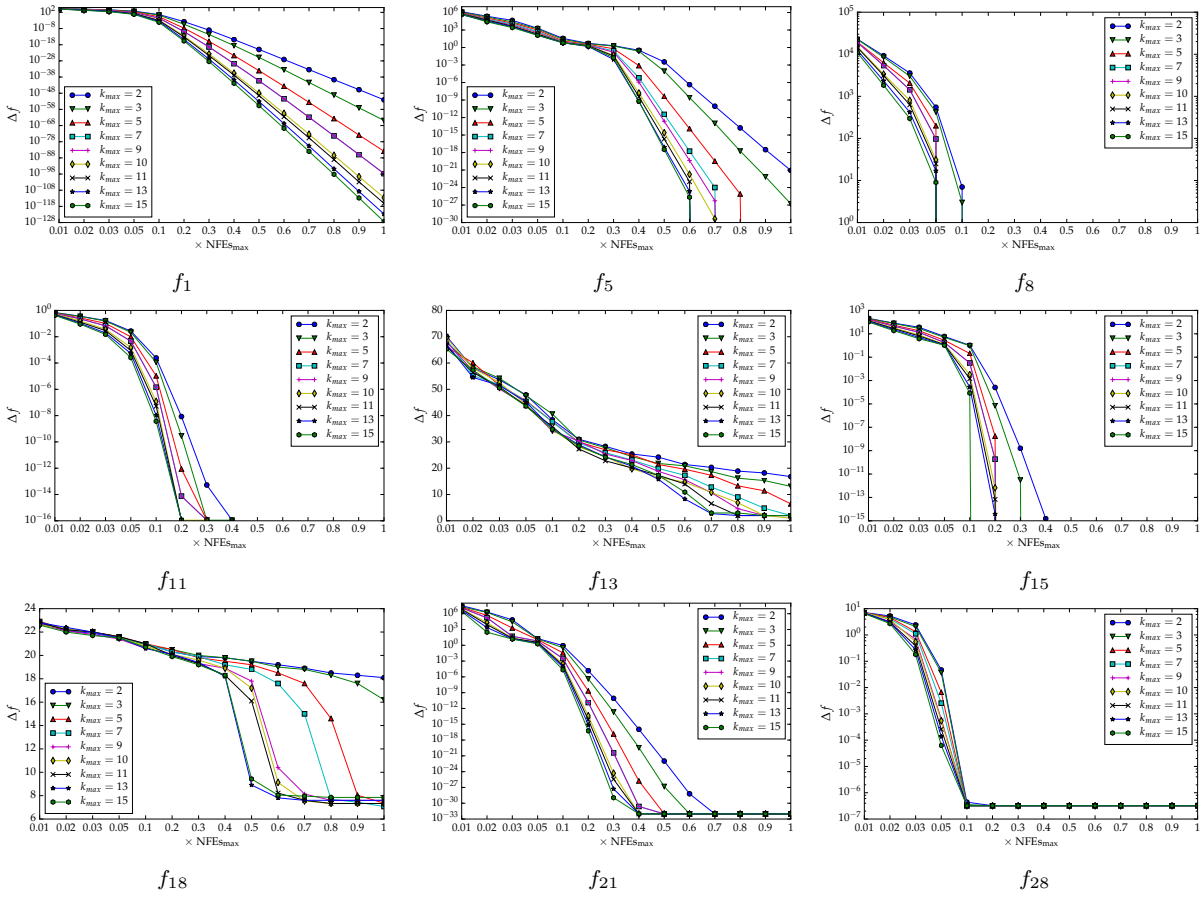


Slika 5.12: Prosječne greške optimizacije na funkcijama CEC 2014 za $d = 10$. Usporedba predložene mutacije pri različitim najvećim dozvoljenim veličinama turnira k_{max}.



Slika 5.13: Prosječne greške optimizacije na funkcijama CEC 2014 za $d = 30$. Usporedba predložene mutacije pri različitim najvećim dozvoljenim veličinama turnira k_{max}.

a pružaju uvid u ponašanje u smislu brzine konvergencije predložene mutacije s različitim najvećim dozvoljenim veličinama turnira. Kao što se može pretpostaviti, a što i slike sugeriraju, najveća dozvoljena veličina turnira izravno je proporcionalna brzini konvergencije. Osim toga, slike pružaju detaljniji uvid u razlike konačno postignutih grešaka optimizacije. U tom pogledu, jasno su vidljive razlike između manjih i većih vrijednosti za k_{max}, gdje je često veća brzina konvergencije rezultirala i manjom konačnom greškom optimizacije. Ovo se posebno odnosi na dvije najmanje korištene vrijednosti ($k_{max} = 2$ i 3).

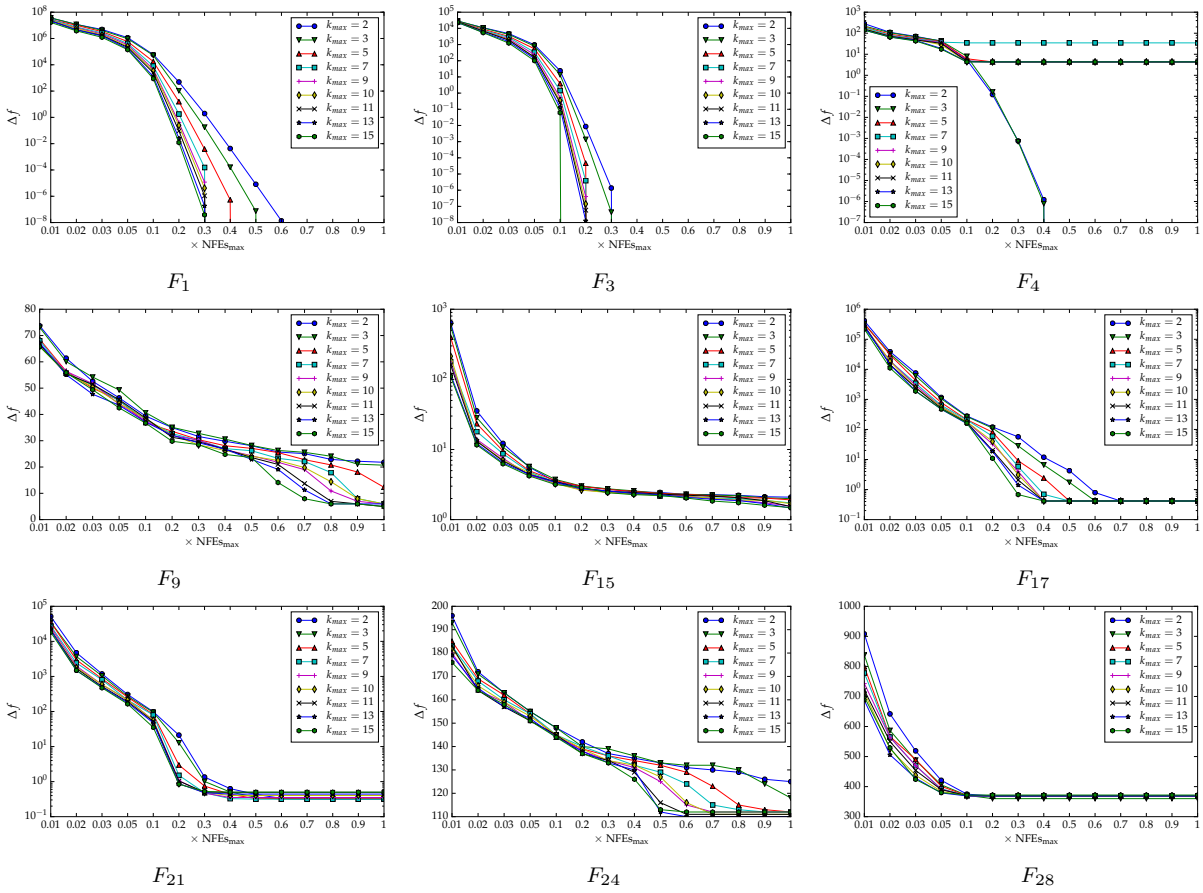


Slika 5.14: Grafovi konvergencije na nekoliko odabranih standardnih testnih funkcija. Usporedba predložene mutacije pri različitim najvećim dozvoljenim veličinama turnira k_{max}.

Sukladno očekivanju, ali i sudeći po dobivenim rezultatima, male vrijednosti za k_{max} često ne mogu pružiti dovoljnu brzinu konvergencije. Treba napomenuti da je $k_{max} \in \{2, 3\}$ razmatran isključivo zbog cjelovitosti analize. Prema tome, od posebnog značaja je što prikazani rezultati sugeriraju da vrijednosti u bližoj okolini u odnosu na vrijednost postavljenu u predloženoj mutaciji ($k_{max} = 10$) pružaju međusobno podjednaku učinkovitost. Blage razlike ogledaju se samo u brzinama konvergencije. Na temelju toga, može se doći do zaključka da predložena mutacija nije osjetljiva na (manje) varijacije u najvećoj dozvoljenoj veličini turnira.

5.4.5 Ponašanje tijekom procesa optimizacije

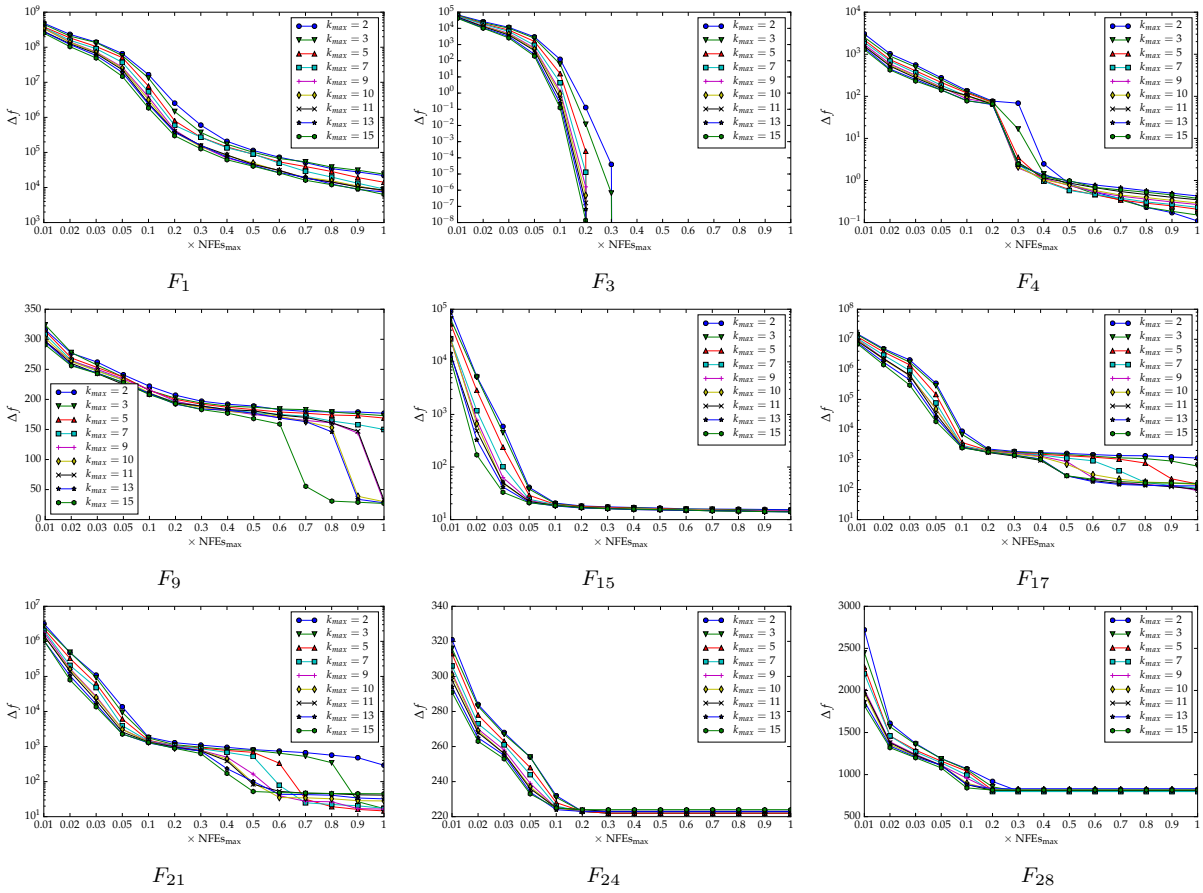
Kao što je već ranije istaknuto, odabir baznog vektora u predloženoj mutaciji obavlja se turnirskom selekcijom čija je veličina za svaki član populacije zadana pridruženom vrijednosti $\kappa_j \in [0, 1\rangle$ koja se prilagođava tijekom pretrage. S ciljem pružanja uvida u proces navedene prilagodbe, bilježeni su odgovarajući podaci tijekom jednog izvođenja. Tako su primjerice bilježene vrijednosti κ_j koje su rezultirale unaprjeđenjem do tada najboljeg pronađenog rješenja, jer s obzirom na izmjene, u danom trenutku u populaciji može postajati cijeli niz takvih



Slika 5.15: Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 10$. Usporedba predložene mutacije pri različitim najvećim dozvoljenim veličinama turnira k_{max}.

vrijednosti. Osim toga, za uspješne (potomak prešao u narednu generaciju) vrijednosti bilježene su prosječna, najmanja i najveća te isto to za neuspješne, a njihov kumulativni broj je također praćen. Slikama 5.17 i 5.18 prikazani su navedeni podaci kroz generacije za nekoliko odabranih funkcija.

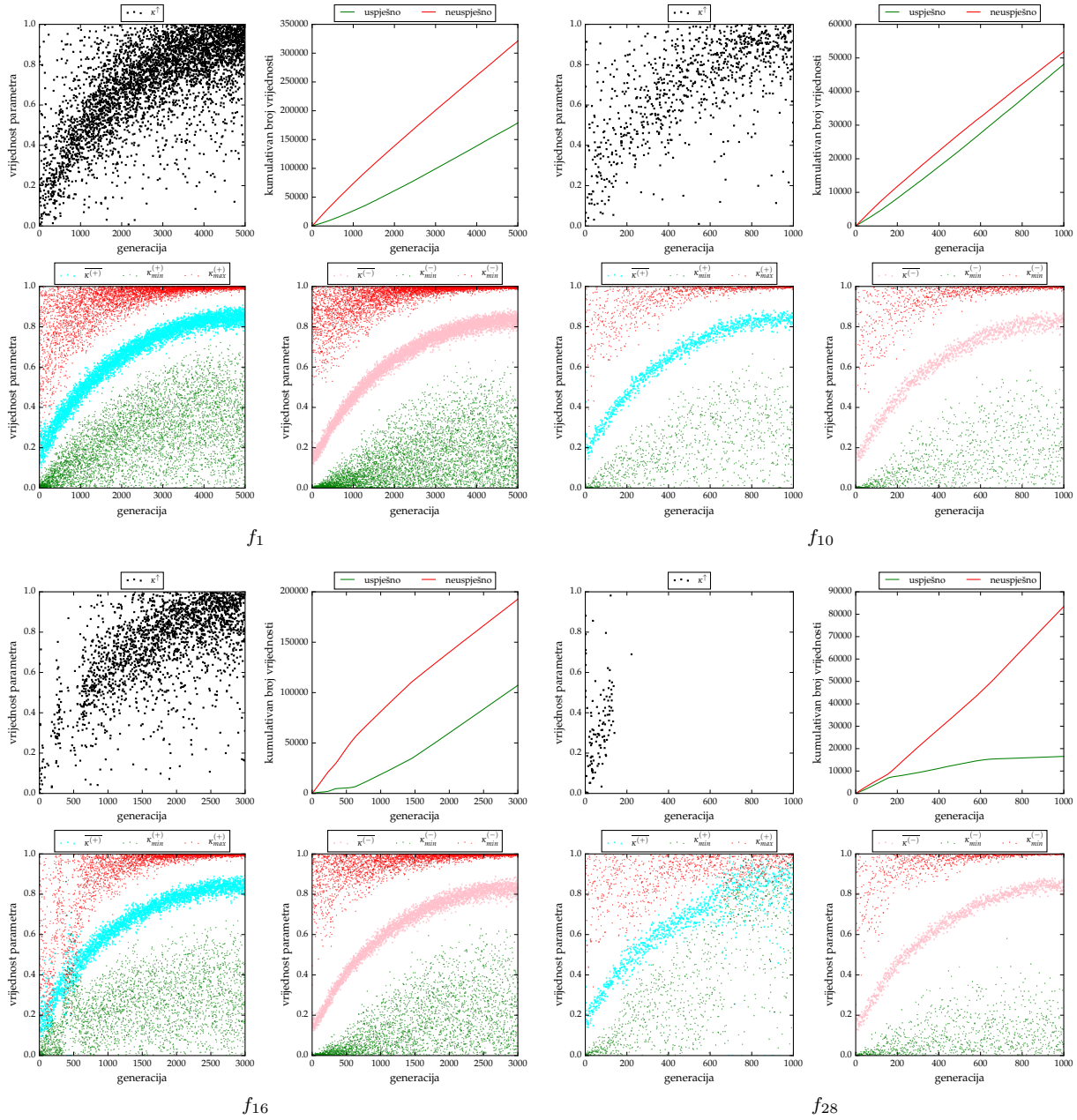
Iz prikazanog se može primijetiti kako vrijednosti koje su rezultirale unapređenjem do tada najboljeg rješenja (označeno s $\kappa^{(\uparrow)}$) vrlo dobro prate temeljnu vrijednost za sve turnire K (vidi Sl. 5.3). Vidljivo je da u ranim fazama većinom manje vrijednosti (< 0.5) doprinose unaprjeđenjima, dok veće vrijednosti to čine poslije. Uz to, promatranjem prosjeka uspješnih i neuspješnih (označeno s $\overline{\kappa^{(+)}}$ i $\overline{\kappa^{(-)}}$, redom) te promatranjem najmanjih i najvećih vrijednosti (označeno s $\kappa^{(+)}_{min}$ i $\kappa^{(+)}_{max}$ te $\kappa^{(-)}_{min}$ i $\kappa^{(-)}_{max}$) vidi se da raspon vrijednosti generiranih kroz proces pretrage uglavnom pokriva cijeli interval $[0, 1\rangle$. Također, može se primijetiti da navedeno nije ovisno o broju uspješnih i neuspješnih vrijednosti, ali su veće ili manje oscilacije prosjeka vidljive ovisno o njima. Prosjeci se kroz generacije ponašaju slično kao vrijednost temeljne veličine K, iako su na početku nešto veći, a pred kraj nešto manji, što je u skladu s načinom generiranja novih vrijednosti κ_j.



Slika 5.16: Grafovi konvergencije na nekoliko odabranih funkcija CEC 2014 za $d = 30$. Usporedba predložene mutacije pri različitim najvećim dozvoljenim veličinama turnira k_{max}.

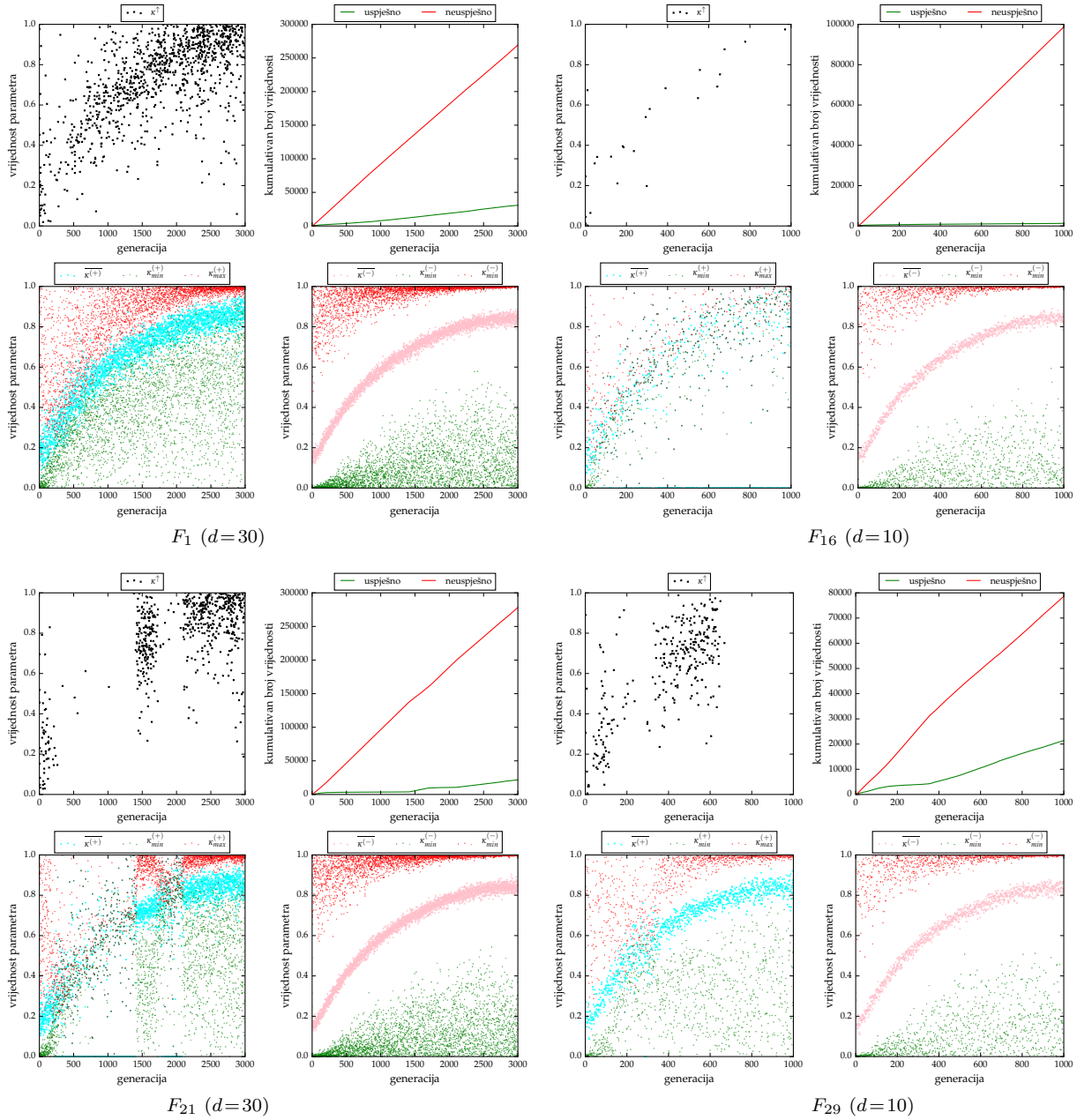
5.5 Osvrt na predloženu mutaciju

Mutacija u diferencijalnoj evoluciji nedvojbeno predstavlja primarni varijacijski operator. Njome se provodi perturbacija baznog vektora skaliranom razlikom drugih vektora populacije. Odabir baznog vektor ima stoga veliki utjecaj na istraživanje neotkrivenih dijelova i iskorištavanje obećavajućih točaka prostora pretrage. Rezultati provedene eksperimentalne analize sugeriraju kako uvođenje selekcijskog pritiska pomoću k-turnira u odabir baznog vektora najčešće pospješuje učinkovitost u smislu kvalitete pronađenih rješenja te očekivano brzine konvergencije. U tom pogledu, predložena mutacija obavlja prilagodbu selekcijskog pritiska preko podešavanja veličina k-turnira, što se pokazalo važnim, jer se time izbjegava traženje pogodnih fiksnih veličina. Štoviše, prilagodbi ide u prilog i mogućnost prijelaza s, primarno, istraživanja na iskorištavanje. U konačnici, dobiveni rezultati ukazuju na visoku učinkovitost predložene mutacije, s obzirom da su bolji rezultati postignuti ne samo u usporedbi s uobičajenom mutacijom, nego i općenito u usporedbi s nekoliko upotrijebljenih mutacija iz literature koje na drukčije načine uvode selekcijski pritisak u odabir baznog vektora.



Slika 5.17: Proces prilagodbe veličine turnira na standardnim testnim funkcijama.

Kao nedostatak predložene mutacije, može se istaći najveća dozvoljena veličina turnira k_{max}. Međutim, ona je unaprijed zadana i fiksna, ali i provedena eksperimentalna analiza sugerira da je predložena mutacija otporna na manje varijacije vrijednosti k_{max} u odnosu na postavljenu ($k_{max} = 10$). Razlike se ogledaju samo u blagim razlikama u brzini konvergencije. Veće promjene vrijednosti k_{max} mogu eventualno biti korisne u slučaju znatno manjih ili znatno većih populacija. Sukladno tome, lako se može zaključiti kako pri nezadovoljavajućoj učinkovitosti treba načiniti promjenu proporcionalnu korištenoj veličini populacije.



F_1 $(d=30)$ F_{16} $(d=10)$

F_{21} $(d=30)$ F_{29} $(d=10)$

Slika 5.18: Proces prilagodbe veličine turnira na funkcijama CEC 2014.

Mutacija DE zasnovana na prilagodljivoj k-turnirskoj selekciji za određivanje baznog vektora opisana u ovom poglavlju uz provedeno eksperimentalno testiranje i analizu te dobre postignute rezultate koji jasno ukazuju na njenu učinkovitost predstavlja ispunjenje prijedloga trećeg izvornog znanstvenog doprinosa ove disertacije.

POGLAVLJE 6

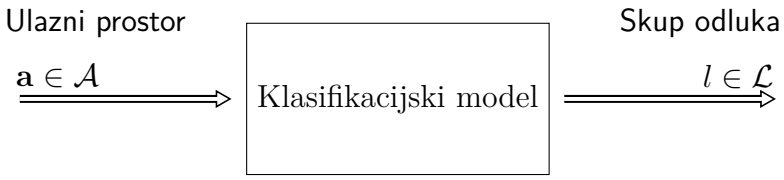
Izgradnja radijalnih mreža diferencijalnom evolucijom

RADIJALNE mreže su vrsta umjetnih neuronskih mreža koje imaju samo jedan skriveni sloj, a njega čine radijalne funkcije. Jedna od brojnih primjena takvih mreže je i klasifikacija. Izgradnja radijalnih mreža kao klasifikacijskih modela zahtijeva pronalazak velikog broja parametara koji ih opisuju. Lako se može zaključiti da se u pravilu radi o računalno zahtjevnom procesu pa je provođenje velikog broja vrednovanja često nepraktično. Stoga je poželjan pronalazak dobrih rješenja, odnosno mreža zadovoljavajućih svojstava, kroz relativno mali broj vrednovanja. U ovom poglavlju, kao prijedlog četvrtog izvornog znanstvenog doprinosa, razmatraju se učinkovitost i ponašanje standardnog algoritma DE pri izgradnji takvih mreža. Uz to, ispitane su i razmatrane prednosti i nedostaci koje pružaju unaprjeđenja DE predložena u prethodnim poglavljima.

