УНИВЕРЗИТЕТ У КРАГУЈЕВЦУ
ФАКУЛТЕТ ИНЖЕЊЕРСКИХ НАУКА

Ненад Костић

РАЗВОЈ И ПРИМЕНА МЕТОДА ХЕУРИСТИЧКЕ ОПТИМИЗАЦИЈЕ МАШИНСКИХ КОНСТРУКЦИЈА

Докторска дисертација

Крагујевац, 2017.

Идентификациона страна:

I. Аутор
Име и презиме: *Ненад Костић*
Датум и место рођења: *26.09.1986. Краљево*
Садашње запослење: *истраживач – сарадник на Факултету инжењерских наука Универзитета у Крагујевцу*
II. Докторска дисертација
Наслов: *Развој и примена метода хеуристичке оптимизације машинских конструкција*
Број страна: *152*
Број слика: *62*
Број библиографских података: *196*
Установа и место где је рад израђен: *Универзитет у Крагујевцу, Факултет инжењерских наука, Крагујевац*
Научна област (УДК): *621.8 (Опште машинско инжењерство, Машинство), 519.8 (Операциона истраживања, Математичко програмирање)*
Ментор: *др Ненад Марјановић, редовни професор*
III. Оцена и одбрана
Датум пријаве теме: *22.01.2015., одлука бр. 01-1/107-5*
Број одлуке и датум прихватања докторске дисертације:
Комисија за оцену подобности теме и кандидата: *др Ненад Марјановић, редовни професор* *др Милорад Бојић, редовни професор* *др Весна Ранковић, редовни професор* *др Мирко Благојевић, ванредни професор* *др Зоран Милојевић, ванредни професор*
Комисија за оцену и одбрану докторске дисертације: *др Мирко Благојевић, ванредни професор, председник комисије* *др Весна Ранковић, редовни професор* *др Милан Рацков, доцент* *др Блажа Стојановић, доцент* *др Весна Марјановић, ванредни професор*
Датум одбране дисертације: ___.___.2017. год.

РАЗВОЈ И ПРИМЕНА МЕТОДА ХЕУРИСТИЧКЕ ОПТИМИЗАЦИЈЕ МАШИНСКИХ КОНСТРУКЦИЈА

Резиме рада:

У овом раду извршено је дефинисање појма оптимизације, њен историјски развој и математичка формулација, са акцентом на хеуристичку оптимизацију. Представљене су групе метода математичке оптимизације, а хеуристичке методе су детаљније описане. Представљене су предности метода *GA, PSO, TLBO*, које су одабране као репрезентативне хеуристичке методе. Развијене су модификације, хибридна метода и нова хеуристичка оптимизациона метода. На основу методе *GA*, његовим модификовањем, добијена је метода *iGA*. Модификација *rTLBO* развијена је на основу оригиналног *TLBO* алгоритма. Развијени су нови приступи хибридизације хеуристичких метода, а „редна“ хибридизација примењена је за развој *hGPT* алгоритма. Овај хибридни алгоритам користи *iGA, PSO* и *TLBO* алгоритме. На основу хеуристичке појаве, развијена је оригинална оптимизациона метода названа *DINDI*. Метода је изведена на основу експерименталних испитивања и праћења детета у току игре (дечак Динди). Развијен је оригинални оптимизациони софтвер, а у софтвер су имплементиране развијене хеуристичке методе. Софтвер је самосталан и омогућава општост у погледу оптимизације. Могуће је оптимизовати било који проблем математички формулисан овим софтвером. Он омогућује кориснику све потребне активности у фазама предпроцесирања, процесирања и постпроцесирања. Развијене методе тестиране су према препорукама из литературе и на начине којима је могуће потврдити њихове квалитете. За тестирања су коришћене тест функције без ограничења, са ограничењима и инжењерске тест функције. Према препорукама из литературе, извршено је поређење развијених метода, са атрактивним методама из литературе. Развијена оптимизациона метода (*DINDI*) примењена је за решавање практичних проблема машинског конструисања. За проблем оптимизације редуктора, развијен је одговарајући облик универзалног математичког модела и спроведена је оптимизација за конкретне примере редуктора. За анализиране примере постижу се значајно боље карактеристике редуктора применом оптимизационог процеса. За проблем контакта елемената циклоредуктора развијен је нови приступ и оригинални математички модел. Овај приступ омогућава одређивање растојања између елемената циклоредуктора, што је значајно, не само са аспекта оптимизације, него и са аспекта развоја алтернативних преносника снаге. Конкретним примерима је потврђен квалитет развијених математичких модела. На основу обимних испитивања и анализа, изведени су закључци и представљене смернице и потенцијални правци даљих истраживања.

Кључне речи:

оптимизација, методе оптимизације, хеуристичке методе, *GA, PSO, TLBO*, модификација *iGA,* модификација *rTLBO,* хибридна метода *hGPT,* нова метода *DINDI*, оптимизациони софтвер, проблеми без ограничења, проблеми са ограничењима, инжењерски оптимизациони проблеми, машинске конструкције, редуктор, запремина редуктора, циклоредуктор, контактни проблем циклоредуктора

DEVELOPMENT AND APPLICATION OF HEURISTIC OPTIMIZATION METHODS IN MECHANICAL CONSTRUCTIONS

Abstract:

This dissertation defines the term optimization, gives its historical development and mathematical formulation, with an accent on heuristic optimization. The mathematical optimization method group is presented, and heuristic methods are described in detail. As representative heuristic methods the advantages of *GA*, *PSO*, *TLBO* methods are shown. Modifications, a hybrid method and a new heuristic optimization method were developed. Based on *GA*, through its modification, the *iGA* method was created. The *rTLBO* modification was developed based on the original *TLBO* method. New approaches to hybridization of heuristic methods were developed, and "inline" hybridization was used to develop the *hGPT* algorithm. This hybrid uses *iGA*, *PSO*, and *TLBO* algorithms. Based on heuristic occurrences, an original optimization method was developed named *DINDI*. The method was developed based on experimental research and monitoring a child playing (a boy named Dindi). An original optimization software was developed which implements the developed heuristic methods. The software is a standalone program and allows for general use in terms of optimization. It is possible to optimize any mathematically formulated problem using this software. The software allows the user to access and use all necessary activities in the phases of preprocessing, processing and post processing. The developed methods of testing are compliant to guidelines from literature and to the methods of confirming their qualities. Unconstrained, constrained, and engineering test functions were used for testing. According to suggestions from literature, a comparison of developed methods with current methods from literature was conducted. The developed optimization method (*DINDI*) was used for solving practical problems in mechanical constructions. For the problem of gear train optimization a form of universal mathematical model was developed and the optimization for specific examples of gear trains was conducted. For the analyzed examples significant improvements of gear train characteristics are achieved through the optimization process. For the clearances problem of cyclodrive a new approach and original mathematical model were created. This approach allows for determining the distance between elements of a cycloid drive, which is notable, not only from an optimization aspect, but for development of alternative transmissions as well. Using specific examples the quality of the mathematical models was confirmed. Based on a vast amount of testing analyses, conclusions were made and presented with the addition of potential further directions of research.

Key words:

optimization, optimization methods, heuristic methods, *GA*, *PSO*, *TLBO*, *iGA* modification, *rTLBO* modification, *hGPT* hybrid method, new method *DINDI*, optimization software, unconstrained optimization, constrained optimization, engineering optimization, mechanical constructions, gear train, gear train volume, cyclodrive, cyclodrive clearances problem

Списак слика:

Списак табела:

Садржај:

1. Увод

1.1 Опште о оптимизацији

Непрестана тежња савременог друштва да задовољи потребе тржишта, учини живот лакшим и бољим и да убрза све процесе које човек треба да обавља представља велики изазов, а посебно за инжењерске дисциплине. Инжењери морају обезбедити да се са што мање уложених напора, времена и са што мање негативних ефеката постигну што боље перформансе производа и што ефикаснији производи. Под производима се подразумевају и машинске конструкције, што је и акценат овог истраживања. Да би се обезбедиле овакве перформансе машинских конструкција неопходно је велико искуство, дуготрајни и скупи процеси развоја и испитивања и стална подршка производње. Ова чињеница креира потребу за стварањем нових метода и техника које би унапредиле, убрзале и појефтиниле поменуте процесе. Методе и технике које се користе у ове сврхе називају се оптимизационим методама и техникама, а процес унапређења, побољшања и усавршавања назива се оптимизацијом.
Оптимизација представља процес и поступак проналажења практично применљивог и практично прихватљивог, оптималног, решења из домена дефинисаног на основу проблема. Постигнуто решење мора да задовољи све утицајне критеријуме како би било прихватљиво и третирано као ваљана репрезентација оптимизационог проблема. Оптимизација представља спој математике, компјутерских наука и операционих истраживања и бави се изучавањем метода и техника за проналажење оптималних решења. Проблеми које разматра ова област називају се оптимизационим проблемима. У практичној примени оптимизација најчешће представља проналажење екстремних вредности, минимума или максимума, функције једне или више међусобно зависних или независних променљивих. Проблеми разматрани у овој области морају бити тачно одређени и математички у потпуности дефинисани. Променљиве оптимизације могу бити реалне вредности, целобројне, дискретне или бинарне. Приступ оптимизацији је углавном исти и заснива се на избору методе или поступка решавања проблема, као и адекватној формулацији проблема.
Оптимизација директно доводи до минимизације негативних, уз истовремену максимизацију позитивних ефеката, што је основни разлог примене оптимизације на широком дијапазону проблема. Оптимизација се примењује у свим гранама инжењерства (машинству, грађевинарству, електротехници, итд.), трговини, транспорту, финансијама и свим процесима доношења одлука. Уштеде у погледу побољшања, зависно од конкретног проблема, могу бити финансијске (директно или индиректно), временске, уштеде у материјалу и енергији, побољшање радних карактеристика, повећање поузданости и степена сигурности посматраног система.

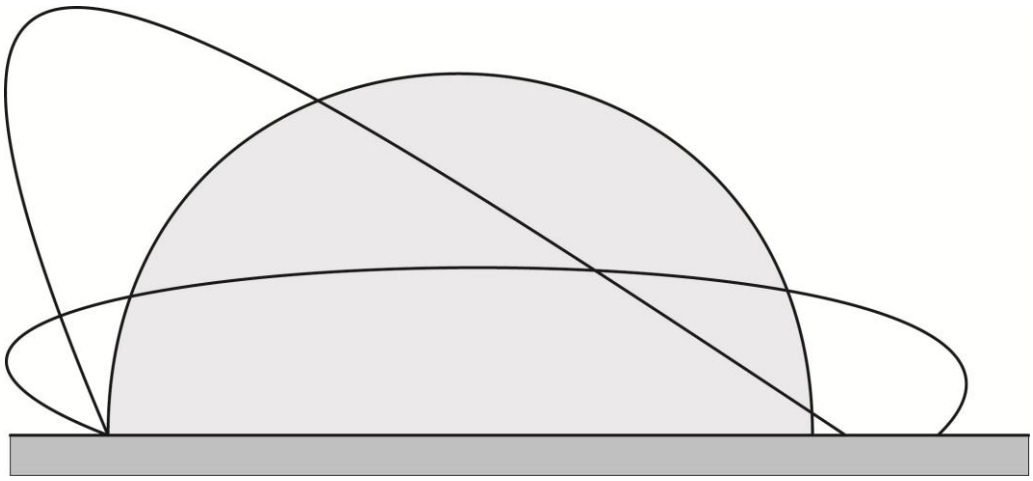
1.2 Историјски развој оптимизације

Оптимизација практично датира од настанка човечанства и развија се са тежњом човека да оствари баланс између постигнутог и уложеног. Човечанство непрестано тежи да повећа квалитет живота уз што мање рада и уложених напора. Примена оптимизације подразумева решавање проблема са чиме се сваки човек свакодневно среће, бар на

елементарном нивоу доношења оптималних одлука. Човек стално врши неки избор, доноси одлуке и покушава да постигне што је више могуће, а да уложи што мање напора. Такође, анализира шта све утиче на проблеме који му се намећу и покушава да нађе компромис између наметнутих околности. Овај ниво оптимизације је елементаран и не захтева математичку формулацију, већ је логика довољна за решавање ових проблема. Када је реч о комплексним проблемима, самом рационалном размишљању, неопходно је додати математичке апарате и различите методе и технике, а све у циљу доношења ваљаних, оптималних одлука. Како је потребно све проблеме оптимизације математички формулисати често се ова област дефинише као математичка оптимизација. У почетним фазама развој ове области био је оријентисан на развој искључиво најбољег решења (најбоље од најбољих), док се постављање ограничења и оријентација на допустиве и реалне услове намеће тек касније. Модерно доба оптимизације почиње са развојем рачунара. Данас постоји велики број докумената, књига, научно-истраживачких радова и других облика писаних форми о поступцима, методама и техникама оптимизације. За област оптимизације није погрешно тврдити да је неисцрпан истраживачки извор. Увек је могуће вршити оптимизацију у односу на постигнуто оптимално решење, док добијено решење само конвергира стварном оптимуму, а питање је да ли га икада постиже. Даље је важна практична применљивост оптималних резултата, што дефинише успешност оптимизационог процеса.

Геометријски проблеми су међу првим комплексним и за човека свесним проблемима на којима је примењена оптимизација. Историјски посматрано постизање оптималних карактеристика вршено је стохастички, подсвесно, без неког јасног познавања процеса оптимизације. Овакав приступ има хеуристички карактер, чему се и данас све више тежи.

Један од карактеристичних историјских проблема које је могуће пронаћи у литератури је проблем краљице Дидо (*Dido*), коју још називају и Елиза (*Elissa*) [1]. Када је краљица Дидо хтела да купи земљу на просторима данашњег Туниса, дозволили су јој да купи само онолико земље колико може да обухвати кожом једног бика. Дидо је наредила да исеку кожу на танке траке, да искористе обалу и поставе кожну траку у полукруг, што представља оптимално решење овог проблема. Приказ решења представљен је на слици 1.1 [1].



Слика 1.1. Графички приказ решења Дидониног проблема [1]

Овај проблем своди се на оптимизацију површи једнаких обима и како се процењује датира из периода од 900 година пре нове ере. На слици 1.1 представљене су криве једнаких дужина.

Проблем инжењерске оптимизације први је дефинисао Галилео Галилеј (*Galileo Galilei*) у записима (књизи) „Дијалози о две науке“ [1]. Галилео је својим анализама око 1638. године поставио темеље данашње структурне оптимизације, са аспекта облика, положаја и димензија носећих конструкција.

Терминолошки оптимум је први дефинисао Лајбниц (*G. Laibnitz*) у 18. веку [2]. Дефиниција оптимума коју је поставио подразумева проналажење максимума или минимума неке математичке функције. Овај тренутак може се сматрати почетком развоја савремене оптимизације. Развоју поступака за проналажење екстремних вредности функција допринели су Њутн (*Newton*), Ојлер (*Euler*), Бернули (*Bernoulli*), Лагранж (*Lagrange*) и други, чија су истраживања била заснована на споју инжењерских знања и математичких апарата.

Тек у 20. веку експлицитно се дефинишу методе оптимизације и врши се њихова класификација према врстама и типу проблема. За проблеме линеарног програмирања, симплекс (*Simplex*) методу 1947. године развио је Данциг (*Dantzig*), док је Белман (*Bellman*) 1957. године био оријентисан на динамичко програмирање [2, 3], са акцентом на методе за оптимизационе проблеме са ограничењима. Основе нелинеарног програмирања поставили су Кан (*Kuhn*) и Тачер (*Tucker*) 1951. године [4]. У периоду 60-их година прошлог века долази до велике експанзије нелинеарне оптимизације. То је период анализе проблема са целобројним вредностима, развој стохастичког програмирања од стране Данцига, Карнеса (*Charnes*) и Купера (*Cooper*) [3]. У периоду 60-их година почиње и анализа проблема вишекритеријумске оптимизације.

Паралелно са овим техникама развијају се и методе хеуристичке оптимизације, првенствено развојем еволуционог програмирања. Најзначајнијим историјским тренутком сматра се дефинисање методе генетског алгоритма 1975. године од стране Џона Холанда (*John Holland*) [5]. Сматра се да је Холанд генетски алгоритам развио још око 1962. године на Универзитету Мичиген са својим сарадницима. Значајне помаке у развоју генетског алгоритма даје Голдберг (*Goldberg*) [6] 1989. године. Године 1995. Кенеди (*Kennedy*) и Еберхарт (*Eberhart*) развијају PSO алгоритам [7], после кога се акцентат на хеуристичке методе враћа тек од 2008. године развојем ABC алгоритма [8]. Од овог историјског тренутка пажња истраживача према хеуристичким методама је све већа, а данас најатрактивнији допринос у овој области подразумева рад на развоју нових хеуристичких метода оптимизације.

1.3 Математичка формулација оптимизације

Први корак процеса оптимизације представља неки вид формулације проблема који се оптимизује. Формулација представља математичку дефиницију која репрезентује разматрани проблем. Ово се постиже креирањем математичког модела. Формирање математичког модела подразумева дефинисање променљивих проблема, формулисање функције циља и реалних ограничења проблема оптимизације. Сложеност формирања математичког модела зависи у највећој мери од сложености система или објекта оптимизације. Математичко моделирање има за циљ да се избегне испитивање или оптимизација на физичком моделу, јер овакав приступ најчешће није уопште могућ, а и ако је могућ, трошкови су неупоредиво већи, а сам поступак технолошки захтевнији. Зато оптимизација подразумева испитивања на математичком моделу. За решавање

комплексних, реалних оптимизационих проблема неопходно је користити компјутерску подршку, а углавном је ова подршка присутна и код једноставнијих оптимизационих проблема. Оптимално решење или скуп оптималних решења представља најприхватљивије решење које припада дефинисаном домену уз поштовање постављених ограничења. Поред променљивих оптимизације, постављањем адекватне функције циља и ограничења проблема и њиховим обједињавањем у функционалну целину добија се математички модел, а математички модел чини постављање и дефинисање оптимизационог задатка.

1.3.1 Променљиве оптимизације

Веома важну улогу при решавању оптимизационих проблема имају променљиве оптимизације и од њиховог броја и типа зависи комплетан оптімизациони процес. Променљиве оптимизације представљају конкретне вредности или квантитативне величине које су ограничене и дефинисане реалним проблемом и садржане су у могућем домену. Променљиве у процесу оптимизације су све вредности које могу варирати, мењати се, а утичу на формулацију конкретног проблема. Сложеност оптимизације, између осталог, зависи и од броја променљивих. Број променљивих зависи директно од реалног оптимизационог проблема или конструкције и овај број је углавном једнозначно условљен реалним условима. Понекад број променљивих зависи од искуства инжењера, као и познавања и комплексности проблема. Практично је могуће учинити одређене апроксимације и поједностављења реалних проблема како би они били решиви. Зато је важно проценити ниво на коме је потребно разлагати проблем, како би број променљивих био потребан и довољан за постизање оптимума. Код конкретних конструкција или проблема, свака одговарајућа комбинација квантитативних вредности променљивих представља решење проблема или конструкцију. У сваком случају, неопходно је имати у виду да повећање броја променљивих директно води у генерисање сложенијег задатка, а самим тим и отежава његово решавање.
Скуп променљивих оптимизације могуће је дефинисати као вектор који је представљен у једначини (1.1).

$$X = \{x_1, x_2, x_3, \dots, x_n\} \tag{1.1}$$

У приказаној једначини (1.1) n представља укупан број променљивих оптимизације. У процесу оптимизације ове квантитативне вредности мењају се итеративно у допустивом домену. Прихватљива (допустива) решења су само она решења која припадају скупу дефинисаних ограничења, а из тог скупа оптимално решење је најбоље од њих (тражена екстремна вредности функције циља).
Највећи проблем практичне оптимизације представљају променљиве које нису континуалне. То значи да вредности променљивих могу бити изабране из неког тачно одређеног скупа решења. Овакве променљиве су дискретне променљиве, а односе се на практичан захтев оптимизације. Користе се у ситуацијама када треба да се представи нпр. број зубаца зупчаника, број степени преноса, број завртањских веза, величина модула зупчаника, дебљина лимова, пречници профила и цеви и томе слично. На први поглед ови проблеми делују једноставно јер су решења вредности променљивих ограничена, али проблеми са дискретним, целобројним или уопште мешовитим променљивима драстично компликују процес оптимизације. Простор дискретних променљивих је прекидан и неконвексан, а овакви оптимизациони проблеми решавају се посебним оптимизационим техникама.

1.3.2 Функција циља

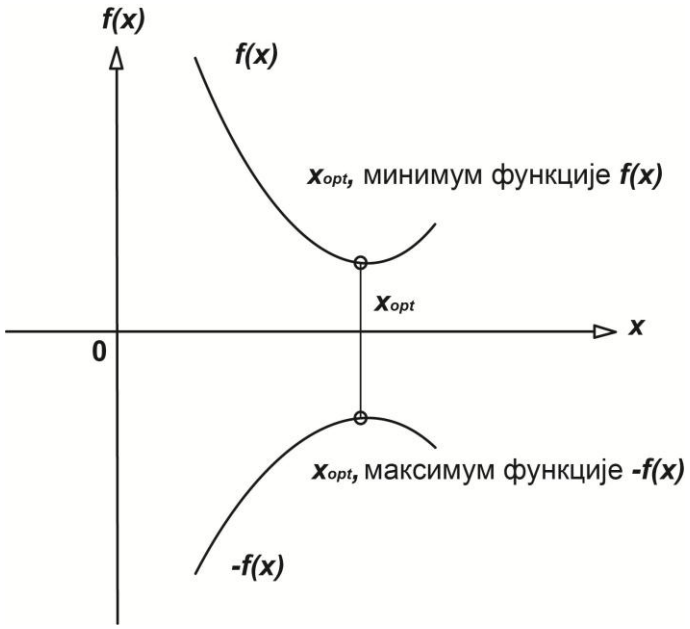
Као би оптимизациони процес био реализован неопходно је поставити и дефинисати циљеве и критеријуме оптимизације. Начин постављања и дефинисања циљева постиже се математичким дефинисањем функције циља. Сваки конкретан проблем захтева појединачни (другачији) приступ при дефинисању ове функције. Првенствено, да би уопште било могуће формулисати функцију циља, неопходно је одредити међусобну функционалну зависност свих променљивих које учествују у оптимизацији, као и свих релевантних зависности које утичу на разматрани проблем. Контрадикције између захтева и ограничења међу утицајним променљивам представља највећу потешкоћу у овом процесу. У практичној оптимизацији циљеви су углавном време, квалитет, трошкови, ефикасност, итд. Коначан исход оптимизационог процеса представља реализацију глобалног циља, а сам проблем је састављен из више често противуречних циљева.

Математички посматрано функција циља представља неку аналитичку функцију $f(X)$, где ова функција дефинише конкретан проблем, а сам оптимизациони проблем представља проналажење глобалне екстремне вредности за ову функцију. Ово заправо значи да вредност вектора променљивих оптимизације представља вредност функције циља, где при испуњењу критеријума заустављања ова вредност треба да буде оптимална. Пошто се овај проблем своди на проналажење екстремне вредности (минимума или максимума), конкретан случај дефинише шта је потребно пронаћи. У оба ова случаја приступ је исти и функција циља је иста, само је неопходно следити зависност из израза (1.2).

$$\min f(X) = \max h(X) = \max\{-f(X)\} \tag{1.2}$$

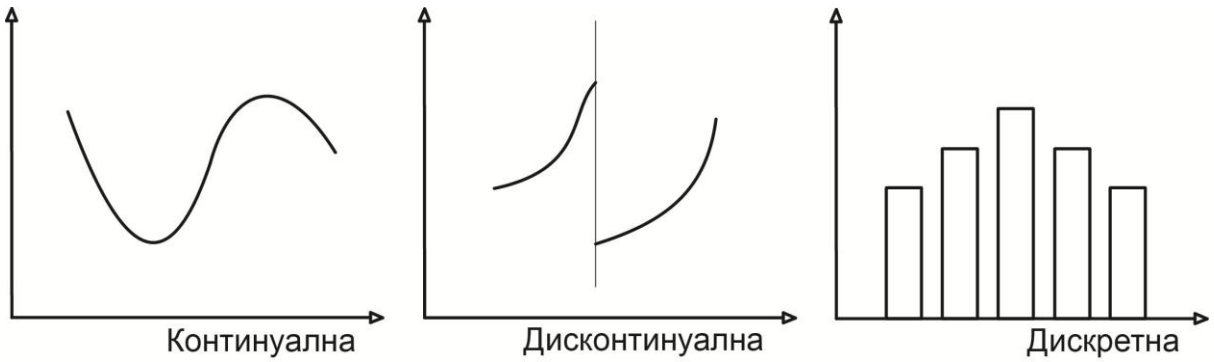
Ова појава назива се еквивалентност код оптимизације. Ако функција $f(X)$, са скупом променљивих од којих зависи, има минималну вредност функције, максимална вредност исте те функције има вредност $-f(X)$. Ова појава представљена је на слици 1.2. То практично значи да се проблеми минимизације и максимизације неке функције могу решавати на исти начин и коришћењем истих метода. Минимална и максимална вредност се поклапају и симетричне су у односу на осу *x*, где се *f(x)* поклапа са *–f(x)*, што је и представљено на слици 1.2.

Еквивалентност код оптимизације може бити заступљена за обе осе, заправо у свим правцима. Поред еквивалентности веома корисно за процес оптимизације је увођење константи у облику $cf(x)$, или $c + f(x)$ [3]. Овим константама добија се облик функције погоднији за оптимизацију у неким случајевима. Множењем константом врши се и померање функције циља и промена облика, док се додавањем константе врши само померање функције по оси $f(x)$.



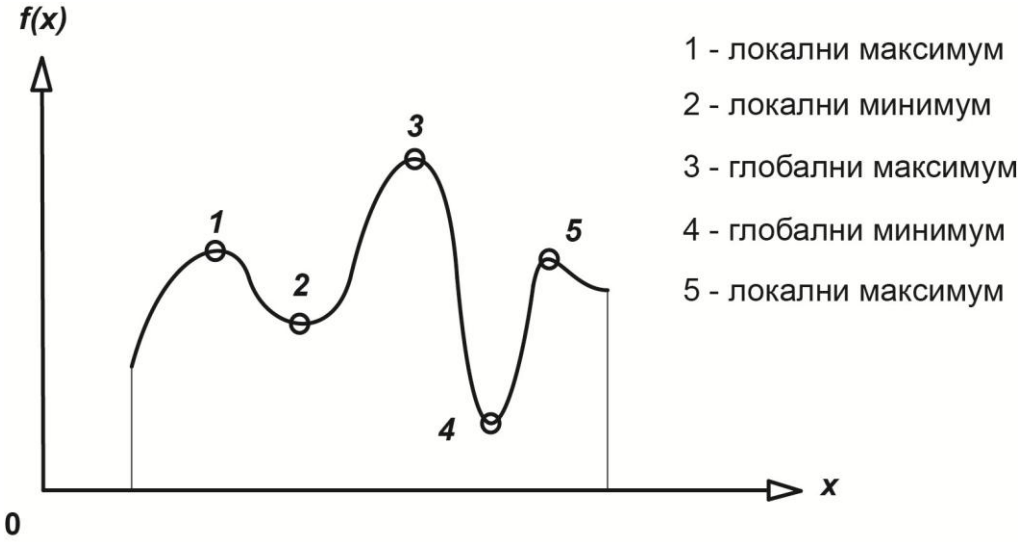
Слика 1.2. Еквивалентност код оптимизације

Дефинисање функције циља у процесу оптимизације представља један од највећих инжењерских изазова, а посебно код реалних проблема и конструкција. Овај процес је веома захтеван јер представља релевантну репрезентацију проблема.
У процесу оптимизације важно је познавати функције са којима је могуће радити. Функција циља може бити континуална, дисконтинуална и дискретна. Облици функције циља представљене су на слици 1.3.



Слика 1.3. Облици функције циља

Када функција циља има велики број локалних екстремних вредности (локалних минимума и максимума), где ове вредности не представљају оптималну вредност, ова функција назива се мултимодалном. Оваквим случајевима неопходно је посветити посебну пажњу јер је чест случај постизања привидног (локалног) минимума, а то условљава неуспех оптимизационог процеса. Пример мултимодалне функције представљен је на слици 1.4.



Слика 1.4. Пример функције са већим бројем локалних екстремних вредности – мултимодалне функције

На слици 1.4 представљена је функција у неком интервалу која има више екстремних вредности. За представљену функцију постоје само по једна глобална екстремна вредност (минимум и максимум), док су све остале екстремне вредности локалног карактера. У оптимизационом процесу веома често се срећу мултимодалне функције, а најтеже у овом случају је превазилажење локалних екстремних вредности.
Најчешће у пракси није могуће ни претпоставити какав је облик функције, а такви проблеми представљају велики оптимизациони изазов. Највећу перспективу у овом случају имају хеуристичке методе оптимизације, јер је њихов карактер да функционишу са малим бројем познатих чињеница.
За један оптимизациони проблем може постојати више циљева, а такав проблем подразумева вишекритеријумску оптимизацију. Постоје технике вишекритеријумске оптимизације, а углавном, када је могуће реализовати једнокритеријумску оптимизацију не постоји велики проблем проширити је за више критеријума.

1.3.3 Скуп ограничења

Реални и стварни услови у којима неки систем или конструкција функционишу дефинишу ограничења оптимизације. Рад система или конструкције могућ је само у оквиру ових услова. Тако, математичка дефиниција проблема увек са собом носи одређена ограничења. Без узимања ових ограничења у обзир модел не би био репрезентативан и реалан.
Математички посматрано ограничења могу бити у виду једначина и / или неједначина или система једначина и / или неједначина [2, 9]. За сваки оптимизациони проблем понаособ овај скуп ограничења могуће је дефинисати системом једначина и / или неједначина G у којима фигуришу променљиве оптимизације X са својим вредностима, као што је представљено у изразу (1.3).

$$g_i(X) = 0; \quad i = 1,2,\dots,m_1. \tag{1.3}$$

У претходном изразу представљена су ограничења у виду једначина, где *m* представља укупан број једначина у систему ограничења. За случај када су ограничења у облику неједначина, израз има следећи облик (1.4).

$$g_i(X) \geq 0; \quad i = m_1 + 1, m_1 + 2, \dots, m. \tag{1.4}$$

По природи решења, променљиве оптимизације *X* у свакој итерацији могу бити у опсегу ограничења или ван овог опсега. Ако се вредности променљивих налазе ван опсега, онда то није задовољавајуће решење. За скуп вредности променљивих, које задовољавају сва ограничења, каже се да су у области претраге. То се такође дефинише као допустиви простор или допустива област модела *D*. Математички посматрано, скуп ограничења представља скуп хипер површина и / или хипер равни у *n* – димензионалном простору.
Процес оптимизације код проблема са ограничењима представља проналажење X^{opt}, при чему функција *f(X)* има екстремну вредност, само из допустиве области *D*. Ова вредност функције назива се и оптималним планом [2].
У процесу стварног решавања оптимизационих проблема, креирање ограничења и управљање ограничењима представља захтеван задатак. У овом процесу неопходно је водити рачуна да променљиве буду у могућим допустивим границама, како би решења уопште постојала. Неопходно је водити рачуна да не дође до противуречности код ограничења, јер постоји могућност да ниједно решење не може да задовољи поједина ограничења. Овај процес решава се искуствено, што доприноси појави грешке.
Могуће је да се догоди да допустива област није ограничена и ако је поставка проблема таква да ни функција циља није ограничена у допустивој области, оптимум није могуће наћи, јер је ово случај бесконачне конвергенције. Ако се оптимизациони процес заустави после дефинисаног броја итерација или после одређеног времена, постиже се неко решење, али оно не значи оптимум. Ова ситуација се може догодити и у случају када је само функција циља неограничена у ограниченој области претраге. То значи да је могуће пронаћи стварни оптимум ако и само ако је функција циља ограничена у допустивој области.

1.3.4 Математички модел

Основа за решавање оптимизационих проблема представља исправно дефинисан и постављен задатак. Ово је предуслов да би уопште било могуће тражити решења проблема. Ово дефинисање и формулација су у пракси представљени математичким апаратима, па се и сам модел који репрезентује проблем назива математичким моделом. Математички модел *M* представља скуп дефинисане функције циља *f(X)*, постојећих ограничења *G(X)* и свих осталих чинилаца који репрезентују предмет и околности оптимизације.
Креирање ваљаног математичког модела у пракси подразумева вишеитерациони поступак. Ретко се дешава да се првом поставком математичког модела уоче све околности и чињенице. Дакле, ово је процес који се састоји из већег броја анализа и корекција иницијалног математичког модела. Веома је важно развити квалитетан математички модел јер се на основу њега врши оптимизација и одређује оптимум. Ако дође до грешке у овом процесу, оптимизацијом ће се одредити оптимална вредност, али за погрешан проблем. Ова вредност може навести инжењера на погрешне закључке. Први корак је креирање иницијалног математичког модела на коме се врше тестирања кроз примену у даљем оптимизационом процесу. Кроз оптимизацију могуће је уочити проблеме и грешке не математичком моделу. Ова испитивања воде у многобројне

измене, а овај процес највише зависи од познавања проблема који се разматра и прикупљених информација о проблему. После већег броја преправки модела могуће је закључити да он, веома често, у потпуности промени иницијални облик и приступ проблему, уз непрестану измену и самих променљивих оптимизације.
У оптимизационом процесу потребно је имати у виду време и напоре који се улажу за дефинисање математичког модела. Оптимизациони процес подразумева да добит и позитивни ефекти буду изнад губитака и негативних ефеката. Зато је могуће одредити и апроксимативни ниво, којим је могуће направити апроксимацију реалног проблема оптимизације на задовољавајућем нивоу, са потребним и довољним бројем променљивих. Овакав приступ не даје апсолутну тачност, али нуди алтернативно оптимално решење на задовољавајућем нивоу.

1.4 Методе математичке оптимизације

Методом математичке оптимизације може се сматрати сваки поступак који конвергира ка оправданом квантитативном решењу у коначном броју итерација. Поред тога, неопходно је да буду укључена реална ограничења наметнута проблемом и да је овај поступак могуће применити на више од једног (групи) оптимизационог проблема. Примена методе на групи проблема подразумева универзалност методе, као једне од најзначајнијих мерила квалитета оптимизационих метода.
Методе математичке оптимизације могуће је представити кроз њихову класификацију. Многи аутори су се бавили груписањем и класификацијом метода, а све у циљу да се дефинише способност неке методе да реши одређене оптимизационе проблеме, тачније интересантне групе проблема којима је метода и намењена. Данас, када је хеуристички приступ (примена хеуристичких метода оптимизације) у пуном јеку, класификација метода користи само за њихову адаптацију врстама оптимизационих проблема, јер ова група метода има велику, можда чак и апсолутну универзалност. Овим радом представљени су најзначајнији сегменти класификације из литературе, неопходни за даљи развој овог истраживања.
Када се говори о критеријумима оптимизације проблеми се своде на техничке, технолошке, социјалне, економске, еколошке, медицинске и многе друге. Ова чињеница указује да оптимизација представља област коју је могуће применити на апсолутно све проблеме, што представља и највећу снагу ове истраживачке дисциплине. Према задацима оптимизације методе је могуће класификовати на следећи начин [2]:

1. Задаци оптимизације могу бити статички и динамички. Статичка оптимизација нема зависност од времена, што значи да у математичком моделу не фигурише променљива времена, док динамичка оптимизација има директну зависност од времена. Код динамичке оптимизације објекат оптимизације је зависан од времена.
2. Према управљачким параметрима оптимизационе задатке могуће је поделити на једнодимензионалне, вишедимензионалне, задатке са дефинисаном почетном тачком и задатке са непознатом почетном тачком. Једнодимензионални оптимизациони проблеми су проблеми са једном променљивом. Број димензија одређен је бројем променљивих и тако код вишедимензионалних проблема подразумева се да број променљивих буде већи од један. Ако проблем има до пет променљивих, онда су то задаци малих димензија, ако има између 5 и 20, онда су то задаци средњих димензија. Задаци са преко 20 променљивих представљају

задатке великих димензија. Дефинисање почетне тачке дефинише и правац конвергенције оптимизационих метода.

3. Према функцији циља оптимизацију је могуће поделити на једнокритеријумску, вишекритеријумску, диференцијабилну, недиференцијабилну, са локализованим екстремумима, непознатом локализацијом екстремума, са једном екстремном вредношћу, са више екстремних вредности, са експериментално одређеним екстремумом и са аналитичким задатом функцијом циља.
4. У зависности од ограничења оптимизације проблеме је могуће поделити на задатке без ограничења, оптимизационе задатке са ограничењима (једнакостима и/или неједнакостима) и оптимизационе задатке са функционалним ограничењима.
5. Подела према методама оптимизације представља можда и најзначајнију поделу за овај рад. Методе оптимизације деле се на аналитичке, нумеричке, графичке, експерименталне и хеуристичке.
6. Према врстама метода поделу је могуће извршити на линеарно програмирање, нелинеарно програмирање, динамичко програмирање, комбинаторно програмирање, оптимално резервирање, теорију игара, мрежно планирање и управљање, хеуристичко програмирање, итд.

У литератури је могуће срести још велики број подела према неким критеријумима. Поделе имају за циљ да потпомогну процес оптимизације, јер је на основу подела могуће уочити које методе и поступци су најпогоднији за конкретне проблеме. У процесу оптимизације прво се одлучује које критеријуме метода мора да задовољи, потом се узимају у обзир све предности и недостаци које одређена метода нуди. Данас су све методе које су у употреби нумеричке природе и захтевају употребу рачунара у хардверском и софтверском смислу. Ова чињеница не потцењује старије и данас слабо коришћене методе јер оне могу веома користити за развој комплетног процеса оптимизације.

Линеарно програмирање

Методе линеарног програмирања развијене су у првој половини прошлог века. Ове методе су веома експлоатисане без обзира на њихова ограничења. Служе за решавање само линеарних проблема оптимизације. Могуће је применити их и на неке нелинеарне проблеме за које је могуће извршити њихову линеаризацију. Пошто се линеаризацијом нелинеарних проблема постиже само апроксимативно решење, овај поступак се зове линеарно апроксимативно програмирање [10]. Како су реални проблеми скоро увек нелинеарни, ове методе немају перспективу, али без обзира на то и даље су у примени.

Нелинеарно програмирање

Код нелинеарног програмирања математички модел мора да садржи бар једну функцију (функцију циља или ограничења) која је нелинеарна. Углавном код практичних примера већи број функција је нелинеарног карактера [10, 11]. Постоје аналитичке и нумеричке методе нелинеарног програмирања, а велики број ових метода је и данас у примени. У групо нумеричких нелинеарних метода могуће је сврстати између осталих и хеуристичко програмирање. Ове методе могу решавати нелинеарне проблеме оптимизације. Пошто су све хеуристичке методе оптимизације засноване на сличном принципу функционисања, а њихове могућности превазилазе класичне нелинеарне проблеме, хеуристичко програмирање постоји као независна група.

Динамичко програмирање

Динамичко програмирање бави се решавањем проблема зависних од времена. Такође, динамичко програмирање користи се за решавање проблема вишеетапних процеса [12, 13]. Динамичко програмирање датира од 1957. године [14].

Хеуристичко програмирање

Хеуристичко програмирање се користи за непотпуно дефинисане оптимизационе проблеме, као иницијалном идејом развоја ове групе метода. Представља поступак који на основу елементарних чињеница обезбеђује проналажење комплексних решења. Хеуристика на бази елементарних чињеница обезбеђује долажење до сложених истина. Хеуристичко програмирање омогућава постизање решења за проблеме који су лоше структуирани у некој мери, а и када друге методе не делују ефикасно. Ове методе ће обезбедити конвергенцију, која макар може дефинисати правац кретања вектора, а то може допринети и бољем структурирању проблема. За примену ове групе метода неопходно је само експлицитно дефинисати редослед рачунских и логичких операција на исти начин како би човек поступно решавао проблем.

Мрежно планирање и управљање користи се првенствено за организацију и управљање производњом [11].
Оптимално резервирање користи се за решавање проблема поузданости техничких система, а базирано је на минималној потрошњи средстава [10].
Теорија игара примењује се за решавање проблема са више потпуно различитих, супротних циљева, попут конкурентности на тржишту [11]. Теорија игара има значајну улогу у развоју вишекритеријумске оптимизације.
За примену већег броја оптимизационих метода подразумева се употреба рачунара.

1.5 Примена и начин функционисања оптимизационих метода

Традиционалне методе су у све мањој практичној примени због својих ограничења. Ове методе могу се углавном употребљавати само на једноставне, познате проблеме оптимизације, што у инжењерској пракси не даје неку употребну вредност. Оријентација у овој области је усмерена ка атрактивним, модерним методама оптимизације, што најчешће подразумева хеуристички приступ. Модерне методе оптимизације обавезно подразумевају софтверску имплементацију и употребу рачунара.
Начин имплементације оптимизационе методе своди се на три функционалне фазе софтвера, а то су предпроцесирање, процесирање и постпроцесирање. Ближе речено прво је потребно обезбедити адекватну интеграцију оптимизационог проблема у процес оптимизације кроз математички модел који га дефинише. Други корак је примена оптимизационе методе и конвергенција ка оптимуму. Трећи корак је заустављање алгоритма по неком унапред дефинисаном критеријуму и обезбеђивање доступности потребним информацијама о процесу оптимизације и решењима актуелног проблема.
У почетној фази оптимизационог процеса, предпроцесирању, дефинише се оптимизациони проблем. То се постиже дефинисањем броја и типа променљивих оптимизације и дефинисањем опсега променљивих уношењем функције циља, или функција циљева ако је реч о вишекритеријумској оптимизацији. Потребно је дефинисати и ограничења, као и њихов облик. Следећи корак предпроцесирања

подразумева избор оптимизационе методе који се врши углавном према врсти и типу проблема. Потребно је дефинисати и функционалне параметре оптимизационе методе и критеријум заустављања.
У фази процесирања оптимизациони алгоритам почиње са радом, врши се генерисање иницијалних вредности, поставе се услови за успешан рад алгоритма и врши се примена једначина и поступака на којима се алгоритам заснива. Алгоритам врши измене иницијалних вредности променљивих оптимизације, а решење конвергира оптималном све док се алгоритам не заустави.
На самом крају процесорског дела неопходно је да се рад алгоритма заустави по неком критеријуму. Природа оптимизационих алгоритама, са акцентом на хеуристичке методе оптимизације, је таква да конвергенција може трајати бесконачно. Зато је неопходно експлицитно дефинисати критеријум или критеријуме на основу којих ће се алгоритам зауставити. Водећи рачуна о успешности процеса оптимизације веома је тешко одредити прецизан критеријум конвергенције (заустављања) који ће оптимизацију учинити ефикасном, јер критеријуми зависе од конкретног оптимизационог проблема. Основни критеријуми за заустављање рада оптимизационог алгоритма су дефинисана тачност оптимизационих резултата, успорена конвергенција, дефинисано време рада алгоритама и број итерација алгоритама. Заустављање алгоритама засновано је на поменутим критеријумима, њиховом комбинацијом или неком варијацијом ових критеријума. Сваки од критеријума има и предности и недостатака, а њихов избор треба бити дефинисан у зависности од конкретног проблема. Најчешћи критеријум који се примењује је број итерација, јер се према зависности овог критеријума дефинише и ефикасност алгоритама и врше њихова испитивања.
Када се рад алгоритма заустави, почиње фаза постпроцесирања, која подразумева визуелизацију свих значајних резултата, њихово управљање, обраду и тумачење. У овој фази важно је постићи доступност резултата за комплетан оптимизациони процес, како би било могуће уочити евентуалне пропусте и недостатке при оптимизацији.
Софтверска имплементација и рад оптимизационих алгоритама увек имају исти или сличан приступ, а сваки алгоритам мора да садржи фазу предпроцесирања, процесирања и постпроцесирања.

1.6 Мотивација

Инжењерско тржиште оријентисано је на успостављање многобројних критеријума у погледу оптималних карактеристика, као што су цена, време, маса, запремина, радне карактеристике, животни век (период експлоатације) конструкција и други. То је основни разлог из кога проистиче мотивација и потреба за истраживањем у области оптимизације машинских конструкција како би се проширила перспектива реализације конструкција оптималних карактеристика, а све у циљу постизања бољих карактеристика уз истовремене уштеде. Машинско конструисање и машинске конструкције подразумевају решавање задатака противуречних циљева, а то је могуће решити једино применом оптимизације. Да би се приступило развоју неке конструкције потребно је да за њом постоји тржишна потреба. Тржиште намеће захтеве инжењерима за испуњење актуелних потреба у свакодневном животу корисника. Конструкцијама оптималних карактеристика могуће је створити потребу на тржишту. Креирањем конструкција које су квалитетне, имају поуздан рад, дуг експлоатациони век, оптималне габарите и масу, оптималну цену и друго, појавила би се тржишна потреба за њима. За овакав развој конструкција, доминантан је хеуристички приступ, којим је могуће

створити нове, оригиналне и другачије конструкције, а да оне притом буду оптималне, а временом и тржишне. То указује на примену процеса оптимизације и у фазама развоја конструкција и касније у њиховој експлоатацији.
Оптимизација као наука развијана је веома дуго, али без обзира на то, ова област и даље није довољно истражена. У данашње време, примена и развој оптимизације је у експанзији и ова научна област представља велики и захтеван изазов. Алтернативни приступ подразумева и потенцира употребу хеуристичких метода оптимизације, на шта је и оријентисан овај рад. Разлози примене хеуристичких метода оптимизације засновани су на њиховим особинама, карактеристикама и флексибилности коју пружају.
Основни изазов и оријентација ка хеуристичкој оптимизацији налази се у истоветности људског процеса учења новог, одлучивања и процеса размишљања. Психолошки посматрано, сваки човек према новим стварима и проблемима има хеуристички приступ, који евентуално потискује када је научен типским стварима и понашању или када преовлада искуствени приступ. Искуствено одлучивање само по себи јесте хеуристичко, али шаблони понашања не морају бити.
Основна мотивација за рад на овом истраживању, глобално посматрано, јесте подстицање употребе оптимизације, као науке и дисциплине због стварних квалитета које овај процес пружа, примена на конкретним машинским конструкцијама и развој ове дисциплине. Највећи допринос могуће је постићи развојем хеуристичких метода које постижу оптимум. У овом раду представљене су хеуристичке методе оптимизације, њихови начини функционисања, њихова популарност и перспектива примене. Посебан осврт је у правцу примене хеуристичких метода оптимизације на машинским конструкцијама. Предмет ове докторске дисертације представља проблем проналажења ваљаних или боље речено оптималних решења у области машинског конструисања и машинских конструкција.
Циљеви су да се поставе и задовоље неке елементарне хипотезе и да се кроз њихову реализацију постигне и стварни допринос у области оптимизације машинских конструкција.
Прва хипотеза дефинише могућност избора оптимизационе хеуристичке методе која ће дати значајне резултате. Ова хипотеза односи се на проналажење методе са потенцијалом примене и употребном вредношћу, методе која пружа флексибилност модификације и адаптације проблемима које је потребно решити.
Друга хипотеза представља могућност формулације методологије којом је могуће селектовати хеуристичке методе – извршити избор оптимизационе хеуристичке методе за примену код машинских конструкција. Из групе хеуристичких метода могуће је издвојити – фаворизовати једну методу, најзначајнијих карактеристика у погледу комплетног оптимизационог процеса.
Трећа хипотеза односи се на могућност практичне примене избора хеуристичке методе и постизање значајно бољих резултата у односу на почетне вредности практичног проблема код машинских конструкција. Ова хипотеза указује на значај примене хеуристичке оптимизације у процесу машинског конструисања и код машинских конструкција.
Четврта хипотеза дефинише могућност модификације хеуристичке методе, међусобно комбиновање - хибридизацију или развој нове хеуристичке методе у циљу постизања жељених резултата у погледу понашања алгоритма (другачија конвергенција, тежина имплементације, превазилажење локалних екстремних вредности, итд.). Реализацијом ове хипотезе могуће је постићи највећи допринос. Потребно је разумети хеуристику и оптимизацију, познавати хеуристичке и оптимизационе процесе и апсолутно владати њима, па тек онда приступити креативном процесу развоја.

Пета хипотеза указује на постојање великог простора за проширење примене метода хеуристичке оптимизације машинских конструкција и простора за даља истраживања. Ова хипотеза се највише односи на саму примену хеуристичких метода оптимизације код машинских конструкција, разноликост потенцијалне примене и доприноси које пружа оптимизација конкретних проблема.

Ове хипотезе дефинишу правац истраживања, а основни задатак ове докторске дисертације је да потврди и подробно објасни постављене хипотезе.

Ова докторска дисертација подељена је на неколико значајних целина. Уводни део садржи основне информације, значајне термине и развој оптимизације кроз историју. Потом је представљена математичка формулација оптимизационих проблема, општи преглед значајних метода оптимизације и рад нумеричких оптимизационих алгоритама. Друга значајна целина односи се на преглед најзначајнијих хеуристичких метода оптимизације. Представљени су начини и критеријуми избора три методе које су детаљно разматране из групе великог броја хеуристичких метода. Представљени су преглед литературе, детаљи рада, начини извођења и софтверска имплементација за одабране методе.

Следећа целина односи се на развој у области хеуристичких метода оптимизације. Ова целина представља детаљну анализу саме хеуристике, начине и могућности измене и напретка у овој области, као и примену ових закључака на конкретан развој. Извршене су модификације хеуристичких метода, комбиновање хеуристичких метода - хибридизација, као и развој потпуно нове хеуристичке методе оптимизације, а сва решења су потпуно оригинална што и представља највећи допринос. Акценат ове целине је на развоју и примени метода хеуристичке оптимизације.

Четврта целина представља развој потпуно оригиналног софтвера у који су имплементиране хеуристичке оптимизационе методе. Овај корак је веома значајан јер је оригиналан и пружа потпуну управљивост методама и рад са методама хеуристичке оптимизације, као њихов развој. Дефинисане су фазе предпроцесирања, процесирања и постпроцесирања са описом кључних сегмената који ове фазе чине функционалним.

Пета целина се односи на тестирање рада хеуристичких метода према препорукама из литературе. Дати су примери из литературе релевантни за тестирање ових метода. Извршена су тестирања развијених метода на одабраним примерима, која на првом месту имају задатак да верификују рад развијених техника и метода. Добијени резултати упоређени су са резултатима одабраних хеуристичких метода и изведени су закључци о напретку и квалитету развијених метода. На основу ових резултата могуће је фаворизовати неку методу и направити поређење између метода. Представљени су и поступци којима је могуће извршити избор и закључак која метода даје значајније резултате.

Следећа целина односи се на хеуристичку оптимизацију машинских конструкција и примену доминантне хеуристичке методе. У овом делу представљен је проблем инжењерске оптимизације и стварна примена оптимизације. Стварана примена оптимизације изведена је на три групе примера. Прва група представља инжењерске проблеме из литературе и овај сегмент је веома значајан за верификацију развијених метода. Верификација подразумева упоређивање резултата са резултатима из литературе, а сами примери су реално репрезентативни и оптимизационо веома захтевни. Друга група представља примену метода на проблем запремине зупчастих преносника снаге. Пример је преузет из литературе, али је адаптиран и прилагођен развијеном софтверу и методама у циљу постизања бољих резултата. Овај проблем је потпуно реалан и екстремно захтеван, а значајно је да постоје бар неки резултати у литератури, како би било могуће извести закључке о раду метода код комплексних проблема. Трећа група представља практичну и потпуно оригиналну примену метода

хеуристичке оптимизације на конструкцију циклоредуктора. Развијен је комплетан математички модел и реализован цео процес оптимизације за решавање проблема одређивања зазора међу елементима циклоредуктора, циклозупчаника и ваљака. Овај поступак такође представља развој нечег потпуно новог и оригиналног, а сам поступак односи се на примену развијених хеуристичких метода машинских конструкција и развој комплетне оптимизације код конкретног проблема машинског конструисања.
На основу постигнутих резултата изведени су закључци којима је могуће дефинисати допринос постигнут овим истраживањем. Направљен је преглед рада и представљени су доминантни закључци за сваку целину понаособ. Анализирана је компатибилност рада са постављеним хипотезама и дате су смернице значајне за истраживања која следе.

2. Хеуристичке методе оптимизације

Хеуристичке методе оптимизације развијене су како би убрзале и олакшале процес постизања оптималног решења. Оне су углавном конципиране на опонашању природних процеса и појава. То су симулирања генетских процеса, кретања колоније мрава, еволуциони процеси, опонашање имунитета и слични процеси и појаве. Ово опонашање чини хеуристику још интересантнијом и приступачнијом кориснику. Хеуристичке методе за практичну примену обавезно се софтверски реализују, тачније, за њихову примену неопходна је употреба рачунара. До данашњих дана развијено је више хеуристичких метода и њихов број непрестано расте, а свака од њих пружа одређене предности. Њихова карактеристика је да имају велику флексибилност у погледу математичког модела, функције циља, броја и типа ограничења, броја и типа променљивих оптимизације и по питању осталих сегмената у процесу оптимизације.
Многобројна истраживања оријентисана су и на поступке тестирања, одређивања методологије или упоредне анализе за одређивање конкретне хеуристичке методе која је погодна за решавање неког практичног проблема или групе проблема. Поједини аутори траже начине да тестирају сам рад методе, док други покушавају да изврше избор међу методама. Овакви приступи најзаступљенији су у инжењерској пракси. Нешто што представља изазов за све истраживаче је модификација, комбиновање или стварање – развој потпуно нових хеуристичких метода са циљем превазилажења недостатака постојећих хеуристичких метода. Хеуристичких метода има више, а најзначајније и најзаступљеније су представљене у овом раду. Дате су основне информације о хеуристичким методама, а на основу постављених критеријума, изабране методе су детаљно описане и разматране.
У литератури се често срећу различити термини, а односе се на исту ствар. Тако се методе оптимизације називају оптимизационим алгоритмима, хеуристичке методе еволуционим методама, а користе се и многи други термини. Хеуристичко програмирање и хеуристичка оптимизација, методе математичке оптимизације и оптимизационе методе, итд., све су примери употребе различитих термина за исту ствар. Не могу се у потпуности сматрати синонима, али означавају исто и подједнако су заступљени у литератури. И у овом раду поменути термини су подједнако заступљени.

2.1 Најчешће коришћене хеуристичке методе оптимизације

Популационо оријентисане хеуристичке методе могуће је поделити у две карактеристичне групе: еволуционе алгоритме – *EA* и алгоритме интелигентних честица – *SI*. Хеуристичке методе данас еволуирају и добијају нове облике и извођења. Најзначајније хеуристичке методе представљене су у овом раду.

TS метода – Претрага табуа (*Tabu Search*)

Метода претраге табуа је развијена са циљем да се не понављају решења која су лоша. Додавање вредности у табу листу врши се елиминисањем старе вредности системом први додаш - први избациш (*FIFO - first in-first out*). Табу претрага тражи решења која су боља од вредности из листе, а која не постоје у табу листи [15]. У основи је градијентна метода са сопственом меморијом. Рад се заснива на матрици вредности држава или стања (*states*) и базиран је на памћењу посећених и нежељених држава или утврђивању тренутног стања. Ове вредности складиштене су у табу листи. Кључни параметри претраге табуа су: дефинисање вредности, простор око ње и укупна дужина табу листе. Постоје и два додатна параметра које користи претрага табуа, а то су: аспирација - тежња (*Aspiration*) и диверсификација - прерасподела (*Diversification*). Аспирација се користи када су у исто време међусобно суседне вредности укључене у табу листу, селекцијом нових вредности. Диверсификација даје стохастички правац иначе детерминистичкој претрази. Алгоритам претраге табуа има карактеристику да у случају када не конвергира, претрага се насумично рестартује.

SA алгоритам – Симулација жарења (*Simulated Annealing*)

Метода симулације жарења развијена је за решавање сложених комбинаторних проблема. Ова метода развијена је око 1980. године [16]. Симулација жарења представља методу опонашања принципа жарења / каљења. Назив ове методе потиче од истоименог физичког процеса, а груписање честица врши се самостално топљењем. Инспирација за развој ове методе заснована је на вероватноћи величине A и B које имају сопствену енергију EA и EB на истој температури T. Највећа ефикасност ове методе је код проблема мрежне конфигурације. Са повећањем система повећава се и ефикасност методе симулације жарења. Метода има добре карактеристике превазилажења локалних екстремних вредности и брзу конвергенцију ка оптимуму.

ES алгоритам – Стратегија еволуције (*Evolution Strategies*)

Метода стратегије еволуције заснива се на креирању иницијалне популације. Једна од најважнијих и доминантних операција претраге је мутација. Мутација издваја најбољу индивидуу из популације. Основни параметри овог алгоритма су: број родитеља у генерацији и број потомака створених у генерацији. Рад стратегије еволуције базиран је на операцијама над реалним бројевима, за разлику од почетног концепта генетског алгоритма који ради са бинарно записаним вредностима (бинарним стрингом). Метода стратегије еволуције развијена је 60-их година [17].

DE метода – Диференцијална еволуција (*Differential Evolution*)

Метода диференцијалне еволуције развијена је пуно после методе еволуционе стратегије [18]. Диференцијална еволуција је развијена као алтернатива еволуционе стратегије. Овај алгоритам оријентисан је на претрагу по целој ширини поља, где је важан међусобни однос између случајно одабраних вредности популације. На решења изабрана по целом пољу претраге мутација има велики утицај, где важи и обрнуто. У

ситуацијама када су случајна решења одабрана из уског поља претраге мутација неће имати значајног утицаја на решења, а тиме и конвергенција.

IA алгоритам – Алгоритам имунитета (*Immune Algorithms*)

Алгоритам *IA* развијен је као алтернатива симулирања учења и памћења. Овај алгоритам има много сличности са генетским алгоритмом, који је и представљао узор за његов развој. Принцип његовог рада потиче од заштитних механизама живих организама у борби против бактерија и вируса. Овај алгоритам ради са антителима, а нове јединке које се селекцијом дефинишу као антитела, клонирају се без промена за даљу употребу. Мутација се примењује на клинове и врши се замена насумичних јединки. Врши се процена афинитета, а могуће је применити елитизам као заштиту најповољнијих решења. Овај алгоритам [19] предложио је Фармер (*Farmer*) 1986. године.

ACO метода – Колонија мрава (*Ant Colony Optimization*)

Алгоритам *ACO* развио је Дориго (*Marco Dorigo*) [20]. Овај алгоритам инспирисан је природним опонашањем колоније мрава. Користећи феромонске (*pheromone*) стазе или путање, мрави се оријентишу и крећу између две тачке (А и В) најкраћим путем. Мрави дакле означе путању којом ће се кретати. Пролажењем више мрава кроз једну тачку нагомилава се феромон, а мрави који тек треба да дођу увек бирају стазу са више феромона. После сваког мрава количина феромона проверава се функцијом циља за сваку тачку.

ABC метода – Колонија пчела (*Artificial Bee Colony*)

Овај алгоритам развио је Карбоџа (*Darvis Karboga*) [8, 21], 2005. године. Овај алгоритам презентује процес размножавања у колонији пчела. Доминантне фазе алгоритма су укрштање и мутација. У колонији пчела постоје матице (*queen-bee*), трутови (*drones*) и радници (*workers*). Матица је усмерена на размножавање, док трутови представљају мушке јединке које се паре са матицом. За креирану популацију трутови су случајно изабране јединке из популације и укрштање се врши на основу разлика вредности функције циља матице и трутова. У другој фази алгоритма креирање новог легла врши се применом процеса мутације. Нова генерација пчела радника дефинише се критеријумима селекција.

HS метода –Хармонијска претрага (*Harmony Search*)

Методу хармонијске претраге развио је Гим (*Yong Woo Geem*) са групом аутора 2001. године [22]. Аналогија овог алгоритма је са хармонијом у музици. Метода имитира процес музичке импровизације, где музичари унапређују сопствене инструменте у циљу постизања хармоније.

GA алгоритам – Генетски алгоритам (*Genetic Algorithm*)

Овај алгоритам ради по принципу природне селекције у генетици. Опонашањем ове појаве резултати који се могу постићи су стохастички, а поступак постизања ових решења хеуристички. Концепт и идеју генетског алгоритма предложио је Холанд (*J. H. Holland*) [5]. Касније, највећу улогу у развоју генетског алгоритма имали су Голдберг (*D. Goldberg*) [6] и Де Јонг (*De Jong*) [23]. Стратегија рада овог алгоритма заснована је на механизмима генетике и природне селекције са три основна оператора: селекција, укрштање и мутација. Овај алгоритам пружио је највећи допринос у историји хеуристичке оптимизације, како за решавање комплексних оптимизационих проблема, тако и за развој саме оптимизације. Овај алгоритам је један од предмета детаљне анализе, што је и представљено у даљем раду.

PSO метода – Рој честица (*Particle Swarm Optimization*)

Алгоритам *PSO* ради иницијализацијом популације случајно одабраних вредности. Овај алгоритам генерише честице (случајно дефинисане вредности) које се налазе у домену проблема. Алгоритам има високу ефикасност конвергенције и претрага се врши по целом пољу у оквиру домена. Алгоритам роја честица су развили Кенеди (*J. Kennedy*) и Еберхарт (*R. Eberhart*) средином деведесетих година прошлог века [7]. Принципи ове методе засновани су на опонашању роја по узору на инсекте и птице. Развијена решења груписана су у рој и крећу се кроз поље претраге уз међусобну интеракцију. Овај алгоритам најчешће се примењује код инжењерских проблема. Ова метода је једна од најзаступљенијих и зато је и она детаљно разматрана у овом раду.

MBA алгоритам (*Mine Blast Algorithm*)

Алгоритам *MBA* такође представља популационо оријентисани алгоритам [24]. Овај алгоритам развио је Садолах (*Ali Sadollah*) са групом аутора 2013. године. Принцип експлозије мине (гранате) искоришћен је за развој овог алгоритма и решавање комплексних оптимизационих проблема.

