UNIVERZITET U BEOGRADU
Matematički fakultet

Marija Pecić

ROBUSNA REŠENJA U VIŠEKRITERIJUMSKOJ OPTIMIZACIJI

Diplomski - master rad

B e o g r a d
2009.

Mentor:	**dr Vladimir Filipović** Matematički fakultet u Beogradu
Komentor:	**dr Jozef Kratica** Viši naučni savetnik Matematički institut SANU
Član komisije:	**dr Miroslav Marić** Matematički fakultet u Beogradu

Datum odbrane: 30. oktobar 2009

Robusna rešenja u višekriterijumskoj optimizaciji

Rezime

Pri optimizaciji realnih problema, korisnici nisu uvek zainteresovani za nalaženje globalno-optimalnih rešenja, pogotovu ako su ona isuviše osetljiva na perturbacije koje ne mogu biti zaobiđene u praksi. Oni su radije zainteresovani da pronađu robusna rešenja koja su manje osetljiva na promene u vrednostima promenljivih i/ili vrednostima funkcije cilja.
Genetski algoritmi (GA) su često jako pogodni za optimizaciju problema koji uključuju više, često suprotstavljenih ciljeva, tj. u tzv. višekriterijumskoj optimizaciji.
Do sada, nije publikovano mnogo radova koji kombinuju ovo dvoje. U ovom radu primenjen je višekriterijumski genetski algoritam za traženje robusnih rešenja u prisustvu neodređenosti. Dobijeni rezultati su ohrabrujući i sugerišu dalju primenu predložene metode na druge još složenije probleme.

Robust solutions in multi-objective optimization

Abstract

When optimize real world problems, users may not always be interested in finding the global optimal solutions, particulary if these solutions are sensitive to the variable perturbations which cannot be avoided in practice. They are rather interested in finding robust solution which are less sensitive to small changes of the decision and/or objective variables.
Genetic algorithms (GAs) are often wellsuited for optimization problems involving several, often conflicting objectives, so-called multi objective optimization.
Until now, there are no many papers combining these two. In this paper we apply multi objective genetic algorithm for finding robust solutions in the presence of uncertainties. The results are encouraging and suggest further application of the proposed method to other more complex design problems.

Predgovor

Rad se sastoji iz šest poglavlja, zaključka, jednog dodatka i literature.

Uvodno, prvo poglavlje daje osnovnu motivaciju i opis samog problema robusnih rešenja u višekriterijumskoj optimizaciji.

Osnovni koncept i terminologija višekriterijumske optimizacije, Pareto dominacije i Hipervolumena date su u drugom poglavlju. Svi ovi pojmovi se suštinski koriste nadalje u ovom radu.

U trećem poglavlju je opisan standardni GA i njegovi operatori.

Dve osnovne vrste neodređenosti: robusnost i šum su detaljno razmatrane u poglavlju broj četiri.

U petom poglavlju dat je opis implementacije koja je realizovana. Test problemi koji su korišćeni i rezultati koji su dobijeni predstavljeni su u poglavlju broj šest.

Zaključak i pravci daljeg razvoja su sadržaj poslednjeg poglavlja.

U dodatku je dat srpsko-engleski rečnik svih važnijih pojmova korišćenih u radu.

Ovom prilikom želela bih da se zahvalim:

Mentoru dr Vladimiru Filipoviću i komentoru dr Jozefu Kratici na rukovođenju pri izradi ovog rada.

Dr Miloslavu Mariću na pažljivom čitanju rada i korisnim savetima.

Prof. dr Eckart Zitzleru i njegovom asistentu Johannes Baderu na obezbeđivanju programskog koda iskorišćenog za poređenje performansi u odnosu na GA implementaciju drugih metoda za traženje robusnih rešenja.

Mariji Milanović na pomoći u tehničkim detaljima.

Mojoj porodici i vereniku na pruženoj podršci i ljubavi bez koje ovaj rad ne bi bio moguć.

Beograd, oktobar 2009

Kandidat
Marija Pecić

Skraćenice

GA	-	Genetski Algoritam
EA	-	Evolucioni Algoritam
SIBEA	-	Simple Indicator-Based Optimization Algorithm
X	-	prostor pretrage
Y	-	prostor vrednosti funkcija cilja
X^*	-	Pareto skup
Y^*	-	Pareto front
α_A	-	funkcija dostignuća skupa A
I	-	Hipervolume indikator
f	-	funkcija cilja
f_{rob}	-	funkcija pod uticajem robusnosti
f_{noise}	-	funkcija pod uticajem šuma
δ	-	perturbacija
f^{eff}	-	efektivna funkcija
ϵ	-	greška šuma
η	-	Debova konstanta
γ	-	nova konstanta

Sadržaj

Glava 1

Uvod

Zadnju dekadu obeležila su istraživanja u oblasti višekriterijumske optimizacije čiji je osnovni cilj nalaženje Pareto optimalnih rešenja. Ovakva rešenja nisu dominirana od strane drugih rešenja, tj. ne postoji ni jedno rešenje iz pretraživačkog skupa koje je bolje, po svim funkcijama cilja, od bilo kog rešenja iz tog Pareto skupa. Posmatrano sa teoretske strane gledišta ovakva rešenja su od izuzetnog značaja. Međutim, u praksi rešenja ne možemo uvek predstaviti sa željenom tačnošću i implementirano rešenje se može razlikovati od svoje teoretske vrednosti. Zbog toga, u praksi, ovakva rešenja nisu od presudnog značaja i poseban napor se ulaže u pronalaženju robusnih rešenja, koja su manje osetljiva na promene u vrednostima promenljivih.
Do sada je publikovan određen broj radova posvećen robusnim rešenjima u jednokriterijumskoj optimizaciji, ali samo mali broj traži robusna rešenja u višekriterijumskoj optimizaciji. Ovo je poslužilo kao dodatna motivacija da ovaj rad bude posvećen baš tom problemu.
Perturbacije koje mogu zahvatiti promenljive zbirno smo nazvali- neodređenosti. Generalno, neodređenosti možemo podeliti u dve osnovne klase: robusnost i šum u zavisnosti od toga da li su obuhvaćene ulazne promenljive ili funkcije cilja. U ovom radu je dat pregled svih dosadašnjih radova iz ove oblasti kao i jedna nova definicija robusnog rešenja u prisustvu ovako opisanih neodređenosti.
Kao alat koristićemo genetski algoritam, metod koji spada u opšte heuristike i koji je izuzetno primenjiv u višekriterijumskom slučaju. Kada se pomene istovremena optimizacija više kriterijuma odmah se nadovezuje priča o Pareto dominaciji a od skora i pojam hipervolume indikatora, kao mere kvaliteta aproksimacije Pareto skupa. Ovaj indikator je stekao ogromnu popularnost u zadnje vreme zahvaljujući svojim dobrim osobimana pre svega zbog izuzetne prilagođenosti na nove uslove i što predstavlja Lebegovu meru u odgovarajućim prostorima. Zadatak implementacije je da se dodefiniše hipervolume indikator koji će funkcionisati i u prisustvu neodređenosti kao i da se modifikuje genetski algoritam baziran na ovom indikatoru koji bi trebalo da optimizuje i skupove robusnih rešenja.

Glava 2

Višekriterijumska optimizacija

2.1 Osnovni koncept i terminologija

Optimizacija, u opštem smislu, predstavlja rešavanje problema minimizacije ili maksimizacije funkcija sistematskim biranjem vrednosti promenljivih iz određenog skupa.
Jednokriterijumska optimizacija se može predstaviti kao:

$$min/max\ y = f(x),\ x \in X$$

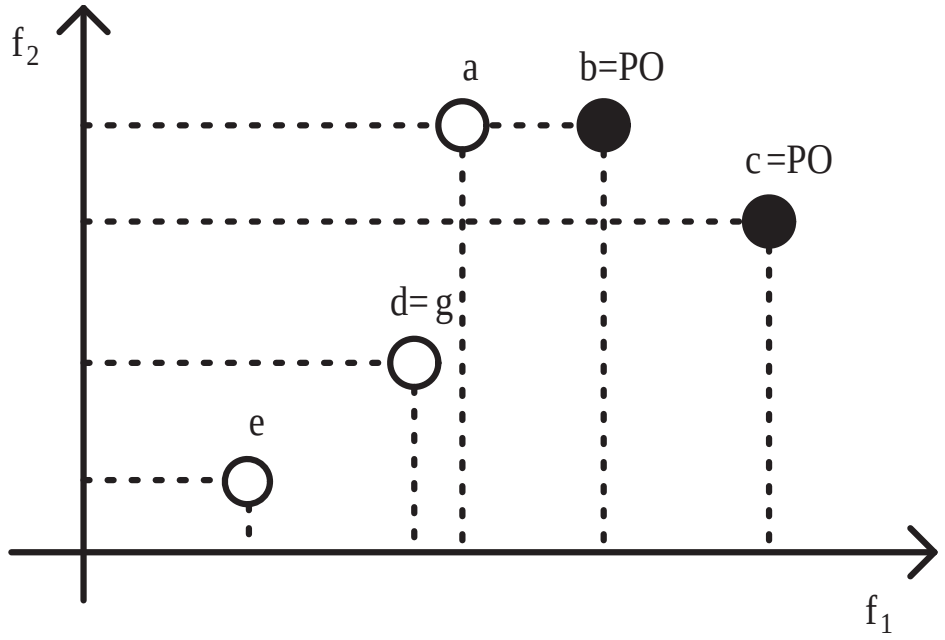
gde vektor promenljivih $x = (x_1, x_1, ..., x_m) \in X$, a vrednost funkcije $y = (f(x_1), f(x_2), ..., f(x_m)) \in Y$. Skup X nazivamo *prostor pretrage*, a skup Y *prostor vrednosti funkcije cilja*. Funkcija koja vrši preslikavanje $f : X \to Y$ ($Y \subseteq R$) naziva se *funkcija cilja*.
Slicno, *višekriterijumska optimizacija* se može predstaviti kao:

$$min/max\ y = (f_1(x), f_2(x), ..., f_n(x)),\ x \in X$$

pri čemu *vektor funkcije cilja* $y = (y_1, y_2, ..., y_n)$ vrši preslikavanja $f : X \to Y$ ($Y \subseteq R^n$), gde $n \geq 2$ predstavlja broj kriterijuma optimizacije.
U jednokriterijumskoj optimizaciji uvek je jasno koje je rešenje najbolje. Tako na primer u jednokriterijumskoj maksimizaciji rešenje $x_1 \in X$ je bolje od resenja $x_2 \in X$ ako je $y_1 > y_2$ gde $y_1 = f(x_1)$ i $y_2 = f(x_2)$. I ako nekoliko optimalnih rešenja mogu postojati u pretraživačkom prostoru, ona se sva preslikavaju u isti vector funkcije cilja, odnosno postoji samo jedan optimum u prostoru vrednosti funkcija cilja.
Za razliku od ovako uređenog (skalarnog) prostora pretrage, višedimenzioni prostori pretrage su samo delimično uređeni pa je poređenje dva rešenja x_1 i x_2 mnogo kompleksnije. To nas uvodi u priču o konceptu Pareto dominacije.

2.2 Pareto dominacija

Bez gubitka na opštosti razmotrimo višekriterijumski problem maksimizacije. Vektor promenjive $a \in X$ *dominira* nad vektorom promenjive $b \in X$ ($a \succ b$) akko $\forall i \in \{1, 2, ..., n\} : f_i(a) \geq f_i(b) \wedge \exists j \in \{1, 2, ..., n\} : f_j(a) > f_j(b)$. Slično, kažemo da *a slabo dominira b* i pisemo $a \succeq b$ akko $\forall (1 \leq i \leq n) : f_i(a) \geq f_i(b)$. a je *neuporedivo* sa b ($a||b$) ako $\exists i \in \{1, 2, ..., n\} : f_i(a) > f_i(b) \wedge \exists j \in \{1, 2, ..., n\} : i \neq j, f_j(b) > f_j(a)$. Dva rešenja mogu biti i *jednaka* ako za $\forall (1 \leq i \leq n) : f_i(a) = f_i(b)$ ($a \sim b$). Znači, rešenje koje nije dominirano ni jednim drugim rešenjem naziva se *nedominiranim* ili *Pareto optimalno*. Skup optimalnih rešenja iz prostora pretrage zovemo *Pareto skup* $X^* \subseteq X$, a odgovarajuci skup u prostoru vrednosti funkcija cilja *Pareto front* $Y^* = f(X^*) \subseteq Y$.
Napomenimo još da Pareto optimalna rešenja ne mogu biti poboljšana po nekom kriterijumu, a da pritom ne budu pogoršana po nekom drugom. U tom smislu ona predstavljaju svojevrsna globalna optimalna rešenja.



Slika 2.1: Pareto dominancija

Na slici 2.1 dati su primeri relacija parova rešenja. Pretpostavimo da se radi o problemu maximizacije, sledeće relacije važe: $a \succ e$, $a \succ d$, $a \succ g$, $b \succ a$ $c||a$, $c||b$, $d \sim g$. Rešenja b i c su Pareto optimalna (PO).

2.3 Hipervolume indikator

Motivacija

U opštem slučaju, u višekriterijumskoj optimizaciji ne postoji jedinstveno rešenje koje je najbolje po svim kriterijumima. Umesto njega tražimo čitav skup rešenja $X^* \subseteq X$ nazvan Pareto optimalan skup za koji važi:

$$\forall x^* \in X^* : \neg\exists x \in X : x > x^*$$

Međutim, u mnogim problemima optimizacije koji se pojavljuju u praksi izračunavanje Pareto optimalnih rešenja nije moguće. Do sada je publikovan veliki broj radova koji se bavi problemom aproksimacije ovog skupa. I ako sama ideja datira iz 1998, Zitzler i Thiele [Zit98], metod baziran na hipervolume indikatoru kao meri kvaliteta aproksimacije Pareto skupa stekao je veliku popularnost poslednjih godina. Zbog svojih dobrih osobina kao na primer osetljivost na Pareto dominaciju i zato što predstavlja Lebegovu meru, izuzetno je primenljiv u višekriterijumskoj optimizaciji.

