УНИВЕРЗИТЕТ У КРАГУЈЕВЦУ

ФАКУЛТЕТ ЗА МАШИНСТВО И ГРАЂЕВИНАРСТВО У КРАЉЕВУ

Горан Р. Миодраговић

РАЗВОЈ НАПРЕДНИХ БИОЛОШКИ ИНСПИРИСАНИХ АЛГОРИТАМА ЗА РЕШАВАЊЕ ОПТИМИЗАЦИОНИХ ПРОБЛЕМА ПРИМЕЊЕНЕ МЕХАНИКЕ

Докторска дисертација

Краљево, 2015. год.

ИДЕНТИФИКАЦИОНА СТРАНИЦА ДОКТОРСКЕ ДИСЕРТАЦИЈЕ

I. Аутор
Име и презиме: *Горан Миодраговић*
Датум и место рођења: *03.11.1964. Краљево*
Садашње запослење: *предавач на Високој техничкој машинској школи струковних студија у Трстенику*
II. Докторска дисертација
Наслов: *Развој напредних биолошки инспирисаних алгоритама за решавање оптимизационих проблема примењене механике*
Број страница:142
Број слика: 61
Број библиографских података: 165
Установа и место где је рад израђен: *Факултет за машинство и грађевинарство у Краљеву Универзитета у Крагујевцу*
Научна област (УДК): Механика:Примењена информатика и рачунарско инжењерство (531: 621.914)
Ментор: *проф. др Радован Булатовић*
III. Оцена и одбрана
Број одлуке и датум прихватања докторске дисертације: Бр. IV-04-603/8, 12.11.2013.год
Комисија за оцену подобности теме и кандидата: др Звонимир Југовић, редовни професор др Србољуб Симић, редовни професор др Миле Савковић, редовни професор др Радован Булатовић, ванредни професор др Славиша Шалинић, доцент.
Комисија за оцену и одбрану докторске дисертације: др Звонимир Југовић, редовни професор др Србољуб Симић, редовни професор др Миле Савковић, редовни професор др Славиша Шалинић, ванредни професор др Радован Булатовић, ванредни професор, ментор
Датум одбране дисертације: __.__.20__ год.

Овај рад посвећујем својој деци

Захвалност аутора

Ова докторска дисертација представља резултат вишегодишњег образовања и истраживачког рада и из тог разлога осећам потребу да искажем своју искрену захвалност и поштовање бројним професорима, колегама, пријатељима и породици који су ми на том путу пружили велику помоћ и подршку.

Своју дубоку захвалност дугујем ментору, проф. др Радовану Булатовићу за несебичну помоћ и савете током израде ове дисертације. Посебну захвалност желим да искажем и члановима комисије за помоћ и сугестије од тренутка дефинисања, затим у току и на самом крају израде докторске дисертације.

Такође, захвалност дугујем свим садашњим и бившим колегама на Факултету за машинство и грађевинарство у Краљеву на посредној или непосредној подршци у досадашњем раду.

Захваљујем се искрено својој породици на исказаном стрпљењу и разумевању у времену израде ове дисертације .

У Краљеву, _____. *мр Горан Миодраговић, дипл. инж. маш.*

РАЗВОЈ НАПРЕДНИХ БИОЛОШКИ ИНСПИРИСАНИХ АЛГОРИТАМА ЗА РЕШАВАЊЕ ОПТИМИЗАЦИОНИХ ПРОБЛЕМА ПРИМЕЊЕНЕ МЕХАНИКЕ

РЕЗИМЕ

У последњих петнаестак година појављују се методе које све боље решавају компликоване оптимизационе проблеме. Све ове методе су настале као инспирација са одговарајућим појавама у природи, па се и зову биолошки инспирисане методе. Најпознатије су: генетски алгоритми (Genetic Algorithm - GA), диференцијална еволуција (Differential Evolution DE), оптимизација појем честица (Particle Swarm Optmization PSO), оптимизација инспирисана кретањем мрава (Ant Colony Optimization - ACO), кукавичја претрага (Cuckoo Search – CS), алгоритам свица (Firefly Algorithm – FA), алгоритам слепог миша (Bat Algorithm – BA), оптимизација инспирисана кретањем планктона (Krill Herd Algorithm – KHA) итд. Сви ови алгоритми се могу применити на велики број проблема, дају могућност постављања широког опсега за почетне вредности пројектних променљивих – тако да није потребно искуство при одређивању блиских почетних вредности, функција која се оптимизира овим методама не мора бити диференцијабилна и непрекидна, нема ограничења у односу на број променљивих који се оптимизира, применљиве су на велики број проблема, затим структуре њихових алгоритама нуде велике могућности надоградње – чиме се може постићи ефикасност алгоритма једноставним модификацијама.