BA алгоритам (*Bat Algorithm*)

BA алгоритам развили су Јанг (*X. S. Yang*) и Гандоми (*A. H. Gandomi*) [25, 26], 2012. године. Алгоритам је заснован на понашању слепих мишева и њиховим чулима у потрази за храном. Слепи мишеви шаљу сигнале како би одредили сопствену позицију, препреке и храну. Фреквенција сигнала може бити променљива. Даље, они имају одређену брзину, а све ове појаве узор су за дефинисање конвергенције BA алгоритма.

WCA алгоритам (*Weather Cycle Algorithm*)

Алгоритам *WCA* [27, 28] развијен је 2012. године, првенствено за проблеме са ограничењима и инжењерске проблеме. Овај алгоритам инспирисан је природним процесом зависности кружења воде у природи. Принцип је кретање воде низводно

потоцима и рекама, све до уливања у мора, на основу чега је формирана конвергенција. Овај алгоритам развио је Ескандар (*Hadi Eskandar*) са групом аутора.

LOA алгоритам (*Lion Optimization Algorithm*)

Већина метахеуристичких алгоритама развијена је око 2015. године, па тако и *LOA* алгоритам. Развили су га Јаздани (*Maziar Yazdani*) и Џолаи (*Fariborz Jolai*) [29]. Основни узор рада овог алгоритма је специфични начин живота лавова и њихове особине функционисања у чопорима. Начин лова, обележавање и освајање територије, понашање лава и лавице у чопору, промена чопора, издвајање јединке из чопора, неки су од карактеристичних сегмената за овај алгоритам. Овај алгоритам је веома перспективан.

HTS алгоритам (*Heat Transfer Search*)

Алгоритам *HTS* заснован је на законима термодинамике и основама преноса топлоте. Овај алгоритам развили су Пател (*Vivek K. Patel*) и Савсани (*Vimal J. Savsani*) [30]. Начин претраге према овом алгоритму врше молекули система у интеракцији у термалном систему. Алгоритам садржи фазе кондукције, радијације (зрачења) и конвекције. Пошто је овај алгоритам развијен 2015. године, представља један од најефикаснијих и најатрактивнијих алгоритама.

PVS алгоритам (*Passing Vehicle Search*)

PVS алгоритам ослања се на дуготрајна искуства која су развијана у области транспорта и кретања возила на аутопуту у два правца. Аутори овог алгоритма [31] П. Савсани (*Poonam Savsani*) и В. Савсани (*Vimal Savsani*) сматрају да су превелика искуства у овој области и да се могу искористити за област метахеуристичке оптимизације. Ова метода је такође популационо оријентисана и фазе овог алгоритма дефинисане су кроз такозване услове. Испуњавањем одређених услова постиже се одговарајућа конвергенције. Овај алгоритам је веома перспективан, а с обзиром да је развијен 2015. године, а публикован 2016., његови квалитети ће тек доћи до изражаја.

JAYA algoritam (*Jaya*)

Веома ефикасна метода хеуристичке оптимизације развијена 2016. године је *Jaya*. Ову методу развио је Рао (*R. Venkata Rao*) [32]. Највећи допринос ове методе у области оптимизације је што ова метода има само једну фазу, која је разумљива и једноставна и не захтева никакво подешавање параметара. То значи да је ову методу једноставно имплементирати, а што је још важније и употребљавати. Аутор у истраживањима објашњава да назив алгоритма значи победа.

TLBO алгоритам (*Teaching-learning-based optimization*)

Алгоритам *TLBO* представља један од новијих еволуционих алгоритама који је привукао велику пажњу научника и истраживача. *TLBO* је савремена хеуристичка метода оптимизације која се у литератури среће од 2011. године [33]. Творац ове методе је Р. В. Рао (*R. V. Rao*), а метода је превасходно креирана за решавање инжењерских оптимизационих проблема. Принцип рада ове методе заснован је на операцијама над популацијом, што значи да је и ова метода популационо оријентисана. Метода се састоји из две кључне фазе: учитељска и студентска фаза. Ово указује да сама метода опонаша процес учења у разреду, што практично и јесте еволуциони процес. Ова метода је једна од најзначајнијих у савременој науци и зато је детаљно анализирана у даљем раду.

Овде су наведене неке од најзначајнијих хеуристичких метода оптимизације. Потребно је поменути још и *COA* алгоритам (*Cuckoo Optimization Algorithm*), *BFO* метода (*Bacteria Foraging Optimization*), *BBO* алгоритам (*Biogeography-Based Optimization*), *BBC* алгоритам (*Big Bang-Big Crunch*), *BFO* алгоритам (*Bacterial Foraging Optimization*), *FA* алгоритам (*Firefly Algorithm*), *LCA* алгоритам (*League Championship Algorithm*), *GSS* алгоритам (*Golden Section Method*), *ISA* алгоритам (*Interior Search Algorithm*), *YYPO* алгоритам (*Yin-Yan-Pair Optimization*). Наведен је већи број метода (скоро све развијене методе), а то значи да укупан број није заправо велики и да се ова област веома споро развија. Сам развој квалитетних метода представља озбиљан и екстремно захтеван истраживачки изазов. Анализом атрактивности ове области и дужина њене заступљености, могуће је рећи да број ефикасних хеуристичких метода није велики. Када се узме у обзир квалитет рада ових метода, све оне воде оптимуму, али нису све методе квалитетне за решавање целокупног спектра постојећих и потенцијалних оптимизационих проблема. Развој хеуристичких метода оптимизације, углавном, оријентисан је само на тачно одређену групу проблема. Тешко је приликом развоја методе узети у обзир све потенцијалне проблеме оптимизације. Овакав приступ доводи до одређених недостатака, где развијена метода нема ефикасан рад за проблеме за које није развијена. Имајући у виду ове чињенице, веома је мало развијених метода које се могу применити као опште, за широк спектар оптимизационих проблема.
Многе од метода које се примењују за решавање реалних и практичних проблема имају многобројне недостатке попут потешкоћа у превазилажењу локалних екстремних вредности, ризик од дивергенције, потешкоће при руковању и управљању ограничењима, комплексност програмирања (имплементације), ограниченост на групе проблема, итд. Традиционалне методе код комплексних реалних проблема најчешће уопште и не постигну оптимум, бар не у разумном временском интервалу. За неке проблеме ове методе није могуће ни применити. Један од великих практичних проблема оптимизације јесте непознавање оптималног решења, заправо онога што се очекује као резултат оптимизације. У пракси је најчешће немогуће одредити шта је оптимум, а ако је оптимум и познат онда најчешће и не постоји потреба за оптимизацијом. За оптимизацију је важно да постоји конструкција, процес или проблем (објекат оптимизације), или бар потреба да се нешто од тога реализује. Од комплексности објекта оптимизације директно зависи и сложеност поступка оптимизације и математичка формулација проблема [9, 34]. Поменуте проблеме у досадашњој пракси успешно превазилазе хеуристичке методе оптимизације. За решавање проблема оптимизације неопходно је извршити правилан избор методе оптимизације [2, 3, 35-39] који се врши у зависности од постављеног проблема или групе проблема. Највећу перспективу имају хеуристичке методе због својих карактеристика и могућности примене на широк спектар оптимизационих проблема. Потребно је имати у виду жељене критеријуме оптимизације

и циљеве који дефинишу шта се оптимизацијом жели постићи. У зависности од конкретног проблема треба имати у виду и управљивост објектом оптимизације.
Поред базичних хеуристичких метода оптимизације и њихових основних извођења, честа истраживачка тема су и модификације самих метода, као и њихово међусобно комбиновање (креирање хибрида), а све у циљу побољшања конвергенционих карактеристика и комплетних перформанси метода. Овде је представљена група метода веома значајних и актуелних у области оптимизације, са перспективом примене на машинске конструкције.

2.2 Предности метода *GA, PSO* и *TLBO*

Поступак инжењерске оптимизације је сам по себи веома комплексан, а један од праваца развоја инжењерске оптимизације је и рад на поједностављењу и убрзавању оптимизације. Ово је разлог зашто је неопходно дефинисати неке базичне чињенице оптимизационог процеса, јер је то покушај да се објасни поступак и дефинишу кораци које је потребно следити у процесу оптимизације. За оптимизацију је неопходно да постоји објекат оптимизације (машина, процес, потреба, делатност, итд.). Потреба за оптимизацијом објеката сама настаје, а могуће је и свесно створити ову потребу. Објекат оптимизације може бити баш све што је могуће у некој мери математички дефинисати, што указује на чињеницу да је скоро све могуће оптимизовати. Математичка формулација објекта оптимизације (позитивних или негативних ефеката) дефинише критеријум оптимизације. Оптимизација представља одређивање екстремне (оптималне вредности) уз испуњење реалних ограничења, тако да критеријум оптимизације значи одређивање екстремне вредности функције циља која описује проблем уз истовремено испуњење постављених ограничења. Следећи важан сегмент је да објекат оптимизације буде управљив. То значи да математички модел мора да садржи променљиве, а ваљаним проналажењем вредности променљивих оптимизације постиже се и оптимална вредност функције циља, а самим тим и оптимални објекат оптимизације. Да би било могуће реализовати овај процес потребно је применити неку методу оптимизације. Метода оптимизације адаптира се објекту оптимизације, или обрнуто, и на тај начин могуће је постићи оптимум. Следећи корак је верификација резултата оптимизације. Овај сегмент је најодговорнији у процесу оптимизације, јер зависи од свих претходно наведених сегмената и ослања се у највећој мери на искуство инжењера. Најчешћа грешка је што инжењери покушавају да пронађу „најбоље решење“, а не оптимално. Најбоље решење не мора уопште бити могуће, практично оправдано и прихватљиво и из тог разлога овакво решење не мора искључиво бити оптимално.
Сложеност процеса оптимизације зависи од многих фактора, а неки од њих су: сложеност система који се разматра, број утицајних параметара на понашање система, услови у којим систем може да функционише, веродостојност математичке формулације, квалитет и адаптивност методе оптимизације и визуелизације и начин прегледа резултата.
Како процес инжењерске оптимизације подразумева исправно доношење одговорних одлука неопходно је испунити следеће услове, јер мора да постоји [35]:

- проблем,
- доносилац одлука,
- потреба да се проблем реши,
- алтерантивно решење проблема.

Процес инжењерске оптимизације представља невероватно перспективну област јер је овим поступком могуће извести виртуелни експеримент који је неупоредиво бржи и јефтинији од експеримента на реалном моделу, а што је још важније овим процесом проналазе се алтернативна решења, што је много теже или немогуће код експеримената на реалном моделу.
Разноликост инжењерских проблема и проблема конструисања захтева правилан избор оптимизационе методе за примену у овој области. Није могуће и бесмислено је за сваки проблем понаособ развијати нову методу и нови софтвер за решавање проблема, али сам развој метода је логичан као адаптација хеуристике проблемима. Хеуристичке методе имају добре карактеристике и могу се применити на све проблеме само уз мање или веће адаптације конкретном задатку. За избор хеуристичке методе важну улогу имају комплексност имплементације, брзина конвергенције ка оптимуму, брзина рада, постигнута тачност резултата, спектар примене на групе проблема, флексибилност, ефикасност, ефективност и друге. Издвајање и избор методе има за циљ да се фаворизује хеуристичка метода која пружа највише погодних карактеристика наспрам уложених напора. Рад свих хеуристичких метода и по питању брзине и по питању квалитета решења даје изузетне резултате. То указује на чињеницу да избор једне методе представља тежак и озбиљан задатак.
Основни задатак је избор хеуристичке методе оптимизације која ће бити погодна за практичну примену и имати добре карактеристике гледано кроз постигнуте резултате. Други задатак је дефинисати поступак, методологију којом је могуће направити ваљани избор хеуристичке методе. При избору методе за оптимизацију морају бити задовољени следећи услови [40]:

- Одабрана метода и математички модел морају бити компатибилни,
- Конвергенција мора бити задовољавајућа у временском или итерацијском смислу,
- Мора постојати могућност дефинисања критеријума заустављања алгоритма,
- На познатим примерима, за разуман временски интервал или број итерација, метода мора да постигне задовољавајућу тачност,
- Уложени напори морају бити мањи од добити оптимизационог процеса,
- Метода мора бити лако управљива у погледу имплементације и контроле параметара,
- Метода мора да подржи и задовољи сва могућа и постављена ограничења,
- Метода мора да има што већу универзалност за што већи број проблема.

Побројани критеријуми презентују општу слику за избор неке методе, а при стварном избору неопходно је темељно анализирати додатне стварне критеријуме који су важни.
Из групе хеуристичких метода неопходно је издвојити најзначајније, истакнуте, и атрактивне методе, јер није могуће и није потребно анализирати сваку методу посебно како би се извели одређени закључци и унапредио сам процес оптимизације. Ова дисертација оријентисана је на групу хеуристичких метода као најперспективнијих и метода најбољих карактеристика. Да би методе из ове групе биле издвојене као најзначајније неопходно је дефинисати конкретне критеријуме. Селекција је сведена на три методе које ће репрезентативно приказати развој, рад и могућности хеуристичких метода. Три одабране методе могу да представе развој хеуристике кроз историју, покажу њихове конвергенционе карактеристике, примену код инжењерских проблема,

разноликост хеуристичких појава, као и многе друге карактеристике целокупне групе хеуристичких метода оптимизације.

Први критеријум је популарност методе. Овај критеријум односи се на заступљеност методе у литератури, примени методе на различите проблеме, развоју и анализи методе. Како су хеуристичке методе већ дуже време у примени, истраживачи који се баве овом оптимизацијом користе поједине методе за решавање конкретних инжењерских проблема. Поред тога истраживачи раде на модификацијама и унапређењу појединих метода. Из групе метода пажња се посвећује само некима од њих, због карактеристика, могућности и перспективе које метода пружа. Ове издвојене методе којима је посвећена највећа пажња сматрају се најпопуларнијим методама. Овај критеријум селекције извршен је на основу оцена за сваку методу која је базирана на доступности и количини литературе за сваку методу понаособ.

Други критеријум је време појављивања и дужина развоја методе. Веома је важно да се методе изаберу из три различита периода како би се видео правац кретања хеуристичке оптимизације и који је пут напретка. На самом почетку развоја хеуристичке методе имале су ограничења код рачунарске примене. Водило се рачуна о заузимању меморије, брзини рада и сличним стварима, које су данас занемарљиве. Поред тога, при развоју метода пажња се посвећивала што већој контроли и управљивости алгоритама, а алгоритми из тог периода имају велики број параметара које корисник мора исправно да подеси. Савремене хеуристичке методе имају велику хардверску подршку која им обезбеђује брз рад и велики капацитет меморије. Оријентација развоја савремених метода је у правцу што мањег броја параметара, како би се смањио утицај инжењера у процесу оптимизације. Ови различити приступи указују на чињеницу да је неопходно груписати методе према периоду настанка базичног алгоритма и анализирати њихов рад, како би се уочио период највећег доприноса области хеуристичке оптимизације. Овај критеријум анализиран је прегледом литературе и праћењем метода према периоду настанка.

Трећи критеријум је различитост појаве и приступ при развоју методе. Ова појава може се дефинисати као хеуристичност и разноликост хеуристичке појаве. У периоду настанка еволуционих метода све методе развијане су на приступу опонашања неке врсте еволуције. У периоду рада са честицама, развијени су алгоритми колоније мрава, колоније пчела, роја честица и птица, итд. Савремене методе оријентисане су на опонашање појава које нас директно окружују, као што су кружење воде у природи, начин учења и слично. На основу овог критеријума извршено је груписање метода по категоријама различитости и врсти појаве на којој је базирана хеуристика, а циљ је одабрати хеуристичке методе које ће имати потпуно различит приступ, како би се уочио значај неке појаве на рад самог оптимизационог процеса.

Четврти критеријум је успешност методе да конвергира за проблеме са великим бројем локалних екстремних вредности, јер су овакви проблеми најчешћи у инжењерској пракси (мултимодалне функције). Овај критеријум оријентисан је на избор методе за примену код инжењерских проблема. Избор се може извршити на основу примене метода у литератури на инжењерске проблеме, у складу са резултатима које пружају. Без обзира што оптимизација представља мултидисциплинарну област, у пракси овим проблемима и развојем метода најчешће се баве инжењери. Најкомплекснији практични проблеми оптимизације такође су инжењерски проблеми. Зато је значајно уочити које методе су развијене циљно за примену код инжењерских проблема и које методе су најчешће примењене за инжењерске проблеме. Када велики број истраживача изврши селекцију неких метода, то је сигурна потврда да оне поседују потребне квалитете. Овај критеријум такође је реализован анализом хеуристичких метода оптимизације у литератури.

За прелиминарни избор хеуристичких метода које ће бити обухваћене овим радом постављена су четири критеријума за које је сматрано да имају највећи утицај у процесу развоја хеуристичких метода. Ови критеријуми утичу и на модификације, хибридизације и развој нове методе, што је такође саставни део ове дисертације.
Извршен је избор три методе, *GA, PSO* и *TLBO* методе које ће на репрезентативан начин презентовати рад комплетне групе хеуристичких метода оптимизације. Избор је извршен на основу анализе литературе, тако да постављени критеријуми буду испуњени.

2.3 *GA* алгоритам

Генетски или генетички алгоритам како га поједини аутори код нас називају представља хеуристичку методу оптимизације. Ова метода је од изузетног значаја, јер опстаје и употребљава се веома дуго и доживела је можда и највећу еволуцију међу хеуристичким методама од свог изворног облика. Рад генетског алгоритма заснован је на опонашању природних, еволуционих, генетских процеса. Метода је заснована на принципима генетике, а основне фазе овог алгоритма су:

- Селекција (*Selection*),
- Укрштање (*Crossover*) и
- Мутација (*Mutation*).

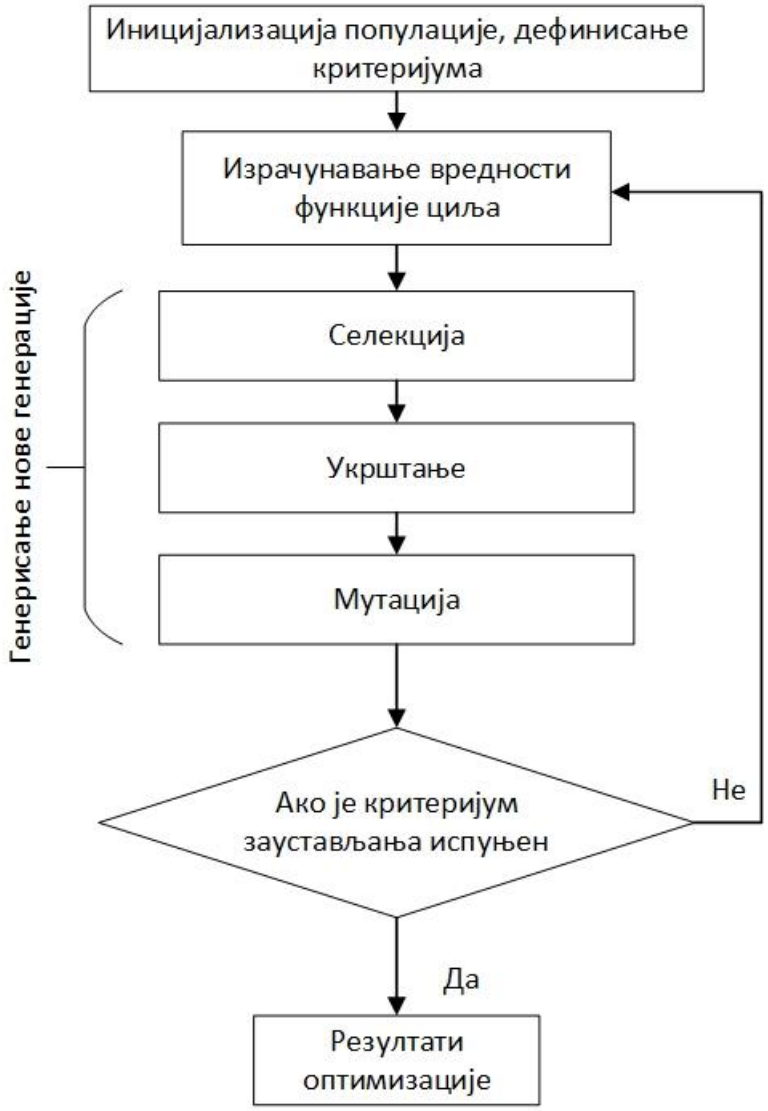
Данашњи облици генетског алгоритма називају се реално кодирани генетски алгоритми –*RCGA* (*Real Coded Genetic Algorithm*), што значи да овај алгоритам врши операције над реалним вредностима, за разлику од традиционалних бинарно кодираних генетских алгоритама – *BCGA* (*Binary Coded Genetic Algorithm*), где се операције врше над бинарним стринговима. Рад савремених алгоритама, *RCGA*, је неупоредиво бржи и лакши за софтверску имплементацију, а саме операције и поступци алгоритма одвијају се на исти начин као код *BCGA.*
Пошто је генетски алгоритам често коришћен у пракси, виђају се ситуације обостране адаптације. Алгоритам се прилагођава проблему и проблем се прилагођава алгоритму. Ефикасан алгоритам мора да садржи све фазе и то правилно имплементиране, како би процес оптимизације био задовољавајући. Пропусти и занемаривање неке фазе углавном узрокују спору или онемогућену конвергенцију или концентрисање решења око локалне екстремне вредности, чиме се не постиже оптимум. Генетски алгоритам је метода са пуно различитих извођења, а самим тим и поседује велики број параметара које је неопходно исправно подесити због конвергенционих карактеристика. Поседовање великог броја параметара може се сматрати основним недостатком ове методе, јер захтева велико искуство и знање корисника овог алгоритма. Због ове чињенице јавља се могућност спровођења неуспешног оптимизационог процеса. Некада је потребно уложити превелике напоре наспрам добити да би се извршило ваљано подешавање параметара алгоритма.

2.3.1 Историјски развој генетског алгоритма и основни појмови

Развој еволуционих алгоритама датира од средине 20. века. Генетски алгоритам заснован је на еволуционој стратегији, а 60-их година 20. века сматра се да су развијени први принципи мутације. Творац генетског алгоритма је Холанд (*Johan H. Holland*), који је са својим сарадницима развио методу генетског алгоритма. Прва значајна публикација везана за овај алгоритам [5] публикована је 1975. године, где је представљен принцип рада генетског алгоритма. Први модел генетског алгоритма радио је са бинарним стринговима, са једноставним укрштањем у једној тачки, једноставном пропорционалном селекцијом и једноставном мутацијом. Овај облик генетског алгоритма сматра се основним генетским алгоритмом, а сви његови принципи користе се и данас у његовој структури.

2.3.2 Структура и делови генетског алгоритма

Генетски алгоритам је веома значајан алгоритам како за примену, тако и за учење о хеуристичком правцу оптимизације. Рад генетског алгоритма заснива се на креирању нових генерација и замене претходних, дакле итеративном поступку какав је заступљен и код других хеуристичких метода оптимизације. Свака нова генерација представља скуп решења функције циља, а повећањем броја генерација, повећава се и тачност решења. Како је *GA* заснован на природном еволуционом процесу, обавезно садржи фазе селекције, укрштања и мутације. Ове фазе базиране су на познавању генетских процеса. Што је већа разлика међу јединкама то је шире поље претраге. Теоријски, код *GA* боље јединке имају већу шансу да опстану у популацији, на основу искустава из генетике. Ако се циљно потенцира на задржавању најбољих јединки, овај процес назива се елитизам. Рад генетског алгоритма заснован је на издвајању (селекцији) јединки које ће учествовати у процесима репродукције (укрштања). У генетској структури могуће су промене генетског материјала мутацијом, чија је суштина проширење поља претраге и превазилажење локалних екстремних вредности. Општа структура функционисања генетског алгоритма представљена је на слици 2.1.



Слика 2.1. Структура генетског алгоритма и приказ фаза алгоритма

На слици 2.1 представљена је само општа структура алгоритма која је увек оваква, а у зависности од начина извођења зависе конкретни детаљи реализације ових фаза.
Суштина фазе селекције по дефиницији је одређивање вероватноће појављивања и учествовања јединке у новим генерацијама. Постоји директна веза између квалитета јединке и вероватноће да ће она опстати кроз генерације. Што јединка даје боље решење, то је већа шанса њеног опстанка. По овом принципу се добар генетски материјал преноси у нове генерације, а лош одбацује. На овај начин решење конвергира оптимуму. У овај поступак неопходно је увести одређени опрез. Према овој логици треба одбацити сва лоша решења, а фаворизовати добре јединке. Овакав приступ доводи до превремене конвергенције, заправо сужавању поља претраге на локалне екстремне вредности. Постоји много врста селекција и подела у оквиру њих. Према начину преношења генетског материјала селекцију код генетског алгоритма могуће је поделити на [41]:

- Генерацијске селекције – директна селекција бољих јединки чији се генетски материјал преноси у следећу генерацију.
- Елиминацијске селекције – бирање лошијих јединки за елиминацију.

Број јединки кроз генерације најчешће остаје непромењен. Елиминисане јединке замењују се новим јединкама добијеним укрштањем.

Укрштање код генетског алгоритма представља процес комбиновања постојећих јединки у циљу добијања потпуно нових јединки, попут родитеља и њиховог потомства. Нове јединке наслеђују особине својих родитеља. Због одржања величине популације (из практичних разлога), најефикаснији начин је да се од два родитеља добију два потомка која их замењују. Најједноставнији, али функционалан и ефикасан вид укрштања је укрштање у једној тачки. Најлакши начин презентовања ове појаве је на бинарном стрингу, што је и представљено на слици 2.2.

Слика 2.2. Укрштање у једној тачки код *BCGA*

Из целе популације издвоје се парови који се укрштају и пресечна тачка у којој се врши укрштање и то на случајан начин. Ако се генетски алгоритам бинарно изводи, стрингови морају бити истих дужина. Кроз генерације могуће је радити са различитом величином популације, али је најпрактичније одржавати је на истој вредности.

Мутација, као фаза генетског алгоритма, представља веома важан и неизбежан процес за успешну конвергенцију. Најчешће се ова фаза изводи потпуно независно од селекције и укрштања. Мутацијом се постиже неконтролисана промена генетског материјала. Променом генетске структуре постижу се потпуно нова решења. Основни циљ је добити јединку коју није могуће добити другим фазама. Ова случајна вредност иницира претрагу по целом дозвољеном домену, а решења не могу превремено конвергирати. Заступљеност мутације мора бити мала од око 0,001% до 0,01%, да се претрага не би свела на случајан, стохастички и неконтролисани поступак. Пример једноставне мутације једног гена код *BCGA* представљен је на слици 2.3.

Слика 2.3. Једноставна мутација једног гена код *BCGA*

Код генетског алгоритма многе модификације алгоритма које су нашле примену у пракси третирају се као варијанте алгоритма. Овим радом представљено је опште

функционисање самог алгоритма и неке од могућих варијанти, без детаљног описивања сваког од могућих извођења. Генетски алгоритам је такође много примењиван и често доступан и у домаћој литератури, па отуда терминологија која презентује поједине сегменте алгоритма.

2.3.3 Начини извођења и модификације генетског алгоритма

Карактер генетског алгоритма је да успешно превазилази локалне екстремне вредности, а време рада алгоритма је неупоредиво краће у односу на традиционалне методе. У основи, традиционални генетски алгоритам ради са бинарним стрингом, док нова решења представљају поступак рада са реалним бројевима. Генетски алгоритам има могућност лаке паралелизације и погодан је за програмирање. Због своје структуре има брзу конвергенцију у пракси, а никада превремену. Генетски алгоритам је настао као последица развоја метода еволуционе стратегије (*Evolution Strategy*). Представља значајан корак за развој хеуристичке оптимизације.

Селекција као саставна целина генетског алгоритма може бити изведена као пропорционална и рангирајућа селекција. Пропорционална селекција може се поделити на једноставну пропорционалну селекцију и стохастичку универзалну селекцију. Рангирајућа селекција дели се на сортирајућу и турнирску селекцију. Турнирска селекција на k – турнирску селекцију и једноставну турнирску селекцију. Сортирајућа на селекцију најбољих $((\mu,\lambda);(\mu+\lambda);$ крња селекција) и линеарно сортирајућу селекцију. Што се тиче укрштања, постоји велики број могућих извођења, а неки од њих су укрштање у једној тачки, укрштање у више тачака, униформно укрштање, укрштање матицом, итд. Мутацију је могуће поделити на једноставну, мешајућу, потпуну, инверзну, мутацију помоћу маске, итд.

Извођење генетског алгоритма са укрштањем у две тачке за проблеме са ограничењима представио је Реид (*D. J. Reid*) [42]. Велики допринос развоју генетског алгоритма поспешила је првенствено његова примена на проблеме решеткастих носача [43-47]. Развијан је и генетски алгоритам са већим бројем популација [48, 49], што представља почетке паралелно програмираног генетског алгоритма. Генетски алгоритам развијан је и за проблеме вишекритеријумске оптимизације [50-54]. Интересантна је примена генетског алгоритма интегрисаног у *CAD* окружење [55]. Развијане су многобројне модификације генетског алгоритма [45, 56-64], као и нови хибриди са генетским алгоритмом [53, 65-70] за различите групе проблема. Кроз литературу [71-73] је представљен генетски алгоритам за примену код различитих проблема и врста математичког модела. Општи приказ структуре генетског алгоритма и начина његове имплементације [74, 75] представља велики значај за практичну употребу ове методе. Генетски алгоритам где је анализирана и унапређена фаза укрштања представљен је у литератури [76-82]. Генетски алгоритам где је анализирана и развијана фаза (оператор) мутације [83] представља модификацију алгоритма. Подешавање параметара алгоритма је директно везано са операцијама алгоритма, али се у литератури срећу анализе само исправног њиховог подешавања [60, 84-86] и развоја алгоритма у циљу повећања ефикасности. Развијан је и генетски алгоритам за рад са целобројним и мешовитим променљивима [87], што је неопходно како би алгоритам било могуће применити на инжењерске оптимизационе проблеме.

2.4 *PSO* алгоритам

Алгоритам *PSO* представља популарну оптимизациону методу. Спада у групу хеуристичких метода оптимизације. Овај алгоритам се понекад тумачи као процес усмерене и контролисане мутације. Постоји само једна фаза алгоритма, али постоје два кључна параметра алгоритма и то су:

- Локација (*Location*) и
- Брзина (*Velocity*).

Параметри алгоритма дефинисани су за сваки временски тренутак. Теоријски посматрано овај алгоритам је једноставан за софтверску имплементацију и лак за коришћење при решавању конкретних оптимизационих задатака и то због разумног броја и експлицитно дефинисаних параметара алгоритма. У пракси, овај алгоритам није значајно једноставнији у односу на друге. Има веома добре конвергенционе карактеристике и то је разлог његове честе примене, а посебно у инжењерској оптимизацији. Поред тога, подешавање параметара алгоритма може се извршити једном од стране стручњака и тиме се отклонити утицај на неповољне конвергенционе карактеристике.
Интересантна је појава на којој је заснован овај алгоритам. Заснива се на принципу групе честица са одређеним карактеристикама које имају квантитативне вредности, где свака вредност представља оптимизационо решење. Кретањем честица мењају се и њихове вредности, а праћењем, контролом и усмеравањем ових честица решења конвергирају ка оптимуму.

2.4.1 Историјски развој алгоритма роја честица и основни појмови

Алгоритам *PSO* развијен је 1995. године и развили су га Кенеди и Еберхарт [7]. Овај алгоритам веома је важан и привукао је велику пажњу истраживача. Његов значај показује чињеница да се он и данас масовно употребљава у инжењерској пракси.
PSO алгоритам представља инспирацију многим истраживачима за развој нових алгоритама. За разлику од генетског алгоритма, овај алгоритам је заснован на раду са честицама, а конвергенција је заснована на њиховом усмеравању. Овај алгоритам отворио је нови правац хеуристичке оптимизације рада са честицама [88, 89]. Од самог појављивања *PSO* алгоритма он представља незаобилазну истраживачку тему у области развоја и примене оптимизационих метода.

2.4.2 Структура и делови алгоритма роја честица

Стандардни *PSO* алгоритам представља управљање честицама које воде оптимуму. Претрага за овај алгоритам врши се по целокупном дозвољеном домену. Карактеристичне величине неопходне за реализацију алгоритма су позиција и брзина. Овај алгоритам има само једну фазу. Важни детаљи засновани су на претходном убрзању честице, растојању од позиције честице од најбоље вредности дате честице и растојања / удаљености или позиције честице од позиције глобално најбоље честице. Позиција неке честице у датом тренутку представља потенцијално решење, а усваја се само најбоља позиција и преноси кроз оптимизациони процес. Ново решење засновано је на раду у складу са једначином (2.1).

$$X_{New,i} = X_{Old,i} + \upsilon_{New,i} \tag{2.1}$$

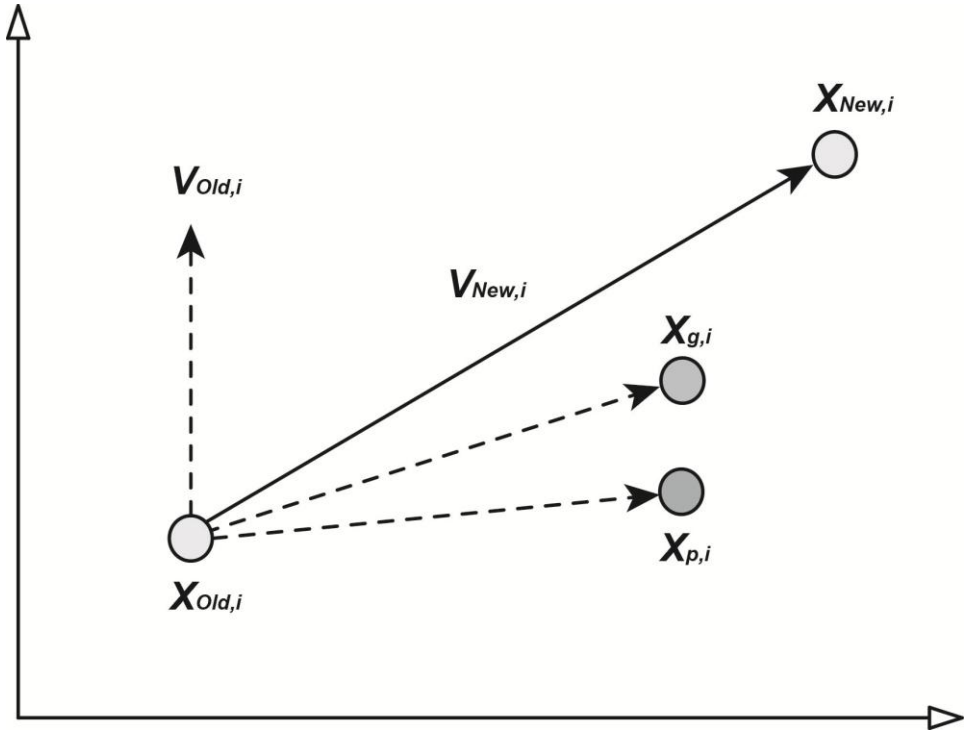
Где је:

$$\upsilon_{New,i} = \omega \cdot \upsilon_{Old,i} + C_p \cdot r_p\left(X_{p,i} - X_{x,i}\right) + C_g \cdot r_g\left(X_{g,i} - X_{x,i}\right) \tag{2.2}$$

Из претходног израза (2.2), ω представља такозвану инертност честица, док C_p и C_g представљају факторе убрзања. Фактори убрзања су позитивне константе које контролишу релативни утицај на честицу у локалном смислу, као и глобално усмеравање дате честице за коју су усвојене вредности. Случајне вредности r_p и r_g поспешују претрагу по комплетном дозвољеном домену и могу имати неку вредност из интервала од 0 до 1. Вредност X_{pi} представља претходну најбољу позицију јединке, X_{gi} је најбоља позиција за целу популацију, док је X_{xi} тренутна позиција. Вредност ω израчунава се као у изразу (2.3).

$$\omega = \omega_{max} - \frac{\omega_{max} - \omega_{min}}{Iteration_{max}} \cdot Iteration \tag{2.3}$$

У изразу (2.3), ω_{max} и ω_{min} су иницијална и коначна вредност инертности, $Iteration$ представља тренутну актуелну итерацију, док $Iteration_{max}$ представља укупан број итерација у процесу оптимизације. Најчешће, према препорукама, параметри ω_{max} и ω_{min} имају вредности 0,9 и 0,4 респективно. Вредности за ω дакле морају бити већи или једнаки 0 и мање од 1, док је $0 < C_p + C_g < 4(1 + \omega)$. Вредности C_p и C_g подешавају се као константе према искуству и испитивањима из литературе, а њихова препоручена вредност је 1,5 за обе константе. Постоје многобројна испитивања ефикасности алгоритма у зависности од подешавања ових константи, а ово је само једна од препорука. Најбољи начин презентовања рада PSO алгоритма је илустровање кретања честице, као што је представљено на слици 2.4.



Слика 2.4. Кретање честице и илустрација конвергирања код *PSO* алгоритма

Слика 2.4 илуструје исто оно што је представљено једначинама које описују рад *PSO* алгоритма. Постоје многи начини извођења овог алгоритма, а претходно је приказан његов основни облик, такозвани стандардни *PSO* алгоритам.

2.4.3 Извођења и модификације *PSO* алгоритма

Теоретски, *PSO* алгоритам је једноставан за имплементацију. Овај алгоритам практично представља поступак усмерене и контролисане мутације. Свака честица контролисана је са два параметра: локацијом и брзином, у сваком тренутку. Брзина је дефинисана са четири критеријума: претходна вредност брзине, најбоље решење честице, најбоља локална решења од тог тренутка и најбоља глобална решења за цео рој. *PSO* алгоритам представља основ за дефинисање интелигенције честица (*Swarm Intelligence*) [88, 89]. Пошто *PSO* алгоритам има велики број параметара од којих зависи квалитет и брзина конвергенције важно је исправно изабрати њихове вредности [90], чиме су се бавили Еберхарт и Ши (*Y. Shi*). Од 2000. године *PSO* алгоритам постаје све популарнији. Анализа је заснована на анализи конвергенције и исправном подешавању параметара [91]. Тек у периоду после 2005. почиње се са све масовнијим публикацијама у часописима. Напредни облици *PSO* алгоритма који су анализирани и испитивани на стандардним тест примерима постају све популарнији [92]. Овај алгоритам примењиван је и на практичне инжењерске оптимизационе проблеме [93-95], превасходно на проблеме структурне оптимизације. Анализа ових проблема вршена је стандардним *PSO* алгоритмом, његовим модификацијама и хибридима. У периоду око 2010. године, највећи акценат везан за овај алгоритам је стављен на развој варијанти вишекритеријумске оптимизације [96-100]. Унапређени *IPSO*, као модификација и

хибридни *PSO* (*GAPSO*), комбинован са генетским алгоритмом коришћени су за практичне инжењерске проблеме управљања [101]. У новије време развијен је велики број модификација и хибридизација *PSO* алгоритма [102-120]. Представљена литература је само део укупног броја истраживања везаних за *PSO* алгоритам. Најзначајнија извођења *PSO* алгоритма за ово истраживање су хибридни *PSO* алгоритам (*PSO-DE*) [121], *CVI-PSO* [122], *NM-PSO* [123] и *CPSO* [124]. Ове варијанте PSO алгоритма су коришћене за анализу постигнутих резултата у овом истраживању.

2.5 *TLBO* алгоритам

TLBO алгоритам представља савремену хеуристичку методу за решавање проблема оптимизације, а првенствено инжењерске оптимизације због чега је и развијен. Рад ове методе заснива се на опонашању процеса учења у школи, преношењу знања са учитеља на студенте и интеракција студената у разреду. Овај процес подељен је у две кључне фазе:

- Учитељску фазу (*Teacher Phase*) и
- Студентску фазу (*Learner Phase*).

Употреба овог алгоритма врши се прилагођавањем проблема алгоритму или прилагођавањем алгоритма проблему. Ова метода је популационо оријентисана (еволуциона), као и већина хеуристичких метода. То значи да се креира популација из које се издвајају учитељи и студенти, а сваки учитељ или студент представља реално решење скупа променљивих оптимизације. Фазе алгоритма су подједнако значајне при имплементацији алгоритма, без обзира што их је могуће независно имплементирати и што могу скоро у потпуности независно да функционишу. „Скоро“ се односи на проблеме конвергенције алгоритма попут превремене конвергенције, споре конвергенције нарочито при локализацији вредности и слично. Дакле, ове две фазе чине алгоритам потпуно функционалним као целину. Важно је имати у виду да је овај алгоритам „без параметара“. Једина интеракција са корисником је помоћу величине популације и броја генерација. Ова чињеница смањује могућност грешке људског фактора у оптимизационом процесу, а самим тим и побољшава процес оптимизације. Овакав приступ је веома перспективан при развоју нових алгоритама.
TLBO алгоритам је у пракси веома заступљен и пружа изузетне резултате, а метода је добра за примену за широк спектар оптимизационих проблема јер се претрага врши по целом допуштеном пољу претраге. Савременост, актуелност и чињеница да су *TLBO* развили експерти за област инжењерске оптимизације представљају основне разлоге и критеријуме избора баш ове хеуристичке методе.

2.5.1 Историјски развој *TLBO* алгоритма и основни појмови

TLBO алгоритам представља хеуристичку методу оптимизације која је привукла велику пажњу научне јавности. Ова метода је савремена, што конкретно указује и време када је развијена, 2011. године. Овај алгоритам развио је Рао (*Ravipudi Venkata Rao*) са Департмана за машинско инжењерство Националног техничког института из Сурата, Индија. Исте године са групом аутора је објавио и прве публикације везане за овај алгоритам [33].

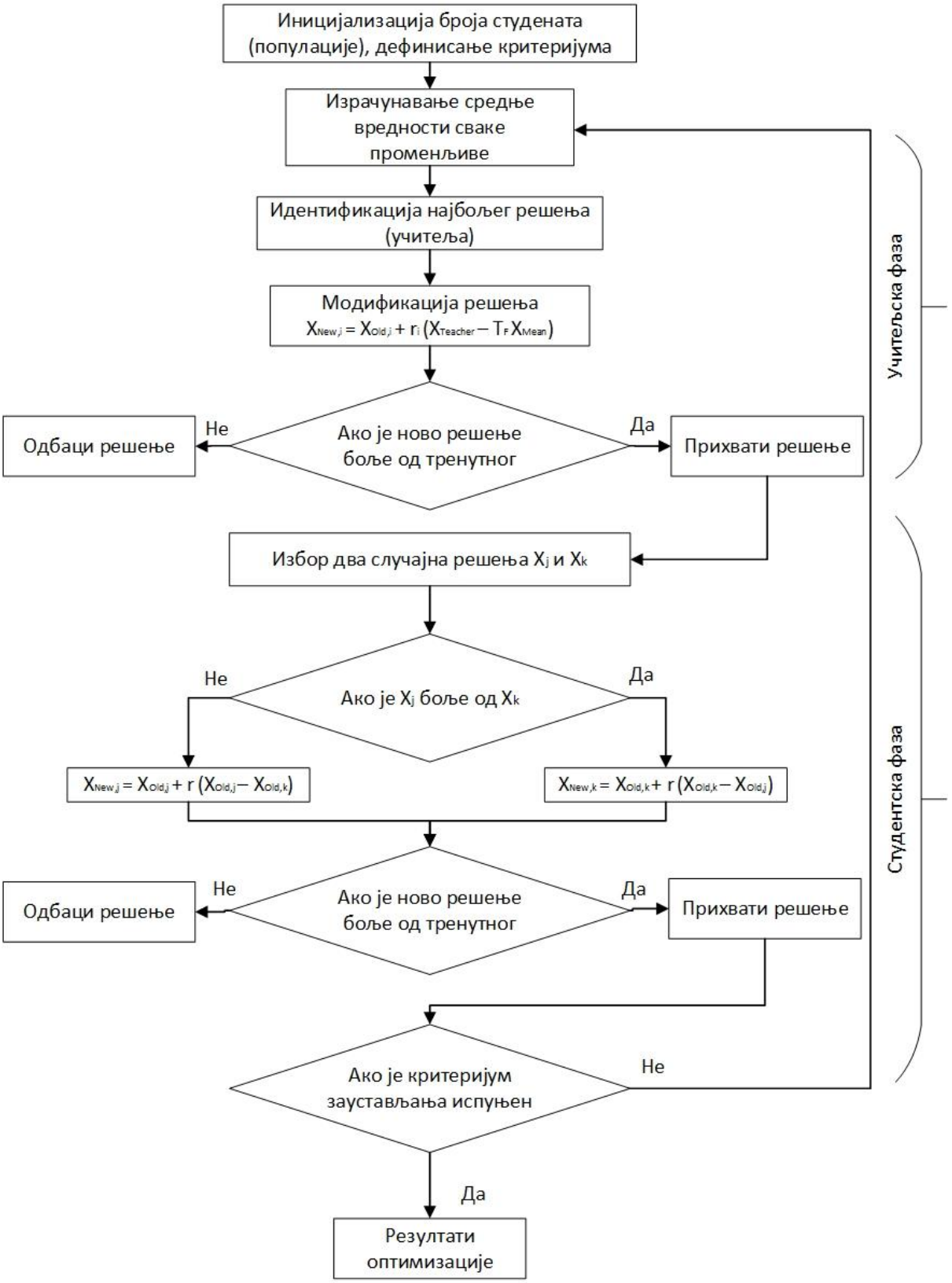
За овако кратко време употреба *TLBO* алгоритма среће се у машинском инжењерству, електро инжењерству, грађевини, производним процесима, компјутерској техници, електроници, физици, хемији, биотехнологији и економији [125]. Развој овог алгоритма одвија се великом брзином. Са првим публикацијама појавиле су се и прве критике [126] на извођење и рад овог алгоритма, где оне прелазе у дискусију и расправу [127], а све у циљу популаризације и унапређења овог алгоритма.

2.5.2 Структура и делови *TLBO* алгоритма

Поред бројних модификација ове методе могуће је дефинисати њен иницијални облик. Првенствено је потребно дефинисати појам иницијалног броја студената (*Number of Students*) – величине популације, који представља укупан број потенцијалних решења оптимизације. Из ове групе издваја се најбоље решење (*The Best Solution - Teacher*) које представља учитеља $X_{Teacher}$. Ова вредност неопходна је за учитељску фазу. Поред ове вредности потребно је израчунати средњу вредност (*Mean*) X_{Mean} за укупно решење, заправо скуп променљивих оптимизације. Средња вредност представља нови скуп променљивих, где свака променљива има средњу вредност. Алгоритам обавља своје задатке све док критеријум заустављања не буде испуњен. Критеријум заустављања је најчешће број генерација (*Number of Generation*) и дефинише колико ће се пута поновити све фазе алгоритма. Ова величина је целобројна, а њеним повећањем се повећава и тачност и квалитет резултата, али процес оптимизације се временски продужава.

Алгоритам учења је метода за постизање и проналажење оптималног решења. Свака нова генерација представља скуп решења оптимизације од којих је једно решење оптимално. *TLBO* алгоритам има две кључне фазе рада, а његова структура представљена је на слици 2.5.

Просечна вредност (средња вредност) квалитета разреда се повећава у зависности од квалитета учитеља. Под квалитетним учитељем сматра се онај који усавршава своје студенте до сопствених могућности у погледу знања. Напредовање разреда не зависи само од квалитета учитеља, на шта указује и друга фаза алгоритма, већ и од индивидуалних способности разреда.



Слика 2.5. Структура *TLBO* алгоритма

Учитељска фаза

Учитељска фаза *TLBO* алгоритма (алгоритма учења) представља прву фазу алгоритма и односи се на утицај квалитета учитеља на напредак студената у разреду. Сваки учитељ ће тежити да разред приближи средњој вредности, тако да се у свакој итерацији одређује нова средња вредност и најбоља јединка која ће представљати учитеља. Нова вредност сваког студената зависи од средње вредности, квалитета учитеља, и случајних вредности које помажу да студенти напредују. Начин на који се постижу нове вредности у овој фази за сваког студента и у свакој итерацији представљен је у једначини (2.4).

$$X_{New,i} = X_{Old,i} + r_i(X_{Teacher} - T_F X_{Mean}) \tag{2.4}$$

Нове вредности студента представљене су са $X_{New,i}$ за сваку итерацију, а претходне вредности студента са $X_{Old,i}$. Величина r_i представља случајну вредност у интервалу од 0 до 1. Такође, за сваку итерацију одређује се случајна вредност за T_F, која је целобројна и може бити 1 или 2. У свакој итерацији, за сваког студента примењује се ова једначина и утврђује се да ли је новодобијена вредност боља од претходне вредности. У случају да је нова вредност боља усваја се нова вредност, а у случају да нова вредност није боља од старе, задржава се стара вредност. Овим поступком је могуће постићи бољи скуп променљивих који ће у следећој итерацији бити третиран као учитељ.

Студентска фаза

Студентска фаза заснована је на интеракцији студената из разреда. Првенствено се врши избор два случајна студента из разреда $X_{Old,j}$ и $X_{Old,k}$. Њихове вредности се међусобно пореде и постоје два могућа случаја. Први случај је када је $X_{Old,j} > X_{Old,k}$ и тада се нова вредност студента добија према изразу (2.5).

$$X_{New,j} = X_{Old,j} + r\left(X_{Old,j} - X_{Old,k}\right) \tag{2.5}$$

Други случај је када је $X_{Old,j} < X_{Old,k}$ и тада се нови студент добија на основу израза (2.6).

$$X_{New,k} = X_{Old,k} + r\left(X_{Old,k} - X_{Old,j}\right) \tag{2.6}$$

У претходним изразима r представља случајну вредност у интервалу од 0 до 1. После израчунавања нове вредности неопходно је упоредити стару и нову вредност студента и одабрати бољу вредност. Овај поступак се примењује на већем делу популације или на целој популацији у свакој итерацији.
Алгоритам због конвергенционих карактеристика мора да садржи обе фазе, али се у свакој итерацији не морају променити све вредности студената у популацији. Алгоритам понавља ове две фазе све до испуњења критеријума заустављања.

2.5.3 Начини извођења и модификације *TLBO* алгоритма

Пошто је алгоритам учења веома примењиван у инжењерској оптимизацији, развијене су његове многобројне модификације и различити начини његове примене. Првенствено, рад са овим алгоритмом могуће је груписати према врсти проблема који се решава. Тако су Рао (*Rao*) и Пател (*Patel*) [128] направили извођење алгоритма које је примењено на комплексне оптимизационе проблеме са ограничењима. Поред тога, унапредили су алгоритам *TLBO* за примену код проблема без ограничења [129]. Рао је са групом аутора [130] применио *TLBO* за решавање проблема са континуалним – нелинеарним функцијама циља великих димензија. Поједини аутори [131, 132] одређивали су перформансе варијанти *TLBO* алгоритма. Ванг (*Wang*) је са групом аутора [133] развио *NSTLBO*, као варијанту унапређеног *TLBO* алгоритма. Верзију алгоритма учења под називом *DGSTLBO* [134] развио је Зоу (*Zou*) са групом аутора. Унапређени *ITLBO* [135] развили су Чен (*Chen*) и група аутора. Развијен је и *GBTLBO* [136] и модификација нове верзије *MGBTLBO*. Поједини аутори [136] анализирали су ефекте *TLBO* алгоритма у инжењерској пракси. *TLBO* са глобалним укрштањем *TLBO-GC*, представљен је у [137]. Једну од значајнијих верзија алгоритма са учитељским искуством звану *LETLBO* развили су [138] Зоу и група аутора. Постоји још пуно верзија овог алгоритма [139], испитивање конвергенционих карактеристика [140], јер је он заиста актуелан и атрактиван. Рао и Раи (*Rai*) [141] успешно су применили *TLBO* алгоритам за стварне оптимизационе проблеме. Развијена је још једна значајна модификација и унапређење [142] *ICTLBO* за оптимизационе проблеме са ограничењима. *TLBO* за вишекритеријумску оптимизацију развили су Рао и Пател [143], где је извршена модификација овог алгоритма. Зоу и група аутора [144] применили су *TLBO* за вишекритеријумску оптимизацију, поред њих неке верзије овог алгоритма срећу се у вишекритеријумској оптимизацији [145]. Такође, Пател и Савсани (*Savsani*) [146] су применили унапређени модел *TLBO* алгоритма за решавање проблема вишекритеријумске оптимизације и назвали га *MO-ITLBO*. Поједини аутори [147] применили су основни модел *TLBO* алгоритма, а неки аутори оријентисани су на истраживања хибридизације и примене *TLBO* алгоритма [148, 149]. Представљена литература је само део актуелних истраживања везаних за алгоритам учења. Рао је представио преглед литературе [141] где је забележио око 200 најзначајнијих истраживања везаних за *TLBO* алгоритам из свих поменутих сфера његове примене закључно са 2015. годином.
Из ове велике групе истраживања издвојено је неколико који се могу сматрати најзначајнијим за развој ове докторске дисертације.

Као што је поменуто [129] развијена је модификација која је оријентисана на повећање самог броја учитеља у раду алгоритма, као и њихово усавршавање. Ова модификација названа је *I-TLBO* (*Improved TLBO*).
Модификација звана *mTLBO* (*Modified TLBO*) [131] представља још један покушај да се перформансе базичног *TLBO* алгоритма унапреде. Унапређење се односи на студентску фазу и подразумева проширење ове фазе, као и увођење случајне вредности у интервалу од 0,5 до 1.
NSTLBO (*Neighborhood search TLBO*) [133] подразумева анализу тополошког окружења решења и процес мутације на крају итерације. Ова модификација показује изузетне резултате, а њене перформансе су детаљно анализиране.
Модификација *ITLBO* (*Improved TLBO*) [135] представља драстичну промену базичног *TLBO* алгоритма. Модификација се заснива на учењу окружења, тачније повећању броја

учитеља. Овакве модификације могу иницирати потпуно нове методе јер се у великој мери разликују од основне.
Модификација названа *TLBO-GC* (*TLBO with global crossover*) [137] заснована је на принципу глобалног укрштања, а резултати које ова модификација пружа су веома значајни.
Учитељско искуство потенцирано је за *LETLBO* (*TLBO with learning experience*) [138]. Ова модификација подразумева измену обе фазе алгоритма. Рад ове модификације тестиран је на тест примерима и практичним примерима, а добијена решења упоређена су са резултатима из литературе.
ICTLBO (*Improved constrained TLBO*) подразумева модификацију и проширење фаза *TLBO* алгоритма за оптимизацију проблема са ограничењима и ова модификација је веома успешна [142].

Варијанте *TLBO* алгоритма могу помоћи у даљим корацима модификације, хибридизације и развоја нове методе. Представљене варијанте могу бити узори, а свакако су од помоћи при развоју хеуристичких метода оптимизације. За анализу рада метода коришћена је основна верзија *TLBO* алгоритма, која и софтверски имплементирана, али су детаљно анализиране све модификације овог алгоритма.

3. Модификација, хибридизација и развој нове хеуристичке методе оптимизације

Хеуристичка оптимизација у потпуности је заснована на хеуристици као појму. Спој метода оптимизације и хеуристике као општег појма заснован је на карактеристикама које хеуристика има, што пружа оптимизационим методама могућност да уопште реше оптимизациони проблем, а поред тога да функционишу и са малим бројем познатих чињеница.

Процес примене и употребе хеуристичких метода подразумева њихово детаљно познавање. За почетак неопходно је дефинисати хеуристику и проанализирати шта хеуристика заправо представља и због чега је хеутистика доминантна у развоју оптимизационих метода.

Хеуристика, као приступ, оријентисан је ка коришћењу уопштених правила, нагађања на основу доступних података, базира се на интуицији или здравом расуђивању. Овакав приступ најчешће се среће у природи и еволуционим процесима, па изучавање хеуристичке оптимизације често захтева њихово познавање. Оно што је суштински важно је да хеуристика представља приступ где се са малим бројем познатих чињеница долази до квалитетног, комплексног оптималног решења. Појам хеуристика према литератури, у научне сврхе уведен је 1945. године од стране Г. Полија (*G. Polya*) [150]. Хеуристичке методе, које се чак и данас употребљавају и представљају основу развоја нових метода, развијене су у другој половини двадесетог века. У то време развијене методе нису идентичне данашњим хеуристичким методама које се примењују, а разлике и ограничења су углавном засновани на доступности компјутерске подршке оптимизацији. Хеуристичке методе развијене су како би пружиле боље конвергенционе карактеристике и могућност решавања до тада нерешивих, комплексних проблема оптимизације, за разлику од традиционалних метода, уз компензацију апсолутне тачности решења. За разлику од егзактних метода, хеуристичке методе су и у теоријској форми неупоредиво јасније јер се оријентишу на опонашање већ познатих природних процеса доступних човеку. Примарни недостатак хеуристичких метода базиран је на њиховој природи резултата, која често не пружа било какву гаранцију идентификације оптимума и саме конвергенције ка оптимуму. Хеуристичке методе никада не дају апсолутно тачно решење. Апроксимативна решења која се добијају овим методама могуће је третирати као квалитетне резултате, чак са неприметним одступањем од апсолутно тачног решења. Проблеми у пракси знају бити веома комплексни, а свако апроксимативно решење, чак и са великим одступањем од оптимума које води побољшању имају значај и допринос. Хеуристичке методе су у овим ситуацијама доминантне, јер уопште и могу да реше проблеме које другим методама није могуће решити.

Хеуристику је могуће дефинисати на велики број начина, а свака од дефиниција презентује основни карактер хеуристике. Хеуристика потиче од грчке речи „хеурикеин“ (εὑρίσκειν), што значи налазити или откривати. Све остале дефиниције имају исту орјентацију да на различите начине дефинишу овај комплексни свеобухватни појам.

Неке од дефиниција хеуристике су:

- Учење и откривање истине базирано на искуству,
- Проналажење приближног, довољно доброг решења,
- Наука о методама и принципима проналажења новог,
- Поступак који води према открићу,
- Процес којим је могуће решити неки проблем, без гаранције успешног решења,
- Поступак решавања нерешивих проблема,
- Кршење уопштених правила размишљања,
- Коришћење лако доступних информација,
- Психолошки процес доношења одлука.

Постоји још много дефиниција хеуристике, а уведени су и појмови који се користе као синоними хеуристици у процесу оптимизације, а можда то нису у потпуности. Један од појмова је метахеуристика коју је могуће дефинисати као комбинацију јасно дефинисаних правила и случајних појава како би се имитирала или опонашала нека природна појава. Овај термин се и у литератури најчешће користи као синоним хеуристици, бар у области оптимизације.
Веома је важно дефинисати, разумети и анализирати појам хеуристике, јер је то једини начин примене хеуристичких принципа за развој хеуристичке оптимизације.

3.1 Развој модификација хеуристичких метода оптимизације

Кроз историју хеуристичке оптимизације развијен је један број веома ефикасних хеуристичких метода оптимизације. Поставља се питање да ли уопште постоји потреба рада на унапређењу процеса хеуристичких метода развојем нечега новог. Свака од развијених метода има пуно предности, али и својих слабости. Поред тога неопходно је адаптирати методе оптимизације актуелним проблемима и потребама, који су све комплекснији, као и искористити све бољу компјутерску подршку која је доступна. Због ових чињеница постоји потреба за прилагођавањем хеуристичких метода оптимизације актуелном стању. Ове адаптације подразумевају модификовање метода оптимизације, кроз одређене измене у структури постојећих метода, како би оне дале боље перформансе. Поред вршења измена метода, тачније модификација, постоји могућност коришћења различитих предности које имају различите хеуристичке методе. Преплитањем ових предности, употребом већег броја метода развијају се хибридне методе, а овај процес назива се хибридизација. Овакви приступи дају боље резултате у односу на базичне методе, али наравно постоје и одређена ограничења. Свака метода носи своје недостатке и са повећањем комплексности и ширине спектра проблема недостаци су све израженији. Овде престаје могућност даље адаптације развијених хеуристичких метода и јавља се потреба за стварањем нове методе која може да превазиђе реалне проблеме. Развој хеуристичких метода оптимизације је дакле загарантован, али направити било какав помак у овом правцу представља изузетно тежак и комплексан задатак, без било какве гаранције успеха. Теоријски, за развој нове хеуристичке методе, потребно је уочити неки процес или појаву која није анализирана, математички је формулисати и софтверски имплементирати. У пракси то није баш тако.

Појаву која је ефикасна, коју човек стварно разуме и која није анализирана није лако наћи. Математичка формулација није увек усмерена у правцу који у оптимизацији даје добре резултате. Оно шта човек мисли да је адекватна математичка формулација не мора дати добре резултате у процесу оптимизације. Током овог процеса непрестано је потребно имати у виду да за развој хеуристичких метода мора да постоји хеуристички приступ. Потпуна математичка формулација и математичка анализа у развоју хеуристичких метода оптимизације може бити погрешна јер такав приступ гарантује удаљавање од хеуристике. Очигледно је да је овај процес веома комплексан, помало конфузан и не постоји гаранција успешности што додатно дестимулише истраживања у овој области. Перспектива развоја ове области је удаљавање у што већој мери од конвенционалног приступа и експериментисање са несвакидашњим идејама које могу дати резултате.

3.1.1 Развој модификације методе *GA* (*iGA*)

Анализом свих хеуристичких метода оптимизације кроз историју *GA* можда има и највећи значај. Ова метода је тема многобројних истраживања. Направљен је велики број модификација ове методе, а сама метода је масовно примењивана на великом броју практичних инжењерских проблема. Овај алгоритам због својих добрих карактеристика веома дуго траје и опстаје и у данашње време је у масовној примени. Што се тиче практичне употребе *GA*, он је имплементиран и у многе комерцијалне софтвере и једна је од најраспрострањенијих хеуристичких метода оптимизације. Све ове чињенице привлаче пажњу и стварају потребу за модификацијом *GA* која ће осавременити овај алгоритам и вратити му доминацију коју заслужује.