Uvod

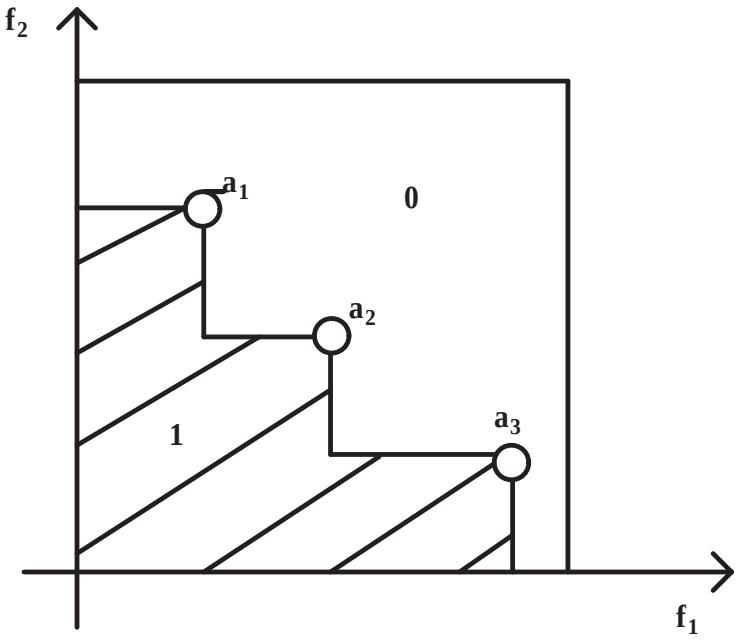
Relacija slabe dominacije ($\succeq$) koja je prethodno bila definisana između pojedinačnih rešenja u prostoru pretrage X može se proširiti i na skupove rešenja u istom prostoru. Skup $A \subseteq X$ *slabo dominira* skupom $B \subseteq X$ ($A \succeq B$) akko $\forall b \in B, \exists a \in A : a \succeq b$ [Zit03].
Uzmimo ovu relaciju i pokušajmo da nađemo skup rešenja koji aproksimira skup Pareto optimalnih rešenja uz uslov da taj skup pritom nije dominiran ni sa jednim drugim skupom. Pretpostavimo da je ishod višekriterijumske optimizacije skup vektora funkcije cilja i označimo ga sa A, a skup svih mogućih skupova vektora funkcije cilja označimo sa Ω.
Kako slaba Pareto dominacija definiše samo delimično uređenje, mogu postojati skupovi u Ω koji su neuporedivi- superiorniji u odnosu na druge skupove po nekom kriterijumu, ali inferiorniji po nekom drugom kriterijumu. Ovaj problem se može rešiti definisanjem potpunog poretka u skupu Ω koji garantuje da se svaka dva skupa vektora funkcije cilja mogu međusobno porediti. Dakle, svakom takvom skupu dodelićemo realni broj- unarni *indikator* I, takav da $I : \Omega \rightarrow R$. To znači da ako skup A slabo dominira skupom B i skup B ne dominira slabo skupom A, onda indikator za skup A ($I(A)$) ne sme biti lošiji od indikatora skupa B ($I(B)$).
Formalno, $A \succeq B \wedge \neg(B \succeq A) \Rightarrow I(A) > I(B)$.

Definicije

U ovom radu daćemo opštu definiciju hipervolume indikatora u odnosu na funkciju dostignuća koja dozvoljava da se posmatraju proizvoljne relacije dominacije. Bez gubitka na opštosti, posmatrajmo problem maksimizacije u n-kriterijumskoj optimizaciji, gde vrednosti funkcije cilja leže u intervalu $(0,1) : f_i : X \rightarrow Z, Z = (0,1)^n, 1 \leq i \leq n$. Postavljen uslov pojednostavljuje narednu diskusiju, ali ne predstavlja ozbiljno ograničenje jer postoji bijektivno preslikavanje skupa R u otvoren interval $(0,1) \subset R$. Funkcija dostignuća dodeljuje svakom vektoru funkcije cilja iz Z verovatnoću da je slabo dominiran skupom A. Za slučaj jednočlanog skupa neznatno pojednostavljena definicija glasi:
Definicija (Funkcija dostignuća) Neka je dat skup $A \in \Omega$. Funkcija dostignuća $\alpha_A : [0,1]^n \rightarrow \{0,1\}$ za skup A je definisana sa:

$$\alpha_A(x) := \begin{cases} 1, \; if \; A \succeq z \\ 0, \; else \end{cases}$$

za $z \in Z$.
Ova definicija je ilustrovana za slabu Pareto dominaciju na slici 2.2.



Slika 2.2: Funkcija dostignuća α_A za skup $A = \{a_1, a_2, a_3\}$

U tom slučaju možemo definisati hipervolume indikator kao površinu prostora vrednosti funkcije cilja oivičen funkcijom dostignuća i koordinatnim osama. Formalno, u opštem slučaju
Definicija (Hipervolume indikator) Hipervolume indikator I_H sa referentnom tačkom $(0, ..., 0)$ se može formulisati kao:

$$I_H(A) := \int_{(0,...,0)}^{(1,...,1)} \alpha_A(z)dz$$

gde je A bilo koji skup vektora funkcije cilja u Ω.
Kao što smo već napomenuli, ovo je samo osnovna definicija i I_H se može lako modifikovati u zavisnosti od specifičnosti zahteva. Ova važna prednost je iskorišćena u mnogim evolucionim optimizacijama baziranim na ovom indikatoru.

Poboljšanja

Nažalost ova mera nije savršena, kao i svaka mera u potpuno uređenom prostoru. To se posebno ogleda u favorizovanju unutrašnjih delova prostora vrednosti funkcije cilja. Funkcija dostignuća je binarna funkcija koja svim slabo dominiranim vektorima funkcije cilja dodeljuje jedinicu, a ostalima nulu. To znači da svi slabo dominirani vektori funkcije cilja imaju istu težinu i da podjednako doprinose ukupnoj vrednosti indikatora. Kako bi se smanjila favorizacija određenih delova prostora vrednosti funkcije cilja ideja je da se različitim regionima dodeli različita težina. Ovo se može postići definisanjem respodele težine tako da vrednost određenog slabo dominiranog vektora funkcije cilja koji učestvuje u ukupnoj vrednosti indikatora bude bilo koji broj veci on nule. U tom smislu je uvedena funkcija raspodele težina $w : Z \to R^+$, pa hipervolume indikator možemo izračunati kao proizvod funkcije raspodele težina i funkcije dostignuća:

$$I_H^w(A) := \int_{(0,...,0)}^{(1,...,1)} w(z)\alpha_A(z)dz$$

Na taj način osnovni hipervolume indikator je modifikovan i sprečeno je favorizovanje rešenja u unutrašnjosti i data je veća šansa rešenjima na krajevima-bližim koordinatnim osama.
Detaljnije o raznim modifikacijama i primenama hipervolume indikatora može se naci u radovima [Zit07] i [Aug09].

Glava 3

Genetski algoritmi

3.1 Uvod

Genetski algoritmi (GA) predstavljaju klasu stohastičkih optimizacionih metoda koji imitiraju procese prirodne evolucije. Poreklo GA se može pratiti unazad do kasnih 60-tih, ali se kao idejni tvorac zvanično uzima John Holland [Hol75]. I ako je osnovna ideja mehanizma jednostavna, ovi algoritmi su se pokazali kao vrlo efikasni u praksi zbog svoje opštosti, robusnosti, paralelnom izvršavanju i brzini [Bae97]. Sve ove poželjne osobine GA prenosi sa jednokriterijumske i na višekriterijumsku optimizaciju gde druge metode je mnogo teže primeniti. Posebno kada problem zahteva simultanu optimizaciju često konfliktnih ciljeva, u izuzetno velikim i kompleksnim prostorima pretrage, GA su se pokazali jako pogodnim. GA pritom ne garantuje da će pronaći optimalno rešenje. Kao alternativu, pokušava da nađe dobre aproksimacije, odnosno skup rešenja čiji vektori funkcije cilja nisu daleko od optimalnih. Zato nije slučajno da je interesovanje za GA postalo veliko i do sada je publikovano mnoštvo radova stranih [Jon75], [Boo87], [Gol89], [Dav91], [Bea93a], [Bea93b], [Yur94], [Mic96], [Mit96], [Mue97], tako i domaćih autora [Čan96], [Fil97], [Kra97a], [Toš97], [Fil98].

3.2 Opis GA

Kao što smo već pomenuli GA oponaša prirodnu evoluciju. Kandidati za rešenje se nazivaju *jedinke* a konačan skup jedinki čini *populaciju*. Sledi šematski prikaz osnovnih elemenata GA, koja će u daljem tekstu biti detaljnije opisana.

```
UčitavanjePodataka();
InicijalizovanjePopulacije();
while not KriterijumZavršetka() do
    for ind:= 1 to N_pop do
      p_i:= FunkcijaCilja();
    endfor
    FunkcijaPrilagođenosti();
    Selekcija();
    Ukrštanje();
    Mutacija();
endwhile
ŠtampanjeIzlaznihPodataka();
```

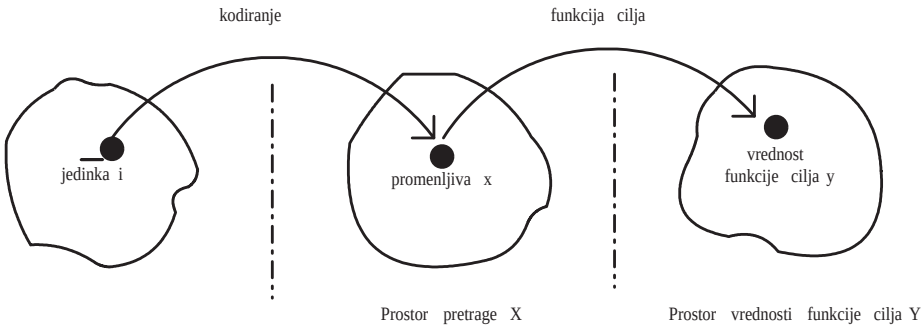
Svaka iteracija ovog preocesa se naziva *generacija*.

3.2.1 Inicijalizacija

Na početku individualna rešenja se slučajno generišu za formiranje *početne populacije*. I ako se unutar populacije može pojaviti više primeraka iste jedinke cilj je da genetski materijal bude što raznovrsniji i da pokriva čitav niz mogućih rešenja. Ukupan broj jedinki tj. *veličina populacije* zavisi od prirode problema, što najčešće iznosi između 20 i 200.

3.2.2 Kodiranje

Svaka jedinka je predstavljena nizom karaktera- *genetskim kodom*. *Kodiranje* može biti binarno (nad binarnom azbukom $\{0, 1\}$) ili nad nekom drugom azbukom veće kardinalnosti (na primer celim ili realnim brojevima). Oprečna su mišljenja koja je reprezentacija bolja. Goldberg u [Gol89] daje prednost binarnom kodiranju, dok u radovima [Ant89] i [Bea93b] se ističe prednost kodiranja brojevima. Postoji i čitava paleta tehnika kodiranja inspirisana i prilagođena konkretnim problemima, kao na primer problemu trgovačkog putnika, problem ranca i drugim. Više o tome u radovima [Gre95], [Hin94], [Rai99]. Ovde ćemo samo naglasiti da kodiranje predstavlja važan korak GA i da neadekvatan izbor koda može dovesti do lošeg rezultata bez obzira na ostalu strukturu algoritma. Na slici 3.1 predstavljena je očigledna razlika između kodiranja i funkcije cilja.



Slika 3.1: Kodiranje vs funkcija cilja

3.2.3 Funkcija prilagođenosti

Prilagođenost je skalar koji se dodeljuje svakoj jedinki, odnosno svakom pojedinačnom rešenju u pretraživačkom prostoru. On ocenjuje kvalitet jedinke odnosno njenu prilagođenost u okviru populacije. Postoje različiti načini računanja *funkcije prilagođenosti*: *direktno preuzimanje*, *linearno skaliranje*, *sigma odsecanje* i druge. Detaljnije o njima se može naći u radovima [Sri94], [Bea93a].
Na izbor odgovarajućeg načina odlučuju specifičnosti samog problema među kojima jako bitan faktor igra i broj funkcija cilja. Dok se kod jednokriterijumske optimizacije funkcija prilagođenosti često poistovećuje sa funkcijom cilja u višekriterijumskoj optimizaciji bi u tom slučaju umesto skalara dobili vektor. Jedan od načina da se to prevaziđe je da se funkciji prilagođenosti dodeli srednja vrednost funkcija cilja.
Opšti cilj svakog GA je da se iz generacije u generaciju poboljša prilagođenost svake jedinke u populaciji, dok je po nekima bitna i srednja prilagođenost cele populacije. To se postiže uzastopnom primenom genetskih operatora: *selekcije*, *ukrštanja* i *mutacije*, čime se dobijaju sve bolja rešenja datog problema.

3.2.4 Selekcija

Tokom svake naredne generacije, deo postojeće populacije obrazuje novu populaciju. Selekcija se primenjuje u skladu sa vrednostima funkcije prilagođenosti. Kandidati sa većom prilagođenošću imaju veće šanse za opstvenom reprodukcijom pri formiranju nove generacije, dok one sa manjom postepeno nestaju iz populacije. Selekcija u kojoj se svakoj jedinki daje šansa proporcionalna njenoj prilagođenosti naziva se *prosta rulet selekcija*, međutim ona može dovesti do opadanja raznovrsnosti genetskog materijala

i preuranjene konvergencije.
Selekcija bazirana na *rangu* nadoknađuje nedostatke koji su prethodno primećeni. Unapred se zada (nenegativan) niz rangova za odgovarajuće pozicije jedinki u populaciji. Funkcija prilagođenosti zatim dobija vrednost ranga date jedinke i zavisi samo od njenog poretka u populaciji. Na taj način jedinka učestvuje na "ruletu"proporcionalno svom rangu i ne zavisi od svoje konkretne vrednosti, čime se sprečava preuranjena konvergencija. Više o ovom operatoru u radu [Whi89].
Još jedan pogodan oblik selekcije je *turnirska selekcija*. Za unapred dati celi broj N simulira se N turnira tj. generiše N podskupova na sledeći način. Proizvoljno se izaberu N jedinki iz populacije i jedinka sa najboljom funkcijom prilagođenosti osvaja turnir, tj. učestvuje u stvaranju nove generacije. Neki put je pogodnije da N ne bude ceo broj. U tom slučaju se radi o *fino gradiranoj turnirskoj selekciji*. Detaljan opis ove i komparativne studije sa drugim operatorima selekcije mogu se naći u radovima [Gol91], [Fil98], [Fil00], [Fil01], [Fil03]. U ovom radu korišćena je posebna vrsta selekcije tzv. *environmental selection* [Zit04]. Više o njoj biće reči u petom poglavlju.