6.1 Radijalne mreže

Umjetne neuronske mreže (engl. *artificial neural networks*, ANNs), jednostavno rečeno, inspirirane su biološkim živčanim sustavom, prvenstveno mozgom, te predstavljaju posebnu paradigmu računalne inteligencije. Navedena paradigma obuhvaća mnoge različite vrste mreža (vidi primjerice [13, 32, 36, 131]). Isto tako, pronašle su raznolike primjene, kao što je klasifikacija i regresija za različite potrebe [62, 100, 151], aproksimacija funkcija [42], upravljanje procesima [69], izgradnja surogat modela u algoritmima optimizacije [114] i brojne druge. Jednu često korištenu vrstu mreže predstavljaju radijalne mreže (engl. *radial basis function networks*, RBFNs) [18, 87] koje su karakteristične po svom skrivenom sloju.

Klasifikacija kao oblik nadziranog učenja (engl. *supervised learning*) predstavlja značajan problem strojnog učenja (engl. *machine learing*) sa širokim rasponom primjene. U tom



Slika 6.1: Skica klasifikacijskog modela.

pogledu, ANNs su uobičajen pristup njegovom rješavanju. S obzirom da je u ovom poglavlju naglasak na izgradnji klasifikacijskih modela predstavljenih RBFNs, u nastavku je prvo sažeto opisan problem klasifikacije.

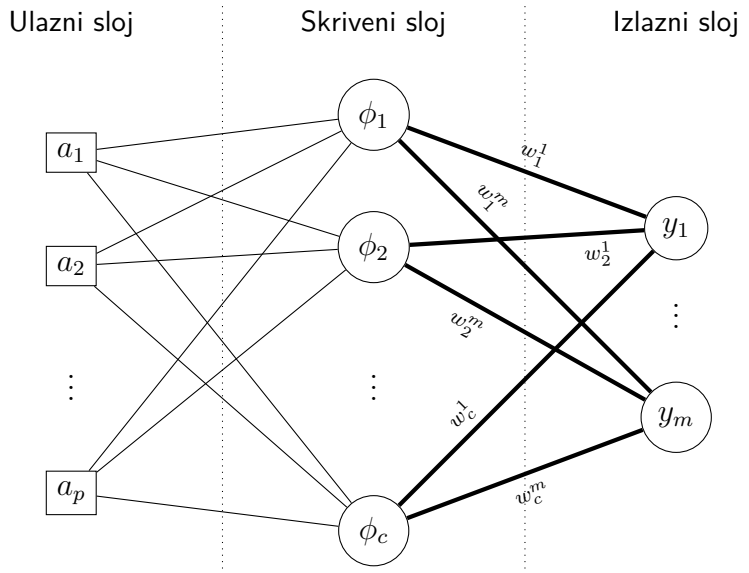
6.1.1 Kratki uvod u problem klasifikacije

Za dani skup oznaka ili klasa $\mathcal{L} = \{l_i \colon i = 1, \ldots, m\}$ i nepoznati uzorak ili podatak opisan vektorom značajki $\mathbf{a} = (a_1, \ldots, a_p) \in \mathcal{A} \subset \mathbb{R}^p$, klasifikacija zahtijeva određivanje klase $l \in \mathcal{L}$ kojoj pripada $\mathbf{a}$. U tom smislu, klasifikator ili klasifikacijski model predstavlja preslikavanje $f \colon \mathcal{A} \to \mathcal{L}$, odnosno postupak koji danom ulaznom vektoru dodjeljuju jednu od dostupnih oznaka. Navedeno je dodatno ilustrirano slikom 6.1. Nadalje, klasifikacija podrazumijeva postojanje skupa označenih uzoraka $\mathcal{Q} = \{(\mathbf{a}^j, l_{q(j)}) \colon \mathbf{a}^j \in \mathcal{A},\ l_{q(j)} \in \mathcal{L},\ j = 1, \ldots, n\}$, kojim se vodi izgradnja klasifikacijskog modela. Općenito, izgradnja klasifikacijskog modela podrazumijeva traženje i postavljanje parametara za odabranu metodu ili pristup klasifikacije (model). Traženje dobrih vrijednosti parametara modela koji će rezultirati što preciznijim preslikavanjem vođeno je skupom označenih uzoraka $\mathcal{Q}$. Isto tako, ukoliko postoji odgovarajuće znanje o problemu, dani skup može pomoći pri odabiru pristupa.

Strojno učenje obuhvaća mnoštvo metoda ili pristupa kao primjerice stabla odluke, algoritam k-najbližih susjeda, strojeve s potpornim vektorima i ANNs. Kao što je slučaj s algoritmima optimizacije, ne postoji univerzalno najbolji pristup. Učinkovitost, u smislu preciznosti klasifikacije ili slične mjere, prvenstveno ovisi o skupu označenih uzoraka i uzorcima koje se klasificira, ali ovisi i o parametrima modela. Prema tome, dok se teško može utjecati na uzroke, situacija je drukčija u pogledu traženja parametara modela. Tako su za mnoge pristupe obično na raspolaganju raznoliki algoritmi za traženje pogodnih vrijednosti parametara, a u koje se redovito ubrajaju EAs i slični algoritmi optimizacije.

6.1.2 Struktura radijalnih mreža

S gledišta ANNs, radijalne mreže su svojstvene po uporabi radijalnih funkcija kao čvorova u skrivenom sloju. Struktura uobičajene RBFN prikazana je slikom 6.2, a može se opisati brojem i pripadnim parametrima čvorova u skrivenom sloju te težinama veza čvorova skrivenog i izlaznog sloja. Broj čvorova u ulaznom sloju odgovara broju značajki kojima je opisan svaki uzorak $\mathbf{a} = (a_1, \ldots, a_p) \in \mathcal{A}$, broj čvorova izlaznog sloja odgovara broju oznaka $m = |\mathcal{L}|$,



Slika 6.2: Struktura uobičajene radijalne mreže.

broj čvorova u skrivenom sloju c je zadan ili, češće, podložan je traženju tijekom izgradnje mreže (klasifikacijskog modela). Veze između čvorova ulaznog i skrivenog sloja su jedinične, što općenito nije slučaj s vezama $w_i^j \in \mathbb{R}$, $i = 1, \ldots, c$, $j = 1, \ldots, m$ između čvorova skrivenog i izlaznog sloja. Svaki čvor skrivenog sloja ugrađuje radijalnu funkciju ϕ_i, $i = 1, \ldots, c$. Kod radijalnih funkcija vrijednost ovisi samo o udaljenosti ulaza od ishodišta ili neke centralne točke, odnosno $\phi(\mathbf{x}, \mathbf{z}) = \phi(\|\mathbf{x} - \mathbf{z}\|)$, $\mathbf{x}, \mathbf{z} \in \mathbb{R}^p$, gdje je $\|\cdot\|$ norma [uobičajeno ℓ_2 (Euklidska) norma]. Prema svemu navedenom, izlaz RBFN može se definirati kao linearna kombinacija

$$y_j = \sum_{i=1}^{c} w_i^j \cdot \phi_i(\|\mathbf{a} - \mathbf{z}^i\|), \quad j = 1, \ldots, m \ . \tag{6.1}$$

Predložena je uporaba nekoliko vrsta radijalnih funkcija u čvorovima skrivenog sloja RBNFs (vidi primjerice [13, 36]) kao što su višekvadratna i njena obrnuta inačica, kubna, logistička. Ipak, uvjerljivo najčešće je korištena Gaussova funkcija [13, 62, 151]

$$\phi(\|\mathbf{a} - \mathbf{z}\|) = e^{-\frac{\|\mathbf{a}-\mathbf{z}\|^2}{2\sigma^2}}, \tag{6.2}$$

gdje parametar $\mathbf{z} = (z_1, \ldots, z_p) \in \mathbb{R}^p$ predstavlja centar funkcije, a parametar $\sigma > 0$ predstavlja pripadajuću širinu. Ona je lokalizirana radijalna funkcija, što znači da joj vrijednosti opada (brže nego eksponencijalno [87]) s udaljenosti ulaza $\mathbf{a}$ od centra $\mathbf{z}$. Prema tome, uz Gaussove radijalne funkcije kao čvorove skrivenog sloja, prethodno definirani izlaz (6.1), može se napisati kao

$$y_j = \sum_{i=1}^{c} w_i^j \cdot e^{-\frac{\|\mathbf{a}-\mathbf{z}^i\|^2}{2\sigma_i^2}}, \quad j = 1, \ldots, m \ . \tag{6.3}$$

Treba primijetiti kako centri $\mathbf{z}^1, \ldots, \mathbf{z}^c$ određuju lokacije funkcija, dok pripadajuće širine

$\sigma_1, \ldots, \sigma_c$ utječu na pokrivenost ulaznog prostora istima. Ti parametri čine parametre čvorova skrivenog sloja, a zajedno s težinama w_i^j, $i = 1, \ldots, c$, $j = 1, \ldots, m$, potpuno opisuju navedenu RBFN.

6.1.3 Treniranje RBFNs i izgradnja klasifikacijskih modela

Izgradnja RBFNs kao klasifikacijskih modela zahtijeva pronalazak niza parametara, a proces traženja istih uobičajeno se naziva učenjem ili treniranjem. Kao što je navedeno, RBFNs opisuju parametri radijalnih funkcija kao čvorova skrivenog sloja te težine veza između čvorova skrivenog i izlaznog sloja. U slučaju Gaussovih radijalnih funkcija (i nekih drugih, kao što je višekvadratna), parametre čvorova skrivenog sloja čine centri i pripadajuće širine. Broj čvorova u skrivenom sloju rijetko je unaprijed poznat pa ga je nužno odrediti na neki način. I dok veći broj čvorova može rezultirati preciznijim modelima, on svakako rezultira većim računalnim zahtjevima za treniranje. Stoga je cilj izgraditi što precizniji model, ali bez suvišnih čvorova u skrivenom sloju.

Treba istaknuti kako je za potrebe izgradnje klasifikacijskih modela umjesto (6.1) pogodnije koristi normalizirane izlaze čvorova skrivenog sloja [19, 137]. Ovo su prvo predložili Moody i Darken [87, 88] pa je prema tome izlaz mreže

$$y_j = \frac{\sum_{i=1}^{c} w_i^j \cdot \phi_i(\|\mathbf{a} - \mathbf{z}^i\|)}{\sum_{i=1}^{c} \phi_i(\|\mathbf{a} - \mathbf{z}^i\|)}, \quad j = 1, \ldots, m, \tag{6.4}$$

gdje svaki čvor u skrivenom sloju morati imati podatke izlaza svih ostalih čvorova tog sloja (implicira postojanje bočnih veza). Prema Bugmann [19], ovakve mreže sposobne su dati značajne vrijednosti na izlazu i za uzorke koji su daleko od centara radijalnih funkcija. Prema tome, imaju i bolja svojstva generalizacije [137] te omogućuju izgradnju manjih mreža u odnosu na standardni oblik. Navedeno se pokazalo i u preliminarnom testiranju i analizi.

Treniranje se obavlja prema skupu označenih uzoraka $\{(\mathbf{a}^1, l_{q(1)}), \ldots, (\mathbf{a}^n, l_{q(n)})\}$. Cilj treniranja je pronaći parametre mreže takve da vrijednosti na izlaznim čvorovima budu što bliže (u nekom smislu) željenim vrijednostima. Svaki izlaz $y_1, \ldots, y_m$ predstavlja jednu od dostupnih oznaka $l_1, \ldots, l_m$. U pogledu klasifikacije, željena vrijednost svakog izlaznog čvora jedna je od dvije moguće vrijednosti, primjerice $o_j \in \{0, 1\}$, $j = 1, \ldots, m$. Shodno tome, za svaki označeni uzorak mogu se jednostavno definirati željeni izlazi, tako da se za o_j koji odgovara oznaci uzorka postavi vrijednost 1, a za sve ostale 0.

Pronalazak parametara mreže koji će rezultirati mrežom poželjnih svojstava, u prvom redu visokom razinom generalizacije te malom složenosti, nije jednostavno. Razlog tome je što se najčešće radi o velikom broju parametara koje je potrebno pronaći ili podesiti. Ipak, mnogo različitih pristupa predloženo je za traženje parametara RBFNs. Slijedi sažeti pregled nekih od njih, ne uvijek nužno s ciljem izgradnje klasifikacijskih modela, ali to ne predstav-

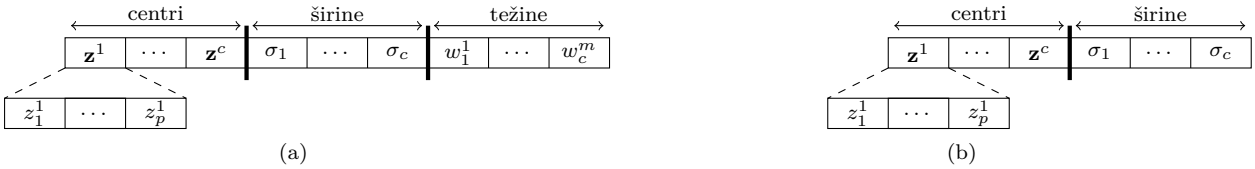
lja ograničavajući čimbenik u smislu traženja parametara RBFNs. Jedan od prvih pristupa predložili su Broomhead i Lowe [18], gdje se za centre Gaussovih funkcija (ili višekvadratnih funkcija) uzimaju nasumično odabrani uzorci iz skupa označenih, odnosno skupa za treniranje, a težine se potom određuju (estimiraju) metodom najmanjih kvadrata (pobliže opisano u nastavku). Moody i Darken [87] predložili su uporabu algoritma k-means za traženje centara Gaussovih radijalnih funkcija, što se postiže raspodjelom dostupnih uzoraka (ne uzimajući u obzir njihove oznake) u c grupa, a predstavnici grupa prikladno se uzimaju za centre. Pripadajuće širine, jednake za sve centre, računaju se kao prosječna udaljenost najbližih parova centara. Nakon što su utvrđeni centri i širine, težine veza skrivenog i izlaznog sloja računaju se metodom gradijentnog spusta (engl. *gradient descent*, GD). Slično tome, za određivanje centara, Vogt [138] je predložio uporabu klasifikacijskog algoritma kvantizacija vektora učenja (engl. *learning vector quantization*). Za razliku od prethodnog pristupa, pripadajuće širine računaju se za svaku funkciju posebno kao $\sigma_i = u \cdot \min\{\|\mathbf{z}^i - \mathbf{z}^j\| : j = 1, \ldots, c,\ j \neq i\}$, $i = 1, \ldots, c$, gdje je c broj čvorova u skrivenom sloju, a u je skalar (jednak 1.2 [138]). Težine se potom estimiraju metodom najmanjih kvadrata. Benoudjit i Verleysen [12], također, navode da treba izbjegavati jednake širine za sve radijalne funkcije te ukazuju na važnost traženja odgovarajućih širina. S obzirom na popularnost raznih prirodom inspiriranih algoritama optimizacije, ne iznenađuje što se u novije vrijeme često neki od njih predlažu za traženje parametara radijalnih mreža. U nastavku je kasnije dan pregled s gledišta navedenog.

6.2 Diferencijalna evolucija u izgradnji radijalnih mreža

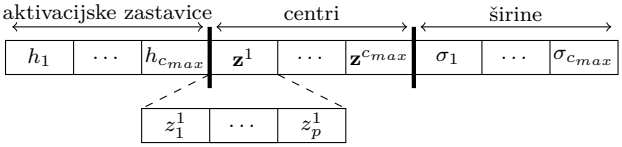
Parametri RBFNs, odnosno njihove komponente su realni brojevi. Stoga, primjena algoritma DE za traženje parametara mreže je jednostavna, te ne zahtijeva posebnu prilagodbu. Prvi korak u tom smjeru je određivanje načina predstavljanja rješenja. Slikom 6.3 pod (a) je prikazano izravno kodiranje svih parametara mreže, što je najjednostavniji način. Lako se može zaključiti kako su, općenito, takva rješenja vektori velike dimenzionalnosti koja je jednaka $c \cdot (p + 1 + m)$. Zbog velikog prostora pretrage pronalazak vrijednosti parametara koji će rezultirati mrežom zadovoljavajućih svojstava može tražiti veliki broj vrednovanja što za sobom izravno povlači poveće računalne zahtjeve. Kao što ukazuje slika 6.3 pod (b), kodiranje težina veza čvorova skrivenog i izlaznog sloja može se izbjeći, jer ih se može estimirati na temelju zadanih centara i pripadajućih širina. Naime, uključivanjem skupa od n označenih uzoraka $\mathbf{a}^1, \ldots, \mathbf{a}^n$ i željenih izlaza za svaki od njih, (6.3) može se zapisati (u matričnom obliku) kao sustav linearnih jednadžbi [63]

$$\mathbf{o}^j = \mathbf{\Phi} \cdot \mathbf{w}^j, \quad j = 1, \ldots, m, \tag{6.5}$$

gdje je $\mathbf{o}^j = [o_1^j \ \cdots \ o_n^j]^T$ vektor željenih vrijednosti na j-tom izlaznom čvoru, $\mathbf{w}^j =$



Slika 6.3: Predstavljanje rješenja (a) s kodiranjem težina i (b) bez kodiranja težina pri zadanom broju čvorova u skrivenom sloju.



Slika 6.4: Predstavljanje rješenja pri nepoznatom broju čvorova u skrivenom sloju.

$[w_1^j \ \cdots \ w_c^j]^T$ je vektor težina veza koje povezuju taj čvor sa svim čvorovima skrivenog sloja, dok je $\mathbf{\Phi} \in \mathbb{R}^{n \times c}$ matrica s elementima $\Phi_{r,s} = e^{-\|\mathbf{a}^r - \mathbf{z}^s\|^2/(2\sigma_s^2)}$, $r = 1, \ldots, n$, $s = 1, \ldots, c$. Kako je u pravilu $c < n$, težine je moguće odrediti metodom najmanjih kvadrata uz uporabu generaliziranog inverza (Moore-Penroseov inverz) $\mathbf{\Phi}^+$ matrice $\mathbf{\Phi}$ [18]. Rješavanjem linearnog problema najmanjih kvadrata, $\mathbf{w}^j = \mathbf{\Phi}^+ \cdot \mathbf{o}^j$, dobivaju se težine koje minimiziraju grešku $\|\mathbf{\Phi} \cdot \mathbf{w}^j - \mathbf{o}^j\|^2$. Prema tome, težine nije nužno kodirati te ih je moguće učinkovito estimirati na temelju pronađenih centara i pripadajućih širina. Time je dimenzionalnost vektora koji predstavljaju rješenja značajno manja u odnosu na prethodni slučaj i jednaka $c \cdot (p + 1)$.

Prethodno razmatranje načina predstavljanja rješenja pretpostavlja da je broj čvorova c u skrivenom sloju unaprijed zadan ili poznat. Međutim, to često nije tako pa je prikladni broj potrebno pronaći. Navedeni broj moguće je tražiti zajedno s ostalim parametrima mreže. Ovo zahtijeva odgovarajući način predstavljanja rješenja. Slikom 6.4 je prikazan mogući način, a korišten je i u [8]. Prema navedenom, svaki vektor populacije DE sastoji se od tri dijela, aktivacijskih zastavica $h_1, \ldots, h_{c_{max}}$ te centara $\mathbf{z}^1, \ldots, \mathbf{z}^{c_{max}}$ i pripadajućih širina $\sigma_1, \ldots, \sigma_{c_{max}}$, gdje je $c_{max} \in \mathbb{N}$ najveći dozvoljeni broj čvorova u skrivenom sloju. Aktivacijske zastavice $h_j \in [0, 1]$, $j = 1, \ldots, c_{max}$ određuju hoće li se koristiti odgovarajući centar i pripadajuća širina u rješenju odnosno kao parametar mreže. Točnije, za svaki $r \in \{1, \ldots, c_{max}\}$ ako je $h_r > 0.5$, odgovarajući centar $\mathbf{z}^r$ i širina σ_r uzimaju se kao dio rješenja, dok ih se suprotnom zanemaruje. Treba napomenuti kako se i u ovom slučaju ne moraju kodirati težine te ih je moguće estimirati na već spomenuti način. Dimenzionalnost vektora, jednaka $c_{max} \cdot (p+2)$, pri navedenom načinu predstavljanja rješenja uvelike ovisi o vrijednosti c_{max} koju je potrebno postaviti. Međutim, to je mnogo jednostavnije nego odrediti pogodan broj čvorova skrivenog sloja. Nadalje, treba uzeti u obzir i najmanji dozvoljeni broj čvorova skrivenog sloja c_{min} koji mora svako valjano rješenje zadovoljavati. Ovo se može nametnuti postavljanjem na vrijednosti > 0.5 nasumično odabranih aktivacijskih zastavica, dok se ne zadovolji uvjet.

Treba primijetiti kako kod navedenih predstavljanja rješenja, redoslijed pojedinih centara zajedno s pripadajućim širina (i težinama) nije bitan. Prema tome, radi se o mapiranju *više prema jedan*. Posljedica toga je veliki broj stacionarnih točaka funkcije cilja. Mogući način rješavanja ovog je sortiranje, primjerice, centara (po normi) uz odgovarajuće preslagivanje ostalih parametara. Ipak, s druge strane, takva vrsta mapiranja može pomoći u održavanju raznolikosti populacije.

Nakon što je odabran način predstavljanja rješenja u algoritmu, idući je korak odabir načina vrednovanja istih. U pravilu se uzima greška dobivenog u odnosu na željeni izlaz, kao što je zbroj kvadrata greški [13]

$$\frac{1}{2} \cdot \sum_{i=1}^{n} \sum_{j=1}^{m} \left(o_i^j - y_{j,i} \right)^2, \tag{6.6}$$

gdje je o_i^j željeni izlaz na j-tom čvoru za i-ti uzorak, a $y_{j,i}$ dobiveni izlaz j-tog čvora za i-ti uzorak. Slično tome, može se također koristiti srednja kvadratna greška (engl. *mean squared error*, MSE) [36]

$$\frac{\sum_{i=1}^{n} \sum_{j=1}^{m} \left(o_i^j - y_{j,i} \right)^2}{n \cdot m}, \tag{6.7}$$

koja je prikladnija ukoliko se želi napraviti usporedba učinkovitosti na različitim skupovima uzoraka. Izrazi (6.6) i (6.7) mogu se izravno koristiti u slučaju poznatog broja čvorova skrivenog sloja, ali kada ga se traži zajedno ostalim parametrima, oni vode prema velikim mrežama, jer greška opada s povećanjem istih. Navedeno se može popraviti uvođenjem kazne koja je proporcionalna broju c odnosno broju čvorova skrivenog sloja [8, 105]. Kazna se jednostavno uvodi kao dodatni član pa, primjerice, (6.7) postaje

$$\frac{\sum_{i=1}^{n} \sum_{j=1}^{m} \left(o_i^j - y_{j,i} \right)^2}{n \cdot m} + \lambda \cdot c, \tag{6.8}$$

gdje je $\lambda \in [0, +\infty\rangle$ težina koja skalira parametar c. Prema tome, kazna $\lambda \cdot c$ je manja za male mreže i obratno. Ovime se pokušava ostvariti ravnoteža između složenosti i sposobnosti generalizacije mreže, odnosno pokušava se pronaći vrijednosti parametara koje rezultiraju relativno malim mrežama zadovoljavajućih svojstava.

S obzirom na izuzetno raznoliku primjenu koju je pronašla DE, ne iznenađuje što je uspješno primijenjena i za treniranje, odnosno izgradnju RBFNs. Osim DE, u literaturi su predložene primjene i brojnih drugih prirodom inspiriranih algoritama globalne optimizacije. Tako su primjerice, Kurban i Beşdok [64] predložili uporabu algoritma umjetne kolonije pčela (engl. *artificial bee colony*, ABC) čije jedinke ili rješenja populacije kodiraju sve parametre mreže. Navedeni pristup usporedili su s GA i traženjem parametara metodom GD te Kalmanovim filtrom. Usporedbom je pokazano da je algoritam ABC učinkovitiji u odnosu

na suparničke pristupe bez obzira na složenost mreže u smislu broja čvorova u skrivenom sloju. Međutim, Horng et al. [52] u svojoj su usporedbi pokazali da je algoritam krijesnice (engl. *firefly algorithm*, FA) učinkovitiji u odnosu na GA, metodu GD te algoritme ABC i optimizacije rojem čestica (engl. *particle swarm optimization*, PSO). Kao i u prethodnom pristupu, u predloženom FA rješenja populacije kodiraju sve parametre mreže. Isto tako, usporedba je provedena s različitim brojevima čvorova u skrivenom sloju. Kako bi pokazali učinkovitost u odnosu na novije prirodom inspirirane algoritme, Bajer et al. [10] usporedili su algoritam DE s algoritmima ABC i FA te s PSO kao starijim i dokazanim algoritmom. Usporedba je pokazala da je DE vrlo konkurentan i jedino mu se PSO uspijeva približiti. U svim navedenim algoritmima, rješenja su predstavljena kao na slici 6.3 pod (b), a težine su estimirane metodom najmanjih kvadrata. Treba napomenuti kako je, za razliku od prije spomenutih usporedbi, promatrana u prvom redu učinkovitost u smislu prilagodbe klasifikacijskog modela poznatim uzorcima, a ne toliko nepoznatim. Ovime se ostvaruje bolji uvid u učinkovitost algoritama optimizacije.