Генетски алгоритам има велики број варијанти и све оне могу се третирати као модификације у односу на основни бинарно кодирани генетски алгоритам. Основна тенденција при развоју модификације јесте смањење огромног броја параметара које генетски алгоритам има. Употреба *GA* код практичних проблема понекад се своди на подешавање параметара како би сам алгоритам ефикасно радио. Честе су ситуације да се код овог алгоритма уложи много више напора за прилагођавање алгоритма проблему него за решавање самог проблема. Дакле, постоји потреба да се *GA* сведе на задавање само основних величина за рад алгоритма, да се задрже фазе алгоритма, а да модификација пружи бољу ефикасност и стабилност у раду у односу на актуелни *GA*.

Развијена је потпуно нова, оригинална модификација методе генетског алгоритма. Модификација *GA* заснована је на фазама селекције, укрштању и мутацији, као и базична метода. Метода је развијана као реално кодирани генетски алгоритам због ефикасности рада са реалним бројевима. Задржава се рад са популацијом, а елитизам је заступљен само за очување најбоље вредности која се преноси током генерација и не мења се све до проналажења нове најбоље вредности. Концепција модификације подразумева измену редоследа рада фаза алгоритма, као инверзни приступ, па је модификација названа као *iGA*. Модификација *iGA* прво обавља фазу мутације, затим врши селекцију, па укрштање. Функција циља се прерачунава један пут за једну генерацију, а укупно у процесу оптимизације као производ величине популације и броја генерација. Искључене су све вредности параметара, а алгоритам подразумева аутоматска подешавања у зависности од дефинисаног проблема.

Мутација

Мутација је једна од незаобилазних фаза генетског алгоритма и веома је важна за нормалан рад и функционисање самог алгоритма. Мутација у општем смислу представља процес или поступак промене генетског материјала у неком облику. Како се мења генетска структура, постижу се потпуно нова решења, такозване јединке. Опонашањем природног процеса, где се мутација јавља на случајан начин, у структуру алгоритма се уводи ова фаза како би поспешила перформансе алгоритма. Циљ мутације је да се добију јединке које се у потпуности разликују од постојећих и које се не могу добити кроз остале фазе алгоритма. Тако добијене јединке проширују поље претраге, а решења нису локалног карактера. Основна идеја *iGA* је да се смањи број параметара чиме ће корисник лакше користити и управљати алгоритмом. Мутација је изведена да зависи од величине популације и броја итерација, тако да додатна подешавања неких параметара нису потребна. Према концепту *iGA* први корак алгоритма је мутација да би функција циља била прерачунавана само једном у току итерације. Мутација се врши у свакој другој итерацији, за случајно изабрану јединку која је различита од најбоље јединке из генерације. Процес избора јединки које ће бити мутиране представља врсту селекције, а заштита најбоље јединке врсту елитизма. Начин извођења мутације представљен је једначином (3.1).

$$X_{New,i} = X_{Old,i} \times X_{Mask} \tag{3.1}$$

Мутација се врши једноставном променом случајно одабране јединке множењем са маском. Вектор маска, X_{Mask}, је потпуно нови вектор, креиран од случајних вредности. Ова вредност има задатак да промени правац вектора изабране јединке и да се добије потпуно нова јединка. Нова вредност добијена мутацијом мора бити у дозвољеним границама уз задовољење постављених ограничења. Број јединки које се мутирају у свакој другој генерацији могуће је дефинисати као производ величине популације и константе 0,1. Ова вредност дефинисана је експериментално за општи случај оптимизације. Сигурно да би адаптацијом ове величине конкретном проблему конвергенција била боља, али за општи случај ова вредност даје изузетне резултате. Уочено је да је највећи проблем употребе генетског алгоритма подешавање параметара алгоритма и да процес оптимизације у великој мери зависи од корисника. Идеја *iGA* јесте да се смањи утицај корисника на квалитет резултата и да се направе опште поставке параметара које ће за све проблеме дати задовољавајуће резултате. Оваквим приступом се обезбеђује потребан и довољан број мутираних јединки које ће отклонити локализацију решења на погрешном месту, а ток оптимизације проширити на комплетно поље претраге и учинити га ефикаснијим.

Селекција

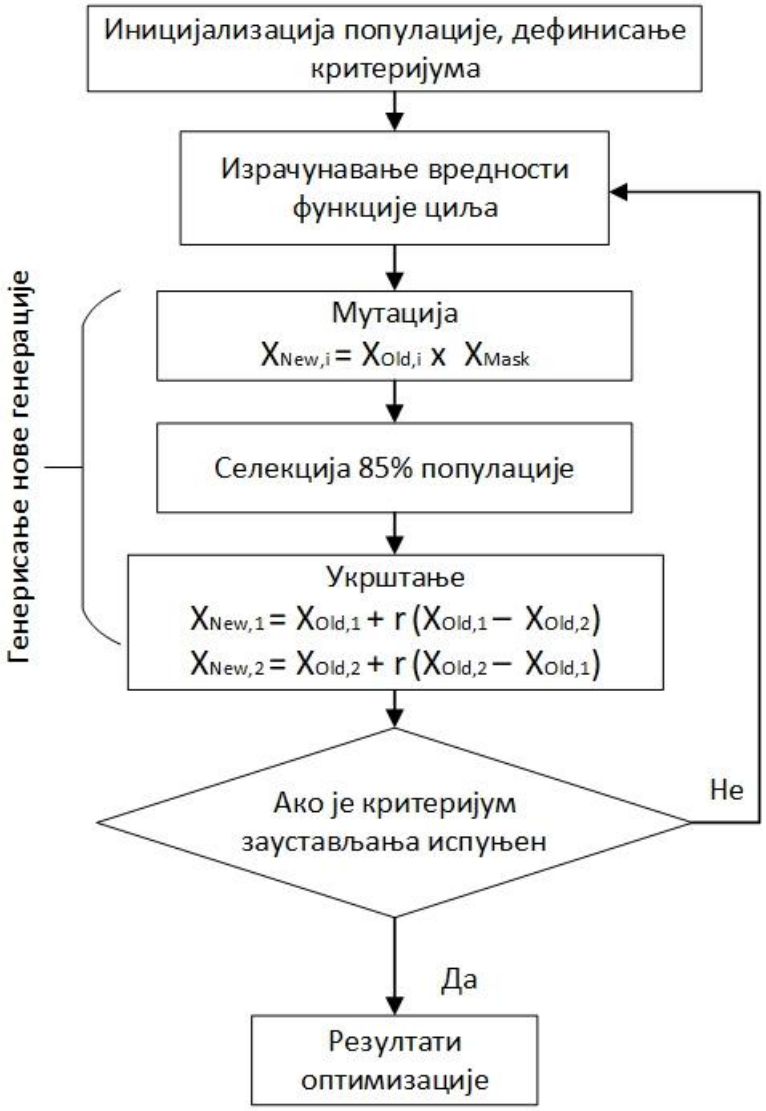
Пошто по дефиницији структуру генетског алгоритма чине селекција, укрштање и мутација, модификовани генетски алгоритам не одступа у том смислу од класичног генетског алгоритма. Селекција дефинише процес или поступак одабира јединки које ће учествовати у процесу укрштања. Правилним избором броја јединки које ће бити промењене у генерацији дефинише се у великој мери и ефикасност алгоритма. Модификација *iGA* подразумева случајан избор 85% јединки из укупне популације јединки које ће бити промењене. Овај проценат јединки улази у процес укрштања и добијају се нове вредности које чине нову популацију. Елитизам је заступљен само у мери да се увек сачува најбоља јединка у популацији све до појаве боље вредности која ће је заменити. Очигледно је да је овај вид селекције једноставан, а сам корисник нема могућност додатних подешавања ове фазе алгоритма.
Вредности дефинисане процесима селекције, укрштања и мутације дефинисане су на основу препорука из литературе и искуствено, а накнадним тестирањем *iGA* потврђено је да дају изузетне резултате.

Укрштање

Укрштање представља неки вид комбиновања постојећих јединки ради добијања потпуно нових, са итерацијама све бољих јединки које чине популацију. Укрштање се врши са две јединке, а потомство наслеђује особине од својих родитеља и на овом принципу је и заснован успех конвергенције код генетског алгоритма. Модификација *iGA* предвиђа да се од два родитеља добију два потомка у свакој генерацији над 85% популације. Потомци замењују своје родитеље и тако се формирају нове генерације. Поступак извођења укрштања, према *iGA* изведен је према изразу (3.2).

$$\begin{aligned} X_{New,1} &= X_{Old,1} + r\left(X_{Old,1} - X_{Old,2}\right) \\ X_{New,2} &= X_{Old,2} + r\left(X_{Old,2} - X_{Old,1}\right) \end{aligned} \tag{3.2}$$

Величине X_{New} представљају нове вредности које настају мутацијом. Јединке случајно одабране из популације које учествују у укрштању означене су са X_{Old}, а вредност r је случајна вредност која помаже конвергенцији по целом пољу претраге. Овим се добијају два нова потомка од два родитеља. Избор родитеља који ће бити међусобно комбиновани изведен је на потпуно случајан начин, чиме је обезбеђен хеуристички приступ оптимизацији. Ова модификација не одступа у великој мери од класичног генетског алгоритма, али је постигнут успех по питању елиминације параметара за подешавање и направљено реструктуирање самог алгоритма. Овај приступ код конкретне примене модификације даје значајно боље резултате у поређењу са класичним генетским алгоритмом.
Општа структура модификације *iGA* представљена је на слици 3.1.



Слика 3.1. Структура модификације генетског алгоритма *iGA*

Од нове модификације *iGA* очекује се рад као и код атрактивних хеуристичких метода оптимизације и рад са малим бројем параметара. Развијена модификација треба да буде ефикасна, лака за софтверску имплементацију и коначну практичну примену код инжењерских проблема.

3.1.2 Развој модификације методе *TLBO* (*rTLBO*)

Током истраживања, спровођено је и испитивање *TLBO* алгоритма, његовог рада и карактеристика. Вршена су испитивања и модификације овог алгоритма. Тек када се приступи процесу софтверске имплементације неке методе, а после тога и практичној употреби, тада се могу уочити проблеми, мане и недостаци неке методе. Основни проблем је веома тешко постизање теоријских претпоставки и рада методе какав је предвиђен, јер и најмања одступања у имплементацији доприносе драстичној промени добијених резултата. Доминантан проблем имплементације оптимизационих метода, па

тако и код *TLBO* методе, представља усклађивање ознака из различитих истраживања. Чак у појединим истраживањима постоје пропусти где структура алгоритма и једначине које описују алгоритам нису у потпуности усклађени и компатибилни. То отвара могућност за неадекватну софтверску имплементацију и другачију конвергенцију у односу на прописану. Други проблем код *TLBO* алгоритма чини експлицитно дефинисање случајних вредности које се дефинишу током рада алгоритма. Неопходно је експлицитно дефинисати да ли је случајно изабрано решење X_i исто за обе фазе алгоритма, јер само искуствено је могуће извести овај закључак. Поред тога проблем представљају и вредности r_i које се бирају из интервала од 0 до 1. Ове вредности могу бити или исте или различите по фазама алгоритма што драстично утиче на његов рад. Поставља се и питање да ли се ова вредност мења у сваком еволуционом процесу или остаје иста, пошто алгоритам еволуира у зависности од величине популације у једној итерацији.

Основна литература за тумачење *TLBO* алгоритма је истраживање творца овог алгоритма [33]. Сва остала литература ослања се на ово истраживање, док у њему постоје појмови који су контрадикторни. Терминолошки посматрано у структури алгоритма представљен је појам $Mean$, а касније у изразима је та величина означена као M_i, што ствара дилему да ли се ради о истој величини. Оваква ситуација понавља се на више места у различитим истраживањима везаним за овај алгоритам. Следећу дилему чини што аутор у структури алгоритма дефинише да се прерачунава средња вредност сваке креиране променљиве, а касније у делу имплементације *TLBO* алгоритма у истом истраживању, аутор тврди да треба израчунати средњу вредност сваке колоне, што представља сетове – средњу вредност појединачних решења. Ове ствари се очигледно драстично разликују и представљају технички пропуст. Није спорно да овакви пропусти постоје, само се они понављају у скоро свим новим истраживањима и дискутабилно је како су други аутори имплементирали овај алгоритам и који од ових приступа су искористили у својим истраживањима. Случајне вредности r_i су у скоро свим истраживањима представљене и означене на исти начин за све фазе алгоритма. Није јасно да ли је ова вредност увек иста за целу итерацију, за једну фазу алгоритма, за једно од решења (студента) или само за једног студента у једној фази алгоритма. Аутори добијају идентичне резултате, а врше различита извођења ове случајне вредности, што је наравно дискутабилно. Без обзира колико је алгоритам једнозначно дефинисан веома је тешко направити његову идентичну копију што наравно треба анализирати код рада модификација и хибрида сваког алгоритма, јер се односе на иницијалне методе.

Постоји још много проблема који се јављају у процесу софтверске имплементације алгоритма, а критике нису упућене на сам рад алгоритма који има изузетне перформансе, већ на техничке детаље у којима се види могућност унапређења рада и развоја хеуристичких метода оптимизације.

Према уоченом простору за измену методе развијена је модификација доминантне, модерне методе *TLBO*. Приступ је базиран на пропустима и недостацима методе који постоје. Извођење модификација у пракси врши се на неколико различитих начина. Метода се модификује уграђивањем неког познатог сегмента или фазе друге методе, променом неког од параметара методе, креирањем зависности параметара и конвергенције и на многе друге начине. Свака измена у односу на базичну методу може бити третирана као модификација методе. Што је метода дуже експлоатисана, то је и број модификација већи. Тако, на пример, генетски алгоритам има огроман број модификација, које су толико употребљаване, да се у данашње време третирају као начини извођења методе. Ово су заправо модификације базичне методе. Код новијих метода, као што је *TLBO* алгоритам, још увек нису модификације масовно примењиване, па се и третирају као модификације. Раздвајање модификације и хибридизације методе

заснива се на учешћу неке друге методе. Модификација може садржати неки сегмент друге методе, али ако учествује цела метода у раду онда је то неопходно третирати као хибридну методу. Разлог настанка модификација је првенствено побољшавање рада базичних метода. Методе могу бити побољшане на многобројне начине, попут брзине рада, лакоће употребе, тачности које могу да постигну, адаптивности одређеним проблемима итд. Сама модификација неке методе представља веома комплексан задатак, који изискује много искуства. Првенствено је неопходно пуно користити методу и применити је на конкретне оптимизационе проблеме и тако уочити да ли постоје недостаци и могућности измене да би метода радила боље. Није могуће само примењивати методу и извести ове закључке, већ је неопходно вршити сталну анализу њеног рада. Анализирањем рада методе могуће је уочити све карактеристичне сегменте рада методе и тек онда тражити алтернативе за унапређење методе. У пракси ни креирање начина за унапређењем методе не представља ништа лакши задатак од уочавања недостатака.
За потребе овог истраживања приликом вршења модификације методе *TLBO* алгоритма било је неопходно узети у обзир неколико ствари, и то да се:

- Уочи простор за модификацију, недостаци методе и делови који могу бити унапређени,
- Развије начин модификације на кључним местима,
- Постигну боље перформансе методе,
- Ради под свим околностима и у условима као и базична метода,
- Модификација направи на више кључних делова алгоритма.

Овај сегмент презентује циљ да се развије модификована метода алгоритма учења која ће имати боље перформансе и омогући решавање истих проблема под истим условима и околностима. Неопходно је пронаћи више од једног кључног сегмента алгоритма учења и за сваки пронађени сегмент развити алтернативну модификацију, која у комплетном раду алгоритма пружа боље перформансе. За потребе овог истраживања извршена је модификација само прве фазе *TLBO* алгоритма, а разлог за то је комплексност проблема модификације и студиозан приступ модификацији.

Опис модификације *rTLBO*

Тестирањем *TLBO* алгоритма могуће је закључити да је његова конвергенција изузетна, али приближавањем глобалној екстремној вредности конвергенција је спорија. Алгоритам *TLBO* ради тако да се ослања на измену вредности променљивих између осталог на основу случајне вредности r_i, као из једначине (3.3).

$$X_{New,i} = X_{Old,i} + r_i(X_{Teacher} - T_F X_{Mean}) \tag{3.3}$$

Када се вредности приближе оптимуму, смањује се могућност проналажења новог решења, јер нове вредности или нису боље од претходних или превазилазе постављени домен. Утицањем на величину случајне вредности повећава се вероватноћа проналажења адекватног решења које ће убрзати конвергенцију. Ово представља први сегмент унапређења и развоја модификације.

Највећи утицај на конвергенцију има сама једначина којом се постижу нове вредности по итерацијама. Ако се изврши корекција ове једначине могуће је постићи значајно боље резултате у оптимизацији. Корекција ове једначине, на основу тестирања алгоритма мора да узме у обзир однос средње вредности популације и тренутне вредности решења. Треба и у једначини узети у обзир брзину конвергенције при локализацији решења око оптимума. На основу базичне методе учења и на основу закључака могуће је дефинисати једначину (3.4) која представља модификацију базичне методе, а утиче на њен бољи рад.

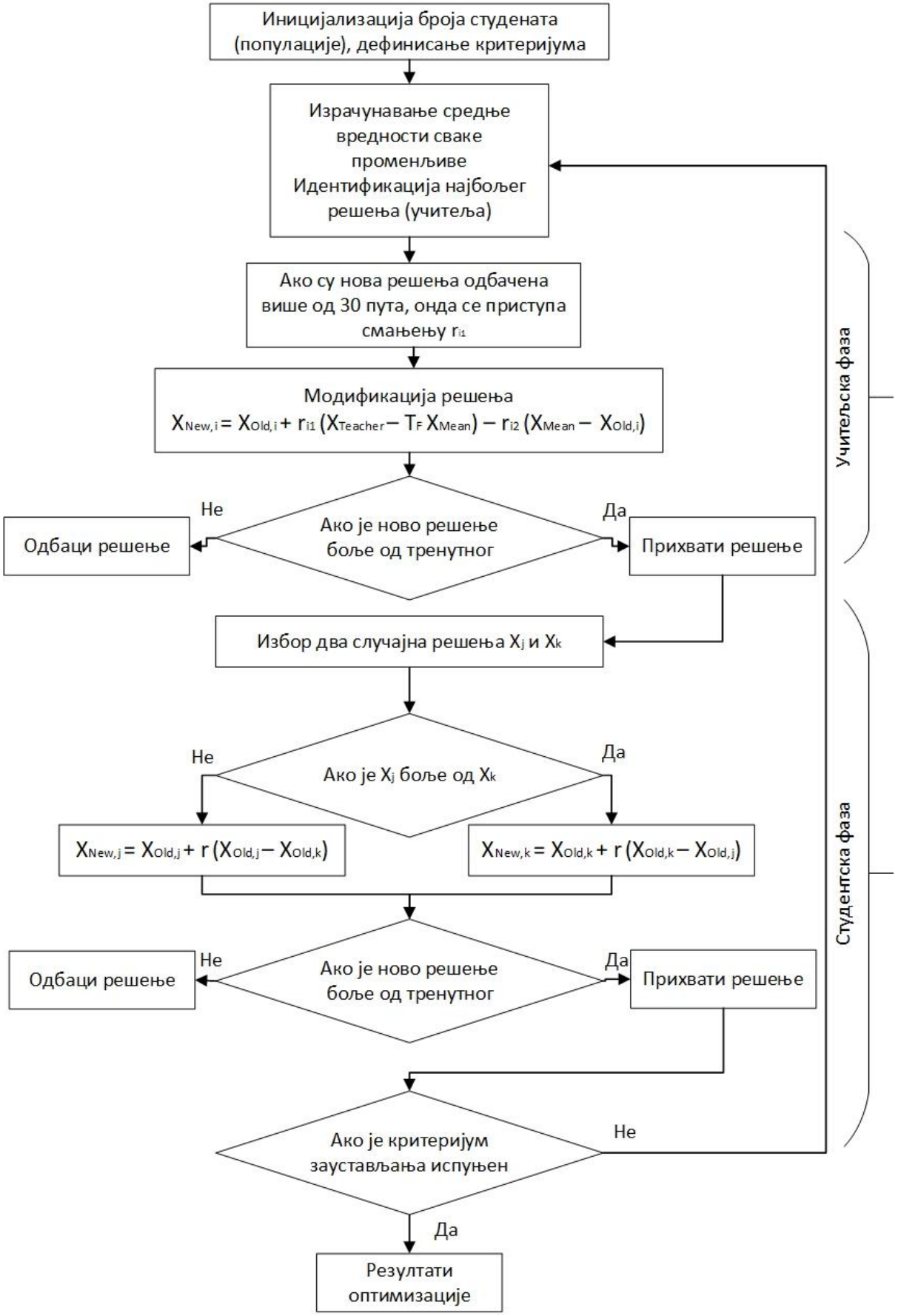
$$X_{New,i} = X_{Old,i} + r_{i1}(X_{Teacher} - T_F X_{Mean}) - r_{i2}\left(X_{Mean} - X_{Old,i}\right) \tag{3.4}$$

Овим се обезбеђује да је модификација извршена на више од једног кључног места. У изразу (3.4) r_{i2} представља случајну вредност, а сви остали елементи су познати из базичног алгоритма учења. Улазни подаци за корисника су такође величина популације и број генерације и сви остали услови идентични су као код базичне методе *TLBO* алгоритма. За потребе овог истраживања развијена је оригинална, нова модификација *rTLBO*. После извршене модификације алгоритам и његов рад представљени су на слици 3.2.

Оваквим алгоритмом се не постижу знатно бољи резултати у свеукупној конвергенцији, већ је поступак доминантан у најкритичнијим деловима рада методе, а то је када се вредности приближе оптимуму.

Први сегмент модификације постаје активан када алгоритам није у стању да пронађе решење које ће у учитељској фази бити прихватљиво. То значи ново решење у дозвољеном домену. У овој ситуацији када се то догађа 30 пута за редом, случајна вредност r_{i1} мора бити смањена за константу 10^{-3}, ако вредност остаје иста још 35 пута онда се смањење врши за константу 10^{-5}, за 38 смањење је 10^{-7}, и коначно ако се вредности не промене 40 пута онда је смањење за 10^{-9}. Овај ефекат доводи до превазилажења делова успорене конвергенције, а нарочито у делу где се вредност приближи оптималној.
Други сегмент модификације ради тако што је измењена операција која чини учитељску фазу. Додатак представља покушај да се вредност одаљава од лоших решења, док истовремено тежи повећању квалитета и приближавање целе популације најбољем решењу (учитељу).

Развијена модификација названа је као *rTLBO* јер је оријентисана на рад са случајним вредностима r_i и представља алтернативу приступа модификовања хеуристичких метода.



Слика 3.2. Модификовани *TLBO* алгоритам (*rTLBO*)

3.2 Развој хибрида хеуристичких метода оптимизације

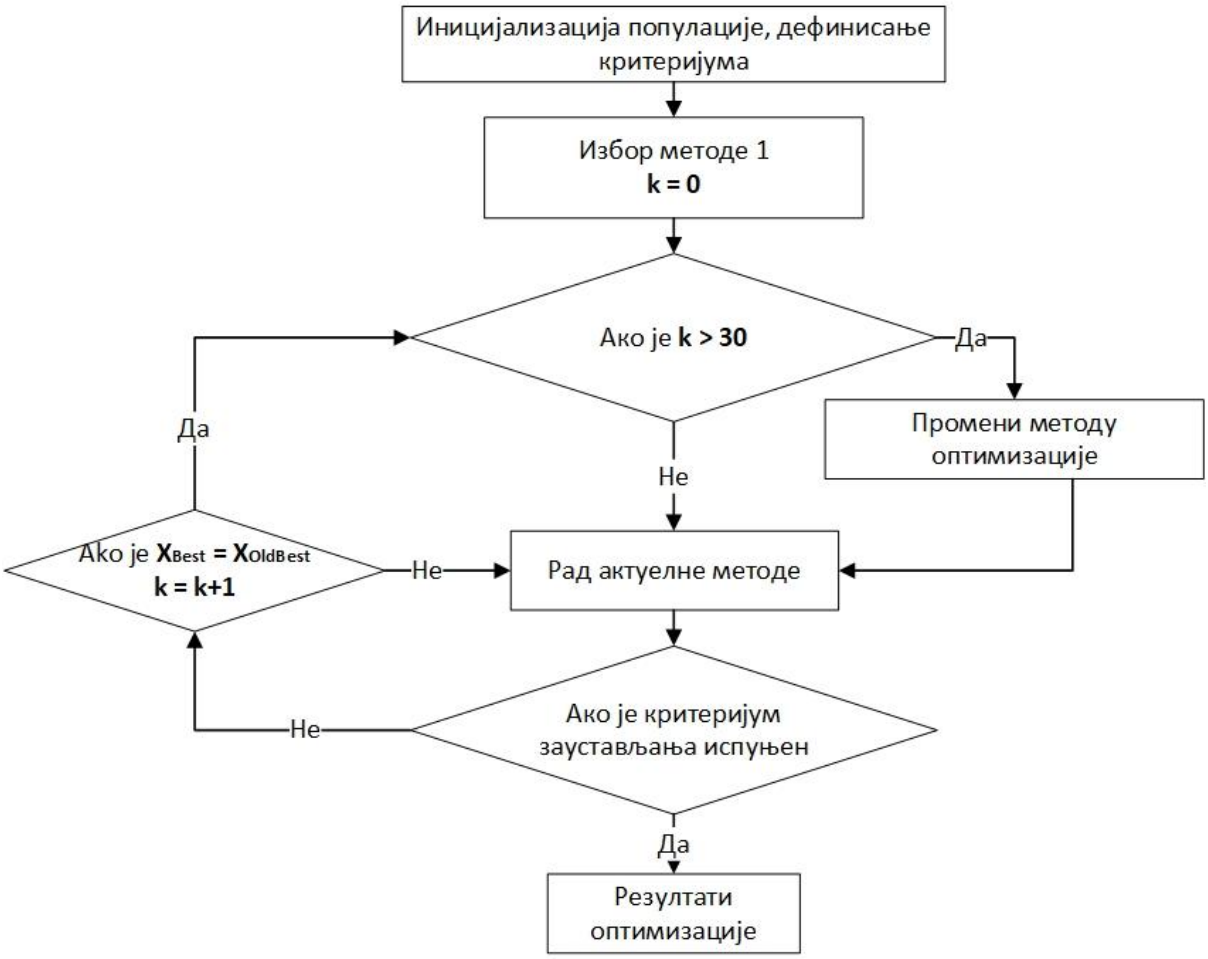
Ова докторска дисертација предвиђа модификацију, хибридизацију и развој потпуно нове, оригиналне хеуристичке методе. Пошто је потврђено да *GA* и *TLBO* алгоритми представљају доминантне методе у односу на остале атрактивне хеуристичке методе чије модификације су и извршене, потребно је представити даљи напредак у области развоја хеуристичких метода.
Пошто хибридизација подразумева комбиновање више од једне методе, на неки начин и у овом истраживању је то случај. У овом истраживању конципиране су и развијене оригиналне хибридне методе. Основна потреба примене већег броја метода је повећање ефикасности и отклањање парцијалних недостатака које методе појединачно имају. Комбиновање метода врши се на многобројне начине, преплитањем целина различитих алгоритама или радом једне методе, па прелажењем на другу, радом једне методе, па локалном претрагом друге, као и на многе друге начине. Пошто ово истраживање подразумева хибридизацију хеуристичких метода, за ове потребе приликом развоја хибридне методе важно је само неколико ствари, где морају да буду испуњени следећи услови:

- Учествовање већег броја хеуристичких метода (*iGA, PSO, TLBO*),
- Методе које учествују морају бити хеуристичке методе оптимизације,
- Да се недостаци метода не решавају директно већ потпуно новим приступом,
- Ради као јединствена хеуристичка метода,
- Постоји могућност праћења активности,
- Хибрид има сличне карактеристике као и парцијалне хеуристичке методе.

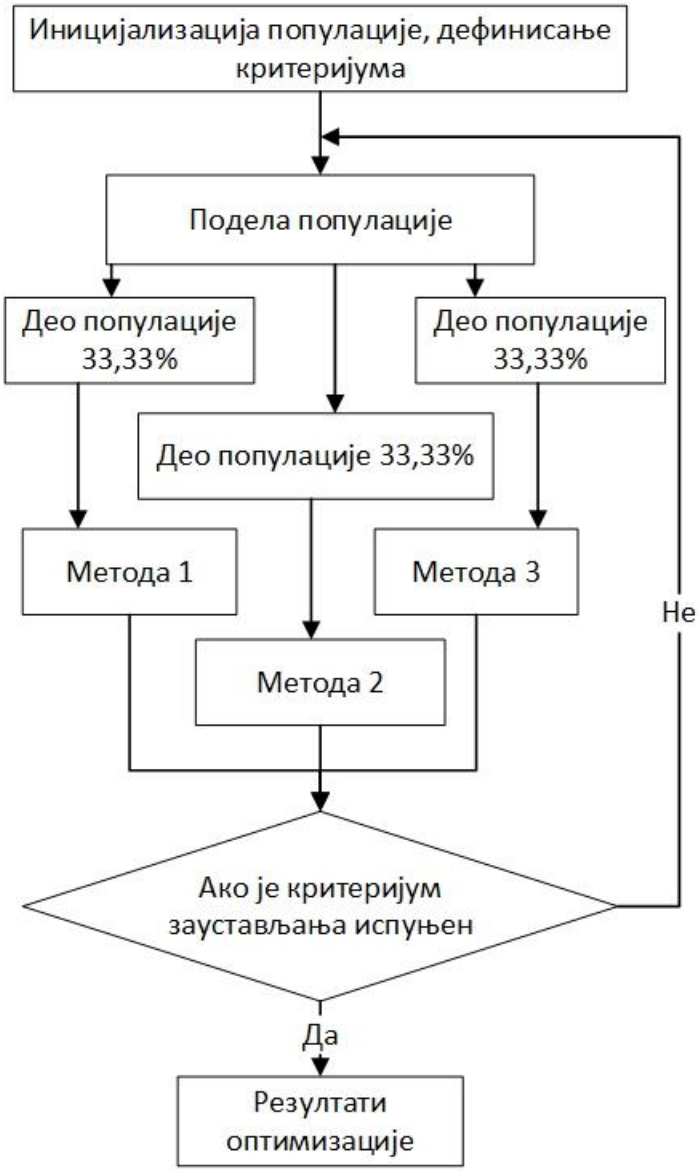
Циљ је развити хибридну хеуристичку методу која на било који начин има повољније карактеристике од рада било које методе која учествује у овом процесу. Хибридне хеуристичке методе представљају неки вид комбинације хеуристичких метода, али под обавезним условом бољег рада, јер у супротном њен развој није оправдан. Уочено је да свака хеуристичка метода оптимизације има својих предности и недостатака и то је разлог хибридизације три хеуристичке методе, *iGA, PSO* и *TLBO*. Основна идеја јесте да хибридна метода ради боље од појединачних, али за ово истраживање то није и обавеза. Мотивација је заснована на развоју приступа и принципа који могу пружити велику перспективу и допринос у даљем раду са хеуристичким методама и њиховом развоју. Дакле, потребно је представити идеју која ће обезбедити будући правац и област рада са хеуристичким методама оптимизације.
Концепт ове хибридне методе настао је као резултат хеуристичког приступа развоју хеуристичких метода. Углавном, у току развоја ових метода, оријентација је на уочавању недостатака неке методе (уочавање мана методе) и коришћење неке друге методе да се ти недостаци превазиђу. Овде је направљен покушај да се потисну недостаци метода и да се истакну само њени квалитети. У том циљу развијена су два потенцијална приступа, редни и паралелни. Редни приступ подразумева смењивање хеуристичке методе другом, када актуелна метода покаже своју слабост, што је илустровано на слици 3.3 а). Овај приступ је уочен као интересантнији, атрактивнији и ефикаснији, па је због тога поред концепције и детаљно разрађен и анализиран. Други, паралелни приступ, заснован је на паралелном раду већег броја хеуристичких метода над делом популације у циљу побољшања перформанси. Овај приступ представљен је на слици 3.3 б), а основни

принцип рада заснован је на дељењу целокупне популације методама, чиме се постиже флексибилност на различите групе проблема. У општем случају овај приступ можда не би дао значајне резултате по питању брзине конвергенције, али спектар проблема који би овакав хибрид ефикасно решавао је много шири у односу на појединачне методе. Основни разлог зашто паралелни приступ није анализиран је велики број прерачунавања функције циља у односу на редни приступ.

Акценат хибридизације усмерен је на редни приступ. То је могуће реализовати тако што у тренутку настанка проблема у раду алгоритма, он мења свој принцип рада. Промена подразумева потпуни прелазак са једне на другу методу. Проблем настаје када се конвергенција успори или заустави. Тада је вредност најбољег решења у популацији непромењена кроз итерације. То указује на слабост методе која ради, а хибрид је замишљен тако да у овом тренутку, са актуелним вредностима популације, почне да користи неку другу методу. Поступак се понавља у зависности од тога колико метода је имплементирано у хибрид и од конкретног проблема оптимизације. Концепт развоја хибридне хеуристичке методе са редним приступом представљен је на слици 3.3 а).



а)

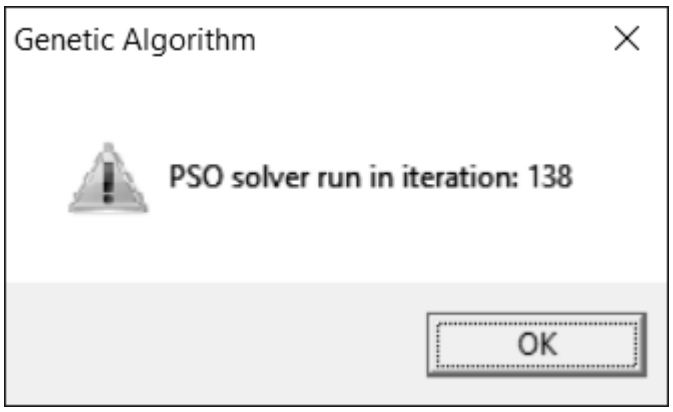


б)

Слика 3.3. Илустрација развоја хибридне методе; а) редни приступ; б) паралелни приступ

Алгоритам представљен на слици 3.3 а) приказује принцип рада новог хибридног алгоритма. Алгоритам почиње са радом користећи једну хеуристичку методу, која због својих карактеристика у неком тренутку успори конвергенцију. Екстремна вредност по итерацијама остаје непромењена, а најчешћи разлог за то је конвергенција око локалних екстремних вредности. Класичан приступ хибридизације подразумева уочавање овог недостатка и убацивање дела који ће превентивно деловати у овим ситуацијама. У овом случају, када се уочи недостатак у раду неке методе прекида се њен рад. Тада са радом почиње друга метода, где је за ту методу иницијална популација иста као и популација у последњој итерацији претходног алгоритма. Од нове методе која је започела са радом очекује се да ће превазићи локалну вредност због потпуно другачијег приступа у односу на претходну методу. Када и друга метода стане са конвергенцијом, прелази се на трећу методу, која има већ формирану иницијалну популацију. За потребе овог хибрида коришћене су три методе оптимизације, али је у општем случају могуће применити две

и више метода у зависности од потреба. Када и трећа метода покаже своје недостатке, алгоритам се враћа на прву методу и тако у круг све до испуњења критеријума заустављања. Пошто је критеријум заустављања број генерација, алгоритам ће радити све до испуњења укупног броја генерација.
Потребно је знати када алгоритам нема адекватну конвергенцију. То је тренутак када из итерације у итерацију не постиже бољу вредност. С тим у вези дефинисан је и прелазак са једног на други алгоритам. Када се за одређени број итерација екстремна вредност не мења, врши се прелазак на другу методу. Пошто ова величина представља целобројну квантитативну вредност, неопходно је извршити њено дефинисање. У структури алгоритма овај бројач означен је са k, на слици 3.3 а). За потребе овог истраживања утврђено је да ако до промене не дође у 30 узастопних итерација треба реаговати, у складу са чим је извршена реализација алгоритма. Ова вредност може бити и параметар хибридног алгоритма за његову даљу употребу, али за потребе истраживања ова вредност је константна. Ова вредност сигурно има утицаја на квалитет резултата, али пошто постоји тенденција смањења броја параметара то је прихваћено и у овом случају и изабрана је искуствено.
За ову хибридну методу коришћен је прво модификовани генетски алгоритам (*iGA*), као друга метода *PSO* и као трећа метода стандардни *TLBO* алгоритам. За каснију употребу могуће је понудити кориснику и ово подешавање, али код комплексних проблема, где се изврши промена свих поменутих метода током прорачуна, нема пресудног утицаја којом методом алгоритам започиње свој рад.
Предност оваквог приступа је у томе што се проблем решава на потпуно нови начин, а не отклањањем недостатака методе. Пошто у раду развијеног хибрида учествују *iGA*, *PSO* и *TLBO* очигледно је да је испуњен захтев да учествује већи број хеуристичких метода. Поред тога, недостаци методе се не решавају директно већ потпуно новим приступом. Ова хибридна метода је јединствена и за корисника ради као потпуно независна, тј. као и свака друга хеуристичка метода. Касније, при развоју оригиналног софтвера, према захтевима, било је потребно омогућити кориснику да зна када се мења метода и која метода ради у датом тренутку. То је постигнуто тако што када до промене дође појави се прозор (слика 3.4) на коме је дефинисан број итерације у којој се врши промена и метода која постаје активна.



Слика 3.4. Пример обавештења из оригиналног софтвера о промени методе код хибрида *hGPT*

Ово је најбољи начин праћења рада хибридног алгоритма. Извршено је и тестирање овог хибрида, јер развијена метода треба имати боље карактеристике у односу на појединачни рад метода учесница.

Нова хибридна метода названа је као *hGPT*, где ознака говори да се ради о хибриду коришћених метода хеуристичке оптимизације.
Слабост хибридне методе заснована је на појединачним слабостима метода које учествују у хибридизацији. Ове парцијалне слабости одражавају се на цео хибрид и смањују његову ефикасност. Хибрид развијен на представљени начин има основну идеју да ако се слабост и појави, постоји начин да се она превазиђе, а једини нежељени ефекат може бити смањење ефикасности хибрида кроз повећање потребног броја укупних итерација за решавање одређеног проблема. Без обзира на поједине слабости, овај приступ је веома перспективан због великог броја варијанти извођења и хибридизације било које хеуристичке методе оптимизације. Велики број алтернатива указује на чињеницу да ће сигурно овим приступом бити могуће пронаћи хибрид који одговара захтевима и потребама практичне оптимизације.

3.3 Развој нове хеуристичке методе оптимизације

Саставни део ове дисертације представља и развој потпуно нове хеуристичке методе оптимизације. То представља веома комплексан процес без гаранције о успешном раду и перспективе постизања бољег. Пресудно за процес развоја потпуно нових хеуристичких метода је искуство и познавање постигнутих резултата у литератури, као и приступ других аутора овом проблему. Веома је значајно одступити од конвенционалног начина размишљања, пронаћи инспирацију у свакодневним појавама и процесима, уочити њихове кључне целине и нумерички представити њихов рад. То значи да баналне свакодневне ствари, на које човек не обраћа пажњу јер су природне, логичне и спонтане, представљају поступке који се могу применити у оптимизационом процесу. Овакав приступ је сам по себи хеуристички, а метода која репрезентује неку овакву појаву је такође хеуристичка по аутоматизму.
У досадашњим истраживањима могуће је уочити потпуно различите приступе, а тенденције указују на креирање стабилне оптимизационе методе повољних карактеристика. Када се говори о повољним карактеристикама, мисли се првенствено на брзину и квалитет конвергенције. Брзина конвергенције не представља само време, већ и ограничен број евалуација функције циља за одређени проблем. Развојем сваке методе кроз историју се види напредак у овом сегменту. Квалитет конвергенције односи се на превазилажење локалних екстремних вредности, што је веома тежак задатак. Поред тога значајно је и постизање стварног оптимума при локализацији вредности. Хеуристичке методе по дефиницији никада не постижу апсолутно тачно решење, већ у тој мери конвергирају ка стварном оптимуму да се решење може третирати као апсолутно тачно. То значи да је грешка толико мала да се може у потпуности занемарити. Ово није недостатак метода, већ предност, јер на првом месту дају решење, а толико су савршене да је грешку тешко и измерити у неким случајевима, па чак и код комплексних проблема оптимизације. Локализација вредности подразумева да када се решење приближи оптималном, конвергенција се успорава и све теже се постиже боље решење. Ово је још један задатак који метода мора да реши.
Акценат у модерној хеуристичкој оптимизацији представља смањење или елиминацију утицаја људске грешке у процесу оптимизације. За разлику од почетног развоја хеуристичких метода кроз историју, данас се развијају методе без параметара (*parameter less* или *parameter free*). То значи да корисник методе поред повезивања проблема са методом има само задатак да дефинише број евалуације функције циља, помоћу броја итерације и величине популације.

За развој хеуристичке методе оптимизације потребно је имати у виду и ток процеса оптимизације применом хеуристичких метода. Глобално, по узору на нека истраживања [12] могуће је дефинисати општи поступак за нумеричко проналажење решења:

- Издвајање иницијалних вредности независних променљивих из скупа доступних вредности које задовољавају сва ограничења,
- Евалуација функције циља за сваку итерацију и скуп променљивих,
- Промена иницијалних вредности хеуристичком методом,
- Поновна евалуација функције циља,
- Упоређивање претходне и постигнуте вредности функције циља,
- Усвајање адекватних вредности за поступак даље оптимизације или усвајање коначног оптимума.

Претходно наведене ставке представљају општу слику коју је потребно непрестано имати у виду при креирању нове хеуристичке методе, како би решење конвергирало.
Када су испуњени наведени критеријуми знања и искуства, следећи корак био је уједно и најтежи. Уочити појаву или процес који је могуће имплементирати и математички и нумерички формализовати. Када се посматрају идеје других аутора то су очигледни, потпуно једноставни процеси и појаве, али када је потребно наћи нову појаву, тек онда је могуће схватити да то није ни очигледно ни једноставно. Иницијалне идеје биле су у анализи многих природних појава, устаљених поступака које људи обављају, психолошким процесима доношења одлука, процесима учења, итд. Сви ови процеси су разматрани и стварно представљају потенцијално нове хеуристичке методе оптимизације. Коначна инспирација за развој нове хеуристичке методе дошла је кроз поступке, процесе и понашања која имају деца. Деца се у потпуности хеуристички понашају и размишљају и немају научене поступке и процесе који их ограничавају у томе, за разлику од старијих људи. И поред апсолутно хеуристичког понашања деце веома је тешко дефинисати шта је то хеуристичка појава код њих. Тек када се то експлицитно утврди могуће је приступити формулацији ове појаве.
Хеуристика по дефиницији представља учење и откривање истине базирано на искуству, а не примена искуства. Ову особину поседују само деца. Друга дефиниција хеуристике подразумева кршење уопштених правила размишљања, што опет могу само деца, јер нису научена да размишљају на тачно одређени начин и немају довољно знања да би их размишљање одвукло у већ познатом правцу. Размишљање детета је потпуно неконвенционално и индивидуално, јер свако дете има сопствени приступ. Утицај околине није занемарљив, али у раном периоду детињства није ни довољан да би дете одступило од хеуристичког приступа.
Све ово не би било могуће уочити да није било деце у окружењу током реализације овог истраживања. Заслужан за проналажење, стварање и развој нове хеуристичке методе је дечак Димитрије. Почетком 2014. године родила се идеја о раду и сарадњи са Димитријем у циљу развоја и уочавања интересантних поступака и појава у његовом понашању. На самом почетку рада Димитрије је имао 1,5. годину, па све до реализације алгоритма када је напунио 4 године. Уз његову сагласност и сагласност његових родитеља започет је рад на откривању ових процеса и појава. Димитрије је дечак који одраста у лепом и безбрижном окружењу, срећан и интелигентан и веома креативан. Поред тога васпитан дечак са зрелим особинама, разуман и вредан. Ове особине су веома важне, јер да Димитрије није такав вероватно никада не би била ни уочена хеуристичка појава у његовом понашању. Како је сам себе назвао Динди, још када је проговорио и овај алгоритам је добио име по њему, *DINDI* алгоритам.

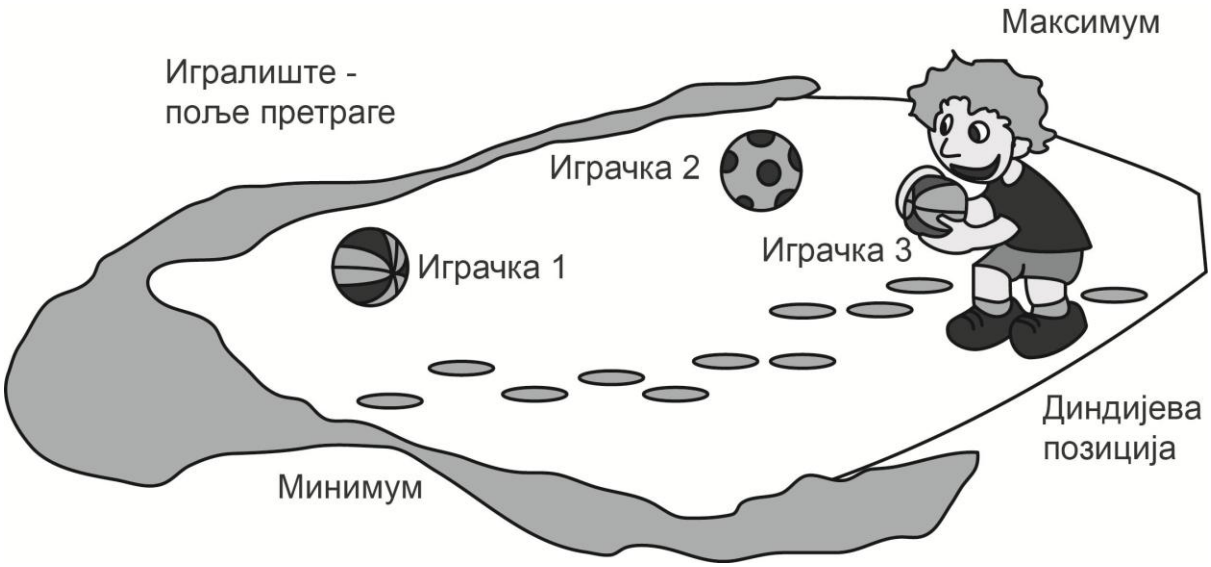
За развој нове методе суштински је важно одступити од конвенционалног приступа, наћи и уочити неки успешан процес или појаву у природи и адекватно је математички формулисати. То подразумева да свакодневни устаљени процеси и појаве, који су спонтани и на које човек не обраћа пажњу представљају потенцијалну алтернативу оптимизације.

3.3.1 Оптимизациона метода *DINDI*

Истраживање је вођено без било каквог наметања одређеног понашања и ставова, већ само праћењем његове игре и учествовањем у његовој игри. Било какво наметнуто испитивање не би било коректно према њему, а само истраживање би одвело у жељеном (наметнутом), а не хеуристичком правцу.
Вршена је анализа његовог кретања, комуникације, играња и слично, и после вишегодишњег рада уочене су неке карактеристичне појаве. Истраживање је укупно трајало 2,5. године, а иза тога налази се велики број неуспешних покушаја.
У овом процесу било је неопходно одговорити на нека кључна питања, као што су да ли Динди:

- Негде примећује екстремне вредности,
- Може да их пронађе,
- Понавља процес проналажења екстремних вредности,
- Разликује глобалне и локалне екстремне вредности,
- Има више фаза проналажења екстремних вредности,
- Спонтано долази до решења у разумном временском интервалу,
- Увек код истих ствари реагује на исти начин,
- Може да нађе међурешење (конвергенција у датом броју итерација),
- Прави разлику између минимума и максимума,
- Може да погреши,
- Итд.

На ова питања спонтано се намећу одговори. Уочено је да Динди у свему што ради одређује границе. Стално испитује колика је зона у којој сме да се игра, колико сме и може да поједе, колико треба да спава и то је прва карактеристика која репрезентује његова устаљена понашања. Друга карактеристика је његова позиција у односу на границе. Где се он налази у односу на границе игралишта, колико је већ појео, да ли је спавао у току дана, итд. Поступак хеуристичке појаве илустрован је на слици 3.5.



Слика 3.5. Концепт развоја хеуристичке појаве – Динди током игре

Слика 3.5 илуструје приступ за развој хеуристичке појаве. *DINDI* алгоритам чине укупно три фазе. Прва фаза (Игралиште) представља Диндијеву позицију на игралишту и границе зоне у којој се он игра. Друга фаза (Игра) представља играчке које су му на располагању и његов избор забаве. Трећа фаза (Засићење) представља тренутак када му досаде неке игре и играчке. Структура алгоритма представљена је на слици 3.6, где је могуће уочити све три кључне фазе.

Фаза игралиште

Прва фаза названа игралиште показује које величине је игралиште на коме се Динди налази и представља зону у којој се Динди игра. Ова фаза алгоритма садржи две целине, где прва целина представља простор, као разлику (вектора) крајње и почетне тачке $X_{max} - X_{min}$, где су X_{max} и X_{min} вредности крајњих граница зоне за игру. Друга целина односи се на Диндијеву позицију на игралишту у односу на границе као $X_{Dindi} - X_{min}$, где је X_{Dindi} његова позиција. Израз који презентује рад ове фазе алгоритма има следећи облик:

$$X_{Dindi,New} = X_{Dindi,Old} + r_r(X_{max} - X_{min}) - r_D\left(X_{Dindi,Old} - X_{min}\right) \qquad (3.5)$$

Игралиште на коме се Динди игра представља допуштено поље претраге, а границе игралишта су екстремне вредности у одређеном тренутку. Зона у којој се игра се стално мења у дозвољеним границама игралишта. Промена зоне врши се помоћу случајне вредности r_r, где ова вредности представља Диндијев покушај да испита границе игралишта, а да остане у дозвољеној зони. Поред тога Динди мења сопствену позицију у односу на границе случајном вредношћу r_D. Обе случајне вредности могу бити само у интервалу од 0 до 1. Сваким Динди кретањем потребно је поново преиспитати и одредити ове вредности, што значи да се ове случајне вредности у свакој итерацији мењају и за сваки скуп променљивих. Ове случајне вредности имају задатак да се претрага врши по комплетном дозвољеном пољу претраге. Код *DINDI* алгоритма нова

вредност је прихватљива само уколико је боља од претходне што је и уочљиво на слици 3.6, где је представљена структура овог алгоритма.
Ова фаза је активна током целокупног рада алгоритма у свим итерацијама. По завршетку ове фазе алгоритам наставља са радом преласком на другу фазу, што се понавља у свакој итерацији.

Фаза игра

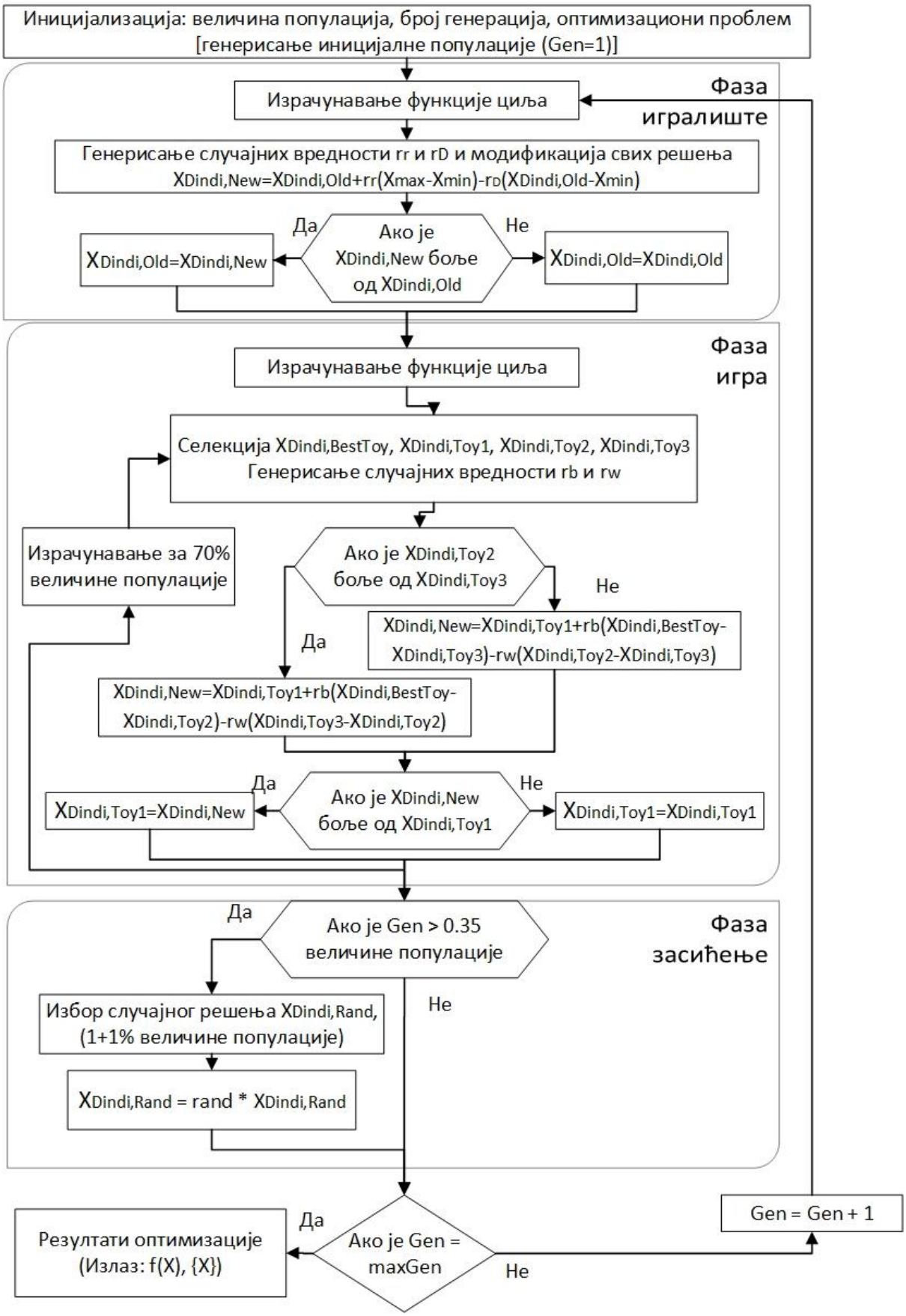
Ради функционалности *DINDI* алгоритма, неопходна је имплементација свих предвиђених фаза алгоритама. Ако нека од фаза буде изузета алгоритам ће радити, али његова ефикасност ће бити знатно смањена. Друга фаза алгоритма названа је игра. Ова фаза се односи на играчке које су му на располагању и игре које су њему интересантне. Динди увек када дође на игралиште почиње са игром, али се не игра са свим расположивим играчкама, већ неке у потпуности запоставља. Динди се активно игра са око 2/3 својих играчака, што је уочљиво на слици 3.6, где је представљена структура алгоритма. Ако се Динди у неком тренутку игра са играчком $X_{Dindi,Toy1}$, он покушава да пронађе нову играчку која тежи најбољој помоћу $\left(X_{Dindi,BestToy} - X_{Dindi,Toy2}\right)$ и пореди је са неком другом играчком која му се мање допада и од које покушава да се удаљи $\left(X_{Dindi,Toy3} - X_{Dindi,Toy2}\right)$. Ова фаза зависи од тога да ли му се више допада играчка 2 или играчка 3. Ако је $X_{Dindi,Toy2} > X_{Dindi,Toy3}$, онда је:

$$X_{Dindi,Toy1New} = X_{Dindi,Toy1} + r_b\left(X_{Dindi,BestToy} - X_{Dindi,Toy2}\right) - r_w\left(X_{Dindi,Toy3} - X_{Dindi,Toy2}\right) \quad (3.6)$$

У супротном, ако је Диндију интересантнија играчка 3 од играчке 2, $X_{Dindi,Toy3} > X_{Dindi,Toy2}$, онда је:

$$X_{Dindi,Toy1New} = X_{Dindi,Toy1} + r_b\left(X_{Dindi,BestToy} - X_{Dindi,Toy3}\right) - r_w\left(X_{Dindi,Toy2} - X_{Dindi,Toy3}\right) \quad (3.7)$$

Случајне вредности које имају задатак претраге по целокупном дозвољеном домену (r_b и r_w) могу имати било коју вредност из домена (0, 1). Динди овом фазом покушава, у односу на тренутну позицију, да се приближи играчки која му се више допада, а да се истовремено одаљи од играчке која му није интересантна. Ако је успешан у том процесу онда он мења тренутно стање, а у супротном оставља све како је и било. То значи да ће нове вредности бити усвојене само ако су боље од претходних. Овај поступак се понавља на 70% величине популације, где $X_{Dindi,Toy1}$, $X_{Dindi,Toy2}$ и $X_{Dindi,Toy3}$ представљају случајно одабрана решења (скуп променљивих), а $X_{Dindi,BestToy}$ представља најбоље решење за дату итерацију.



Слика 3.6. Структура *DINDI* алгоритма

Фаза засићење

Трећа фаза *DINDI* алгоритма представља и описује ситуацију када Диндију досади играчка којом се игра и жели у потпуности да промени играчку неком новом, без обзира на њен квалитет. Трећа фаза постаје активна мало после почетка његове активне игре. Прву трећину времена (око 35%) Динди се без икаквих примедби игра са расположивим играчкама, а касније му постају досадне. Рад ове фазе представљен је на структури алгоритма (слика 3.6). Динди временом има потребу да у потпуности промени неку играчку новом, а ова фаза реализује се случајним избором неке играчке $X_{Dindi,Rand}$. Ова вредност се на случајан начин модификује и добија се потпуно нова вредност $X_{Dindi,RandNew}$, која мора бити у дозвољеном домену. У овој фази изабрана случајна вредност мора бити различита од најбоље вредности, јер Динди никада не би заменио најбољу играчку, што је могуће третирати као елитизам, а ову фазу као врсту мутације. Овај процес служи за превенцију постизања локалних екстремних вредности, а одвија се на $(1 + 0.01 \times Population\ Size)$.

Дакле, алгоритам функционише тако што се на случајан начин креира иницијална популација. Претходно дефинисане вредности су величина популације и број генерација. Скуп променљивих које учествују у математичком моделу представљају решење, а иницијалних решења у дозвољеном домену биће онолико колика је величина популације. Сваки овај скуп променљивих мора проћи кроз алгоритам и ако се постигне боље решење да се промени. Када сваки скуп променљивих прође кроз ове фазе завршава се итерација, наравно са потребним прерачунавањем вредности функције циља и провере допуштених граница. Минимална и максимална вредност представљају вредности скупа променљивих које дају најповољнију функцију циља за сваку итерацију. Поступак се понавља све до испуњења критеријума заустављања, што је најчешће број генерација.

Развојем овог алгоритма формализована је једна веома интересантна појава, а квалитет рада алгоритма зависи највише од реалности репрезентације појаве и квалитета имплементације. У даљем раду утврђена је и компетентност хеуристичке методе да парира квалитетним и модерним хеуристичким методама. Суштински, најважнија ствар је да се самим хеуристичким приступом може развијати хеуристичка оптимизација, а колико ће процес трајати и да ли ће се релевантни закључци уопште постићи не постоји никаква гаранција.

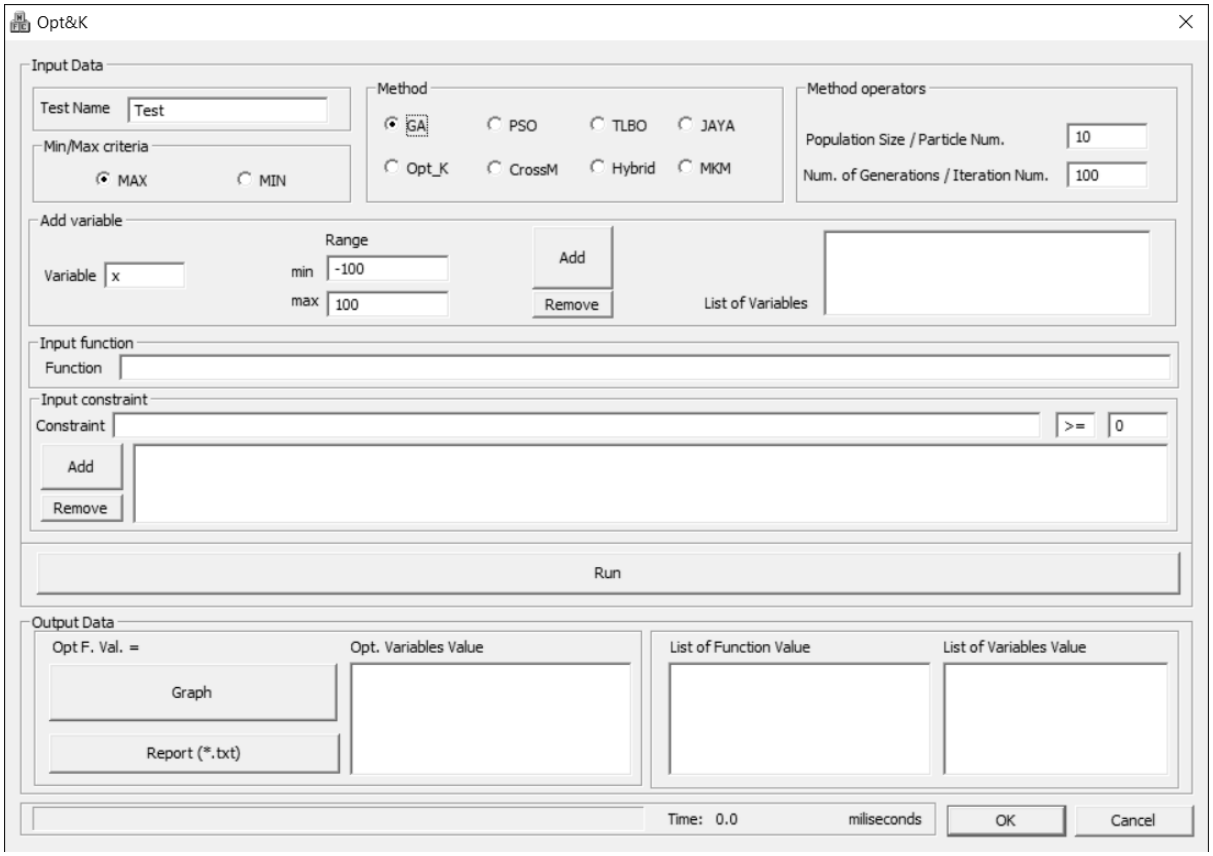
4. Развој оригиналног софтвера за оптимизацију

Свака хеуристичка метода оптимизације захтева софтверску имплементацију, како би ефикасност методе имала смисла. Хеуристичке методе оптимизације су искључиво нумеричке. Процес оптимизације без софтвера био би толико дуг и напоран да сама добит чак и за најједноставније проблеме никада не би била већа од уложених напора. У том случају би највећи напор представљао процес оптимизације. Зато се методе хеуристичке оптимизације имплементирају у софтвер, а софтвер се примењује за примену оптимизационих проблема. Корисник наравно мора да познаје позадину рада методе, како би адаптирао проблем софтверу или софтвер проблему. Генерално, постоје три кључна приступа при развоју софтвера за оптимизацију. Први приступ представља развој једнозначног софтвера за тачно одређени проблем. У овом случају оптимизација се врши увек на истом моделу, само се разликују поједини утицајни фактори и опсег променљивих оптимизације. Други приступ је развој оптимизационог софтвера за групу оптимизационих проблема. Код оваквих софтвера такође су уграђени математички модели у софтвер, само је спектар обухваћених проблема већи. Трећи приступ је развој општег оптимизационог софтвера за решавање „свих“ оптимизационих проблема. Овакви софтвери су универзални и могу се применити на све проблеме оптимизације. Наравно да постоје и недостаци овог приступа иначе би било бесмислено постојање другог приступа. Овакви софтвери захтевају директан унос математичког модела, што имплицира велико знање корисника. Поред тога, сва подешавања су општа, па корисник мора много да зна о процесу саме оптимизације и о раду метода које користи. Сам развој софтвера за опште проблеме је много компликованији и комплекснији. Што је једнозначније развијан софтвер то је лакши за употребу. Општи софтвери највећи значај имају у истраживачке сврхе и у случају рада са широким спектром оптимизационих проблема.