3.2.5 Ukrštanje

Genetski operator ukrštanje omogućava mešanje genetskog koda dve (ili više) jedinki koje predstavljaju roditelje i stvaranje unapred definisanog broja jedinki- potomaka. Cilj ovog mehanizma je da obezbedi najveći mogući stepen naslednosti, tj. da potomstvo treba da ima što je više moguće zajedničkih osobina sa svojim roditeljima. Na taj način bolje prilagođene jedinke će preneti dobar genetski materijal na svoje potomke, ali i neki dobri geni relativno lošijih jedinki dobijaju svoju šansu za bolju reprodukciju. Zajedničko za sve operatore ukrštanja je da se trude da imitiraju stohastičku prirodu evolucije dok za izbor konkretnog u mnogome odlučuje međuzavisnost gena u genetskom kodu. U slučajevima kada je međuzavisnost bitova velika korisno je *jednopoziciono ukrštanje*, jer će ono najviše sačuvati strukturu genetskog koda. Ukoliko želimo da u većoj meri razbijemo strukturu jedinke možemo izabrati *dvopoziciono* ili *višepoziciono ukrštanje*. Ako su geni skoro ili potpuno nezavisni najpogodnije je *uniformno ukrštanje*. Više karakteristika o ovim kao i o složenijim vidovima ovog operatora može se naći u radovima [Bea93b], [Sri94], [Mue97], [Jog89], [Mos89], [Spe91], [Sys89].

3.2.6 Mutacija

Višestrukom primenom selekcije i ukrštanja moguće je gubljenje genetskog materijala. Nasuprot tome, mutacija uvodi inovaciju i raznovrsnost u populaciju menjajući delove jedinke (određenog gena jedinke) sa datom malom verovatnoćom.
Izbor operatora mutacije bitno utiče na performanse GA, ali se u praksi

često koristi *prosta mutacija*. Operator prolazi kroz gene genetskog koda i za svaki od njih proverava da li je mutiran ili ne. Pretraga se može ubrzati ako se za svaku reč u genetskom kodu formira poseban binarni niz tzv. maska, koja nosi informaciju o tome na kojoj poziciji u genetskom kodu je došlo do promene gena. Korišćenjem binomne ili normalne raspodele može se takođe ubrzati realizacija proste mutacije.
Neka slučajna promenljiva X_{ind} odgovara celom genetskom kodu jedinke i označava broj mutiranih gena u njoj. Ako X_{ind} ima binomnu raspodelu $B(n, p_{mut})$ gde je n dužina genetskog koda i p_{mut} nivo mutacije, tada govorimo o *mutaciji pomoću binomne raspodele*.
Za slučaj kada je n dovoljno veliko ($n \to \infty$), a p_{mut} dovoljno malo, slučajna promenljiva X_{ind} se može aproksimirati normalnom raspodelom, pa je tada pogodnije koristiti *mutaciju pomoću normalne raspodele*.
U oba slučaja inverzna funkcija odgovarajuće funkcije raspodele daće nam broj gena koje treba mutirati, dok se pozicije gena koji mutiraju biraju uniformno iz skupa $\{0, 1, ..., n-1\}$. Detaljnije o ovim konceptima mutacije može se naći u radovima [Kra00], [Toš04].
Postoji i određen broj operatora mutacije koji direktno zavise od prirode problema. To se najčešće dešava usled specifičnosti načina kodiranja ili usled velikog broja ograničenja. I u takvim slučajevima cilj je očuvanje korektnosti genetskih kodova jedinki kao i osobina kanonskog GA.

3.2.7 Kriterijum zaustavljanja

Kao što je prikazano na šemi, operatori genetskog algoritma se uzastopno primenjuju do zadovoljenja nekog od *kriterijuma zaustavljanja*. Uobičajeni uslovi su: dostignut maksimalni broj generacija, najbolja jedinka je ponovljena zadati broj puta, sličnost jedinki u generaciji je veći od zadate granice, dokazana je optimalnost najbolje jedinke (ukoliko je to moguće), najbolja jedinka je dostigla optimalno rešenje (u koliko je ono unapred poznato), ograničeno vreme izvršavanja, prekid izvršavanja od strane korisnika i drugi. U praksi se ipak pokazalo, da najbolje rezultate daje njihovo proizvoljno kombinovanje, jer se tako smanjuje mogućnost loše procene prekida GA.

3.2.8 Ostali aspekti GA

Odabir *politike zamene generacije* može u velikoj meri uticati na performanse GA. *Generacijski GA* predviđa da se u svakoj generaciji promeni čitava populacija novim jedinkama (potomcima). U *stacionarnom GA* u svakoj generaciji se generiše samo deo populacije, dok ostatak prelazi direktno u narednu generaciju. Kod *elitičke strategije* samo jako mali broj i to najboljih (elitnih) jedinki može direktno preći u narednu generaciju bez primene genetskih operatora i računanja njihove prilagođenosti. Na taj način se obezbeđuje čuvanje dobrih rešenja i vrši ušteda procesorskog vremena. Detaljnije o ovom aspektu GA moze se naći u radovima [Sys91] i

[Smi93].
Pri izvršavanju GA je primećeno da fiksni izbor parametra nije uvek najpogodniji. Različite parametre kao na primer: veličinu početne populacije, nivo mutacije, nivo ukrštanja i druge je moguće promeniti i u toku izvršavanja GA. Kod *fiksne promene parametara* se unapred zadaje smer promene (povećanje ili smanjenje vrednosti tokom generacija) kao i formula po kojoj se promena vrši. *Adaptivna promena parametra* se vrši na osnovu dotadašnje uspešnosti i rezultata koji je dao. Detaljnije o različitim načinima promene parametara može se naći u radovima: [Bra91], [Bee92a], [Bee93], [Sri94].
Još jedan jako bitan aspekt GA je da se može kombinovati i sa drugim *heuristikama* radi poboljšanja performansi. Heuristike su metode koje se koriste kod rešavanja kompleksnih problema gde je korišćenje determinističkih tehnika, kao na primer linearnog programiranja, gradijentnih metoda i drugih, nije moguće. U opštem slučaju, one daju samo približno rešenje ali je vreme izvršavanja kratko. Do sada je razvijen čitav niz najraznovrsnijih heurističkih metoda koje su zavisne od prirode problema, kao i opšti heuristički pristupi- *metaheuristike*. Najčešće korišćene metaheuristike su: *tabu pretraživanje* [Glo86], [Glo90], *simulirano kaljenje* [Kir83], *Lagranževa relaksacija* [Geo74], *metoda promenljivih okolina* [Mla95], [Mla97], *neuralne mreže* [Fan90], *mravlji sistemi* [Dor96] i drugi. Ovde treba naglasiti da je i sam GA jedna vrsta metaheuristika i da u kombinaciji sa još jednom ili više drugih može uticati ne samo na poboljšanje početne populacije, već i svake naredne generacije.

Glava 4

Neodređenost

Osnovni cilj optimizacije je da nađe rešenja koja najbolje reprezentuju vrednosti funkcija cilja, kao sto su globalna ili Pareto optimalna rešenja. U praksi, ipak, rešenja se mogu razlikovati od svojih teoretskih vrednosti i mogu rezultirati u drugačijem skupu vrednosti funkcija cilja. Nije redak slučaj da se rešenja ne mogu dobiti sa željenom tačnošću, kao posledica njihove osetljivosti na parametarska variranja. U nekom eksperimentu usled grešaka u merenju, izmerene vrednosti se mogu razlikovati od stvarnih. To se posebno može primetiti ako eksperiment ponavljamo uzastopno više puta, a dobijamo donekle različite vrednosti. Na primer, vrednost izmerenog krvnog pritiska kod pacijenta može poprilično varirati u kratkom vremenskom intervalu. Zbog ovakvih devijacija- *neodređenosti*, traženje globalnog optimuma je postalo manje zanimljivo, pogotovu ako je takvo rešenje isuviše osetljivo na male promene parametara. Umesto toga, odskora poseban značaj je dat traženju *robusnih rešenja*. Ona su relativno neosetljiva na takve promene i mnogo su prikladnija sa praktične strane gledišta. Različite tipove neodređenosti možemo podeliti u dve glavne klase:

- za iste ulazne promenljive, funkcije cilja se neznatno razlikuju i/ili
- ulazne promenljive nisu tačne i variraju do određene tolerancije.

Pri modeliranju ovih neodređenosti, prvu nazivamo *šum*, a drugu *robusnost*. Nalaženje robusnog rešenja u prisustvu ovakvih neodređenosti ima ogroman značaj. Do sada su publikovani neki radovi koji uključuju razvoj strategija za robusnu optimizaciju kako za jedan tako i za višekriterijumski slučaj. U ovom odeljku je dat pregled postojećih pristupa u rešavanju različitih neodređenosti. Pored toga, data je i primena već poznatih pristupa za robusnost na slučaj šuma a uvedena je i nova definicija robusnosti.

4.1 Robusnost

Prilikom optimizacije realnih problema, različiti parametri pri modeliranju mogu biti neodređeni, kao na primer promenljive ili vrednosti funkcije cilja. Posmatrajmo slučaj gde su promenljive predmet variranja. Zbog tolerancije u proizvodnji promenljive se mogu predstaviti samo do određenog stepena tačnosti. Kao posledica toga, one nisu egzaktne. Ovu vrstu neodređenosti nazivamo *robusnost*.

4.1.1 Formalna definicija

Robusnost se može definisati kao funkcija cilja f u zavisnosti od perturbacije δ i promenljive x:

$$f_{rob}(x) = f(x + \delta) \tag{4.1}$$

gde $f(x + \delta)$ ne predstavlja očekivanu funkciju prilagođenosti, već slučajnu promenljivu koja opisuje nepoznatu raspodelu prilagođenosti.

4.1.2 Različiti načini za razmatranje robusnosti

Postoji mnoštvo radova posvećenih robusnosti u jednokriterijumskoj optimizaciji, međutim samo nekolicina istražuje robusnost u evolucionoj višekriterijmskoj optimizaciji. Zajedničko za te radove je da pokušavanju da pronađu robusna rešenja, koja su manje osetljiva na bilo kakvo parametarsko variranje. Ovde je akcenat dat na perturbacije promenljivih u njihovoj okolini. U ovom poglavlju, različiti već postojeći koncepti koji razmatraju robusnost će biti razmatrani i određena poboljšanja će biti data. Takođe biće predložena i nova definicija robusnosti, dok će njena primena nešto kasnije, u poglavlju 6, biti demonstrirana na odabranim test problemima.

Bez gubitka opštosti, posmatrajmo slučaj jednokriterijumske optimizacije sledećeg tipa:

$$min\ f(x) \tag{4.2}$$

i pretpostavimo da je x^o optimalno rešenje gore navedenog problema. Robusno rešenje x^* je definisano kao ono koje je neosetljivo na perturbacije promenljivih u njihovoj okolini i samim tim ne mora nužno da odgovara x^o. Dakle, umesto rešavanja problema optimizacije (4.2), mi želimo da razmotrimo drugačiju formulaciju koja će da uzme u razmatranje ponašanje funkcije u okolini rešenja.
U literaturi se mogu naći sledeći pristupi:

Postojeći načini rešavanja

- Kao direktna primena prethodne definicije (4.1), pronalaženje robusnog rešenja se može svesti na integraciju očekivane funkcije prilagođenosti. Ona zavisi od verovatnoće raspodele $p(\delta)$ mogućeg pomeranja δ, za koji se pretpostavlja da je po svakoj promenljivoj nezavisno i da ima normalnu raspodelu.

$$min\ F(x) = \int_{-\infty}^{+\infty} f(x+\delta)p(\delta)d\delta \tag{4.3}$$

  $F(x)$ se naziva *efektivna funkcija prilagođenosti* i ovaj termin se prvi put pominje u radu [Tsu96] i predstavlja ekvivalent očekivanoj funkciji prilagođenosti f sa parametrom $x+\delta$. Ovaj pristup se može naći u literaturi [Jin05].

- Slična ideja bi bila optimizacija srednje vrednosti efektivne funkcije umesto originalne funkcije cilja. Deb i Gupta u [Deb06] definišu robusna rešenja u prisustvu robusnosti za jedan i višekriterijumski slučaj:

  Definicija 1 (Robusno rešenje I tipa jednokriterijumske optimizacije): Za problem minimizacije funkcije cilja $f(x)$, rešenje x^* se naziva *robusno rešenje I tipa* ako je to globalni minimum srednje vrednosti efektivne funkcije $f^{eff}(x)$ definisane u zavisnosti od δ okoline na sledeći način:

  $min\ f^{eff}(x) = \frac{1}{|B_\delta(x)|}\int_{y\in B_\delta(x)} f(y)dy,\ x \in S$

  gde je $B_\delta(x)$- δ okolina rešenja x, $|B_\delta(x)|$ hipervolume okoline i S prostor pretrage.

  Definicija 2 (Robusno rešenje I tipa višekriterijumske optimizacije): Rešenje x^* se naziva *robusno rešenje I tipa višekriterijumske optimizacije* ako predstavlja globalno izvodljivo Pareto optimalno rešenje za sledeci višekriterijumski problem minimizacije:

  $min\ (f_1^{eff}(x), f_2^{eff}(x), ..., f_M^{eff}(x)),\ x \in S$

  gde je $f_j^{eff}(x)$ definisano kao: $f_j^{eff}(x) = \frac{1}{|B_\delta(x)|}\int_{y\in B_\delta(x)} f_j(y)dy$.

- treći način bi bio optimizacija originalne funkcije cilja ali sa ograničenjem da je norma razlike između perturbovane funkcije i funkcije cilja manja od zadatog praga η. Ovu definiciju su takođe predložili Deb i Gupta u [Deb06] i njena formulacija glasi:

  Definicija 3 (Robusno rešenje II tipa jednokriterijumske optimizacije): Kod problema minimizacije funkcije cilja $f(x)$, rešenje x^* se naziva *robusno rešenje II tipa* ako je globalno rešenje sledećeg problema:

  $min\ f(x)$
  $\frac{\|f^p(x)-f(x)\|}{\|f(x)\|} \leq \eta,\ x \in S$
  gde je f^p perturbovana funkcija, $\|.\|$ norma i η- ograničavajući parametar.