U svim prethodno navedenim pristupima traženja parametara mreže, pretpostavljalo se da je broj čvorova skrivenog sloja unaprijed zadan. Međutim, kao što je ranije spomenuto, to je rijetko tako, te je potrebno odrediti pogodan broj čvorova skrivenog sloja. Qin et al. [105] predložili su traženje broja čvorova skrivenog sloja usporedno s ostalim parametrima mreže pomoću PSO. Za predstavljanje rješenja korišteno je blago izmijenjeno predstavljanje kao na slici 6.4 (drukčiji raspored zastavica, ali ista su svojstva) uz određivanje težina veza skrivenog i izlaznog sloja metodom najmanjih kvadrata. Dodatno je predložena, problemu prilagođena, metoda inicijalizacije populacije, gdje se za centre odabiru nasumično uzorci iz skupa, a za svaki se centar širina računa kao prosječna udaljenost do preostalih centara. Inačicu ovog prijedloga u kojoj je unaprijed zadan broj čvorova u skrivenom sloju, iskoristili su Korürek i Doğan [62] i to za traženje vrijednosti parametara mreže koja je primijenjena na problem klasifikacije EKG otkucaja. Pristup za određivanje, odnosno za traženje broja čvorova skrivenog sloja zajedno s ostalim parametrima zasnovan na DE predložili su Bajer et al. [8]. U navedenom su rješenja predstavljena kao na slici 6.4, a težine veza su određivane metodom najmanjih kvadrata. Kako bi se povećala učinkovitost algoritma DE, populacija je dijelom inicijalizirana rješenjima u kojim su centri pronađeni algoritmom k-means, a pripadajuće širine stohastičkom inačicom pravila predloženog u [138]. Osim toga, prostor pretrage je dinamički smanjivan u smislu dozvoljenog broja čvorova u skrivenom sloju, što je postignuto podešavanjem donje i gornje granice. Drukčiji pristup traženju prikladnog broja čvorova u skrivenom sloju koristili su O'Hora et al. [94], gdje su redom, od donje do gornje zadane granice, povećavali broj, pri čemu je svaki put izvođen algoritam DE za traženje parametara mreže.

6.3 Eksperimentalni rezultati i analiza: Izgradnja RBFNs pomoću DE

Iz prethodnih razmatranja jasno je da izgradnja klasifikacijskih modela predstavljenih RBFNs zahtijeva pronalazak velikog broja parametara. Ovo se posebno odnosi na slučaj kada broj čvorova skrivenog sloja nije unaprijed poznat ili zadan. Stoga je provedena eksperimentalna analiza s ciljem dobivanja uvida u ponašanje i učinkovitost DE pri rješavanju opisanog problema, jer rezultate prethodnih analiza nije moguće izravno preslikati na isti. U analizi su razmatrani standardni algoritam DE i isti algoritam koji ugrađuje po jedno od predloženih unaprjeđenja (analogno testiranjima provedenim u prethodnim poglavljima). Analiza je podijeljena u dva dijela. Prvi dio se odnosi na testiranje i analizu standardnog algoritma i njegovu usporedbu s nekoliko drugih pristupa izgradnji modela. U drugom dijelu ispitane su prednosti i nedostaci do sada predloženih unaprjeđenja algoritma pri izgradnji RBFNs. Nadalje, testiranje i analiza provedeni su nad nekoliko standardnih i u literaturi često upotrijebljenih skupova podataka (uzoraka) za klasifikaciju. Karakteristike navedenih skupova sažeto su prikazane tablicom 6.1, a opisani su u dodatku B.

6.3.1 Postavke eksperimenata

Za svaki skup podataka i algoritam izvršeno je 25 nezavisnih izvođenja. Zbog dosljednosti s prethodno provedenim testiranjima korištene su iste postavke parametara, odnosno veličina populacije $NP = 100$, faktor skaliranja $F = 0.5$ i stopa križanja $CR = 0.9$. Pri svakom izvođenju bilo je dozvoljeno $c_{max} \cdot 10^4$ vrednovanja funkcije cilja, a uvjet prekida bio je dosezanje toga broja. Populacije algoritama incijalizirane su nasumično stvorenim rješenjima unutar cijelog prostora pretrage.

Vrijednosti značajki koje opisuju uzorke u skupovima normalizirane su u interval $[0, 1]$ s ciljem uklanjanja utjecaja eventualno različitih raspona u kojima se mogu nalaziti pojedine vrijednosti značajki. Normalizacija je izvršena prema [118], a postupak je opisan u dodatku B. Nadalje, kao što je pravilo u izgradnji klasifikacijskih modela, svaki skup označenih uzoraka $\mathcal{Q}$ podijeljen je u podskup za treniranje $\mathcal{Q}_T \subset \mathcal{Q}$ i podskup za nezavisno vrednova-

Tablica 6.1: Karakteristike korištenih skupova podataka.

skup	ime	broj uzoraka	broj značajki	broj oznaka
$\mathcal{Q}_1$	Banknote authentication	1372	4	2
$\mathcal{Q}_2$	Glass identification	214	9	6
$\mathcal{Q}_3$	Ionosphere	351	34	2
$\mathcal{Q}_4$	Iris	150	4	3
$\mathcal{Q}_5$	Pima indians diabetes	768	8	2
$\mathcal{Q}_6$	Seeds	210	7	3
$\mathcal{Q}_7$	Thyroid	215	5	3
$\mathcal{Q}_8$	Wine	178	13	3

nje $\mathcal{Q}_V = \mathcal{Q} \setminus \mathcal{Q}_\mathcal{T}$. Podjela je izvršena nasumično u zadržavanje omjera broja zastupljenih uzoraka svake oznake (engl. *stratified split*). S obzirom da se $\mathcal{Q}_T$, koristi pri traženju parametara, on čini približno 70 % skupa $\mathcal{Q}$, a preostalih 30 % čini $\mathcal{Q}_V$ koji se koriste za naknadno i nezavisno vrednovanje. Ova podjela i način testiranja okvirno odgovaraju onima u [105]. Treba napomenuti da je podjela skupa napravljena samo jednom kako bi svi algoritmi radili s istim podacima i time se osigurala pravedna usporedba.

U svim algoritmima rješenja su predstavljena kao što je prikazano slikom 6.4. Vrednovanje rješenja je provođeno prema izrazu (6.8) uz $\lambda = \frac{0.01}{2 \cdot m}$ (prema [8]), gdje je m broj izlaza mreže (jednako broju oznaka za dani skup uzoraka). Kao što je već napomenuto, pri određivanju broja čvorova u skrivenom sloju zajedno s ostalim parametrima mreže, potrebno je postaviti donju i gornju granicu. U tom pogledu, za donju granicu odabrano je $c_{min} = 2$, a $c_{max} = 20$ za gornju granicu.

6.3.2 Učinkovitost i ponašanje standardnog algoritma DE

Brojni algoritmi optimizacije, kao što je ranije opisano, predloženi su za traženje dobrih ili pogodnih vrijednosti parametara RBFNs. Kako bi se dao uvid u učinkovitost standardnog algoritma DE, provedeno je testiranje i njegova usporedba s tri drukčija prirodom inspririrana algoritama te usporedba s po dvije inačice dva klasična pristupa određivanja parametara mreže. Točnije, u usporedbu su uključene standardne inačice algoritma ABC [84], FA [148, 149] i PSO [62, 105]. Osim toga, uključene su dvije inačice pristupa koji koristi algoritam k-means (označene s K) za određivanje centara te dvije inačice pristupa koji nasumično odabire uzorke za centre (označene s R). Inačice se razlikuju u pravilima za postavljanje pripadajućih širina. Prva inačica, označena s $\mathrm{K_{max}}$ odnosno s $\mathrm{R_{max}}$, računa širine kao $\sigma_i = \frac{d_{max}}{\sqrt{2 \cdot c}}$ [12, 36], gdje je $d_{max} = \max\{\|\mathbf{z}^r - \mathbf{z}^s\| : \forall r, s = 1, \ldots, c\}$, dok druga, označena s $\mathrm{K_{avg}}$ odnosno s $\mathrm{R_{avg}}$, širine računa prema [138] (prethodno opisano). U svim navedenim pristupima težine veza skrivenog i izlaznog sloja estimirane su metodom najmanjih kvadrata.

Parametri korištenih algoritama prikazani su u tablici 6.2, a koji su za ABC i FA preuzeti iz [10], dok su za PSO preuzeti iz [62]. Za razliku od ostalih, pristupi zasnovani na algoritmu k-means te nasumičnom odabiru centara izvođeni su na poseban način. Naime, broj čvorova u skrivenom sloju c je redom povećavan od c_{min} do c_{max} te je svaki put izvršeno 10^4 traženja parametara pri čemu su bilježeni najbolji pronađeni u smislu izraza (6.7) za svaki c. Postavke za algoritam k-means, odnosno najveći dozvoljeni broj iteracija i ciljana

Tablica 6.2: Korišteni parametri pristupa u usporedbi.

alg.	parametri
ABC	$NS = 30$, $limit = 200$
FA	$NS = 30$, $\alpha = 0.1$, $\gamma = \beta_0 = 1$
PSO	$NS = 20$, $c_1 = c_2 = 1.496$, $\omega = 0.7298$
k-means	$t_{max} = 50$, $\epsilon = 0.01$

Tablica 6.3: Rezultati u smislu MSE za prirodnom inspirirane algoritme u usporedbi.

$\mathcal{Q}$	ABC	FA	PSO	DE
	prosjek ± std. dev.	prosjek ± std. dev.	prosjek ± std. dev.	prosjek ± std. dev.
1	1.626E−02↓ ± 3.15E−03	2.411E−02↓ ± 1.97E−03	8.183E−03↓ ± 6.14E−03	**3.181E−03** ± 2.09E−07
2	4.577E−02↓ ± 3.10E−03	7.410E−02↓ ± 1.99E−03	3.647E−02∘ ± 6.50E−03	**3.360E−02** ± 4.97E−03
3	2.046E−02∘ ± 3.40E−03	8.623E−02↓ ± 5.16E−03	**8.520E−03**↑ ± 4.50E−03	2.25E−02 ± 4.60E−03
4	8.370E−03∘ ± 1.65E−03	2.129E−02↓ ± 2.81E−03	**1.304E−03**↑ ± 2.30E−03	7.852E−03 ± 6.44E−03
5	1.343E−01∘ ± 4.30E−03	1.598E−01↓ ± 2.87E−03	**1.259E−01**↑ ± 6.59E−03	1.36E−01 ± 3.73E−03
6	1.296E−02↓ ± 2.73E−03	3.207E−02↓ ± 2.64E−03	**3.658E−03**↑ ± 5.38E−03	9.920E−03 ± 3.04E−03
7	1.484E−02↓ ± 2.72E−03	2.875E−02↓ ± 2.33E−03	6.614E−03↓ ± 6.06E−03	**2.916E−03** ± 2.61E−12
8	4.941E−03↓ ± 1.90E−03	3.382E−02↓ ± 2.07E−03	4.518E−04↓ ± 1.23E−03	**3.679E−04** ± 1.09E−06
—	5	8	3	↓
—	3	0	1	∘
—	0	0	4	↑

Tablica 6.4: Rezultati u smislu MSE za klasične pristupe u usporedbi.

$\mathcal{Q}$	K_{max}	R_{max}	K_{avg}	R_{avg}	DE
	najbolje	najbolje	najbolje	najbolje	najbolje
1	1.236E−02	1.169E−02	1.367E−02	1.381E−02	**3.181E−03**
2	5.803E−02	5.829E−02	5.396E−02	5.112E−02	**2.310E−02**
3	5.082E−02	6.670E−02	4.089E−02	4.060E−02	**1.182E−02**
4	1.160E−02	1.164E−02	8.452E−03	8.851E−03	**1.271E−03**
5	1.464E−01	1.430E−01	1.439E−01	1.430E−01	**1.291E−01**
6	1.848E−02	1.693E−02	2.140E−02	2.032E−02	**3.807E−03**
7	1.368E−02	1.117E−02	9.313E−03	9.012E−03	**2.916E−03**
8	4.641E−03	8.371E−03	1.379E−02	1.683E−02	**3.669E−04**

razlika (udaljenost) između bilo kojeg para centara dvije uzastopne iteracije (pretpostavka konvergencije), također su dani u tablici.

Tablicom 6.3 prikazani su rezultati usporedbe DE s ABC, FA i PSO algoritmima, a tablicom 6.4 rezultati usporedbe s inačicama pristupa zasnovanog na algoritmu k-means i pristupa zasnovanog na nasumičnom odabiru uzoraka kao centara. U slučaju prve usporedbe, proveden je Wilcoxonov test ranga s predznakom uz interval pouzdanosti od 95 %. Sukladno tome, znak ↓ označava statistički značajnu razliku prosjeka u korist DE, suprotno je označeno s ↑, a znakom ∘ odsutnost iste. Na dnu tablice 6.3 sažeto su prikazani rezultati provedenih statističkih testova.

Prema prikazanom, jasno se vidi da su PSO i DE osjetno učinkovitiji u odnosu na ABC i posebno FA, što se ogleda u lako primjetnim manjim prosjecima MSE na svim skupovima podataka. Štoviše, PSO se pokazao blago učinkovitijim od DE. Međutim, treba imati na umu veliku razliku u veličinama populacija pa se može očekivati ostvarenje veće brzine konvergencije DE pri smanjenju populacije. Ovo sugeriraju i rezultati iz [10], gdje se učinkovitijim pokazao algoritam s $NP = 50$, nego s $NP = 100$ (ostali parametri bili su jednaki kao ovdje korišteni). Nadalje, mora se napomenuti da su u tablicama prikazani MSE, a ne stope pogrešne klasifikacije (engl. *misclassification rate*, MCR). Stoga ne treba iznenaditi manja učinkovitosti ABC i FA iako su oni, kao što je prethodno opisano, pokazani vrlo učinkovitima za treniranje RBFNs (vidi [52, 64]). Naime, pretraga je u svim algoritmima vođena

Tablica 6.5: Prosječne vrijednosti MCR za prirodnom inspirirane algoritme u usporedbi.

$\mathcal{Q}$	ABC		FA		PSO		DE	
	MCR($\mathcal{Q}_T$)	MCR($\mathcal{Q}_V$)	MCR($\mathcal{Q}_T$)	MCR($\mathcal{Q}_V$)	MCR($\mathcal{Q}_T$)	MCR($\mathcal{Q}_V$)	MCR($\mathcal{Q}_T$)	MCR($\mathcal{Q}_V$)
1	0.006	0.010	0.014	0.016	0.001	0.002	**0.000**	**0.000**
2	0.152	0.322	0.288	0.375	0.101	0.324	**0.101**	**0.312**
3	0.009	**0.082**	0.087	0.132	**0.005**	0.091	0.015	0.082
4	0.007	0.024	0.017	0.026	**0.000**	0.030	0.004	**0.023**
5	0.190	0.239	0.222	**0.235**	**0.175**	0.251	0.193	0.245
6	0.006	0.081	0.037	0.074	**0.001**	0.096	0.012	**0.063**
7	0.020	0.091	0.042	0.114	0.003	0.079	**0.000**	**0.062**
8	0.001	0.023	0.018	0.041	**0.000**	0.033	**0.000**	**0.018**

Tablica 6.6: Prosječne vrijednosti MCR za klasične pristupe u usporedbi.

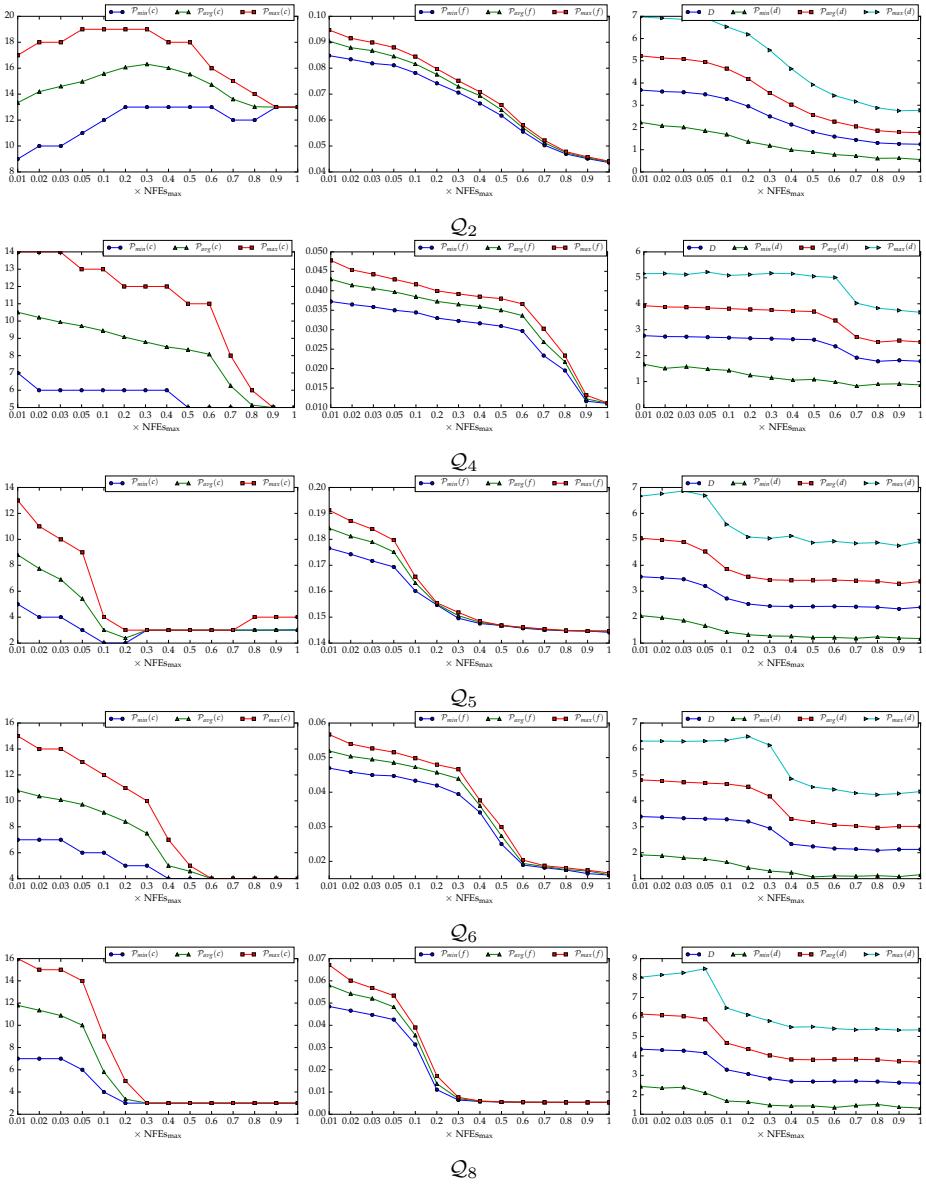
$\mathcal{Q}$	K_{max}		R_{max}		K_{avg}		R_{avg}		DE	
	MCR($\mathcal{Q}_T$)	MCR($\mathcal{Q}_V$)	MCR($\mathcal{Q}_T$)	MCR($\mathcal{Q}_V$)	MCR($\mathcal{Q}_T$)	MCR($\mathcal{Q}_V$)	MCR($\mathcal{Q}_T$)	MCR($\mathcal{Q}_V$)	MCR($\mathcal{Q}_T$)	MCR($\mathcal{Q}_V$)
1	0.003	0.005	0.003	0.005	0.005	0.007	0.001	**0.000**	**0.000**	**0.000**
2	0.162	0.364	0.203	0.409	0.169	0.394	0.176	0.318	**0.061**	**0.303**
3	0.053	0.085	0.078	0.104	0.053	**0.047**	0.037	0.057	**0.008**	0.057
4	0.019	0.022	0.010	0.022	0.010	0.022	0.010	0.022	**0.000**	0.022
5	0.220	**0.221**	0.216	0.234	0.190	0.225	0.210	0.238	**0.171**	0.260
6	0.014	0.111	0.020	**0.079**	0.034	0.095	0.034	0.111	**0.000**	**0.079**
7	0.020	0.092	**0.000**	0.108	**0.000**	0.077	0.007	0.092	**0.000**	**0.062**
8	**0.000**	**0.000**	0.016	0.036	0.008	0.018	**0.000**	0.036	**0.000**	0.018

Tablica 6.7: Rezultati u smislu složenosti pronađenih mreža za prirodom inspirirane algoritme u usporedbi.

$\mathcal{Q}$	ABC	FA	PSO	DE
	prosjek ± std. dev.	prosjek ± std. dev.	prosjek ± std. dev.	prosjek ± std. dev.
1	4.84 ± 0.97	5.20 ± 0.85	4.76 ± 1.14	**3.00** ± 0.00
2	13.00 ± 1.62	**12.64** ± 2.17	14.20 ± 1.70	12.96 ± 1.73
3	7.68 ± 1.43	9.40 ± 2.08	7.28 ± 1.84	**3.28** ± 0.66
4	6.20 ± 0.80	7.92 ± 1.47	6.56 ± 1.02	**5.04** ± 1.15
5	7.16 ± 1.87	6.00 ± 1.20	5.96 ± 1.28	**3.64** ± 0.56
6	7.64 ± 1.38	7.96 ± 1.40	7.52 ± 0.94	**3.92** ± 0.84
7	5.28 ± 1.43	8.60 ± 1.50	7.76 ± 2.57	**3.00** ± 0.00
8	5.24 ± 1.21	9.12 ± 1.18	5.08 ± 1.90	**3.00** ± 0.00

izrazom (6.8) (MSE uz dodanu kaznu) pa je MSE bolji pokazatelj učinkovitosti algoritma u odnosu na MCR. Ovo potvrđuju i rezultati u tablicama 6.5 i 6.6, gdje su prikazani prosječni MCR na skupovima za treniranje i vrednovanja. I dok se u pojedinim slučajevima MCR ostvaren na skupu za treniranje ($\mathcal{Q}_T$) relativno dobro slaže s MSE (manji prosječni MSE odgovara manjem prosječnom MCR i obratno), to nikako nije istina na skupu za nezavisno vrednovanje ($\mathcal{Q}_V$). Nije teško zaključiti da će učinkovit algoritam optimizacije gotovo uvijek pretjerano prilagoditi model uzorcima za treniranje (engl. *over-fitting*), što predstavlja dobro poznat problem u izradi klasifikacijskih modela (vidi primjerice [143, 156]). Ipak, takav algoritam može dovesti do pronalaska dobrih vrijednosti parametara kroz manje vrednovanja funkcije cilja (FEs) što na koncu može dovesti do značajne uštede vremena računanja. Treba napomenuti da su upravo zbog navedenog razloga, rezultati u smislu MCR prikazani samo zbog cjelovitosti testiranja i analize.

Što se tiče usporedbe DE s klasičnim inačicama pristupa izgradnji RBFNs kao klasifika-



Slika 6.5: Ponašanje populacije algoritma DE.

cijskih modela, jasna je nadmoć algoritma DE. Ne samo što su najbolja ostvarena rješenja, u smislu MSE, kvalitetnija, nego su kvalitetniji i prosjeci (kao što se može vidjeti usporedbom rezultata iz tablica 6.3 i 6.4). Štoviše, najčešće su dobivene mreže rezultirale manjim MCR, kako na skupovima za treniranje, tako i na skupovima za vrednovanje. Sukladno tim rezultatima, može se opravdati uporaba DE i sličnih algoritama optimizacije (za izgradnju RBFNs), barem u odnosu na pristupe upotrijebljene u usporedbi. Osim toga, treba promotriti složenost mreža dobivenih algoritmom DE s gledišta broja čvorova skrivenog sloja koje su uglavnom osjetno manje u odnosu na mreže dobivene drugim prirodom inspiriranim algoritmima u usporedbi kao što je vidljivo iz tablice 6.7. Navedeno je mnogo više izraženo u slučaju klasičnih pristupa čija su najbolja rješenja (u smislu MSE) ostvarena mrežama koje redovito imaju između 17 i 20 čvorova u skrivenom sloju.

Piotrowski [100] je pokazao kako mnoge različite varijante ili inačice algoritma DE ispoljavaju stagnaciju pri treniranju višeslojnih perceptrona (engl. *multi-layer perceptrons*, MLPs) koji su, uz RBFNs, često korištena vrsta ANNs. Međutim, RBFNs i MLPs razlikuju se u strukturi i načinu traženja parametara odnosno treniranja pa se ti rezultati se ne mogu izravno preslikati na slučaj RBFNs. Dodatni razlog je što se u [100] radilo o unaprijed zadanom broju čvorova te o problemima mnogo manjih dimenzionalnosti (46 varijabli u odnosu na 120 do 720 varijabli). Nastavno na razmatranje u [10], gdje je broj čvorova skrivenog sloja bio zadan, ovdje je razmatrano ponašanje populacije DE kada taj podatak nije unaprijed poznat. U tu svrhu, praćeno je nekoliko pokazatelja stanja populacije. Točnije, praćeni su najmanja, prosječna i najveća mreža [označeno s $\mathcal{P}_{min}(c)$, $\mathcal{P}_{avg}(c)$ i $\mathcal{P}_{max}(c)$, redom], isto to za vrijednosti funkcije cilja [označeno s $\mathcal{P}_{min}(f)$, $\mathcal{P}_{avg}(f)$ i $\mathcal{P}_{max}(f)$, redom] i (Euklidske) udaljenosti između pojedinih vektora populacije [označeno s $\mathcal{P}_{min}(d)$, $\mathcal{P}_{avg}(d)$ i $\mathcal{P}_{max}(d)$, redom] te posebno raznolikost (označeno s D) računata analogno (3.7).