За све три групе поменутих приступа постоји могућност употребе већ развијених софтвера и самостални развој оригиналног софтвера. У општем случају употреба развијених софтвера највише има смисла јер самостални развој оваквих софтвера представља комплексан и гломазан задатак и захтева велику стручност. У пракси се најчешће овакви софтвери развијају у великим тимовима. За рад и истраживања са методама хеуристичке оптимизације потпуна општост постиже се тек развојем самосталног софтвера, што је случај и са овим истраживањем. Развијен је оригинални софтвер за хеуристичку оптимизацију, који омогућава приступ апсолутно свим фазама процеса оптимизације, што не би било могуће постићи употребом било ког другог софтвера. Други разлог је развој оригиналних поступака и метода, које не би било могуће применити без оваквог приступа. Једино овакав приступ може да понуди стварни напредак у процесу хеуристичке оптимизације и њеној практичној примени. Касније је једноставно смањити општост софтвера и адаптирати га и конкретизовати за одређене проблеме.

За развој софтвера коришћен је програмски језик *C++*. Избор програмског језика заснован је на његовим перформансама, јер апликације развијане у *C++* програмском језику раде веома брзо у поређењу са другим. Са друге стране овај програмски језик нуди општост у односу на друге модерне програмске језике, а од тога у великој мери (зависно од програмера) зависи и квалитет софтвера. Сам избор програмског језика не

утиче значајно на процес оптимизације, пошто данас постоји велика хардверска подршка и није неопходно обраћати пажњу на све детаље заузимања меморијског простора, брзине процесора и слично. Развојем оригиналног софтвера оставља се могућност сталне надоградње и развоја нових процеса и метода оптимизације. Ово је још један разлог развоја оригиналног софтвера. Поменуто је да је овакав приступ веома гломазан и захтеван и зато је можда и чудна одлука развоја оригиналног софтвера, али не би били постигнути планирани ефекти без оваквог приступа. Софтвер је развијен у окружењу *Microsoft Visual Studio 2008*, а корисничка апликација (*Dialog Based Application*) развијена је као *Microsoft Foundation Classes Application*, а изглед апликације представљен је на слици 4.1.



Слика 4.1. Приказ радног окружења развијене апликације

За истраживачке потребе и конзолна апликација би задовољила све потребе, али оваквом апликацијом је убрзан и олакшан процес испитивања и употребе софтвера. Највише напора и пажње је уложено у сам рад метода и алгоритама, јер је овај сегмент кључан у овом истраживању. Дакле, комуникација са корисником није најважнија ствар, али је ипак узета у обзир при развоју софтвера. Овакав софтвер може се надограђивати и адаптирати у складу са жељеним тенденцијама и актуелностима употреба апликација и то је стални процес. Развој софтвера извршен је прављењем појединачних делова софтвера и спајањем у функционалну целину. Софтвер садржи, као што је раније објашњено, три основне целине:

- Предпроцесор,
- Процесор,
- Постпроцесор.

Функционална целина мора да има адекватан улаз и обраду величина, даљу калкулацију и њихово управљање, представљање и визуелизацију постигнутих резултата.
Да би софтвер био адекватно реализован неопходно је јасно дефинисати приоритете рада софтвера и реализацију вршити у складу са њима. За приоритете су неопходне поставке циљева у овом процесу, јер ова корисничка апликација треба да:

- Врши оптимизацију за „све“ оптимизационе проблеме,
- Омогући имплементацију различитих метода оптимизације,
- Функционише са неограниченим бројем променљивих, како би примена била могућа на све проблеме независно од броја променљивих и њиховог назива,
- Ради за све математички формулисане функције циља,
- Омогућава све врсте математички формулисаних ограничења,
- Омогућава управљање алгоритмом помоћу величине популације и броја генерација (величине евалуације функције),
- Омогућава итерациони преглед резултата ради праћења оптимизационог процеса,
- Омогућава визуелизацију конвергенције.

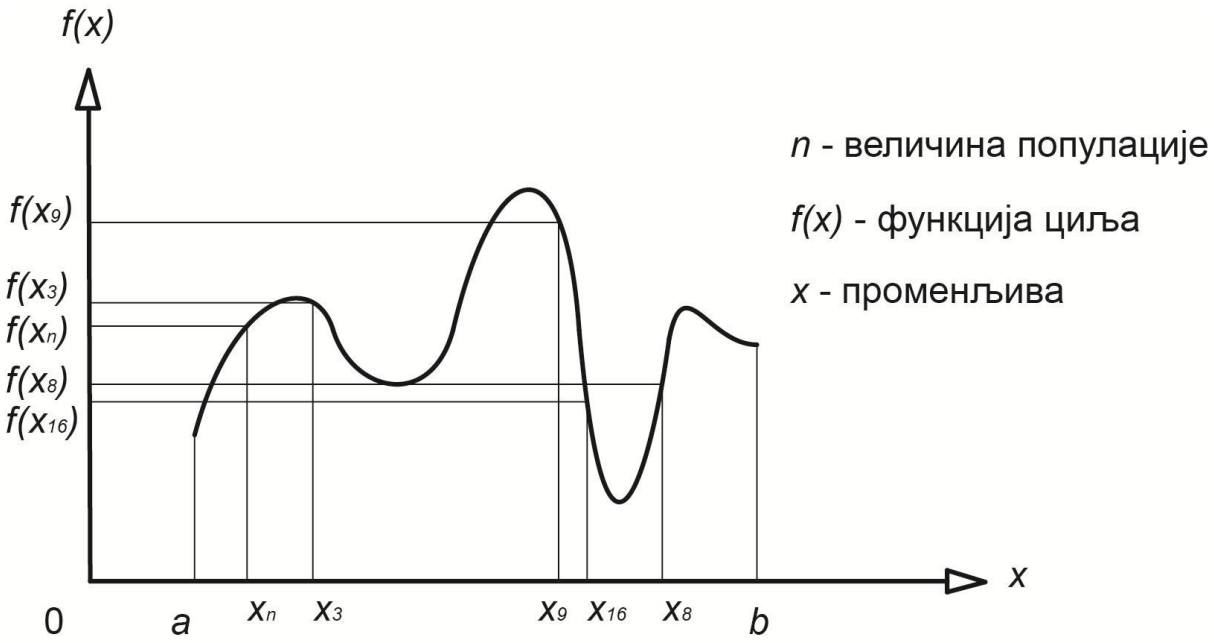
Као приоритете развоја софтвера неопходно је истаћи:

- Функционалност и флексибилност формулације и решавања оптимизационих проблема,
- Независност целина предпроцесирања, процесирања и постпроцесирања, што омогућава имплементацију већег броја хеуристичких оптимизационих метода,
- Контрола рада, праћење конвергенције и комуникација са корисником.

Да би се реализовали циљеви, са акцентом на приоритете, потребно је решити низ захтевних проблема. Највећу пажњу потребно је посветити конципирању и реализацији метода хеуристичке оптимизације. Метода мора да обезбеди теоријски прописани рад, са квалитетном конвергенцијом, превазилажењем локалних екстремних вредности и брзим радом. Све методе морају радити са великим бројем променљивих уз задовољење свих постављених ограничења. Пажња при изради софтвера посвећена је и кориснику. Без обзира што ово није комерцијална апликација, посвећена је пажња да се спрече и отклониле потенцијалне грешке корисника, а сам изглед апликације је осмишљен и реализован тако да буде што једноставнији за коришћење. Дакле, омогућено је да се ток решавања оптимизационих проблема одвија несметано. Представљени циљеви и приоритети су смернице о којима треба стално водити рачуна током развојног процеса, а реализација софтвера захтева конкретне кораке у имплементацији процеса предпроцесирања, процесирања и постпроцесирања.

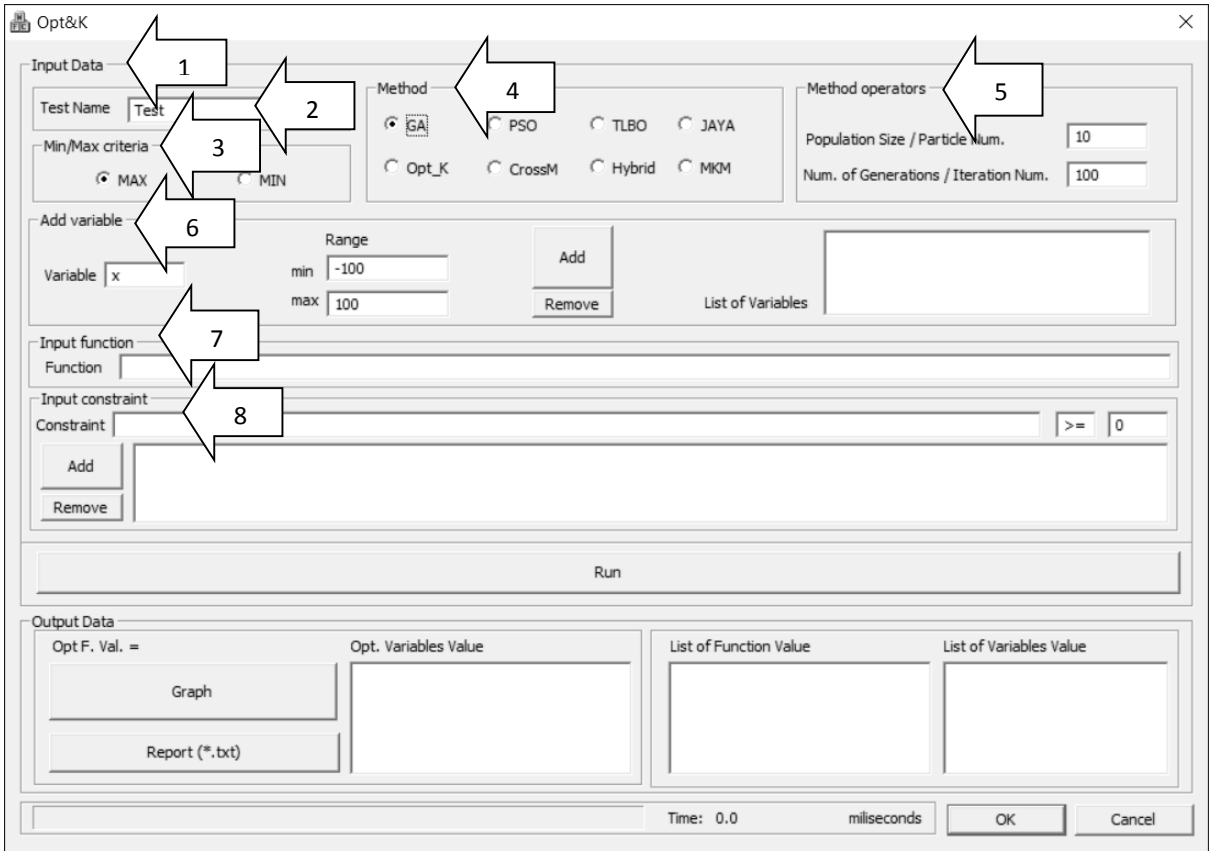
4.1 Предпроцесор

За почетак је неопходно дефинисати проблем који се оптимизује у виду математичког модела (функција циља, промењиве оптимизације, ограничења). Потом се врши креирање иницијалне популације. Поред математичког модела улазни подаци морају да садрже и величину популације и број генерација. Величина популације дефинише колико ће јединки бити креирано у иницијалној популацији и у каснијим генерацијама, а број генерација представља критеријум заустављања алгоритма. Број генерација дефинише колико ће пута бити поновљен комплетан поступак током оптимизације. Иницијална популација представља матрицу случајних вредности промењивих оптимизације из дозвољеног домена. Скуп свих промењивих које учествују у оптимизацији назива се сет. Један сет има једнозначно решење функције циља. Број сетова у популацији је број величине популације. Код иницијалне популације случајне вредности дају решења функције циља у дозвољеним границама, што је и представљено на слици 4.2 за проблем са једном промењивом.



Слика 4.2. Креирање иницијалне популације за проблем са једном промењивом

Основни задатак предпроцесорске фазе јесте унос података потребних за процес оптимизације и интеракција корисника са апликацијом. Из тог разлога формирана је јасна апликација на којој корисник може визуелно да раздвоји фазе које га очекују у оптимизационом процесу. Визуелно су раздвојени делови за унос, покретање и излазне резултате. Оваква структура директно утиче на комплексност процеса оптимизације и у овом случају чини га једноставнијим и доступнијим. Део апликације који је предвиђен за унос података представљен је на слици 4.3.



Слика 4.3. Део апликације за унос података потребних за процес оптимизације

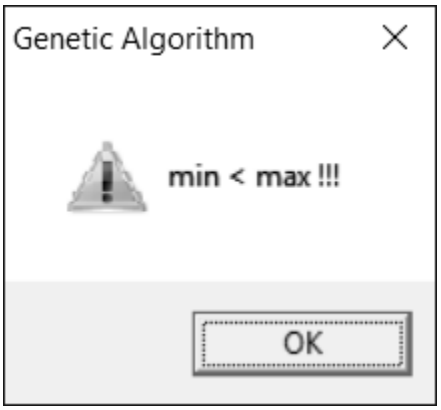
Група алата који су развијени ради уноса података нумерисана је бројем 1 на слици 4.3 (*Input Data*).

Бројем 2 означена је зона у коју се уноси назив теста (*Test Name*) који се обавља. Иницијални текст предвиђен за назив теста увек је „тест“. Назив сваког теста појединачно корисник може сам да дефинише, а софтвер ће радити и без уноса ове вредности. Софтвер је развијен тако да се овај назив по правилу користи за именовање извештаја који је могуће креирати у постпроцесорској фази, што такође корисник може да измени.

Бројем 3 означен је критеријум оптимизације (*Min / Max criteria*). Овај критеријум представља избор између минимизације или максимизације предвиђене функције, коју бира корисник. Предвиђено подешавање је на (*MAX*) максимум, ако корисник не подеси другачије ову опцију.

Следећа ознака, број 4, означава избор методе оптимизације. Овде је представљена група развијених метода и корисник може да изврши избор којом методом хеуристичке оптимизације ће вршити процес оптимизације. На слици је представљен финални изглед софтвера који садржи све уграђене методе оптимизације. У циљу квалитетног истраживања, у развијени софтвер уграђене су многобројне актуелне хеуристичке методе оптимизације. Ове методе користиле су за анализу, оцену, процену и развој нових метода. Софтвером је омогућена употреба 8 различитих оптимизационих метода. Коначне ознаке у софтверу можда не асоцирају директно корисника на методу, јер се са *GA* покреће модификовани *iGA*. Ознакама *PSO, TLBO, JAYA*, дефинисани су оригинални алгоритми *PSO, TLBO* и *Jaya*. Са *Opt_K* означен је *DINDI* алгоритам, са *CrossM* модификација *rTLBO*, док је са *Hybrid* означена хибридна метода *hGPT*. Испитивање модификације са паралелним приступом са *TLBO* и *Jaya* вршена је помоћу *MKM* поља.

Оператори метода нумерисани су бројем 5 и представљају вредности величине популације и броја генерација. Ове величине подешава корисник у зависности од потребног броја прерачунавања функције циља.
Бројем 6 представљено је поље које се односи на иницијализацију и ограничавање променљивих оптимизације. Поље променљива (*Variable*) представља поље где је могуће дати назив променљивој. При уносу променљивих могуће је дати било који назив, а *x* је предложени назив прве променљиве. Само једна променљива може имати један назив, а тај назив може бити комбинација слова и бројева, само не могу бити називи постојећих функција. Када се додели име променљивој подешавају се границе (*Range*) променљиве, што представља вредности које може променљива имати. Овде је могуће унети било коју реалну или целобројну, позитивну или негативну вредност. Када је извршено додељивање опсега променљиве, променљиву је потребно додати у листу променљивих (*List of Variables*). Притиском на тастер (*Add*) задата променљива уписује се у листу променљивих која садржи све додате променљиве. Пошто је предвиђено да софтвер извршава функцију за неограничен број променљивих, у листу променљивих могуће је додати колико год је потребно променљивих. Променљиве из листе не морају обавезно учествовати у оптимизацији. Ако кориснику због прегледности смета нека вредност у листи или је извршен унос исте променљиве, могуће је уклонити је притиском на тастер (*Remove*). Пре тога је неопходно селектовати променљиву која се уклања. У процесу развоја софтвера вођено је рачуна и о интеракцији са корисником. Тако на пример минимална вредност променљиве не може бити већа од максималне. Ако корисник случајно превиди ову нелогичност, софтвер ће појавити прозор са упозорењем, као на слици 4.4, где ће кориснику указати на грешку.



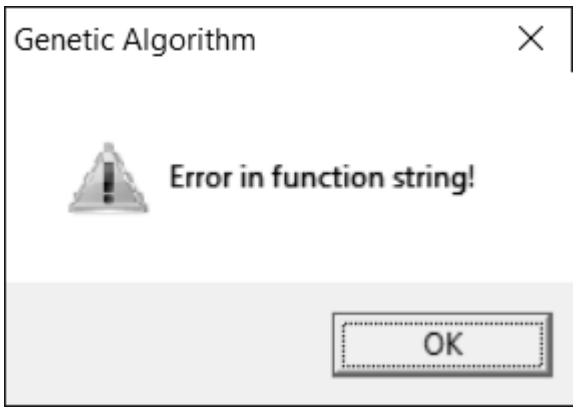
Слика 4.4. Упозорење кориснику на грешку при постављању граница променљиве

Табела 4.1
Неке од могућих функција и операција које препознаје софтвер

+	Сабирање	-	Одузимање	*	Множење
/	Дељење	^	Степен	sqrt()	Корен
π	3,1415926535897932	e	2,718281828	abs()	Апсолутна вредност
cos()	Косинус задатог угла	sin()	Синус задатог угла	tan()	Тангенс задатог угла
acos()	Аркус косинус	asin()	Аркус синус	atan()	Аркус тангенс
cot()	Котангенс	cosh()	Хиперболички косинус	atanh()	Аркус угла хиперболичког тангенса

if(A,B,C)	Логичка операција. Ако је int(A) различито од 0 враћа вредност B, а у супротном C.	int()	Враћа интеџер вредност	log()	Логаритам са основом e
log10()	Логаритам са основом 10	max(A,B)	Враћа већу вредност. Ако је A>B, враћа A и обрнуто.	min(A,B)	Враћа мању вредност од A и B

У листи промењљивих дуплим кликом на промењљиву исписује се њена вредност у зони записа функције циља (*Function*). Ово поље нумерисано је бројем 7 и омогућава кориснику унос било које функције циља. За рад овог дела коришћен је подпрограм под називом *Function Parser for C++*. Овај подпрограм служи за очитавање функције, превођење функције из текстуалног у функционални облик, који су развили *Juha Neiminen* и *Joel Yliluoma* и овај подпрограм заштићен је *GNU Lesser General Public License*. Овај подпрограм је бесплатан за употребу, само није дозвољена његова комерцијализација без поштовања одређених услова. Овај софтвер је развијен због истраживања, тако да његова комерцијализација није ни планирана. Постоје још многобројне алтернативе за очитавање текстуалног записа функције, али је у овом истраживању то споредна ствар. У поље уноса функције могуће је унети велики број рачунских операција и функција. Неке од ових функција представљене су у табели 4.1. Корисник није ограничен само представљеним операцијама из табеле 4.1, али су у табели представљене најчешће коришћене. У поље за унос функције није могуће унети промењљиву која није претходно декларисана. Када корисник то покуша, појавиће се упозорење као на слици 4.5.

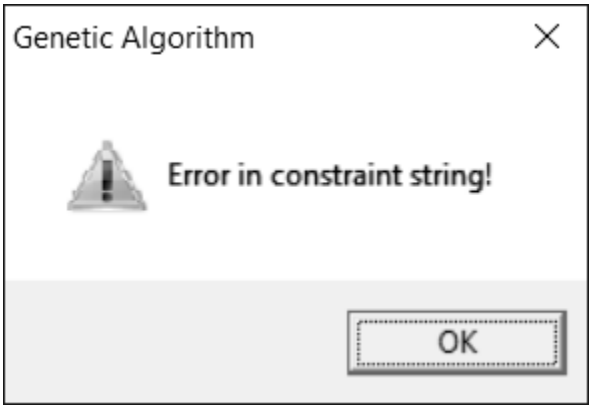


Слика 4.5. Упозорење на грешку у функцији циља

Ово упозорење односи се и на погрешно унете функције и друге грешке, а помаже кориснику да ваљано обави процес оптимизације. Касније за добијање резултата користи се особина еквивалентности код оптимизације, као у изразу (4.1), тако да за сам рад софтвера не представља проблем израчунавања и минималне и максималне вредности неке функције. Корисник има могућност избора критеријума оптимизације, али без обзира на критеријум, оптимизација се врши на потпуно идентичан начин, што и дефинише овај израз:

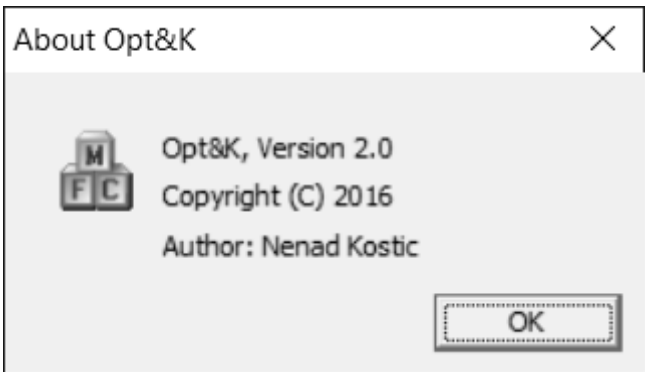
$$\min f(X) = max\{-f(X)\}. \tag{4.1}$$

На слици 4.3, бројем 8 представљено је поље за унос ограничења оптимизације. У ово поље могу бити унети било који облици неједнакости. Могуће је унети било који број ограничења, а када се изврши дефинисање неког ограничења потребно га је додати у листу ограничења коришћењем одговарајућег тастера (*Add*). Поред тога, могуће је и обрисати нежељено ограничење из листе (*Remove*). Ако постоји нека неправилност приликом додавања ограничења појавиће се прозор са упозорењем, као на слици 4.6.



Слика 4.6. Упозорење на грешку у ограничењима

Постоји још велики број упозорења која су уграђена у софтвер, али и поред њих вероватно постоји неки сегмент који није превентивно обезбеђен од неправилне употребе. Ова апликација није комерцијална, па то не представља проблем, само је важно знати да постоји могућност појаве грешке услед неправилне употребе.
Да би корисник видео податке о софтверу потребно је да кликне у горњи леви угао апликације, изабере опцију *About* и на екрану ће се појавити прозор као на слици 4.7.

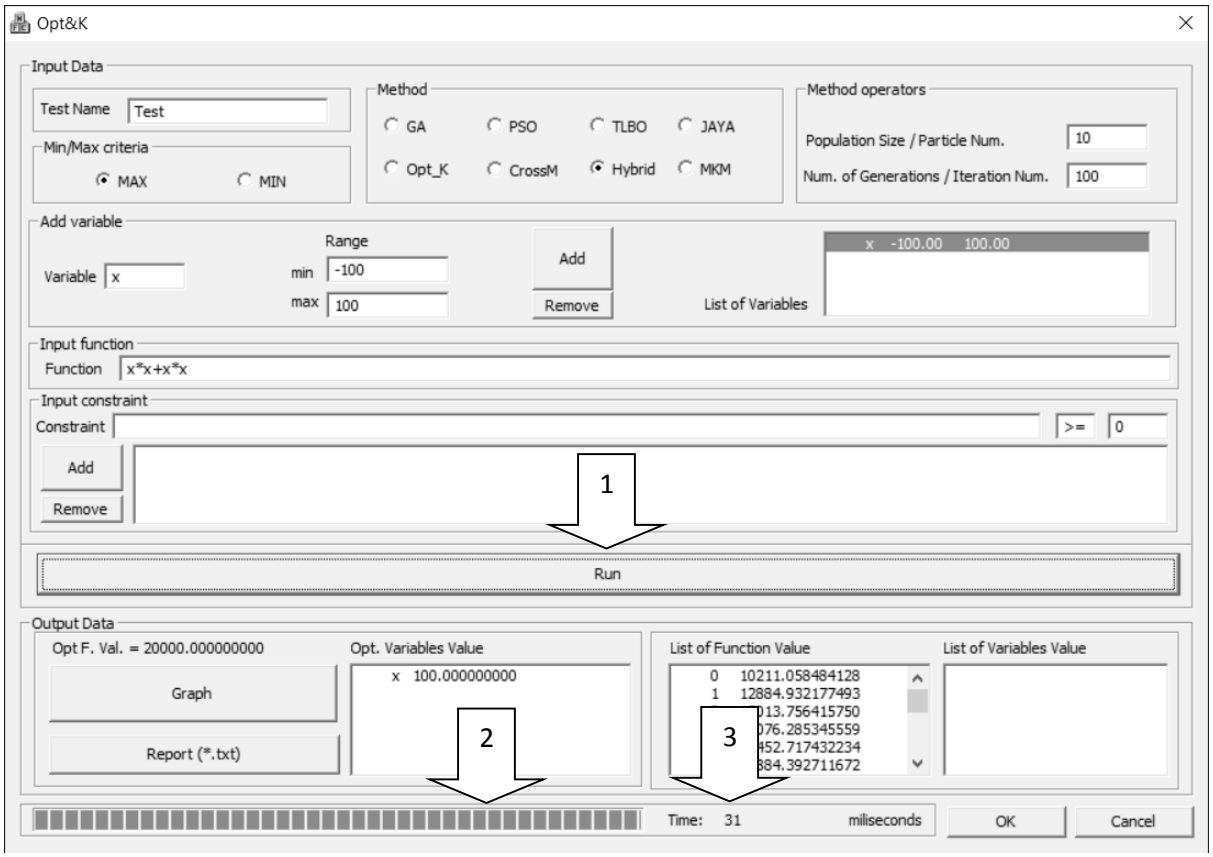


Слика 4.7. Основне информације о развијеном софтверу

Вредности које су поменуте не морају бити попуњене представљеним редоследом, а софтвер је развијен тако да се у случају недостатка неке величине појави упозорење које ће обавестити корисника о томе шта недостаје. Предпроцесорски део подразумева и позадину иницијализације промењљивих, креирање иницијалне популације и очитавање функције циља и ограничења. Ту почиње веза између препроцесорског и процесорског дела.

4.2 Процесор

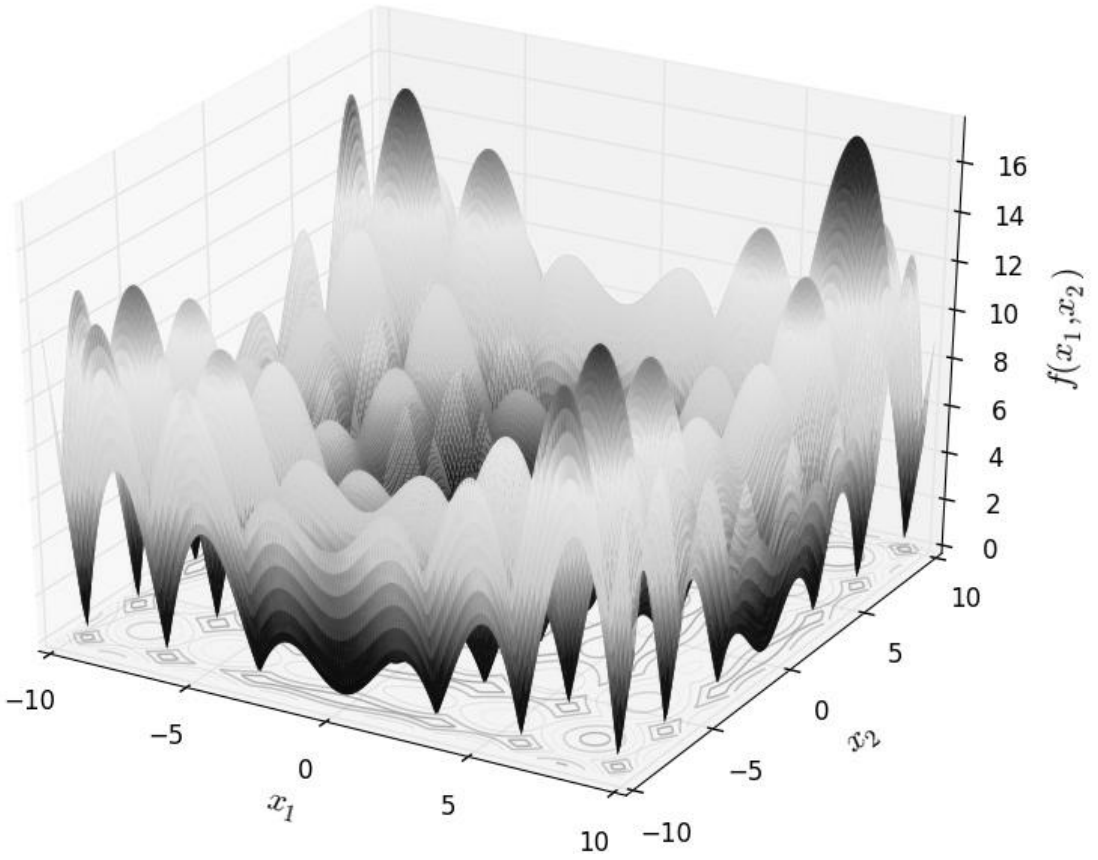
Процесорски део представља све операције над иницијализованим вредностима, рачунајући и алгоритамски рад изабране методе оптимизације. Ова целина софтвера почиње са радом притиском на тастер (*Run*) за покретање прорачуна. Софтвер ради под условима које је поставио корисник у делу уноса података. Првенствено се изврши провера релевантности ових вредности и пријављују се евентуалне грешке. Део који представља процесор налази се у позадини софтвера, а корисник може да прати ток овог процеса. Изглед дела апликације за процесирање представљен је на слици 4.8.



Слика 4.8. Део софтвера за покретање и праћење процесирања

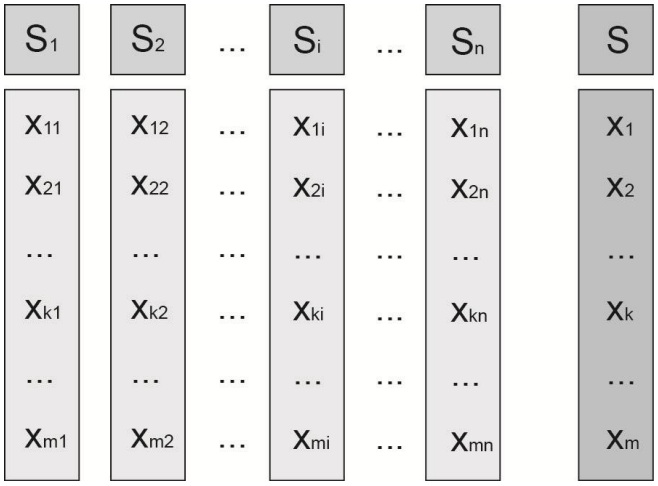
Када је унос комплетан и исправан могуће је покренути процесорски део притиском на одговарајући тастер (*Run*) који је на слици 4.8 нумерисан бројем 1. Пошто се овим поступком покреће део који није доступан кориснику (оптимизациони процес), неопходно је да постоји могућност да се прати процес, како би корисник уопште знао да се нешто дешава. Ово је постигнуто прогрес баром где се са повећањем броја итерација види докле је прорачун стигао, а друга величина којом је могуће пратити је време. Време је често важно и за сам процес оптимизације, а не само за праћење да ли се овај процес одвија. Прогрес бар је обележен бројем 2, док је индикатор времена нумерисан бројем 3 на слици 4.8. Време је циљно дато у милисекундама, пошто софтвер изузетно брзо функционише. На овај начин могуће је пратити време рада софтвера и за најједноставније проблеме и за најкомплексније проблеме.

Приликом развоја процесорског дела и имплементације хеуристичких метода неопходно је водити рачуна о решавању функција са великим бројем локалних екстремних вредности, као што је приказана на слици 4.9.



Слика 4.9. Функција са више локалних екстремних вредности – мултимодална функција (функција *Alpine01*)

У оваквим функцијама постоји само по један глобални минимум и максимум, а све остале екстремне вредности су локалног карактера. Ова функција има одређени домен и мора да задовољи постављена ограничења. То значи да је потребно наћи оптимум за целу могућу област претраге – глобални оптимум. Ово представља озбиљан задатак у процесу оптимизације, а превазилази се адекватном имплементацијом метода оптимизације. Хеуристичке методе су веома успешне при проналажењу глобалних екстремних вредности, а њихове фазе обавезно садрже механизме за превазилажење локалних екстремних вредности. Ово се првенствено постиже радом са популацијом. За време прорачуна у позадини софтвера прво се врши генерисање иницијалне популације. Популација се креира у зависности од величине популације и броја променљивих које фигуришу у оптимизационом процесу. Популација практично представља матрицу димензија „број променљивих“ x „величина популације“. Пример популације представљен је на слици 4.10.

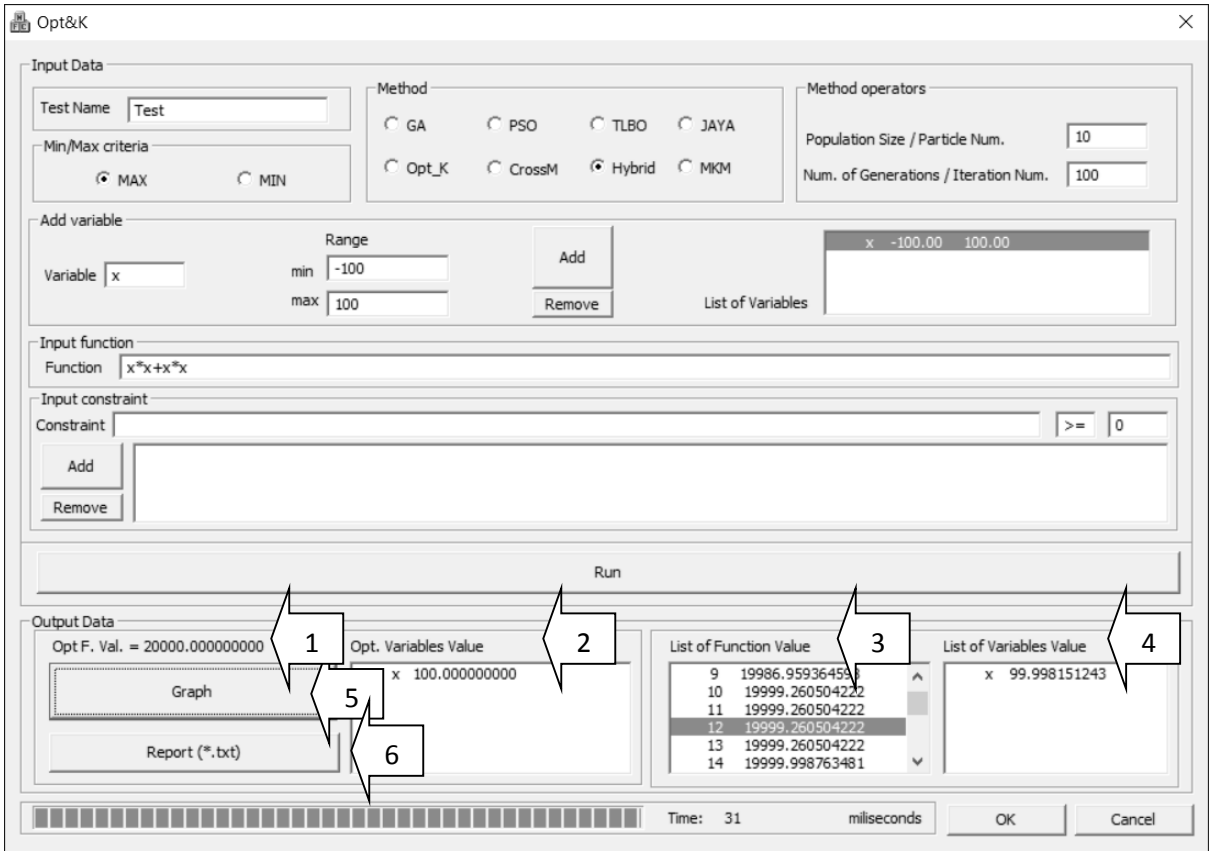


Слика 4.10. Популација у процесу оптимизације

Скуп свих променљивих (x_1, x_2, …, x_n) оптимизације представља један сет (S). Скуп свих сетова чини популацију (S_1, S_2, …, S_n). Сетова има онолико колика је квантитативна вредност величине популације. Свака представљена променљива има квантитативну вредност у дозвољеном интервалу, а сваки сет даје квантитативну вредност функције циља. Ове вредности распоређене су по целом пољу претраге. На овај начин функционишу све хеуристичке методе, а рад са популацијом представља њихову основну предност. У свакој итерацији алгоритма врше се операције над овим сетовима, а променљиве мењају своје вредности у дозвољеним интервалима, где се за то време врши израчунавање и провера вредности функције циља. Сетови се мењају тако да се итерационо појављују боље вредности функције циља, а тиме решење конвергира ка оптимуму. Хеуристичке методе оптимизације у математичком облику имплементиране су у процесорски део према претходно представљеним алгоритмима.
Ток процесирања могуће је и накнадно анализирати у делу постпроцесирања. Процесорски део представља најзначајнију целину за ово истраживање, јер приказује начине и поступке решавања оптимизационих проблема хеуристичким методама оптимизације.

4.3 Постпроцесор

Када је прорачун завршен почиње фаза постпроцесирања. Софтверски је предвиђено да добијене вредности могу бити накнадно визуализоване и анализиране. На слици 4.11. представљена су поља која се односе на постпроцесорски део.



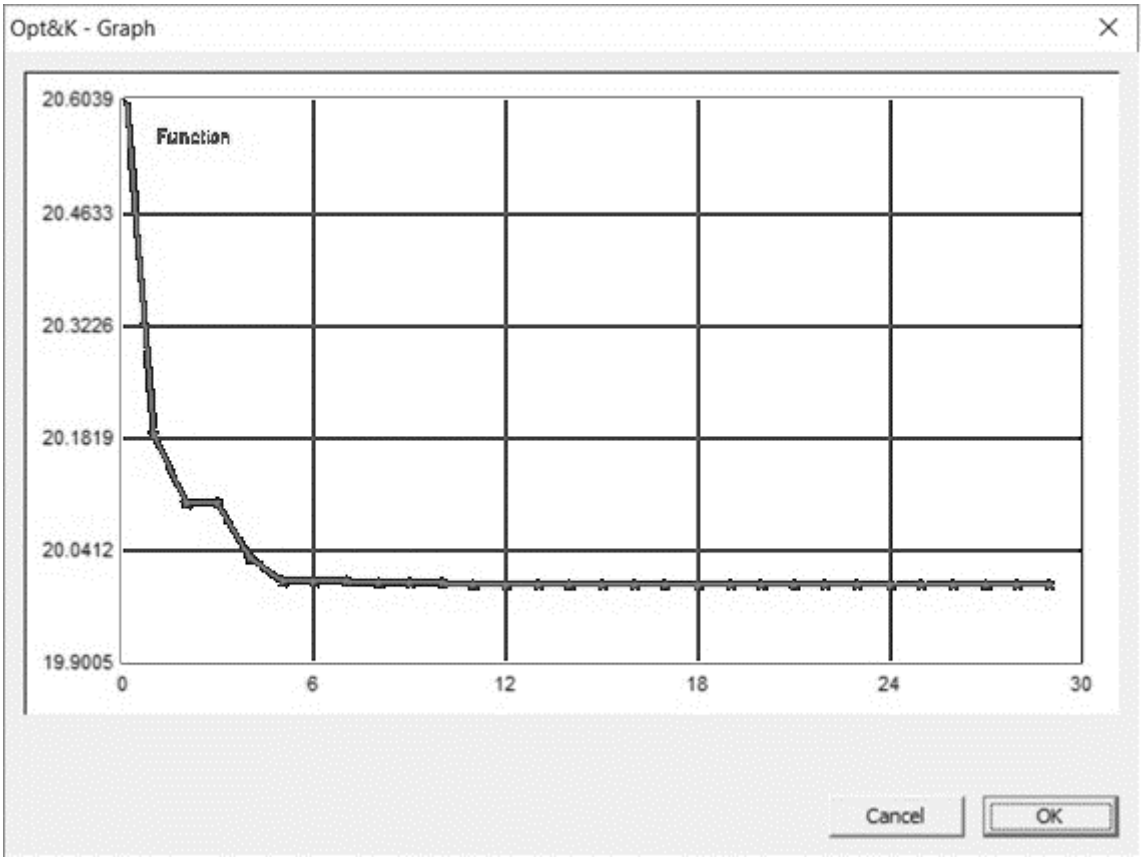
Слика 4.11. Део софтвера за приказ, обраду, анализу и архивирање резултата оптимизације (постпроцесирање)

Резултат оптимизације представљен је бројем 1, и означава оптималну вредност функције циља, за унете променљиве уз задовољење ограничења. Ова вредност је оптимум (или тежи оптимуму) који је постигнут у односу на укупан, дефинисани број итерација. Ова вредност не мора бити апсолутно тачна (оптимална) ако је у питању комплексан проблем, а прерачунавања функције циља у процесу оптимизације незнатна. Оптималном вредношћу подразумева се квантитативна вредност функције циља у последњој итерацији.

Бројем 2 означена је листа вредности свих променљивих које учествују у оптимизацији. За сваку променљиву исписује се назив и вредност променљиве са којом функција циља постиже оптимум.

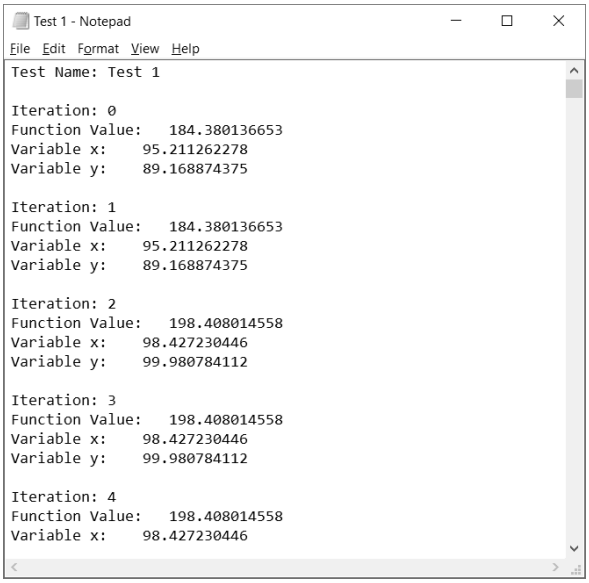
Бројем 3 представљена је листа најбољих вредности функције циља за одређену итерацију. Исписује се прво итерација у којој је постигнута вредност, а потом и вредност функције за ту итерацију. Овом листом могуће је прегледати ток оптимизационог процеса и ток утврђивања конвергенционих карактеристика. За сваку вредност функције циља за било коју итерацију и променљиве имају одређене вредности. Вредности променљивих могуће је очитати из листе 4, селекцијом жељене итерације из листе 3. На овај начин могуће је анализирати комплетан ток оптимизационог процеса.

Оптимизацију је најлакше пратити одређивањем екстремне (најбоље) вредности за сваку итерацију, а начин на који је приказано праћење, представља график зависности екстремне вредности функције циља и броја итерација. Кориснику је омогућен и преглед овог графика који је нумерисан бројем 5 (*Graph*) и пример графика представљен је на слици 4.12.



Слика 4.12. Графички приказ тока оптимизације

Кориснику је омогућено да креира извештај (*Report*) који има екстензију (**.txt*), и чији је препоручени назив при снимању документа исти као и назив теста. Овим се омогућава чување података експеримента, где извештај садржи назив експеримента и итерационе вредности функције и променљивих. Пример извештаја представљен је на слици 4.13.



Слика 4.13. Извештај резултата оптимизације

Са поменутим вредностима могуће је извршити све потребне анализе процеса оптимизације и адекватно употребити постигнуте резултате.
Дакле, поступак оптимизације у пракси своди се на три кључна сегмента: предпроцесирање, процесирање и постпроцесирање. Предпроцесорски сегмент своди се на једнозначно дефинисање проблема, успостављање односа утицајних параметара и постављање услова у којима систем треба да ради и функционише. Овај сегмент подразумева и избор неке од познатих метода, њихову адаптацију – модификацију или развој нове методе, као и софтверску имплементацију одабране методе. Процесорски део подразумева повезивање математичке формулације са методом оптимизације и методом дефинисане операције над математичким моделом. Постпроцесорски сегмент подразумева визуелизацију резултата, анализу ваљаности оптимизационог процеса и одређивање квантитативних вредности стварног или стварних оптимума.

5. Тестирање развијених хеуристичких метода оптимизације

Практични проблеми инжењерске оптимизације подразумевају проналажење оптимума код нових (до тада непознатих) проблема, што за последицу има боље карактеристике система или конструкције која се анализира и уштеде са друге стране. Како би у процесу оптимизације уопште знали да ће конвергенција бити усмерена ка оптимуму, неопходно је извршити верификацију рада оптимизационог процеса (методе, математичког модела, софтвера, итд.) на проблемима за које је оптимум познат, бар на нивоу неког решења које се тако третира. Практично, у процесу оптимизације, никада са сигурношћу не можемо тврдити да ли је оптимум постигнут или је постигнуто само неко алтернативно решење које се третира као оптимум.
Модерне оптимизационе методе (хеуристичке методе оптимизације) обезбеђују да конвергенција ка оптимуму на првом месту буде свакако могућа, а поред тога да буде и повољна и поуздана. Хеуристичке методе са веома малим бројем познатих чињеница, тачније само са основном математичком формулацијом проблема могу постићи оптимум задатог проблема. Због погодности које ове методе пружају, посвећена им је велика пажња истраживача, а њихова популарност непрестано расте. Сва истраживања усмерена су на правац развоја нових метода бољих карактеристика, хибридизацију познатих метода и модификацију неке такође познате методе у циљу повећања тренутних перформанси. Дакле, потреба сталног развоја нових метода заиста постоји због ограничења која постојеће методе имају код њихове примене. Ако је развој неке методе усмерен само на одређену групу (типове) проблема, не постоји никаква гаранција да ће метода бити успешна за глобалну оптимизацију за све врсте оптимизационих проблема. Често се тестирањем хеуристичких метода покаже супротно. Методе које су превише добре за тачно одређену групу проблема своје недостатке покажу када се примене на потпуно другом типу проблема.
Хеуристичке методе које су развијене у блиској прошлости су све боље, брже, лакше за софтверску имплементацију и адаптацију проблему и покривају велики опсег оптимизационих задатака. Како би било могуће одредити да ли нека метода задовољава поменуте критеријуме и у којој мери, неопходно је на адекватан начин извршити тестирање методе на стандардним тест проблемима. Проблеми (тест примери) који се користе за тестирање оптимизационих алгоритама могу се класификовати у три групе:

- Оптимизациони проблеми без ограничења,
- Оптимизациони проблеми са ограничењима,
- Инжењерски оптимизациони проблеми.

Без обзира којој групи припадају, проблеми су веома комплексни и није могуће тврдити да су проблеми без ограничења лакши за решавање од проблема са ограничењима или доносити закључке у неком другом правцу. За тестирање метода оптимизације неопходно је извршити верификацију на свим поменутим групама проблема како би било могуће дефинисати ефикасност и стабилност рада неке методе. За потребе овог

истраживања извршена су тестирања за све три групе проблема, при чему су испитивања спроведена према препорукама из претходно наведене литературе (устаљена тестирања) еминентних стручњака, а вредности упоређене са вредностима постигнутим у литератури.
Рад сваког еволуционог алгоритма или алгоритма честица (хеуристичких метода) садржи креирање неког вида иницијалне популације у дозвољеном домену, а као основна карактеристика рада подразумева се величина која дефинише број прерачунавања функције циља у процесу оптимизације (*FEs – function evolutionary*). Величина *FEs* је индивидуална величина за сваку методу и зависи од начина рада алгоритма и броја фаза. Ова величина представља производ броја прерачунавања функције циља у једној итерацији и укупног броја итерација у оптимизационом процесу. Може да зависи и од неких додатних, специфичних израчунавања функције циља, али то зависи од конкретног случаја. Што је мања вредност *FEs* – а метода је ефикаснија и обрнуто. На основу резултата оваквог тестирања биће могуће извести закључке о раду развијених метода и њихова класификација према перформансама које нуде.

5.1 Тестирање метода за решавање оптимизационих проблема без ограничења

За почетно експериментално испитивање рада развијених метода изабрана је група проблема без ограничења. Тестирање је извршено за *DINDI, iGA, rTLBO* и *hGPT* оптимизационе алгоритме, а вредности су упоређене са радом алгоритама из литературе. Укупно 13 функција без ограничења разматрано је за потребе овог истраживања, а резултати су анализирани за 30 узастопно поновљених симулација, што је препорука за тестирање рада метода [28, 32, 131, 138, 151]. Математичка формулација, карактеристике проблема, број димензија проблема представљене су у табели 5.1, где су дати сви потребни детаљи за примену функције циља код тестирања.

У табели 5.1 представљене су функције различитих карактеристика, што може да одреди квалитет рада оптимизационих метода. Оптимизациони алгоритми из литературе који су коришћени за поређење су *GA, PSO, DE, ABC, TLBO* и *Jaya*. Интересантне вредности анализе експеримента представљене су у табелама 5.2.1 и 5.2.2. За овај експеримент су важне средње вредности за 30 узастопно поновљених симулација и стандардна девијација. Циљ овог испитивања је верификација рада развијених метода за глобалну оптимизацију на широком спектру проблема, а првенствено на тест проблемима без ограничења. Као основне препоруке испитивања коришћене су препоруке из [32], где је вредност *FEs* – а 500000, такође према препорукама узета као максимална за све разматране алгоритме и проблеме.

Табела 5.1

Тест функције без ограничења коришћене за тестирање хеуристичких метода

Бр.	Функција	Формулација	D	Домен	C
BF1	Sphere	$F_{min} = \sum_{i=1}^{D} x_i^2$	30	[-100;100]	US
BF2	Sum Squares	$F_{min} = \sum_{i=1}^{D} i x_i^2$	30	[-10;10]	US
BF3	Beale	$(1{,}5 - x_1 + x_1x_2)^2 + (2{,}25 - x_1 + x_1x_2^2)^2 + +(2{,}625 - x_1 + x_1x_2^3)^2$	5	[-4,5;4,5]	UN
BF4	Easom	$-\cos(x_1)\cos(x_2)\exp(-(x_1-\pi)^2 - (x_2-\pi)^2)$	2	[-100;100]	UN
BF5	Matyas	$0{,}26(x_1^2 + x_2^2) - 0{,}48x_1x_2$	2	[-10;10]	UN
BF6	Zakharov	$\sum_{i=1}^{D} x_i^2 + \left(\sum_{i=1}^{D} 0{,}5ix_i\right)^2 + \left(\sum_{i=1}^{D} 0{,}5ix_i\right)^4$	10	[-5;10]	UN
BF7	Trid 6	$\sum_{i=1}^{D}(x_i - 1)^2 - \sum_{i=2}^{D} x_i x_{i-1}$	6	[-D²;D²]	UN
BF8	Bohachevsky 1	$x_1^2 + 2x_2^2 - 0{,}3\cos(3\pi x_1) - 0{,}4\cos(4\pi x_2) + 0{,}7$	2	[-100;100]	MS
BF9	Branin	$\left(x_2 - \frac{5{,}1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	2	[-5;10] [0;15]	MS
BF10	Booth	$(x_1 - 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$	2	[-10;10]	MS
BF11	Michalewicz 2	$-\sum_{i=1}^{D} \sin x_i \left(\sin\left(\frac{ix_i^2}{\pi}\right)\right)^{20}$	2	[0;π]	MS
BF12	Bohachevsky 3	$x_1^2 + 2x_2^2 - 0{,}3\cos(3\pi x_1 + 4\pi x_2) + 0{,}3$	2	[-100;100]	MN
BF13	GoldStein-Price	$\left[1 + (x_1 + x_2 + 1)^2 \cdot \cdot (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\right] \cdot \left[30 + (2x_1 - 3x_2)^2 \cdot \cdot (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)\right]$	2	[-2;2]	MN

D: Димензија, *C*: Карактеристика, *U*: Једномодалне, *M*: Мултимодалне,
S: Дисконтинуалне, *N*: Континуалне

Табела 5.2.1

Упоредни оптимизациони резултати за тест функције без ограничења (разматрано за 500000 *FEs* – а и 30 узастопно поновљених симулација)

Бр.		GA	PSO	DE	ABC	TLBO	Jaya
BF1	Средња вредност	1,11E+03	0	0	0	0	0
	Стандардна девијација	74,214474	0	0	0	0	0
BF2	Средња вредност	1,48E+02	0	0	0	0	0
	Стандардна девијација	12,409289	0	0	0	0	0
BF3	Средња вредност	0	0	0	0	0	0
	Стандардна девијација	0	0	0	0	0	0
BF4	Средња вредност	-1	-1	-1	-1	-1	-1
	Стандардна девијација	0	0	0	0	0	0
BF5	Средња вредност	0	0	0	0	0	0
	Стандардна девијација	0	0	0	0	0	0
BF6	Средња вредност	0,013355	0	0	0,0002476	0	0
	Стандардна девијација	0,004532	0	0	0,000183	0	0
BF7	Средња вредност	-49,9999	-50	-50	-50	-50	-50
	Стандардна девијација	2,25E-05	0	0	0	0	0
BF8	Средња вредност	0	0	0	0	0	0
	Стандардна девијација	0	0	0	0	0	0
BF9	Средња вредност	0,397887	0,3978874	0,3978874	0,3978874	0,3978874	0,3978874
	Стандардна девијација	0	0	0	0	0	0
BF10	Средња вредност	0	0	0	0	0	0
	Стандардна девијација	0	0	0	0	0	0
BF11	Средња вредност	-1,8013	-1,572869	-1,8013	-1,8013	-1,8013	-1,8013
	Стандардна девијација	0	0,11986	0	0	0	0
BF12	Средња вредност	0	0	0	0	0	0
	Стандардна девијација	0	0	0	0	0	0
BF13	Средња вредност	5,870093	3	3	3	3	3
	Стандардна девијација	1,071727	0	0	0	0	0

Табела 5.2.2

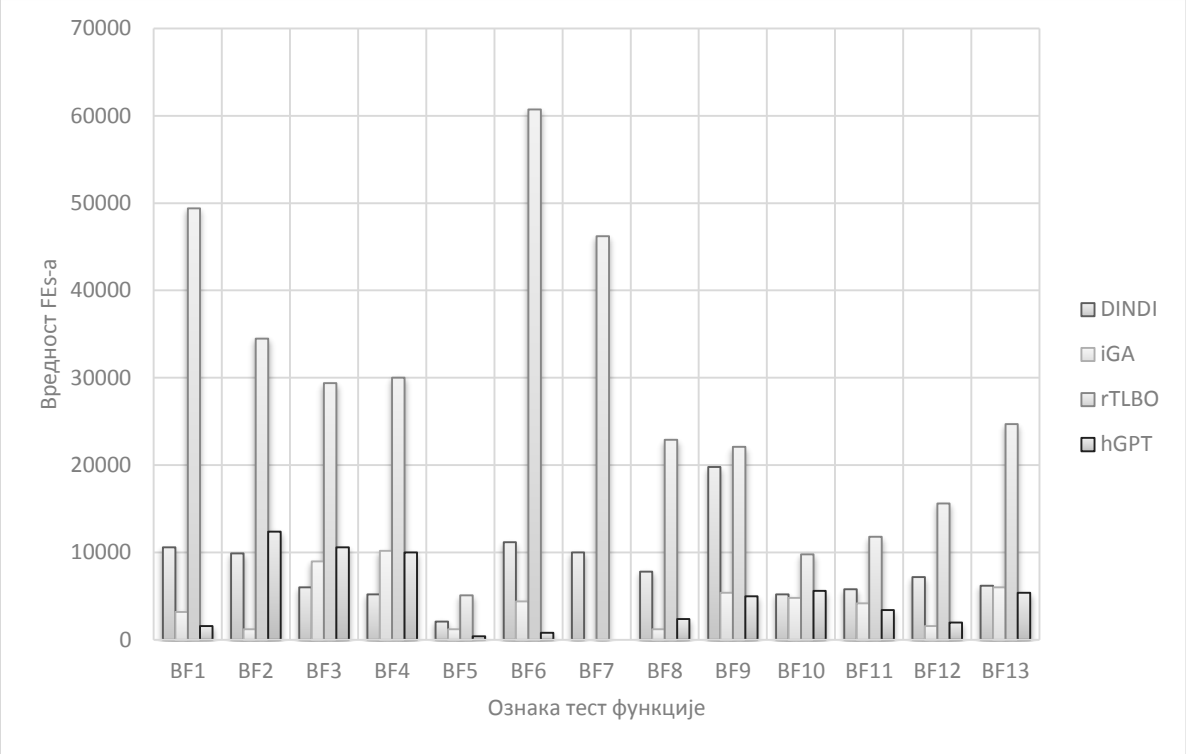
Упоредни оптимизациони резултати за тест функције без ограничења (разматрано за 500000 *FEs* – а и 30 узастопно поновљених симулација)

Бр.		**DINDI**	**iGA**	**rTLBO**	**hGPT**
BF1	Средња вредност (Најбоља)	0	0,000060 (0)	0,000002 (0)	1,540958 (0)
	Стандардна девијација	0	0,000206	4,47E-06	5,41995
BF2	Средња вредност (Најбоља)	0	0,000002 (0)	0,000002 (0)	0,209328 (0)
	Стандардна девијација	0	4,62E-06	1,05E-05	1,032261
BF3	Средња вредност (Најбоља)	0	0	0	0
	Стандардна девијација	0	0	0	0
BF4	Средња вредност (Најбоља)	-1	-1	-1	-1
	Стандардна девијација	0	0	0	0
BF5	Средња вредност (Најбоља)	0	0	0	0
	Стандардна девијација	0	0	0	0
BF6	Средња вредност (Најбоља)	0	0	0,000009 (0)	0
	Стандардна девијација	0	0	3,29E-05	0
BF7	Средња вредност (Најбоља)	-50	-49,988542 (-49,9996)	-49,99991 (-50)	-49,925425 (-49,9809)
	Стандардна девијација	0	0,014083	1,59E-05	0,039074
BF8	Средња вредност (Најбоља)	0	0	0	0
	Стандардна девијација	0	0	0	0
BF9	Средња вредност (Најбоља)	0,39788736	0,397887	0,397887	0,397887
	Стандардна девијација	0	0	0	0
BF10	Средња вредност (Најбоља)	0	0	0	0
	Стандардна девијација	0	0	0	0
BF11	Средња вредност (Најбоља)	-1,8013	-1,8013	-1,8013	-1,8013
	Стандардна девијација	0	0	0	0
BF12	Средња вредност (Најбоља)	0	0	0	0
	Стандардна девијација	0	0	0	0
BF13	Средња вредност (Најбоља)	3	3	3	3
	Стандардна девијација	0	0	0	0

У табели 5.2.1 представљене су вредности из литературе за *GA, PSO, DE, ABC, TLBO* и *Jaya*, док су у табели 5.2.2 представљене вредности добијене експерименталним испитивањем (како у литератури називају ова испитивања) за развијене алгоритме *DINDI, iGA, rTLBO* и *hGPT*. Све приказане методе из ових табела имају изузетне конвергенционе карактеристике, без обзира што је уочљиво да у неким ситуацијама вредности одступају од апсолутних оптимума. На првом месту одступања су мала и често зависе само од броја разматраних децимала, а код практичне примене би углавном и овај проблем могао бити превазиђен повећањем величине *FEs* – а. За проблеме BF1-BF10 алгоритми *PSO, DE, TLBO* и *Jaya* дају оптималне вредности (најбоље познате) за свих 30 поновљених испитивања. *ABC* алгоритам за проблеме од BF1-BF10 једино одступа за BF6, где се не постижу исте вредности за сваку поновљену симулацију, али су резултати такође веома добри. Алгоритам GA има стабилан рад за проблеме BF3, BF4, BF5, BF8, BF9, BF10, док су решења нестабилна за BF1, BF2, BF6 и врло мало за BF7. За тест проблем BF11 оптимум постижу *GA, DE, ABC, TLBO, Jaya*, једино се јављају одступања код PSO алгоритма који има најлошије перформансе за овај проблем. Сви анализирани алгоритми из литературе за BF12 и BF13 постижу захтевани оптимум према дефинисаним подешавањима експеримента.