  Definicija 4 (Robusno rešenje II tipa višekriterijumske optimizacije): Rešenje x^* se naziva *robusno rešenje II tipa višekriterijumske optimizacije* ako je globalno izvodljivo Pareto optimalno rešenje sledećeg višekriterijumskg problema optimizacije:

  $min\ f(x) = (f_1(x), f_2(x), ..., f_M(x))$
  $\frac{\|f^p(x)-f(x)\|}{\|f(x)\|} \leq \eta,\ x \in S$

Novi pristup

- U ovom radu je predložen novi pristup u traženju robusnih rešenja u prisustvu robusnosti. Ideja bazirana na intervalu poverenja daje nam procenu opsega vrednosti u kome bi trebalo da se nalaze tražene vrednosti naših parametara. Koristeći istovremeno ograničenja data u Debovoj definiciji konačna definicija glasi:

  $min\ f(x)$
  $P\left\{\frac{\|f(sample)-f(x)\|}{\|f(x)\|} \leq \eta\right\} \geq \gamma,\ x \in S$
  gde je η konstanta iz Debove definicije, čija je vrednost jako bliska nuli, i γ nova konstanta bliska jedinici.

4.1.3 Različiti načini da se odredi prilagođenost

Formalne definicije očekivane funkcije prilagođenosti su date u odeljku 4.1.2. Međutim u većini slučajeva, ova funkcija nije dostupna u bliskom obliku i njeno analitičko računanje bi bilo teško ili čak nemoguće izvodljivo. Zbog toga, prilagođenost moramo da aproksimiramo na osnovu podataka generisanih putem eksperimenta ili simulacije.

Jedan od jednostavnijih a pritom jako delotvoran način je *metod Monte Karlo* (tvorci- John von Neumann i Stanislaw Ulam 40-tih godina prošlog veka). To je algoritam široke namene koji rezultate dobija uzastopnim odabirom slučajnih uzoraka. U našem konkretnom slučaju može se primeniti za aproksimacije vrednosti funkcije prilagođenosti $f(x + \delta)$. Uprkost fleksibilnosti, ova metoda zahteva puno iteracija (ponavljanja) kako bi se postigla željena tačnost. Da bi se smanjili troškovi računanja (povećala brzina izvršavanja) neophodno je smanjiti broj uzoraka.

Nešto drugačiji pristup bi bio uvećanje populacije. To može umanjiti uticaj robusnosti i samim tim osigurati ispravnu konvergenciju algoritma. Teorijski je pokazano da je GA sa beskonačnom populacijom, gde je korišćena proporcionalna selekcija i dodavana perturbacija promenljivih u svakoj generaciji ekvivalentno optimizaciji očekivane funkcije prilagođenosti. Dobar pregled o tome je dat u radovima [Tsu96] i [Tsu96].

4.1.4 Implementacija

U jednokriterijumskoj optimizaciji postoje mnogobrojni radovi koji su posvećeni nalaženju robusnih rešenja.

Branke u [Bra98] predlaže nekoliko heuristika koje modifikuju standardni EA kako bi proizveo više robusnih rešenja.
Jin i Branke [Jin03] sugerišu procenu robusnosti rešenja kao dopunski kriterijum optimizacije. Drugim rečima, traže kompromis između optimalnog i robusnog rešenja pri čemu nikakva dodatna aproksimacija prilagođenosti nije potrebna.

Tsutsui i Ghosh su u radu [Tsu97] prezentovali matematički model koji se bazira na teoriji šema za dobijanje robusnih rešenja.
Parme u [Par96] sugeriše hierarhijsku strategiju za traženje pojedinih regiona sa visokim performansama u kompleksnom prostoru pretrage.

U skorije vreme, objavljeno je nekoliko radova koji uzimaju u obzir višekriterijumski slučaj. Jin i Sendhoff [Jin03] ističu kako optimizacija očekivane funkcije prilagođenosti nije dovoljna u nekim slučajevima. Sa očekivanom funkcijom prilagođenosti kao jedinim kriterijumom, pozitivne i negativne devijacije od prave prilagođenosti se mogu međusobno poništiti u okolini traženog rešenja. Zbog toga oni predlažu uzimanje varijanse prilagođenosti kao dodatni kriterijum.

U radu [Ray02] potraga za robusnim rešenjima je rešena dodavanjem novih kriterijuma funkcije cilja. Za dodatne kriterijume se uzimaju prilagođenost, srednja vrednost i standardna devijacija prilagođenosti. Poslednja dva kriterijuma se računaju uzimanjem slučajnih uzoraka iz okoline.

Nedavno su Deb i Gupta [Deb06] definisali dve nove vrste robusnih rešenja (pomenuto u odeljku 4.1.2) kao i dva pristupa za pronalaženje rešenja optimizacije po datim definicijama. Prvi pristup predstavlja nadograđivanje tehnike koja je korišćena za jednokriterijumski slučaj- bazirana na optimizaciji srednje vrednosti efektivne funkcije cilje, dok druga omogućuje korisniku da kontroliše željenu količinu robusnostu u problemu.
U sledećem odeljku biće reči o još jednoj jako važnoj klasi neodređenosti- šumu. Postojeće i nove metode će biti prezentovane i razlike sa robusnošću će biti pokazane.

4.2 Šum

Prilikom uzastopnog izračunavanja funkcije cilja, dobijene vrednosti se mogu međusobno razlikovati usled promena uzrokovanih okruženjem. Na primer: promena radne temperature, pritiska, vlažnosti, različite osobine materijala, samo su neki od pokazatelja. Ovakvu vrstu neodređenosti zovemo *šum*. Ona se ne može ukloniti i dizajner treba u skladu sa njom da prilagodi svoj model.

4.2.1 Formalna definicija

Uobičajen način da se definiše šum je dodavanje greške ϵ na vrednost funkcije cilja, pri čemu se pretpostavlja da ta greška ima normalnu raspodelu i vrednost očekivanja nula:

$$f_{noise}(x) = f(x) + \epsilon,\ \epsilon \in N(0, \sigma^2) \tag{4.4}$$

4.2.2 Različiti načini za razmatrenje šuma

Određeni broj radova je posvećen problemu optimizacije u prisustvu šuma. Zajedničko za sve te radove je da pokušavaju da nađu rešenja čija se vrednost neće mnogo promeniti usled malog variranja vrednosti parametara. Ovakav tip rešenja nazivamo robusna. U ovom slučaju nas posebno zanimaju robusna rešenja koja su manje osetljiva na promene u vrednostima fukcije cilja.
U ovom poglavlju, navedeni su različiti koncepti koji razmatraju šum, a već poznate ideje za robusnost su ovde primenjene na slučaj šuma. Takođe, jedan novi pristup će biti definisan, a njegova primena će biti demonstrirana na odabranim test primerima u poglavlju 6.

Bez gubitka opštosti, posmatrajmo slučaj jednokriterijumske optimizacije sledećeg tipa:

$$min\ f(x) \tag{4.5}$$

i pretpostavimo da je x^o optimalno rešenje gore navedenog problema. Robusno rešenje x^* je definisano kao ono koje je neosetljivo na promene vrednosti funkcije cilja. Pošto je šum uzet u obzir, ovo robusno rešenje ne mora nužno odgovarati vrednosti x^o. Dakle, umesto rešavanja problema optimizacije (4.5), ideja je da razmotrimo drugačiju formulaciju koja bi uzimala u obzir šum.
U literaturi se mogu naći sledeći pristupi:

Postojeći načini rešavanja

- Kao direktna primena prethodne definicije (4.4), umesto optimizacije funkcije cilja možemo optimizirati funkciju prilagođenosti koja je pod uticajem šuma. Šum u funkciji prilagođenosti može biti prouzrokovan osetljivim merenjem ili usred simulacije. Formalno se to može predstaviti na sledeći način:

$$min\ f(x) = \int_{-\infty}^{\infty} [f(x) + \epsilon] p(\epsilon) d\epsilon \tag{4.6}$$

gde je x promenljiva, $f(x)$ funkcija prilagođenosti i ϵ dodati šum, za koji se obično uzima da ima normalnu raspodelu sa očekivanjem nula i disperzijom σ^2.

Nova razmatranja i pristupi

- Slična ideja bi bila- optimizacija srednje vrednosti efektivne funkcije umesto originalne funkcije cilja. Deb i Gupta u [Deb06] su definisali robusno rešenje u prisustvu robusnosti. Mi čemo pokušati da ove definicije prilagodimo za slučaj šuma. Podsećanja radi, originalna formulacija glasi:

Definicija 1 (Robusno rešenje I tipa jednokriterijumske optimizacije): Za problem minimizacije funkcije cilja $f(x)$, rešenje x^* se naziva *robusno rešenje I tipa* ako je to globalni minimum srednje vrednosti efektivne funkcije $f^{eff}(x)$ definisane u zavisnosti od δ okoline na sledeći način:

$min\ f^{eff}(x) = \frac{1}{|B_\delta(x)|} \int_{y \in B_\delta(x)} f(y) dy,\ x \in S$

gde je $B_\delta(x)$- δ okolina rešenja x, $|B_\delta(x)|$ hipervolume okoline i S prostor pretrage.
Ako koristimo definiciju šuma (4.4), gde smo funkciji cilja dodali grešku za koju smo pretpostavili da ima normalnu raspodelu sa očekivanjem nula, u tom slučaju efektivna funkcija postaje ekvivalentna funkciji cilja

$$f^{eff}(x) = E[f(X) + \epsilon] = E[f(X)] + E[\epsilon] = f(x) \tag{4.7}$$

i optimizacija srednje vrednosti efektivne funkcije odgovara optimizaciji originalne funkcije cilja (4.5)

$$min\ f^{eff}(x) = min\ f(x)$$

- Optimizacija funkcije (4.5) ali sa ograničenjem da je norma razlike između perturbovane i originalne $f(x)$ manja od zadatog praga η. Ova definicija je predložena od strane Deb i Gupta u [Deb06] i da se podsetimo njena originalna formulacija za slučaj robusnog rešenja u prisustvu robusnosti je glasila:

  Definicija 3 (Robusno rešenje II tipa jednokriterijumske optimizacije): Kod problema minimizacije funkcije cilja $f(x)$, rešenje x^* se naziva *robusno rešenje II tipa* ako je globalno rešenje sledećeg problema:

  $min\ f(x)$
  $\frac{\|f^p(x)-f(x)\|}{\|f(x)\|} \leq \eta,\ x \in S$ (4.8)
  gde je f^p perturbovana funkcija, $\|.\|$ norma i η- ograničavajući parametar.
  Za vrednost perturbovane funkcije f^p možemo izabrati srednju vrednost efektivne funkcije f^{eff}. U tom slučaju, na osnovu prethodne diskusije, perturbovana funkcija je ekvivalentna funkciji cilja:

  $$f^p(x) = f^{eff}(x) = f(x)$$

  Ovo bi vodilo celu jednačinu (4.8) da bude jednaka nuli. Kao alternativu, možemo zameniti perturbovanu funkciju sa najgorom vrednošću funkcije, tada bi jednačina težila beskonačnosti. Na osnovu ovoga možemo zaključiti da ovaj pristup nije naročito pogodan u slučaju šuma. U daljem tekstu biće predloženi novi pristupi koji bi trebalo da nadoknade nedostatke koji su ovde primećeni.

- Zamena funkcije prilagođenosti nekim drugim vrednostima osim srednje vrednosti je takođe uzeta u razmatranje. Vodeći se pretpostavkom da šum ima normalnu raspodelu i nulu za očekivanje nikakvu kvalitativnu razliku ne bi dobili ako uzmemo medijanu (posto je jednaka nuli). Druga ideja bi bila da razmotrimo centralni moment, koja bi mogla da da bolje rezultate, ali to izlazi izvan okvira ovog rada.

- U ovom radu je predložen novi pristup u traženju robusnih rešenja u prisustvu šuma. Ideja bazirana na intervalu poverenja daje nam procenu opsega vrednosti u kome bi trebalo da se nalaze tražene vrednosti naših parametara. Koristeći istovremeno ograničenja data u Debovoj definiciji konačna definicija glasi:

  $min\ f(x)$
  $P\left\{\frac{\|f(sample)-f(x)\|}{\|f(x)\|} \leq \eta\right\} \geq \gamma,\ x \in S$
  gde je η konstanta iz Debove definicije, čija je vrednost jako bliska nuli, i γ nova konstanta bliska jedinici.

4.2.3 Različiti načini da se odredi prilagođenost

Formalno računanje očekivane funkcije prilagođenosti je dato jednačinom (4.6). Međutim u većini slučaja ovako teoretski zadata funkcija u praksi nije rešiva. Zbog toga, prilagođenost moramo da aproksimiramo na osnovu podataka generisanih putem eksperimenta ili simulacije. Dva naredna pristupa sumirana u radu [Jin05] pokušavaju da na eksplicitan način dođu do rešenja.

Kao što je ranije pomenuto, najčešći pristup za aproksimaciju prilagođenosti je preko nalaženja proseka slučajnih uzoraka uzetih tokom vremena. Uzorkovanje n puta reducira odgovarajuću standardnu devijaciju za faktor $\sqrt{n}$, ali sa druge strane povećava vreme izvršavanja za faktor n. Otuda, možemo da biramo između funkcije zahvaćene šumom čije je vreme izvršavanja brzo ili tačnije funkcije prilagođenosti čije je vreme izvršavanja sporije. Drugim rečima, povećanjem broja uzoraka uticaj šuma se smanjuje, ali sa druge strane veliki broj uzoraka usporava proces. Rešenje je u nalazenju pravog balansa tako da se očuva kvalitet performansi.

Drugi način bi bio nalaženje proseka uzoraka biranih među tačkama koje se nalaze u okolini naše tražene tačke. U radovima [Bra01] i [San00] je pokazano da šum u okolini ima iste karakteristike kao i šum posmatrane tačke.

U daljem tekstu razmatrajmo različite pristupe kako bismo došli do rešenja koja će optimizovati pomenute definicije robusnih rešenja.

4.2.4 Implementacija

Fitzpatrick i Grefenstette u svom radu [Fit88] predlazu korišćenje velike populacije kako bi se smanjio uticaj šuma. Oni su takođe zaključili da je bolje povećavati veličinu populacije nego uvećati broj uzoraka.

Sa druge strane Beyer u [Bey93] i godinu kasnije Hammel i Baeck u radu [Ham94] su potpuno drugačijeg mišljenja, što su u ovim radovima i empirijski dokazali.

Ovako oprečne zaključke pomirili su teoretski modeli koji su razvijeni i omogućavaju istovremenu optimizaciju veličine populacije i veličine uzorka [Mil96] i [Mil97].

Brojni autori sugerišu modifikaciju selekcije kako bi se oslobodili šuma. Markon [Mar01] predlaže korišćenje treshold-a u selekciji, Teich i Hughes [Tei01], [Hug01]- Pareto dominaciju- koja se pokazala primenljiva kako za jedno tako i za višekriterijumsku optimizaciju.

Ostale heuristike su takođe uzele šum u razmatranje, ali su komparativne studije pokazale da evolucione strategije imaju jasnu prednost u takvim sredinama [Bau95], [Arn03].