Slikom 6.5 prikazani su medijani izvođenja u smislu navedenih pokazatelja za nekoliko odabranih skupova, a vrlo slična ponašanja su vidljiva i za preostale skupove. Zanimljivo je primijetiti da populacija konvergira u smislu broja čvorova u skrivenom sloju, što za sobom povlači i (praktički) konvergenciju u vrijednosti funkcije cilja [izraz (6.8)]. Iako, kao što ukazuju međusobne udaljenosti vektora populacije, ona ne konvergira prostorno. To nije ni nužno kako bi sva rješenja bila ista, jer zbog aktivacijskih zastavica, ne sudjeluju sve komponente vektora u opisu mreže. Očuvanje prostorne raznolikosti može se smatrati korisnom, jer kao što sugeriraju grafovi konvergencije u smislu funkcije cilja, uglavnom ne dolazi do vidljive stagnacije pretrage, barem u opsegu dozvoljenog NFEs. Prema tome, može se zaključiti kako stagnacija pretrage općenito nije problem pri izgradnji RBFNs. Nadalje, ranije spomenuti lošiji rezultati u usporedbi s PSO mogu se pripisati usporenju konvergencije koja je primjetna na nekoliko skupova podataka (primjerice na $\mathcal{Q}_5$ i posebno $\mathcal{Q}_8$).

6.3.3 Učinkovitost unaprjeđenih inačica algoritma DE

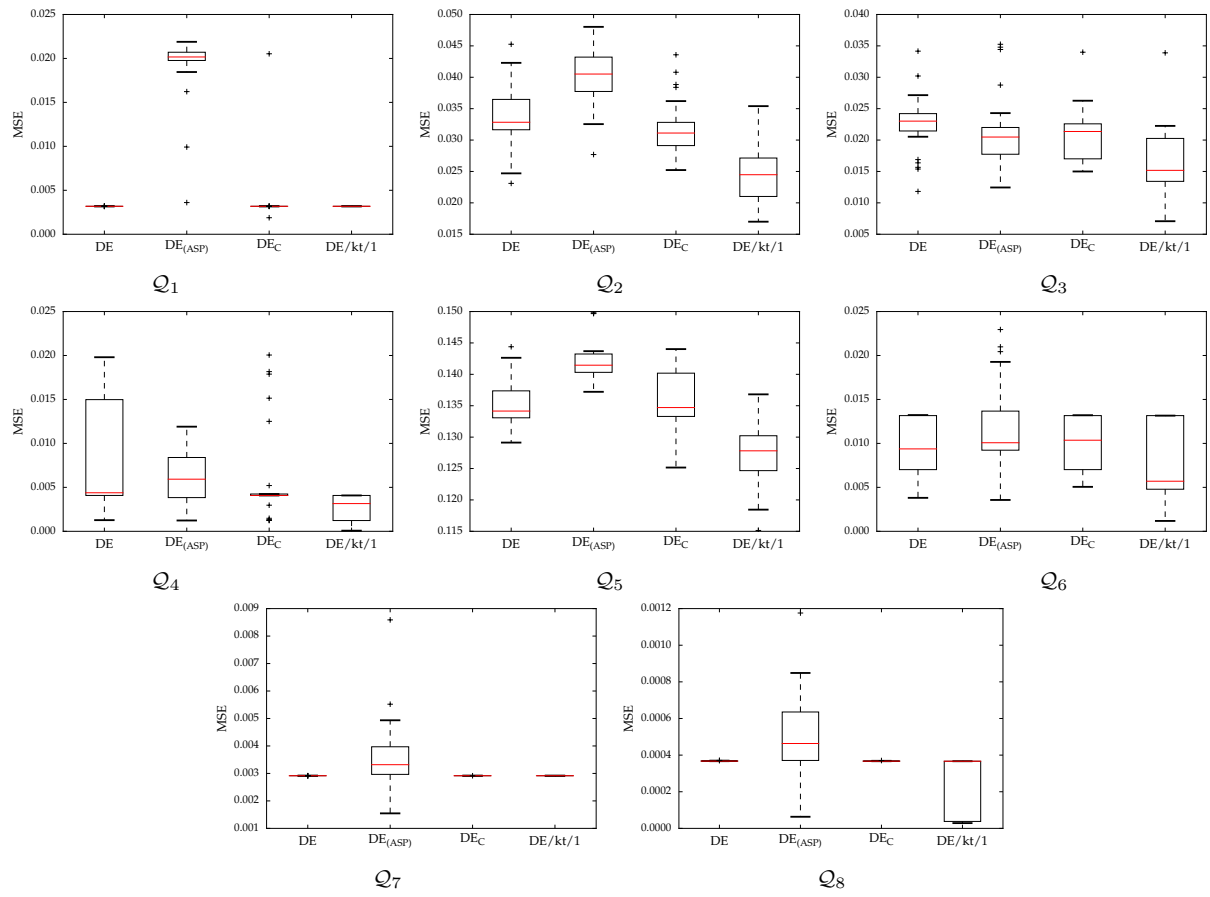
Iz prethodno prikazanih rezultata vidljivo je da standardni algoritam DE pruža vrlo konkurentu učinkovitost u odnosu na algoritme ABC, FA i PSO. Iako se PSO pokazao blago boljim u usporedbi, u prilog DE idu mreže relativno malih složenosti koje on uobičajeno pronalazi. Prema tome, standardni algoritam (DE/rand/1/bin) predstavlja valjan izbor za traženje parametara RBFNs za potrebe klasifikacije. Pitanje koje se odmah nameće je pružaju li i u kojoj mjeri veću učinkovitost predložene unaprjeđene inačice algoritma. Kako bi se pružio, barem donekle, odgovor na ovo pitanje, provedeni su testiranje i analiza utjecaja unaprjeđenja predloženih u poglavljima 3, 4 i 5 na ponašanje algoritma u koji su ugrađena.

Tablicom 6.8 prikazani su rezultati ostvareni algoritmima koji ugrađuju predložena unaprjeđenja. Oni su dopunjeni dijagramima pravokutnika (engl. *box-and-whisker plots*) danim

Tablica 6.8: Rezultati u smislu MSE za standardnu i unaprjeđene inačice algoritma DE.

$\mathcal{Q}$	$\text{DE}_{(\text{ASP})}$	DE_{C}	DE/kt/1	DE
	prosjek ± std. dev.	prosjek ± std. dev.	prosjek ± std. dev.	prosjek ± std. dev.
1	1.907E−02↓ ± 3.90E−03	3.823E−03∘ ± 3.42E−03	**3.181E−03**↑ ± 4.34E−19	3.181E−03 ± 2.09E−07
2	4.026E−02↓ ± 4.90E−03	3.187E−02∘ ± 4.52E−03	**2.469E−02**↑ ± 4.88E−03	3.360E−02 ± 4.97E−03
3	2.117E−02∘ ± 6.25E−03	2.095E−02∘ ± 4.12E−03	**1.632E−02**↑ ± 5.86E−03	2.250E−02 ± 4.60E−03
4	6.070E−03∘ ± 2.99E−03	6.188E−03∘ ± 5.51E−03	**2.646E−03**↑ ± 1.45E−03	7.852E−03 ± 6.44E−03
5	1.422E−01↓ ± 3.18E−03	1.363E−01∘ ± 4.96E−03	**1.271E−01**↑ ± 4.67E−03	1.356E−01 ± 3.73E−03
6	1.165E−02∘ ± 5.15E−03	1.022E−02∘ ± 3.07E−03	**7.215E−03**↑ ± 4.00E−03	9.920E−03 ± 3.04E−03
7	3.692E−03↓ ± 1.34E−03	2.916E−03∘ ± 1.02E−12	**2.916E−03**↑ ± 4.34E−19	2.916E−03 ± 2.61E−12
8	4.875E−04↓ ± 2.52E−04	3.674E−04∘ ± 8.45E−07	**2.395E−04**↑ ± 1.56E−04	3.679E−04 ± 1.09E−06
—	5	0	0	↓
—	3	8	0	∘
—	0	0	8	↑

slikom 6.6, u kojima pravokutnici predstavljaju prvi, drugi (medijan) i treći kvartil, a brkovi najmanju i najveću vrijednost koja nije sumnjiva (engl. *outlier*). Među rezultatima prikazani su i oni (prethodno) dobiveni standardnim algoritmom za potrebe usporedbe. Jednako kao i prethodno, proveden je i Wilcoxonov test ranga s predznakom. Treba istaknuti da su rezultati u smislu MCR, prikazani tablicom 6.9, dani zbog cjelovitosti analize, jer kao što je ranije istaknuto, rezultati u smislu MSE pružaju jasniji uvid u učinkovitost algoritama.



Slika 6.6: Dijagrami pravokutnika za standardnu i predložene unaprjeđenje inačice algoritma DE.

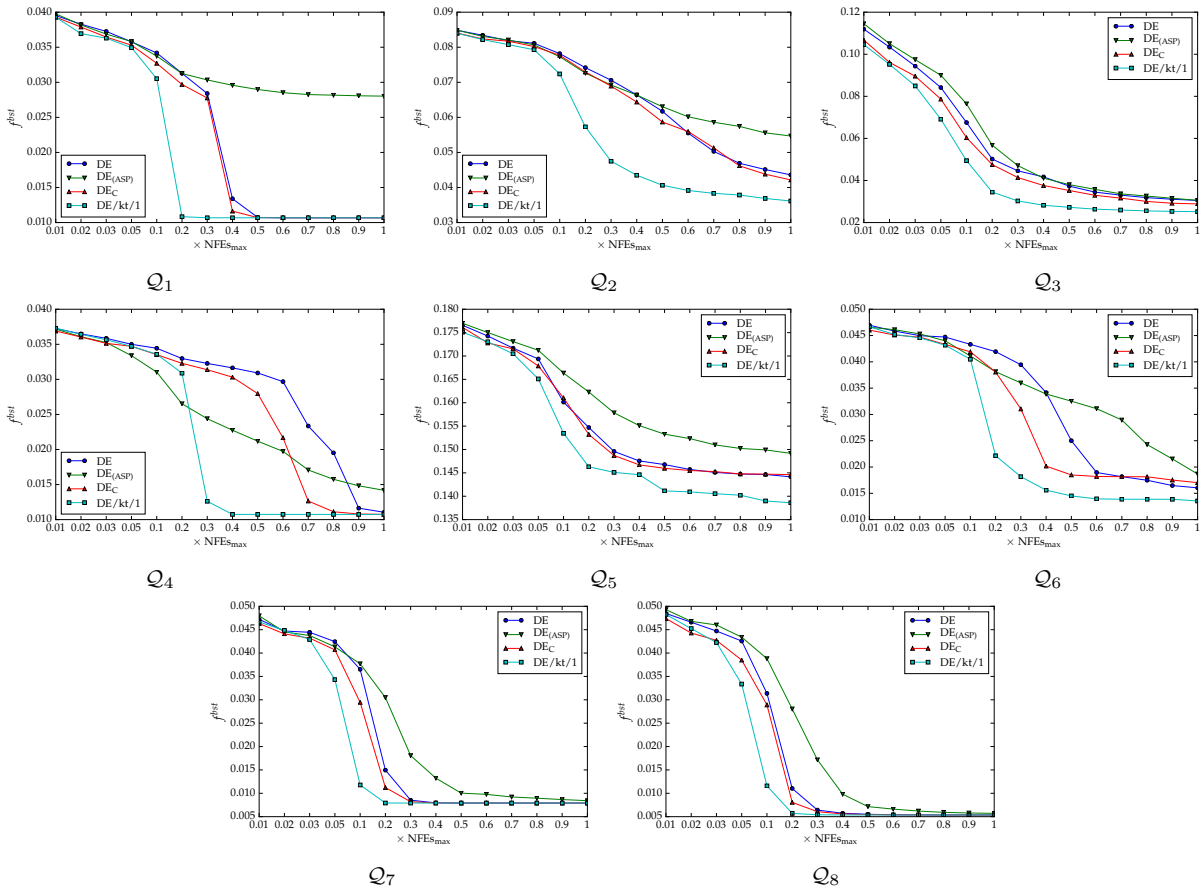
Tablica 6.9: Rezultati u smislu MCR za standardnu i unaprjeđene inačice algoritma DE.

$\mathcal{Q}$	$\text{DE}_{(\text{ASP})}$		DE_{C}		DE/kt/1		DE	
	$\text{MCR}(\mathcal{Q}_T)$	$\text{MCR}(\mathcal{Q}_V)$	$\text{MCR}(\mathcal{Q}_T)$	$\text{MCR}(\mathcal{Q}_V)$	$\text{MCR}(\mathcal{Q}_T)$	$\text{MCR}(\mathcal{Q}_V)$	$\text{MCR}(\mathcal{Q}_T)$	$\text{MCR}(\mathcal{Q}_V)$
1	0.010	0.014	0.0004	0.0005	**0.000**	**0.000**	**0.000**	**0.000**
2	0.126	0.332	0.090	0.315	**0.069**	**0.306**	0.101	0.312
3	0.014	**0.075**	0.013	0.085	**0.009**	0.091	0.015	0.082
4	0.003	0.025	0.004	**0.023**	**0.000**	0.024	0.004	**0.023**
5	0.202	**0.243**	0.193	0.245	**0.181**	0.247	0.193	0.245
6	0.007	0.067	0.013	0.070	**0.006**	0.077	0.012	**0.063**
7	0.001	0.068	**0.000**	**0.062**	**0.000**	**0.062**	**0.000**	**0.062**
8	0.000	0.021	0.000	**0.018**	0.000	0.023	0.000	**0.018**

Tablica 6.10: Rezultati u smislu složenosti mreža za standardnu i unaprjeđene inačice algoritma DE.

$\mathcal{Q}$	$\text{DE}_{(\text{ASP})}$	DE_{C}	DE/kt/1	DE
	prosjek ± std. dev.	prosjek ± std. dev.	prosjek ± std. dev.	prosjek ± std. dev.
1	3.08 ± 0.27	3.08 ± 0.27	**3.00** ± 0.00	**3.00** ± 0.00
2	16.40 ± 1.33	**12.60** ± 1.13	14.08 ± 1.57	12.96 ± 1.73
3	4.08 ± 1.13	3.36 ± 0.69	3.80 ± 0.80	**3.28** ± 0.66
4	5.60 ± 0.80	**4.52** ± 0.75	4.56 ± 0.64	5.04 ± 1.15
5	**3.16** ± 0.83	3.60 ± 0.57	4.72 ± 0.72	3.64 ± 0.56
6	5.72 ± 1.00	**3.76** ± 0.81	4.36 ± 0.97	3.92 ± 0.84
7	3.16 ± 0.37	**3.00** ± 0.00	**3.00** ± 0.00	**3.00** ± 0.00
8	3.56 ± 0.70	**3.00** ± 0.00	3.40 ± 0.49	**3.00** ± 0.00

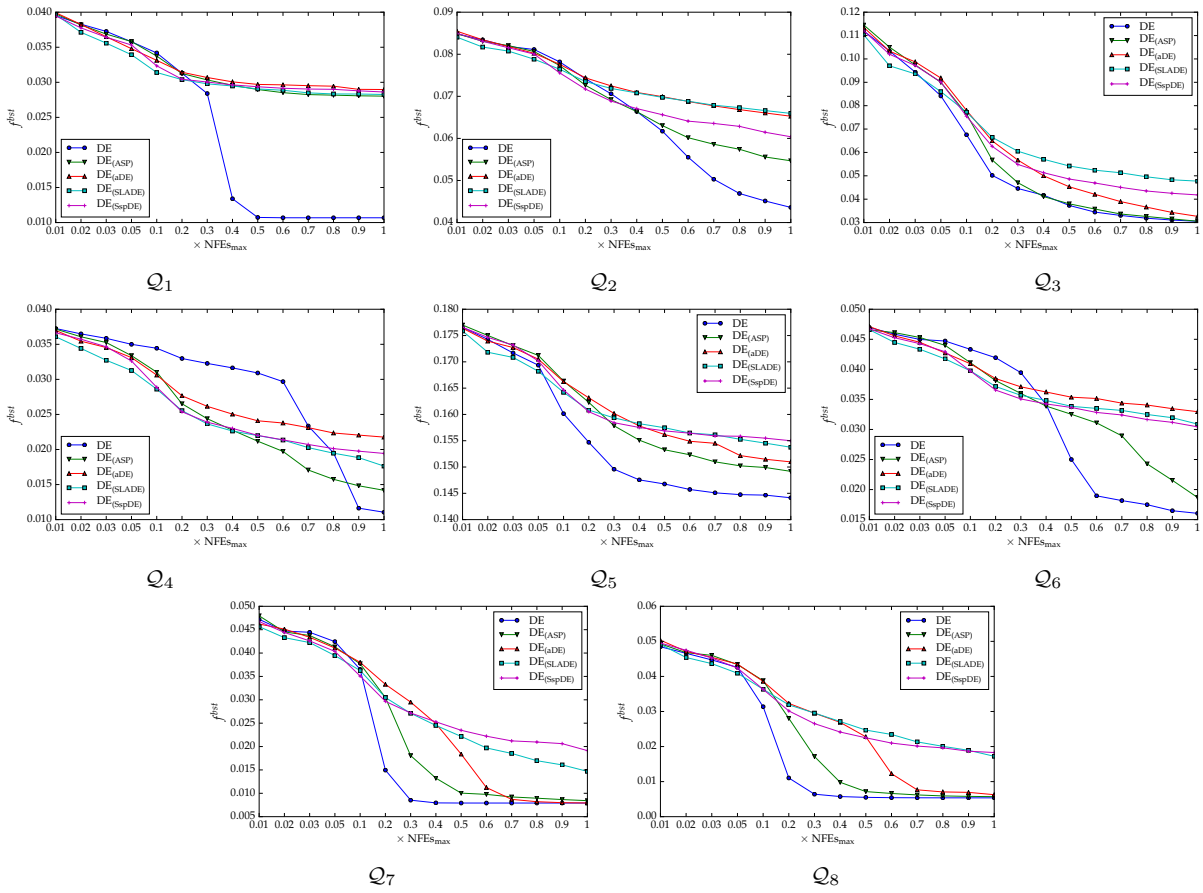
Iz prikazanih rezultata lako se mogu uočiti razlike u učinkovitosti pojedinih predloženih unaprjeđenja algoritma DE. Razlike dodatno potvrđuje slika 6.7 koja prikazuje medijane izvođenja (u smislu vrijednosti funkcije cilja najboljeg rješenja u odnosu na NFEs) i pruža uvid u ponašanje algoritama. Ono što posebno iznenađuje je negativan utjecaj predloženog postupka samopodešavanja faktora skaliranja i stope križanja (korišteno u $\text{DE}_{(\text{ASP})}$) na učinkovitost standardnog algoritma (u koji je ugrađen). Nadalje, razlike između algoritma sa i bez ugrađene predložene metode inicijalizacije (korištena u DE_{C}) nisu statistički značajne, iako cjelokupno gledano, idu blago u korist algoritmu s ugrađenom predloženom metodom. Ipak, promatranjem slika 6.6 i 6.7 vidljive su veće razlike u rasponu kvalitete ostvarenih rješenjima i brzini konvergencije. Bitno je napomenuti kako je, pri navedenom, parametar lokacije Cauchyjeve razdiobe smanjen na $s = 0.01$ i to zbog malih raspona vrijednosti koje mogu poprimiti pojedine komponente (posljedica normalizacije na $[0, 1]$). Algoritam s ugrađenom predloženom mutacijom (DE/kt/1), uvjerljivo se pokazao najučinkovitijim u usporedbi, što je jasno vidljivo iz prikazanih rezultata (redovito najmanji prosjeci MSE koji se vrlo dobro slažu s prosjecima MCR na skupu za treniranje). S obzirom na izuzetno veliki prostor pretrage, odnosno visoke dimenzionalnosti problema i relativno ograničen broj dozvoljenih FEs, usmjerenije istraživanje (postignuto s DE/kt/1), očekivano vodi do bržeg pronalaska dobrih rješenja. Na koncu, s gledišta složenosti pronađenih mreža (broj čvorova u skrivenom sloju), što je prikazano tablicom 6.10, razlike su uglavnom male, a većinom idu u korist algoritmu DE_{C}.



Slika 6.7: Grafovi konvergencije za standardnu i predložene unaprjeđenje inačice algoritma DE.

Rezultati prethodnih testiranja i analize na standardnim testnim funkcijama i funkcijama CEC 2014 ukazuju na pozitivan utjecaj predloženog postupka samopodešavanja faktora skaliranja i stope križanja. Stoga se rezultati pri izgradnji RBFNs mogu s pravom smatrati iznenađujućim. Dodatno testiranje i analiza provedeni su kako bi se pokušao odgonetnuti razlog njegovog negativnog utjecaja na učinkovitost algoritma u koji je ugrađen.

Kako bi se otkrilo je li u usporedbi mala učinkovitost svojstvena predloženom postupku samopodešavanja ili je pak posljedica samog podešavanja faktora skaliranja i stope križanja, izvršeno je testiranje uključivanjem ranije upotrijebljenih postupaka podešavanja tih parametara (korištenih u poglavlju 3). Ostvareni rezultati sažeto su dani slikom 6.8 koja prikazuje medijane izvođenja. Jasno je vidljivo da niti jedan upotrijebljeni postupak podešavanja parametara ne pruža pozitivan utjecaj na ponašanje osnovnog algoritma u koji su ugrađeni. Štoviše, algoritam s ugrađenim predloženim postupkom samopodešavanja ($\mathrm{DE}_{(\mathrm{ASP})}$) pokazao se sposobnijim u odnosu na ostale suparničke postupke, ostvarujući najčešće veću brzinu konvergencije i bolja konačno pronađena rješenja. U okviru prikazanog, može se zaključiti da stalne izmijene ili prilagodbe parametara nisu korisne za potrebe rješavanje danog problema. U svrhu daljnjeg ispitivanja, izvršeni su testiranje i analiza na nekoliko standardnih testnih funkcija za $d = 500$, da bi se utvrdilo je li visoka dimenzionalnost problema uzrok relativno



Slika 6.8: Grafovi konvergencije za standardni algoritam i isti s ugrađenim različitim postupcima podešavanja faktora skaliranja i stope križanja.

male učinkovitosti. Pri testiranju je NFEs_{max} bio $c_{max} \cdot 10^4$ kao i do sada. Rezultati testiranja, u smislu prosječnih grešaka optimizacije i pripadajućih standardnih devijacija, prikazani su tablicom 6.11. Vidi se da je $\text{DE}_{(\text{ASP})}$ na svim, osim na zadnje dvije funkcije ostvario bolje rezultate u odnosu na standardni algoritam. Treba primijetiti da funkcije f_{21} i f_{22} sadrže poseban član koji predstavlja kaznu, a funkcija f_{19} se razlikuju od f_{21} upravo po njemu (vidi dodatak A). Ovo sugerira da je uzrok lošeg ponašanja algoritama s ugrađenim postupcima podešavanja faktora skaliranja i stope križanja, član koji predstavlja kaznu u izrazu (6.8). Potporu ovoj pretpostavci pruža i slika 6.9 koja daje uvid u proces podešavanja parametara predloženog postupaka pri izgradnji RBFNs. Prikazani su prosjeci vrijednosti parametara koji u rezultirali pokusnim vektorom koji je prešao u narednu generaciju (označeno s $\overline{F^{(+)}}$ i $\overline{CR^{(+)}}$) te vrijednosti koje su dovele do poboljšanja do tada najboljeg pronađenog rješenja (označeno s $F^{(\uparrow)}$ i $CR^{(\uparrow)}$). Bitno je uočiti veliku razliku između $\overline{CR^{(+)}}$ i $CR^{(\uparrow)}$. Tako se vrlo male vrijednosti CR mogu povezati s dijelom rješenja koji predstavlja aktivacijske zastavice, jer čak i male izmijene istih, zbog postojanja kazne, mogu dovesti do velikih promjena u vrijednosti funkcije cilja (ukoliko se radi o aktivnim komponentama). Prema tome, može se zaključiti da predloženi postupak samopodešavanja (isto je vrlo vjerojatno i slučaju drugih

Tablica 6.11: Rezultati za standardni DE i algoritam s predloženim postupkom samopodešavanja parametara na nekoliko odabranih standardnih testnih funkcija za $d = 500$.

f	$\text{DE}_{(\text{ASP})}$	DE
	prosjek ± std. dev.	prosjek ± std. dev.
5	**2.825E+06**∘ ± 1.41E+06	3.084E+06 ± 5.99E+05
11	**1.379E−01**↑ ± 5.37E−02	4.125E−01 ± 4.57E−02
13	**7.046E+02**↑ ± 7.24E+01	4.146E+03 ± 6.75E+02
17	**2.073E+02**↑ ± 1.45E+00	2.412E+02 ± 6.08E−01
19	**6.560E−02**↑ ± 2.36E−02	1.912E−01 ± 3.72E−02
20	**1.871E+01**↑ ± 1.43E+00	2.080E+01 ± 9.59E−01
21	1.753E+06↓ ± 1.21E+06	**9.028E+04** ± 7.03E+04
22	9.761E+06↓ ± 4.78E+06	**2.311E+06** ± 9.00E+05
—	2	↓
—	1	∘
—	5	↑

Tablica 6.12: Rezultati u smislu MSE za standardni i algoritam s predloženim postupkom samopodešavanja parametara.

$\mathcal{Q}$	$\text{DE}_{(\text{ASP})}$	DE
	prosjek ± std. dev.	prosjek ± std. dev.
1	**6.606E-03**↑ ± 7.36E-04	1.137E-02 ± 8.46E-04
2	3.518E-02↓ ± 3.71E-03	**2.754E-02** ± 3.43E-03
3	**6.246E-03**↑ ± 2.29E-03	7.912E-03 ± 1.89E-03
4	**2.860E-03**↑ ± 1.11E-03	8.990E-03 ± 1.18E-03
5	**1.171E-01**∘ ± 3.83E-03	1.200E-01 ± 7.08E-03
6	**8.518E-03**↑ ± 1.88E-03	1.503E-02 ± 4.51E-03
7	**8.110E-03**∘ ± 1.66E-03	9.059E-03 ± 4.11E-03
8	2.596E-03∘ ± 8.78E-04	**2.255E-03** ± 7.50E-04
—	1	↓
—	3	∘
—	4	↑

Tablica 6.13: Rezultati u smislu složenosti mreža za standardni i algoritam s predloženim postupkom samopodešavanja parametara.