DINDI алгоритам за сваки од анализираних проблема постиже захтевани оптимум у сваком од понављања симулација. Алгоритам *iGA* показује веома добре карактеристике у поређењу са осталим методама. Проблематичне функције су BF1, BF2 и BF7 за овај алгоритам. Код BF1 и BF2 проблема једино се ради о броју анализираних децимала, док за проблем BF7 алгоритам показује благу слабост, али и даље даје изузетне резултате. За тип проблема без ограничења *iGA* у општем смислу даје неупоредиво боље резултате од *GA* из литературе, што је могуће третирати као веома успешну модификацију. Алгоритам *rTLBO* постиже оптимум за све анализиране проблеме. У односу на друга истраживања овде је узет у обзир већи број децимала, па ако се критички посматра овај алгоритам најлошије ради за проблеме BF1, BF2, BF6 и BF7. Неопходно је напоменути да рад сваке методе зависи од великог броја фактора, као што су начин имплементације, подешавање алгоритма, хардверска и софтверска подршка и многи други. Често је тешко у потпуности поновити оригинална истраживања, а посебно на основу саме доступне литературе. У том циљу могуће је потврдити да у неким случајевима рад метода или није баш онакав каквим се представља, што значи да постоји пропуст у самој презентацији резултата, или као чешћи случај постоји пропуст и превид у поступку понављања (имплементације) датог алгоритма. То су основни разлози зашто постоји разлика између алгоритма *TLBO* и *rTLBO* у резултатима, без обзира што се очекује бољи рад *rTLBO* алгоритма. Развијени хибрид *hGPT* даје нешто лошије, али и даље добре резултате. Слабости овог хибрида показане су у примерима BF1, BF2 и BF7.

Дозвољени максимални број *FEs* –а према препорукама је 500000, за шта се и очекује постизање оптималног резултата. Ако се упореде вредности FEs –а за алгоритме *DINDI, iGA, rTLBO* и *hGPT* за све анализиране проблеме, као минималне величине *FEs* у 30 поновљених симулација могу се анализирати ефикасности рада развијених метода. На слици 5.1 представљене су минималне вредности *FEs* –а за постизање оптимума.



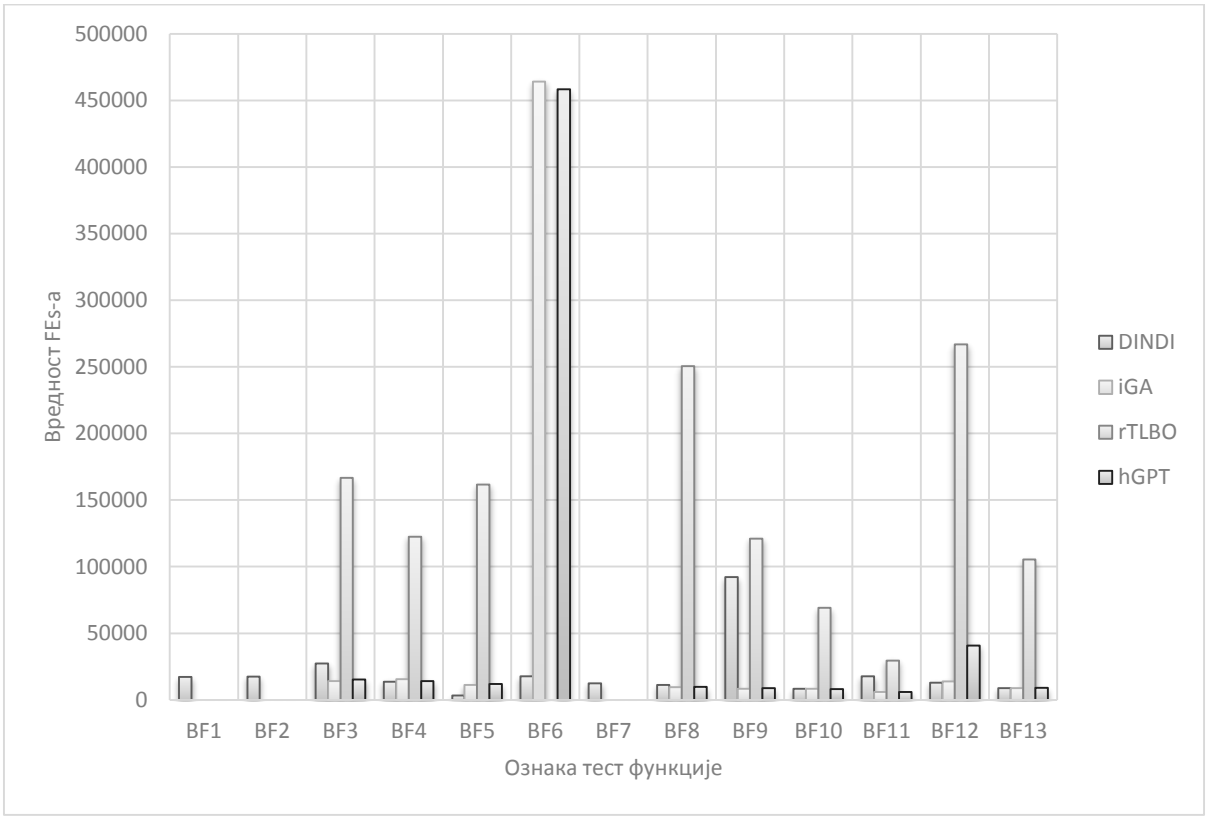
Слика 5.1. Минималне вредности *FEs* – а за постизање оптимума за тест функције BF1-BF13 за алгоритме *DINDI, iGA, rTLBO* и *hGPT*

За проблем BF7, *iGA* и *hGPT* никада не постигну апсолутни оптимум, тако да су ове вредности представљене као 0 на слици 5.1, а њихова стварна вредност би износила нешто преко дозвољене вредности од 500000.

DINDI алгоритам је просечан по питању захтева броја прерачунавања функције циља. За овај развијени алгоритам *FEs* се израчунава као $(2 \times \text{величина популације} \times \text{број генерација})$, док се за све остале алгоритме ова вредност израчунава као $(\text{величина популације} \times \text{број генерација})$. По питању величине *FEs* – а *DINDI* алгоритам није најбољи, али није ни најлошији. Брзо постизање решења може значити превремену конвергенцију и зато може бити неповољно за процес оптимизације и сама решења. Важно је да су за анализиране проблеме решења стабилна. За сам процес оптимизације су важнија максимална решења *FEs* – а са којима ће алгоритам свакако, сваким узастопним покретањем постићи оптимално решење. Анализа ове величине представљена је на слици 5.2, на којој су приказане вредности за сва четири развијена алгоритма. На слици 5.2 је приказано да *DINDI* алгоритам за сваки од анализираних проблема сваким покретањем постиже оптималну вредност. То указује да су резултати стабилни и да овај алгоритам има изузетне перформансе за проблеме без ограничења.

Модификација генетског алгоритма, *iGA* опште гледано има бржу конвергенцију у односу на *DINDI* алгоритам, осим за проблеме BF3, BF4, BF7. За пример BF7, овај алгоритам не постиже оптимум чак ни са вредношћу од 500000 *FEs* – а, али је вредност веома блиска оптималној. Овај алгоритам има поједине слабости које су већ поменуте, али генерално гледано и у поређењу са *GA* перформансе овог алгоритма су изузетне. Што се тиче максималне вредности *FEs* – а са којом овај алгоритам свакако постиже оптимум за свако поновно покретање алгоритма, за проблеме BF1, BF2 и BF7 та је вредност преко 500000, па зато није приказана на слици 5.2. За проблем BF6 овај алгоритам је такође веома захтеван, али не прелази критичну границу од 500000. Код

осталих проблема *iGA* се показао као веома ефикасан и мало захтеван по питању прерачунавања функције циља.



Слика 5.2. Максималне вредности *FEs* –а за постизање оптимума за тест функције BF1-BF13 за алгоритме *DINDI, iGA, rTLBO* и *hGPT* при сваком поновном покретању

Алгоритам *rTLBO* за решавање свих 13 тест проблема захтева највише прерачунавања функције, али има прилично стабилне резултате. Минималне вредности *FEs* – а (слика 5.1) су високе, што указује да алгоритам није много ефикасан, али стабилност резултата компензује ову слабост. Што се тиче максималне вредности *FEs* – а за *rTLBO* оне су и даље највеће у односу на остале алгоритме. За примере BF1, BF2, BF6 и BF7 ова вредност није приказана на слици 5.2, јер алгоритам не постиже оптимум сваким покретањем за 500000 FEs – а. Вредности које постиже овај алгоритам за критичне проблеме су веома блиске оптимуму, зато их је могуће третирати као оптималне. Зато је могуће рећи да је максимална вредност *FEs* – а за *rTLBO* алгоритам за примере BF1, BF2, BF6 и BF7 око 500000.

Алгоритам *hGPT* за вредности *FEs* – а има сличне вредности као *iGA*, али су резултати које постиже овај хибрид лошији у односу на *iGA*. Вредности које нису представљене на слици 5.1 и слици 5.2 прелазе вредност од 500000. Овај хибрид се показао као најлошији за решавање проблема без ограничења у односу на остале анализиране алгоритме. За анализирану групу проблема најбоље резултате даје *DINDI* алгоритам, затим *rTLBO,* који је захтеван по питању величине *FEs* – а, затим *iGA*, као веома ефикасан и на крају *hGPT*.

5.2 Тестирање метода за решавање оптимизационих проблема са ограничењима

Експериментална испитивања важна за дефинисање ефикасности рада хеуристичких метода извршена су за групу тест проблема са ограничењима. Ови проблеми су веома комплексни за решавање и помажу да се презентује перспектива примене метода на практичне проблеме. Извршено је тестирање *DINDI* алгоритма и алгоритама *iGA, rTLBO* и *hGPT*, а резултати су упоређени са резултатима из литературе за методе *GA, PSO, DE, ABC, BBO, TLBO, Jaya* и *HTS*. Експериментални резултати представљени су у табелама 5.4.1 и 5.4.2. Услови спровођења експеримената јасно су дефинисани у литератури из ове области и захтев је да хеуристичка метода постигне решење за величину *FEs* – а од 240000 за све тест функције и да се спроведе 100 узастопних покретања алгоритма. Сваки од експеримената из литературе спроведен је на поменути начин, а то пружа могућност компарације метода. Као интересантне вредности за ову анализу узете су: најбоља, најлошија и средња вредност постигнуте у 100 покретања. Све анализиране функције математички су формулисане у Додатку А. (Математичка формулација коришћених тест функција са ограничењима). Основне карактеристике тест функција са ограничењима представљене су у табели 5.3, где су дефинисани тип функције циља, тип и број ограничења (*LI* – број линеарних ограничења у облику неједначине, *NI* - број нелинеарних ограничења у облику неједначине, *LE* - број линеарних ограничења у облику једначине, *NE* - број нелинеарних ограничења у облику једначине), број променљивих проблема (*n*). Процењени однос између домена (*F*) и поља претраге (*S*) је означен са ρ (*F/S*) и број активних ограничења оптимизационог проблема (*ac*).

Табела 5.3
Карактеристике тест функција са ограничењима

Ознака	n	Тип функције	ρ (%)	LI	NI	LE	NE	ac	Оптимум
CB1	13	Квадратна	0,0111%	9	0	0	0	6	-15
CB2	10	Полиномска	0,0000%	0	0	0	1	1	-1,0005
CB3	5	Квадратна	52,1230%	0	6	0	0	2	-30665,5
CB4	8	Линеарна	0,0010%	3	3	0	0	6	7049,248
CB5	2	Кубна	0,0066%	0	2	0	0	2	-6961,81
CB6	2	Нелинеарна	0,8560%	0	2	0	0	0	-0,09582
CB7	7	Полиномска	0,5121%	0	4	0	0	2	680,63
CB8	2	Квадратна	0,0000%	0	0	0	1	1	0,7499
CB9	2	Линеарна	76,6556%	0	2	0	0	2	-5,50801

Приметно је у табелама 5.4.1 и 5.4.2 да сваки од анализираних алгоритама за тест функције са ограничењима BC1-BC9 за 240000 *FEs* – а даје различите резултате. Разноликост резултата много више указује на квалитет алгоритама за решавање неког типа проблема, него на саме недостатке алгоритама. Из презентованих резултата могуће је закључити да сваки од алгоритама има своје предности, али и недостатке у погледу ефикасности и перформанси, али је очигледно да модерни хеуристички алгоритми доносе боље перформансе, са бржим и квалитетнијим радом у погледу стабилности резултата. Резултати експерименталних испитивања за *DINDI* алгоритам и алгоритме *iGA, rTLBO* и *hGPT* упоређени су са резултатима из литературе [30, 32, 152-154]. За тест проблем BC1, алгоритми *ABC, TLBO, Jaya* и *HTS* постижу најбоље познате резултате за

овај проблем према условима експеримента. Алгоритми *PSO* и *DE* у неким од симулација постижу оптимум, али не у свих 100 поновљених симулација. *BBO* алгоритам за задата подешавања никада не постиже најбољу вредност, али постигнута вредност је блиска очекиваној. Најлошије резултате према литератури постиже *GA*. Све вредности које нису анализиране означене су са Н/А. Алгоритми *BBO, TLBO, Jaya* и *HTS* дају изузетне резултате за пример СВ2. *TLBO, Jaya* и *HTS* су најмодернији нови хеуристички алгоритми, пажљиво креирани за решавање комплексних проблема, за ефикасан и стабилан рад, што потврђују и резултати. Нешто лошији рад код овог проблема имају *ABC* и *PSO*. Најлошије резултате дају методе *GA* и *DE*. За проблем СВ3 оптимум постижу *TLBO,* Jaya и *HTS*. Са нешто нижом тачношћу, решења постижу и *PSO, DE* и *ABC*. Такође оптимум постиже и *BBO*, али не за свако понављање симулације, док најлошије перформансе поново има *GA.* Специфичан проблем је СВ4 за који ни једна од анализираних метода из литературе не постиже познати оптимум. Свакако добре перформансе имају *Jaya* и *HTS*, затим *TLBO* који је нешто лошији, *PSO, DE* и *ABC* који су додатно лошији, затим *BBO* и поново *GA* има најлошије перформансе, јер најбоље постигнуто решење у великој мери одступа од стварног оптимума. Проблем СВ5 успешно решавају *PSO, TLBO, Jaya* и *HTS* алгоритми. За нијансу лошије перформансе има *ABC*, додатно лошије *BBO*, док су најлошије *GA* и *DE*. Методе *PSO, DE, ABC, TLBO, Jaya* и *HTS* за проблем СВ6 постижу очекиване резултате, док лошије карактеристике имају *BBO* и *GA.* За проблем СВ7 једино *GA* не постиже оптимум, чак су и решења прилично стабилна за све остале алгоритме. Алгоритми *PSO, TLBO, Jaya* и *HTS* постижу оптимум за проблем СВ8 за свих 100 понављања, чак и *ABC* и *GA* алгоритми имају исте карактеристике са нешто нижом тачношћу решења. Алгоритам *BBO* такође постиже оптимум за овај проблем, али не за свако понављање симулације, док *DE* има нешто лошије вредности. За проблем СВ9 све анализиране методе постижу оптималне резулате, док стабилност резултата једино нема *BBO*. Алгоритам *GA* није анализиран за овај проблем.

Чињеница је да *GA* представља хеуристичку методу која веома дуго траје, има много модификација и представљена је као метода са изузетним карактеристикама, што је наравно и тачно. То што код анализираних проблема показује најлошије перформансе потврђује велики напредак при развоју хеуристичких метода оптимизације и комплексност проблема према којима се врши тестирање алгоритама. Постићи решење које је уопште блиско оптимуму код овако комплексних проблема представља изузетан резултат.

Табела 5.4.1

Упоредни резултати CB1-CB9 тест функција са ограничењима за анализиране алгоритме (за 240000 *FEs* – а и 100 узастопно поновљених симулација)

		GA	PSO	DE	ABC	BBO	TLBO	Jaya	HTS
	Најбоља	-14,44	-15	-15	-15	-14,977	-15	-15	-15
CB1	Најлошија	H/A	-13	-11,828	-15	-14,5882	-15	-15	-15
	Средња	-14,236	-14,71	-14,555	-15	-14,4698	-15	-15	-15
	Најбоља	-0,99	-1	-0,99393	-1	-1,0005	-1,0005	-1,0005	-1,0005
CB2	Најлошија	H/A	-0,464	-1	-1	-0,0455	-1	-1	0
	Средња	-0,976	-0,7648	-1	-1	-0,3957	-1	-1	-0,9004
	Најбоља	-30626,053	-30665,539	-30665,539	-30665,539	-30665,539	-30665,5387	-30665,5387	-30665,5387
CB3	Најлошија	H/A	-30665,539	-30665,539	-30665,539	-29942,3	-30665,5387	-30665,5387	-30665,5387
	Средња	-30590,455	-30665,539	-30665,539	-30665,539	-30411,865	-30665,5387	-30665,5387	-30665,5387
	Најбоља	9079,77	7049,481	7049,548	7053,904	7679,0681	7052,329	7049,312	7049,4836
CB4	Најлошија	H/A	7894,812	9264,886	7604,132	9570,5714	7152,0813	7119,632	7252,0546
	Средња	10003,225	7205,5	7147,334	7224,407	8764,9864	7114,4893	7104,6201	7119,7015
	Најбоља	-6952,472	-6961,814	-6954,434	-6961,814	-6961,8139	-6961,814	-6961,814	-6961,814
CB5	Најлошија	H/A	-6961,814	-6954,434	-6961,805	-5404,4941	-6961,814	-6961,814	-6961,814
	Средња	-6872,204	-6961,814	-6954,434	-6961,813	-6181,7461	-6961,814	-6961,814	-6961,814
	Најбоља	-0,095825	-0,095825	-0,095825	-0,095825	-0,095825	-0,095825	-0,095825	-0,095825
CB6	Најлошија	H/A	-0,095825	-0,095825	-0,095825	-0,095817	-0,095825	-0,095825	-0,095825
	Средња	-0,095799	-0,095825	-0,095825	-0,095825	-0,95824	-0,095825	-0,095825	-0,095825
	Најбоља	685,994	680,63	680,63	680,634	680,6301	680,6301	680,6301	680,6301
CB7	Најлошија	H/A	680,631	680,63	680,634	721,0795	680,6334	680,6301	680,644
	Средња	692,064	680,63	680,63	680,634	692,7162	680,6313	680,6301	680,6329
	Најбоља	0,75	0,749	0,752	0,75	0,7499	0,7499	0,7499	0,7499
CB8	Најлошија	H/A	0,749	1	0,75	0,92895	0,7499	0,7499	0,7499
	Средња	0,75	0,749	0,901	0,75	0,83057	0,7499	0,7499	0,7499
	Најбоља	H/A	-5,5080	-5,5080	-5,5080	-5,5080	-5,5080	-5,5080	-5,5080
CB9	Најлошија	H/A	-5,5080	-5,5080	-5,5080	-5,4857	-5,5080	-5,5080	-5,5080
	Средња	H/A	-5,5080	-5,5080	-5,5080	-5,4982	-5,5080	-5,5080	-5,5080

Табела 5.4.2

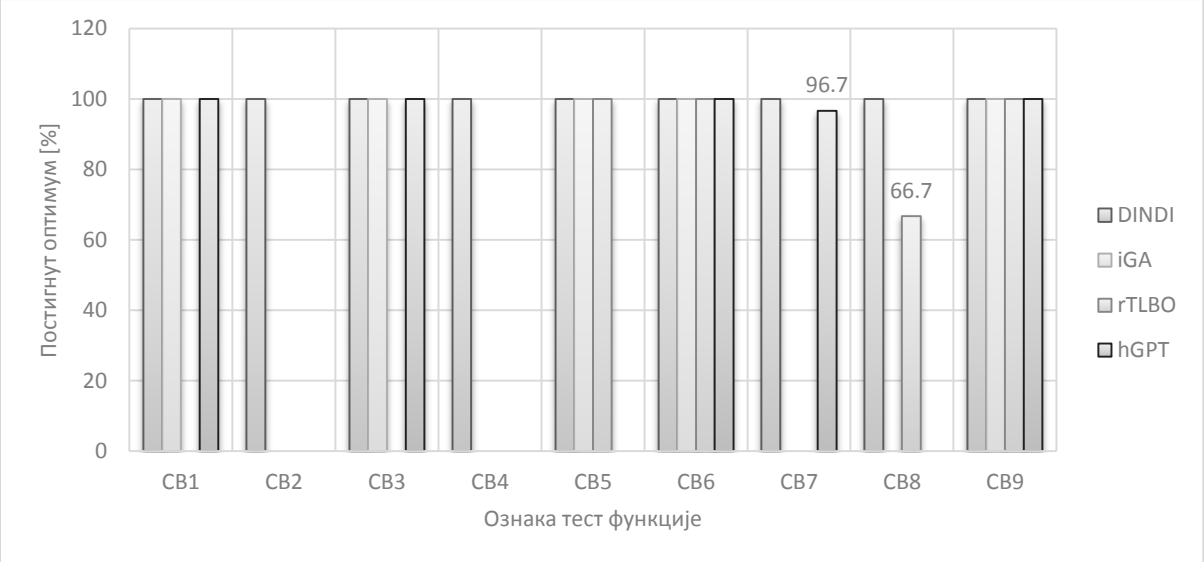
Упоредни резултати CB1-CB9 тест функција са ограничењима за анализиране алгоритме (за 240000 *FEs* – а и 100 узастопно поновљених симулација)

		DINDI	**iGA**	**rTLBO**	**hGPT**
CB1	Најбоља	-15	-15	-14,999843	-15
	Најлошија	-15	-15	-14,999107	-15
	Средња	-15	-15	-14,999592	-15
CB2	Најбоља	-1,0005001	H/A	H/A	H/A
	Најлошија	-1,0005001	H/A	H/A	H/A
	Средња	-1,0005001	H/A	H/A	H/A
CB3	Најбоља	-30665,5386718	-30665,538672	-30665,538668	-30665,538672
	Најлошија	-30665,5386718	-30665,538672	-30665,538649	-30665,538672
	Средња	-30665,5386718	-30665,538672	-30665,538658	-30665,538672
CB4	Најбоља	7049,2480205	7218,713049	7112,905607	7052,57824
	Најлошија	7049,2480205	7547,505666	7295,817595	7446,721949
	Средња	7049,2480205	7389,671077	7213,067662	7237,498634
CB5	Најбоља	-6961,8138756	-6961,813876	-6961,813876	H/A
	Најлошија	-6961,8138756	-6961,813875	-6961,813875	H/A
	Средња	-6961,8138756	-6961,813876	-6961,813875	H/A
CB6	Најбоља	-0,095825041	-0,095825	-0,095825	-0,095825
	Најлошија	-0,095825041	-0,095825	-0,095825	-0,095825
	Средња	-0,095825041	-0,095825	-0,095825	-0,095825
CB7	Најбоља	680,6300574	680,640616	680,638070	680,630057
	Најлошија	680,6300574	680,669094	680,761806	680,630311
	Средња	680,6300574	680,654669	680,672343	680,630066
CB8	Најбоља	0,7499	H/A	0,7499	H/A
	Најлошија	0,7499	H/A	0,749993	H/A
	Средња	0,7499	H/A	0,749923	H/A
CB9	Најбоља	-5,508013272	-5,508013	-5,508013	-5,508013
	Најлошија	-5,508013272	-5,508013	-5,508013	-5,508013
	Средња	-5,508013272	-5,508013	-5,508013	-5,508013

Резултати из табеле 5.4.2 презентују решења за примере CB1- CB9 за *DINDI, iGA, rTLBO* и *hGPT*. За сваки од проблема, сваком од понављања, према дефинисаним условима експеримента *DINDI* алгоритам постиже оптимум. Ова чињеница издваја развијени алгоритам у односу на све друге методе и указује на изузетне перформансе, ефикасност и флексибилност ове методе.

За проблем CB2, *iGA, rTLBO* и *hGPT* нису анализирани, док за проблем CB8 нису анализирани *iGA* и *hGPT*. За проблем CB5 није анализиран само *hGPT* алгоритам. Разлог за то јесте специфичност проблема. Проблеми су изузетно захтевни и очекује се проналажење решења за уско поље претраге. Овај захтев одражава се на креирање иницијалне популације. За *iGA, rTLBO* и *hGPT* због начина имплементације креирање иницијалне популације може да траје веома дуго, због чега су изостављени поменути проблеми. То не умањује квалитет ових алгоритама, већ су само технички разлози изостављања решења за ове проблеме.

Своје слабости *iGA* показује највише код проблема CB4 и CB7, док за све остале анализиране проблеме овај алгоритам постиже изузетне резултате. И слабости које има алгоритам нису драстичне, тако да је могуће потврдити квалитет ове методе. У поређењу са *GA*, ова модификација представља значајно побољшање. Метода *rTLBO* слабости показује на проблемима CB1, CB4 и CB7. Опште гледано, резултати ове методе су веома добри, а модификација има другачији рад у односу на *TLBO*, можда подједнаких квалитета. Хибрид *hGPT* своје квалитете показује на анализираним проблемима. Ова метода има бољи рад у поређењу са *iGA* и *rTLBO*.



Слика 5.3. Успешност метода за решавање тест функција са ограничењима при 240000 *FEs* – а

Од укупно 100 узастопно поновљених покретања алгоритама, није сваки пут могуће постићи оптимум. На слици 5.3 представљена је зависност процентуалног постизања оптимума у односу на методу према анализираном проблему. Методе које нису анализиране за неки од проблема представљене су као да немају вредност. Општи

закључак је да *DINDI* алгоритам на овој групи проблема нема конкуренцију. За сваки од проблема са 100% успешности постиже оптимум, што није могуће рећи ни за једну другу методу.

5.3 Тестирање метода за решавање инжењерских оптимизационих проблема

Због потребе да се развијене хеуристичке методе примене на проблеме оптимизације машинских конструкција, неопходно је извршити верификацију рада метода на инжењерским практичним проблемима – проблемима машинског конструисања. У овом делу истраживања извршена су експериментална испитивања за пет различитих, познатих инжењерских проблема. Карактеристике ових проблема представљене су у табели 5.5, а детаљна математичка формулација ових проблема представљена је у Додатку В (Математичка формулација инжењерских проблема оптимизације). Појединачно, за сваки од проблема, дефинисана је величина *FEs* – а за *DINDI* алгоритам, тако да постигнута решења за 30 узастопно поновљених симулација увек буду оптимална, тачније речено, алгоритам постиже сваким покретањем исту вредност. За остале развијене алгоритме вредност *FEs* – а узета је као фиксна и износи 20000 за све проблеме, осим за проблем оптимизације редуктора, где ове методе показују предности које поседују. Захтев литературе је да се испитивање врши за 25 понављања, али је у овом истраживању повећана та вредност и проблеми су анализирани за 30 понављања. За проблеме заварене греде и суда под притиском циљ је минимизација трошкова, док је за проблеме опруге и редуктора циљ минимизација масе. За проблем носача са три штапа циљ је минимизација запремине. Сви поменути проблеми су проблеми машинског конструисања, а то помаже у одређивању стварних карактеристика развијених метода и њихову практичну примену на ову групу проблема.

Табела 5.5

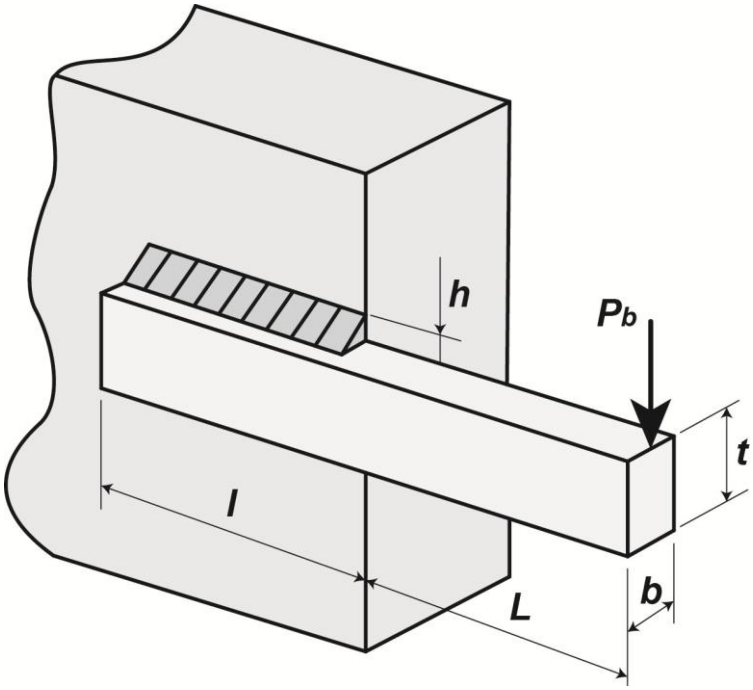
Карактеристике тест инжењерских проблема (машинских конструкција)

Проблем	Променљиве	Континуал. пром.	Дискрет. пром.	Ограничења	Активна огр.	F/S*	Циљ
Заварена греда	4	4	0	7	2	0,035	Минимизација трошкова
Опруга	3	3	0	4	2	0,01	Минимизација масе
Редуктор	7	6	1	11	3	0,004	Минимизација масе
Суд под притиском	4	4	0	4	2	0,4	Минимизација трошкова
Носач са три штапа	2	2	0	3	3	Н/А	Минимизација запремине

* Процењени однос између домена (F) и поља претраге (S)

5.3.1 Заварена греда

Инжењерски оптимизациони проблем заварене греде представља проблем који може помоћи у одређивању перформанси и позиционирању метода у њиховом раду. Овај проблем развио је Коело (*Coello*) [155], а он је постао стандардни тест проблем за испитивање метода погодних за решавање инжењерских проблема. Карактеристике овог проблема представљене су у табели 5.5, а шематски приказ проблема представљен је на слици 5.4.



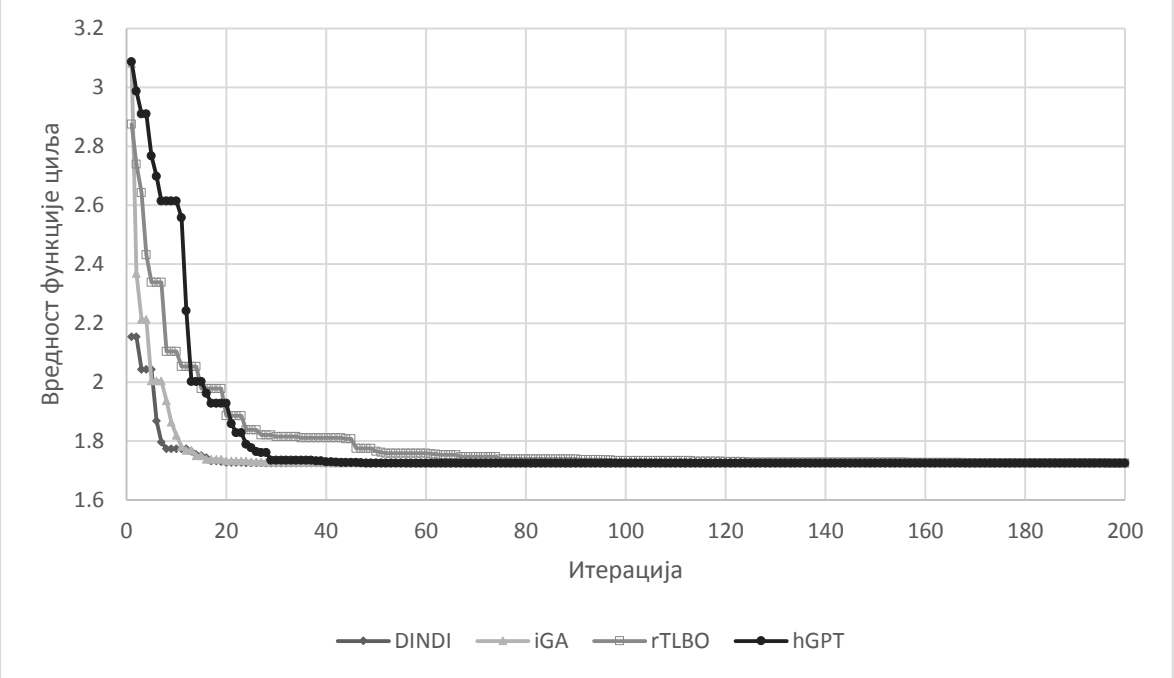
Слика 5.4. Шематски приказ проблема заварене греде

Проблем заварене греде детаљно је формулисан у Додатку B.1. (Математичка формулација инжењерских проблема оптимизације), где је очигледно да се оптимизација врши на основу димензија са слике 5.4. За развијене методе извршено је 30 узастопних понављања симулација са истим подешавањима, а добијене вредности упоређене су са вредностима из литературе са нижом тачношћу, тачније са захтевом од 25 узастопних понављања. У табели 5.6 представљене су упоредне вредности резултата добијених експериментом и резултата из литературе. У табели су представљени статистички подаци и за све методе анализиране су најбоља, најлошија, средња вредност и величина *FEs* – а. Према литератури, очекивана вредност функције циља је $f(X) = 1{,}724852.$

Табела 5.6
Поређење резултата анализираних оптимизационих метода за оптимизациони проблем заварене греде

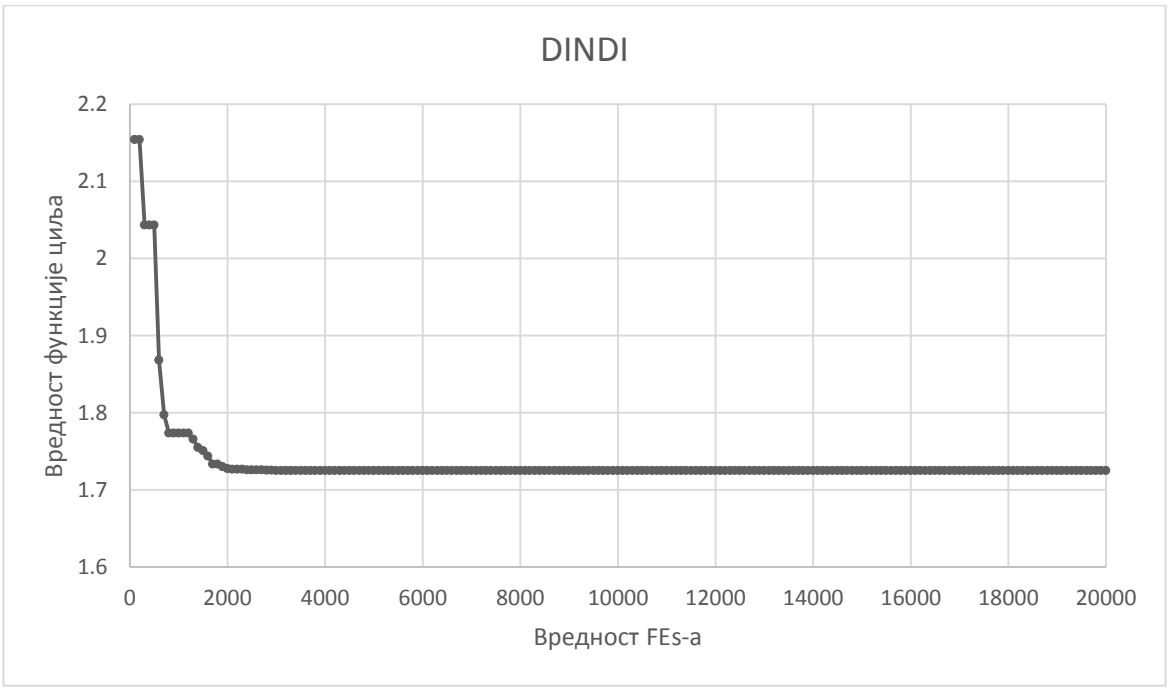
Метода	Најбоља	Најлошија	Средња	FEs
GA (Matlab GA-toolbox) [31]	2,026769	3,162137	2,76033	25000
PSO-DE [121]	1,724852	1,724852	1,724852	66600
TLBO [33]	1,724852	H/A	1,728447	20000
ABC [156]	1,724852	H/A	1,741913	30000
CIV-PSO [122]	1,724852	1,727665	1,725124	25000
BA [25]	1,7312	2,345579	1,878656	20000
MBA [24]	1,724853	1,724853	1,724853	47340
FFA [157]	1,724852	1,724852	1,724852	50000
WCA 1 [27]	1,724856	1,744697	1,726427	46450
WCA 2 [27]	1,724857	1,801127	1,73594	30000
PVS 1 [31]	1,724852	1,724852	1,724852	500000
PVS 2 [31]	1,724852	1,725056	1,724887	20000
DINDI	1,7248523	1,7248523	1,7248523	10000
iGA	1,724852	2,134913	1,753641	20000
rTLBO	1,724891	1,725307	1,725014	20000
hGPT	1,724852	2,372224	1,800219	20000

Методе анализиране у литератури за проблем заварене греде су *GA* (*Matlab GA-toolbox*), *PSO-DE*, *TLBO*, *ABC*, *CIV-PSO*, *BA*, *MBA*, *FFA*, *WCA* и *PVS*. Величина *FEs* – а креће се у интервалу од 10000 до 500000, док је постигнути опсег најбољих вредности од 1,724852 до 2,026769. *DINDI* алгоритам за овај проблем показује убедљиво најбоље перформансе, јер са само 10000 *FEs* – а сваким покретањем постиже оптимум, што говоре најлошија и средња вредност за овај алгоритам из табеле. *iGA* постиже очекивани оптимум, али стабилност резултата за постављене резултате није добра. Стабилнији је *rTLBO*, али постиже нешто нижу тачност оптимума (не постиже очекивани оптимум). Хибрид *hGPT* постиже очекивани оптимум, али су средња и минимална вредност још лошије у односу на *iGA* алгоритам. Конвергенција развијених алгоритама за проблем заварене греде представљена је на слици 5.5.



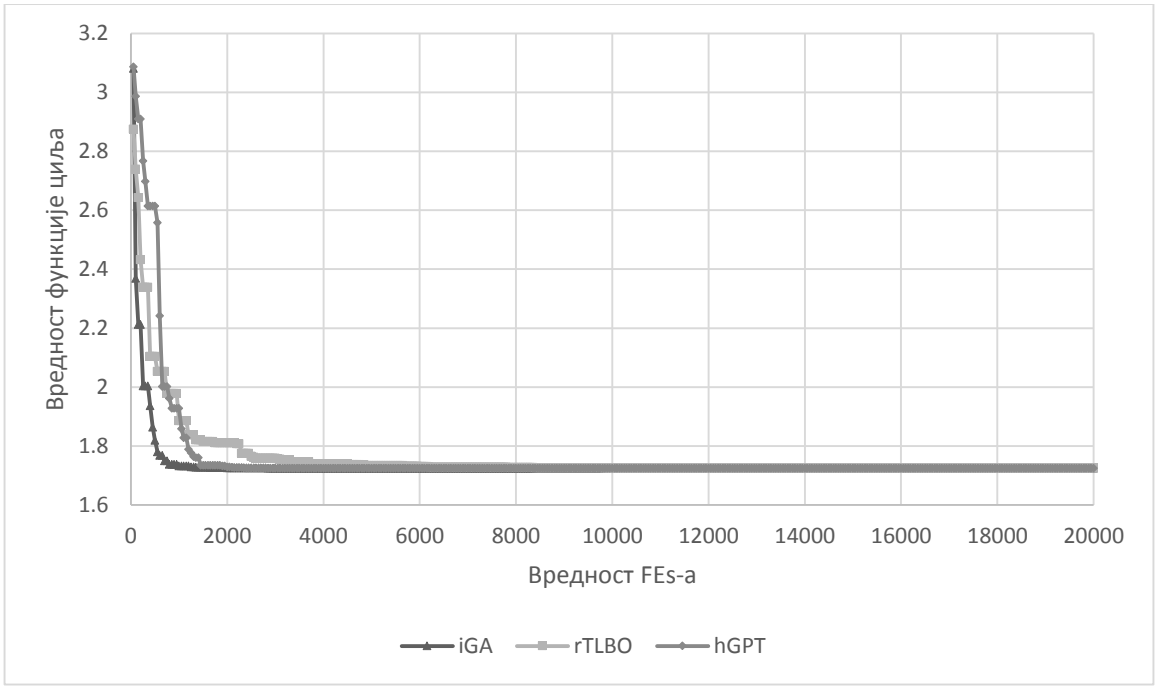
Слика 5.5. Конвергенција развијених алгоритама за проблем заварене греде

На слици 5.5 конвергенција је представљена као зависност вредности функције циља и броја итерација за анализиране алгоритме. Да би се анализирао ток оптимизације потребно је посматрати зависност величине *FEs* – а и вредности функције циља што је и представљено на слици 5.6 а) и слици 5.6 б).



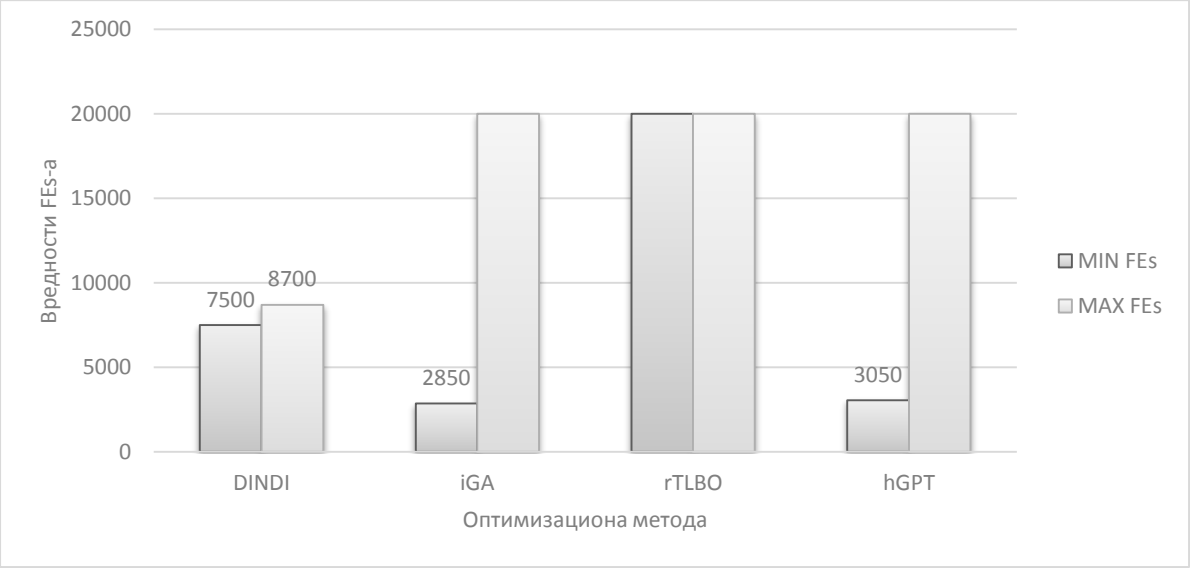
а)

б)



Слика 5.6. Зависност вредности функције циља и вредности *FEs* – а; а) Конвергенција за *DINDI* алгоритам; б) Конвергенција за *iGA*, *rTLBO* и *hGPT* алгоритме

На слици 5.6 вредности су цињно одвојено приказане за *DINDI* и друге алгоритме, јер се разликује прерачунавање вредности *FEs* – а. За приказ конвергенције узете су случајне симулације из скупа од 30 узастопних покретања. Када се посматрају све симулације, најмања вредност за *FEs* и највећа вредност *FEs* – а за одређену методу који је коришћен за постизање приказаног оптимума, представљене су на слици 5.7 за анализиране методе.

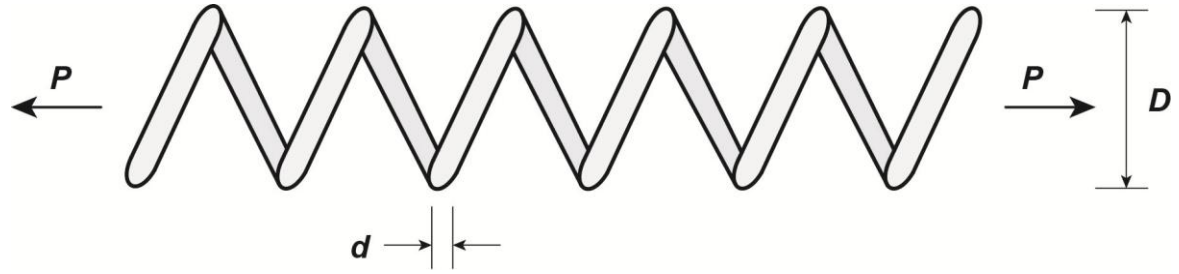


Слика 5.7. Минимална и максимална вредност *FEs* – а анализираних метода за проблем заварене греде

Са слике 5.7 могуће је уочити да најпре *iGA* и *hGPT* могу постићи оптимум, али постоје тренуци када се то уопште не догоди. Алгоритам *rTLBO* никада не постиже оптимум, док *DINDI* алгоритам увек постигне оптимум за сличне вредности *FEs* – а. *DINDI* алгоритам је доминантан код овог оптимизационог проблема, али и *iGA*, *rTLBO* и *hGPT* алгоритми показују веома добре карактеристике. Повећавањем вредности *FEs* – а било би могуће и код ових метода сваким понављањем постићи постизање оптимума.

5.3.2 Опруга

На слици 5.8 представљен је шематски приказ другог анализираног инжењерског проблема. Овај проблем формулисао је Арора (*Arora*) [158], а формулација овог проблема представљена је у Додатку Б.2. Карактеристике овог проблема представљене су у табели 5.5. Промењљиве оптимизације дефинишу димензије опруге.



Слика 5.8. Шематски приказ опруге за оптимизацију

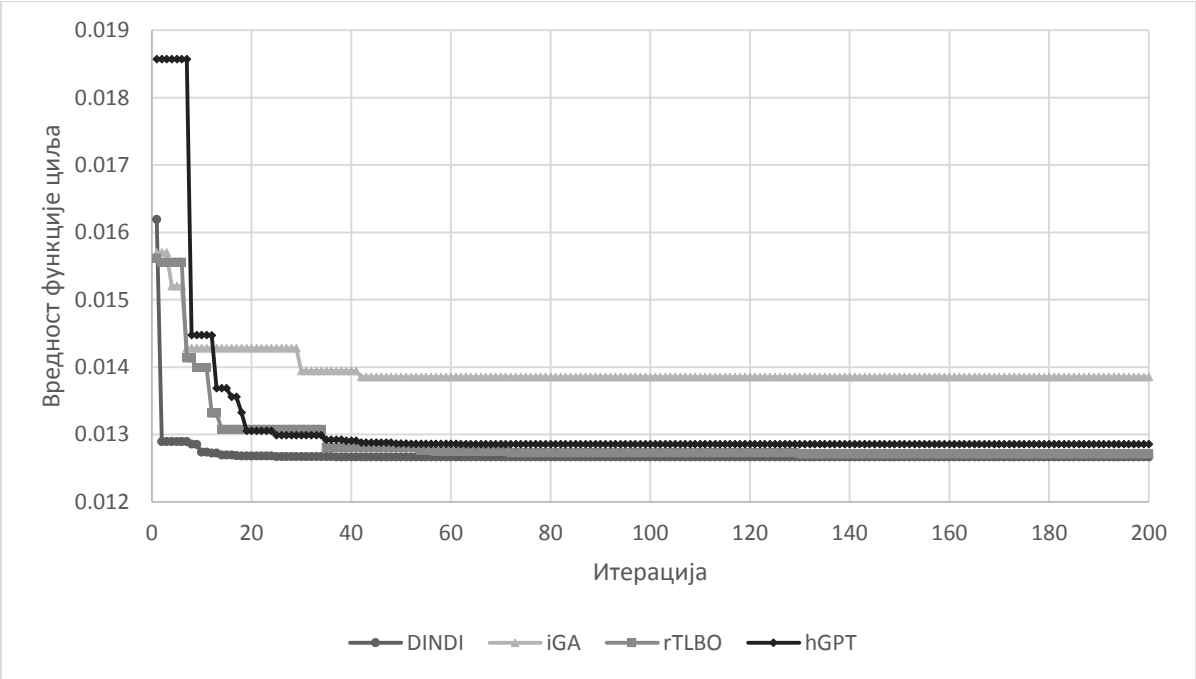
Овај оптимизациони проблем формулисан је са четири континуалне промењљиве, два линеарна и пет нелинеарних ограничења у виду неједначина. Величине представљене на слици 5.8 фигуришу у математичком моделу. Извршена су експериментална испитивања и у табели 5.7 представљене су добијене вредности и вредности алгоритама из литературе. Очекивана вредност функције циља за овај проблем је $f(X) = 0{,}012665$.

Табела 5.7
Поређење резултата анализираних оптимизационих метода за оптимизациони проблем опруге

Метода	Најбоља	Најлошија	Средња	FEs
GA (Matlab GA-toolbox) [31]	0,012671	0,012693	0,012683	25000
CPSO [27]	0,012674	0,012924	0,012730	240000
PSO [121]	0,012857	0,071802	0,019555	2000
HEAA [159]	0,012665	0,012665	0,012665	24000
PSO-DE [121]	0,012665	0,012665	0,012665	42100
TLBO [33]	0,012665	H/A	0,012666	20000
ABC [156]	0,012665	H/A	0,012709	30000
CVI-PSO [122]	0,012666	0,012843	0,012731	25000
BA [25]	0,012665	0,016895	0,013501	20000
MBA [24]	0,012665	0,012900	0,012713	7650
ISA [160]	0,012665	0,013165	0,012799	8000
FFA [157]	0,012665	0,000013	0,012677	50000
WCA 1 [27]	0,012665	0,012952	0,012746	11750

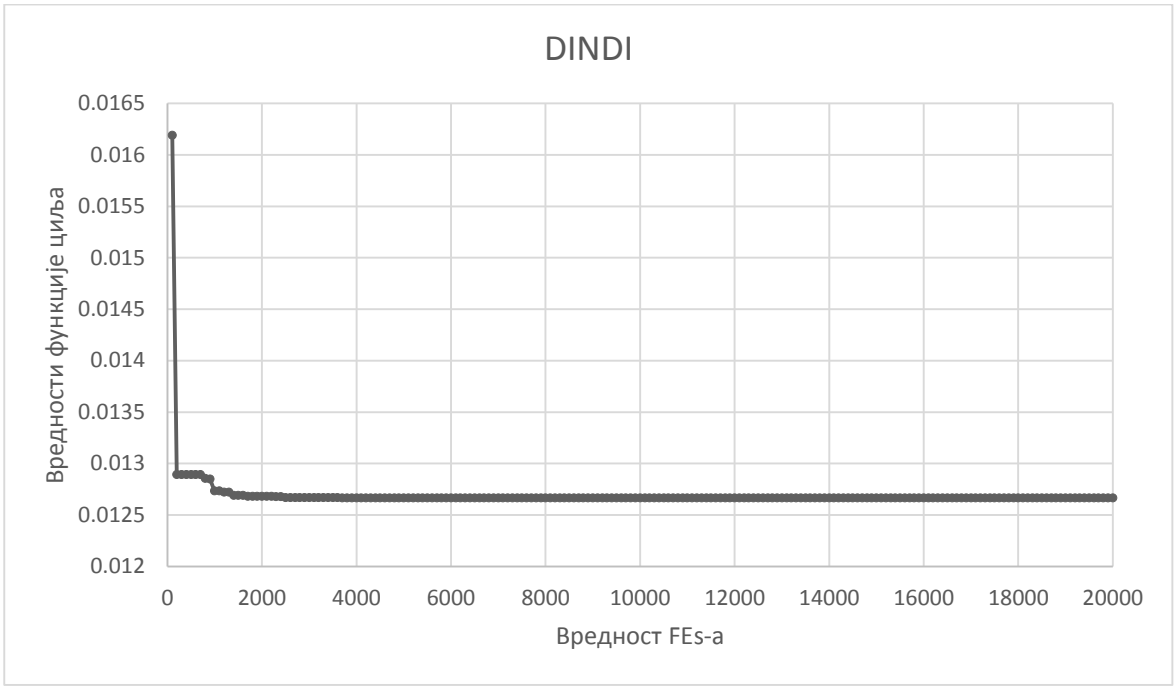
WCA 2 [27]	0,012665	0,015021	0,013013	2000
PVS 1 [31]	0,012665	0,012665	0,012665	42100
PVS 2 [31]	0,012665	0,012667	0,012666	20000
PVS 3 [31]	0,012665	0,012710	0,012670	8000
PVS 4 [31]	0,012680	0,013141	0,012838	2000
DINDI	0,0126652	0,0126652	0,0126652	14000
iGA	0,012784	0,016049	0,014155	20000
rTLBO	0,012668	0,012693	0,012679	20000
hGPT	0,012665	0,013524	0,01277	20000

Вредности *FEs* – а које се срећу у литератури за овај проблем крећу се у интервалу од 2000 до 240000, где са 2000 није постигнут оптимум. Методе које су анализиране за овај проблем су *GA* (*Matlab GA-toolbox*), *CPSO*, *PSO*, *HEAA*, *PSO-DE*, *TLBO*, *ABC*, *CVI-PSO*, *BA*, *MBA*, *ISA*, *FFA*, *WCA* и *PVS*. За *DINDI* алгоритам вредност *FEs* – а са којом овај алгоритам сваким покретањем постиже оптимум је 14000, док за алгоритме *iGA*, *rTLBO* и *hGPT* вредност је ограничена на 20000. У литератури се захтева анализа за 25 узастопних покретања алгоритама, док је за потребе овог истраживања повећана тачност на 30 узастопних покретања алгоритама. *DINDI* постиже оптималну вредност $f(X) = 0{,}0126652$ при сваком поновном покретању. Ове вредности постижу и методе из литературе, али са неупоредиво већом величином *FEs* – а. Алгоритам *iGA* постиже блиску вредност оптималној, али резултати нису много стабилни. Стабилнија решења постиже *rTLBO*, где је најбоља постигнута вредност веома блиска оптималној. Оптимум постиже *hGPT* са веома стабилним решењима за свих 30 поновних покретања алгоритма. Конвергенција алгоритама *DINDI*, *iGA*, *rTLBO* и *hGPT* за проблем опруге представљена је на слици 5.9. На слици је конвергенција представљена као зависност вредности функције циља и броја итерација за анализиране алгоритме.

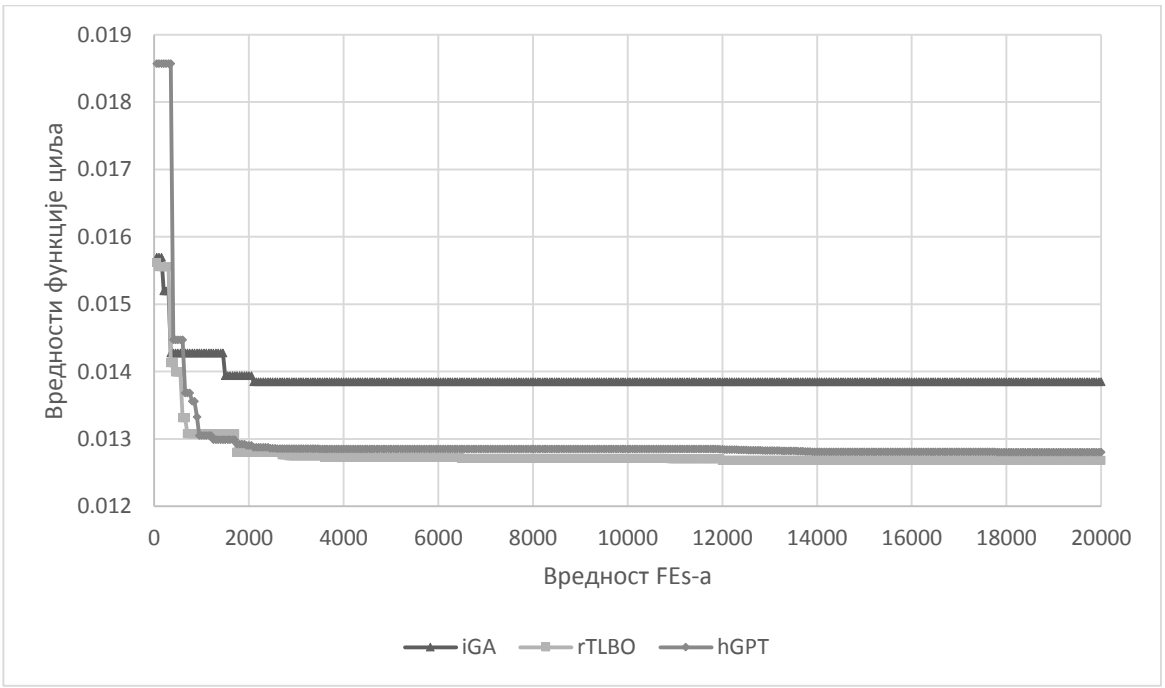


Слика 5.9. Конвергенција развијених алгоритама за проблем опруге

На слици 5.10 а) и слици 5.10 б) представљена је зависност величине *FEs* – а и вредности функције циља. Ова зависност је веома важна за анализу тока процеса оптимизације.



а)

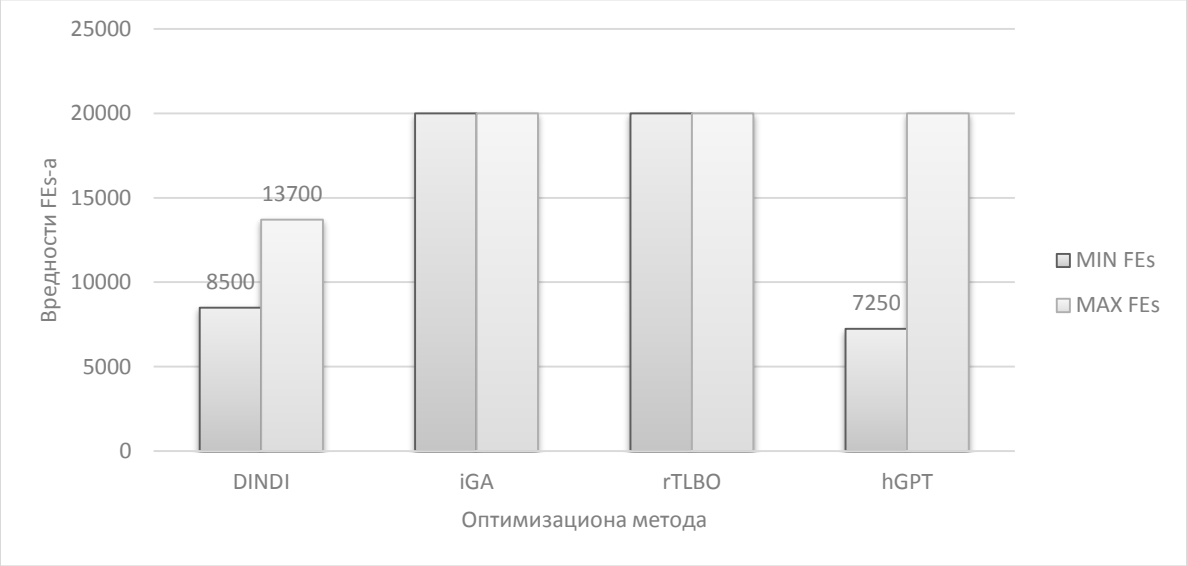


б)

Слика 5.10. Зависност вредности функције циља и вредности *FEs* – а; а) Конвергенција за *DINDI* алгоритам; б) Конвергенција за *iGA*, *rTLBO* и *hGPT* алгоритме

На слици 5.10 за *DINDI* и друге алгоритме вредности су циљно одвојено приказане, јер се разликује прерачунавање вредности *FEs* – а. За приказ конвергенције узете су случајне симулације из скупа од 30 узастопних покретања. Када се посматрају све симулације, најмања вредност за *FEs* и највећа вредност *FEs* – а, за одређену методу,

који је коришћен за постизање приказаног оптимума, представљене су на слици 5.11 за анализиране методе.



Слика 5.11. Минимална и максимална вредност *FEs* – а анализираних метода за проблем опруге

Са слике 5.11 могуће је уочити да најпре *hGPT* постиже оптимум, али постоје тренуци када се то уопште не догоди. Алгоритми *iGA* и *rTLBO* никада не постижу оптимум, али постижу вредност веома блиску оптималној. *DINDI* алгоритам увек постиже оптимум за сличне вредности *FEs* – а. *DINDI* алгоритам је доминантан и код овог оптимизационог проблема, али и *iGA, rTLBO* и *hGPT* алгоритми показују веома добре карактеристике. Повећавањем вредности *FEs* – а било би могуће и код ових метода сваким понављањем постићи оптимум.