Glava 5

Višekriterijumski GA baziran na hipervolume indikatoru

5.1 Zadatak implementacije

U cilju nalaženja rešenja koja su robusna na osnovu definicija datih u prethodnom poglavlju ovde je prikazana modifikacija postojećeg GA baziranog na hipervolume indikatoru [Zit07]. Na taj način dodata je nova definicija robusnosti koja će biti upoređena sa već postojećim definicijama poznatih iz literature. Takođe važnu novinu u implementaciji predstavlja Pareto dominacija primenjena na upoređivanje skupova rešenja tako da je potom moguća direktna primena hipervolume indikatora.

5.2 Osnovna implementacija

Postojeća GA implementacija, kao i dodati delovi, je napisana u programskom jeziku JAVA. U daljem tekstu biće navedene i kraće opisane sve važnije postojeće klase koje su korišćene, dok će nove klase i funkcije biti detaljno opisane u sledećem odeljku.
Početna populacija se slučajno generiše iz intervala $(0, 1)$. Vrednosti funkcija cilja se računaju na osnovu funkcija datih u test problemima: ntp1, ntp2, ntp3 i ntp7. Pritom treba naglasiti da se radi o višekriterijumskoj maksimizaciji.

Klase *Individual* i *Population* služe za formiranje kako početne tako i svake naredne, poboljšane populacije tokom ciklusa GA. One takođe sadrže metodu za računanje vrednosti funkcija cilja kao i metodu za međusobno poređenje jedinki. Ovde je vazno istaći da se vrednosti funkcija cilja poistovećuju sa vrednostima funkcije prilagođenosti. Poređenje jedinki se onda

vrši upoređivanjem nizova tih vrednosti.

EfficiencyFormula je interfejs koji sadrži samo jednu funkciju koja služi za implementaciju različitih definicija robusnosti. Tako njena naslednica, klasa *Deb* implementira Debovu definiciju robusnog rešenja II tipa, a u klasi *Marija* biće implementirana nova definicija. Njihove matematičke formulacije date su u odeljku (4.1.2).

Algorithm predstavlja roditeljsku klasu za sve naslednice koje implementiraju različite algoritme poznate iz literature. U ovom radu izkorišćeni su postojeći algoritmi: *constraint* i *simann* (simulirano kaljenje). Pored standardnih operatora ukrštanja i mutacije, ovi algoritmi koriste posebnu vrstu selekcije- environmental selection, detaljno opisane u radu [Zit04]. Kao novina, tom spisku algoritama biće pridodat *MarijaAlgorithm* koji nudi određena poboljšanja.

Paket *Selector* sadrži delove optimizacionog procesa koji su nezavisni od konkretnog optimizacionog problema i koji se uglavnom baziraju na operatoru selekcije. Ovde je korišćen algoritam *SIBEA* [Zit07], koji koristi standardni hipervolume, opisan u odeljku (2.3).

U paketu *Variator* su date instance. Svaka klasa predstavlja zaseban test problem. Detaljan opis izabranih problema može se naći u odeljku (6.1).

Programski kod algoritma sadrži i posebne klase koje omogučavaju čitanje ulaznih podataka kao i štampanje dobijenih rezultata.
Detaljan opis cele implementacije izlazi iz okvira ovog rada i može se naći u [Bad08], [Bad09], [Zit08].

5.3 Novine u implementaciji

Kao što je već pomenuto, novine u radu predstavljaju: nova definicija robusnosti kao i algoritam koji pored ove definicije implementira i nov način poređenja skupova robusnih rešenja.

Matematička formulacija nove definicije je data u odeljku (4.1.2). Implementacija je realizovana u funkciji calc() u okviru nove klase Marija na sledeći način: prvo se porede razlike vrednosti funkcije cilja uzoraka i funkcije cilja stvarnih promenljivih sa Debovom konstantom η. Svaki put kada je taj uslov zadovoljen brojač se uvećava za jedan. Potom se poredi vrednost brojača podeljena sa brojem uzoraka sa novom konstantom γ. Ako je i taj uslov zadovoljen vraćamo niz vrednosti funkcija cilja, u suprotnom vrednosti niza postavljamo na beskonačno.

U ovom radu, ponuđene su i dve varijante algoritma: marija(n) i marija. Oba koriste novu definiciju robusnih rešenja, dok je samo u marija(n) iskorišćena nova funkcija poređenja pod nazivom dominatedBy(). Za svaku promenljivu generiše se skup od N uzoraka na koji se onda primenjuju funkcije cilja. Tako dobijene matrice vrednosti se porede sa svakim elementom odgovarajuće matrice jedinke sa kojom se poredi. Svaki put kada je vrednost funkcije uzoraka tekuće jedinke manja ili jednaka od vrednosti funkcije upoređujuće jedinke, uvećavamo brojač za jedan. Ako na kraju vrednost brojača bude manja ili jednaka od $(1-\beta)N^2$, gde je β nova konstanta bliska jedinici, onda funkcija vraća logičku vrednost true.
Uzorkovanje se vrši u proizvoljno maloj okolini tražene tačke u zavisnosti od izabrane raspodele (uniformna, gaussian ili debova).

Oba ova algoritma bazirana su na već pomenutom SIBEA algoritmu koji prilikom selekcije koristi hipervolume indikator. Ovaj indikator rangira Pareto skupove rešenja i samo α najbolje rangiranih ulaze u formiranje naredne populacije.
U daljem tekstu data je JAVA implementacija pomenutih funkcija i klasa.

```
package bd.ea.efficiencies;

import bd.ea.Individual;
import bd.ea.Parameters;

public class Marija implements EfficiencyFormula {

private int tests;
String stat;

public Marija(int tests, String stat) {

this.tests = tests;
this.stat = stat;
}

public double[] calc(Individual ind) {

ind.calculateObjectiveValues(); //f(x)
ind.calculatePertubedObjectiveValues(tests); //fp(x)

double[] fp = null;
double[] f = ind.getObjValues();
double[] efficiency = new double[ f.length ];

final int divisions = Parameters.getInt("robust.algorithm.tests");
      // number of samples

int count = 0;
for( int i = 0; i < divisions; i++ ) {
fp = ind.getPertubedObjectiveValues().get(i);
if(typeIIformula(fp, f) <= Parameters.getDouble("robust.algorithm.deb.eta"))
count++;
}

if( !ind.isInfeasible() &&
     ((count + 0.0) / divisions >= Parameters.getDouble("marija.gamma")))
for(int e=0; e < f.length; e++)
efficiency[e] = f[e];
else
for( int e = 0; e < f.length; e++)
efficiency[ e ] = Double.POSITIVE_INFINITY;

return efficiency;
}
```

```
public double typeIIformula( double[] fp, double[] f ) {
double n = 0.0;
double d = 0.0;

for( int i = 0; i < fp.length; i++ ) {
n += Math.pow( fp[i] - f[i], 2);
d += Math.pow( f[i], 2);
}
n = Math.sqrt(n);
d = Math.sqrt(d);

return n/d;
}

}
-----------------------------------------------------------------
public boolean dominatedBy(final Individual contendor) {
boolean equal = true;
boolean abetter = false;

final double[] here = getEfficiency();
final double[] comp = contendor.getEfficiency();

String type = Parameters.getString("marija.type");

if (!type.equalsIgnoreCase("new")) {

for (int i = 0; i < here.length && !abetter; i++) {
Double h = new Double(here[i]);
Double c = new Double(comp[i]);
abetter = h.compareTo(c) < 0;
equal = (h.compareTo(c) == 0) && equal;
}
return !equal && !abetter;
} else {
```

```
// novi kod
final double beta = Parameters.getDouble("marija.beta"); //beta = 0.99
final int divisions = Parameters.getInt("robust.algorithm.tests");

for (int k = 0; k < pertubedObjectiveValues.get(0).length && !abetter; k++) {
int count = 0;
for (int i = 0; i < divisions; i++)
for (int j = 0; j < divisions; j++)
if (pertubedObjectiveValues.get(i)[k]
    <= contendor.pertubedObjectiveValues.get(j)[k])
count++;
abetter = count >= (1 - beta) * divisions * divisions;
}
return !abetter;
}
}

-----------------------------------------------------------------
package bd.ea.algorithms.oneclass;

import bd.ea.Population;
import bd.ea.algorithms.Algorithm;
import bd.ea.efficiencies.EfficiencyFormula;
import bd.selector.Hypervolume;

public class MarijaAlgorithm extends Algorithm {

public MarijaAlgorithm(EfficiencyFormula ef) {
super(ef);
}

public Population envSelection( Population pop, int alpha, int generation) {

pop.calcEfficiency( ef );
      //calculates the efficiency values using function calc

/* After that, use standard environmental selection */
pop.setEfficiencyActive(null, true);

Hypervolume.select(pop, alpha);
      //normal hypervolume indicator to remove individuals

return pop;
}
}
```

Glava 6

Eksperimentalni rezultati

6.1 Formulacija test problema

Genetski algoritam je testiran na primerima preuzetih iz literature. Njihova matematička formulacija je data u nastavku:
NTP1:

$$f_1(x) = 10 + x(0) + S(g(x))$$
$$f_2(x) = 11 - x(0) + S(g(x))$$
$$g(x) = \frac{1}{n}\sum_{i=1}^{n} x_i$$
$$S(x) = 10\Big(x + \frac{sin(6\pi x)}{6\pi}\Big)$$

NTP2:

$$f_1(x) = 50 + 50x(0) + S(g(x))S(x(0))$$
$$f_2(x) = 100 - 50x(0) + S(g(x))S(x(0))$$
$$g(x) = \frac{1}{n}\sum_{i=1}^{n} x_i$$
$$S(x) = 8\Big(x + \frac{sin(10\pi x)}{10\pi}\Big)$$

NTP3:

$$f_1(x) = 9 + x(0) + T(x)$$
$$f_2(x) = 11 - \frac{sin(10\pi x(0))}{\pi + 11x(0)} + T(x)$$

$$g(x) = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$T(x) = 200\Big(x - \frac{4}{3}(x-0.5)^3 - \frac{1}{6}\Big)$$

NTP7:

$$f_1(x) = 100 + x(0) + S(g(x))$$
$$f_2(x) = 100 - x(0) + S(g(x))$$
$$S(x) = x + 1000x^6$$

6.2 Rezultati

6.2.1 Opis parametara

Testiranje je izvršeno na prethodno opisanim primerima: ntp1, ntp2, ntp3 i ntp7 i primenjene su četiri varijante algoritma: constraint, marija, marija(n) i simann, čije su najvažnije klase i funkcije takođe već date u prethodnom poglavlju. Ovde će biti navedene informacije o korišćenim parametrima:

- Broj jedinki, alpha=20
- Broj funkcija cilja, dim=2
- Težina problema, nrdecvars=20
- Broj generacija, nrofgenerations=100
- Vrsta algoritma, algorithm=constraint/marija/marija(n)/simann
- Debova konstanta, deb.eta=0.01
- Vrsta test problema, problem=ntp1/ntp2/ntp3/ntp7
- Ograničujući parametar, beta=0.99
- Ograničujući parametar, gamma=0.99

6.2.2 Tabelarni prikaz rezultata

Primenom GA dobijeni su sledeći rezultati. U svakoj tabeli prva kolona predstavlja Index (broj od 0-19), zatim slede vrednosti funkcija cilja f1 i f2.

Tabela 6.1: ntp1-Constraint

Index	f_1	f_2
0	11.81949597407109	12.55010497966431
1	11.89292715787002	12.437940496809695
2	11.854725394570849	12.479864787788195
3	12.243925968410394	12.08999599139905
4	12.20274210267302	12.13038469547279
5	11.91953617161541	12.387262287886289
6	12.260650208426554	12.071072743038187
7	12.115822608787965	12.188689131901775
8	12.030560886150855	12.282839940199999
9	12.040385403622814	12.235841647361479
10	12.664440859447065	11.664440859447065
11	11.661756676807538	12.59910411733761
12	12.187750788542402	12.151402484873262
13	11.81949597407109	12.55010497966431
14	12.325756751909609	11.67935017412637
15	11.81949597407109	12.55010497966431
16	11.947922404731237	12.36610295332829
17	11.81949597407109	12.55010497966431
18	11.661756676807538	12.59910411733761
19	11.842463657866421	12.489689197637375

Tabela 6.2: ntp1-Marija

Index	f_1	f_2
0	12.588758302214565	11.744472246875844
1	12.481102865907268	11.851969327768483
2	12.621246622186234	11.690160207812838
3	12.666669049299026	11.666669049299026
4	12.100709919896465	12.216873004068129
5	12.255571048590895	12.07761194086735
6	12.544023620221521	11.790703870469807
7	12.237316846538246	12.096018232685136
8	11.530178667298513	12.530178667298513
9	11.530178667298513	12.530178667298513
10	12.412138067608833	11.952164505483136
11	11.766108316950039	12.221323123753233
12	12.554596001742635	11.78004040754977
13	12.563014128937418	11.768482755203756
14	12.30208152712317	12.035015950797087
15	12.10710928167619	12.167987672023601
16	12.422309484783682	11.895031275671286
17	12.219706101106569	12.14052854198271
18	12.318233592201773	12.016070334062993
19	11.59931712533426	12.421358700728032

Tabela 6.3: ntp1-Marija(n)

Index	f_1	f_2
0	11.968106080679197	12.327848977790799
1	11.771175136452712	12.56550878135197
2	12.520174832061398	11.892818831304028
3	12.425788747338489	11.907594704097251
4	12.384213525854543	11.949307696224077
5	11.904804777402688	12.394505282910092
6	12.003574261995404	12.309766082614715
7	11.764437035703127	12.591208917267052
8	11.807974292693336	12.459258417134153
9	11.775730505887466	12.557469256872288
10	11.93544234893818	12.382242466016868
11	11.664772089376733	12.664772089376733
12	11.71662523736661	12.616793761557844
13	12.203224130824696	12.150612353741545
14	12.0165977062412	12.29584728740415
15	11.687327097162978	12.64600712492436
16	11.894884111616143	12.436309173719145
17	12.124166421147647	12.245981754569643
18	12.055914135249655	12.271805890283105
19	12.659404976998092	11.659404976998092