$\mathcal{Q}$	$\text{DE}_{(\text{ASP})}$	DE
	prosjek ± std. dev.	prosjek ± std. dev.
1	20.00 ± 0.00	**19.80** ± 0.40
2	20.00 ± 0.00	20.00 ± 0.00
3	**18.04** ± 1.59	19.00 ± 0.98
4	20.00 ± 0.00	**19.60** ± 0.49
5	**19.64** ± 0.48	19.96 ± 0.20
6	19.92 ± 0.27	**19.84** ± 0.37
7	19.96 ± 0.20	**18.84** ± 1.89
8	**19.88** ± 0.32	20.00 ± 0.00

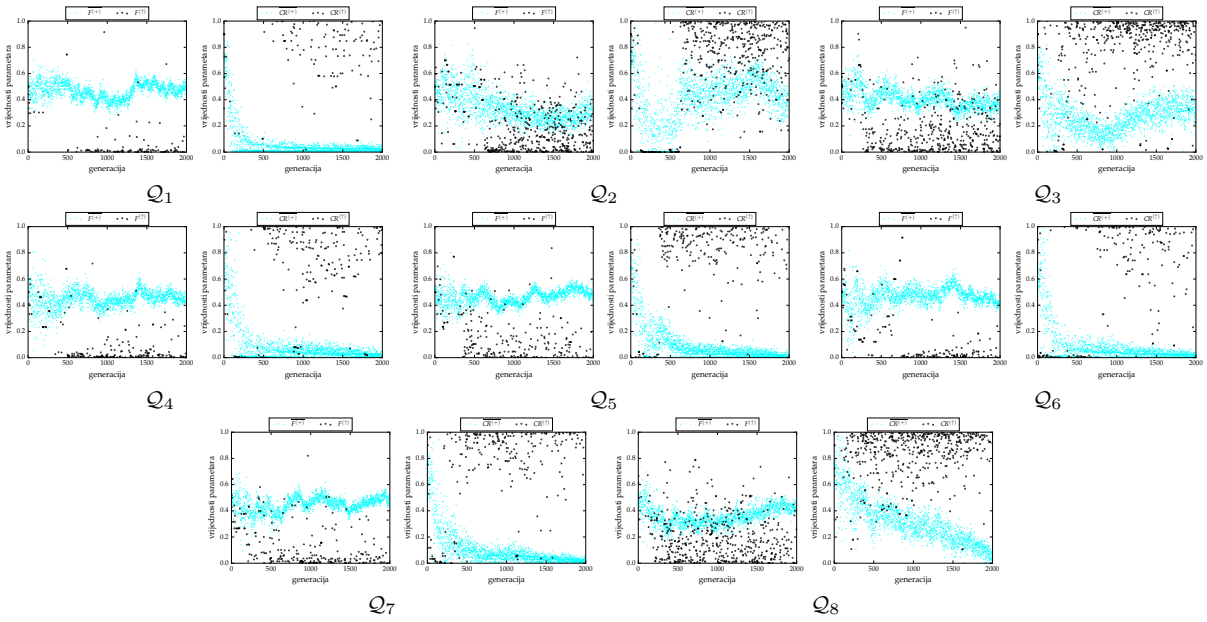
korištenih postupaka) prilagođava parametre upravo dijelu rješenja koji su zastavice. Ovo se odnosi na parametar stope križanja i, kao što sugerira slika, faktor skaliranja ne igra važnu ulogu, jer su $\overline{F^{(+)}}$ uvijek blizu vrijednosti 0.5. Isto tako, važnu ulogu igraju i neaktivne komponente čijom zamjenom pri križanju neće doći od izmjene vrijednosti funkcije cilja što je vjerojatnije uz manje vrijednosti parametra *CR*. Nadalje, ostvareni rezultati u slučaju kada funkcija cilja ne uključuje kaznu [pretraga vođenja izrazom (6.7)] podupiru prethodnu tvrdnju. Kao što se može vidjeti iz tablice 6.12, algoritam $\text{DE}_{(\text{ASP})}$ je samo na dva problema ostvario slabiji rezultat (veći prosječni MSE) u odnosu na standardni algoritam. Važno je uvidjeti složenosti tako pronađenih mreža, prikazanih tablicom 6.13, koje su praktički uvijek na granici dozvoljenog ($c_{max} = 20$) bez obzira na strukturu skupa podataka. Ovo uvelike povećava računalnu složenost procesa treniranja. Ilustracije radi, jedno izvođenje standardnog algoritma DE na skupu $\mathcal{Q}_5$ sa i bez kazne okvirno je trajalo 100 i 1140 sekundi, redom. Dodatno opravdanje za uporabu kazne je što, prema tablici 6.14, njeno izostavljanje ne donosi ni korisne prednosti u smislu MCR (vidi rezultate iz tablice 6.5 za usporedbu). Ovo podupire ranije navedenu težnju o izgradnji mreža bez suvišnih čvorova u skrivenom sloju.

Tablica 6.14: Rezultati u smislu MCR za standardni i algoritam s predloženim postupkom samopodešavanja parametara.

$\mathcal{Q}$	$\text{DE}_{(\text{ASP})}$		DE	
	$\text{MCR}(\mathcal{Q}_T)$	$\text{MCR}(\mathcal{Q}_V)$	$\text{MCR}(\mathcal{Q}_T)$	$\text{MCR}(\mathcal{Q}_V)$
1	**0.00138**	**0.002**	0.00142	0.0021
2	0.099	0.327	**0.076**	**0.313**
3	0.004	0.078	**0.004**	**0.063**
4	**0.000**	0.043	0.002	**0.035**
5	**0.161**	0.255	0.167	**0.243**
6	**0.003**	0.086	0.007	**0.066**
7	**0.003**	0.108	0.005	**0.088**
8	0.000	0.045	0.000	**0.032**

Kao što je pokazano u poglavlju 4, predložena metoda inicijalizacije populacije može stvoriti početne populacije koje sadrže relativno dobra rješenja i u slučaju problema viših dimenzionalnosti. Ta prednost posebno je korisna kada je vrednovanje funkcije cilja računalno zahtjevno ili kada broj vrednovanja ograničen. Ovdje razmatrani problemi pak poprimaju i značajno veće dimenzionalnosti, nego što su razmatrane ranije u poglavlju 4. S ciljem daljnjeg ispitivanja korisnosti predložene metode, provedeni su dodatno testiranje i analiza. U tu svrhu je $\text{NFEs}_{\max}$ smanjen na 10^4, a osim toga, u usporedbu je uključena metoda inicijalizacije predložena u [105] koja se koristi znanjem o problemu i koja je ranije sažeto opisana (korištena u algoritmu DE_{PI}).

Ostvareni rezultati u smislu postignutih MSE i složenosti mreža dani su tablicom 6.15 i 6.16, redom. Uz to, slika 6.10 prikazuje medijane izvođenja. Zanimljivo je primijetiti na slici da je algoritam DE_{PI} samo u dva slučaja (na skupovima $\mathcal{Q}_2$ i $\mathcal{Q}_7$) vidljivo učinkovitiji u



Slika 6.9: Proces podešavanja faktora skaliranja i stope križanja predloženim postupkom pri izgradnji RBFNs.

Tablica 6.15: Rezultati u smislu MSE za algoritme s različitim metodama inicijalizacije populacije.

$\mathcal{Q}$	DE_C	DE_{PI}	DE
	prosjek ± std. dev.	prosjek ± std. dev.	prosjek ± std. dev.
1	2.438E−02∘ ± 1.85E−03	**2.404E−02**∘ ± 1.81E−03	2.411E−02 ± 1.42E−03
2	6.690E−02∘ ± 1.97E−03	**6.204E−02**↑ ± 1.44E−03	6.786E−02 ± 1.66E−03
3	**5.833E−02**↑ ± 3.97E−03	5.858E−02↑ ± 5.75E−03	6.169E−02 ± 6.17E−03
4	**2.011E−02**∘ ± 2.00E−03	2.235E−02↓ ± 2.58E−03	2.014E−02 ± 2.33E−03
5	**1.581E−01**∘ ± 2.88E−03	1.583E−01∘ ± 2.28E−03	1.593E−01 ± 2.47E−03
6	**2.996E−02**∘ ± 1.75E−03	3.008E−02∘ ± 2.20E−03	3.063E−02 ± 1.74E−03
7	2.679E−02∘ ± 1.93E−03	**2.097E−02**↑ ± 2.58E−03	2.720E−02 ± 1.96E−03
8	**2.563E−02**↑ ± 2.06E−03	2.737E−02∘ ± 2.10E−03	2.753E−02 ± 2.46E−03
—	0	1	↓
—	6	4	∘
—	2	3	↑

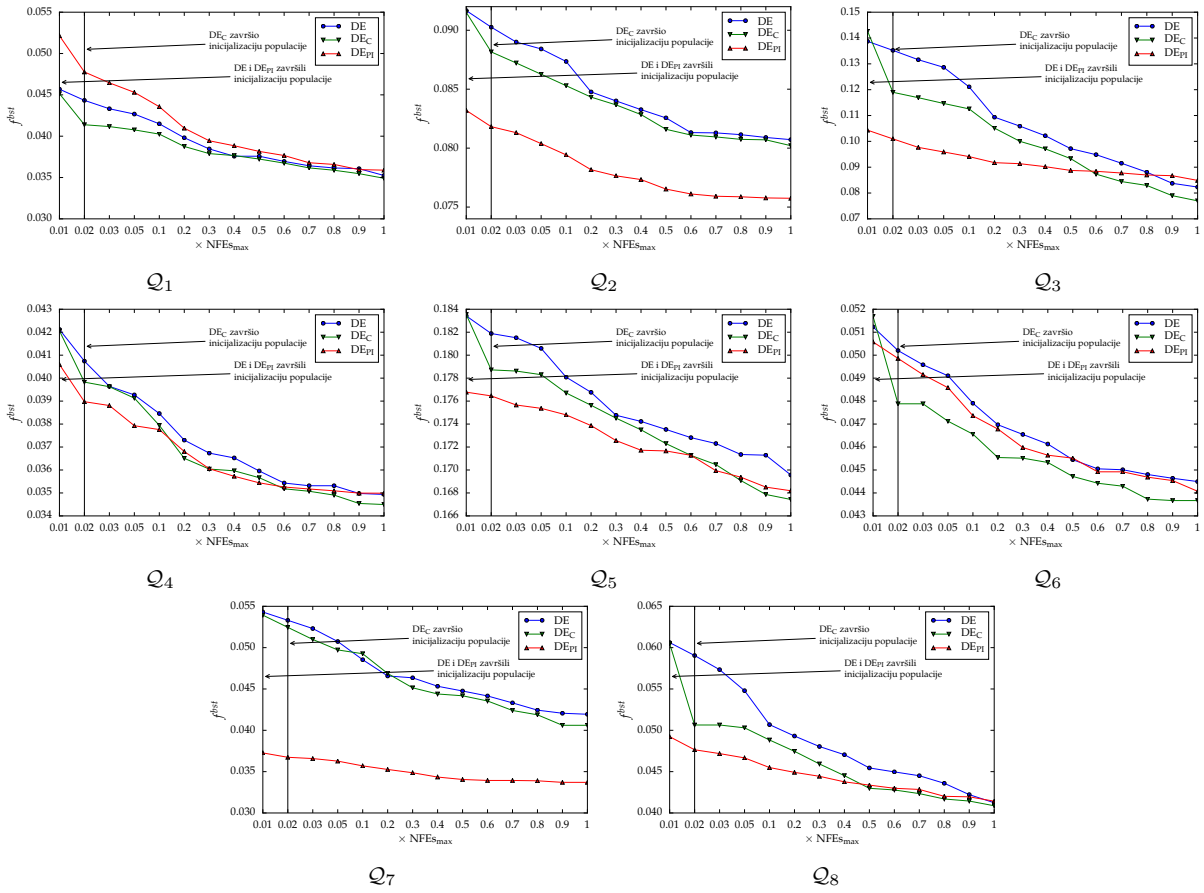
Tablica 6.16: Rezultati u smislu složenosti mreža za algoritme s različitim metodama inicijalizacije populacije.

$\mathcal{Q}$	DE_C	DE_{PI}	DE
	prosjek ± std. dev.	prosjek ± std. dev.	prosjek ± std. dev.
1	**4.20** ± 0.75	4.76 ± 0.71	4.52 ± 0.70
2	16.00 ± 1.47	16.00 ± 1.20	**15.36** ± 1.62
3	**7.28** ± 1.73	9.96 ± 1.84	8.56 ± 1.77
4	8.68 ± 1.16	**7.40** ± 1.30	8.84 ± 1.35
5	**3.64** ± 1.26	3.92 ± 0.89	4.12 ± 1.31
6	**7.92** ± 0.98	8.48 ± 1.30	8.20 ± 1.02
7	**7.28** ± 2.01	7.60 ± 1.57	8.68 ± 1.43
8	8.40 ± 1.70	8.48 ± 1.36	**8.04** ± 1.75

odnosu na standardni (koristi nasumični pristup inicijalizaciji populacije) i isti s ugrađenom predloženom metodom inicijalizacije populacije, iako u većini slučajeva ima osjetnu bolju polaznu poziciju. S druge strane, predložena metoda inicijalizacije uspijeva redovito stvoriti značajno bolje populacije u odnosu na nasumični pristup iako se ne koristi znanjem o problemu, a ta prednost uglavnom se održava do kraja izvođenja. U prilog algoritmu DE_C idu i pronađena rješenja koja često predstavljaju mreže manjih složenosti u smislu broja čvorova u skrivenom sloju.

6.4 Osvrt na RBFNs i njihovu izgradnju pomoću DE

Struktura RBFNs opisana je nizom parametara koje je nužno odrediti i postaviti prema problemu kojeg se rješava. Jedan od problema kojem se često pristupa s RBFNs je izgradnja klasifikacijskih modela. U tom pogledu, složenost mreže i prema tome broj parametara izravno ovisi o skupu uzoraka za koji se izgrađuje model. Tražnje prikladnog broja čvorova skrivenog sloja i vrijednosti parametara koji opisuju mrežu predstavlja iznimno zahtjevan problem globalne optimizacije. Rezultati provedene eksperimentalne analize sugeriraju da standardni algoritam DE predstavlja valjan izbor za izgradnju klasifikacijskih modela pomoću RBFNs. Usporedive ili bolje rezultate je ostvario u odnosu na nekoliko klasičnih i



Slika 6.10: Grafovi konvergencije za algoritme s različitim metodama inicijalizacije populacije.

prirodom inspiriranih pristupa. Iako se algoritam PSO pokazao blago boljim (drukčije vrijednosti parametara DE vrlo vjerojatno bi mogle dovesti do manjih razlika), algoritmu DE u prilog idu mreže manjih složenosti (u smislu broja čvorova u skrivenom sloju). Osim toga, praćenjem i ispitivanjem ponašanja populacije algoritma DE dobiveni su povoljni rezultati koji općenito ne ukazuju na pojave stagnacije pretrage.

Brzina konvergencije standardnog algoritma, koja u nekim slučajevima zaostaje za onom koji ispoljava PSO, značajno je unaprjeđenja ugradnjom predložene mutacije. Takvo povećanje brzine rezultiralo je redovito i kvalitetnijim rješenjima, odnosno mrežama bolje prilagođenim uzorcima iz skupa za treniranje. Prednost, posebno kada je broj vrednovanja ograničen, može pružiti i predložena metoda inicijalizacije populacije. Osim što se pokazala boljim izborom u odnosu na nasumični pristup inicijalizaciji, pokazala se pouzdanijom u odnosu na metodu iz literature koja iskorištava znanje o problemu. S druge strane, podešavanje parametara tijekom izvođenja algoritma nije preporučljivo kada funkcija cilja sadrži član kazne, jer dolazi do prilagodbe koja rezultira vrlo malim izmjenama rješenja. U slučaju kada nema takvog člana, podešavanje parametara može biti korisno, ali kazna predstavlja ključan čimbenik za održavanje ravnoteže između složenosti i kvalitete pronađenih mreža.

Eksperimentalno vrednovanje i ispitivanje ponašanja standardnog algoritma DE pri izgradnji RBFNs te ispitivanje učinka na isti algoritam koji ima ugradnja unaprjeđenja predloženih u prethodnim poglavljima, predstavljaju ispunjenje prijedloga četvrtog izvornog znanstvenog doprinosa ove disertacije.

POGLAVLJE 7

Zaključak i budući rad

Rješavanje složenih problema numeričke optimizacije, posebno ukoliko ne postoji znanje o njima (unutrašnjost modela nije dostupna – nalik crnim kutijama), predstavlja izniman izazov. Algoritam DE nudi se pri tom kao koristan alat. Iako je sam po sebi često učinkovit, prostora za njegova unaprjeđenja ne nedostaje, kao što je pokazano predloženim unaprjeđenjima. Ona su uz to iskazala dobru učinkovitost u usporedbi s nekim srodim unaprjeđenjima iz literature. Isto tako, prostora ima za daljnje dorade predloženih unaprjeđenja, koje bi dodatno mogle doprinijeti njihovoj učinkovitosti i otpornosti na različita svojstva problema.

7.1 Zaključci

Problemi optimizacije koji se sve češće pojavljuju u inženjerstvu i znanosti postaju sve složeniji i time zahtjevniji za rješavanje. Štoviše, detalji modela, odnosno analitički oblik funkcije cilja često nije poznat ili dostupan. U takvim slučajevima, moguće je samo dobiti odziv na dani ulazni niz parametara koji opisuje problem. Prema tome, njihovo rješavanje zahtijeva metode optimizacije kojima osim navedenog, nije ništa potrebno kako bi istraživale prostor pretrage. Kao mogući pristup rješavanju nude se raznoliki EAs, a ukoliko se radi o problemima numeričke optimizacije, može se posebno istaknuti DE zbog svoje (relativne) jednostavnosti, ali i dokazane učinkovitosti. Algoritam DE je i u središtu istraživanja ove disertacije u kojoj su predložena tri njegova unaprjeđenja.

Osjetljivost na postavke parametara je nedostatak svih prirodom inspiriranih algoritama optimizacije (i mnogih drukčijih), a ni DE nije izuzetak u tom smislu. Traženje prikladnih vrijednosti zahtijeva posebna i izdvojena izvođenja algoritma prije pristupanja konkretnom rješavanju problema. Osim što prikladne vrijednosti ovise o problemu kojeg se rješava, održavanje učinkovitog istraživanja prostora pretrage može zahtijevati njihove stalne izmjene.

U disertaciji je predložen postupak samopodešavanja faktora skaliranja i stope križanja koji pokušava riješiti potonji problem čime se implicitno rješava i prvi. Postupak se koristi prethodno uspješnim vrijednostima parametara za generiranje novih. Svakom vektoru populacije pridružene su zasebne vrijednosti parametara F i CR koje se koriste prilikom mutacije i križanja. Dodatno su im dodijeljeni popisi od $r = 3$ prethodno uspješnih u čijoj okolini se traže nove. U popisima se održava zadnjih r uspješnih vrijednosti parametara F i CR. Rezultati ostvareni u eksperimentalnoj analizi ukazuju da predloženi postupak uspijeva podesiti navedene parametre, jer su najčešće postignuti bolji rezultati u odnosu na različite fiksne postavke istih. Prema tome, on omogućuje izbjegavanje posebnih troškova vremena računanja na traženje dobrih postavki parametara. S obzirom na brojne postupke podešavanja parametara, predloženi postupak uspoređen je s tri različita postupaka za podešavanje faktora skaliranja i stope križanja iz literature. Povoljni rezultati su postignuti i u toj usporedbi, gdje je također pokazano da je predloženi postupak po vremenskoj složenosti blizak ostalima, što može opravdavati njegovu ugradnju i uporabu. Iako je broj pohranjenih prethodno uspješnih parametara r zadana i fiksan, on utječe na ponašanje postupka. Međutim, predloženi postupak otporan je na manje varijacije te veličine, a osjetno veće vrijednosti su rijetko korisne kao što sugerira provedena analiza.

Osim što je nužno postaviti parametre algoritma, nužno je odabrati i početnu populaciju prije izvođenja. Najčešće korištena metoda u tu svrhu je nasumična inicijalizacija populacije. Dobra raznolikost i rješenja loše kvalitete neka su od svojstava takvih populacija. Međutim, uvođenje relativno dobrih rješenja u početnu populaciju može doprinijeti učinkovitosti algoritma u smislu bolje polazne pozicije za daljnje istraživanje i time brzine konvergencije, ali to nije jednostavno ostvarivo ako znanje o problemu nije dostupno (problem nalik crnoj kutiji). U disertaciji je predložena metoda inicijalizacije populacije koja pokušava pronaći, odnosno približiti se obećavajućim dijelovima ili regijama prostora pretrage. U tu svrhu, koristi se grupiranjem zadane nasumične populacije i potom, jednostavnom Cauchyjevom mutacijom za daljnje istraživanje okoline predstavnika dobivenih grupa. Opsežna usporedba s uobičajenim (nasumičnim) pristupom inicijalizaciji populacije te s dvije metode za inicijalizaciju iz literature, pokazala je da predložena metoda uspijeva stvoriti početne populacije koje sadrže relativno dobra rješenja, bez obzira na dozvoljeni budžet za njihovo stvaranje. Prednost predložene metode u odnosu na druge upotrijebljene u eksperimentalnoj analizi, posebno je izražena u slučaju funkcija koje nisu male dimenzionalnosti ($d > 10$). Iako je složenija u odnosu na ostale metode upotrijebljene u usporedbi, to je nadoknađeno u velikoj većini slučajeva postizanjem ciljane greške optimizacije kroz manje vrednovanja funkcije cilja. To se često ogledalo i u ukupno manjim vremenima izvođenja. Na koncu, testiranje i analiza na problemima viših dimenzionalnosti ($d = 80$ i 120) sugeriraju da predložena metoda može biti korisna i u takvim slučajevima, pružajući dobru polaznu točku za daljnje istraživanje prostora pretrage.

Ključan element razlike između DE i drugih uobičajenih EAs ujedno je ključan čimbenik njene učinkovitosti. Mutacija je izravno odgovorna za sposobnost algoritma da obavlja istraživanje prostora pretrage i iskorištavanje obećavajućih točka istog. Ona odabrani bazni vektor perturbira razlikom drugih vektora populacije. Stoga, selekcijski pritisak pri odabiru baznog vektora uvelike utječe na mogućnost istraživanja i iskorištavanja. Za učinkovitu pretragu nužno je uravnotežiti oba vida pretrage. U disertaciji je predložena mutacija DE u kojoj se prilagođava selekcijski pritisak odabira baznog vektora. To postiže uporabom k-turnirske selekcije čija se veličina za svaki vektor posebno prilagođava. Prilagodba je vođena temeljnom veličinom za sve turnire koja se tijekom pretrage povećava, a time i selekcijski pritisak. Uvođenje selekcijskog pritiska u odabir baznog vektora k-turnirskom selekcijom rezultiralo je značajnim povećanjem učinkovitosti u odnosu na često korištene mutacije rand/1 i best/1. Navedeno se pokazalo kako pri fiksnoj i jednakoj veličini turnira za sve članove populacije, tako i pri prilagodbi veličina turnira za svaki posebno. Prednosti prilagodbe su izbjegavanje potrebe za određivanjem prikladne veličine turnira te prijelaz s, primarno, istraživanje na iskorištavanje što se pokazalo važnim u slučaju najsloženijih korištenih instanci problema. Osim u odnosu na često korištene standardne mutacije, predložena mutacija je u odnosu na četiri mutacije, iz literature, s drukčijim pristupima određivanju baznog vektora postigla većinom bolje ili usporedive rezultate uz usporedivu razinu vremenske složenosti. Prilagodbe veličine turnira odvija se unutar $[1, k_{max}]$, gdje je $k_{max} = 10$ zadana i fiksna gornja granica. Eksperimentalna analiza u tom pogledu ukazala je da razlike u učinkovitosti uglavnom nisu primjetne pri manjim razlikama k_{max} u odnosu na zadanu.

Izgradnja, odnosno traženje parametara RBFNs koji će rezultirati odgovarajućim modelom klasifikacije (ili modelom za druge potrebe), predstavlja složen problem globalne optimizacije, odnosno numeričke optimizacije, jer su parametri mreže realni. Mreže su opisane parametrima radijalnih funkcija kao čvorova skrivenog sloja te težina veza s izlaznim slojem. Svojstva mreža ne zavise samo o vrijednostima tih parametara, nego i o broju čvorova skrivenog sloja koji često nije poznat. Traženje broja čvorova u skrivenom sloju dodatno i značajno doprinosi složenosti problema. U disertaciji su prvo ispitani ponašanje i učinkovitost standardnog algoritma DE pri izgradnji RBFNs. Usporedba s nekoliko klasičnih i drukčijih prirodom inspiriranih pristupa tražnja parametara sugerira kako je standardni algoritma više nego konkurentan te predstavlja valjan izbor za rješavanje navedenog problema. U nekoliko slučajeva lošiji rezultati u odnosu na PSO, redovito su nadoknađeni mrežama manje složenosti u smislu broja čvorova u skrivenom sloju. Značajno povećanje učinkovitosti ostvareno je korištenjem predložene mutacije kojom je očekivano, pospješena brzina konvergencije. Iskorištavanje dobrih pronađenih rješenja važno je s obzirom na dimenzionalnost problema i zahtjevno vrednovanje rješenja pa je poželjno dostići zadovoljavajuća svojstva mreža što prije moguće. Pozitivan utjecaj na učinkovitost standardnog algoritma DE postignut je i ugradnjom predložene metode inicijalizacije populacije, a posebno u slučaju ograničenog

broja vrednovanja. U tom pogledu, prednost dobivena inicijalizacijom najčešće je očuvana do kraja. Isprva iznenađujuće, negativan utjecaj imala je ugradnja predloženog postupka samopodešavanja faktora skaliranja i stope križanja, ali isto tako i drugih prethodno korištenih postupaka podešavanja tih parametara. Daljim istraživanjem ustanovljen je razlog tomu. Naime, podešavanje teži parametrima koji će prouzročiti što manje izmjene vektora populacije, jer i male promjene mogu izazvati osjetno narušavanje kvalitete (prvenstveno zbog člana kazne u funkciji cilja). Iako uklanjanjem člana kazne podešavanje parametara može pospješiti učinkovitost, kazna je nužna za ostvarivanje relativno malih mreža.