5.3.3 Редуктор

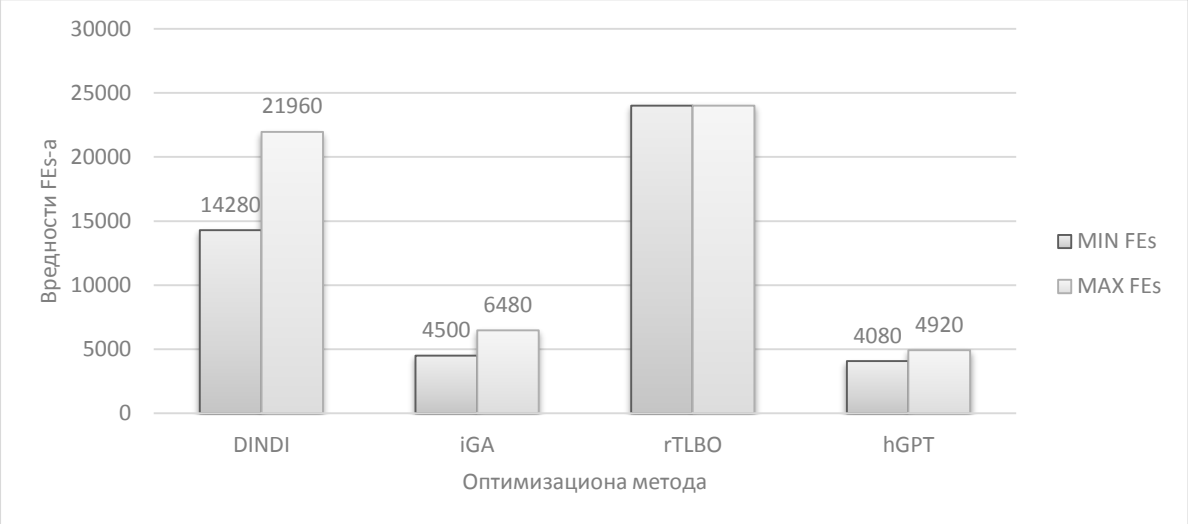
Најбоље познато решење оптимизационог проблема редуктора, које је могуће наћи у литератури је $f(X) = 2996{,}348165$. Проблем је формулисан са једном дискретном (целобројном) и шест континуалних променљивих. Такође, дефинисана су 4 линеарна и 7 нелинеарних ограничења у облику неједначина. Извршена су експериментална испитивања према дефинисаним критеријумима за развијене методе, а резултати су упоређени са резултатима из литературе. Методе из литературе које су коришћене за поређење у овом истраживању су *GA* (*Matlab GA-toolbox*), *PSO-DE*, *ABC*, *TLBO*, *CSA*, *FFA* и *PVS*.

Проблем редуктора анализиран је у свакој литератури која садржи испитивања код инжењерских проблема. Формулација проблема представљена је у додатку Б.3. Као променљиве оптимизације учествују: ширине зупчаника, модул зупчаника, пречници вратила и ширине кућишта на местима где су позиционирана вратила.

Табела 5.8
Поређење резултата анализираних оптимизационих метода за оптимизациони проблем редуктора

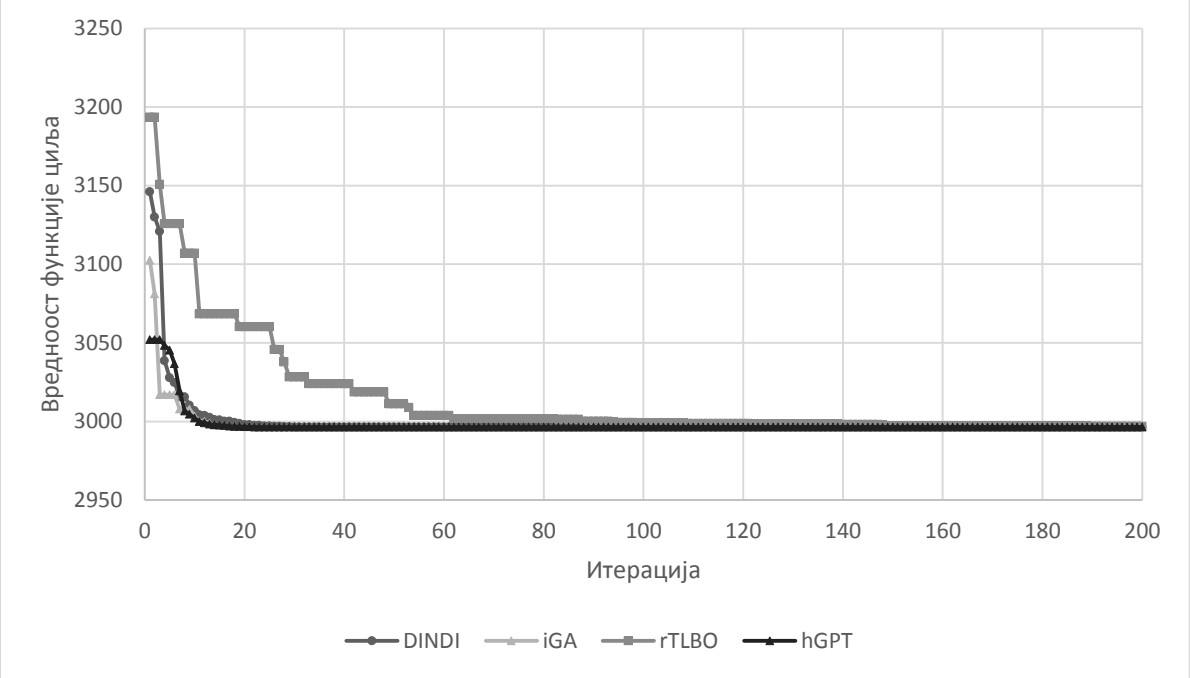
Метода	Најбоља	Најлошија	Средња	FEs
GA (Matlab GA-toolbox) [31]	2996,415435	2997,57529	2996,627632	25000
PSO-DE [121]	2996,348167	2996,348174	2996,348174	54350
ABC [156]	2997,058	H/A	2997,058	30000
TLBO [33]	2996,34817	H/A	3996,34817	20000
CSA [25]	3000,98	30090	3007,1997	5000
FFA [157]	2996,37	2996,669	2996,51	50000
PVS 1 [31]	2996,4	2996,7123	2996,48271	5000
PVS 2 [31]	2996,348165	2996,366218	2996,350001	20000
PVS 3 [31]	2996,348165	2996,348165	2996,348165	54350
DINDI	2996,34816497	2996,34816497	2996,34816497	22000
iGA	2996,348165	2996,348165	2996,348165	6500
rTLBO	2996,35366	2996,38979	2996,37219	25000
hGPT	2996,348165	2996,348165	2996,348165	5000

DINDI алгоритам за овај проблем постиже најбоље решење са *FEs* 22000. Неке од метода анализиране су за мање вредности *FEs* – а, али резултати нису толико добри. Интервал разматран за *FEs* код метода из литературе је од 5000 до 54350. Своје предности код овог оптимизационог проблема показује *iGA*. Овај алгоритам са само 6500 *FEs* – а постиже увек оптимум, сваким поновним покретањем. Метода *rTLBO* има веома добар и стабилан рад и ако не постиже оптимум при постављеним ограничењима. Хибрид *hGPT* се за овај проблем најбоље показао и сваким покретањем, применом ове методе проблем је могуће решити са 5000 *FEs* – а.
Када се анализира минимална и максимална вредност *FEs* – а (слика 5.12) којом методе постижу оптимум, могуће је извести закључак да је *hGPT* најефикаснија метода, затим следи *iGA, DINDI* алгоритам је стандардно добар и за овај проблем, а *rTLBO* у односу на остале методе има нешто лошије, али и даље изузетне карактеристике.



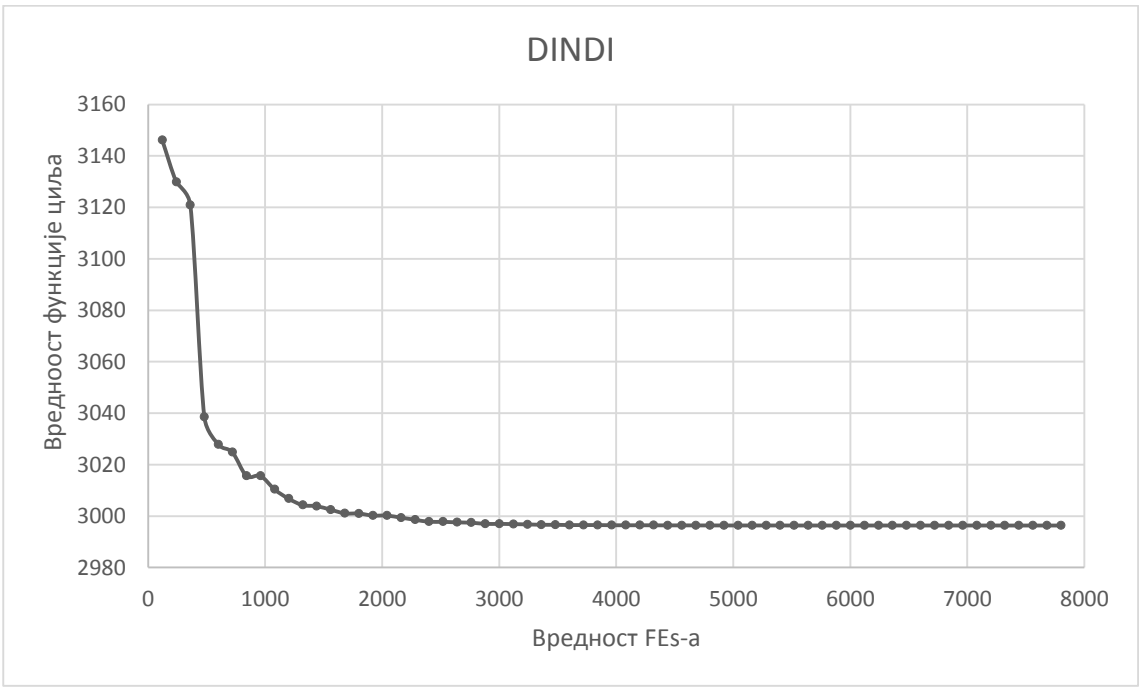
Слика 5.12. Минимална и максимална вредност *FEs* – а анализираних метода за проблем редуктора

Конвергенција алгоритама *DINDI*, *iGA*, *rTLBO* и *hGPT* за проблем редуктора представљена је на слици 5.13. На слици конвергенција је представљена као зависност вредности функције циља и броја итерација за анализиране алгоритме.



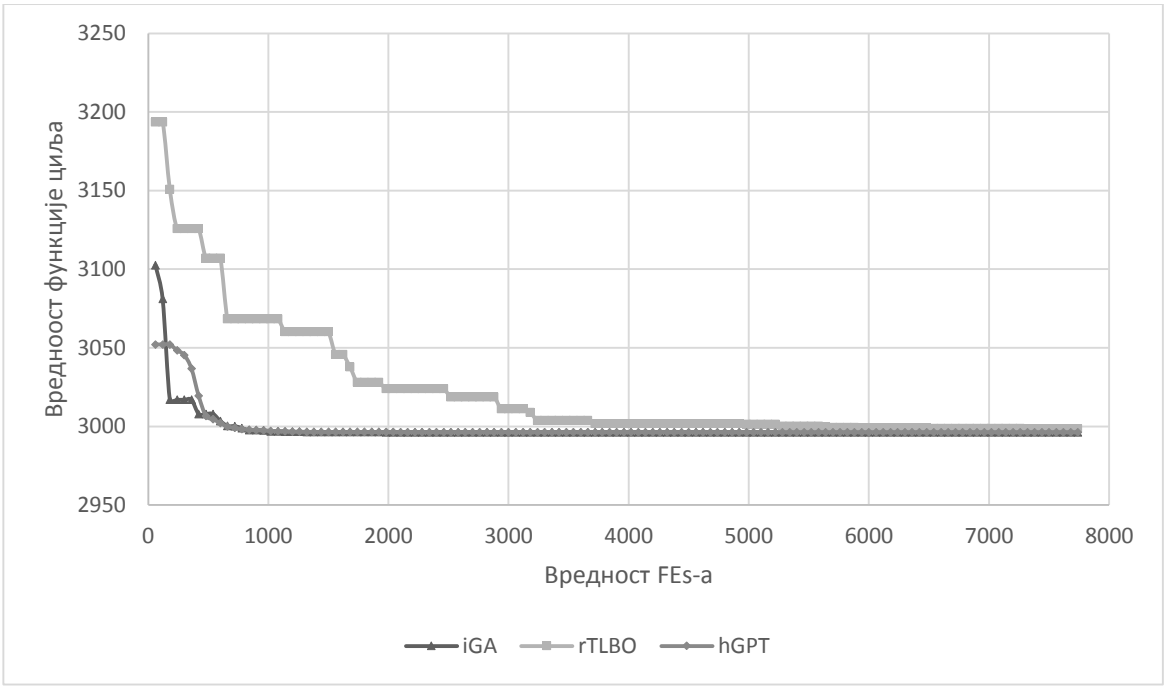
Слика 5.13. Конвергенција развијених алгоритама за проблем редуктора

На слици 5.14 а) (за *DINDI* алгоритам) и слици 5.14 б) (за *iGA*, *rTLBO* и *hGPT* алгоритме) представљена је зависност величине *FEs* – а и вредности функције циља.



а)

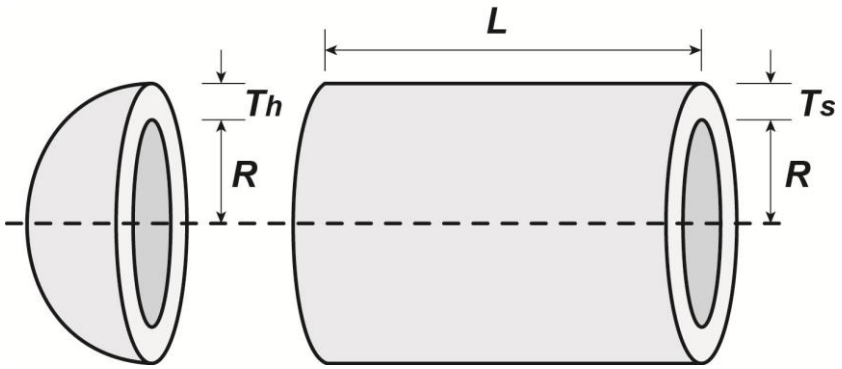
б)



Слика 5.14. Зависност вредности функције циља и вредности *FEs* – а; а) Конвергенција за *DINDI* алгоритам; б) Конвергенција за *iGA*, *rTLBO* и *hGPT* алгоритме

5.3.4 Суд под притиском

Оптимизациони проблем суда под притиском анализиран је само за континуалне промењиве. Циљ овог проблема је да се минимизују укупни трошкови материјала, обликовања и заваривања. Проблем садржи три линеарна и једно нелинеарно ограничење у облику неједначине, као и 4 континуалне промењиве. Очекивана оптимална вредност функције циља овог проблема је $f(X) = 5885{,}3327$, а резултати су представљени у табели 5.9.



Слика 5.15. Шематски приказ оптимизационог проблема суда под притиском

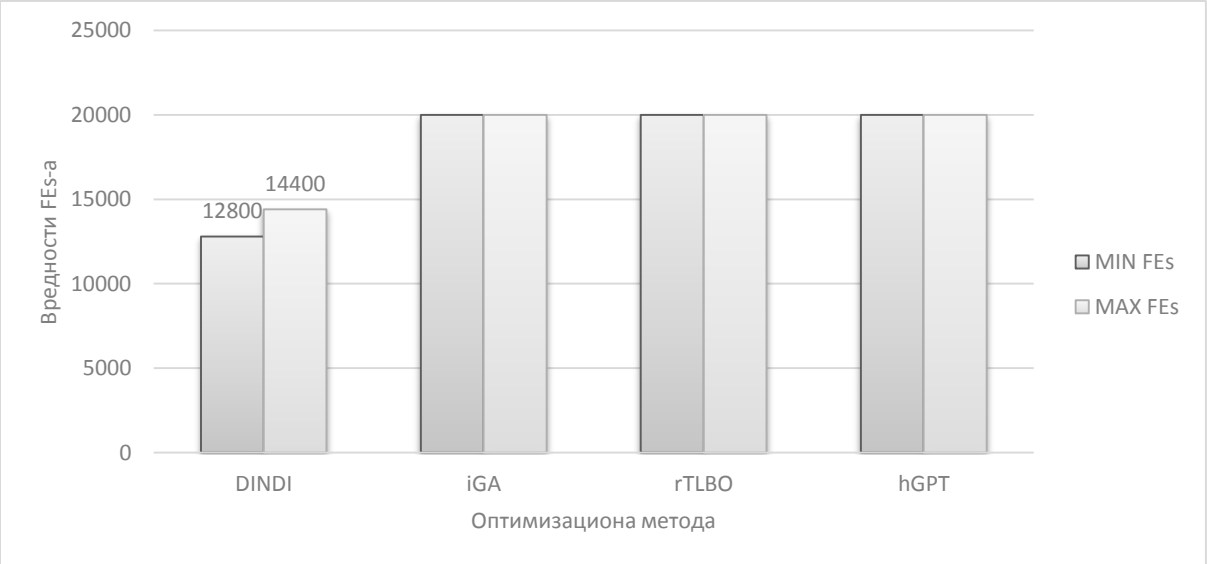
Оптимизационе методе које су анализиране у литератури за овај проблем су *NM-PSO, WCA, MBA* и *PVS*. Детаљна формулација проблема суда под притиском, у складу са сликом 5.15, представљена је у додатку Б.4.

Табела 5.9

Поређење резултата анализираних оптимизационих метода за оптимизациони проблем суда под притиском

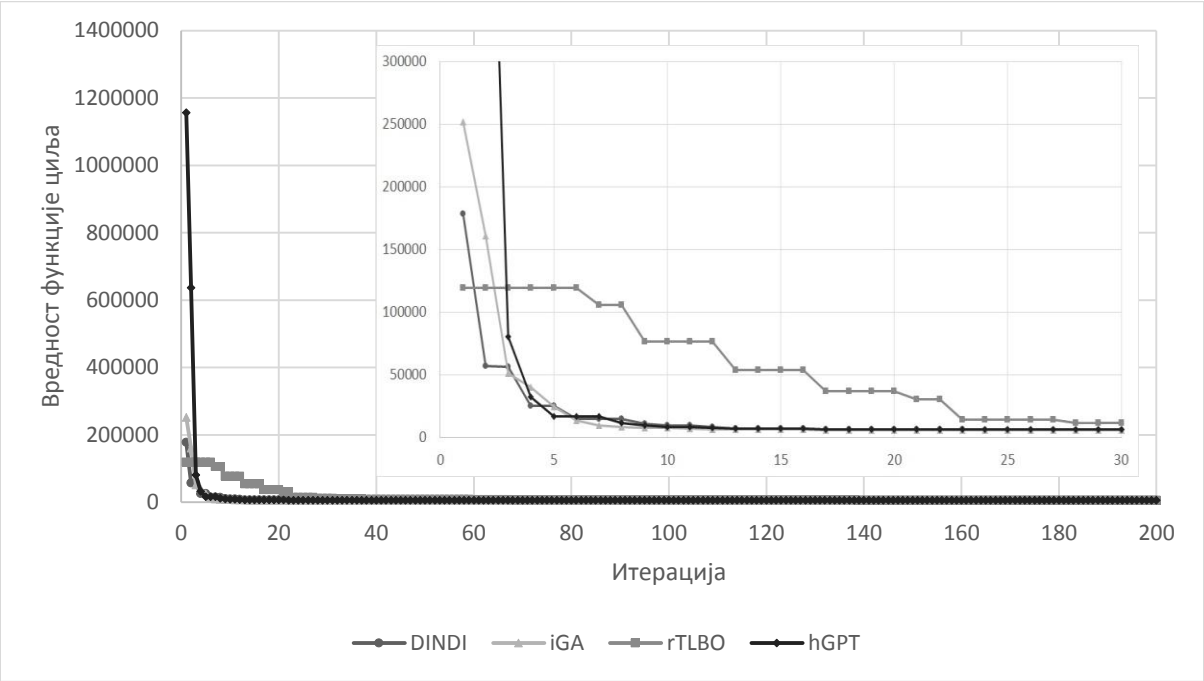
Метода	Најбоља	Најлошија	Средња	FEs
NM-PSO [123]	5930,3137	5960,0557	5946,7901	80000
WCA 1 [27]	5885,3711	7319,0197	6230,4247	8000
WCA 2 [27]	5885,3327	6590,2129	6198,6172	27500
MBA [24]	5889,321	6392,5	6200,64	70650
PVS [31]	5885,333	5886,035	5885,409	20000
DINDI	5885,3327362	5885,3327362	5885,3327362	15000
iGA	5888,853079	6362,87922	6054,319315	20000
rTLBO	5888,506029	5921,302442	5904,223753	20000
hGPT	5885,348281	6630,681239	6083,642611	20000

Вредности *FEs* – а које су коришћене у литератури крећу се у интервалу од 8000 до 80000, док је за *iGA, rTLBO, hGPT* узета фиксна вредност 20000. *DINDI* алгоритам оптимум за свако покретање постиже са 15000 *FEs*. После тога, најбоље вредности постиже *hGPT*, али су резултати нестабилни. Још лошије резултате постиже *rTLBO* са нешто већом стабилношћу решења, а најлошије се показао *iGA*.
Минимална и максимална вредност *FEs* – а представљена је на слици 5.16. Са слике је очигледно да само *DINDI* алгоритам постиже оптимум у дефинисаном интервалу.



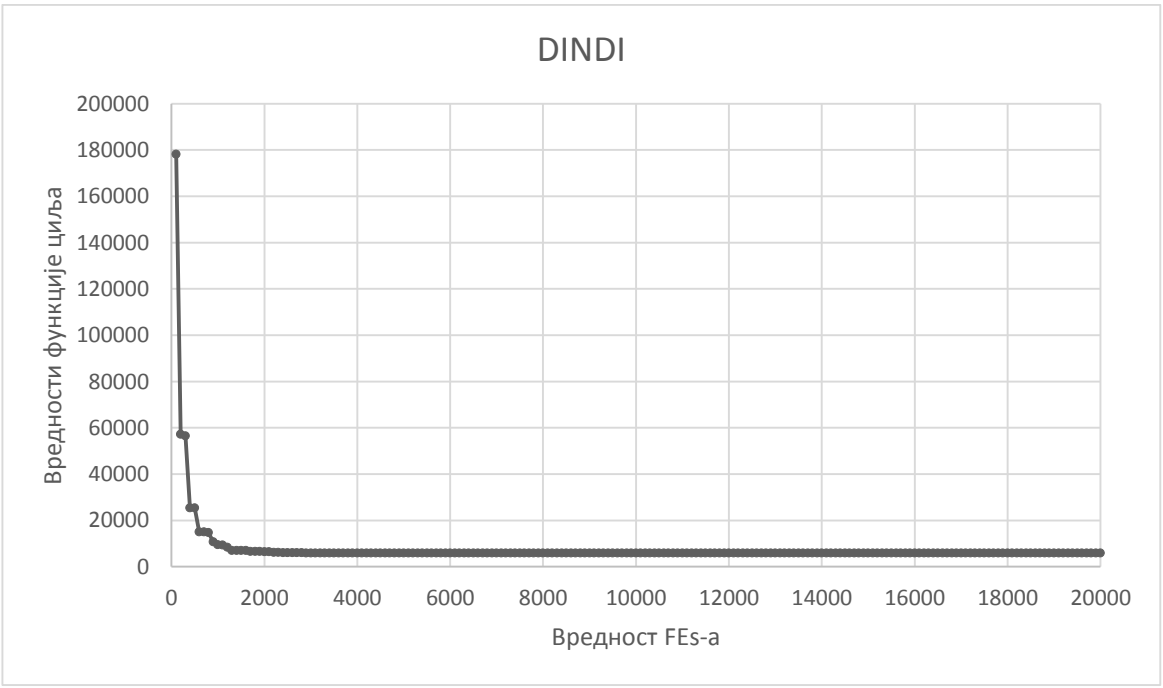
Слика 5.16. Минимална и максимална вредност *FEs* – а анализираних метода за проблем суда под притиском

Конвергенција алгоритама за проблем суда под притиском представљена је на слици 5.17. На слици конвергенција је представљена као зависност вредности функције циља и броја итерација за анализиране алгоритме.



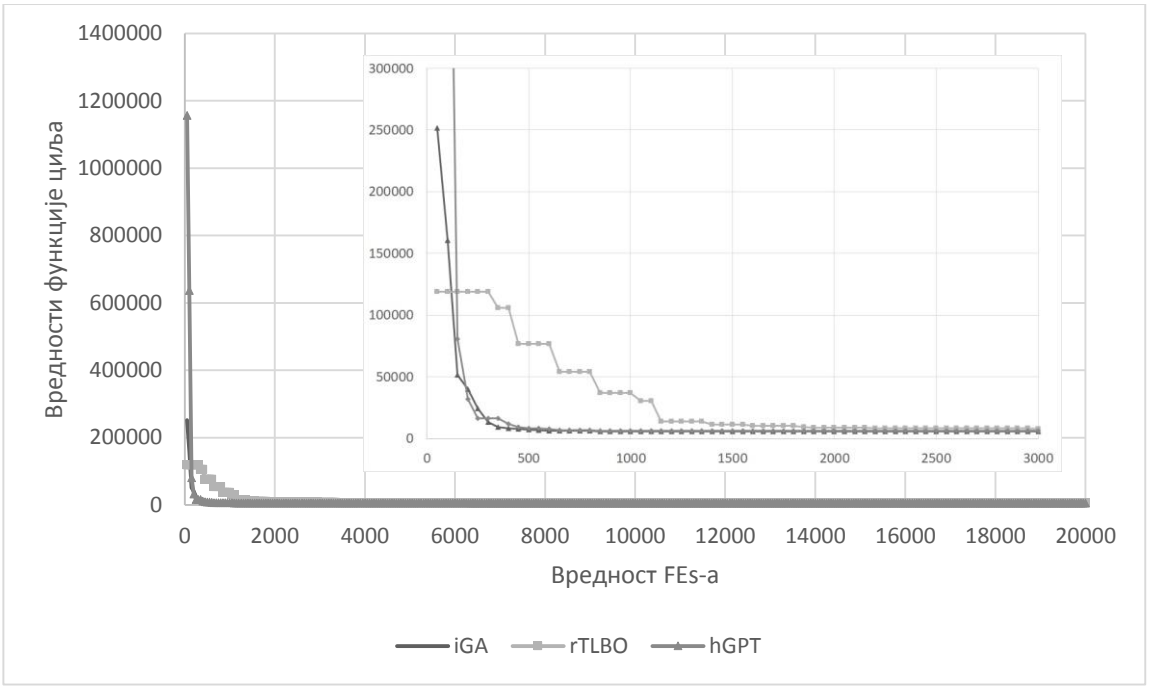
Слика 5.17. Конвергенција развијених алгоритама за проблем суда под притиском

На слици 5.18 а) (за *DINDI* алгоритам) и слици 5.18 б) (за *iGA, rTLBO* и *hGPT* алгоритме) представљена је зависност величине *FEs* – а и вредности функције циља.



а)

б)



Слика 5.18. Зависност вредности функције циља и вредности *FEs* – а; а) Конвергенција за *DINDI* алгоритам; б) Конвергенција за *iGA*, *rTLBO* и *hGPT* алгоритме

Код проблема оптимизације суда под притиском најбоље карактеристике показао је *DINDI* алгоритам, што поново фаворизује овај алгоритам у односу на све остале.

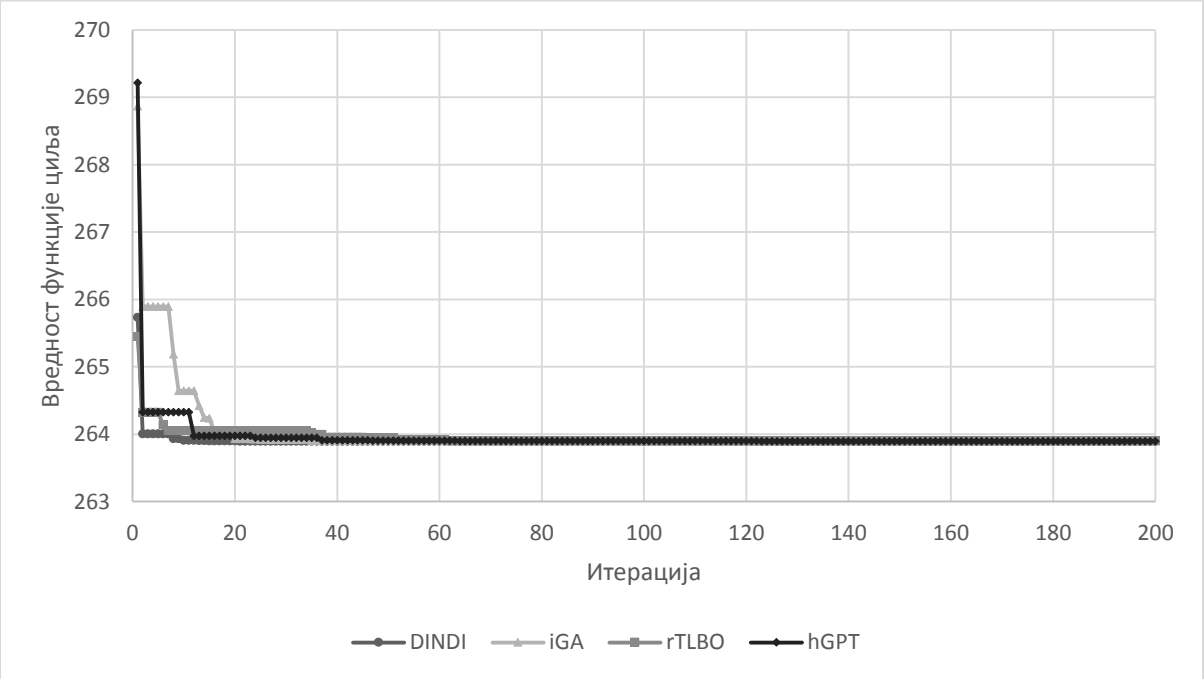
5.3.5 Носач са три штапа

Овај оптимизациони проблем садржи две промењиве и три ограничења у облику неједначина. Проблем је практичан и присутан у свакој литератури где се врши испитивање машинских конструкција са аспекта оптимизације. Димензије штапова представљају промењиве оптимизације, а проблем је формулисан у додатку Б.5. Очекивана вредност функције циља је $f(X) = 263{,}895843$, што је дефинисано у литератури. Интервал анализираних вредности FEs – а у литератури је од 5250 до 17610 за методе *SC*, *PSO-DE*, *DEDS*, *HEAA* и *WCA*. Резултати оптимизације представљени су у табели 5.10.

Табела 5.10
Поређење резултата анализираних оптимизационих метода за оптимизациони проблем носача са три штапа

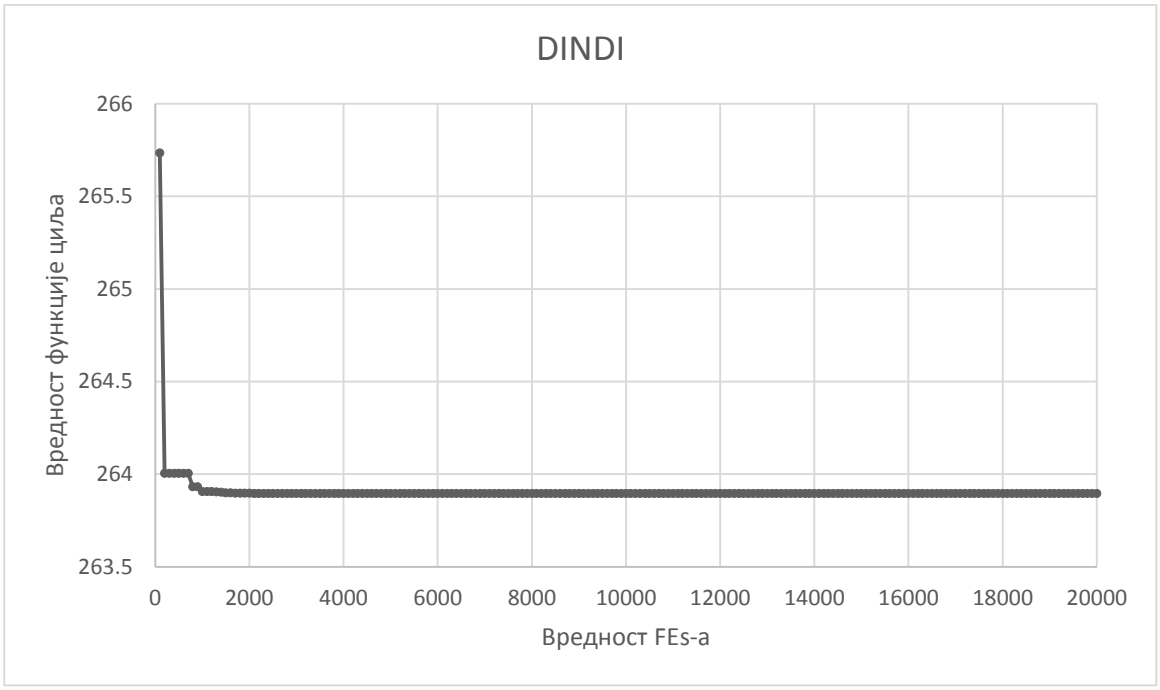
Метода	Најбоља	Најлошија	Средња	FEs
SC [27]	263,895846	263,969756	263,903356	17610
PSO-DE [121]	263,895843	263,895843	263,895843	17600
DEDS [161]	263,895843	263,895849	263,895843	15000
HEAA [159]	263,895843	263,896099	263,895865	15000
WCA [27]	263,895843	263,896201	263,895903	5250
DINDI	263,8958434	263,8958434	263,8958434	5000
iGA	263,895843	264,412771	263,968945	20000
rTLBO	263,89593	263,898066	293,896681	20000
hGPT	263,895843	263,92089	263,89668	20000

DINDI алгоритам са само 5000 *FEs* – а постиже увек оптимум, што га поново фаворизује. Методе *iGA, rTLBO* и *hGPT* су изузетне за решавање овог оптимизационог проблема. Метода *iGA* постиже вредност блиску оптимуму и има прилично стабилне резултате. Боља у односу на *iGA* је *rTLBO*, са још бољом вредношћу и стабилнијим резултатима. Најбоље се показао хибрид *hGPT*, који је веома добар по питању минималног решења и стабилности резултата.
Конвергенција алгоритама *DINDI, iGA, rTLBO* и *hGPT* за проблем носача са три штапа представљена је на слици 5.19. На слици конвергенција је представљена као зависност вредности функције циља и броја итерација за анализиране алгоритме.

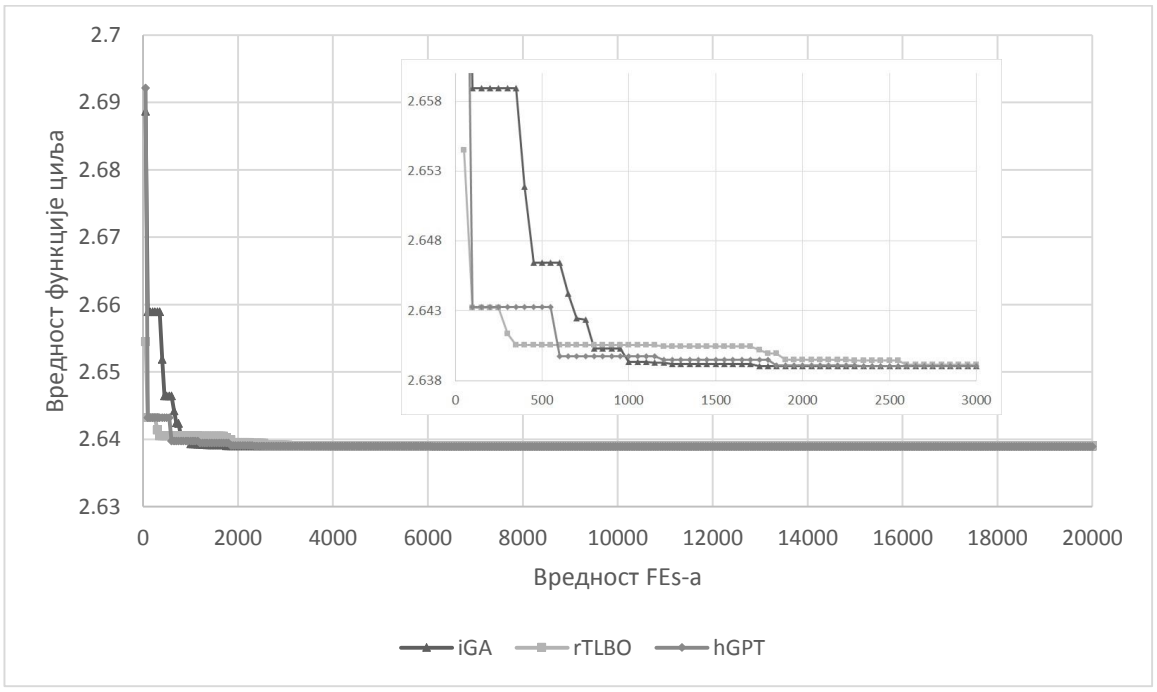


Слика 5.19. Конвергенција развијених алгоритама за проблем носача са три штапа

На слици 5.20 а) и слици 5.20 б) представљена је зависност величине *FEs* – а и вредности функције циља. Ова зависност је веома важна за анализу тока процеса оптимизације.



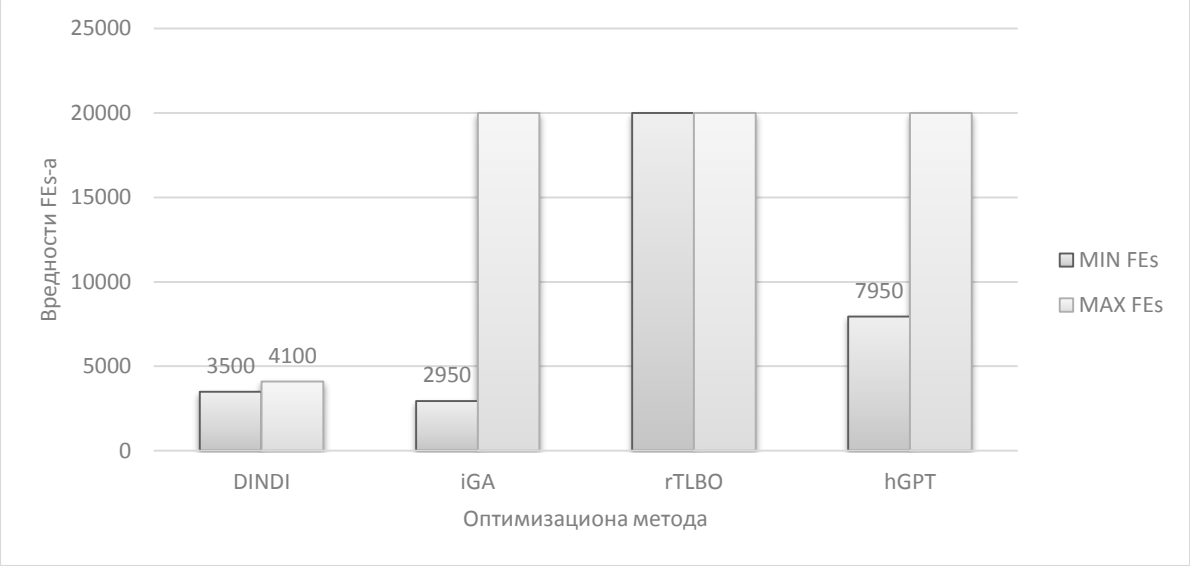
а)



б)

Слика 5.20. Зависност вредности функције циља и вредности *FEs* – а; а) Конвергенција за *DINDI* алгоритам; б) Конвергенција за *iGA, rTLBO* и *hGPT* алгоритме

За приказ конвергенције узете су случајне симулације из скупа од 30 узастопних покретања. Када се посматрају све симулације, најмања вредност за *FEs* и највећа вредност *FEs* – а за одређену методу који је коришћен за постизање приказаног оптимума, представљене су на слици 5.21 за анализиране методе.



Слика 5.21. Минимална и максимална вредност *FEs* – а анализираних метода за проблем носача са три штапа

Алгоритам *iGA* најбрже постигне оптимум, али се то догоди једном од 30 узастопних покретања. *DINDI* алгоритам је и веома ефикасан и добар као и код свих претходних проблема. Алгоритам *rTLBO* даје добре резултате али при постављеним ограничењима не постиже захтевани оптимум. Хибрид *hGPT* је веома добар и може решити овај проблем са вредношћу *FEs* – а од 7950.
Према представљеним резултатима *DINDI* алгоритам се значајно издваја у односу на друге хеуристичке методе оптимизације. За инжењерске примере резултати постигнутих вредности функције циља и променљивих представљене су у табели 5.11.

Табела 5.11
Најбољи постигнути резултати *DINDI* алгоритмом за анализиране инжењерске проблеме

	Заварена греда	Опруга	Редуктор	Суд под притиском	Носач са три штапа
$f(x)$	**1,7248523**	**0,0126652**	**2996,34816497**	**5885,3327362**	**263,8958434**
x_1	0,2057296	0,0516884	3,5000000	0,7781686	0,7886751
x_2	3,4704887	0,3567029	0,7000000	0,3846492	0,4082483
x_3	9,036624	11,2898364	17,0000000	40,3196187	
x_4	0,2057296		7,3000000	200,0000000	
x_5			7,8000000		
x_6			3,3502147		
x_7			5,2866832		

У табели су приказане вредности променљивих за које *DINDI* алгоритам постиже оптимална решења инжењерских проблема, при задовољењу постављених ограничења. Ове резултате могуће је третирати као најбоља позната решења ових проблема оптимизације, што фаворизује *DINDI* алгоритам.

5.4 Поређење развијених метода оптимизације

Под поређењем оптимизационих метода подразумева се одређивање једне методе, погодне у општем смислу за реализацију процеса оптимизације. Пошто је чињеница да свака од развијених метода оптимизације има своје предности и недостатке, потребно је пронаћи баланс између њих. Под појмом добре хеуристичке методе оптимизације неопходно је посматрати могућност методе да постигне оптимум, стабилност резултата при поновном покретању методе, ефикасност методе, флексибилност у погледу управљања параметрима методе, флексибилност у смислу могућности модификације и хибридизације методе, сличност са хеуристичком појавом и др. Брзина рада методе је нешто што је такође веома важно, али са актуелном софтверском и хардверском подршком, модерне хеуристичке методе имају веома мале разлике у брзини рада (чак и код комплексних проблема), тако да је разлике могуће занемарити. Сигурно да је могуће испитивати и овај сегмент, али он не дефинише квалитет рада методе, тако да није узет у обзир за ово истраживање.

Од развијених хеуристичких метода оптимизације очекује се да задовоље поменуте критеријуме, да буду једноставне за имплементацију и да се смањи њихова комплексност употребе. Код нове методе, као што је *DINDI*, потребно је да може да се пореди са савременим методама у сваком смислу. Веома је важно развијати нове методе јер се нови приступ развија у складу са расположивим ресурсима (хардвер и софтвер), као и у складу са актуелним проблемима оптимизације који су све захтевнији и комплекснији. Нове методе морају бити лаке и брже за употребу у циљу популаризације оптимизационог процеса код конкретних практичних оптимизационих проблема. Модификације и хибриди треба да представе напредак у односу на иницијалне методе у односу на које се и врши развој. Ове методе морају бити ефиканије и флексибилније у односу на иницијалне.

Флексибилност развијене методе подразумева могућност методе да решава потпуно различите типове проблема, а да притом задржи ефикасност. Поред тога под флексибилношћу методе подразумева се могућност подешавања параметара, измена, модификације и хибридизације методе ради постизања још веће ефикасности или прилагођавања методе проблемима.

Новоразвијена хеуристичка метода *DINDI* показала се као изузетна, чак је могуће рећи да не показује слабости анализираним проблемима оптимизације. Метода је атрактивна, стабилна, флексибилна, репрезентује интересантну хеуристичку појаву и веома је ефикасна. Алгоритми *iGA, rTLBO* и *hGPT* такође представљају напредак у односу на иницијалне методе на основу којих су и развијене. Наравно да постоје и недостаци ових метода. За *DINDI* алгоритам није откривена група проблема на којој ће показати слабост, а могуће је очекивати да такви проблеми постоје. Затим, *DINDI* алгоритам има поједине параметре који су дефинисани искуствено и на основу анализе хеуристичке појаве. Потребно је одредити тачне вредности ових параметара експерименталним испитивањима како би са сигурношћу било могуће тврдити да је *DINDI* алгоритам изведен на најефикиснији начин. Модификација *iGA* према резултатима проблема без ограничења, са ограничењима и инжењерских проблема представља изванредан напредак у односу на *GA* какав је познат у основном облику. Развијена модификација постиже чак и добре резултате у односу на модерне хеуристичке методе оптимизације, тако да је могуће рећи да је постигнут жељени ефекат. За *rTLBO* модификацију, резултати нису значајно бољи од иницијалног *TLBO* алгоритма. Сам *TLBO* алгоритам је веома модеран и атрактиван и има изванредан рад. Модификација *rTLBO* пружа само већу стабилност за примену ове методе на различитим типовима проблема

оптимизације, што наравно представља напредак. Хибридизација *hGPT* поред повољних карактеристика које показује при експерименталним испитивањима, презентује и нови приступ хибридизацији. Овај и слични видови хибридизације не отклањају недостатке које нека метода садржи, већ их превазилазе потпуно другим приступом (методом) решавања оптимизационог проблема. Могуће је рећи да овакав приступ представља перспективу у развоју методе оптимизације и да је само овим приступом могуће направити универзалне методе оптимизације.

Постоје многобројни начини оцењивања и селекције методе оптимизације, али на основу резултата је и више него очигледно да се за све анализиране оптимизационе проблеме *DINDI* алгоритам значајно издваја у односу на друге методе. Имајући ту чињеницу у виду са сигурношћу је могуће тврдити да је овај алгоритам у предности у односу на остале развијене алгоритме, као и многе друге алгоритме из литературе. Ова чињеница потврђује да *DINDI* метода може да буде активирана за даље анализе практичних проблема машинског конструисања и да се успех може гарантовати.

6. Оптимизација машинских конструкција

Примена метода хеуристичке оптимизације представља алтернативни правац у решавању инжењерских проблема. Оптимизација је процес који је могуће уградити у готово све фазе инжењерства. Применом оптимизационих метода постижу се значајне уштеде. Те уштеде огледају се у квалитету рада, смањењу масе и запремине, уштеди енергије, искоришћењу људских ресурса, итд. Свакако, потребно је да труд, ресурси и сви други уложени напори буду мањи од позитивних ефеката, заправо коначне добити по обављеној оптимизацији. Када је овај услов испуњен, процес оптимизације успешно је завршен. Постићи овај циљ у инжењерској пракси представља комплексан и значајан корак. Циљ је да се практично примени изабрана хеуристичка метода и постигну значајно бољи резултати у односу на почетне вредности практичног проблема код машинских конструкција.

Машинско конструисање представља креативни процес са јасно дефинисаним циљевима уз одређена ограничења и потребу за адекватним доношењем одлука. Како би овај процес био успешан, а конструкционо решење оптимално, погодно за стварну практичну примену, јавља се потреба за применом оптимизације у процесу конструисања. Потреба за оптимизацијом расте сразмерно са повећањем комплексности проблема. Имплементација оптимизације подразумева јасно дефинисање циљева (функције циља), дефинисање поља претраге и стварних ограничења, као и избор адекватне оптимизационе методе. Оптимизација представља проналажење адекватног могућег решења из групе алтернативних могућих решења. Хеуристичке методе издвајају се у први план за примену код проблема машинских конструкција и машинског конструисања због својих повољних карактеристика попут рада са великим бројем променљивих, превазилажења локалних екстремних вредности, своје брзине и ефикасности, разноликости решивих проблема и малог броја потребних параметара да би проблем био решив.

Анализирани инжењерски проблеми представљају конкретне примере машинских конструкција. Ови примери су опште познати (примери из литературе) и зато су искоришћени за анализу рада оптимизационих метода. У овом поглављу анализирани су приступи оптимизације за конкретне проблеме преносника снаге. Први део односи се на конвенционалне преносника – зупчасте преносника снаге, тачније редукторе са аспекта адекватног позиционирања оса вратила зупчаника у циљу смањења запремине редуктора, а самим тим и масе. За анализу овог проблема коришћени су математички модели из литературе, који су адаптирани и, на оригиналан начин, процес оптимизације је спроведен до краја. За други практични оптимизациони проблем машинског конструисања примењен је оригинални приступ, тј. анализа. Развијен је потпуно оригинални математички модел и спроведен процес оптимизације за проблем корекције профила циклозупчаника код циклоредуктора.

Инжењерске проблеме је веома тешко оптимизовати у пракси. Када су нам проблеми познати (знамо решења) можемо да пратимо процес оптимизације јер постоје очекивани резултати. Тада је могуће повећавати број потребног прерачунавања функције циља и постићи оптимум. Код практичних проблема често није познато решење, тешко га је предвидети, а тиме је тешко оценити и успех оптимизационог процеса. Неопходно је имати у виду да решења оптимизације морају бити потпуно применљива у пракси, јер у

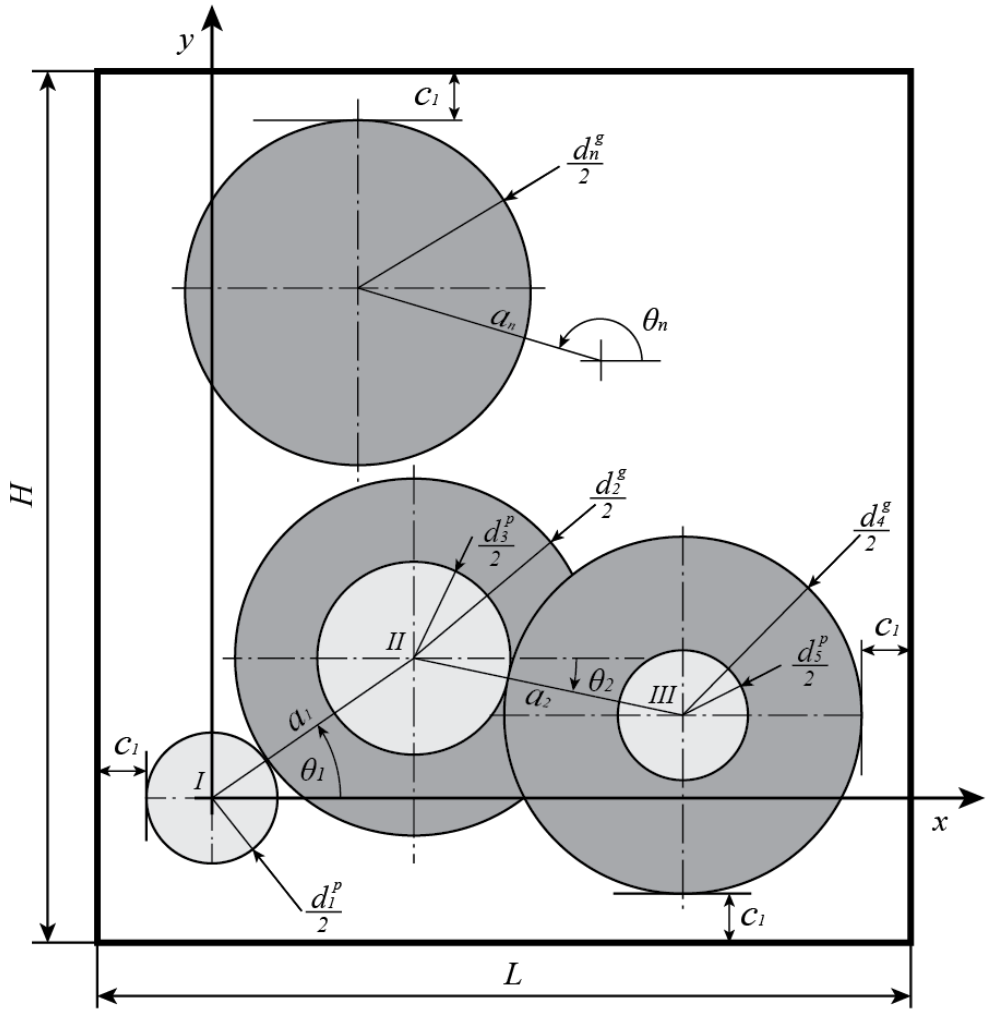
супротном значи да математички модел није развијен на адекватан начин и да не презентује стварни процес или појаву. Зато је неопходно поменути да је проблем практичне инжењерске оптимизације веома комплексан и захтева много знања и искуства инжењера који га спроводи.

6.1 Оптимизација запремине редуктора позиционирањем оса вратила

Комплексан проблем који представља машинску конструкцију, примењује се масовно, а његово усавршавање и оптимизација непрестано трају, јесте редуктор. Баш због масовне примене редуктора, неопходно је представити начин и могућности примене хеуристичких метода оптимизације на ову машинску конструкцију. Оптимизација редуктора је изузетно захтевна, без обзира на који критеријум оптимизације се односи, јер садржи велики број променљивих оптимизације, математички модели су комплексни, а садрже и велики број захтевних ограничења.
За потребе овог истраживања уочено је да највећи допринос може да се постигне применом хеуристичких метода оптимизације. Разматран је критеријум оптимизације запремине редуктора са паралелним осама вратила. Запремина представља и један од највећих проблема код конструкције редуктора, а смањењем запремине постиже се и мања маса редуктора, као и нижи производни трошкови. Истраживања из ове области већ постоје по питању математичког модела. Математички модел је развијен по узору на постојећа решења и адаптиран је за употребу. Примењене су нове хеуристичке методе оптимизације и анализирана су реална ограничења у оваквом приступу оптимизацији. Основа овог истраживања представља истраживање Н. Марјановића [2], где је развијен практичан приступ оптимизације редуктора, а само један сегмент представља оптимизација редуктора са паралелним осама вратила. За потребе овог истраживања извршена је адаптација и креирање математичког модела погодног за рад са хеуристичким методама оптимизације.
Оптимизација запремине редуктора [162] представља честу истраживачку тему, а при томе је важно узети у обзир све утицајне параметре на анализирани проблем. Чонг (*Chong*) и други [163] представили су општи модел методологије за оптимизацију преносног односа, димензија и запремине за вишестепене редукторе и дефинисали фазе прелиминарног дизајна редуктора. Марјановић је са групом аутора [164] развио практични приступ оптимизацији редуктора, узимајући у обзир све утицајне параметре. Ово истраживање базирано је на матрици селекције, концепцији редуктора, оптималном избору материјала, оптималном преносном односу, оптималној запремини и друго. Голаби (*Golabi*) је са групом аутора [165] у истраживањима представио оптимизацију запремине и масе (минимизацију) једностепених и вишестепених редуктора. У литератури [166] се потенцира примена еволуционих (хеуристичких) алгоритама за вишекритеријумску оптимизацију запремине и снаге редуктора истовремено. Од метода, *GA* је најзаступљенији и најмасовније примењивана метода. Менди (*Mendi*) је са осталим ауторима [167] анализирао минимизацију запремине редуктора оптимизацијом димензија елемената редуктора применом генетског алгоритма. Генетски алгоритам су користили и Савсани (*Savsani*) и други [168] за оптимизацију масе редуктора. За оптимизацију параметара редуктора овај алгоритам су користили и Гологлу (*Gologlu*) и Зејвели (*Zeyveli*) [169]. Самим позиционирањем оса вратила зупчаника код редуктора у циљу оптимизације његове запремине бавили су се и други аутори [170, 171].

Постоји реална и практична потреба да се објасни и математички формулише запремина зупчастих преносника снаге, проналажење утицајних променљивих и да се пронађу приступи и методе којима је могуће постићи оптималне карактеристике ове конструкције. Мотивација представља потребу да се практично примени хеуристичка оптимизација и да се укаже на стварне предности које овакав приступ доноси. Направљен је универзални математички модел за оптимизацију свих зупчастих преносника снаге (редуктора) са паралелним осама вратила, а верификација приступа извршена је на три реалне практичне конструкције двостепеног, тростепеног и четворостепеног преносника.

Преносници са паралелним осама вратила зупчаника у пракси се углавном изводе тако да им осе вратила леже у истој равни и то најчешће у хоризонталној. Овако изведен редуктор заузима велику запремину коју је могуће смањити оптимизацијом. Овај приступ оптимизације запремине редуктора подразумева промену позиција оса вратила зупчаника, а резултат тога је смањење запремине. Запремина редуктора представља производ дужине, ширине и висине. Због прорачунских вредности чврстоће зупчаника, ширина зупчаника у овој оптимизацији мора да остане константна, а самим тим и ширина редуктора. То указује на чињеницу да код овог оптимизационог приступа запремина редуктора зависи од дужине (L) и висине (H), које се касније множе са константном ширином. Приступ позиционирања оса вратила зупчаника могуће је видети на слици 6.1.

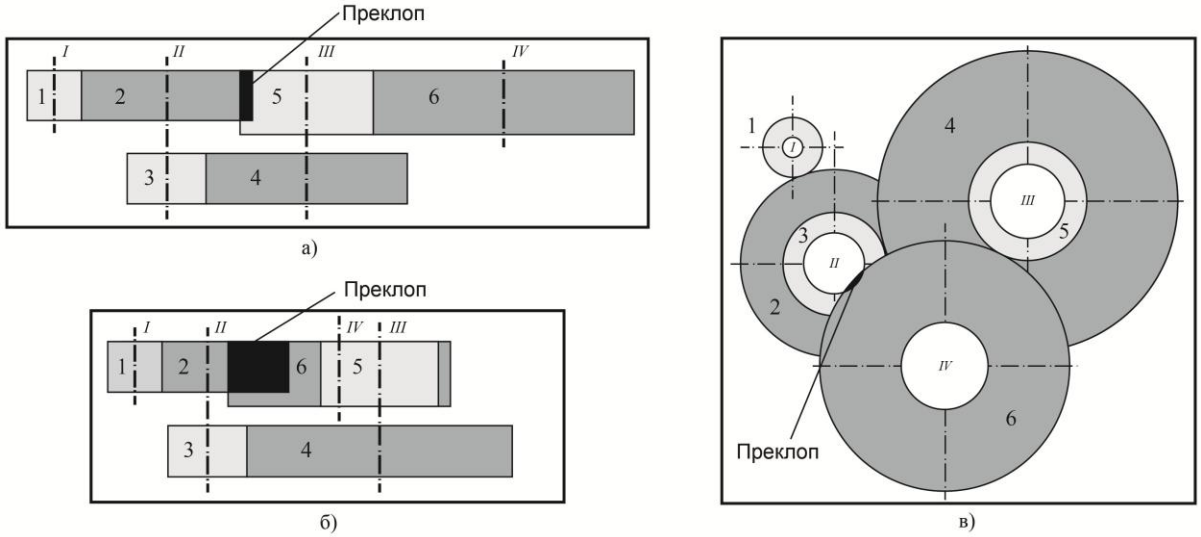


Слика 6.1. Формулација оптимизације запремине редуктора

Координатни почетак је везан за осу улазног вратила, док су позиције других вратила дефинисане углом θ_i у односу на хоризонталну осу, што је могуће уочити на слици 6.1. Запремина редуктора директно зависи од промене ових углова, где одговарајуће вредности углова представљају оптимално решење. Проблем је постављен у *x-y* координатни систем, а са свих страна у односу на зупчанике остављен је технички зазор, који је означен са c_1. Укупну дужину и висину је неопходно посматрати са овим зазорима, како би извођење редуктора у пракси било могуће.
Код вишестепених редуктора, морају бити испуњени одређени услови, како би редуктор функционисао. Основни услов је да међуосно растојање зупчастог пара мора да испуњава услов као у неједначини (6.1).

$$a_{uv} > \frac{d_{u-1}^{g}}{2} + \frac{d_{v+1}^{p}}{2} \tag{6.1}$$

Представљени услов у неједначини (6.1) обезбеђује да нема контакта између неспрегнутих елемената који се налазе у истој равни. Пример овог контакта представљен је на слици 6.2 а). Ознака u на слици представља зупчаник 3, док ознака v представља зупчаник 4. Међуосно растојање 3 – 4 мора бити веће од збира половина пречника зупчаника 2 и 5, за овај конкретан пример. Ако је испуњен услов да се елементи редуктора не преклапају могуће је извршити оптимизацију редуктора позиционирањем оса вратила зупчаника променом одговарајућег угла. Минимална запремина је последица минималне површине где се позиционирање зупчаника врши у односу на хоризонталну раван. Угао који линије међуосних растојања могу да заклапају са хоризонталном осом крећу се у интервалу $-180° \leq \theta_i \leq 180°$.



Слика 6.2. Приказ могућих контаката елемената а) иницијални услов за елементе који нису у пару; б) услов контаката елемената у процесу оптимизације (ограничење); в) ограничење контаката зупчаника и вратила у оптимизацији

Општи математички модел, где су узети у обзир и технички зазори могуће је представити као у једначини (6.2). Овај модел креиран је на основу слике 6.1 и представља функцију која описује запремину редуктора (производ дужине и висине) у

зависности од димензија зупчаника, међуосних растојања и углова које заклапају међуосна растојања са хоризонталном осом. Минимизацијом ове функције, минимизује се и површина редуктора, а тиме и укупна запремина. Општост ове функције подразумева примену без обзира колико степени преноса редуктор има.

$$f = \left[\max\left(\frac{d_i^{(p)}}{2}, \sum_{i=1}^{n}(a_i \cdot cos\theta_i) + \frac{d_i^{(g)}}{2}\right) + c_1 - \min\left(-\frac{d_i^{(p)}}{2}, \sum_{i=1}^{n}(a_i \cdot cos\theta_i) - \frac{d_i^{(g)}}{2}\right) + c_1\right] \cdot \left[\max\left(\frac{d_i^{(p)}}{2}, \sum_{i=1}^{n}(a_i \cdot sin\theta_i) + \frac{d_i^{(g)}}{2}\right) + c_1 - \min\left(-\frac{d_i^{(p)}}{2}, \sum_{i=1}^{n}(a_i \cdot sin\theta_i) - \frac{d_i^{(g)}}{2}\right) + c_1\right] \tag{6.2}$$

Да би математички модел био реална репрезентација стварног проблема, то подразумева укључивање стварних ограничења која постоје у процес оптимизације. Осе зупчаника могу бити позициониране било где у равни применом постављеног математичког модела. Мора постојати ограничење које не дозвољава да се ротацијом неког елемента догоди преклоп између неспрегнутих елемената, што је и представљено на слици 6.2 б). Могуће је да се променом положаја позиције оса вратила зупчаника догоди преклоп између неког зупчаника и неког вратила редуктора. Потребно је и ово посматрати као ограничење, а пример овог случаја представљен је на слици 6.2 в). Сви зазори, тачније растојање међу елементима третирани су тако да не смеју бити мањи од 15 милиметара. Функција циља тежи да позиционира зупчанике на што је могуће мањој површини, а ограничења не дозвољавају да позиција зупчаника буде нереална.