Tabela 6.4: ntp1-Simann

Index	f_1	f_2
0	12.486790083441786	11.907742034050646
1	12.664071358854978	11.664071358854978
2	12.534550765458244	11.776102900428883
3	11.974937402657986	12.345669805941142
4	12.00600943631382	12.334955140012402
5	12.614128530021409	11.7309339747297
6	11.948676617170126	12.38553993215093
7	12.03008023786877	12.197432231960992
8	12.498370019154866	11.835071834937395
9	11.727283536758913	12.419457732031132
10	12.151828604811865	12.181562569509925
11	12.37389633474768	11.959400139439648
12	12.254315697253597	12.07268780177356
13	12.640898272634674	11.691463917567463
14	11.93530384150603	12.39806630259012
15	12.286173018308668	12.011449021629792
17	12.570517675780547	11.768307731815593
18	12.177276485844983	12.156380139938674
19	11.445602965688263	12.445602965688263

Tabela 6.5: ntp2-Constraint

Index	f_1	f_2
0	50.0	100.0
1	99.2951582025295	77.52047405977706
2	81.851965240052	85.34808370677219
3	86.04524205097317	80.64550552238231
4	64.94465718426974	92.52582830526552
5	110.90624474275874	71.47521448703989
6	73.1396000468347	88.01043014813975
7	69.23029212319545	90.40814154254619
8	104.09359695929246	75.48946380142726
9	55.117512029776606	97.77186561430602
10	56.78484081311662	96.14113032670824
11	80.14448007240618	85.96918825237803
12	52.020058655905466	99.55181440213087
13	97.0673264505065	78.26442470974658
14	59.781286487355544	94.15350591402776
15	83.30251790471782	83.63623376564009
16	105.39477591745742	74.24747815833328
17	116.67074976266306	66.67074976266306
18	50.0	100.0
19	101.1643309526628	76.621840184213

Tabela 6.6: ntp2-Marija

Index	f_1	f_2
0	50.0	100.0
1	54.544801708338746	98.87638560317473
2	83.13880055928465	86.03758387778586
3	71.67118400945178	89.92215700389964
4	85.90683993241096	83.30903788388453
5	101.36332871192353	75.55491933202636
6	63.178971742621066	93.92845954741512
7	110.25480604755502	73.99645161552966
8	115.75124864998	70.75119760333186
9	119.18542960780745	69.18542960780745
10	97.5590379170674	79.29022766694538
11	94.64126701695483	80.3554862216501
12	99.4847938803121	76.7273341897682
13	56.99903786525535	96.84716396376565
14	76.48170393657871	87.84233724904978
15	85.22701683529856	84.03099399568298
16	69.34477868044735	91.17499934810306
17	80.89745982729306	86.78309942415206
18	59.385642474468	94.6378536419792
19	89.44049669927193	81.92912454735048

Tabela 6.7: ntp2-Marija(n)

Index	f_1	f_2
0	50.0	100.0
1	55.93747321423532	97.4744156005658
2	58.76480451390849	95.36428666644606
3	57.393929810645695	96.37657050144443
4	59.58565484650964	94.85340996488904
5	87.29375594667323	72.86439244545272
6	116.07147571505661	69.79538154967389
7	52.94537478532574	99.2154251279767
8	87.11534320079578	78.74409225983383
9	54.083667911947394	98.98370977862191
10	62.570344214147546	94.01624280892213
11	72.82897199213909	84.169613251638
12	52.17886414493579	99.6489389094064
13	116.23370655924298	66.23370655924298
14	77.61237385573341	80.39283603347411
15	68.0754670454876	86.31746174602672
16	64.27876148192341	93.77643244148776
17	60.46142296652922	94.4658424133579
18	74.86226412116856	81.7937161015927
19	65.88643259988355	93.44569367802342

Tabela 6.8: ntp2-Simann

Index	f_1	f_2
0	50.0	100.0
1	56.83671993673458	96.89388662882723
2	54.55830335307146	98.83869720023289
3	75.76321828414228	87.96958472246124
4	58.23714327597291	95.70277899243733
5	119.15608603674424	69.15608603674424
6	109.85733881297475	71.28916724617105
7	102.16021824374552	75.59476780642622
8	97.6830752100861	78.7416150803517
9	105.38545253487162	73.95991109048997
10	86.79891136924927	82.7227952784361
11	99.36640554802051	77.31808425234132
12	80.99764413996003	86.30770304898398
13	89.29997896499555	81.35093928519859
14	72.34025667647941	89.03970036936883
15	65.243252600132	93.32366253292119
16	61.58900601753089	94.19663877573758
17	69.91200111458451	91.60869944949371
18	84.00631540175102	83.7620683093897
19	92.81943047753093	80.01455165560354

Tabela 6.9: ntp3-Constraint

Index	f_1	f_2
0	10.992528243461145	2.4737309150186917
1	10.218849162605148	6.25883770067277
2	10.979948437107653	4.540280883271757
3	10.218849162605148	6.25883770067277
4	10.218849162605148	6.25883770067277
5	10.218849162605148	6.25883770067277
6	10.218849162605148	6.25883770067277
7	10.218849162605148	6.25883770067277
8	10.218849162605148	6.25883770067277
9	10.218849162605148	6.25883770067277
10	10.218849162605148	6.25883770067277
11	10.218849162605148	6.25883770067277
12	10.218849162605148	6.25883770067277
13	10.218849162605148	6.25883770067277
14	10.218849162605148	6.25883770067277
15	10.218849162605148	6.25883770067277
16	10.218849162605148	6.25883770067277
17	10.218849162605148	6.25883770067277
18	10.218849162605148	6.25883770067277
19	10.295910342657793	5.792927484885769

Tabela 6.10: ntp3-Marija

Index	f_1	f_2
0	13.468275983244396	4.685549291104967
1	13.468275983244396	4.685549291104967
2	13.468275983244396	4.685549291104967
3	13.468275983244396	4.685549291104967
4	13.468275983244396	4.685549291104967
5	13.468275983244396	4.685549291104967
6	13.468275983244396	4.685549291104967
7	13.468275983244396	4.685549291104967
8	13.468275983244396	4.685549291104967
9	13.468275983244396	4.685549291104967
10	13.468275983244396	4.685549291104967
11	13.468275983244396	4.685549291104967
12	13.468275983244396	4.685549291104967
13	13.468275983244396	4.685549291104967
14	13.468275983244396	4.685549291104967
15	13.468275983244396	4.685549291104967
16	13.468275983244396	4.685549291104967
17	13.468275983244396	4.685549291104967
18	13.468275983244396	4.685549291104967
19	12.948708353671199	13.44420130579202

Tabela 6.11: ntp3-Marija(n)

Index	f_1	f_2
0	11.211474874696899	1.2114748746968995
1	11.420183234829652	1.4201832348296528
2	11.211474874696899	1.2114748746968995
3	11.211474874696899	1.2114748746968995
4	11.211474874696899	1.2114748746968995
5	11.211474874696899	1.2114748746968995
6	11.420183234829652	1.4201832348296528
7	11.420183234829652	1.4201832348296528
8	11.211474874696899	1.2114748746968995
9	11.420183234829652	1.4201832348296528
10	11.211474874696899	1.2114748746968995
11	11.211474874696899	1.2114748746968995
12	11.211474874696899	1.2114748746968995
13	11.420183234829652	1.4201832348296528
14	11.211474874696899	1.2114748746968995
15	11.211474874696899	1.2114748746968995
16	11.420183234829652	1.4201832348296528
17	11.420183234829652	1.4201832348296528
18	11.420183234829652	1.4201832348296528
19	11.211474874696899	1.2114748746968995

Tabela 6.12: ntp3-Simann

Index	f_1	f_2
0	11.75771515378273	6.068381402864539
1	11.483951032432223	7.444617410796172
2	12.81386944991715	2.81386944991715
3	13.53954342935071	3.6430844274016976
4	11.63403037374983	8.447960161224778
4	11.63403037374983	8.447960161224778
4	11.63403037374983	8.447960161224778
4	11.63403037374983	8.447960161224778
4	11.63403037374983	8.447960161224778
4	11.63403037374983	8.447960161224778
4	11.63403037374983	8.447960161224778
4	11.63403037374983	8.447960161224778
4	11.63403037374983	8.447960161224778
4	11.63403037374983	8.447960161224778
4	11.63403037374983	8.447960161224778
4	11.63403037374983	8.447960161224778
4	11.63403037374983	8.447960161224778
4	11.63403037374983	8.447960161224778
4	11.63403037374983	8.447960161224778
4	11.63403037374983	8.447960161224778
18	11.956997655758585	7.204037060801809
19	11.651104255928303	8.033660087208615

Tabela 6.13: ntp7-Constraint

Index	f_1	f_2
0	100.0603665287655	100.0603665287655
1	100.0603665287655	100.0603665287655
2	100.0603665287655	100.0603665287655
3	100.0603665287655	100.0603665287655
4	100.30189322967104	99.86890496252707
5	100.30189322967104	99.86890496252707
6	100.0603665287655	100.0603665287655
7	100.20287180923354	100.00534502388807
8	100.30189322967104	99.86890496252707
9	100.0603665287655	100.0603665287655
10	100.0603665287655	100.0603665287655
11	100.30189322967104	99.86890496252707
12	101.19820047967846	99.19820047967846
13	100.0603665287655	100.0603665287655
14	100.20287180923354	100.00534502388807
15	100.0603665287655	100.0603665287655
16	100.20287180923354	100.00534502388807
17	100.0603665287655	100.0603665287655
18	100.0603665287655	100.0603665287655
19	100.0603665287655	100.0603665287655

Tabela 6.14: ntp7-Marija

Index	f_1	f_2
0	100.20010339187014	99.9882454437362
1	100.06953818764103	100.06953818764103
2	101.19325139750646	99.19325139750646
3	100.48791872318161	99.74667807770193
4	100.4929446490279	99.75170400354823
5	100.4929446490279	99.75170400354823
6	100.4929446490279	99.75170400354823
7	100.4929446490279	99.75170400354823
8	100.4929446490279	99.75170400354823
9	100.4929446490279	99.75170400354823
10	100.4929446490279	99.75170400354823
11	100.4929446490279	99.75170400354823
12	100.4929446490279	99.75170400354823
13	100.4929446490279	99.75170400354823
14	101.70249326073338	99.70249326073338
15	100.4929446490279	99.75170400354823
16	100.4929446490279	99.75170400354823
17	100.4929446490279	99.75170400354823
18	100.4929446490279	99.75170400354823
19	100.07687997832046	100.07687997832046

Tabela 6.15: ntp7-Marija(n)

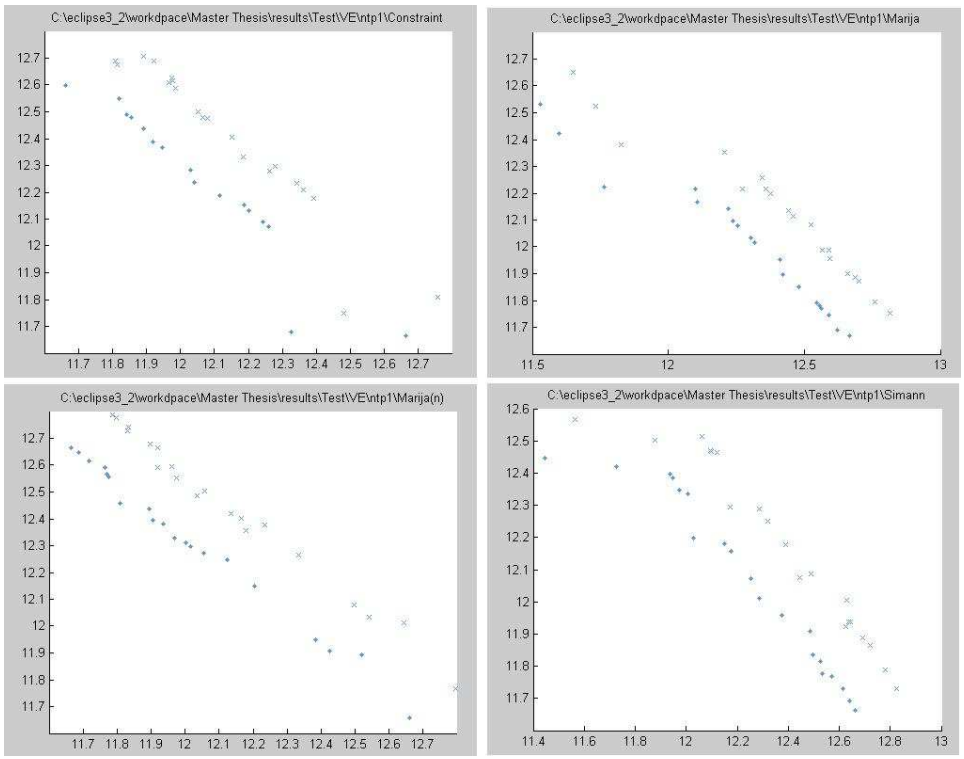
Index	f_1	f_2
0	100.2122635507022	100.0227471079572
1	100.22969822568649	99.9330481379657
2	100.13553236705216	100.04430285323984
3	100.22969822568649	99.9330481379657
4	100.13549764207265	100.04426812826033
5	100.22969822568649	99.9330481379657
6	100.2122635507022	100.0227471079572
7	100.73482678435106	99.60530280420481
8	100.13549764207265	100.04426812826033
9	100.2122635507022	100.0227471079572
10	100.13549764207265	100.04426812826033
11	100.2122635507022	100.0227471079572
12	100.13549764207265	100.04426812826033
13	100.07909788993953	100.07909788993953
14	100.13549764207265	100.04426812826033
15	100.2122635507022	100.0227471079572
16	100.21107766677089	100.02156122402589
17	100.2122635507022	100.0227471079572
18	100.2122635507022	100.0227471079572
19	100.2122635507022	100.0227471079572