Na kraju, predložena unaprjeđenja, kao što sugeriraju predstavljeni i analizirani rezultati, pružaju osjetno povećanje učinkovitosti pri rješavanju problema numeričke optimizacije u odnosu na standardni algoritam DE. Osim u odnosu na standardni algoritam postignuti su bolji ili usporedivi rezultati i u odnosu na isti algoritam u koji su ugrađenja srodna unaprjeđenja iz literature. Uz to, predložena unaprjeđenja pokazala su se od značajne koristi pri izgradnji RBFNs kao klasifikacijskih modela. Na temelju navedenog, može se izvesti zaključak da su ciljevi disertacije i time očekivani izvorni znanstveni doprinosi ispunjeni.

7.2 Budući rad

Kao što sugeriraju predstavljeni rezultati dobiveni u eksperimentalnim analizama, ugradnja i uporaba predloženih unaprjeđenja algoritam DE općenito pruža povećanje učinkovitosti i vrlo su konkurentna u odnosu na srodna unaprjeđenjima iz literature. Međutim, prostor za daljnje dorade postoji. Osim toga, prostor za daljnje pospješivanje učinkovitosti pružaju i njihove moguće kombinacije.

Svakom vektoru populacije pri predloženom postupku samopodešavanja faktora skaliranja i stope križanja dodijeljeni su popisi $\mathcal{A}_j^F$ i $\mathcal{A}_j^{CR}$ koji održavaju prethodno uspješne vrijednosti tih parametara. S obzirom da se računaju težinske srednje vrijednosti elemenata tih popisa, u čijoj okolini se potom generiraju nove vrijednosti parametara, manje njihove veličine omogućuje brže promjene i obratno. Kao što sugerira eksperimentalna analiza, u određenim situacijama moguće je postići bolje rezultate s blago manjim odnosno većim veličinama popisa u odnosu na zadanu $r = 3$. Prema tome, podešavanje te veličine tijekom izvođenja predstavlja prostor za budući rad. Linearno i postupno povećanje ili smanjenje veličine r predstavljaju dva jednostavna pristupa u tom pogledu. Ugradnja predloženog postupka (kao i velike većine postupaka dostupnih u literaturi) uklanja potrebu za traženjem parametara F i CR, dok veličina populacije NP ostaje korisnički definiran parametar. Iako je pretpostavka da postupak može prilagoditi vrijednosti navedenih parametara zadanoj veličini populacije, zanimljivu smjernicu za budući rad predstavlja i uključivanje nekog mehanizma za podešavanje parametra NP, a nekolicina ih je dostupna u literaturi (vidi primjerice [14, 126, 139]).

Kao što je ranije istaknuto za predloženu metodu inicijalizacije populacije te je napravljeno za potrebe izgradnje RBFNs, parametar skaliranja Cauchyjeve razdiobe nužno je postaviti prema rasponima parametara koji opisuju dani problem. Zadana vrijednost $s = 1$ nije uvijek prikladna, posebno u slučaju vrlo malih raspona (kao što je bio slučaj pri izgradnji RBFNs zbog noramliziranih skupova uzoraka), gdje se praktički dobiva učinak kao pri uporabi uniformne razdiobe. U tom pogledu, kao budući rad nameće se automatsko određivanje parametra skaliranja Cauchyjeve razdiobe koji izravno određuje intenzitet mutacije kojom se generiraju nova rješenja nakon obavljenog grupiranja. Osim za inicijalizaciju populacije, predložena metoda može eventualno poslužiti i tijekom pretrage kao poseban mehanizam za generiranje novih rješenja, slično kao u [21, 112]. Pri tom, važno je odrediti koja i koliko bi tih novih rješenja bilo uključeno u trenutnu populaciju, jer bi veliki broj mogao brzo narušiti raznolikost populacije i time prouzročiti preuranjenu konvergenciju. Nadalje, iako je korištena u DE, predložena metoda nije ograničena na nju pa testiranje i analiza njene ugradnje na učinkovitost nekih drugih EAs može predstavljati smjernicu za daljnje ispitivanje korisnosti i općenitosti metode.

Veličine turnira u predloženoj mutaciji ograničene su s $[1, k_{max}]$, gdje je $k_{max} = 10$ zadan kao gornja granica. Međutim, u slučaju vrlo malih ili pak vrlo velikih populacija može se biti korisno ili potrebno k_{max} prilagoditi istima. Stoga se kao budući rad nudi automatsko određivanje gornje granice za veličine turnira. Jednostavan pristup predstavlja postavljenje gornje granice kao $k_{max} = NP/10$ što odgovara slučaju u eksperimentalnoj analizi (zbog korištenja $NP = 100$). Osim navedenog, treba razviti te ispiti i alternativne pristupe kao što je primjerice, dinamičko podešavanje vrijednosti k_{max}. Nadalje, osim uvođenja selekcijskog pritiska u odabir baznog vektora, korisnim se može pokazati i isto pri odabiru terminalnog vektora (prvog u razlici za perturbaciju). Slično kao u [108], pruža se mogućnost provođenja posebnog turnira jednake veličine u tu svrhu. Međutim, uporabom manjih turnira manje bi se narušio broj mogućih vektora razlike (perturbacija) i time sposobnost istraživanja još neotkrivenih dijelova prostora pretrage. Još jednu mogućnost za odabir terminalnog vektora predstavlja i uzimanje najboljeg gubitnika iz turnira za odabir baznog vektora.

Predložena unaprjeđenja su testirana i njihova učinkovitost te ponašanje analizirani su odvojeno. Međutim, ne isključuju jedno drugo i u konačnici ih je moguće zajedno koristiti u istom algoritmu. Shodno tome, kao budući rad nameće se razmatranje njihovih kombinacija. Iako pojedinačno pružaju povećanje učinkovitosti, teško je predvidjeti ponašanje njihovih kombinacija. U tom smislu, potrebna su opsežna testiranja i eventualna usklađivanja. Vodeći se literaturom, postupci podešavanja parametara i unaprjeđenih mutacija često predstavljaju uspješne kombinacije (vidi primjerice [54, 125, 152, 162]). Prema tome, kao obećavajuću kombinaciju može se istaknuti predloženi postupak samopodešavanja faktora skaliranja i stope križanja zajedno s predloženom mutacijom.

Literatura

[1] Q. Abbas, J. Ahmad i H. Jabeen. A novel tournament selection based differential evolution variant for continuous optimization problems. *Mathematical Problems in Engineering*, 2015:21 stranica, 2015.

[2] H. A. Abbass. The self-adaptive pareto differential evolution algorithm. *Proceedings of the 2002 Congress on Evolutionary Computation (CEC)*, svezak 1, stranice 831–836, 2002.

[3] M. Ali, M. Pant i A. Abraham. Unconventional initialization methods for differential evolution. *Applied Mathematics and Computation*, 219(9):4474–4494, 2013.

[4] M. M. Ali, C. Khompatraporn i Z. B. Zabinsky. A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems. *Journal of Global Optimization*, 31(4):635–672, 2005.

[5] M. W. Ayech i D. Ziou. Segmentation of terahertz imaging using k-means clustering based on ranked set sampling. *Expert Systems with Applications*, 42(6):2959–2974, 2015.

[6] K. Bache i M. Lichman. UCI machine learning repository, 2013. `http://archive.ics.uci.edu/ml`.

[7] D. Bajer i G. Martinović. A parameter control scheme for DE inspired by ACO. *Proceedings of the 7th International Conference on Bioinspired Optimization Methods and their Applications (BIOMA)*, stranice 79–92, 2016.

[8] D. Bajer, B. Zorić i G. Martinović. Automatic design of radial basis function networks through enhanced differential evolution. *Proceedings of the 10th International Conference on Hybrid Artificial Intelligent Systems (HAIS)*, stranice 244–256, 2015.

[9] D. Bajer, G. Martinović i J. Brest. A population initialization method for evolutionary algorithms based on clustering and cauchy deviates. *Expert Systems with Applications*, 60:294–310, 2016.

[10] D. Bajer, B. Zorić i G. Martinović. Effectiveness of differential evolution in training radial basis function networks for classification. *Proceedings of the 1st International Conference on Smart Systems and Technologies (SST)*, stranice 179–184, 2016.

[11] J. C. Bansal i H. Sharma. Cognitive learning in differential evolution and its application to model order reduction problem for single-input single-output systems. *Memetic Computing*, 4(3):209–229, 2012.

[12] N. Benoudjit i M. Verleysen. On the kernel widths in radial-basis function networks. *Neural Processing Letters*, 18(2):139–154, 2003.

[13] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, Inc., 1995.

[14] J. Brest i M. S. Maučec. Self-adaptive differential evolution algorithm using population size reduction and three strategies. *Soft Computing*, 15(11):2157–2174, 2011.

[15] J. Brest, S. Greiner, B. Boškovic, M. Mernik i V. Žumer. Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. *IEEE Transactions on Evolutionary Computation*, 10(6):646–657, 2006.

[16] J. Brest, B. Bošković, S. Greiner, V. Žumer i M. S. Maučec. Performance comparison of self-adaptive and adaptive differential evolution algorithms. *Soft Computing*, 11(7): 617–629, 2007.

[17] J. Brest, A. Zamuda, B. Bošković, S. Greiner i V. Žumer. An analysis of the control parameters' adaptation in DE. *Advances in Differential Evolution*, stranice 89–110. Springer Berlin Heidelberg, 2008.

[18] D. S. Broomhead i D. Lowe. Multivariable functional interpolation and adaptive networks. *Complex Systems*, 2(3):321–355, 1988.

[19] G. Bugmann. Normalized gaussian radial basis function networks. *Neurocomputing*, 20(1-3):97–110, 1998.

[20] Y. Cai i J. Wang. Differential evolution with neighborhood and direction information for numerical optimization. *IEEE Transactions on Cybernetics*, 43(6):2202–2215, 2013.

[21] Z. Cai, W. Gong, C. X. Ling i H. Zhang. A clustering-based differential evolution for global optimization. *Applied Soft Computing*, 11(1):1363–1379, 2011.

[22] P. Civicioglu i E. Besdok. A conceptual comparison of the cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms. *Artificial Intelligence Review*, 39(4):315–346, 2013.

[23] A. R. Conn, K. Scheinberg i L. N. Vicente. *Introduction to Derivative-Free Optimization*. Society for Industrial and Applied Mathematics, 2009.

[24] S. Das i P. N. Suganthan. Differential evolution: A survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation*, 15(1):4–31, 2011.

[25] S. Das, A. Konar i U. K. Chakraborty. Two improved differential evolution schemes for faster global search. *Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation (GECCO)*, stranice 991–998, 2005.

[26] S. Das, S. S. Mullick i P. N. Suganthan. Recent advances in differential evolution – an updated survey. *Swarm and Evolutionary Computation*, 27:1–30, 2016.

[27] V. V. de Melo i A. C. B. Delbem. Investigating smart sampling as a population initialization method for differential evolution in continuous problems. *Information Sciences*, 193:36–53, 2012.

[28] A. Deb, J. S. Roy i B. Gupta. Performance comparison of differential evolution, particle swarm optimization and genetic algorithm in the design of circularly polarized microstrip antennas. *IEEE Transactions on Antennas and Propagation*, 62(8):3920–3928, 2014.

[29] F. M. Dekking, C. Kraaikamp, H. P. Lopuhaä i L. E. Meester. *A Modern Introduction to Probability and Statistics, Understanding Why and How*. Springer-Verlag London, 2005.

[30] J. Derrac, S. García, D. Molina i F. Herrera. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, 1(1):3–18, 2011.

[31] N. Dong, C.-H. Wu, W.-H. Ip, Z.-Q. Chen, C.-Y. Chan i K.-L. Yung. An opposition-based chaotic GA/PSO hybrid algorithm and its application in circle detection. *Computers & Mathematics with Applications*, 64(6):1886–1902, 2012.

[32] R. O. Duda, P. E. Hart i D. G. Stork. *Pattern Classification*. Wiley-Interscience, 2000.

[33] A. E. Eiben i S. K. Smit. Parameter tuning for configuring and analyzing evolutionary algorithms. *Swarm and Evolutionary Computation*, 1(1):19–31, 2011.

[34] A. E. Eiben i J. E. Smith. *Introduction to Evolutionary Computing*. Springer-Verlag Berlin Heidelberg, 2015.

[35] A. E. Eiben, R. Hinterding i Z. Michalewicz. Parameter control in evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 3(2):124–141, 1999.

[36] A. P. Engelbrecht. *Computational Intelligence: An Introduction, Second Edition*. Wiley Publishing, 2007.

[37] M. G. Epitropakis, V. P. Plagianakos i M. N. Vrahatis. Evolutionary adaptation of the differential evolution control parameters. *Proceedings of the 2009 IEEE Congress on Evolutionary Computation (CEC)*, stranice 1359–1366, 2009.

[38] M. G. Epitropakis, D. K. Tasoulis, N. G. Pavlidis, V. P. Plagianakos i M. N. Vrahatis. Enhancing differential evolution utilizing proximity-based mutation operators. *IEEE Transactions on Evolutionary Computation*, 15(1):99–119, 2011.

[39] I. Fajfar, J. Puhan, S. Tomažič i A. Bűrmen. On selection in differential evolution. *Elektrotehniški Vestnik*, 78(5):275–280, 2011.

[40] H.-Y. Fan i J. Lampinen. A trigonometric mutation operation to differential evolution. *Journal of Global Optimization*, 27(1):105–129, 2003.

[41] V. Feoktistov. *Differential Evolution: In Search of Solutions*. Springer-Verlag New York, Inc., 2006.

[42] S. Ferrari i R. F. Stengel. Smooth function approximation using neural networks. *IEEE Transactions on Neural Networks*, 16(1):24–38, 2005.

[43] V. Filipović. Fine-grained tournament selection operator in genetic algorithms. *Computing and Informatics*, 22(2):143–161, 2003.

[44] R. Gämperle, S. D. Müller i P. Koumoutsakos. A parameter study for differential evolution. *Proceedings of the WSEAS International Conference on Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation*, stranice 293–298, 2002.

[45] W. Gao, S. Liu i L. Huang. Particle swarm optimization with chaotic opposition-based population initialization and stochastic search technique. *Communications in Nonlinear Science and Numerical Simulation*, 17(11):4316–4327, 2012.

[46] M. Golub, D. Jakobović i L. Budin. Parallelization of elimination tournament selection without synchronization. *Proceedings of the 5th IEEE International Conference on Intelligent Engineering Systems (INES)*, stranice 85–89, 2001.

[47] W. Gong i Z. Cai. Differential evolution with ranking-based mutation operators. *IEEE Transactions on Cybernetics*, 43(6):2066–2081, 2013.

[48] W. Gong, Z. Cai, C. X. Ling i J. Du. Hybrid differential evolution based on fuzzy c-means clustering. *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation (GECCO)*, stranice 523–530, 2009.

[49] J. L. Guerrero, A. Berlanga i J. M. Molina. Initialization procedures for multiobjective evolutionary approaches to the segmentation issue. *Proceedings of the 7th International Conference on Hybrid Artificial Intelligent Systems (HAIS)*, stranice 452–463, 2012.

[50] S. M. Guo i C. C. Yang. Enhancing differential evolution utilizing eigenvector-based crossover operator. *IEEE Transactions on Evolutionary Computation*, 19(1):31–49, 2015.

[51] S. M. Guo, C. C. Yang, P. H. Hsu i J. S. H. Tsai. Improving differential evolution with a successful-parent-selecting framework. *IEEE Transactions on Evolutionary Computation*, 19(5):717–730, 2015.

[52] M. Horng, Y. Lee, M. Lee i R. Liou. Firefly meta-heuristic algorithm for training the radial basis function network for data classification and disease diagnosis. *Theory and New Applications of Swarm Intelligence*. InTech, 2012.

[53] Z. Huang i Y. Chen. An improved differential evolution algorithm based on adaptive parameter. *Journal of Control Science and Engineering*, 2013:5 stranica, 2013.

[54] S. M. Islam, S. Das, S. Ghosh, S. Roy i P. N. Suganthan. An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization. *IEEE Transactions on Systems, Man, and Cybernetics., Part B: Cybernetics*, 42(2):482–500, 2012.

[55] H. Jabeen, Z. Jalil i A. R. Baig. Opposition based initialization in particle swarm optimization (O-PSO). *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation (GECCO): Late Breaking Papers*, stranice 2047–2052, 2009.

[56] M. Jamil i X. Yang. A literature survey of benchmark functions for global optimisation problems. *International Journal of Mathematical Modelling and Numerical Optimisation*, 4(2):150–194, 2013.

[57] R. Joshi i A. C. Sanderson. Minimal representation multisensor fusion using differential evolution. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 29(1):63–76, 1999.

[58] P. Kaelo i M. M. Ali. A numerical study of some modified differential evolution algorithms. *European Journal of Operational Research*, 169(3):1176–1184, 2006.

[59] G. Karafotias, M. Hoogendoorn i A. E. Eiben. Parameter control in evolutionary algorithms: trends and challenges. *IEEE Transactions on Evolutionary Computation*, 19(2):167–187, 2015.

[60] B. Kazimipour, X. Li i A. K. Qin. Initialization methods for large scale global optimization. *Proceedings of the 2013 IEEE Congress on Evolutionary Computation (CEC)*, stranice 2750–2757, 2013.

[61] H.-K. Kim, J.-K. Chong, K.-Y. Park i D. A. Lowther. Differential evolution strategy for constrained global optimization and application to practical engineering problems. *IEEE Transactions on Magnetics*, 43(4):1565–1568, 2007.

[62] M. Korürek i B. Doğan. ECG beat classification using particle swarm optimization and radial basis function neural network. *Expert Systems with Applications*, 37(12): 7563–7569, 2010.

[63] R. Kruse, C. Borgelt, F. Klawonn, C. Moewes, M. Steinbrecher i P. Held. *Computational Intelligence: A Methodological Introduction*. Springer-Verlag London, 2013.

[64] T. Kurban i E. Beşdok. A comparison of RBF neural network training algorithms for inertial sensor based terrain classification. *Sensors*, 9(8):6312–6329, 2009.

[65] W. Kwedlo. A clustering method combining differential evolution with the k-means algorithm. *Pattern Recognition Letters*, 32(12):1613–1621, 2011.

[66] M. Laguna i R. Martí. Experimental testing of advanced scatter search designs for global optimization of multimodal functions. *Journal of Global Optimization*, 33(2): 235–255, 2005.

[67] J. Lampinen i I. Zelinka. On stagnation of the differential evolution algorithm. *Proceedings of the 6th Interational Conference on Soft Computing MENDEL*, stranice 76–83, 2000.

[68] R. Li, L. Xu, X. Shi, N. Zhang i Z. Lv. Improved differential evolution strategy for antenna array pattern synthesis problems. *Progress In Electromagnetics Research*, 113: 429–441, 2011.

[69] S. Li, M. Fairbank, C. Johnson, D. C. Wunsch, E. Alonso i J. L. Proao. Artificial neural networks for control of a grid-connected rectifier/inverter under disturbance, dynamic and power converter switching conditions. *IEEE Transactions on Neural Networks and Learning Systems*, 25(4):738–750, 2014.

[70] J. J. Liang, B. Qu i P. N. Suganthan. Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization. Technical Report 201311, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, Kina i Nanyang Technological University, Singapur, 2013.

[71] T. Liao, D. Molina i T. Stützle. Performance evaluation of automatically tuned continuous optimizers on different benchmark sets. *Applied Soft Computing*, 27(C):490–503, 2015.

[72] C. Lin, A. Qing i Q. Feng. A new differential mutation base generator for differential evolution. *Journal of Global Optimization*, 49(1):69–90, 2011.

[73] C. Lin, A. Qing i Q. Feng. A comparative study of crossover in differential evolution. *Journal of Heuristics*, 17(6):675–703, 2011.

[74] J. Liu i J. Lampinen. A fuzzy adaptive differential evolution algorithm. *Soft Computing*, 9(6):448–462, 2005.

[75] M. Liu, P. Huo i W. Gong. Differential evolution with $(\mu + \lambda)$-selection technique. *International Journal of Computer Science Issues*, 10(2):286–293, 2013.

[76] M. A. Lones. Metaheuristics in nature-inspired algorithms. *Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation (GECCO)*, stranice 1419–1422, 2014.

[77] H. Maaranen, K. Miettinen i A. Penttinen. On initial populations of a genetic algorithm for continuous optimization problems. *Journal of Global Optimization*, 37(3):405–436, 2007.

[78] R. Mallipeddi, P. N. Suganthan, Q. K. Pan i M. F. Tasgetiren. Differential evolution algorithm with ensemble of parameters and mutation strategies. *Applied Soft Computing*, 11(2):1679–1696, 2011.

[79] A. C. Martinez-Estudillo, C. Hervas-Martinez, F. J. Martinez-Estudillo i N. Garcia-Pedrajas. Hybridization of evolutionary algorithms and local search by means of a clustering method. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 36(3):534–545, 2006.

[80] G. Martinović i D. Bajer. The effect of swapping vectors during mutation in differential evolution. *Proceedings of the 4th International Conference on Swarm, Evolutionary, and Memetic Computing (SEMCCO)*, stranice 534–546, 2014.

[81] G. Martinović i D. Bajer. Impact of NNA implementation on GA performance for the TSP. *Proceedings of the 5th International Conference on Bioinspired Optimization Methods and their Applications (BIOMA)*, stranice 173–184, 2012.

[82] G. Martinović, D. Bajer i B. Zorić. A differential evolution approach to dimensionality reduction for classification needs. *International Journal of Applied Mathematics and Computer Science*, 24(1):111–122, 2014.

[83] M. Matsumoto i T. Nishimura. Mersenne twister: A 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation*, 8(1):3–30, 1998.

[84] M. Mernik, S.-H. Liu, D. Karaboga i M. Črepinšek. On clarifying misconceptions when comparing variants of the artificial bee colony algorithm by offering a new implementation. *Information Sciences*, 291:115–127, 2015.

[85] E. Mezura-Montes, J. Velázquez-Reyes i C. A. Coello Coello. A comparative study of differential evolution variants for global optimization. *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation (GECCO)*, stranice 485–492, 2006.

[86] M. Modiri-Delshad i N. A. Rahim. Fast initialization of population based methods for solving economic dispatch problems. *Proceedings of the 3rd IET International Conference on Clean Energy and Technology (CEAT)*, stranice 1–5, 2014.

[87] J. Moody i C. J. Darken. Learning with localized receptive fields. Technical report, Yale University, Computer Science Departmant, New Haven, Connecticut, SAD, 1988.

[88] J. Moody i C. J. Darken. Fast learning in networks of locally-tuned processing units. *Neural Computation*, 1(2):281–294, 1989.

[89] M. C. Naldi, R. J. G. B. Campello, E. R. Hruschka i A. C. P. L. F. Carvalho. Efficiency issues of evolutionary k-means. *Applied Soft Computing*, 11(2):1938–1952, 2011.

[90] F. Neri i V. Tirronen. On memetic differential evolution frameworks: A study of advantages and limitations in hybridization. *Proceedings of the 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, stranice 2135–2142, 2008.

[91] F. Neri i V. Tirronen. Recent advances in differential evolution: A survey and experimental analysis. *Artificial Intelligence Review*, 33(1-2):61–106, 2010.

[92] J. Nocedal i S. Wright. *Numerical Optimization*. Springer-Verlag, 2006.

[93] N. Noman, D. Bollegala i H. Iba. An adaptive differential evolution algorithm. *Proceedings of the 2011 IEEE Congress on Evolutionary Computation (CEC)*, stranice 2229–2236, 2011.

[94] B. O'Hora, J. Perera i A. Brabazon. Designing radial basis function networks for classification using differential evolution. *Proceedings of the 2006 IEEE International Joint Conference on Neural Networks (IJCNN)*, stranice 2932–2937, 2006.

[95] M. G. Omran, S. Al-Sharhan, A. Salman i M. Clerc. Studying the effect of using low-discrepancy sequences to initialize population-based optimization algorithms. *Computational Optimization and Applications*, 56(2):457–480, 2013.

[96] M. M. Öztürk, U. Cavusoglu i A. Zengin. A novel defect prediction method for web pages using k-means++. *Expert Systems with Applications*, 42(19):6496–6506, 2015.

[97] Q.-K. Pan, P. N. Suganthan, L. Wang, L. Gao i R. Mallipeddi. A differential evolution algorithm with self-adapting strategy and control parameters. *Computers & Operations Research*, 38(1):394–408, 2011.

[98] M. Pant, R. Thangaraj, C. Grosan i A. Abraham. Improved particle swarm optimization with low-discrepancy sequences. *Proceedings of the 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, stranice 3011–3018, 2008.

[99] P. V. Paul, N. Moganarangan, S. S. Kumar, R. Raju, T. Vengattaraman i P. Dhavachelvan. Performance analyses over population seeding techniques of the permutation-coded genetic algorithm: An empirical study based on traveling salesman problems. *Applied Soft Computing*, 32:383–402, 2015.

[100] A. P. Piotrowski. Differential evolution algorithms applied to neural network training suffer from stagnation. *Applied Soft Computing*, 21:382–406, 2014.

[101] A. P. Piotrowski. Review of differential evolution population size. *Swarm and Evolutionary Computation*, 2016. u tisku.

[102] K. Price. Differential evolution: a fast and simple numerical optimizer. *Proceedings of the Biennial Conference of the North American Fuzzy Information Processing Society (NAFIPS)*, stranice 524–527, 1996.

[103] K. Price, R. Storn i J. A. Lampinen. *Differential Evolution: A Practical Approach to Global Optimization*. Springer-Verlag New York, Inc., 2005.

[104] A. K. Qin, V. L. Huang i P. N. Suganthan. Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE Transactions on Evolutionary Computation*, 13(2):398–417, 2009.