Ограничења су конципирана тако да растојање a_{mn} између нека два центра зупчаника мора бити довољно велико да не дође до преклопа између елемената редуктора. За решавање овог проблема (примене ограничења) потребно је узети у обзир димензије елемената, међуосна растојања и углове под којима се елементи налазе. Општи облик ових ограничења, без обзира на њихов потребан број могуће је представити као у једначини (6.3). Ова ограничења могуће је применити и за спречавање контакта између зупчаника и за спречавање контакта између зупчаника и вратила.

$$a_{mn} = \sqrt{x_{mn}^2 + y_{mn}^2} \geq \frac{d_m}{2} + \frac{d_n}{2} + c_2\,; \quad c_2 = 15mm$$
$$x_{mn} = \sum_{j=m}^{n} a_j \cdot cos\theta_j\,; \qquad y_{mn} = \sum_{j=m}^{n} a_j \cdot sin\theta_j\,. \tag{6.3}$$

Цео математички модел је геометријски оријентисан и применљив за све степене преноса. Број ограничења зависи од тога колико степени преноса постоји (зупчастих парова). Проблем оптимизације ове машинске конструкције је веома комплексан јер функција има веома велики број локалних екстремних вредности и потребно је узети у обзир велики број ограничења која процес оптимизације чине додатно комплексним. Најтеже у оптимизационом процесу је постићи нешто што је могуће практично применити, а оптимизација ове машинске конструкције представља баш такав приступ. Најрелевантнија верификација развијеног приступа јесте примена ове методологије на реалне зупчасте преносника снаге, што је и реализовано овим истраживањем. Важно је указати на укупан допринос примене хеуристичких метода оптимизације код решавања проблема или само унапређење машинских конструкција. Одабрана су три примера редуктора, и то двостепени, тростепени и четворостепени редуктор, како би био представљен приступ и допринос примене оптимизације у односу на општу уобичајену реализацију редуктора са осама вратила у истој равни. Улазни подаци редуктора

преузети су из литературе, а величине потребне за оптимизацију представљене су у табели 6.1.

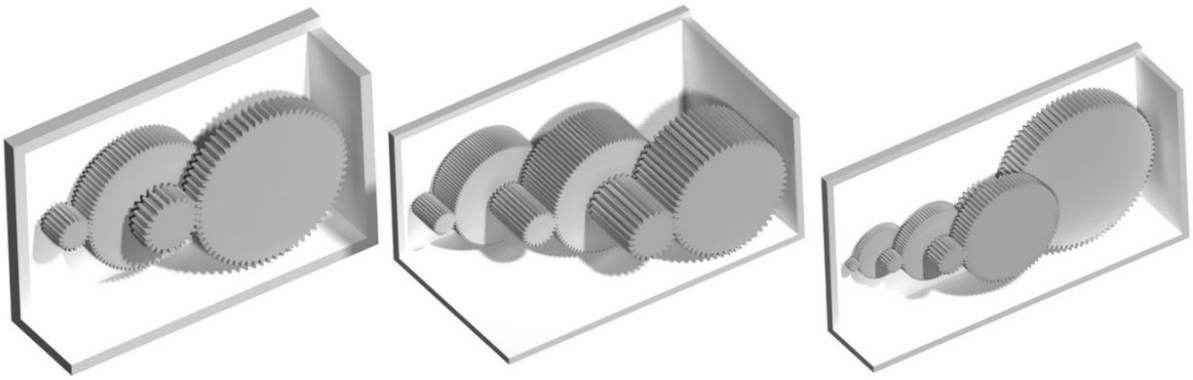
Табела 6.1

Конкретне вредности улазних података за анализиране редукторе

	Двостепени редуктор			Тростепени редуктор			Четворостепени редуктор			
Степен преноса	1	2		1	2	3	1	2	3	4
Модул *mm*	2	3		2	3	4,5	1,5	2	3	4
Број зубаца погонског зупчаника	20	23		24	24	23	14	18	20	25
Број зубаца гоњеног зупчаника	83	67		110	84	56	77	79	74	82
Подеони пречник погонског зупчаника *mm*	40	49		48	72	103,5	21	36	60	100
Подеони пречник гоњеног зупчаника *mm*	166	201		220	252	252	115,5	158	222	328
Међуосно растојање *mm*	103	135		134	162	177,75	68,25	97	141	214
Дужина, висина, ширина, *mm*	388,5	231	110,6	653,75	282	392,4	724,75	358	153	

Улазне величине примера 1 (двостепеног преносника) преузете су из [164], пример 2 (тростепени редуктор) из [164], док је пример 3 (четворостепени редуктор) преузет из [163]. Иницијална вредност запремине редуктора за пример 1 износи 9 925 631,1 mm^3, где се подразумева да осе вратила зупчаника леже у истој равни и да је узет зазор исти као код оптимизације од 15 милиметара. За примере 2 и 3 иницијалне запремине износе 72 341 883 mm^3 и 39 697 456,5 mm^3, респективно.

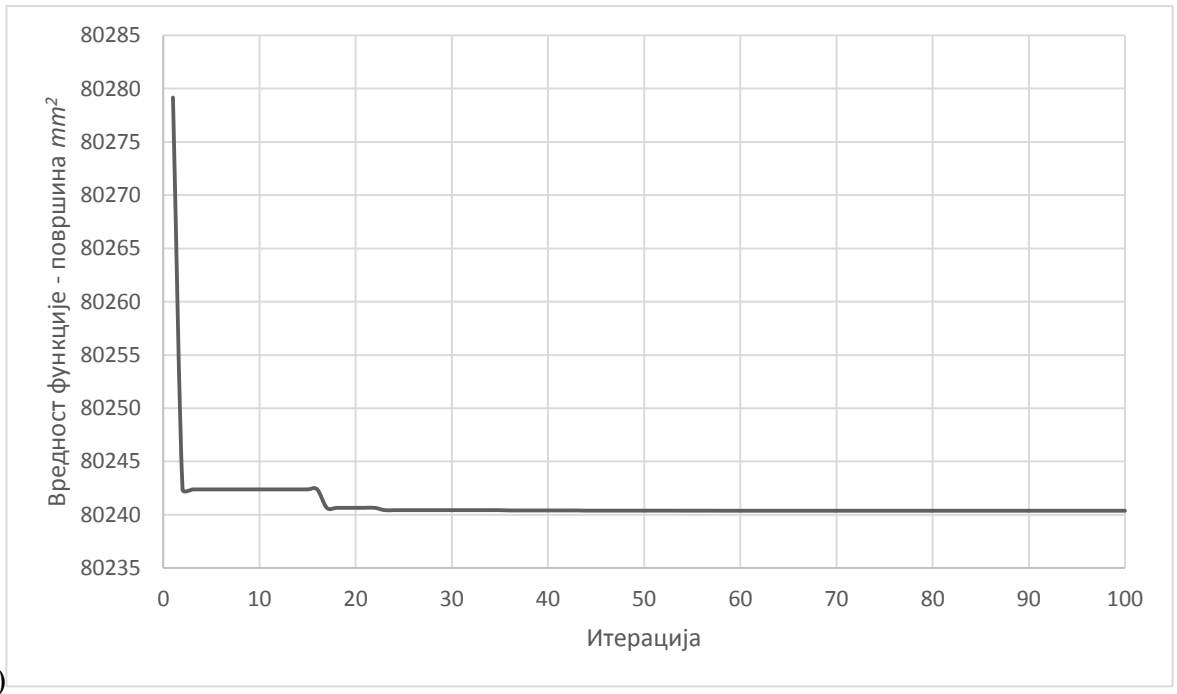
Математички модел је формулисан на основу геометријских карактеристика вишестепених редуктора у зависности од положаја оса вратила зупчаника. Проблем практичне оптимизације јесте постизање неког решења које је могуће третирати као оптимално. За потребе оптимизације за овај практични проблем коришћена је нова *DINDI* метода, развијена за решавање комплексних проблема, која се у претходним анализама истакла по својим карактеристикама. Тачност која се захтева за овај проблем није висока, јер су решења представљена у mm^2, па укупна одступања (после нпр. друге децимале) немају практичног утицаја на практичну примену. На слици 6.3 представљени су иницијални примери двостепеног, тростепеног и четворостепеног редуктора који су анализирани.



а) б) в)

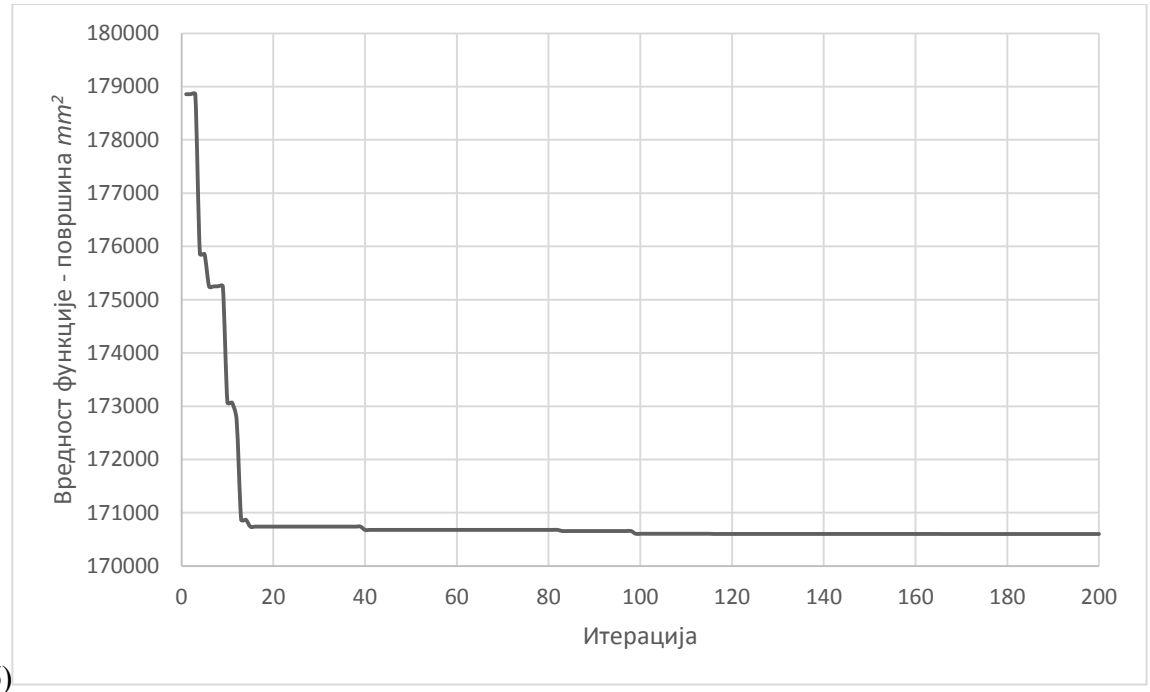
Слика 6.3. Положај оса вратила зупчаника за иницијалне концепције а) двостепени редуктор; б) тростепени редуктор; в) четворостепени редуктор

За сваки конкретан пример вредности из табеле 6.1 интегрисане су у општи облик једначине (6.2), при задовољењу ограничења из неједначина (6.1) и (6.3). Извршена је минимизација функције циља *DINDI* алгоритмом помоћу развијеног софтвера, а процес конвергенције представљен је на слици 6.4, где је дат ограничени број итерација за које се решење веома приближава оптималном.

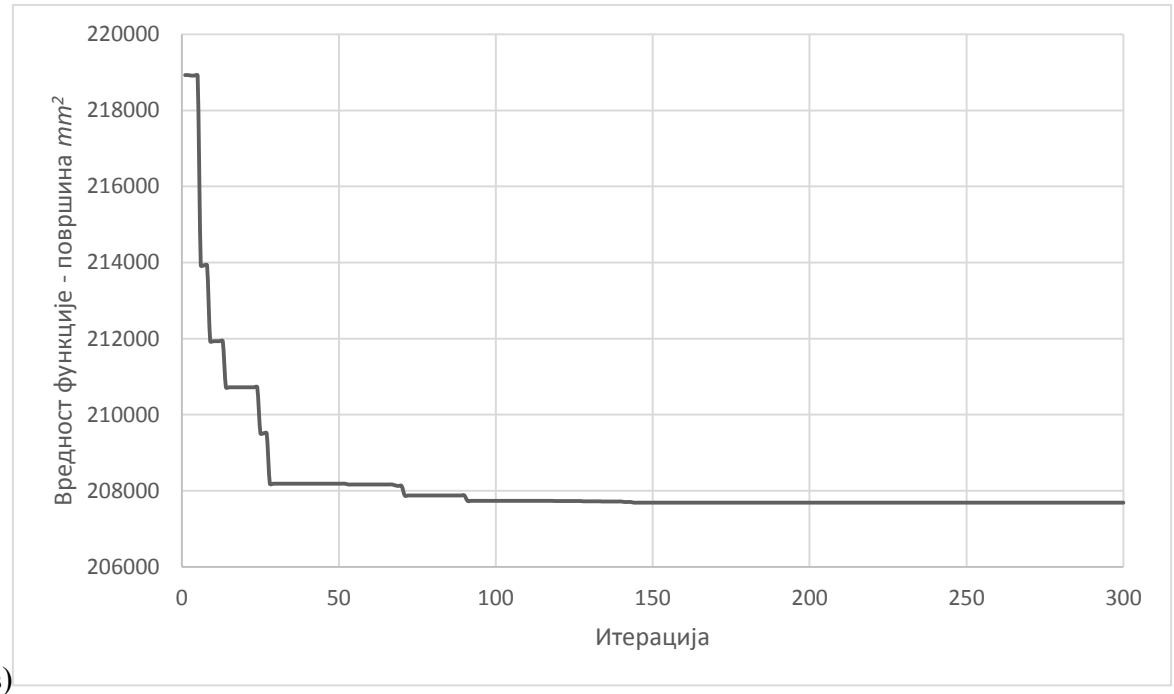


а)

б)



в)



Слика 6.4. Конвергенција *DINDI* алгоритма за проблем положаја оса вратила зупчаника а) двостепени редуктор; б) тростепени редуктор; в) четворостепени редуктор

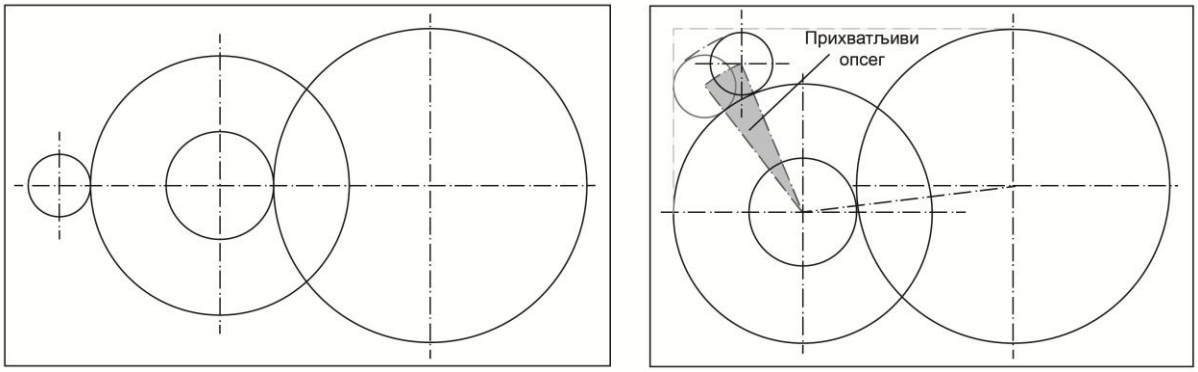
Са повећањем величине *FEs* – а, алгоритам постиже решења веће тачности. Оптимизацијом се добија решење које представља угао под којим је постављена оса вратила зупчаника редуктора, а као решење се постиже површина – производ дужине и висине. Ширина се третира као константна јер зависи од прорачуна зупчастих парова и неопходних техничко / технолошких зазора. У табели 6.2 представљена су могућа решења добијена оптимизацијом анализираних редуктора.

Табела 6.2

Резултати оптимизације анализираних редуктора

	θ_1 °	θ_2 °	θ_3 °	θ_4 °	L *mm*	H *mm*	B *mm*	V mm^3
Двостепени редуктор	-55,0359	7,4482	-	-	347,361	231	110,6	8874585,592
Тростепени редуктор	-61,3595	5,6681	0	-	604,958	282	392,4	66942709,8
Четворостепени редуктор	-154,262	-144,935	-57,732	14,3393	555,61	373,702	153	31767763,44

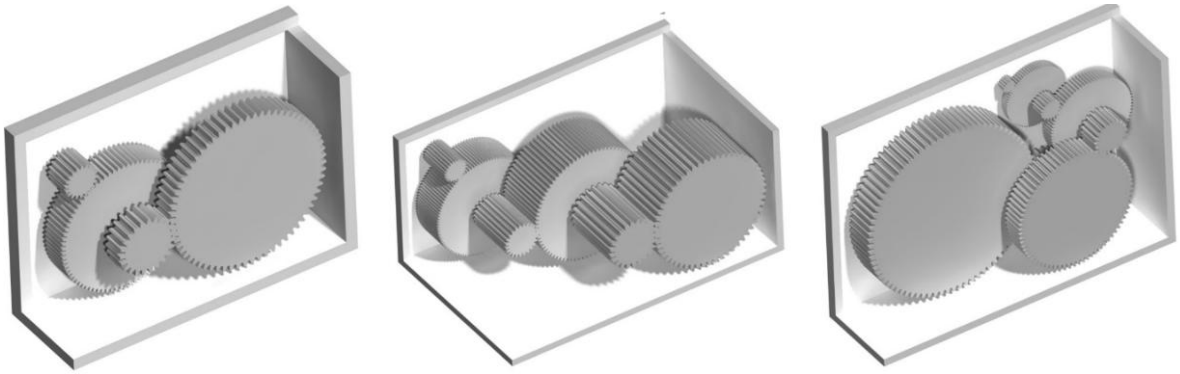
Оптимизацијом се добијају решења у могућем и дозвољеном опсегу, тачније постоји скуп решења од којих су сва оптимална. Пример опсега решења представљен је на слици 6.5, где је за пример двостепеног редуктора представљен опсег угла за чије вредности ће запремина имати исту вредност и бити оптимална.



а) б)

Слика 6.5. а) Иницијални концепт двостепеног редуктора; б) Прихватљиви опсег положаја осе вратила зупчаника оптималног решења двостепеног редуктора

Са слике 6.5 било која вредност угла у представљеном опсегу даје оптимално решење запремине редуктора. Тако на пример дозвољена вредност угла код примера двостепеног редуктора може бити у опсегу $-52.4912° \leq \theta_1 \leq -72.0083°$ и за било коју од ових вредности решење ће бити оптимално. У табели 6.2 представљен је један од могућих случајева решења, вредности за углове положаја оса вратила, дужина, висина и ширина редуктора, као и оптимална запремина. За представљене вредности узети су у обзир технички зазори. Визуелизација оптималних решења анализираних редуктора представљена је на слици 6.6, где су приказане вредности добијене у табели 6.2 за двостепени, тростепени и четворостепени редуктор.



а) б) в)

Слика 6.6. Оптималне концепције а) двостепени редуктор; б) тростепени редуктор; б) четворостепени редуктор

Поређењем иницијалних и оптималних концепција, очигледно је да се постиже значајно смањење запремине и то за проблем двостепеног редуктора за 10,589%, тростепеног редуктора за 7,463% и за проблем четворостепеног редуктора за 19,975%. Приметно је да смањење запремине не зависи само од броја степени преноса, већ и од димензија зупчастих парова. Ови резултати су веома значајни јер само позиционирањем оса вратила зупчаника могуће је смањити запремину редуктора у овој мери. За ово истраживање адаптиран је и примењен математички модел који узима у обзир сва реална ограничења. Примењен је *DINDI* алгоритам, који је свој квалитет показао и за примену код практичних оптимизационих проблема. Овај пример представља практичну оптимизацију машинских конструкција, што је веома комплексан и захтеван процес. Потребно је добити практично прихватљива и применљива решења, што је и постигнуто овим процесом оптімизације. Применом добијених решења постиже се мања запремина редуктора, мање габаритне мере, смањење утрошеног материјала, лакша конструкција и друго. Оптимизација проблема редуктора представља стално интересовање истраживача, највише због масовне примене ове конструкције. Могуће је приступити оптимизацији редуктора са пуно различитих аспеката, а у овом истраживању представљен је један алтернативни приступ.

6.2 Одређивање зазора елемената циклоредуктора применом хеуристичких метода оптимизације

У процесу машинског конструисања инжењер стално тежи да смањи апроксимацију реалне репрезентације конструкције коју развија. Што верније теоријски формализује конструкцију, то ће и рад машинске конструкције бити бољи, јер ће понашање конструкције у пракси бити скоро идентично планираном понашању. Циклоредуктор је машинска конструкција која има све значајнију примену у инжењерској пракси. Ова конструкција је такође значајна за развој и истраживање јер садржи све елементе који се уобичајено користе у конструисању, па може послужити као пример едукације

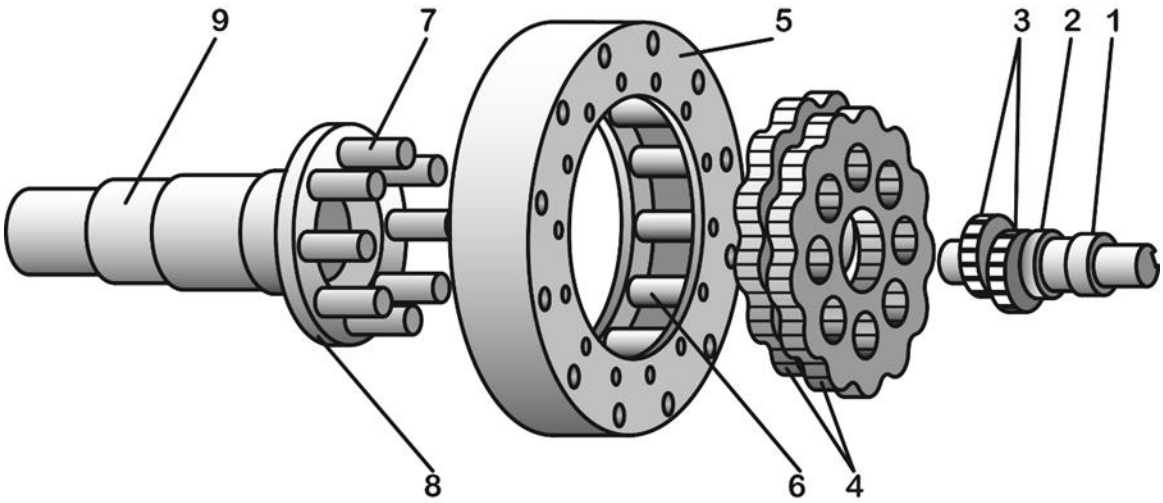
конструкторима. Циклоредуктор представља конструкцију која није у потпуности испитана и зато привлачи велику пажњу истраживачке јавности.
Развој циклоредуктора подразумева постизање добрих радних карактеристика овог преносника снаге. Под добрим карактеристикама подразумева се адекватна геометрија, задовољавајући габарити и маса, повољне кинематске и динамичке карактеристике, висок степен искоришћења, велика носивост и дуг експлоатациони и животни век. Ове карактеристике зависе и од начина конструисања циклоредуктора и од начина његове израде. Контролисањем и управљањем процесом конструисања и технологијом израде постижу се поменуте карактеристике. Један од највећих проблема циклоредуктора, мада и свих других конструкција, представљају толеранције мера елемената циклоредуктора. Ово се посебно односи на елементе у контакту који преносе снагу. Није могуће израдити елементе са апсолутно тачним мерама, а сви теоријски модели подразумевају да су елементи апсолутно тачни. Дакле, основни задатак је предвидети шта се дешава ако елементи циклоредуктора нису апсолутно тачно израђени. Контакт међу овако спрегнутим елементима могуће је третирати као контакт између коригованог профила циклозупчаника и ваљака циклоредуктора. Ово се може дефинисати као геометријска зависност, а применом оптимизације могуће је дефинисати најмање растојање између елемената, што и представља контакт. Овакав приступ је апсолутно хеуристички, што представља перспективу решавања изузетно комплексних инжењерских проблема. Поред тога, за решавање овог проблема примењује се и хеуристичка метода оптимизације. Приступ је у потпуности оригиналан и развијен је за потребе ове дисертације, а за велики број проблема представља и једини могући начин да се уопште и постигне решење. Методологија је верификована репрезентативним примерима, где је извршена упоредна анализа при различитим величинама корекције циклозупчаника у односу на теоријски модел.
У модерној индустрији због својих изузетно повољних карактеристика циклоредуктори имају све већу примену. Ово се односи на висок степен корисног дејства, компактну конструкцију, велики преносни однос, једноставно одржавање, високу поузданост и друге погодности преносника. Основни елементи циклоредуктора су циклозупчаник и ваљци централног зупчаника. За профил циклозупчаника у пракси се најчешће користи крива еквидистанта скраћене епитрохоиде [172-175]. Ретко, у неким специфичним ситуацијама, због постизања бољих радних карактеристика користе се и различити модификовани профили [176, 177]. Највећи проблем циклоидног озубљења представља његова геометрија [178-182], и ова област је веома истражена што омогућава лакшу примену циклоредуктора. Поред геометрије, за процес конструисања веома је важна и визуелизација 2D и 3D [183-185] профила циклозупчаника. Да би проширили примену циклоредуктора и понудили боље радне карактеристике, поједини аутори [186-189] су се бавили развојем нових концепција циклоредуктора. Ради постизања оптималних карактеристика преносника вршена је оптимизација [190, 191] у правцу постизања оптималних параметара профила зубаца циклоредуктора, степена искоришћења, расподеле напона и других параметара. Веома важно за ово истраживање је знати да се у скоро свим теоријским анализама циклоредуктора полази од претпоставке да су сви зупци циклозупчаника у контакту са одговарајућим ваљцима централног зупчаника и да половина од њих преносе оптерећење [192]. Код реалне (стварне) конструкције циклоредуктора то заправо није случај. Између елемената циклоредуктора, ваљака централног зупчаника и циклозупчаника, постоје одређени зазори и то на нивоу толеранција израде – компензација грешака израде и ради обезбеђивања адекватних услова подмазивања, монтаже и демонтаже елемената. На основу тога могуће је извести закључак да у стварности оптерећење не преноси половина зубаца циклозупчаника. Ово представља потпуно нову, широку област истраживања, којом се делом бави и овај

докторат на нивоу одређивања зазора између елемената у контакту. Бланш (*J.G. Blanche*) и Јанг (*D.C.H. Yang*) [193, 194] су се у својим истраживањима бавили утицајем зазора на пулсирање обртног момента, преносног односа и осталих параметара циклоредуктора. Аутори су дефинисали функционалне зависности између погонског угла и анализираних излазних параметара преносника. Увели су и појам угла кашњења и извели су одговарајуће математичке изразе за његов прорачун. Ли Ликсинг (*Li Lixing*) је за прорачун зазора између зубаца циклозупчаника и ваљака централног зупчаника дефинисао многобројне математичке моделе. Поред тога [195] радио је на дефинисању одговарајућих еластичних деформација елемената који учествују у процесу преношења оптерећења. Такође се бавио и прорачуном сила које дејствују на зупце циклозупчаника у случају постојања зазора. Аутори Хсих (*Hsieh*) и Ли (*Lee*) [196] су развили софтвер за анализу кинематске грешке код двостепеног циклоредуктора. Овај софтвер су директно повезали са *CAD* окружењем.

Чињеница да од ових истраживања није направљен значајан помак у области одређивања зазора између елемената циклоредуктора указује да је овај проблем веома комплексан и аналитички готово и нерешив.

Пошто постоји разлика између теоријског и стварног модела циклозупчаника неопходно је анализирати зазоре између елемената циклозупчаника. Зазори се јављају услед конструкционих толеранција и тачности израде елемената. Приступом познавања и анализе зазора могуће је смањити тачност израде, појефтинити производњу и постићи адекватне радне карактеристике преносника. Зато је ова анализа изузетно важна. Приликом појаве зазора, за разлику од теоријског модела, само један ваљак стварно преноси оптерећење који је у контакту са циклозупчаником. Оваква појава драстично мења карактеристике преносника у односу на теоријске претпоставке и прорачун. Дакле, постоји потреба да се разматра величина зазора између елемената циклоредуктора, а ово је могуће реализовати анализом коригованог профила циклозупчаника. Корекције при којима се јављају зазори могуће је реализовати на више различитих начина [178], као што су кориговање профила циклозупчаника, повећање пречника кружнице по којој су распоређени ваљци централног зупчаника и смањење пречника ваљака. На све ове начине могуће је постићи исту слику преносника, а за ово истраживање одабран је приступ кориговања профила циклозупчаника.

Поступак одређивања зазора између елемената циклоредуктора, циклозупчаника и ваљака централног зупчаника, могуће је анализирати као геометријски проблем. Елементи конструкције циклоредуктора су представљени на слици 6.7.



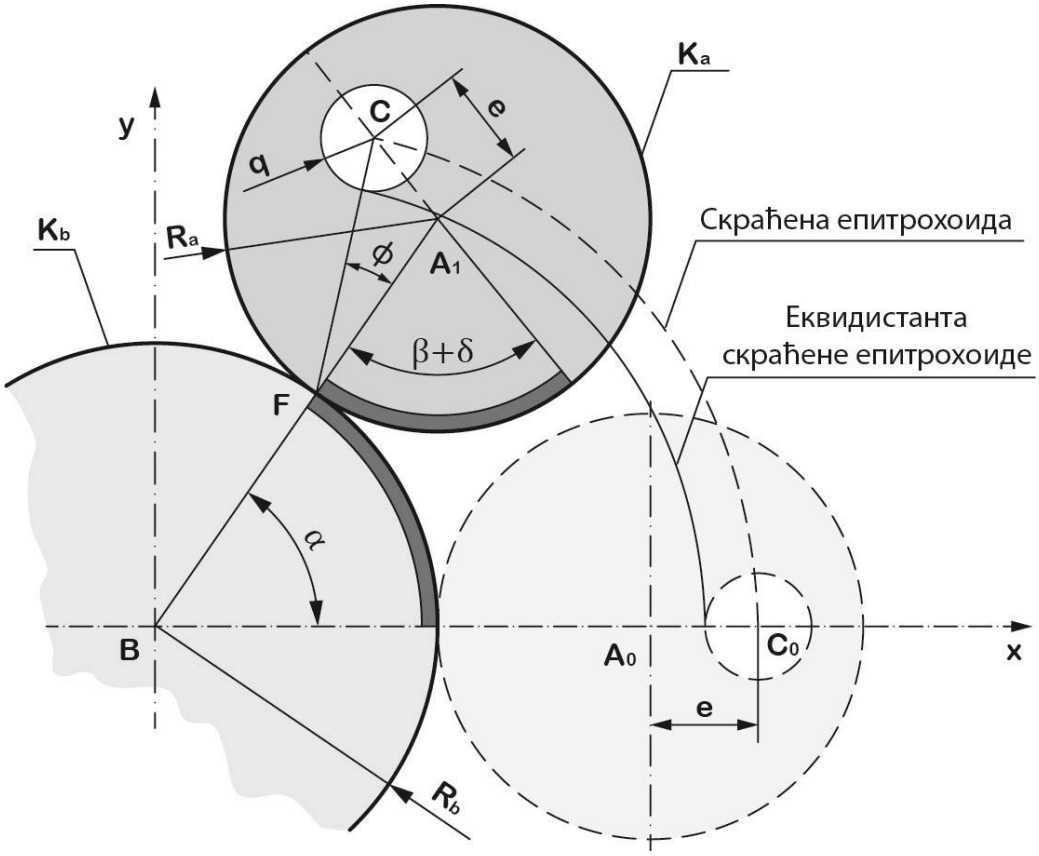
Слика 6.7. Структура циклоредуктора [178] 1) улазно вратило; 2) ексцентар; 3) лежај; 4) циклозупчаници; 5) прстен централног зупчаника; 6) ваљци централног зупчаника; 7) излазни ваљци; 8) носач излазних ваљака; 9) излазно вратило

Еквидистанта скраћене епитрохоиде представља криву линију која дефинише профил циклозупчаника. Ову криву могуће је математички формулисати [178] као у изразу (6.4). Концепт одређивања зазора елемената циклоредуктора развијен је на основу његове геометрије и зато је неопходно искористити поменуту једначину која геометријски формулише проблем.

$$\begin{aligned} x_c &= (R_b + R_a) \cdot \cos\alpha + e \cdot \cos(\alpha + \beta) - q \cdot \cos(\alpha + \phi), \\ y_c &= (R_b + R_a) \cdot \sin\alpha + e \cdot \sin(\alpha + \beta) - q \cdot \sin(\alpha + \phi). \end{aligned} \tag{6.4}$$

где су:
R_a – полупречник котрљајуће кружнице,
R_b – полупречник основне кружнице,
α – угао међусобног положаја почетне и тренутне тачке додира основне и котрљајуће кружнице у односу на центар основне кружнице,
β – угао заокретања котрљајуће кружнице (слика 6.8),
e – величина ексцентрицитета,
q – полупречник ваљака централног зупчаника – еквидистантно растојање,
ϕ – помоћни угао.

Профил криве еквидистанте скраћене епитрохоиде добијен је на основу величина са слике 6.8. На слици је геометријски представљен поступак генерисања ове криве. Величина K_b представља (слика 6.8) основну кружницу, K_a котрљајућу кружницу, δ представља помоћни угао за описивање почетног положаја.



Слика 6.8. Генерисање еквидистанте скраћене епитрохоиде

Како би математичка формулација криве циклозупчаника била погодна за генерисање математичког модела извршена је адаптација израза (6.4). Пошто је познато да су:

$$\beta = \frac{R_b}{R_a} \cdot \alpha,$$
$$\phi = arctan\left(\frac{\sin\left(\frac{R_b}{R_a} \cdot \alpha\right)}{\frac{R_a}{e} + \cos\left(\frac{R_b}{R_a} \cdot \alpha\right)}\right), \tag{6.5}$$

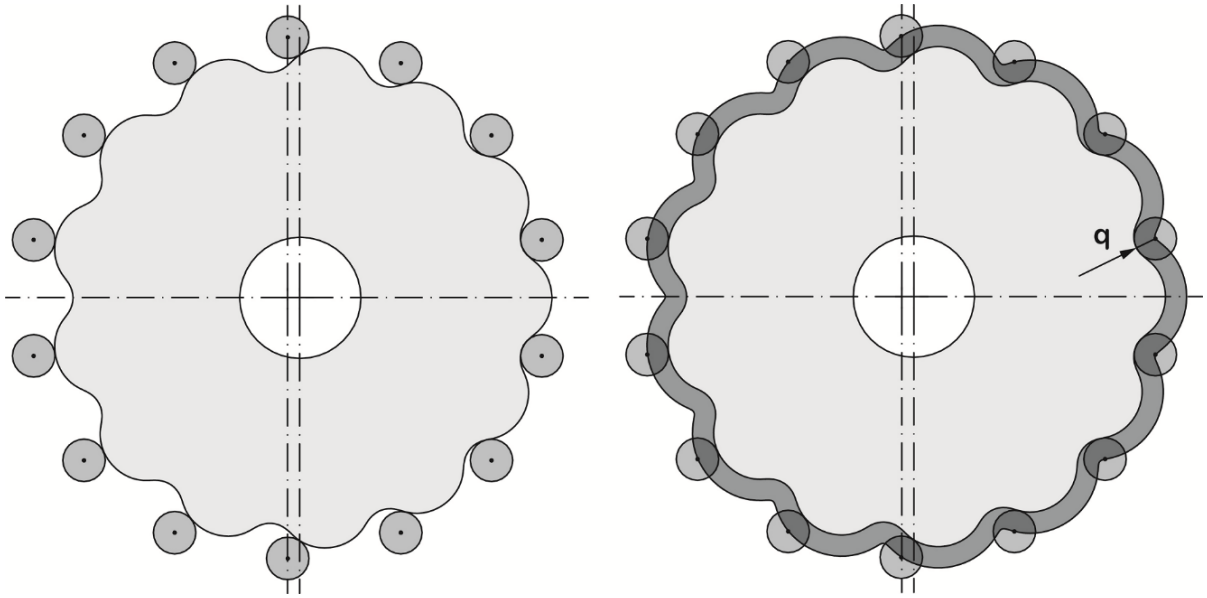
Из претходних једначина (6.4) и (6.5) могуће је написати:

$$x_c = (R_b + R_a) \cdot \cos\alpha + e \cdot \cos\left(\alpha + \frac{R_b}{R_a} \cdot \alpha\right) - q$$
$$\cdot \cos\left(\alpha + arctan\left(\frac{\sin\left(\frac{R_b}{R_a} \cdot \alpha\right)}{\frac{R_a}{e} + \cos\left(\frac{R_b}{R_a} \cdot \alpha\right)}\right)\right) \tag{6.6}$$

$$y_c = (R_b + R_a) \cdot \sin\alpha + e \cdot \sin\left(\alpha + \frac{R_b}{R_a} \cdot \alpha\right) - q \cdot \sin\left(\alpha + arctan\left(\frac{\sin\left(\frac{R_b}{R_a} \cdot \alpha\right)}{\frac{R_a}{e} + \cos\left(\frac{R_b}{R_a} \cdot \alpha\right)}\right)\right) \tag{6.6}$$

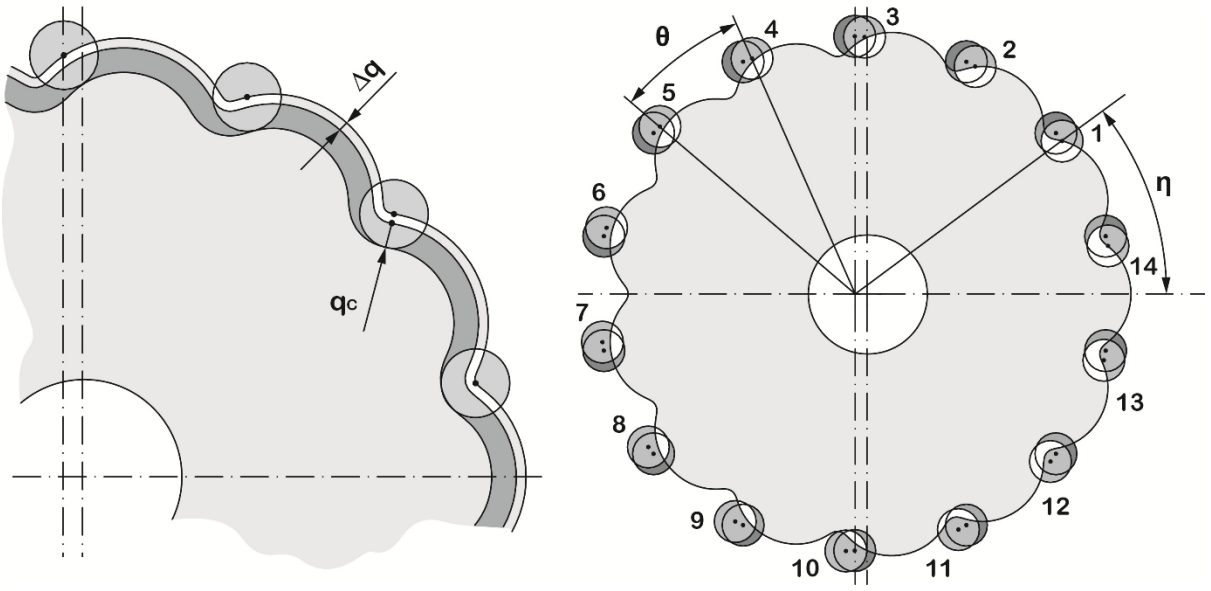
Једначине (6.6) представљају основу за креирање математичког модела. Њиховом применом и евалуацијом помоћу оптимизационих метода могуће је одредити зазоре код елемената циклоредуктора.

Пошто зазор представља минимално растојање између ваљака централног зупчаника и циклозупчаника, то значи да је потребно креирати такав математички модел који репрезентује растојање и минимизовати ову функцију циља. Математички модел представља потпуно оригинални приступ овом проблему и развијен је на основу геометрије елемената циклоредуктора. Основа теоријског модела дефинише контакт свих зубаца циклозупчаника са ваљцима циклоредуктора што је представљено на слици 6.9 а). Могуће је поставити еквидистантно растојање у односу на криву зупчаника и то растојање пролази кроз центре свих ваљака, слика 3 б).



Слика 6.9. а) Теоријски контакт елемената циклоредуктора; б) Еквидистантни положај циклозупчаника који пролази кроз центре свих ваљака

Математички модел креиран је тако да се одређује растојање између тачке центра ваљка и криве циклозупчаника циклоредуктора. Корекцијом профила криве циклозупчаника за неку величину сваки центар ваљка постављен је на истом растојању од криве циклозупчаника, што је представљено на слици 6.10 а). Како би преносник био функционалан неопходно је да циклозупчаник буде у контакту са бар једним ваљком централног зупчаника, што је представљено на слици 6.10 б). Математички модел развијен је тако да тачке центара циклозупчаника могу да се крећу по кружници по којој су распоређени ваљци до остварења стварног контакта.



Слика 6.10. а) Корекција профила – еквидистантно растојање са удаљењем од центара ваљака; б) Заокретање центара ваљака по кружници на којој су распоређени, до остваривања контакта

Изразом (6.7) могуће је представити основни приступ за креирање математичког модела:

$$min\sqrt{X^2 + Y^2} \tag{6.7}$$

Овај израз представља приступ одређивања геометријског растојања између циклозупчаника и ваљака централног зупчаника. Тачније, израз представља минимално растојање између елемената, где су координатама представљена растојања између елемената по осама X и Y.

$$\begin{aligned} X &= x_r - x_c, \\ Y &= y_r - y_c. \end{aligned} \tag{6.8}$$

Где су:

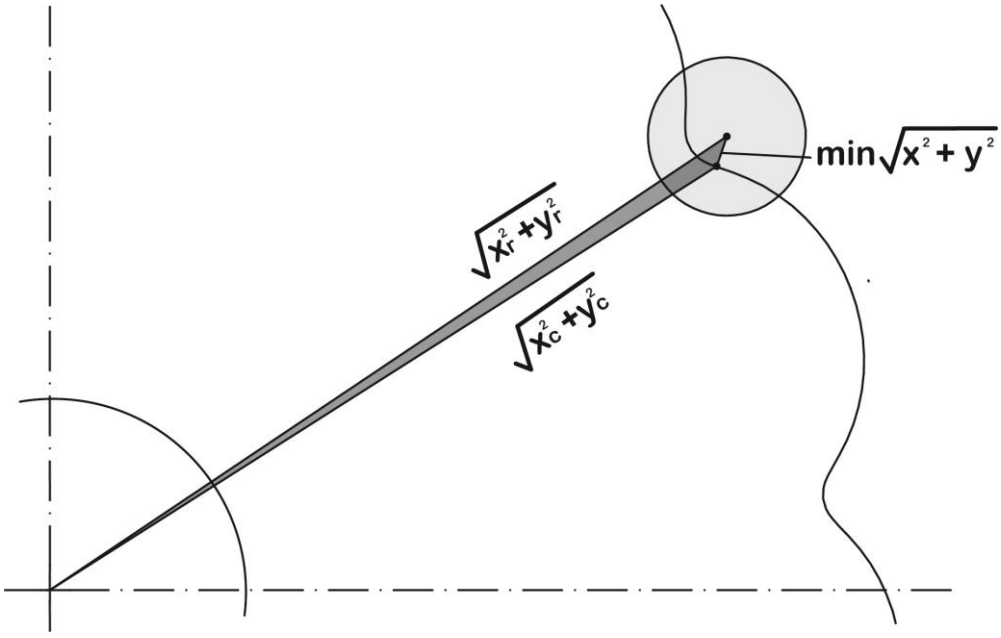
$$\begin{aligned} x_r &= (R_b + R_a) \cdot \cos(\eta + n \cdot \theta), \\ y_r &= (R_b + R_a) \cdot \sin(\eta + n \cdot \theta), \end{aligned} \tag{6.9}$$

$$\begin{aligned} x_c = (R_b + R_a) & \\ & \cdot \cos(\alpha) + e \cdot \cos\left(\alpha + \frac{R_b}{R_a} \cdot \alpha\right) - (q - q_c) \\ & \cdot \cos\left(\alpha + arctan\left(\frac{\sin\left(\frac{R_b}{R_a} \cdot \alpha\right)}{\frac{R_a}{e} + \cos\left(\frac{R_b}{R_a} \cdot \alpha\right)}\right)\right), \end{aligned} \tag{6.10}$$

$$y_c = (R_b + R_a) \cdot \sin(\alpha) + e \cdot \sin\left(\alpha + \frac{R_b}{R_a} \cdot \alpha\right) - (q - q_c) \cdot \sin\left(\alpha + arctan\left(\frac{\sin\left(\frac{R_b}{R_a} \cdot \alpha\right)}{\frac{R_a}{e} + \cos\left(\frac{R_b}{R_a} \cdot \alpha\right)}\right)\right),$$

η – променљива оптимизације, дефинише вредности угла на којим се налази центар ваљка који се спреже са циклозупчаником,
n – редни број ваљка за који се израчунава растојање (за контакт је вредност 0),
$\theta = \frac{2\pi}{z_2}$ – угао за који су заокренути ваљци (z_2 = укупан број ваљака),
q_c – полупречник коригованог ваљка централног зупчаника – кориговано еквидистантно растојање.

Индексом *r* су означене величине које се односе на ваљак, а индексом *c* величине које се односе на циклозупчаник. На слици 6.10 представљене су поменуте величине. Као променљиве оптимизације учествују угао међусобног положаја почетне и тренутне тачке додира основне и котрљајуће кружнице у односу на центар основне кружнице, чији је интервал $0 \leq \alpha \leq 2\pi$ и променљива која дефинише вредности угла на којем се налази центар ваљка који се спреже са циклозупчаником, у интервалу $0 \leq \eta \leq 2\pi$.
Величина растојања које је потребно минимизовати између еквидистанте циклозупчаника и центра ваљка централног зупчаника представљена је на слици 6.11. Минимална вредност овог растојања може бити 0, што заправо представља контакт између коригованог профила циклозупчаника и ваљка централног зупчаника циклоредуктора.



Слика 6.11. Растојање између еквидистанте циклозупчаника и центра ваљка централног зупчаника које је потребно минимизовати

Применом претходно дефинисаног математичког модела из једначине (6.8), може се одредити контакт између циклозупчаника и ваљака централног зупчаника, али је неопходно одредити и прихватљива решења или решења где се ни на једном ваљку неће појавити преклопи између елемената. Овим приступом добија се скуп потенцијалних решења које је неопходно испитати и проверити њихову практичну применљивост. Без провере ограничења, применом само функције циља биле би могуће појаве преклопа између елемената, што не репрезентује реалан случај. То је разлог зашто су ограничења неопходна за валидацију овог приступа.

6.2.1 Примери за тестирање развијеног математичког модела

На основу креираног математичког модела, ради верификације развијеног поступка, саставни део овог истраживања представља и репрезентација практичне примене. За пример је узет једностепени циклоредуктор, а корекције су извршене са различитим вредностима. Циклозупчаник је коригован за 0,05 *mm*, 0,1 *mm* и 0,3 *mm*. Примери презентују могућност примене развијеног математичког модела, софтвера и *DINDI* методе у складу са реалним потребама оптимизације машинских конструкција. Прорачунске вредности реалног циклоредуктора потребне за оптимизацију представљене су у табели 6.3.

Табела 6.3

Прорачунске вредности параметара циклоредуктора

Опис величине	**Ознака**	**Вредност**
Преносни однос	u	13
Полупречник котрљајуће кружнице *mm*	R_a	6,143
Полупречник основне кружнице *mm*	R_b	79,857
Величина ексцентрицитета *mm*	e	4
Полупречник ваљака централног зупчаника – еквидистантно растојање *mm*	q	6,88
Број зубаца циклозупчаника	z_1	13
Број ваљака централног зупчаника	z_2	14
Помоћна величина *mm*	$R_a + R_b$	86
Помоћна величина	R_b/R_a	12,9996744
Помоћна величина	R_a/e	1,53571425
Полупречник коригованог ваљка централног зупчаника – кориговано еквидистантно растојање *mm*	q_c	6,83, 6,78, 6,58
Величина корекције профила *mm*	$\Delta q = q - q_c$	0,05, 0,1, 0,3
Угао међусобног положаја ваљака *rad*	θ	0,44879895

Код теоријског модела сви зупци циклозупчаника остварују контакт са ваљцима, а половина зубаца преноси оптерећење. Према изразу (6.8), што представља функцију циља, за решавање представљеног проблема израз је могуће извести за конкретну вредност корекције профила циклозупчаника. За конкретне квантитативне вредности

функције циља врши се минимизација *DINDI* алгоритмом и постиже се скуп потенцијалних решења. Решења представљају вредности угла на коме се остварује контакт елемената циклоредуктора, где су прихватљива само она решења код којих нема појаве преклопа између циклозупчаника и ваљака централног зупчаника.

6.2.2 Резултати оптимизације циклоредуктора

Математички модел формулисан је на основу геометријских карактеристика циклоредуктора. За оптимизацију је коришћен развијени софтвер и развијени *DINDI* алгоритам. Пошто се ради о контакту елемената, где растојања могу бити веома мала а да контакт не постоји, тачност коју подразумева ова анализа је седам децимала. Величина *FEs* – а није важна за ову анализу. За ову величину узете су велике вредности како би била постигнута висока тачност, а ефикасност саме методе је већ анализирана. Скуп потенцијалних решења јесу све пресечне тачке кружнице на којој су распоређени ваљци са циклозупчаником. Ове вредности представљене су у табели 6.4.

Табела 6.4
Скуп потенцијалних контаката

Скуп решења за корекцију од 0,3 *mm*								
Угао положаја ваљака у контакту, °	12,431	14,015	38,261	41,406	63,970	69,876	89,631	99,838
	115,224	131,585	140,602	219,406	228,426	244,784	260,172	270,377
	290,133	296,039	318,603	321,747	345,994	347,578		
Скуп решења за корекцију од 0,1 *mm*								
Угао положаја ваљака у контакту, °	12,697	13,744	38,468	41,183	64,182	69,629	89,880	99,512
	115,556	131,021	141,169	164,477	166,097	193,910	195,535	218,839
	228,989	244,453	260,498	270,129	290,380	295,827	318,826	321,540
	346,265	347,312						
Скуп решења за корекцију од 0,05 *mm*								
Угао положаја ваљака у контакту, °	12,773	13,666	38,520	41,127	64,235	69,568	89,941	99,432
	115,637	130,889	141,302	163,854	166,721	193,287	196,158	218,706
	229,121	244,372	260,578	270,067	290,441	295,774	318,882	321,488
	346,343	347,235						

Решења која су постигнута неопходно је проверити тако да не постоји преклоп између неког ваљка и циклозупчаника. Ово је могуће извршити на два начина. Први начин је интеграцијом израза (6.11) у математички модел, чиме се аутоматизује процес избора допуштених решења.

$$x_{cmin} \leq x_r \leq x_{cmax}$$
$$y_{cmin} \leq y_r \leq y_{cmax} \qquad (6.11)$$

Други приступ, који је коришћен у овом истраживању је примена *CAD* софтвера (*Autodesk Inventor*), повезаног са развијеним оптимизационим софтвером. Оптимизациони софтвер је искоришћен да одреди апсолутну вредност растојања за

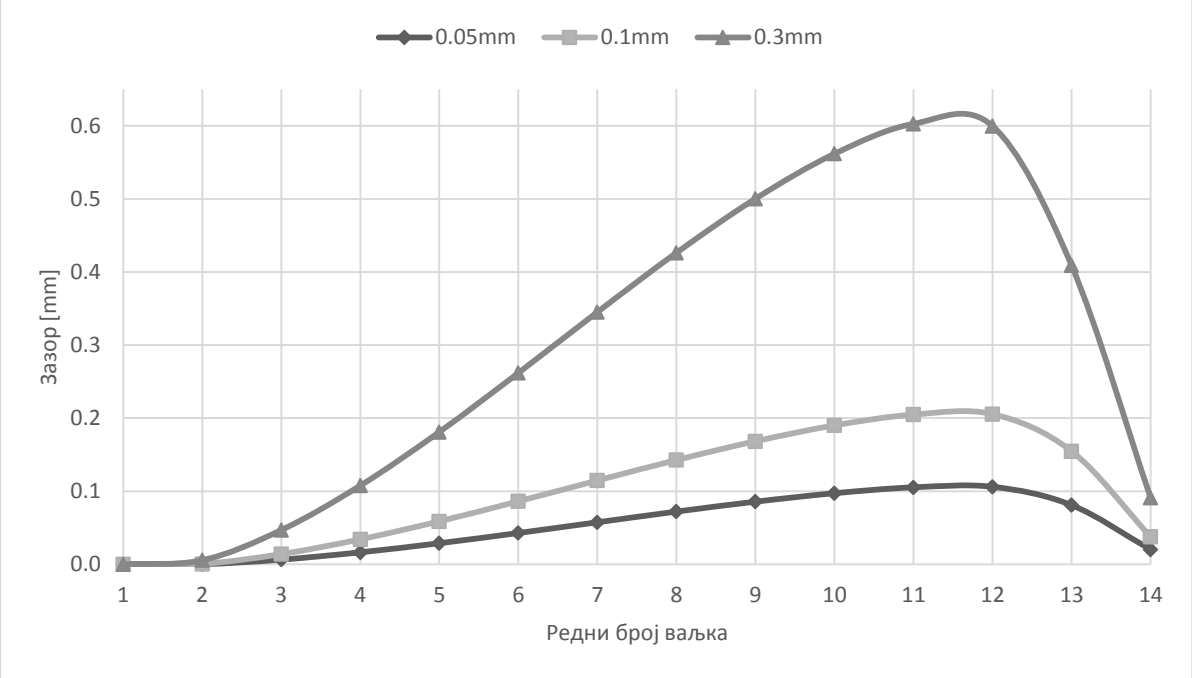
сваки ваљак од циклозупчаника, а у *CAD* софтверу је креиран параметарски модел у који се уносе оптимизацијом добијене вредности и констатује се да ли су растојања позитивна или негативна, тачније да ли постоји преклоп између елемената. Сва решења где бар један ваљак из табеле 6.4 има негативно растојање морају бити одбачена јер није практично прихватљиво. Практично, ако елементе циклоредуктора посматрамо као апсолутно крута тела, контакт је остварен искључиво између једног зупца циклозупчаника и ваљка, а остали зупци се налазе на одређеном растојању од ваљака централног зупчаника. Ваљак под редним бројем 1 у табели 6.5 је онај ваљак који остварује контакт са циклозупчаником, а остали ваљци нумерисани су у позитивном математичком смеру. Ова решења представљају једина прихватљива решења за реалну употребу.

Табела 6.5

Зазори између елемената циклоредуктора код коригованог профила циклозупчаника

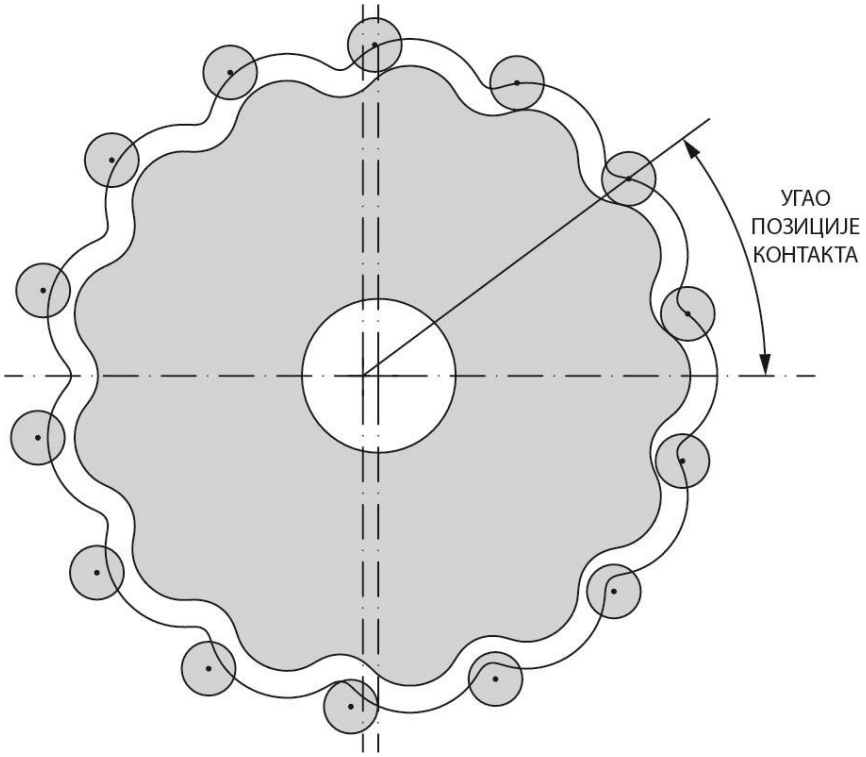
профил коригован за 0,05 *mm*			**профил коригован за 0,1 *mm***			**профил коригован за 0,3 *mm***		
Редни број ваљка	Угао положаја ваљака °	Растојање од циклоредуктора *mm*	Редни број ваљка	Угао положаја ваљака °	Растојање од циклоредуктора *mm*	Редни број ваљка	Угао положаја ваљака °	Растојање од циклоредуктора *mm*
1	38,520	0,0000000	1	38,468	0,0000000	1	38,261	0,0000000
2	64,234	0,0001034	2	64,182	0,0006017	2	63,975	0,0055156
3	89,949	0,0063563	3	89,897	0,0139653	3	89,690	0,0468390
4	115,663	0,0163896	4	115,611	0,0341669	4	115,404	0,1077171
5	141,377	0,0288215	5	141,325	0,0588210	5	141,118	0,1811158
6	167,092	0,0427985	6	167,040	0,0862049	6	166,833	0,2618805
7	192,806	0,0575030	7	192,754	0,1147003	7	192,547	0,3452111
8	218,520	0,0720919	8	218,468	0,1426730	8	218,261	0,4263282
9	244,234	0,0856738	9	244,182	0,1684298	9	243,975	0,5003464
10	269,949	0,0972395	10	269,897	0,1900837	10	269,690	0,5618578
11	295,663	0,1053207	11	295,611	0,2048959	11	295,404	0,6028518
12	321,377	0,1060969	12	321,325	0,2055595	12	321,118	0,5998447
13	347,092	0,0810017	13	347,040	0,1548181	13	346,833	0,4091832
14	372,806	0,0201856	14	372,754	0,0377965	14	372,547	0,0913279

Очигледно да *DINDI* алгоритам и код овог проблема показује изузетне перформансе, што значи да га је могуће примењивати код практичних инжењерских проблема. Код решења представљених у табели 6.5 само један ваљак централног зупчаника се налази у контакту са циклозупчаником и његово растојање је 0 *mm*. Између осталих ваљака и циклозупчаника постоји зазор. На слици 6.12 представљени су зазори који се јављају између ових елемената за сва три примера.



Слика 6.12. Зазори између ваљака централног зупчаника и циклозупчаника за корекцију од 0,05 *mm*, 0,1 *mm* и 0,3 *mm*

Вредности величине корекције (0,05 *mm*, 0,1 *mm* и 0,3 *mm*) изабране су тако да се покаже да је корекцију могуће извршити за било коју величину, према реалним потребама толеранције и могућности израде елемената циклоредуктора и како би на репрезентативан начин представиле методолошки поступак одређивања ових величина. На слици 6.13 представљено је позиционирање контакта код коригованог профила циклозупчаника. Заокретање кружнице по којој су распоређени центри ваљака централног зупчаника врши се до остваривања контакта.



Слика 6.13. Позиционирање контакта на основу решења оптимизације

Постоји велика перспектива примене ових резултата и директне интеграције овог поступка у *CAD* окружење. Тиме је могуће анализирати и конструисати циклоредукторе са реалном сликом контакта и зазора која се заправо јавља у пракси. Овај приступ презентује потребу да се интегрише процес оптимизације код развоја машинских конструкција, за њихову практичну и рентабилну примену. Ово истраживање представља хеуристички приступ којим је могуће решити комплексне инжењерске проблеме на једноставан начин. Решење проблема одређивања реалног зазора елемената циклоредуктора представља значајан корак у практичној примени преносника снаге са нееволвентним озубљењем. Такође, ова анализа има утицај на побољшање перформанси циклоредуктора и смањење производних трошкова. Овај приступ отвара велики истраживачи простор за унапређење циклоредуктора као алтернативног преносника снаге.

7. Закључак

Оптимизација у општем смислу представља веома важан, значајан и перспективан процес јер може допринети бољем и квалитетнијем животу, балансу и складу, а тиме дати допринос у еволуцији цивилизације и човечанства. Процес оптимизације је могуће применити на апсолутно све проблеме када је потребно донети било какву одлуку. Могуће је уочити ситуацију када је потребно применити овај процес, а могуће је и наметнути процес оптимизације како би нека појава била унапређена. Једна од области која данас не може да се замисли без примене оптимизације јесте инжењерство, где спада и област машинских конструкција. Потребно је много радити на унапређењу оптимизационог процеса, како би он био доступнији инжењерима и како би се комерцијализовао, јер са тренутним стањем неопходно је превелико искуство корисника.