Tabela 6.16: ntp7-Simann

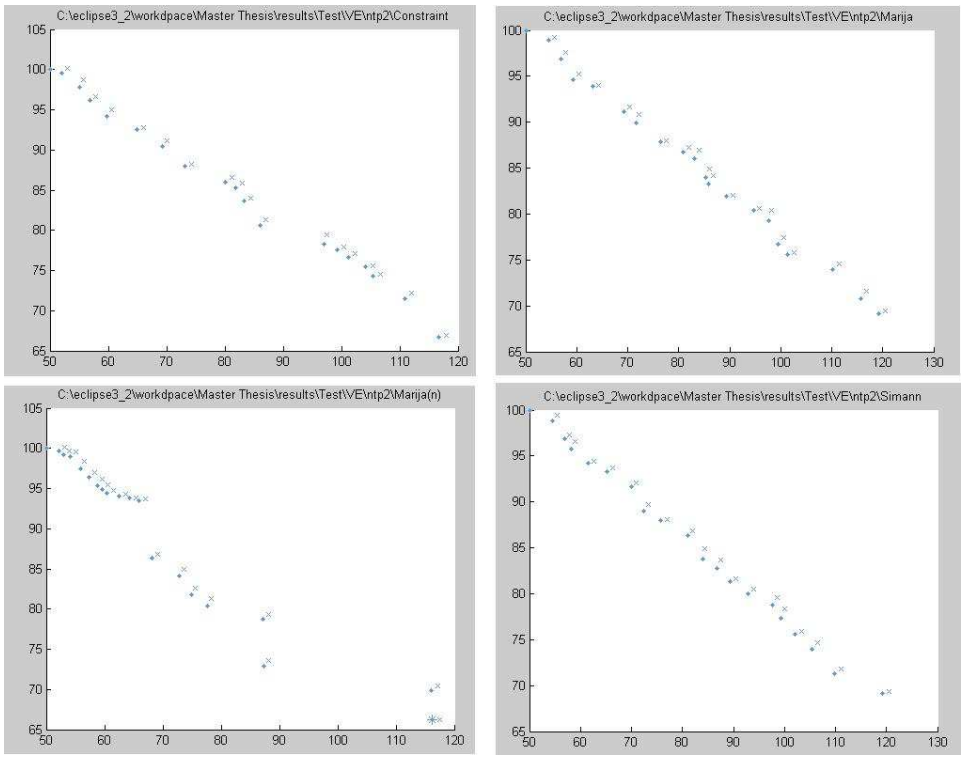
Index	f_1	f_2
0	100.07217040707071	100.07217040707071
1	100.07217040707071	100.07217040707071
2	100.07217040707071	100.07217040707071
3	100.07217040707071	100.07217040707071
4	100.07217040707071	100.07217040707071
5	100.14093497881582	100.06562873520484
6	100.15626848817097	100.05807424439847
7	100.14093497881582	100.06562873520484
8	100.07217040707071	100.07217040707071
9	100.07217040707071	100.07217040707071
10	100.07217040707071	100.07217040707071
11	100.07217040707071	100.07217040707071
12	100.07217040707071	100.07217040707071
13	100.07217040707071	100.07217040707071
14	100.07217040707071	100.07217040707071
15	100.14093497881582	100.06562873520484
16	100.07217040707071	100.07217040707071
17	101.27205447392632	99.27205447392632
18	100.07217040707071	100.07217040707071
19	100.07217040707071	100.07217040707071

6.2.3 Vizuelna ocena rešenja

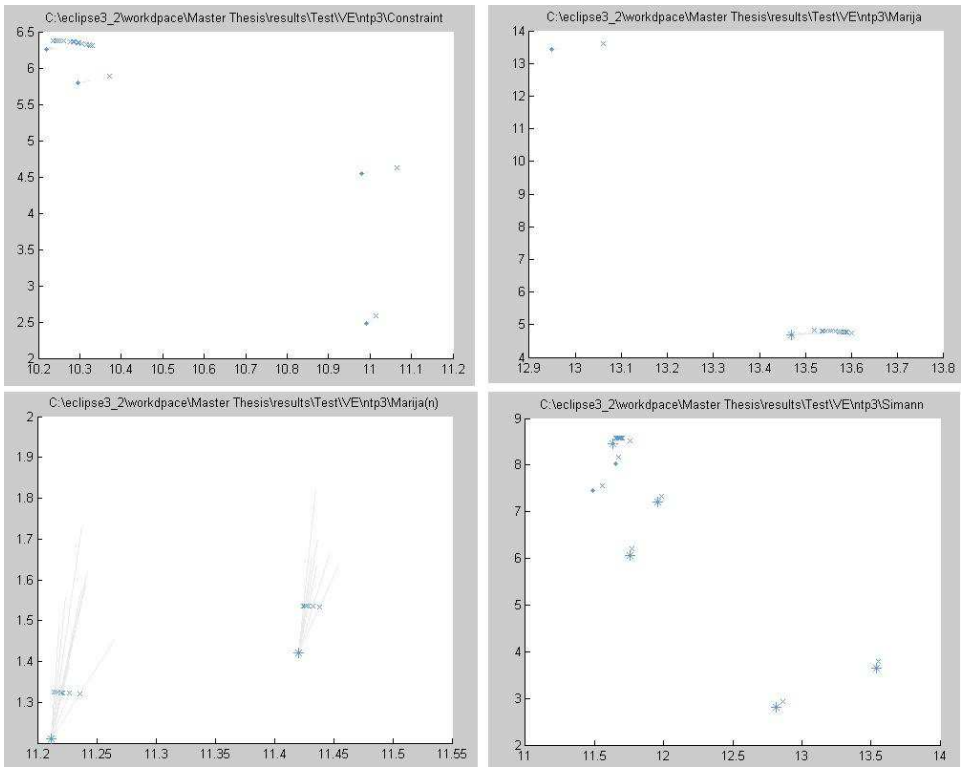
Jedan od bitnijih problema višekriterijumske optimizacije predstavlja komparacija performansi različitih algoritama. U slučaju višekriterijumskog GA, rezultat predstavlja aproksimaciju Pareto optimalnog skupa, tako da je pitanje kako oceniti kvalitet takvih skupova. Jedan od načina je da se svakom aproksimiranom skupu dodeli realni broj koji će reflektovati različite aspekte kvaliteta. Hipervolume indikator predstavlja jednu takvu meru i poslednjih godina je stekao izuzetnu popularnost.
U cilju bolje reprezentacije rezultata urađeni su plotovi sva četiri razmatrana test problema: ntp1, ntp2, ntp3 i ntp7 i upoređena četiri različita algoritma: constraint, marija, marija(n) i simann. Treba još naglasiti da su sa znakom iks označena ne robusna, a sa rombom robusna rešenja.



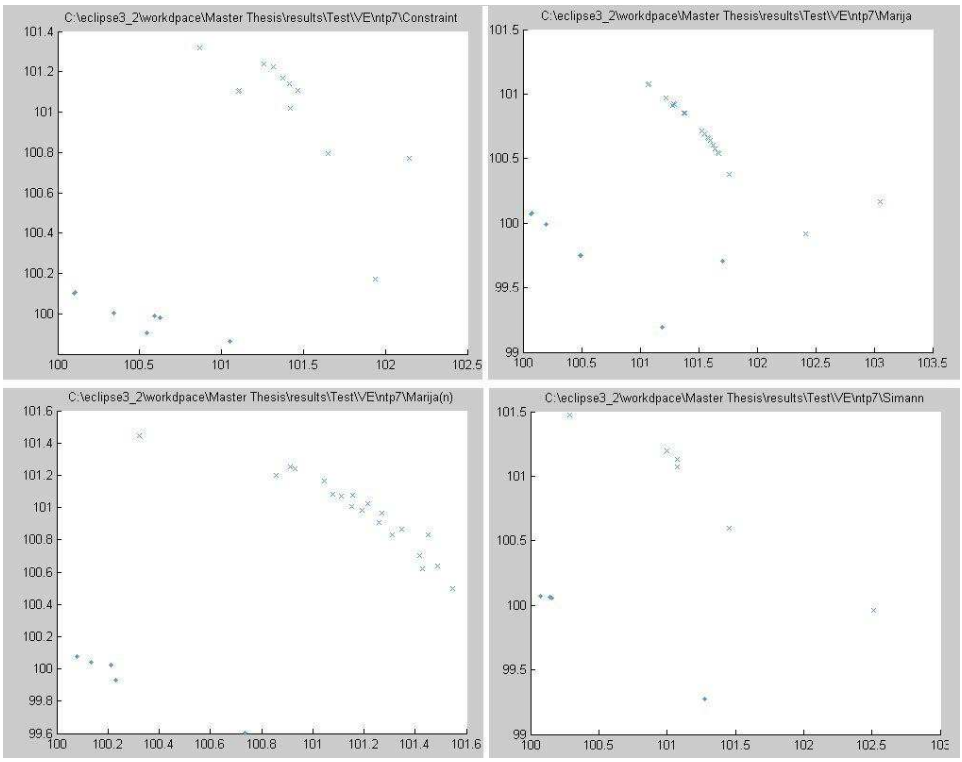
Slika 6.1: NTP1



Slika 6.2: NTP2



Slika 6.3: NTP3



Slika 6.4: NTP7

Analizirajući date slike 6.1, 6.2, 6.3 i 6.4, opšti zaključak je da novi algoritam (marija(n)) koji sadrži novu definiciju robusnosti kao i nov pristup za poređenje skupova rešenja, u proseku daje jako dobre rezultate. U test primeru NTP1 on daje čak i najbolje rezultate. U problemu NTP2 na početku (za manje vrednosti prve funkcije cilja) daje najbolje, a onda se algoritam marija pokazao kao bolji. U problemu NTP3 je teško bilo odrediti robusna rešenja, dok u problemu NTP7 algoritam constraint daje najbolje rezultate.

Glava 7

Zaključak

Ovaj rad je posvećen traženju robusnih rešenja u prisustvu neodređenosti. Nova definicija robusnosti uključuje i stohastičke karakteristike robusnih rešenja u višekriterijumskoj optimizaciji, što do sada nije bilo razmatrano. Na taj način se pojam robusnosti prirodno uklapa u definicije Pareto dominacije primenjene na skupove rešenja, a samim tim i na skup Pareto optimalnih rešenja, kao i na hipervolume indikator. Postojeći programski paket je efikasno i fleksibilno proširen novom definicijom uz očuvanje svih postojećih karakteristika.
Nova definicija robusnosti kao i modifikovani GA koji implementira datu definiciju, testiran je na odabranim primerima iz literature. Performanse datog pristupa poređene su sa postojećim metoda iz literature. Eksperimentalni rezultati pokazuju da je dati pristup generisao robusna rešenja visokog kvaliteta koja su uporediva ili bolja od rezultata drugih metoda. Ovaj rad predstavlja značajan doprinos u primeni modifikovanog hipervolume indikatora za rešavanje robusnih rešenja u višekriterijumskoj optimizaciji. Dalje proširenje i unapređivanje datih rezultata može se izvršiti u nekoliko pravaca:

- testiranje na većem broju test primera iz literature
- primena datog pristupa na druge GA koji ne uključuju hipervolume indikator
- testiranje datog pristupa na nekim složenijim problemima

7.1 Naučni doprinos rada

Najvažniji novi rezultati dobijeni istraživanjem prikazanim u ovom radu su:

- Novi model robusnih rešenja koji na bolji način opisuje njihove stohastičke osobine.

- Nov način poređenja Pareto optimalnih skupova rešenja. Pri čemu se nova definicija prirodno svodi na standardnu definiciju Pareto optimalnih rešenja tako da je moguća direktna primena Hipervolume indikatora.

Kao što se može videti iz eksperimentalnih rezultata, predložena nova definicija robusnosti na datim primerima daje u proseku bolje rezultate od prethodnih metoda opisanih u literaturi. Zbog svega gore navedenog, naučno istraživanje opisano u ovom radu daje doprinos oblastima višekriterijumske optimizacije, robusnih rešenja i genetskih algoritama.

Glava 8

Dodatak

Prevod važnijih pojmova na engleski jezik

direktno preuzimanje	direct scaling
dominira	dominates
dvopoziciono ukrštanje	two-point crossover
efektivna funkcija prilagođenosti	effective fitness function
elitičke strategije	elitist strategy
fino gradirana turnirska selekcija	fine grained tournament selection
funkcija cilja	objective function
funkcija dostignuća	attainment function
funkcija prilagođenosti	fitness function
generacija	generation
generacijski GA	generational GA
genetski algoritmi	genetic algorithms
heuristika	heuristic
hipervolume indikator	hypervolume indicator
jedinke	individuals
jednaka rešenja	indifferent solutions
jednokriterijumska optimizacija	single optimizatio
jednopoziciono ukrštanje	one-poit crossover
kodiranje	encoding
kriterijum zaustavljanja	termination criterion
Lagranžova relaksacija	Lagrangean relaxation
linearno skaliranje	linear scaling
metaheuristika	metaheuristic
metod Monte Karlo	Monte Carlo Method
metoda promenljivih okolina	variable neighborhood search
mravlji sistemi	ant systems
mutacija	mutation
ne dominira	nondominates

neodređenosti	uncertanties
neuporediva rešenja	incomparable solutions
neuralne mreže	neural networks
ograničenja	constraints
optimizacija	optimization
Pareto front	Pareto front
Pareto optimalno rešenje	Pareto optimal solution
Pareto skup	Pareto set
početna populacija	initial population
populacija	population
prilagođenost	fitness value
problem ranca	knapsack problem
problem trgovačkog putnika	traveling salesman problem
prosta rulet selekcija	simple roulette selection
prostor pretrage	decision/search space
prostor vrednosti funkcije cilja	objective space
robusna rešenja	robust solutions
robusnost	robustness
selekcija	selection
selekcija bazirana na rangu	rank based selection
sigma odsecanje	sigma truncation sheme
simulirano kaljenje	simulated annealing
slabo dominira	weakly dominates
stacionarni GA	steady-state GA
šum	noise
tabu pretraživanje	tabu search
turnirska selekcija	tournament selection
ukrštanje	recombination
uniformno ukrštanje	uniform crossover
vector promenljive	decision vector
vektor funkcije cilja	objective vector
veličina populacije	population size
višekriterijumska optimizacija	multi objective optimization
višepoziciono ukrštanje	multi point crossover

Bibliografija

[Ant89] J. Antonisse: *A new interpretation of schema that overturns the binary encoding constraint*, Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, Calif., pp. 86-91, 1989

[Arn03] D.V. Arnold, H.-G. Beyer: *A comparison of evolution strategies with other direct search methods in the presence of noise*, Comput. Optim. applicat., vol.24, pp. 135-159, 2003

[Aug09] A. Auger, J. Bader, D. Brockhoff, E. Zitzler: *Theory of the Hypervolume Indicator: Optimal μ - Distributions and the Choice of the Reference Point*, 2009, ftp://ftp.tik.ee.ethz.ch/pub/people/brockho/abbz2009a.pdf

[Bae97] T. Baeck, U. Hammel, H.-P. Schwefel: *Evolutionary computation: Comments on the history and current state*, IEEE Transactions on Evolutionary Computation, 1(1):317, 1997

[Bad08] J. Bader and E. Zitzler: *HypE: An Algorithm for Fast Hypervolume-Based Many-Objective Optimization*, TIK Report 286, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, 2008 http://www.tik.ee.ethz.ch/sop/publicationListFiles/bz2008a.pdf