[105] Z. Qin, J. Chen, Y. Liu i J. Lu. Evolving RBF neural networks for pattern classification. *Proceedings of the 2005 International Conference on Computational Intelligence and Security (CIS)*, stranice 957–964, 2005.

[106] A. Qing. Dynamic differential evolution strategy and applications in electromagnetic inverse scattering problems. *IEEE Transactions on Geoscience and Remote Sensing*, 44(1):116–125, 2006.

[107] A. Qing. *Differential Evolution: Fundamentals and Applications in Electrical Engineering*. Wiley-IEEE Press, 2009.

[108] C. Qiu, M. Liu i W. Gong. Differential evolution with tournament-based mutation operators. *International Journal of Computer Science Issues*, 10(2):180–187, 2013.

[109] S. Rahnamayan i G. G. Wang. Toward effective initialization for large-scale search spaces. *WSEAS Transactions on Systems*, 8(3):355–367, 2009.

[110] S. Rahnamayan, H. R. Tizhoosh i M. M. A. Salama. A novel population initialization method for accelerating evolutionary algorithms. *Computers & Mathematics with Applications*, 53(10):1605–1614, 2007.

[111] S. Rahnamayan, H. R. Tizhoosh i M. M. A. Salama. Quasi-oppositional differential evolution. *Proceedings of the 2007 IEEE Congress on Evolutionary Computation (CEC)*, stranice 2229–2236, 2007.

[112] S. Rahnamayan, H. R. Tizhoosh i M. M. A. Salama. Opposition-based differential evolution. *IEEE Transactions on Evolutionary Computation*, 12(1):64–79, 2008.

[113] S. Rahnamayan, G. G. Wang i M. Ventresca. An intuitive distance-based explanation of opposition-based sampling. *Applied Soft Computing*, 12(9):2828–2839, 2012.

[114] R. G. Regis. Evolutionary programming for high-dimensional constrained expensive black-box optimization using radial basis functions. *IEEE Transactions on Evolutionary Computation*, 18(3):326–347, 2014.

[115] M. Richards i D. Ventura. Choosing a starting configuration for particle swarm optimization. *Proceedings of the 2004 IEEE International Joint Conference on Neural Networks (IJCNN)*, svezak 3, stranice 2309–2312, 2004.

[116] J. Rönkkönen, S. Kukkonen i K. Price. Real-parameter optimization with differential evolution. *Proceedings of 2005 IEEE International Conference on Evolutionary Computation (CEC)*, stranice 506–513, 2005.

[117] A. Salman, A. P. Engelbrecht i M. G. H. Omran. Empirical analysis of self-adaptive differential evolution. *European Journal of Operational Research*, 183(2):785–804, 2007.

[118] R. Scitovski, I. Vidović i D. Bajer. A new fast fuzzy partitioning algorithm. *Expert Systems with Applications*, 51:143–150, 2016.

[119] W. Sheng, S. Swift, L. Zhang i X. Liu. A weighted sum validity function for clustering with a hybrid niching genetic algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 35(6):1156–1167, 2005.

[120] K. Sörensen. Metaheuristics—the metaphor exposed. *International Transactions in Operational Research*, 22(1):3–18, 2015.

[121] R. Storn. On the usage of differential evolution for function optimization. *Proceedings of the Biennial Conference of the North American Fuzzy Information Processing Society (NAFIPS)*, stranice 519–523, 1996.

[122] R. Storn. Differential evolution research – trends and open questions. *Advances in Differential Evolution*, stranice 1–31. Springer Berlin Heidelberg, 2008.

[123] R. Storn i K. Price. Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces. Technical report, International Computer Science Institute, Berkeley, Kalifornija, SAD, 1995.

[124] R. Storn i K. Price. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4):341–359, 1997.

[125] R. Tanabe i A. S. Fukunaga. Success-history based parameter adaptation for differential evolution. *Proceedings of the 2013 IEEE Congress on Evolutionary Computation (CEC)*, stranice 71–78, 2013.

[126] R. Tanabe i A. S. Fukunaga. Improving the search performance of SHADE using linear population size reduction. *Proceedings of the 2014 IEEE Congress on Evolutionary Computation (CEC)*, stranice 1658–1665, 2014.

[127] R. Tanabe i A. S. Fukunaga. Tuning differential evolution for cheap, medium, and expensive computational budgets. *Proceedings of the 2015 IEEE Congress on Evolutionary Computation (CEC)*, stranice 2018–2025, 2015.

[128] M. F. Tasgetiren, P. N. Suganthan i Q. Pan. An ensemble of discrete differential evolution algorithms for solving the generalized traveling salesman problem. *Applied Mathematics and Computation*, 215(9):3356–3368, 2010.

[129] V. A. Tatsis i K. E. Parsopoulos. Differential evolution with grid-based parameter adaptation. *Soft Computing*, stranice 1–23, 2015.

[130] M. Teboulle. A unified continuous optimization framework for center-based clustering methods. *Journal of Machine Learning Research*, 8:65–102, 2007.

[131] S. Theodoridis i K. Koutroumbas. *Pattern Recognition, Fourth Edition*. Academic Press, 2008.

[132] J. Tvrdík. Competitive differential evolution. *Proceedings of the 12th International Conference on Soft Computing MENDEL*, stranice 7–12, 2006.

[133] N. Q. Uy, N. X. Hoai, R. McKay i P. M. Tuan. Initialising PSO with randomised low-discrepancy sequences: the comparative results. *Proceedings of the 2007 IEEE Congress on Evolutionary Computation (CEC)*, stranice 1985–1992, 2007.

[134] M. Črepinšek, S. Liu i M. Mernik. Exploration and exploitation in evolutionary algorithms: A survey. *ACM Computing Surveys*, 45(3):1–33, 2013.

[135] L. Vendramin, R. J. G. B. Campello i E. R. Hruschka. Relative clustering validity criteria: A comparative overview. *Statistical Analysis and Data Mining*, 3(4):209–235, 2010.

[136] J. Vesterstrøm i R. Thomsen. A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems. *Proceedings of the 2004 IEEE Congress on Evolutionary Computation (CEC)*, svezak 2, stranice 1980–1987, 2004.

[137] M. Visani, C. Garcia i J. M. Jolion. Normalized radial basis function networks and bilinear discriminant analysis for face recognition. *Proceedings of the 2005 IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS)*, stranice 342–347, 2005.

[138] M. Vogt. Combination of radial basis function neural networks with optimized learning vector quantization. *Proceedings of the 1993 IEEE International Conference on Neural Networks (ICNN)*, svezak 3, stranice 1841–1846, 1993.

[139] H. Wang, S. Rahnamayan i Z. Wu. Adaptive differential evolution with variable population size for solving high-dimensional problems. *Proceedings of the 2011 IEEE Congress of Evolutionary Computation (CEC)*, stranice 2626–2632, 2011.

[140] Z. Wang, H. Duan i X. Zhang. An improved greedy genetic algorithm for solving travelling salesman problem. *Proceedings of the 5th International Conference on Natural Computation (ICNC)*, svezak 5, stranice 374–378, 2009.

[141] M. Weber, F. Neri i V. Tirronen. Shuffle or update parallel differential evolution for large-scale optimization. *Soft Computing*, 15(11):2089–2107, 2011.

[142] X. Wu, V. Kumar, J. Ross Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, P. S. Yu, Z.-H. Zhou, M. Steinbach, D. J. Hand i D. Steinberg. Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14(1):1–37, 2007.

[143] L. Xi, H. Muzhou, M. H. Lee, J. Li, D. Wei, H. Hai i Y. Wu. A new constructive neural network method for noise processing and its application on stock market prediction. *Applied Soft Computing*, 15:57–66, 2014.

[144] H. Xie i M. Zhang. Parent selection pressure auto-tuning for tournament selection in genetic programming. *IEEE Transactions on Evolutionary Computation*, 17(1):1–19, 2013.

[145] Q. Xu, L. Wang, N. Wang, X. Hei i L. Zhao. A review of opposition-based learning from 2005 to 2012. *Engineering Applications of Artificial Intelligence*, 29:1–12, 2014.

[146] R. Xu i D. Wunsch. *Clustering*. Wiley-IEEE Press, 2009.

[147] Y. Xu, J.-a. Fang, W. Zhu, X. Wang i L. Zhao. Differential evolution using a superior–inferior crossover scheme. *Computational Optimization and Applications*, 61(1):243–274, 2015.

[148] X.-S. Yang. *Nature-Inspired Optimization Algorithms*. Elsevier Science Publishers B. V., 2014.

[149] X.-S. Yang, Z. Cui, R. Xiao, A. H. Gandomi i M. Karamanoglu. *Swarm Intelligence and Bio-Inspired Computation: Theory and Applications*. Elsevier Science Publishers B. V., 2013.

[150] X. Yao, Y. Liu i G. Lin. Evolutionary programming made faster. *IEEE Transactions on Evolutionary Computation*, 3(2):82–102, 1999.

[151] S. Yoo, S. Oh i W. Pedrycz. Optimized face recognition algorithm using radial basis function neural networks and its practical applications. *Neural Networks*, 69:111–125, 2015.

[152] W. Yu, J. Li, J. Zhang i M. Wan. Differential evolution using mutation strategy with adaptive greediness degree control. *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation (GECCO)*, stranice 73–80, 2014.

[153] W. Yu, M. Shen, W. Chen, Z. Zhan, Y. Gong, Y. Lin, O. Liu i J. Zhang. Differential evolution with two-level parameter adaptation. *IEEE Transactions on Cybernetics*, 44 (7):1080–1099, 2014.

[154] W.-j. Yu i J. Zhang. Adaptive differential evolution with optimization state estimation. *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation (GECCO)*, stranice 1285–1292, 2012.

[155] X. Yu i M. Gen. *Introduction to Evolutionary Algorithms*. Springer-Verlag London, 2010.

[156] Z. Yu, S. Song, G. Duan, R. Pei i W. Chu. The design of RBF neural networks for solving overfitting problem. *Proceedings of the 6th World Congress on Intelligent Control and Automation (WCICA)*, stranice 2752–2756, 2006.

[157] D. Zaharie. Critical values for the control parameters of differential evolution algorithms. *Proceedings of the 8th International Conference on Soft Computing MENDEL*, stranice 62–67, 2002.

[158] D. Zaharie. A comparative analysis of crossover variants in differential evolution. *Proceedings of the International Multiconference on Computer Science and Information Technology (IMCSIT)*, stranice 171–181, 2007.

[159] A. Zamuda i J. Brest. Vectorized procedural models for animated trees reconstruction using differential evolution. *Information Sciences*, 278:1–21, 2014.

[160] A. Zamuda i J. Brest. Self-adaptive control parameters' randomization frequency and propagations in differential evolution. *Swarm and Evolutionary Computation*, 25:72–99, 2015.

[161] G. Zhang, L. Gao i Y. Shi. An effective genetic algorithm for the flexible job-shop scheduling problem. *Expert Systems with Applications*, 38(4):3563–3573, 2011.

[162] J. Zhang i A. C. Sanderson. JADE: Adaptive differential evolution with optional external archive. *IEEE Transactions on Evolutionary Computation*, 13(5):945–958, 2009.

[163] J. Zhang, H. S. H. Chung i W. L. Lo. Clustering-based adaptive crossover and mutation probabilities for genetic algorithms. *IEEE Transactions on Evolutionary Computation*, 11(3):326–335, 2007.

[164] Z. Zhao, J. Yang, Z. Hu i H. Che. A differential evolution algorithm with self-adaptive strategy and control parameters based on symmetric latin hypercube design for unconstrained optimization problems. *European Journal of Operational Research*, 250 (1):30–45, 2016.

[165] Y. Zhou, J. Wang, Y. Zhou, Z. Qiu, Z. Bi i Y. Cai. Differential evolution with guiding archive for global numerical optimization. *Applied Soft Computing*, 43(C):424–440, 2016.

[166] K. Zielinski i R. Laur. Parameter adaptation for differential evolution with design of experiments. *Proceedings of the 2nd IASTED International Conference on Computational Intelligence*, stranice 216–221, 2006.

[167] B. Zorić, D. Bajer i G. Martinović. Employing different optimisation approaches for SMOTE parameter tuning. *Proceedings of the 1st International Conference on Smart Systems and Technologies (SST)*, stranice 191–196, 2016.

[168] D. Zou, H. Liu, L. Gao i S. Li. An improved differential evolution algorithm for the task assignment problem. *Engineering Applications of Artificial Intelligence*, 24(4): 616–624, 2011.

[169] D. Zou, J. Wu, L. Gao i S. Li. A modified differential evolution algorithm for unconstrained optimization problems. *Neurocomputing*, 120:469–481, 2013.

Sažetak

Potrebu ili težnju za stalnim poboljšavanjem različitih sustava ili modela moguće je susresti u mnogim vidovima inženjerstva i znanosti. U tu svrhu često je nužno suočiti se s različitim problemima optimizacije. Oni uobičajeno posjeduju mnoga svojstva koja ih čine vrlo zahtjevnima za rješavanje. Uz to, nerijetko su nalik crnim kutijama iz kojih je moguće dobiti samo odziv na danu pobudu, odnosno ulaz. Ukoliko se radi o problemima numeričke optimizacije, može se istaknuti diferencijalna evolucija (DE) kao predstavnik evolucijskih algoritama (EAs) koja je u središtu ove disertacije. Iako relativno jednostavna, dokazane je učinkovitosti, a u disertaciji su predložena tri njena unaprjeđenja. S obzirom na osjetljivost i problem određivanja prikladnih vrijednosti parametara algoritma, predložen je postupak samopodešavanja faktora skaliranja i stope križanja kao podskupa istih. On se ističe održavanjem zadanog broja prethodno uspješnih vrijednosti parametara za svakog člana populacije, a na temelju kojih se generiraju nove vrijednosti. Također, početna populacije može biti bitan čimbenik u učinkovitosti DE te je shodno tome predložena metoda za njenu inicijalizaciju, a zasnovana je na grupiranju podataka i Cauchyjevim slučajnim varijablama. Ključan element i glavna razlika DE u odnosu na druge uobičajene EAs je njena mutacija. U disertaciji je predložena mutacija koja obavlja odabir baznog vektora pomoću prilagodljive k-turnirske selekcije. Predložena unaprjeđenja opsežno su testirana na skupu odabranih standardnih testnih funkcija i na funkcijama pripremljenim za CEC 2014 natjecanje u numeričkoj optimizaciji. Analiza ostvarenih rezultata dovela je zaključka da predložena unaprjeđenja značajno pospješuju učinkovitost algoritma u koji su ugrađena te da su vrlo konkurentna u odnosu na neka srodna unaprjeđenja dostupna u literaturi. Na koncu, ispitani su ponašanje i učinkovitost standardnog algoritma DE pri izgradnji radijalnih mreža kao klasifikacijskih modela. Izgradnja tih mreža predstavlja složeni problem globalne optimizacije u prvom redu s gledišta broja parametara koje je nužno podesiti te zahtijevnog vrednovanja. Iako standardni algoritam predstavlja valjan izbor za rješavanje navedenog problema, testiranjem i analizom pokazano je da predložena unaprjeđenja i u ovom slučaju pospješuju njegovu učinkovitost.

Ključne riječi: Diferencijalna evolucija, inicijalizacija populacije, klasifikacija, mutacija, numerička optimizacija, radijalne mreže, samopodešavanje parametara

Abstract

Enhancements of differential evolution algorithm using parameters adaptation and population initialization

The need or tendency to improve different systems or models can be encountered in numerous forms of engineering and science. For that purpose, frequently optimization problems must be tackled. Such problems usually posses different properties that make them hard to solve. Besides, it is not uncommon that they are like black boxes which only provide responses to given inputs. Dealing with numerical optimization problems, differential evolution (DE) as an representative evolutionary algorithm (EA) may be pointed out, which is also in the center of the thesis. Although relatively simple, its performance is acclaimed. Three enhancements of DE are proposed in the thesis. Due to sensitivity to parameter values and the problem of determining appropriate ones, a self-adaptive scheme for controlling the scale factor and crossover-rate is proposed. It features, for each population member, the maintenance of a number of previously successful parameter values that are used for generating new ones. Also, the initial population may play an important factor in the performance of DE, and an initialization method is proposed which is based on data clustering and Cauchy deviates. The key element and difference between DE and other common EAs is its mutation. A mutation that employs an adaptive k-tournament selection for determining the base vector is proposed. All proposed enhancements were extensively tested on a set of selected standard functions and benchmark functions prepared for the CEC 2014 competition on numerical optimization. The analysis of the obtained results led to the conclusion that the enhancements considerably improve the performance of the algorithm incorporating them, but also that they compare favorably against similar enhancements from the literature. Finally, the behavior and performance of the standard DE algorithm were investigated in designing radial basis function networks for classification. The design of such networks represents a complex global optimization problem primarily from the viewpoint of the number of parameters that need to be adjusted and expansive evaluations. Although the standard algorithm is a viable choice for tackling this problem, the performed testing and analysis showed that the proposed enhancements yield improved performance in this case as well.

Keywords: Differential evolution, population initialization, classification, mutation, numerical optimization, radial basis function networks, self-adaptation of parameters

Životopis

Dražen Bajer rođen je 21.2.1986. godine u Brčkom, BiH. Akademski naziv sveučilišni prvostupnik inženjer računarstva stekao je 2008. godine na Elektrotehničkom fakultetu Osijek, a akademski naziv magistar inženjer računarstva stekao je 2010. godine također na Elektrotehničkom fakultetu Osijek. Od 1.7.2011. godine zaposlen je kao znanstveni novak u suradničkom zvanju asistenta na Elektrotehničkom fakultetu Osijek u sklopu znanstvenog projekta MZOŠ RH, „Postupci raspoređivanja u samoodrživim raspodijeljenim računalnim sustavima“. U navedenoj instituciji održavao je nastavu iz niza predmeta na stručnom, preddiplomskom i diplomskom studiju te je sudjelovao kao sumentor na završnim i diplomskim radovima. Dobitnik je priznanja za iznadprosječne rezultate na poslijediplomskom studiju Elektrotehničkog fakulteta Osijek (sada Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek) za 2015. godinu. Glavna područja znanstvenog i stručnog interesa su mu evolucijski algoritmi i njihova primjena te nadzirana i nenadzirana klasifikacija. Suautor je nekoliko znanstvenih radova objavljenih u međunarodnim znanstvenim časopisima i zbornicima konferencija. Aktivno se služi engleskim i njemačkim jezikom.

U Osijeku, 2016.

Dražen Bajer

DODATAK A

Skupovi testnih funkcija

A.1 Standradne testne funkcije

U nastavku su predstavljene odabrane standardne ili klasične testne funkcije, a njihov sažeti pregled dan je tablicom A.1. Prikazani su definicije, lokacije globalnih optimuma (minimuma), vrijednosti funkcija u tim točkama i rasponi dozvoljenih vrijednosti za nezavisne varijable (prostor pretrage). Navedene funkcije su preuzete iz nekoliko izvora u literaturi (vidi originalne izvore unutar). Funkcije $f_1 \sim f_{11}$ su unimodalne, funkcije $f_{12} \sim f_{22}$ su multimodalne s velikim brojem minimumima, dok su funkcije $f_{23} \sim f_{30}$ isto multimodalne, ali s nekoliko minimuma. Nadalje, funkcije $f_1 \sim f_{22}$ su skalabilne za razliku od funkcija $f_{23} \sim f_{30}$, što znači kako su definirane za različite proizvoljne brojeve nezavisnih varijabli d (dimenzionalnosti).

1. Model sfere (prva De Jongova funkcija) [56, 150]

$$f_1(\mathbf{x}) = \sum_{i=1}^{d} x_i^2, \quad -100 \leq x_i \leq 100 \ .$$

   Funkcija ima globalni minimum u $\mathbf{x}^\star = (0, \ldots, 0)$, gdje je $f_1(\mathbf{x}^\star) = 0$. Unimodalna, konveksna i razdvojiva funkcija.

2. Schwefelov problem 2.22 [56, 150]

$$f_2(\mathbf{x}) = \sum_{i=1}^{d} |x_i| + \prod_{i=1}^{d} |x_i| \ , \quad -10 \leq x_i \leq 10 \ .$$

   Funkcija ima globalni minimum u $\mathbf{x}^\star = (0, \ldots, 0)$, gdje je $f_2(\mathbf{x}^\star) = 0$. Unimodalna funkcija.

Tablica A.1: Pregled korištenih standardnih testnih funkcija.

Funkcija	Ime	$\mathcal{S}$	$f^\star$
f_1	Model sfere	$[-100,100]^d$	0
f_2	Schwefelov problem 2.22	$[-10,10]^d$	0
f_3	Schwefelov problem 1.2	$[-100,100]^d$	0
f_4	Schwefelov problem 2.21	$[-100,100]^d$	0
f_5	Općenita Rosenbrockova funkcija	$[-30,30]^d$	0
f_6	Zbroj različitih potencija	$[-1,1]^d$	0
f_7	Zakharova funckija	$[-5,10]^d$	0
f_8	Koračna funkcija	$[-100,100]^d$	0
f_9	Četvrta De Jongova funkcija	$[-1.28,1.28]^d$	0
f_{10}	Chung Reynolds funkcija	$[-100,100]^d$	0
f_{11}	Eksponencijalni problem	$[-1,1]^d$	-1
f_{12}	Općeniti Schwefelov problem 2.26	$[-500,500]^d$	0
f_{13}	Općenita Rastriginova funkcija	$[-5.12,5.12]^d$	0
f_{14}	Ackleyjeva funkcija	$[-32,32]^d$	0
f_{15}	Općenita Griewankova funkcija	$[-600,600]^d$	0
f_{16}	Alpine funkcija	$[-10,10]^d$	0
f_{17}	Patološka funkcija	$[-10,10]^d$	0
f_{18}	Funkcija obrnutog kosinusnog vala	$[-5,5]^d$	$-d+1$
f_{19}	Levy i Montalvo 1 problem	$[-10,10]^d$	0
f_{20}	Salomon problem	$[-100,100]^d$	0
f_{21}	Općenita penalizirana funkcija 1	$[-50,50]^d$	0
f_{22}	Općenita penalizirana funkcija 2	$[-50,50]^d$	0
f_{23}	Becker i Lago problem	$[-10,10]^2$	0
f_{24}	Kowalikova funkcija	$[-5,5]^4$	≈ 0.00030748
f_{25}	Branin funkcija	$[-5,10]\times[0,15]$	$5/(4\pi)$
f_{26}	Hartmann 3 funkcija	$[0,1]^3$	≈ -3.862782
f_{27}	Hartmann 6 funkcija	$[0,1]^6$	≈ -3.322368
f_{28}	Shekel 5 problem	$[0,10]^4$	≈ -10.1532
f_{29}	Shekel 7 problem	$[0,10]^4$	≈ -10.4029
f_{30}	Shekel 10 problem	$[0,10]^4$	≈ -10.5364

3. Schwefelov problem 1.2 [56, 150]

$$f_3(\mathbf{x}) = \sum_{j=1}^{d} \left(\sum_{i=1}^{j} x_i \right)^2 , \quad -100 \leq x_i \leq 100 \ .$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (0, \ldots, 0)$, gdje je $f_3(\mathbf{x}^\star) = 0$. Unimodalna funkcija.

4. Schwefelov problem 2.21 [56, 150]

$$f_4(\mathbf{x}) = \max_i \{|x_i|, 1 \leq i \leq d\}, \quad -100 \leq x_i \leq 100 \ .$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (0, \ldots, 0)$, gdje je $f_4(\mathbf{x}^\star) = 0$. Unimodalna funkcija.

5. Općenita Rosenbrockova funkcija (Rosenbrockova dolina ili banana funkcija) [4, 56, 150]

$$f_5(\mathbf{x}) = \sum_{i=1}^{d-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right], \quad -30 \leq x_i \leq 30 \ .$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (1, \ldots, 1)$, gdje je $f_5(\mathbf{x}^\star) = 0$. Unimodalna

funkcija za koju teško pronaći globalni optimum zbog točke sedla.

6. Zbroj različitih potencija [56]

$$f_6(\mathbf{x}) = \sum_{i=1}^{d} |x_i|^{i+1}, \quad -1 \leq x_i \leq 1 \ .$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (0, \ldots, 0)$, gdje je $f_6(\mathbf{x}^\star) = 0$. Unimodalna funkcija.

7. Zakharova funckija [56, 66]

$$f_7(\mathbf{x}) = \sum_{i=1}^{d} x_i^2 + \left(\sum_{i=1}^{d} 0.5 i x_i\right)^2 + \left(\sum_{i=1}^{d} 0.5 i x_i\right)^4, \quad -5 \leq x_i \leq 10 \ .$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (0, \ldots, 0)$, gdje je $f_7(\mathbf{x}^\star) = 0$. Unimodalna funkcija.

8. Koračna funkcija [56, 150]

$$f_8(\mathbf{x}) = \sum_{i=1}^{d} \left(\lfloor x_i + 0.5 \rfloor\right)^2, \quad -100 \leq x_i \leq 100 \ .$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (0, \ldots, 0)$, gdje je $f_8(\mathbf{x}^\star) = 0$. Unimodalna funkcija. Nekoliko različitih inačica u dano je u [56].

9. Četvrta De Jongova funkcija bez šuma (za inačice sa šumom vidi primjerice [56, 124])

$$f_9(\mathbf{x}) = \sum_{i=1}^{d} i x_i^4, \quad -1.28 \leq x_i \leq 1.28 \ .$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (0, \ldots, 0)$, gdje je $f_9(\mathbf{x}^\star) = 0$. Unimodalna funkcija.

10. Chung Reynolds funkcija [56]

$$f_{10}(\mathbf{x}) = \left(\sum_{i=1}^{d} x_i^2\right)^2, \quad -100 \leq x_i \leq 100 \ .$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (0, \ldots, 0)$, gdje je $f_{10}(\mathbf{x}^\star) = 0$. Unimodalna funkcija.