Историјски, оптимизација постоји колико и човечанство, а може се рећи и дуже од тога, јер већина природних процеса и појава има оптималне зависности и начине функционисања. Ако се говори о самом појму оптимизације, када овај процес постаје свестан и циљно примењиван, оптимизација је релативно млада наука и дисциплина. Први кораци у развоју ове области оријентисани су на доступна математичка сазнања. Дуго после тога развој оптимизације заснован је искључиво на математици, искључујући интуицију у потпуности. Са појавом рачунара, процес оптимизације доживљава велики напредак, а оријентација и сам правац оптимизације добијају потпуно нови облик. Тај тренутак је кључан за појаву и развој хеуристичке оптимизације која данас потпуно преовладава у овој области. На самом почетку овај процес је био интересантан, али није уочен као потпуно перспективан. Стручњаци су упорно покушавали да и хеуристичке методе потпуно математички формулишу и аналитички докажу конвергенције, чиме хеуристика губи смисао и удаљава се од суштине. Напредовањем у развоју хеуристичких метода оптимизације и њиховом практичном применом, доказано је колико предности ова група оптимизационих метода има у односу на остале.

Важно је знати да оптимизација мора бити математички дефинисана математичким моделом. Функција циља, ограничења и промењљиве представљају саставни део оптимизације који се не мења без обзира на развој. Данас је процес оптимизације незамислив без употребе рачунара. Рачунари омогућавају да на првом месту комплексни проблеми буду решиви, а затим да уложени напори у процесу оптимизације буду мањи од коначног доприноса овог процеса. Процес имплементације своди се на три кључне фазе: предпроцесирање, процесирање и постпроцесирање. Прва фаза подразумева дефинисање проблема и адаптацију методе проблему или проблема методи. Друга фаза, процесирање, чини поступак рада методе и конвергенције ка оптималном решењу. Трећа фаза представља визуелизацију и анализу оптимизационих резултата.

Циљ ове дисертације је да се постигне напредак у области хеуристичке оптимизације, са освртом на примену проблема машинских конструкција. Било је потребно анализирати хеуристичке методе оптимизације које постоје из ове групе метода издвојити неке од најважнијих. Уочено је да оваква селекција и анализа метода може допринети развоју у овој области. Развој подразумева рад на модификацијама, хибридизацији и развоју потпуно новог алгоритма хеуристичке оптимизације. Потребно је утврдити да ли развој представља допринос, испитивањем самих метода и поређењем са атрактивним методама хеуристичке оптимизације које постоје. Метода која је доминантна је примењена на проблеме машинских конструкција, где је важно дефинисати неку добит

у односу на иницијалне конструкције. Конструкције оптималних карактеристика морају бити нешто боље у односу на иницијалне. Комплетан овај процес спроведен је тако да се представи простор за примену метода хеуристичке оптимизације и да се укаже на алтернативу даљих истраживања.

У уводном делу ове дисертације дефинисан је општи појам оптимизације, његов значај и примена, затим је представљен историјски развој оптимизације. Уводни део садржи и опис математичке формулације оптимизације, представљање група метода математичке оптимизације, њихову нумеричку имплементацију и дефинисање самих циљева, мотива и потреба за развој који је обухваћен.

Други део дисертације односи се на хеуристичке методе оптимизације, издвајање историјски најзначајнијих метода оптимизације, представљање актуелних и атрактивних метода и издвајање метода које могу утицати на развој. У овом делу детаљније су представљене методе *GA, PSO* и *TLBO*, јер су ове методе касније и употребљене за практичну примену.

Акценат ове дисертације је на модификацијама, хибридизацији и развоју потпуно нове хеуристичке методе. Сва решења представљена у дисертацији су нова и потпуно оригинална. Допринос је представљен кроз модификације *iGA* и *rTLBO*, хибридну методу *hGPT* и нови *DINDI* алгоритам. Важно за ову целину било је разумевање хеуристике као појаве, на основу чијих принципа су методе и развијене.

Комплетна структура софтвера развијена је као оригинална и развијене методе су интегрисане у њега. Сви ови задаци појединачно (развој метода, развој софтвера, имплементација метода) представљају комплексне задатке, а постигнути резултати доприносе.

Ова дисертација обухвата и анализу рада развијених метода хеуристичке оптимизације, а тиме и тестирање развијеног софтвера за оптимизацију. У овај процес укључени су сви типови комплексних оптимизационих проблема (без ограничења, са ограничењима и инжењерски проблеми). Поред тога коришћен је велики број актуелних хеуристичких метода из литературе за упоредну анализу. Анализа ефикасности и квалитета рада развијених метода извршена је према препорукама експерата из ове области и према устаљеним процесима. Извршена су обимна и дуготрајна испитивања, са великим (захтеваним) бројем понављања прорачуна и симулација. Овај процес је захтеван и дуготрајан, а постигнути резултати су веома значајни, имајући у виду уложене напоре.

Смисао целокупног рада на оптимизационом процесу, развоју нових метода, развоју софтвера, верификацији њиховог рада, јесте примена ових метода на машинске конструкције. Основни циљ је развој конструкција оптималних перформанси. Код постојећих конструкција потребно је обезбеђење њиховог бољег рада, а тиме и показати простор у коме је могуће применити и интегрисати процес хеуристичке оптимизације у машинском конструисању. Овај приступ је представљен кроз допринос на конструкцијама конвенцијалног редуктора и на атрактивном преноснику снаге – циклоредуктору.

Могуће је извести закључак да оптимизација представља велику и перспективну област коју је могуће применити на било коју врсту проблема. Обим ове области подразумева велико и широко знање истраживача и стручњака како би било могуће само користити, а притом и допринети области инжењерске оптимизације. Кроз историју је развијен велики број метода оптимизације, чак и група којима те методе припадају. Може се рећи, а на основу савремених истраживања и доступне литературе и потврдити да група хеуристичких метода оптимизације предњачи у односу на све остале познате групе. Могуће је у перспективи очекивати додатне поделе у оквиру хеуристичких метода и развој њихових појединачних сегмената. За сада математичка формулација проблема у

општем смислу мора остати иста, а оптимизација ће, бар још дуго, функционисати применом неког облика математичког модела.
Чињеница је да постоји велики број веома добрих хеуристичких метода оптимизације, али постоји и потреба за сталним развојем ове области. Прво, проблеми оптимизације који се анализирају су све комплекснији, а поред тог перформансе које се захтевају су све веће. Постоје методе које су доживеле успехе засноване на њиховим карактеристикама и примени код инжењерских проблема. Генетски алгоритам је метода која је најпопуларнија, једна од првих хеуристичких метода оптимизације. Генетски алгоритам је метода која је имплементирана у скоро све комерцијалне оптимизационе софтвере, а доживела је и велики број модификација. Поред ове методе, интересантан је *PSO* алгоритам. Мали број метода развијен је и примећен у периоду између развоја ове две методе. *TLBO* метода представља стварни допринос у области хеуристичке оптимизације јер је појава ове методе мотивисала истраживаче из области да развијају нешто потпуно другачије и ново у области. Од тог тренутка акценат је на развоју нових метода, а мање на модификацијама и хибридизацији, чиме ова област убрзано напредује. Рад ових алгоритама и њихове фазе најзначајнији су у свом изворном облику. Можда изворне методе не пружају најбоље перформансе, али подстичу на креативни развој и стварање метода задовољавајућих карактеристика.
Закључак је да је могуће модификовати, извршити хибридизацију (комбиновати) и развити нову методу хеуристичке оптимизације. Поступак развоја можда делује као логичан, баналан и једноставан, јер се ради о хеуристичким методама. Када се анализира нека развијена хеуристичка метода, она делује потпуно једноставно и логично (презентује појаву која је таква), а веома је тешко схватити колико је напора уложено да се постигне таква логичност и једноставност. Потребно је велико познавање процеса оптимизације, затим познавање развијених метода, њиховог рада и начина конвергенције. Даље је потребно уочити сегменте у којима је могуће остварити допринос, што је и најтежи корак. Код модификација неопходно је уочити слабости методе која се модификује или недостатке у формулацији процеса на основу кога је развијена метода. Код хибрида је неопходно уочити парцијалне слабости метода које чине хибрид и објединити их у функционалну целину. При развоју нове методе потребно је уочити недостатке постојећих хеуристичких метода, уочити потпуно нови хеуристички процес, направити апроксимацију процеса и обезбедити да апроксимација успешно конвергира. Ови поступци су веома тешки и комплексни, али без обзира на то, овим истраживањем су развијене атрактивне модификације метода, хибридна метода и потпуно нова (оригинална) хеуристичка метода оптимизације.
Када год се у истраживањима користе алати или софтвери који су већ развијени, неки део мора остати црна кутија. За процес оптимизације, како би био комплетан и функционалан, било је потребно развити све сегменте који чине оптимизациони софтвер. Овај задатак је захтеван, дуготрајан и тежак, али да није примењен не би било могуће уочити шта је све важно за хеуристичку оптимизацију машинских конструкција. Поред тога рад развијених метода можда не би било могуће у потпуности имплементирати у изворном облику да целокупан софтвер није развијан од почетка. Зато је могуће закључити да без обзира на напоре, развој софтвера директно доприноси развоју хеуристичке оптимизације, а тиме је испуњен стратешки циљ овог рада.
При анализи рада развијених хеуристичких метода и анализи њихове конвергенције потребно је вршити испитивања за проблеме без ограничења, проблеме са ограничењима, а пошто су овде методе развијане за потребе оптимизације машинских конструкција, испитивања је неопходно спровести и на инжењерским проблемима. Да би било могуће уочити да ли метода ради добро и колико добро ради потребно је анализе вршити на познатим проблемима, чија су решења такође позната. Успех рада неке

развијене методе дефинише се у односу на друге методе оптимизације. Према постигнутим резултатима развијени *iGA, rTLBO, hGPT* и *DINDI* алгоритми имају изузетне особине. Првенствено *iGA* ради много боље од самог *GA*, а поред тога и парира најновијим добрим методама хеуристичке оптимизације. Методама *rTLBO* и *hGPT* су такође постигнути жељени ефекти. Сам хибрид *hGPT* без обзира на добре резултате које даје, значајнији је због перспективе и новог приступа који представља за развој хеуристичких метода оптимизације. Алгоритам *DINDI* је у потпуности одговорио на захтеве и представља искорак, првенствено због појаве коју репрезентује, а поред тога и квалитета који поседује. У испитивањима која су спроведена све развијене методе су веома истакнуте и могу бити сврстане у групу најзначајнијих према тренутном стању у области хеуристичке оптимизације.

Што се тиче практичних проблема оптимизације, потребно је поменути да сви инжењерски проблеми представљају практичне проблеме. Интересантни за анализу били су преносници снаге. Овим радом анализирани су редуктори са паралелним осама вратила, коју су конвенционални, и алтернативни циклоидни редуктори.

Код конвенционалног редуктора извршена је анализа и дефинисање општег облика математичког модела новог приступа. Иницијална идеја је да се позиционирањем оса вратила у одговарајућој равни смањи запремина, а тиме и маса и утрошени материјал вишестепеног редуктора. У обзир су узета сва ограничења, а верификација развијеног модела спроведена је на три карактеристична примера применом *DINDI* оптимизационе методе. За испитивања су коришћени двостепени, тростепени и четворостепени редуктори са конкретним и реалним бројним прорачунским вредностима. Доказано је да овом оптимизацијом редуктора постоји могућност да се смањи иницијална запремина за 10,59% код конкретног двостепеног редуктора, за 7,46% код конкретног тростепеног редуктора и за 19,97% код конкретног четворостепеног редуктора. Ове вредности представљају огромне уштеде, нарочито код серијске или масовне производње ових преносника. Редукторе је могуће анализирати још са великог броја аспеката.

Други практични инжењерски проблем који је анализиран је појава зазора код елемената циклоредуктора. Постоје многобројни покушаји да се одреди зазор између еквидистанте скраћене епитрохоиде и ваљака централног зупчаника циклоредуктора. Досадашњи аналитички приступ није давао најбоље резултате и зато је у овом раду коришћен приступ тражења алтернативе. Код циклоредуктора, на основу геометрије самог преносника направљен је целокупни оригинални поступак којим је могуће одредити зазор и растојања између елемената циклоредуктора и дефинисање позиције појаве контакта у спрези. За циклоредуктор на основу једначина које дефинишу профил криве циклозупчаника и на основу геометрије ваљака, развијен је потпуно оригинални приступ и математички модел оптимизације. Како би приступ био верификован, извршена је оптимизација конкретног редуктора за корекцију од 0,05 *mm*, 0,1 *mm* и 0,3 *mm*. Ове вредности су узете за конкретна испитивања, а сам модел је у општем облику и могуће га је применити за било коју величину корекције. Оптимизацијом је добијен скуп потенцијалних решења од којих су нека реална, а нека не могу испунити ограничења. За ово истраживање процес није у потпуности аутоматизован, па се ограничења одређују у посебном софтверу, али је дефинисан приступ којим је могуће у потпуности аутоматизовати овај поступак. Из скупа потенцијалних решења одређују се решења која су практично прихватљива. Ова решења дефинишу угао на коме се налази контакт између елемената циклоредуктора. Растојање између елемената на тој позицији је 0. Даље је извршено одређивање растојања између свих других ваљака и циклозупчаника. Код анализираног примера редуктора, за све величине корекција, постоји само један потенцијални контакт.

Даља истраживања морају узети у обзир ефикасност циклоредуктора са коригованим профилом циклозупчаника, као и корекцију у виду толеранције, која може бити варијабилна по кривој циклозупчаника. Ово решење може отворити велико истраживачко поље, а представља и огроман допринос у развоју самих циклоидних преносника снаге.

Даља перспектива развоја може бити усмерена на процес „оптимизације“ оптимизације. Овим поступком вероватно је могуће створити методу оптималних карактеристика и извршити оптимизацију параметара методе и других виталних делова овог процеса, чиме би оптимизација сама себе усавршавала.

Што се тиче перспективе оптимизације машинских конструкција, развој се отвара само у правцу који је представљен и у делу имплементације при развоју конструкција. Оптимизација код машинских конструкција мора да буде доступна и прихваћена, без обзира колико боље резултате даје. Боље је да напредак буде мали и да постоји, него да је могуће направити велики допринос, али да се то не догађа. Овом дисертацијом отворено је ново поглавље развоја и примене хеуристичке оптимизације машинских конструкција, а простор за даља истраживања је бесконачно велики.

Додатак – тест функције

Додатак А. Математичка формулација коришћених тест функција са ограничењима

Функција са ограничењима 1
(CB1)

Минимизација функције циља:

$$f(X) = 5\sum_{i=1}^{4} x_i - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i$$

Следећи ограничења:

$$g_1(X) = 2x_1 + 2x_2 + x_{10} + x_{11} - 10 \leq 0$$
$$g_2(X) = 2x_1 + 2x_3 + x_{10} + x_{12} - 10 \leq 0$$
$$g_3(X) = 2x_2 + 2x_3 + x_{11} + x_{12} - 10 \leq 0$$
$$g_4(X) = -8x_1 + x_{10} \leq 0$$
$$g_5(X) = -8x_2 + x_{11} \leq 0$$
$$g_6(X) = -8x_3 + x_{12} \leq 0$$
$$g_7(X) = -2x_4 - x_5 + x_{10} \leq 0$$
$$g_8(X) = -2x_6 - x_7 + x_{11} \leq 0$$
$$g_9(X) = -2x_8 - x_9 + x_{12} \leq 0$$

Где су променљиве $0 \leq x_i \leq 1 \ \ (i = 1, \dots, 9);\ \ 0 \leq x_i \leq 100 \ \ (i = 10,11,12);\ \ 0 \leq x_{13} \leq 1$

Функција са ограничењима 2
(CB2)

Минимизација функције циља:

$$f(X) = -\left(\sqrt{n}\right)^n \prod_{i=1}^{n} x_i$$

Следећи ограничења:

$$h_1(X) = \sum_{i=1}^{n} x_i^2 - 1 = 0$$

Где је $n = 10$ а променљиве $0 \leq x_i \leq 1 \ \ (i = 1, \dots, n)$

Функција са ограничењима 3
(CB3)

Минимизација функције циља:

$$f(X) = 5{,}3578547x_3^2 + 0{,}8356891x_1x_5 + 37{,}293239x_1 - 40792{,}141$$

Следећи ограничења:

$$g_1(X) = 85{,}334407 + 0{,}005685x_2x_5 + 0{,}0006262x_1x_4 - 0{,}0022053x_3x_5 - 92 \leq 0$$
$$g_2(X) = -85{,}334407 - 0{,}005685x_2x_5 - 0{,}0006262x_1x_4 - 0{,}0022053x_3x_5 \leq 0$$
$$g_3(X) = 80{,}51249 + 0{,}0071317x_2x_5 + 0{,}002995x_1x_2 + 0{,}0021813x_3^2 - 110 \leq 0$$
$$g_4(X) = -80{,}51249 - 0{,}0071317x_2x_5 - 0{,}002995x_1x_2 - 0{,}0021813x_3^2 + 90 \leq 0$$
$$g_5(X) = 9{,}300961 + 0{,}0047026x_3x_5 + 0{,}0012547x_1x_3 + 0{,}0019085x_3x_4 - 25 \leq 0$$
$$g_6(X) = -9{,}300961 - 0{,}0047026x_3x_5 - 0{,}0012547x_1x_3 - 0{,}0019085x_3x_4 + 20 \leq 0$$

Где су променљиве $78 \leq x_1 \leq 102; 33 \leq x_2 \leq 45;\ 27 \leq x_i \leq 45\ \ (i = 3{,}4{,}5)$,

Функција са ограничењима 4
(CB4)

Минимизација функције циља:

$$f(X) = x_1 + x_2 + x_3$$

Следећи ограничења:

$$g_1(X) = -1 + 0{,}0025(x_4 + x_6) \leq 0$$
$$g_2(X) = -1 + 0{,}0025(x_5 + x_7 - x_4) \leq 0$$
$$g_3(X) = -1 + 0{,}01(x_8 - x_5) \leq 0$$
$$g_4(X) = -x_1x_6 + 833{,}33232x_4 + 100x_1 - 83333{,}333 \leq 0$$
$$g_5(X) = -x_2x_7 + 1250x_5 + x_2x_4 - 1250x_4 \leq 0$$
$$g_6(X) = -x_3x_8 + 1250000 + x_3x_5 - 2500x_5 \leq 0$$

Где су променљиве $100 \leq x_1 \leq 10000;$
$1000 \leq x_i \leq 10000, (i = 2{,}3);$
$10 \leq x_i \leq 1000, (i = 4, \ldots, 8)$

Функција са ограничењима 5
(CB5)

Минимизација функције циља:

$$f(X) = (x_1 - 10)^3 + (x_2 - 20)^3$$

Следећи ограничења:

$$g_1(X) = -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \leq 0$$
$$g_2(X) = (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \leq 0$$

Где су променљиве $13 \leq x_1 \leq 100;\ 0 \leq x_2 \leq 100$

Функција са ограничењима 6
(CB6)

Минимизација функције циља:

$$f(X) = -\frac{\sin^3(2\pi x_1)\sin(2\pi x_2)}{x_1^3(x_1 + x_2)}$$

Следећи ограничења:

$$g_1(X) = x_1^2 - x_2 + 1 \leq 0$$
$$g_2(X) = 1 - x_1 + (x_2 - 4)^2 \leq 0$$

Где су променљиве $0 \leq x_1 \leq 10;\ 0 \leq x_2 \leq 10$

Функција са ограничењима 7
(CB7)

Минимизација функције циља:

$$f(X) = (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2 + 10x_5^6 + 7x_6^2 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7$$

Следећи ограничења:

$$g_1(X) = -127 + 2x_1^2 + 3x_2^4 + x_3 + 4x_4^2 + 5x_5 \leq 0$$
$$g_2(X) = -282 + 7x_1 + 3x_2 + 10x_3^2 + x_4 - x_5 \leq 0$$
$$g_3(X) = -196 + 23x_1 + x_2^2 + 6x_6^2 - 8x_7 \leq 0$$
$$g_4(X) = 4x_1^2 + x_2^2 - 3x_1x_2 + 2x_3^2 + 5x_6 - 11x_7 \leq 0$$

Где су променљиве $-10 \leq x_i \leq 10 \ \ (i = 1,2, \dots ,7)$

Функција са ограничењима 8
(CB8)

Минимизација функције циља:

$$f(X) = x_1^2 + (x_2 - 1)^2$$

Следећи ограничења:

$$h(X) = x_2 - x_1^2 = 0$$

Где су променљиве $-1 \leq x_1 \leq 1; \ -1 \leq x_2 \leq 1$

Функција са ограничењима 9
(CB9)

Минимизација функције циља:

$$f(X) = -x_1 - x_2$$

Следећи ограничења:

$$g_1(X) = -2x_1^4 + 8x_1^3 - 8x_1^2 + x_2 - 2 \leq 0$$
$$g_2(X) = -4x_1^4 + 32x_1^3 - 88x_1^2 + 96x_1 + x_2 - 36 \leq 0$$

Где су променљиве $0 \leq x_1 \leq 3; \ 0 \leq x_2 \leq 4$

Додатак В. Математичка формулација инжењерских проблема оптимизације

Заварена греда

Минимизација функције циља:

$$f(x) = 1{,}10471x_1^2x_2 + 0{,}04811x_3x_4(14 + x_2)$$

Следећи ограничења:

$$g_1(x) = \tau(x) - \tau_{max} \le 0; \quad g_2(x) = \sigma(x) - \sigma_{max} \le 0; \quad g_3(x) = x_1 - x_4 \le 0;$$
$$g_4(x) = 0{,}10471x_1^2 + 0{,}04811x_3x_4(14 + x_2) - 5 \le 0;$$
$$g_5(x) = 0{,}125 - x_1 \le 0; \quad g_6(x) = \delta(x) - \delta_{max} \le 0; \quad g_7(x) = P - P_c(x) \le 0;$$

Где су:

$$x_1 = h; \quad x_2 = l; \quad x_3 = t; \quad x_4 = b;$$
$$\tau(x) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}, \quad \tau' = \frac{P}{\sqrt{2}x_1x_2}, \quad \tau'' = \frac{MR}{J},$$
$$M = P\left(L + \frac{x_2}{2}\right); \quad R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}; \quad J = 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\};$$
$$\sigma(x) = \frac{6PL}{x_4x_3^2}; \quad \delta(x) = \frac{4PL^3}{Ex_3^3x_4}; \quad P_c(x) = \frac{4{,}013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right);$$
$$P = 6000\ lb; \quad L = 14\ in; \quad E = 30 \times 10^6\ psi; \quad G = 12 \times 10^6\ psi;$$
$$\tau_{max} = 13600\ psi; \quad \sigma_{max} = 30000\ psi; \quad E = 30 \times 10^6\ psi; \quad \delta_{max} = 0{,}25\ in;$$
$$0{,}1 \le x_1 \le 2; \quad 0{,}1 \le x_2 \le 10; \quad 0{,}1 \le x_3 \le 10; \quad 0{,}1 \le x_4 \le 2.$$

Опруга

Минимизација функције циља:

$$f(x) = (x_3 + 2)x_2x_1^2$$

Следећи ограничења:

$$g_1(x) = 1 - \frac{x_2^3x_3}{71785x_1^4} \leq 0; \quad g_2(x) = \frac{4x_2^2 - x_1x_2}{12566(x_2x_1^3 - x_1^4)} + \frac{1}{5108x_1^2} - 1 \leq 0;$$
$$g_3(x) = 1 - \frac{140{,}45x_1}{x_2^2x_3} \leq 0; \quad g_4(x) = \frac{x_1 + x_2}{1{,}5} \leq 0; \quad x_1 = d; \quad x_2 = D; \quad x_3 = P;$$
$$0{,}05 \leq x_1 \leq 2; \quad 0{,}25 \leq x_2 \leq 1{,}3; \quad 2 \leq x_3 \leq 15;$$
$$x_1 = d; \quad x_2 = D; \quad x_3 - \text{дужина опруге.}$$

Редуктор

Минимизација функције циља:

$$f(x) = 0{,}7854x_1x_2^2(3{,}3333x_3^2 + 14{,}9334x_3 - 43{,}0934) - 1{,}508x_1(x_6^2 + x_7^2)$$
$$+ 7{,}4777(x_6^3 + x_7^3) + 0{,}7854(x_4x_6^2 + x_5x_7^2)$$

Следећи ограничења:

$$g_1(x) = \frac{27}{x_1x_2^2x_3} - 1 \leq 0; \quad g_2(x) = \frac{397{,}5}{x_1x_2^2x_3^2} - 1 \leq 0; \quad g_3(x) = \frac{1{,}93x_4^3}{x_2x_3x_6^4} - 1 \leq 0;$$
$$g_4(x) = \frac{1{,}93x_5^3}{x_2x_3x_7^4} - 1 \leq 0; \quad g_5(x) = \frac{\sqrt{\left(\frac{745x_4}{x_2x_3}\right)^2 + 16{,}9e6}}{110x_6^3} - 1 \leq 0;$$
$$g_6(x) = \frac{\sqrt{\left(\frac{745x_5}{x_2x_3}\right)^2 + 157{,}5e6}}{85x_7^3} - 1 \leq 0; \quad g_7(x) = \frac{x_2x_3}{40} - 1 \leq 0; \quad g_8(x) = \frac{5x_2}{x_1} - 1 \leq 0;$$
$$g_9(x) = \frac{x_1}{12x_2} - 1 \leq 0; \quad g_{10}(x) = \frac{1{,}5x_2 + 1{,}9}{x_4} - 1 \leq 0; \quad g_{11}(x) = \frac{1{,}1x_7 + 1{,}9}{x_5} - 1 \leq 0;$$
$$2{,}6 \leq x_1 \leq 3{,}6; \quad 0{,}7 \leq x_2 \leq 0{,}8; \quad 17 \leq x_3 \leq 28; \quad 7{,}3 \leq x_4 \leq 8{,}3;$$
$$7{,}8 \leq x_5 \leq 8{,}3; \quad 2{,}9 \leq x_6 \leq 3{,}9; \quad 5 \leq x_7 \leq 5{,}5;$$
$$x_1 - \text{ширина зупчаника}; \quad x_2 - \text{модул зупчаника};$$
$$x_3 - \text{број зубаца погонског зупчаника};$$
$$x_4 - \text{ширина кућишта код улазног вратила};$$
$$x_5 - \text{ширина кућишта код излазног вратила};$$
$$x_6 - \text{пречник улазног вратила};$$
$$x_7 - \text{пречник излазног вратила}.$$

Суд под притиском

Минимизација функције циља:

$$f(x) = 0{,}6224x_1x_3x_4 + 1{,}7781x_2x_3^2 + 3{,}1661x_1^2x_4 + 19{,}84x_1^2x_3$$

Следећи ограничења:

$$g_1(x) = -x_1 + 0{,}0193x_3 \leq 0; \quad g_2(x) = -x_2 + 0{,}00954x_3 \leq 0;$$
$$g_3(x) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \leq 0; \quad g_4(x) = x_4 - 240 \leq 0;$$
$$0 \leq x_i \leq 100; \quad i = 1{,}2;$$
$$10 \leq x_i \leq 200; \quad i = 3{,}4;$$
$$x_1 = T_h; \quad x_2 = T_s; \quad x_3 = R; \quad x_4 = L.$$

Носач са три штапа

Минимизација функције циља:

$$f(x) = \left(2\sqrt{2}x_1 + x_2\right) \times l$$

Следећи ограничења:

$$g_1(x) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \leq 0; \quad g_2(x) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \leq 0;$$
$$g_3(x) = \frac{1}{\sqrt{2}x_2 + x_1}P - \sigma \leq 0;$$
$$0 \leq x_i \leq 1; \quad i = 1{,}2;$$
$$l = 100\ cm; \quad P = 2\ kN/cm^2; \quad \sigma = 2\ kN/cm^2.$$

Литература

[1] J.S. Arora, Optimization of Structural and Mechanical Systems, World Scientific Publishing, University of Iowa, USA, 2007.

[2] N. Marjanović, Optimizacija zupčastih prenosnika snage, monografija, Mašinski fakultet u Kragujevcu, CAD Laboratorija, Kragujevac, 2007.

[3] S.S. Rao, Engineering Optimization, Theory and Practice, Fourth Edition ed., John Wiley & Sons, Inc., 2009.

[4] H. W. Kuhn, A.W. Tucker, Nonlinear Programming, in: Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, University of California Press, Berkeley, California, 1951.

[5] J. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, 1975.

[6] D.E. Goldberg, Genetic Algorithms in Search, Addisom-Wesley, 1989.

[7] J. Kennedy, R. Eberhart, Particle swarm optimization, in: 1995. Proceedings., IEEE International Conference on Neural Networks, 1995, pp. 1942-1948 vol.1944.

[8] D. Karaboga, B. Basturk, On the performance of artificial bee colony (ABC) algorithm, Applied Soft Computing, 8 (2008) 687-697.

[9] A.B. Levy, The Basics of Practical Optimization, SIAM - Society for Industrial and Applied Mathematics, 2009.

[10] J.J. Petrić, Operaciona istraživanja 1, Fakultet organizacionih nauka Beograd, Beograd, 1972.

[11] J.J. Petrić, Operaciona istraživanja 2, Fakultet organizacionih nauka Beograd, Beograd, 1973.

[12] J. Petrić, Nelinearno programiranje, Univerzitet u Beogradu, Beograd, 1979.

[13] J. Petrić, S. Zlobec, Nelinearno programiranje, Naučna knjiga, Beograd, 1983.

[14] R.E. Bellman, Dynamic Programming, Pricenton University Press, Pricenton, New Yersey, 1957.

[15] F. Glover, Future Paths for Integer Programming and Links to Artificial Intelligence, Computers and Operations Research, 13 (1986).

[16] S. Kirkpatrick, C. D. Gelatt, M.P. Vecchi, Optimization by Simulated Annealing, Science, 220 (1983) 671-680.

[17] I. Rechenberg, Cybernetic Solution Path of an Experimental Problem, Royal Aircraft Establishment Library Translation, 1965.

[18] K.P. R. Storm, Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces, Journal of Global Optimization, 11 (1997).

[19] J. D. Farmer, N. Packard, A. Perelson, The immune system, adaptation and machine learning, Physica, 22 (1986).

[20] M. Dorigo, Optimization, Learning and Natural Algorithms, in, Polit.di Milano, 1992.

[21] D. Karaboga, An Idea Based on Honey Bee Swarm for Numerical Optimization, in, Erciyes University, Engineering Faculty Computer Engineering Department Kayseri, Türkiye, 2005.

[22] Z. W. Geem, J. K. Kim, G.V. Loganathan, A new heuristic optimisation: Harmony search, Simulation, 76 (2001).

[23] K.A.D. Jong, Genetic Algorithms: A 10 Year Perspective, in: Proceedings of the 1st Internal Conf. on GAs and Their Appl, 1985, pp. 169-177.

[24] A. Sadollah, A. Bahreininejad, H. Eskandar, M. Hamdi, Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems, Applied Soft Computing, 13 (2013) 2592-2612.

[25] A.H. Gandomi, X.-S. Yang, A.H. Alavi, S. Talatahari, Bat algorithm for constrained optimization tasks, Neural Computing and Applications, 22 (2012) 1239-1255.

[26] X. S. Yang, A.H. Gandomi, Bat Algorithm: a novel approach for global engineering optimization, Engineering Computation, 29 (2012).

[27] H. Eskandar, A. Sadollah, A. Bahreininejad, M. Hamdi, Water cycle algorithm – A novel metaheuristic optimization method for solving constrained engineering optimization problems, Computers & Structures, 110-111 (2012) 151-166.

[28] A. Sadollah, H. Eskandar, A. Bahreininejad, J.H. Kim, Water cycle algorithm with evaporation rate for solving constrained and unconstrained optimization problems, Applied Soft Computing, 30 (2015) 58-71.

[29] M. Yazdani, F. Jolai, Lion Optimization Algorithm (LOA): A nature-inspired metaheuristic algorithm, Journal of Computational Design and Engineering, 3 (2016) 24-36.

[30] V.K. Patel, V.J. Savsani, Heat transfer search (HTS): a novel optimization algorithm, Information Sciences, 324 (2015) 217-246.

[31] P. Savsani, V. Savsani, Passing vehicle search (PVS): A novel metaheuristic algorithm, Applied Mathematical Modelling, 40 (2016) 3951-3978.

[32] R. Venkata Rao, Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems, International Journal of Industrial Engineering Computations, (2016) 19-34.

[33] R.V. Rao, V.J. Savsani, D.P. Vakharia, Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems, Computer-Aided Design, 43 (2011) 303-315.

[34] G. Appa, L. Pisoulis, P.H. Williams, Handbook on Modeling for Discrete Optimization, Springer, 2006.

[35] B. Isailović, Strukturna optimizacija elemenata mašinskih konstrukcija u cad okruženju, Magistarski rad, in, Univerzitet u Kragujevcu, Mašinski fakultet u Kragujevcu, Kragujevac, 2010.

[36] A. F. Cristodoulos, M.P. Panos, Encyclopedia of Optimization, Second Edition ed., Springer, 2009.

[37] S.J. Arora, Introduction to Optimum Design, Second Edition ed., Elsevier, 2004.

[38] A. Antoniou, W.S. Lu, Practical Optimization, Algorithms and Engineering Applications, Springer, 2007.

[39] J. Nocedal, J.S. Wright, Numerical Optimization, Springer, 1999.

[40] N. Kostić, Razvoj i primena softvera za optimizaciju tehničkih sistema primenom metode genetskog algoritma, Master rad, in: Mašinske konstrukcije i mehanizacija, Univerzitet u Kragujevcu, Mašinski fakultet u Kragujevcu, Kragujevac, 2011.

[41] V. Ranković, Inteligentno upravljanje, Mašinski fakultet u Kragujevcu, Kragujevac, 2008.

[42] D.J. Reid, Genetic Algorithms in Constrained Optimization Mathl. Comput. Modelling, 23 (1996).

[43] K. Deb, An efficient constraint handling method for genetic algorithms, Comput. Methods Appl. Mech. Engrg., 186 (2000).

[44] K. Deb, S. Gulati, Design of truss-structures for minimum weight using genetic algorithms, Finite Elements in Analysis and Design, 37 (2001).

[45] V. Toğan, A.T. Daloğlu, An improved genetic algorithm with initial population strategy and self-adaptive member grouping, Computers & Structures, 86 (2008) 1204-1218.

[46] H. Rahami, A. Kaveh, Y. Gholipour, Sizing, geometry and topology optimization of trusses via force method and genetic algorithm, Engineering Structures, 30 (2008) 2360-2369.

[47] T. Dede, S. Bekiroğlu, Y. Ayvaz, Weight minimization of trusses with genetic algorithm, Applied Soft Computing, 11 (2011) 2565-2575.

[48] Patrick Siarry, Alain Petrowski, M. Bessaou, A multipopulation genetic algorithm aimed at multimodal optimiyation, Advances in Engineering Software, 33 (2002).

[49] K. Deep, Dipti, A self-organizing migrating genetic algorithm for constrained optimization, Applied Mathematics and Computation, 198 (2008) 237-250.

[50] T. Murata, H. Ishibuchi, H. Tanaka, Multi-Objective Genetic Algorithm and Its Applications to Flowshop Scheduling Computers ind. Engng, 30 (1996).

[51] V.S. Summanwar, V.K. Jayaraman, B.D. Kulkarni, H.S. Kusumakar, K. Gupta, J. Rajesh, Solution of constrained optimization problems by multi-objective genetic algorithm, Computers & Chemical Engineering, 26 (2002).

[52] A. Konak, D.W. Coit, A.E. Smith, Multi-objective optimization using genetic algorithms: A tutorial, Reliability Engineering & System Safety, 91 (2006) 992-1007.

[53] X. Li, G. Du, BSTBGA: A hybrid genetic algorithm for constrained multi-objective optimization problems, Computers & Operations Research, 40 (2013) 282-302.

[54] Q. Long, A constraint handling technique for constrained multi-objective genetic algorithm, Swarm and Evolutionary Computation, 15 (2014) 66-79.

[55] G. Renner, A. Ekárt, Genetic algorithms in computer aided design, Computer-Aided Design, 35 (2003) 709-726.

[56] I.G. Tsoulos, Modifications of real code genetic algorithm for global optimization, Applied Mathematics and Computation, 203 (2008) 598-607.

[57] M.S. Gibbs, G.C. Dandy, H.R. Maier, A genetic algorithm calibration method based on convergence due to genetic drift, Information Sciences, 178 (2008) 2857-2869.

[58] I.G. Tsoulos, Solving constrained optimization problems using a novel genetic algorithm, Applied Mathematics and Computation, 208 (2009) 273-283.

[59] A. Reese, Random number generators in genetic algorithms for unconstrained and constrained optimization, Nonlinear Analysis: Theory, Methods & Applications, 71 (2009) e679-e692.

[60] M. Kaya, The effects of two new crossover operators on genetic algorithm performance, Applied Soft Computing, 11 (2011) 881-890.

[61] F. Musharavati, A.S.M. Hamouda, Modified genetic algorithms for manufacturing process planning in multiple parts manufacturing lines, Expert Systems with Applications, 38 (2011) 10770-10779.

[62] H.-C. Kuo, C.-H. Lin, A Directed Genetic Algorithm for global optimization, Applied Mathematics and Computation, 219 (2013) 7348-7364.

[63] C.-H. Lin, A rough penalty genetic algorithm for constrained optimization, Information Sciences, 241 (2013) 119-137.

[64] S.M. Elsayed, R.A. Sarker, D.L. Essam, A new genetic algorithm for solving optimization problems, Engineering Applications of Artificial Intelligence, 27 (2014) 57-69.

[65] A. Misevicius, Genetic algorithm hybridized with ruin and recreate procedure: application to the quadratic assignment problem, Knowledge-Based Systems, 16 (2003) 261-268.

[66] A. Misevicius, An improved hybrid genetic algorithm: new results for the quadratic assignment problem, Knowledge-Based Systems, 17 (2004) 65-73.

[67] L. Wei, M. Zhao, A niche hybrid genetic algorithm for global optimization of continuous multimodal functions, Applied Mathematics and Computation, 160 (2005) 649-661.

[68] Q. Yuan, Z. He, H. Leng, A hybrid genetic algorithm for a class of global optimization problems with box constraints, Applied Mathematics and Computation, 197 (2008) 924-929.

[69] M.M. Kabir, M. Shahjahan, K. Murase, A new local search based hybrid genetic algorithm for feature selection, Neurocomputing, 74 (2011) 2914-2928.

[70] K.N. Das, R. Mishra, Chemo-inspired genetic algorithm for function optimization, Applied Mathematics and Computation, 220 (2013) 394-404.

[71] E.K. Nyarko, R. Scitovski, Solving the parameter identification problem of mathematical models using genetic algorithms, Applied Mathematics and Computation, 153 (2004) 651-658.

[72] D. Bunnag, M. Sun, Genetic algorithm for constrained global optimization in continuous variables, Applied Mathematics and Computation, 171 (2005) 604-636.

[73] N. Tutkun, Parameter estimation in mathematical models using the real coded genetic algorithms, Expert Systems with Applications, 36 (2009) 3342-3345.

[74] E.G. Shopova, N.G. Vaklieva-Bancheva, BASIC—A genetic algorithm for engineering problems solution, Computers & Chemical Engineering, 30 (2006) 1293-1309.

[75] I.G. Tsoulos, I.E. Lagaris, GenMin: An enhanced genetic algorithm for global optimization, Computer Physics Communications, 178 (2008) 843-851.

[76] P. Kaelo, M.M. Ali, Integrated crossover rules in real coded genetic algorithms, European Journal of Operational Research, 176 (2007) 60-76.

[77] K. Deep, M. Thakur, A new crossover operator for real coded genetic algorithms, Applied Mathematics and Computation, 188 (2007) 895-911.

[78] C. García-Martínez, M. Lozano, F. Herrera, D. Molina, A.M. Sánchez, Global and local real-coded genetic algorithms based on parent-centric crossover operators, European Journal of Operational Research, 185 (2008) 1088-1113.

[79] N. Tutkun, Optimization of multimodal continuous functions using a new crossover for the real-coded genetic algorithms, Expert Systems with Applications, 36 (2009) 8172-8177.

[80] A. Banerjee, A novel probabilistically-guided context-sensitive crossover operator for clustering, Swarm and Evolutionary Computation, 13 (2013) 47-62.

[81] S.-H. Chen, M.-C. Chen, P.-C. Chang, V. Mani, Multiple parents crossover operators: A new approach removes the overlapping solutions for sequencing problems, Applied Mathematical Modelling, 37 (2013) 2737-2746.

[82] Y.-C. Chuang, C.-T. Chen, C. Hwang, A real-coded genetic algorithm with a direction-based crossover operator, Information Sciences, 305 (2015) 320-348.

[83] K. Deep, M. Thakur, A new mutation operator for real coded genetic algorithms, Applied Mathematics and Computation, 193 (2007) 211-230.

[84] M.A. Khalik, M. Sherif, S. Saraya, F. Areed, Parameter identification problem: Real-coded GA approach, Applied Mathematics and Computation, 187 (2007) 1495-1501.

[85] T. Kellegöz, B. Toklu, J. Wilson, Comparing efficiencies of genetic crossover operators for one machine total weighted tardiness problem, Applied Mathematics and Computation, 199 (2008) 590-598.

[86] M. Thakur, A new genetic algorithm for global optimization of multimodal continuous functions, Journal of Computational Science, 5 (2014) 298-311.

[87] K. Deep, K.P. Singh, M.L. Kansal, C. Mohan, A real coded genetic algorithm for solving integer and mixed integer optimization problems, Applied Mathematics and Computation, 212 (2009) 505-518.

[88] E. Bonabeau, M. Dorigo, G. Theraulaz, Swarm intelligence: From natural to artificial systems, Oxford University Press, Oxford, 1999.

[89] J. Kennedy, R. Eberhart, Swarm Intelligence, Morgan Kaufmann, San Mateo, CA, 2001.

[90] R. C. Eberhart, Y. Shi, Compating inertia weights and constriction factors in particle swarm optimization, in: Proceedings of the IEEE Congress on Evolutionary Computation, San Diego, USA, 2000, pp. 84-88.

[91] I.C. Trelea, The particle swarm optimization algorithm: convergence analysis and parameter selection, Information Processing Letters, 85 (2003) 8.

[92] M.M. Ali, P. Kaelo, Improved particle swarm algorithms for global optimization, Applied Mathematics and Computation, 196 (2008) 578-593.

[93] R.E. Perez, K. Behdinan, Particle swarm approach for structural design optimization, Computers & Structures, 85 (2007) 1579-1588.

[94] L.J. Li, Z.B. Huang, F. Liu, A heuristic particle swarm optimization method for truss structures with discrete variables, Computers & Structures, 87 (2009) 435-443.

[95] A. Kaveh, S. Talatahari, A particle swarm ant colony optimization for truss structures with discrete variables, Journal of Constructional Steel Research, 65 (2009) 1558-1568.

[96] Y. Wang, Y. Yang, Particle swarm optimization with preference order ranking for multi-objective optimization, Information Sciences, 179 (2009) 1944-1959.

[97] C.K. Goh, K.C. Tan, D.S. Liu, S.C. Chiam, A competitive and cooperative co-evolutionary approach to multi-objective particle swarm optimization algorithm design, European Journal of Operational Research, 202 (2010) 42-54.

[98] S.-J. Tsai, T.-Y. Sun, C.-C. Liu, S.-T. Hsieh, W.-C. Wu, S.-Y. Chiu, An improved multi-objective particle swarm optimizer for multi-objective problems, Expert Systems with Applications, 37 (2010) 5872-5886.

[99] M. Rabbani, M. Aramoon Bajestani, G. Baharian Khoshkhou, A multi-objective particle swarm optimization for project selection problem, Expert Systems with Applications, 37 (2010) 315-321.

[100] Y. Wang, Y. Yang, Particle swarm with equilibrium strategy of selection for multi-objective optimization, European Journal of Operational Research, 200 (2010) 187-197.

[101] D. Hu, A. Sarosh, Y.-F. Dong, An improved particle swarm optimizer for parametric optimization of flexible satellite controller, Applied Mathematics and Computation, 217 (2011) 8512-8521.

[102] G. He, N.-j. Huang, A modified particle swarm optimization algorithm with applications, Applied Mathematics and Computation, 219 (2012) 1053-1060.

[103] M. Gang, Z. Wei, C. Xiaolin, A novel particle swarm optimization algorithm based on particle migration, Applied Mathematics and Computation, 218 (2012) 6620-6626.

[104] M.-S. Leu, M.-F. Yeh, Grey particle swarm optimization, Applied Soft Computing, 12 (2012) 2985-2996.

[105] M. Hu, T. Wu, J.D. Weir, An intelligent augmentation of particle swarm optimization with multiple adaptive methods, Information Sciences, 213 (2012) 68-83.

[106] H. Huang, H. Qin, Z. Hao, A. Lim, Example-based learning particle swarm optimization for continuous optimization, Information Sciences, 182 (2012) 125-138.

[107] F. Neri, E. Mininno, G. Iacca, Compact Particle Swarm Optimization, Information Sciences, 239 (2013) 96-121.

[108] W. Zhang, D. Ma, J.-j. Wei, H.-f. Liang, A parameter selection strategy for particle swarm optimization based on particle positions, Expert Systems with Applications, 41 (2014) 3576-3584.

[109] Y.-B. Shin, E. Kita, Search performance improvement of Particle Swarm Optimization by second best particle information, Applied Mathematics and Computation, 246 (2014) 346-354.

[110] S. Sun, J. Li, A two-swarm cooperative particle swarms optimization, Swarm and Evolutionary Computation, 15 (2014) 1-18.

[111] S.M. Elsayed, R.A. Sarker, E. Mezura-Montes, Self-adaptive mix of particle swarm methodologies for constrained optimization, Information Sciences, 277 (2014) 216-233.

[112] H. Haklı, H. Uğuz, A novel particle swarm optimization algorithm with Levy flight, Applied Soft Computing, 23 (2014) 333-345.

[113] O.E. Turgut, M.S. Turgut, M.T. Coban, Chaotic quantum behaved particle swarm optimization algorithm for solving nonlinear system of equations, Computers & Mathematics with Applications, 68 (2014) 508-530.

[114] X. Yu, X. Zhang, Enhanced comprehensive learning particle swarm optimization, Applied Mathematics and Computation, 242 (2014) 265-276.

[115] X. Xia, J. Liu, Z. Hu, An improved particle swarm optimizer based on tabu detecting and local learning strategy in a shrunk search space, Applied Soft Computing, 23 (2014) 76-90.

[116] M.J. Mahmoodabadi, Z. Salahshoor Mottaghi, A. Bagheri, HEPSO: High exploration particle swarm optimization, Information Sciences, 273 (2014) 101-111.

[117] G. He, N.-j. Huang, A new particle swarm optimization algorithm with an application, Applied Mathematics and Computation, 232 (2014) 521-528.

[118] A. Yadav, K. Deep, An efficient co-swarm particle swarm optimization for non-linear constrained optimization, Journal of Computational Science, 5 (2014) 258-268.

[119] L. Wang, B. Yang, Y. Chen, Improving particle swarm optimization using multi-layer searching strategy, Information Sciences, 274 (2014) 70-94.

[120] I. Gosciniak, A new approach to particle swarm optimization algorithm, Expert Systems with Applications, 42 (2015) 844-854.

[121] H. Liu, Z. Cai, Y. Wang, Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization, Applied Soft Computing, 10 (2010) 629-640.

[122] M. Montemurro, A. Vincenti, P. Vannucci, The Automatic Dynamic Penalisation method (ADP) for handling constraints with genetic algorithms, Computer Methods in Applied Mechanics and Engineering, 256 (2013) 70-87.

[123] E. Zahara, Y.-T. Kao, Hybrid Nelder–Mead simplex search and particle swarm optimization for constrained engineering design problems, Expert Systems with Applications, 36 (2009) 3880-3886.

[124] C. Li, S. Yang, A clustering particle swarm optimizer for dynamic optimization, in: Proceeding of the 2009 congress on evolutionary computation, 2009, pp. 439-446.

[125] R. Venkata Rao, Review of applications of TLBO algorithm and a tutorial for beginners to solve the unconstrained and constrained optimization problems, Decision Science Letters, (2016) 1-30.

[126] M. Črepinšek, S.-H. Liu, L. Mernik, A note on teaching–learning-based optimization algorithm, Information Sciences, 212 (2012) 79-93.

[127] G. Waghmare, Comments on "A note on teaching–learning-based optimization algorithm", Information Sciences, 229 (2013) 159-169.

[128] R.V. Rao, V. Patel, An elitist teaching-learning-based optimization algorithm for solving complex constrained optimization problems, international journal of industrial engineering computations, 3 (2012) 535-560.

[129] R.V. Rao, V. Patel, An improved teaching-learning-based optimization algorithm for solving unconstrained optimization problems, Scientia Iranica, (2012).

[130] R.V. Rao, V.J. Savsani, D.P. Vakharia, Teaching–Learning-Based Optimization: An optimization method for continuous non-linear large scale problems, Information Sciences, 183 (2012) 1-15.

[131] S.C. Satapathy, A. Naik, Modified Teaching–Learning-Based Optimization algorithm for global numerical optimization—A comparative study, Swarm and Evolutionary Computation, 16 (2014) 28-37.

[132] A. Baykasoğlu, A. Hamzadayi, S.Y. Köse, Testing the performance of teaching–learning based optimization (TLBO) algorithm on combinatorial problems: Flow shop and job shop scheduling cases, Information Sciences, 276 (2014) 204-218.

[133] L. Wang, F. Zou, X. Hei, D. Yang, D. Chen, Q. Jiang, An improved teaching–learning-based optimization with neighborhood search for applications of ANN, Neurocomputing, 143 (2014) 231-247.

[134] F. Zou, L. Wang, X. Hei, D. Chen, D. Yang, Teaching–learning-based optimization with dynamic group strategy for global optimization, Information Sciences, 273 (2014) 112-131.

[135] D. Chen, F. Zou, Z. Li, J. Wang, S. Li, An improved teaching–learning-based optimization algorithm for solving global optimization problem, Information Sciences, 297 (2015) 171-190.

[136] J. Huang, L. Gao, X. Li, An effective teaching-learning-based cuckoo search algorithm for parameter optimization problems in structure designing and machining processes, Applied Soft Computing, 36 (2015) 349-356.

[137] H.-b. Ouyang, L.-q. Gao, X.-y. Kong, D.-x. Zou, S. Li, Teaching-learning based optimization with global crossover for global optimization problems, Applied Mathematics and Computation, 265 (2015) 533-556.

[138] F. Zou, L. Wang, X. Hei, D. Chen, Teaching–learning-based optimization with learning experience of other learners and its application, Applied Soft Computing, 37 (2015) 725-736.

[139] R. Kadambur, P. Kotecha, Multi-level production planning in a petrochemical industry using elitist Teaching–Learning-Based-Optimization, Expert Systems with Applications, 42 (2015) 628-641.

[140] J.K. Pickard, J.A. Carretero, V.C. Bhavsar, On the convergence and origin bias of the Teaching-Learning-Based-Optimization algorithm, Applied Soft Computing, 46 (2016) 115-127.

[141] R.V. Rao, D.P. Rai, Optimization of fused deposition modeling process using teaching-learning-based optimization algorithm, Engineering Science and Technology, an International Journal, 19 (2016) 587-603.

[142] K. Yu, X. Wang, Z. Wang, Constrained optimization based on improved teaching–learning-based optimization algorithm, Information Sciences, 352-353 (2016) 61-78.

[143] R.V. Rao, V. Patel, Multi-objective optimization of heat exchangers using a modified teaching-learning-based optimization algorithm, Applied Mathematical Modelling, 37 (2013) 1147-1162.

[144] F. Zou, L. Wang, X. Hei, D. Chen, B. Wang, Multi-objective optimization using teaching-learning-based optimization algorithm, Engineering Applications of Artificial Intelligence, 26 (2013) 1291-1300.

[145] J.M. Chaves-González, M.A. Pérez-Toledano, A. Navasa, Teaching learning based optimization with Pareto tournament for the multiobjective software requirements selection, Engineering Applications of Artificial Intelligence, 43 (2015) 89-101.

[146] V.K. Patel, V.J. Savsani, A multi-objective improved teaching–learning based optimization algorithm (MO-ITLBO), Information Sciences, 357 (2016) 182-200.

[147] K. Xia, L. Gao, W. Li, K.-M. Chao, Disassembly sequence planning using a Simplified Teaching–Learning-Based Optimization algorithm, Advanced Engineering Informatics, 28 (2014) 518-527.

[148] Z. Xie, C. Zhang, X. Shao, W. Lin, H. Zhu, An effective hybrid teaching–learning-based optimization algorithm for permutation flow shop scheduling problem, Advances in Engineering Software, 77 (2014) 35-47.

[149] T. Dokeroglu, Hybrid teaching–learning-based optimization algorithms for the Quadratic Assignment Problem, Computers & Industrial Engineering, 85 (2015) 86-101.

[150] G. Polya, How to Solve It, Princeton Univ. Pr, 1945.

[151] J.A. Koupaei, S.M.M. Hosseini, F.M.M. Ghaini, A new optimization algorithm based on chaotic maps and golden section search method, Engineering Applications of Artificial Intelligence, 50 (2016) 201-214.

[152] D. Karaboga, B. Akay, A modified Artificial Bee Colony (ABC) algorithm for constrained optimization problems, Applied Soft Computing, 11 (2011) 3021-3031.

[153] R.V. Rao, G.G. Waghmare, A new optimization algorithm for solving complex constrained design optimization problems, Engineering Optimization, 49 (2017) 60-83.

[154] P. Savsani, R.L. Jhala, V. Savsani, Effect of hybridizing Biogeography-Based Optimization (BBO) technique with Artificial Immune Algorithm (AIA) and Ant Colony Optimization (ACO), Applied Soft Computing, 21 (2014) 542-553.

[155] C.A.C. Coello, Use of a self-adaptive penalty approach for engineering optimiyation problems, Computers in Industry, 41 (2000) 113-127.

[156] B. Akay, D. Karaboga, Artificial bee colony algorithm for large-scale problems and engineering design optimization, Journal of Intelligent Manufacturing, 23 (2010) 1001-1014.

[157] A. Baykasoğlu, F.B. Ozsoydan, Adaptive firefly algorithm with chaos for mechanical design optimization problems, Applied Soft Computing, 36 (2015) 152-164.

[158] J.S. Arora, Introduction to Optimum Design, New York, McGraw-Hill, 1989.

[159] Y. Wang, Z. Cai, Y. Zhou, Z. Fan, Constrained optimization based on hybrid evolutionary algorithm and adaptive constraint-handling technique, Structural and Multidisciplinary Optimization, 37 (2008) 395-413.

[160] A.H. Gandomi, Interior search algorithm (ISA): A novel approach for global optimization, ISA Transactions, 53 (2014) 1168-1183.

[161] M. Zhang, W. Luo, X. Wang, Differential evolution with dynamic stochastic selection for constrained optimization, Information Sciences, 178 (2008) 3043-3074.

[162] L. P. Pomrehn, P.Y. Papalambros, Discrete Optimal Design Formulations With Application to Gear Train Design, ASME Journal of Mechanical Design, 117 (1995).

[163] Tae Hyong Chong, Inho Bae, G.-J. Park, A new and generalized methodology to design multi-stage gear drives by integrating the dimensional and the configuration design process, Mechanism and Machine Theory, 37 (2002).

[164] N. Marjanovic, B. Isailovic, V. Marjanovic, Z. Milojevic, M. Blagojevic, M. Bojic, A practical approach to the optimization of gear trains with spur gears, Mechanism and Machine Theory, 53 (2012) 1-16.

[165] S.i. Golabi, J.J. Fesharaki, M. Yazdipoor, Gear train optimization based on minimum volume/weight design, Mechanism and Machine Theory, 73 (2014) 197-217.

[166] R. Mendi, T. Başkal, K. Boran, F.E. Boran, Optimization of module, shaft diameter and rolling bearing for spur gear through genetic algorithm, Expert Systems with Applications, 37 (2010).

[167] F. Mendi, T. Başkal, K. Boran, F.E. Boran, Optimization of module, shaft diameter and rolling bearing for spur gear through genetic algorithm, Expert Systems with Applications, 37 (2010) 8058-8064.

[168] V. Savsani, R.V. Rao, D.P. Vakharia, Optimal weight design of a gear train using particle swarm optimization and simulated annealing algorithms, Mechanism and Machine Theory, 45 (2010) 531-541.

[169] C. Gologlu, M. Zeyveli, A genetic approach to automate preliminary design of gear drives, Computers & Industrial Engineering, 57 (2009) 1043-1051.

[170] X. Li, G. R. Symmons, G. Cockerham, Optimal design of involute profile helical gears, Mechanism and Machine Theory, 31 (1996).

[171] M. M. A. Kader, S. P. Nigam, G.K. Grover, A study on mode of failures in spur gears under optimized conditions, Mechanism and Machine Theory, 33 (1998).

[172] V.N. Kudrijevcev, Planetary Gear Train (in Russian), Mechanical Engineering, Leningrad, 1966.

[173] M. Lehmann, Calculation and Measurement of Forces Acting on Cycloid Speed Reducer (in German), in, Technical University Munich, Munich, Germany, 1976.

[174] S. K. Malhotra, M.A. Parameswaran, Analysis of a Cycloid Speed Reducer, Mechanism and Machine Theory, 18 (1983).

[175] M. Yunhong, W. Changlin, L. Liping, Mathematical Modeling of the Transmission Performance of 2K-H Pin Cycloid Planetary Mechanism, Mechanism and Machine Theory, 42 (2007).

[176] W. S. Lin, Y. P. Shin, J.J. Lee, Design of a Two-Stage Cycloidal Gear Reducer with Tooth Modifications, Mechanism and Machine Theory, 79 (2014).

[177] Y. W. Hwang, C.F. Hsieh, Geometric Design Using Hypotrochoid and Nonundercutting Conditions for an Internal Cycloidal Gear, Journal of Mechanical Design, 129 (2007).

[178] M. Blagojević, Naponsko i deformaciono stanje elemenata cikloreduktora pri dinamičkim opterećenjima, Doktorska disertacija, in, Univerzitet u Kragujevcu, Mašinski fakultet u Kragujevcu, Kragujevac, 2008.

[179] F. Litvin, P. Feng, Computerized Design and Generation of Cycloidal Gearings, Mechanism and Machine Theory, 31 (1996).

[180] H. S. Yan, T.S. Lai, Geometry Design of an Elementary Planetary Gear Train with Cylindrical Tooth-Profiles, Mechanism and Machine Theory, 37 (2002).

[181] Y. W. Hwang, C.F. Hsieh, Geometry Design and Analysis for Trochoidal – Type Speed Reducers: with Conjugate Envelopes, Transactions of the CSME/de la SCGM, 30 (2006).

[182] B. K. Chen, T. T. Fang, C. Y. Li, S.Y. Wang, Gear Geometry of Cycloid Drives, Science in China Series E: Technological Sciences, 51 (2008).

[183] J. H. Shin, S.M. Kwon, On the Lobe Profile Design in a Cycloid Reducer Using Instant Velocity Center, Mechanism and Machine Theory, 41 (2006).

[184] A. Dascalescu, M. Ungureanu, CAD – CAM Programs Applied to the Cycloid Profile Wheels Processing, Annals of the University of Petrosani, Mechanical Engineering, 12 (2010).

[185] S. Li, Design and Strenght Analysis Methods of the Trochoidal Gear Reducers, Mechanism and Machine Theory, 81 (2014).

[186] B. Chen, H. Zhong, J. Liu, C. Li, T. Fang, Generation and Investigation of a New Cycloid drive with Double Contact, Mechanism and Machine Theory, 49 (2012).

[187] M. Blagojevic, N. Marjanovic, Z. Djordjevic, Z. Stojanovic, A. Disic, A New Design of a Two-Stage Cycloidal Speed Reducer, Journal of Mechanical Design, 133 (2011).

[188] C.F. Hsieh, The Effect on Dynamics of Using a New Transmission Design for Eccentric Speed Reducer, Mechanism and Machine Theory, 80 (2014).

[189] C.F. Hsieh, Traditional Versus Improved Designs for Cycloidal Speed Reducers with a Small Tooth Difference: The Effect on Dynamics, Mechanism and Machine Theory, 86 (2015).

[190] J. Sensinger, Unified Approach to Cycloid Drive Profile, Stress, and Efficiency Optimization, Journal of Mechanical Design, 132 (2010).

[191] X. Y. Qian, L. Zou, B. Guo, Multi – Objective Moderate Optimization in Cycloid Drive, International Conference on Digital Manufacturing & Automation, (2010).

[192] I. Gu, Design of Antibacklash Pin-Gearing, Journal of Mechanical Design, 120 (1998).

[193] J. G. Blanche, D.C.H. Yang, Cycloid Drives with Machining Tolerances, Journal of Mechanisms, Transmissions, and Automation in Design, 111 (1989).

[194] D. C. H. Yang, J.G. Blanche, Design and Application Guidelines for Cycloid Drives with Machining Tolerances, Mechanism and Machine Theory, 25 (1990).

[195] L. Lixing, L. Xin, H. Weidong, Q. Yuanmei, Profile Modification and Accurate Force Analysis on Cycloid Drive, in: World Congress on Gearing and Power Transmission, Paris, France, 1999.

[196] W. P. Kao, C. C. Hsieh, J.J. Lee, Computer-Aided Kinematic Error Analysis of a Two-Stage Cycloidal Drive, in: The 14th IFToMM World Congress, Taipei, Taiwan, 2015.