[Bad09] J. Bader and E. Zitzler: *A Hypervolume-Based Multiobjective Optimizer for High-Dimensional Objective Spaces*, In Conference on Multiple Objective and Goal Programming (MOPGP 2008), Lecture Notes in Economics and Mathematical Systems. Springer, 2009

[Bau95] E. Baum, D. Boneh, C. Garret: *On genetic algorithms*, Proc. 8th Annu. Conf. Comput. Learn. Theory, pp. 230-239, 1995

[Bea93a] D. Beasley, D.R. Bull, R.R. Martin: *An Overview of Genetic Algorithms, Part 1, Fundamentals*, University Computing, Vol. 15, No. 2, pp. 58-69, 1993

[Bea93b] D. Beasley, D.R. Bull, R.R. Martin: *An Overview of Genetic Algorithms, Part 2, Research Topics*, University Computing, Vol. 15, No. 4, pp. 170-181, 1993

[Bee92a] T. Beeck: *Self-adaptation in Genetic Algorithms*, Proceedings of the First European Conference on Artificial Life, MIT Press, 1992

[Bee93] T. Beeck: *Optimal Mutation Rates in Genetic Search*, Proceedings of the Fifth International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, Calif., pp. 2-8, 1993

[Bey93] H.-G. Beyer: *Toward a theory of evolution strategies: Some asymptotical results from the (1 + λ)-theory*, Evol. Comput., vol.1, no.2, pp.165-188, 1993

[Boo87] L. Booker: *Improving Search in Genetic Algorithms*, Genetic Algorithms and Simulated Annealing, Pitman Publishing, London, pp. 61-73, 1987

[Bra01] J. Branke, C.Schmidt, H.Schmeck et al.: *Efficient fitness estimation in noisy environment, Genetic and Evolutionary Computation*, L. Spector et all., Eds. San Mateo, CA: Morgan Maufmann, pp.243-250, 2001

[Bra91] M.F. Bramlette: *Initialisation, mutation and selection methods in genetic algorithms for function optimization*, Proceedings of the Fourth International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, Calif., pp. 100-107, 1991

[Bra98] J. Branke: *Creating Robust Solutions by Means of Evolutionary Algorithms*, A. E. Eiben and T. Baeck and M. Schoenauer and H.-P. Schwefel, editor, Parallel Problem Solving from Nature - PPSN V, pages 119-128, Berlin, 1998. Springer. Lecture Notes in Computer Science 1498

[Čan96] M. Čangalović: *Opšte heuristike za rešavanje problema kombinatorne optimizacije*, Kombinatorna optimizacija: Matematička teorija i algoritmi, str. 320-350, 1996

[Dav91] L. Davis: *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, New York, 1991

[Deb06] K. Deb, H. Gupta: *Introducing robustness in multi-objective optimization*, Evolutionary Computation, 14(4):463-494, 2006

[Dor96] M. Dorigo, V. Maniezzo, A. Colorni: *Ant System: Optimizing by a Colony of Cooperating Agents*, IEEE Transactions on Systems, Man and Cybernetics -Part B, Vol. 26, No. 1, pp. 29-41, 1996

[Fan90] L. Fang, T. Li: *Design of competition based neural networks for combinatorial optimization*, International Journal on Neural System, Vol. 3, pp. 221-235, 1990

[Fil00] V. Filipović, J. Kratica, D. Tošić, I. Ljubić: *Fine Grained Tournament Selection for the Simple Plant Location Problem*, Proceedings of the 5th Online World Conference on Soft Computing Methods in Industrial Applications - WSC5, pp.152-158, 2000

[Fil01] V. Filipović, J. Kratica, D. Tošić, I. Ljubić: *Experimental Results in Applying of Fine Grained Tournament Selection*, Proceedings of the 10th Congress of Yugoslav Mathematicians, pp. 331-336, 2001

[Fil03] V. Filipović: *Fine-Grained Tournament Selection Operator in Genetic Algorithms*, Computing and Informatics 22(2), pp. 143-161, 2003

[Fil97] V. Filipović: *Određivanje performansi genetskih algoritama u teoriji i praksi*, Prolećna škola o programskim jezicima, Institut za Matematiku PMF, Novi Sad, str. 131-141, 1997

[Fil98] V. Filipović: *Predlog poboljšanja operatora turnirske selekcije kod genetskih algoritama*, Magistarski rad, Univerzitet u Beogradu, Matematički fakultet, 1998

[Fit88] J.M. Fitzpatrick , J.I. Grefenstette: *Genetic algorithms in noisy environments*, Mach. Learn, vol.3, pp.101-120, 1988

[Geo74] A. Geoffrion, R. McBride: *Lagrangean Relaxation for Integer Programming*, Mathematical Programming Study, Vol. 2, pp. 82-114, 1974

[Glo86] F. Glover: *Future paths for integer programming and links to artificial intelligence*, Computers & Operations Research, Vol. 5, pp. 533-549, 1986

[Glo90] F. Glover: *Tabu search: A Tutorial*, Interfaces, Vol 20, pp. 74-94, 1990

[Gol89] D.E. Goldberg: *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Publ. Comp., Reading, Mass., 412 pp, 1989

[Gol91] D.E. Goldberg, K. Deb: *A comparative analysis of selection schemes used in genetic algorithms*, Foundations of Genetic Algorithms, Rawlins G. (ed.), Morgan Kaufmann, San Mateo, pp. 69-93, 1991

[Gre95] J. J. Grefenstette, R. Gopal, B. J. Rosmaita, and D. V. Gucht: *Genetic algorithms for the traveling salesman problem*, J. J. Grefenstette, editor, Proceedings of the First International Conference on Genetic Algorithms, pages 160-168. Lawrence Erlbaum, 1985

[Ham94] U. Hammel, T. Baeck: *Evolution strategies on noisy functions, how to improve convergence properties*, Parallel Problem Solving from Nature. ser. LNCS, Y. Davidor, H.P. Schwefel and R. Maenner, Eds. Berlin, Germany:Springer-Verlag, vol866, pp.159-168, 1994

[Hin94] R. Hinterding: *Mapping, order-independent genes and the knapsack problem*, D. Schaffer,H.-P. Schwefel, and D. B. Fogel, editors, Proceedings of the First IEEE Conference on Evolutionary Computation, pages 13-17. IEEE Press, 1994

[Hol75] J.H. Holland: *Adaptation in Natural and Artificial Systems*, The University of Michigan Press, Ann Arbor, 1975

[Hug01] E.J. Hughes et al.: *Evolutionary multiobjective ranking with uncertainty and noise*, Evolutionary Multi-Criterion Optimization. ser. LNCS, E. Zitzler et al., Eds. Berlin, Germany:Springer-Verlag, vol.1993, pp.329-343, 2001

[Jin03] Y. Jin, B. Sendhoff: *Trade-Off between Performance and Robustness: An Evolutionary Multiobjective Approach*, C. M. Fonseca et al., editors, EMO 2003, volume 2632, pages 237-251. Springer, 2003

[Jin05] Y. Jin, J. Branke: *Evolutionary Optimization In Uncertain Environments-A Survey*, IEEE Transactions on Evolutionary Computation,9(3):303-317, 2005

[Jog89] P. Jog, J.Y. Suh, D. Van Gucht: *The effects of Population size, Heuristic Crossover and Local Improvement on a Genetic Algorithm for the Traveling Salesman Problem*, Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, Calif., pp. 110-115, 1989

[Jon75] K.E. De Jong: *An analysis of the behavior of a class of genetic adaptive systems*, PhD thesis, University of Michigan, 1975

[Kir83] S. Kirkpatrick, C. Gellat, M. Vecchi: *Optimization by simulated annealing*, Science, Vol. 220, pp. 671-680, 1983

[Kra00] J. Kratica: *Paralelizacija genetskih algoritama za rešavanje nekih NP kompletnih problema*, Doktorska disertacija, Matematički fakultet, Beograd, 2000

[Kra97a] J. Kratica: *Napredne tehnike genetskih algoritama i njihova implementacija*, Prolećna škola o programskim jezicima, Institut za Matematiku PMF, Novi Sad, str. 123-130, 1997

[Lou99] D.H. Loughlin, S.R. Ranjithan: *Chance-constrained genetic algorithms*, Proc. Genetic Evol. Comput. Conf., pp.369-376, 1999

[Mar01] S. Markon, D. Arnold, T. Baeck, T. Beielstein, H.-G. Beyer: *Thresholding-A selection operator for noisy ES*, Proc. Congr. Evol. Comput., pp.465-472, 2001

[Mic96] Z. Michalewicz: *Genetic Algorithms + Data Structures = Evolution Programs*, Third Edition, Springer Verlag, Berlin Heidelberg, 1996

[Mil96] B. L. Miller, D.E. Goldberg: *Genetic algorithms, selection schemes, and the varying effects of noise*, Evol. Comput., vol.4, no.2, pp.113-131, 1996

[Mil97] B. L. Miller: *Noise, sampling, and efficient genetic algorithms*, Ph.D. fissertation, Dept. Comput. Sci., Univ. Illinois at Urbana-Champaign, Urbana, IL, available as TR 97001, 1997

[Mit96] M. Mitchell: *An Introduction to Genetic Algorithms*, MIT Press, 1996

[Mla95] N. Mladenović: *A variable neighborhood algorithm - a new metaheuristics for combinatorial optimization*, Abstracts of papers presented at Optimization days, Montreal, 1995

[Mla97] N. Mladenović, P. Hansen: *Variable neighborhood search*, Computers Operations Research, Vol. 24, pp. 1097-1100, 1997

[Mos89] P. Moscato: *On Genetic Crossover Operators for Relative Order Preservation*, Caltech Concurent Computation Program, C3P Report 778, 1989

[Mue97] H. Muehlenbein: *Genetic algorithms*, Local Search in Combinatorial Optimization, eds. Aarts E.H.L., Lenstra J.K., John Wiley & Sons Ltd., pp. 137-172, 1997

[Par96] I.C. Parmee: *The maintenance of search diversity for effective design space decomposition using cluster-oriented genetic algorithms(cogas) and multi-agent strategies(gaant)*, ACEDC, 1996

[Rai99] G. R. Raidl: *Weight-codings in a genetic algorithm for the multi-constraint knapsack problem*, P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, A. Zalzala, and W. Porto, editors, Proceedings of the 1999 IEEE Congress on Evolutionary Computation, pages 596-603. IEEE Press, 1999

[Ray02] T. Ray: *Constrained robust optimal design using a multiobjective evolutionary algorithm*, Proc. Congr. Evol. Comput., pp.419-424, 2002

[San00] Y. Sano, H. Kita et al.: *Optimization of noisy fitness function by means of genetic algorithms using history of search*, Parallel Problem Solving from Nature. ser. LNCS, M. Schoenauer et al., Eds. Berlun, Germany: Springer-Verlag, vol.1917, pp.571-580, 2000

[Smi93] R. Smith, S. Forrest, A.S. Perelson: *Searching for diverse, cooperative populations with genetic algorithms*, Evolutionary Computation, Vol. 1, No. 2, pp. 127-149, 1993

[Spe91] W. Spears, K. De Jong: *On the virtues of parametrized uniform crossover*, Proceedings of the Fourth International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, Calif., pp. 230-236, 1991

[Sri94] M. Srinivas, L.M. Patnaik: *Genetic Algorithms: A Survey*, IEEE Computer, pp. 17-26, 1994

[Sys89] G. Syswerda: *Uniform Crossover in Genetic Algorithms*, Proceedings of the Third International Conference on Genetic Algorithms - ICGA '89, Morgan Kaufmann, San Mateo, Calif., pp. 2-9, 1989

[Sys91] G. Syswerda: *A study of reproduction in generational and steady-state genetic algorithms*, Foundations of Genetic Algorithms - FOGA, Rawlins G.J. (ed.), Morgan Kaufmann Publishers, pp. 94-101, 1991

[Tei01] J.Teich et al.: *Pareto-front exploration with uncertain objectives* , Evolutionary Multi–Criterion Optimization. ser. LNCS, E. Zitzler et al., Eds. Berlin, Germany:Springer-Verlag, vol.1993, pp.314-328, 2001

[Tošo4] D. Tošić, N. Mladenović, J. Kratica, V. Filpović: *Genetski algoritmi*, Matematički institut SANU, Beograd, 2004

[Tošo7] D. Tošić: *Pregled razvoja i opis osnovnih karakteristika evolucionih (genetskih) algoritama*, Prolećna škola o programskim

jezicima, Institut za Matematiku PMF, Novi Sad, str. 115-122, 1997

[Tsu96] S. Tsutsui, A. Ghosh, Y. Fujimoto: *A robust solution searching scheme in genetic search*, Parallel Problem Solving from Nature, Berlin, Germany: Springer-Verlag, pp.543-552, 1996

[Tsu97] S. Tsutsui, A. Ghosh: *Genetic Algorithms with a Robust Solution Searching Scheme*, IEEE Trans. on Evolutionary Computation, 1(3):201-208, 1997

[Whi89] D. Whitley: *The GENITOR Algorithm and Selection Pressure: Why Rank-Based Allocation of Reproductive Trials is Best*, Proceedings of the Third International Conference on Genetic Algorithms - ICGA '89, Morgan Kaufmann, San Mateo, Calif., pp. 116-123, 1989

[Yur94] D. Yuret: *From Genetic Algorithms to Efficient Optimization*, MSc Thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, 1994

[Zit03] E. Zitzler, L. Thiele, M. Laumanns, C. M. Foneseca, and V. Grunert da Fonseca: *Performance assessment of multiobjective optimizers: An analysis and review*, IEEE Transactions on Evolutionary Computation, 7(2):117-132, 2003

[Zit04] E. Zitzler, M. Laumanns, and S. Bleuler: *A Tutorial on Evolutionary Multiobjective Optimization*, Workshop on Multiple Objective Metaheuristics (MOMH 2002), Springer-Verlag, Berlin, Germany, 2004

[Zit07] E. Zitzler, D. Brockhoff, and L. Thiele: *The Hypervolume Indicator Revisited: On the Design of Pareto-compliant Indicators Via Weighted Integration*, In S. Obayashi et al., editors, Conference on Evolutionary Multi- Criterion Optimization (EMO 2007), volume 4403 of LNCS, pages 862-876, Berlin, Springe, 2007

[Zit08] E. Zitzler, L. Thiele, and J. Bader: *On Set-Based Multiobjective Optimization (Revised Version)*, TIK Report 300, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, 2008,

[Zit98] E.Zitzler and L.Thiele: *Multiobjective Optimization Using Evolutionary Algorithms- A Comparative case study*, In PPSN-V, pages 292-301, Amsterdam, 1998