11. Eksponencijalni problem [4, 56]

$$f_{11}(\mathbf{x}) = -\exp\left(-0.5\sum_{i=1}^{d} x_i^2\right), \quad -1 \leq x_i \leq 1 \ .$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (0, \ldots, 0)$, gdje je $f_{11}(\mathbf{x}^\star) = -1$. Unimodalna funkcija. Izvorno predstavlja problem maksimiziranja, koji je ovdje sveden na problem minimiziranja [množenjem funkcije s -1, jer $\max f(\mathbf{x}) = -\min(-f(\mathbf{x}))$] zbog dosljednosti s ostalima.

12. Općeniti Schwefelov problem 2.26 [56, 150]

$$f_{12}(\mathbf{x}) = d \cdot 418.982887 - \sum_{i=1}^{d} \sin\left(\sqrt{|x_i|}\right), \quad -500 \leq x_i \leq 500 \ .$$

Funkcija ima globalni minimum (približno) u $\mathbf{x}^\star = (420.968746, \ldots, 420.968746)$, gdje je $f_{12}(\mathbf{x}^\star) = 0$. Multimodalna funkcija za koju broj lokalnih nije poznat za proizvoljnu dimenzionalnost (d).

13. Općenita Rastriginova funkcija [4, 66, 150]

$$f_{13}(\mathbf{x}) = \sum_{i=1}^{d} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right], \quad -5.12 \leq x_i \leq 5.12 \ .$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (0, \ldots, 0)$, gdje je $f_{13}(\mathbf{x}^\star) = 0$. Multimodalna funkcija za koju broj lokalnih minimuma nije točno poznat. Primjerice, prema [4], ima oko 50 lokalnih minimuma za dimenzionalnost $d = 2$.

14. Ackleyjeva funkcija [4, 56, 66, 150]

$$f_{14}(\mathbf{x}) = -20\exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos\left(2\pi x_i\right)\right) + 20 + e,$$
$$-32 \leq x_i \leq 32 \ .$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (0, \ldots, 0)$, gdje je $f_{14}(\mathbf{x}^\star) = 0$. Multimodalna funkcija za koju broj lokalnih minimuma nije poznat.

15. Općenita Griewankova funkcija [4, 56, 66, 150]

$$f_{15}(\mathbf{x}) = 1 + \frac{1}{4000}\sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d}\cos\left(\frac{x_i}{\sqrt{i}}\right), \quad -600 \leq x_i \leq 600 \ .$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (0, \ldots, 0)$, gdje je $f_{15}(\mathbf{x}^\star) = 0$. Multimodalna

funkcija za koju nije poznat broj lokalnih minimuma za proizvoljnu dimenzionlnost (d). Primjerice, prema [4], ima oko 500 lokalnih minimuma za $d = 2$.

16. Alpine funkcija [56]

$$f_{16}(\mathbf{x}) = \sum_{i=1}^{d} |x_i \sin(x_i) + 0.1x_i|, \quad -10 \leq x_i \leq 10 \ .$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (0, \ldots, 0)$, gdje je $f_{16}(\mathbf{x}^\star) = 0$. Multimodalna funkcija.

17. Patološka funkcija [56]

$$f_{17}(\mathbf{x}) = \sum_{i=1}^{d-1} \left(0.5 + \frac{\sin^2 \sqrt{100x_i^2 + x_{i+1}^2} - 0.5}{1 + 0.001(x_i^2 - 2x_i x_{i+1} + x_{i+1}^2)^2} \right), \quad -100 \leq x_i \leq 100 \ .$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (0, \ldots, 0)$, gdje je $f_{17}(\mathbf{x}^\star) = 0$. Multimodalna i vrlo složena funkcija.

18. Funkcija obrnutog kosinusnog vala [110]

$$f_{18}(\mathbf{x}) = -\sum_{i=1}^{d-1} \left(\exp\left(\frac{-(x_i^2 + x_{i+1}^2 + 0.5x_i x_{i+1})}{8} \right) \cdot \cos\left(4\sqrt{x_i^2 + x_{i+1}^2 + 0.5x_i x_{i+1}} \right) \right), \quad -5 \leq x_i \leq 5 \ .$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (0, \ldots, 0)$, gdje je $f_{18}(\mathbf{x}^\star) = -d + 1$. Multimodalna funkcija.

19. Levy i Montalvo 1 problem [4]

$$f_{19}(\mathbf{x}) = \frac{\pi}{d} \left(10 \sin^2(\pi y_1) + \sum_{i=1}^{d-1} (y_i - 1)^2 [1 + 10 \sin^2(\pi y_{i+1})] + (y_d - 1)^2 \right),$$
$$y_i = 1 + \frac{1}{4}(x_i + 1), \quad -10 \leq x_i \leq 10 \ .$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (-1, \ldots, -1)$, gdje je $f_{19}(\mathbf{x}^\star) = 0$. Multimodalna funkcija s približno 5^d lokalnih minimuma.

20. Salomon problem [4, 56]

$$f_{20}(\mathbf{x}) = 1 - \cos\left(2\pi \sqrt{\sum_{i=1}^{d} x_i^2} \right) + 0.1 \sqrt{\sum_{i=1}^{d} x_i^2}, \quad -100 \leq x_i \leq 100 \ .$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (0, \ldots, 0)$, gdje je $f_{20}(\mathbf{x}^\star) = 0$. Multimodalna funkcija za koju broj lokalnih nije poznat za proizvoljnu dimenzionalnost.

21. Općenita penalizirana funkcija 1 [150]

$$f_{21}(\mathbf{x}) = \frac{\pi}{d}\left\{10\sin^2(\pi y_1) + \sum_{i=1}^{d-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})] + (y_d-1)^2\right\} + \sum_{i=1}^{d} u(x_i, 10, 100, 4), \quad y_i = 1 + \frac{1}{4}(x_i+1), \quad -50 \le x_i \le 50,$$

$$u(x_i, a, k, m) = \begin{cases} k(x_i-a)^m, & x_i > a, \\ 0, & -a \le x_i \le a, \\ k(-x_i-a)^m, & x_i < -a \end{cases}.$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (-1, \ldots, -1)$, gdje je $f_{21}(\mathbf{x}^\star) = 0$. Multimodalna funkcija.

22. Općenita penalizirana funkcija 2 [150]

$$f_{22}(\mathbf{x}) = 0.1\Big\{\sin^2(3\pi x_1) + \sum_{i=1}^{d-1}(x_i-1)^2[1+\sin^2(3\pi x_{i+1})] + (x_d-1)^2[1+\sin^2(2\pi x_d)]\Big\} + \sum_{i=1}^{d} u(x_i, 5, 100, 4),$$

$$-50 \le x_i \le 50, \quad u(x_i, a, k, m) = \begin{cases} k(x_i-a)^m, & x_i > a, \\ 0, & -a \le x_i \le a, \\ k(-x_i-a)^m, & x_i < -a \end{cases}.$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (1, \ldots, 1)$, gdje je $f_{22}(\mathbf{x}^\star) = 0$. Multimodalna funkcija.

23. Becker i Lago problem [4]

$$f_{23}(\mathbf{x}) = (|x_1|-5)^2 + (|x_2|-5)^2, \quad -10 \le x_i \le 10 \ .$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (\pm 5, \pm 5)$, gdje je $f_{23}(\mathbf{x}^\star) = 0$. Multimodalna funkcija s četiri mimimuma.

24. Kowalikova funkcija [4, 150]

$$f_{24}(\mathbf{x}) = \sum_{i=1}^{11} \left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4} \right]^2, \quad -5 \leq x_i \leq 5,$$
$$(a_i) = [0.1957 \quad 0.1947 \quad 0.1735 \quad 0.1600 \quad 0.0844 \quad 0.0627 \quad 0.0456$$
$$0.0342 \quad 0.0323 \quad 0.0235 \quad 0.0246],$$
$$(b_i^{-1}) = [0.25 \quad 0.5 \quad 1 \quad 2 \quad 4 \quad 6 \quad 8 \quad 10 \quad 12 \quad 14 \quad 16] \ .$$

Funkcija ima globalni minimum (približno) u $\mathbf{x}^\star = (0.1928, 0.1908, 0.1231, 0.1358)$, gdje je $f_{24}(\mathbf{x}^\star) \approx 0.00030748$. Multimodalna funkcija s nekolicinom lokalnih minimuma. Funkcija predstavlja problem najmanjih kvadrata.

25. Branin funkcija [4, 56, 150]

$$f_{25}(\mathbf{x}) = \left(x_2 - \frac{5.1}{4\pi^2} x_1^2 + \frac{5}{\pi} x_1 - 6 \right)^2 + 10 \left(1 - \frac{1}{8\pi} \right) \cos(x_1) + 10,$$
$$-5 \leq x_1 \leq 10, \ 0 \leq x_2 \leq 15 \ .$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (-\pi, 12.275)$, $(\pi, 2.275)$, $(3\pi, 2.475)$, gdje je $f_{25}(\mathbf{x}^\star) = 5/(4\pi)$. Multimodalna funkcija s tri lokalna minimuma koji su ujedno i globalni.

26. Hartmann 3 funkcija [4, 56, 66, 150]

$$f_{26}(\mathbf{x}) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2 \right], \quad 0 \leq x_j \leq 1,$$
$$(c_i) = [1 \quad 1.2 \quad 3 \quad 3.2], \quad (a_{ij}) = \begin{bmatrix} 3 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3 & 10 & 30 \\ 0.1 & 10 & 35 \end{bmatrix},$$
$$(p_{ij}) = \begin{bmatrix} 0.36890 & 0.11700 & 0.26730 \\ 0.46990 & 0.43870 & 0.74700 \\ 0.10910 & 0.87320 & 0.55470 \\ 0.03815 & 0.57430 & 0.88280 \end{bmatrix} \ .$$

Funkcija ima globalni minimum (približno) u $\mathbf{x}^\star = (0.114614, 0.555649, 0.852547)$, gdje je $f_{26}(\mathbf{x}^\star) = -3.862782$. Multimodalna funkcija s četiri lokalna minimuma.

27. Hartmann 6 funkcija [4, 56, 66, 150]

$$f_{27}(\mathbf{x}) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right], \quad 0 \le x_j \le 1,$$

$$(c_i) = [1 \quad 1.2 \quad 3 \quad 3.2], \quad (a_{ij}) = \begin{bmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{bmatrix},$$

$$(p_{ij}) = \begin{bmatrix} 0.1312 & 0.1696 & 0.5569 & 0.0124 & 0.8283 & 0.5886 \\ 0.2329 & 0.4135 & 0.8307 & 0.3736 & 0.1004 & 0.9991 \\ 0.2348 & 0.1451 & 0.3522 & 0.2883 & 0.3047 & 0.6650 \\ 0.4047 & 0.8828 & 0.8732 & 0.5743 & 0.1091 & 0.0381 \end{bmatrix}.$$

Funkcija ima globalni minimum (približno) u $\mathbf{x}^\star = (0.20169, 0.150011, 0.476874, 0.275332, 0.311652, 0.6573)$, gdje je $f_{27}(\mathbf{x}^\star) = -3.322368$. Multimodalna funkcija s četiri lokalna minimuma.

28. Shekel 5 problem [4, 56, 66]

$$f_{28}(\mathbf{x}) = -\sum_{i=1}^{5} \frac{1}{\sum_{j=1}^{4}(x_j - a_{ij})^2 + c_i}, \quad 0 \le x_j \le 10,$$

$$(c_i) = [0.1 \quad 0.2 \quad 0.2 \quad 0.4 \quad 0.4], \quad (a_{ij}) = \begin{bmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \end{bmatrix}.$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (4, 4, 4, 4)$, gdje je $f_{28}(\mathbf{x}^\star) \approx -10.1532$. Multimodalna funkcija s pet lokalnih minimuma.

29. Shekel 7 problem [4, 56, 66]

$$f_{29}(\mathbf{x}) = -\sum_{i=1}^{7} \frac{1}{\sum\limits_{j=1}^{4}(x_j - a_{ij})^2 + c_i}, \quad 0 \leq x_j \leq 10,$$

$$(c_i) = [0.1 \quad 0.2 \quad 0.2 \quad 0.4 \quad 0.4 \quad 0.6 \quad 0.3], \quad (a_{ij}) = \begin{bmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \\ 2 & 9 & 2 & 9 \\ 5 & 5 & 3 & 3 \end{bmatrix}.$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (4, 4, 4, 4)$, gdje je $f_{29}(\mathbf{x}^\star) \approx -10.4029$. Multimodalna funkcija sa sedam lokalnih minimuma.

30. Shekel 10 problem [4, 56, 66]

$$f_{30}(\mathbf{x}) = -\sum_{i=1}^{10} \frac{1}{\sum\limits_{j=1}^{4}(x_j - a_{ij})^2 + c_i}, \quad 0 \leq x_j \leq 10,$$

$$(c_i) = [0.1 \quad 0.2 \quad 0.2 \quad 0.4 \quad 0.4 \quad 0.6 \quad 0.3 \quad 0.7 \quad 0.5 \quad 0.5],$$

$$(a_{ij}) = \begin{bmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \\ 2 & 9 & 2 & 9 \\ 5 & 5 & 3 & 3 \\ 8 & 1 & 8 & 1 \\ 6 & 2 & 6 & 2 \\ 7 & 3.6 & 7 & 3.6 \end{bmatrix}.$$

Funkcija ima globalni minimum u $\mathbf{x}^\star = (4, 4, 4, 4)$, gdje je $f_{30}(\mathbf{x}^\star) \approx -10.5364$. Multimodalna funkcija s 10 lokalnih minimuma.

A.2 Funkcije CEC 2014

Skup funkcija pripremljen za CEC 2014 posebnu sekciju i natjecanje u numeričkoj optimizaciji s jednom funkcijom cilja sastoji se od 30 različitih funkcija. One su sažeto prikazane tablicom A.2, a podijeljene su u četiri razreda, gdje su funkcije $F_1 \sim F_3$ unimodalne, $F_4 \sim F_{16}$ su multimodalne, $F_{17} \sim F_{22}$ su hibridne funkcije, dok su $F_{23} \sim F_{30}$ kompozicijske funkcije. Navedene funkcije ovdje neće biti dalje razmatrane s obzirom da su detaljno opisane u [70], a njihova ugradnja dostupna je u nekoliko različitih programskih jezika (uključujući ugradnju u programskom jeziku C koja je korištena u ovoj disertaciji).

Tablica A.2: Pregled funkcija CEC 2014.

Funkcija	Ime	$\mathcal{S}$	$F^\star$
F_1	Rotirana eliptična funkcija visoke kondicije		100
F_2	Rotirana funkcija savijene cigare		200
F_3	Rotirana disk funkcija		300
F_4	Pomaknuta i rotirana Rosenbrockova funkcija		400
F_5	Pomaknuta i rotirana Ackleyjeva funkcija		500
F_6	Pomaknuta i rotirana Weierstrassova funkcija		600
F_7	Pomaknuta i rotirana Griewankova funkcija		700
F_8	Pomaknuta Rastriginova funkcija		800
F_9	Pomaknuta i rotirana Rastriginova funkcija		900
F_{10}	Pomaknuta Schwefelova funkcija		1000
F_{11}	Pomaknuta i rotirana Schwefelova funkcija		1100
F_{12}	Pomaknuta i rotirana Katsuura funkcija		1200
F_{13}	Pomaknuta i rotirana HappyCat funkcija		1300
F_{14}	Pomaknuta i rotirana HGBat funkcija		1400
F_{15}	Pomaknuta i rotirana proširena Griewankova + Rosenbrockova funkcija	$[-100, 100]^d$	1500
F_{16}	Pomaknuta i rotirana proširena Schafferova F6 funkcija		1600
F_{17}	Hibridna funkcija 1		1700
F_{18}	Hibridna funkcija 2		1800
F_{19}	Hibridna funkcija 3		1900
F_{20}	Hibridna funkcija 4		2000
F_{21}	Hibridna funkcija 5		2100
F_{22}	Hibridna funkcija 6		2200
F_{23}	Kompozicijska funkcija 1		2300
F_{24}	Kompozicijska funkcija 2		2400
F_{25}	Kompozicijska funkcija 3		2500
F_{26}	Kompozicijska funkcija 4		2600
F_{27}	Kompozicijska funkcija 5		2700
F_{28}	Kompozicijska funkcija 6		2800
F_{29}	Kompozicijska funkcija 7		2900
F_{30}	Kompozicijska funkcija 8		3000

A.3 Postupak procjene vremenske složenosti

U uputama za CEC 2014 natjecanje u rješavanju problema numeričke optimizacije s jednom funkcijom cilja [70] opisan je postupak za procjenu vremenske složenosti algoritama optimizacije. Ovdje je predstavljen u usporedbi blago prošireni postupak. Proširenje se odnosi na višestruka izvođenja svih mjerenja vremena kao i na uporabu više različitih funkcija. Slijede opisi koraka.

K1. Odredi T_0 kao vrijeme potrebno za izvođenje programskog koda prikazanog algoritmom A.1.

K2. Odredi T_1 kao vrijeme potrebno za 200000 vrednovanja funkcije f dane dimenzionalnosti d.

K3. Odredi $\overline{T_2}$ kao prosjek pet vremena potrebnih za izvođenje algoritma uz 200000 vrednovanja funkcije f dane dimenzionalnosti d (isti kao u koraku K2).

K4. Odredi $C = (\overline{T_2} - T_1)/T_0$ kao procjenu vremenske složenosti algoritma.

K5. Ponovi korake K1 do K4 25 puta i pohrani dobivene rezultate.

K6. Odredi medijan rezultata, u smislu C, dobivenih u koraku K5.

Koraci K1 do K4 odgovaraju onima u [70], dok su koraci K5 i K6 uvedeni kako bi se smanjio utjecaj varijacija u vremenima izvođenja. Također, nije navedena konkretna funkcija na kojoj procjenu treba obaviti kao u [70], jer obavljanje procjene na više različitih funkcija je pouzdaniji pokazatelj.

Algoritam A.1: Pseudo-kod pomoćnog programa za procjenu vremenske složenosti

```
za i := 1 → 1000000 čini
    x := 0.55 + (double)i;
    x := x + x;
    x := x / 2;
    x := x * x;
    x := sqrt(x);
    x := log(x);
    x := exp(x);
    x := x / (x + 2);
kraj
```

DODATAK B

Podatkovni skupovi za potrebe klasifikacije

B.1 Opisi skupova

U nastavku je sažeto opisano osam podatkovnih skupova koje se često koriste u literaturi za testiranje klasifikacijskih postupaka i metoda. Skupovi su preuzeti s repozitorija za strojno učenje sveučilišta Kalifornija, Irvine [6], gdje su dostpune detaljnije informacije. Radi se o stvarnim skupovima podataka doniranim od strane različitih istraživača ili institucija. Uz ime skupa u zagradama su navedeni veličina skupa (broj uzoraka), broj značajki koji opisuje svaki uzorak te broj oznaka ili klasa, redom.

1. Banknote authentication (1372/4/2)

   Skup od 1372 označenih uzoraka. Uzorci predstavljaju podatke dohvaćene iz slika novčanica. Značajke koje opisuju uzorke dobivene su Wavelet transformacijom slika. One redom čine varijancu, iskrivljenost, oštrinu vrha krivulje frekvencijske razdiobe i entropiju slike. U skupu su zastupljene dvije vrste slika: stvarne i krivotvorene novčanice (dvije oznake ili klase).

2. Glass identification (214/9/6)

   Skup od 214 označenih uzoraka. Uzroci predstavljaju podatke o vrstama stakla u smislu kemijskog sastava i indeksa refrakcije. Značajke koje opisuju uzorke su indeks refrakcije te redom težinski postoci u odgovarajućim oksidima koje predstavljaju natrij (Na), aluminij (Al), silicij (Si), kalij (K), kalcij (Ca), barij (Ba) i željezo (Fe). U skupu je zastupljeno šest vrsta stakla (šest oznaka ili klasa).

3. Ionosphere (351/34/2)

   Skup od 351 označenih uzoraka. Uzorci predstavljaju podatke dobivene radarskim sustavom kojim su ciljani slobodni elektroni u jonosferi. Značajke koje opisuju uzorke su primljeni signalni obrađeni autokorelacijskom funkcijom s dva ulaza (vrijeme i broj pulsa), dok je izlaz kompleksna vrijednost. U sustavu se koristi 17 brojeva pulsa čime

se dobiva 34 značajke koje su rezultat obrade navedenom funkcijom. U skupu su zastupljene dvije vrste povratnih signala (dvije oznake ili klase).

4. Iris (150/4/3)

   Skup od 150 označenih uzoraka. Uzorci predstavljaju podatke o vrstama cvijeta perunike. Značajke koje opsuju uzorke su duljina i širina čašičnog listića te duljina i širina latica, sve u centimetrima. U skupu su zastupljene tri vrste cvijeta perunike (tri oznake ili klase).

5. Pima indians diabetes (768/8/2)

   Skup od 768 označenih uzoraka. Uzorci predstavljaju medicinske podatke ženskih osoba starijih od 20 godina i Pima indijanskog podrijetla. Značajke koje opisuju uzorke su broj trudnoća, koncentracija plazma glukoze, dijastolički krvni tlak, debljina kože na tricepsu, 2-satni serum inzulin, indeks mase tijela, pedigre funkcija dijabetesa, godine života. U skupu su zastupljene dvije vrste nalaza: pozitivni i negativni na dijabetes (dvije oznake ili klase).

6. Seeds (210/7/3)

   Skup od 210 označenih uzoraka. Uzorci predstavljaju geometrijske podatke o vrstama zrna pšenice. Značajke koje opisuju uzorke su dobivene iz rentgenskih slika zrna i čine površinu, opseg, kompaktnost (funkcija površine i opsega), duljinu i širinu zrna, koeficijent asimetrije i duljinu žlijeba zrna. U skupu su zastupljene tri vrste pšenice (tri oznake ili klase).

7. Thyroid (215/5/3)

   Skup od 215 označenih uzoraka. Uzorci predstavljaju medicinske podatke o štitnjači. Značajke koje opisuju uzroke su T3 smola, tiroksin, trijodtironin, tiroidni stimulirajući hormon (TSH) i TSH vrijednost. U skupu su zastupljene tri vrste nazala: uredan, hipertireoza, hipotireoza (tri oznake ili klase).

8. Wine (178/13/3)

   Skup od 178 označenih uzoraka. Uzroci predstavljaju podatke kemijske analize vina. Značajke koje opisuju uzorke su alkohol, jabučna kiselina, pepeo i njegova lužnatost, magnezij, ukupno fenola, flavonoidi, neflavonoidni fenoli, proantocijanidi, intenzitet i nijansa boje, OD280/OD315 razrijeđenog vina te prolin. U skupu su zastupljena vina dobivena iz tri kulture s istog područja u Italiji (tri oznake ili klase).

B.2 Normalizacija podataka

Značajke u svakom od opisanih skupova su cjelobrojne ili realne. Međutim, rasponi vrijednosti pojedinih značajki koje opisuju uzorke danog skupova često se uvelike razlikuju. Ovo može osjetno narušiti učinkovitost ili otežati traženje dobrih parametara klasifikacijskih modela pa je stoga poželjno njihove vrijednosti svesti na jednak interval, primjerice $[0, 1]$. To znači kako je potrebno skup uzoraka $\mathcal{A} = \{\mathbf{a}^i = (a_1^i, \ldots, a_p^i) \colon i = 1, \ldots, n\} \subset [\alpha, \beta] \subset \mathbb{R}^p$, gdje je $\alpha = (\alpha_1, \ldots, \alpha_p)$ i $\beta = (\beta_1, \ldots, \beta_p)$, transformirati u skup $\mathcal{B} = \{T(\mathbf{a}^i) \colon \mathbf{a}^i \in \mathcal{A}\} \subset [0, 1]^p$ uz preslikavanje $T \colon [\alpha, \beta] \to [0, 1]^p$. Jedno od mogućih preslikavanja je [118]

$$T(\mathbf{a}) = \mathbf{D} \cdot (\mathbf{a} - \alpha)^T, \quad \mathbf{D} = \operatorname{diag}\left(\frac{1}{\beta_1 - \alpha_1}, \ldots, \frac{1}{\beta_p - \alpha_p}\right), \tag{B.1}$$

gdje je $\mathbf{D} \in \mathbb{R}^{p \times p}$ dijagonalna matrica.

DODATAK C

Korišteni programski jezici i računalno sklopovlje

U nastavku su navedeni programski jezici korišteni za ugradnju algoritama i provođenje statističkih testova te računala korištena za izvođenje algoritama u svrhu njihovog testiranja.

- Za potrebe testiranja algoritama u poglavljima 3, 4 i 5 razvijena je okolina u programskom jeziku C, dok je za iste potrebe u poglavlju 6 razvijena okolina u programskom jeziku C# uz uporabu Accord.NET radnog okvira. Taj radni okvir korišten je samo za potrebe izračuna Moore-Penrose-ovog inverza matrice (rješavanje linearnog problema najmanjih kvadrata). Navedenim okolinama obuhvaćeni su svi algoritmi predstavljeni u eksperimentalnim analizama ove disertacije.
- S obzirom na vrlo veliki broj testiranja koje bilo nužno provesti, korištena su tri računala, a njihove karakteristike sažeto su prikazane tablicom C.1. Mora se napomenuti da je za potrebe testiranja u poglavlju 4, i to gdje je bilo nužno bilježiti i vrijeme izvođenja algoritama, korišteno samo računalo označeno u tablici s Rač. 2.
- Za provođenje statističkih testova, odnosno Wilcoxonovog testa ranga s predznakom (neparametarski – statistički test) korišten je programski jezik i okolina za statističko računanje R. Točnije, korištena je funkcija `wilcox.test()` s argumentom `paired=TRUE`.

Tablica C.1: Karakteristike računala korištenih za potrebe eksperimentalnih analiza.

	Vrsta	Procesor	Radna mem.	OS
Rač. 1	Stolno	Core™ i5-2550K @ 3.40 GHz	12 GB	Windows® 8.1 x64
Rač. 2	Prijenosno	Pentium® E5200 @ 2.50 GHz	4 GB	Windows® 7 x64
Rač. 3	Stolno	Pentium® E5800 @ 3.20 GHz	8 GB	Windows® 7 x64