Методологија истраживања у овом раду је фокусирана на четири од горе поменутих метода: кукавичја претрага (Cuckoo Search – CS), алгоритам свица (Firefly Algorithm – FA), алгоритам слепог миша (Bat Algorithm – BA), оптимизација инспирисана кретањем планктона (Krill Herd Algorithm – KHA). Циљ истраживања је да се направе одговарајуће модификације и хибридизације поменутих метода, које ће постизати боље решење у пољу глобалних минимума. Тако добијени алгоритми, тестирани су на бенчмарк оптимизационим проблемима примењене механике који постоје у литератури. Такође циљ истраживања је и моделирање неких од наведених проблема више сложености и тестирање овако унапређених алгоритама на такве проблеме. Идеја је да се успостави универзални алгоритам како би се са лакоћом применио у решавању различитих оптимизационих проблема у машинству, односно примењеној механици у циљу добијања глобалног минимума.

Кључне речи: *алгоритам слепог миша; ограничена оптимизација; метахеуристика; Lévy-лет; циклични алгоритам фамилије слепих мишева; модификовани алгоритам крила, димензиона синтеза, грешка растојања, алгоритам кукавичје претраге, алгоритам свица, хибридни алгоритам кукавичје претраге и алгоритма свица.*

ADVANCED BIO-INSPIRED ALGORITHMS DEVELOPMENT FOR SOLVING OPTIMIZATION PROBLEMS IN APPLIED MECHANICS

SUMMARY

In the last fifteen years methods that better solve complex optimization problems appear. All these methods have emerged as an inspiration to the corresponding phenomena in nature, so they are called biologically inspired methods. The best known are: Genetic Algorithms (Genetic Algorithm - GA), differential evolution (DE Differential Evolution), Particle Swarm Optimization (PSO Particle Swarm optmization), optimization inspired by the movement of ants (Ant Colony Optimization - ACO), cuckoo searches (Cuckoo Search - CS) algorithm firefly (Firefly Algorithm - FA) algorithm bat (Bat Algorithm - BA), optimization inspired by the movement artick krill (Krill Herd Algorithm - KHA) etc. All of these algorithms can be applied to a large number of problems, give the possibility of setting up a wide range of initial values of the design variables - so you do not need experience in determining close initial value, a function that optimizes these methods may not be differentiable and continuous, no restrictions on the the number of variables that optimizes, are applicable to a large number of problems and structure of their algorithms offer great possibilities for upgrades - which can be achieved by simple modification of the efficiency of the algorithm.

The research methodology, in this thesis, is focused on four of the above-mentioned methods: cuckoo searches (Cuckoo Search - CS) algorithm firefly (Firefly Algorithm - FA) algorithm bat (Bat Algorithm - BA), optimization inspired by the movement of plankton (Krill Herd Algorithm - KHA). The aim of the research is to make appropriate modifications and hybridization of these methods, which will achieve a better solution in the field of global minimum. The thus-obtained algorithms were tested on a benchmark problems by optimization in applied mechanics, that exist in the literature. Also the aim of the research is modeling some more complex problems and testing this advanced algorithms on such problems. The idea is to establish a universal algorithm which will be easily applied in solving various optimization problems in mechanical engineering or applied mechanics, in order to obtain the global minimum.

Key words: *the bat algorithm; limited optimization; metaheuristics; Lévy-flight; the Loop family bat algorithm; the modified krill algorith, dimensional synthesis, distance error, the cuckoo search algorithm, firefly algorithm, the hybrid cuckoo search and firefly algorithm*

САДРЖАЈ

СПИСАК СКРАЋЕНИЦА ПО РЕДОСЛЕДУ ПОЈАВЉИВАЊА

GA	genetic algorithm – генетски алгоритам
DE	differential evolution – диференцијална еволуција
PSO	particle swarm optmization – оптимизација ројем честица
ACO	ant colony optimization – оптимизација колоније марава
CS	cuckoo search - кукавичја претрага
FA	firefly algorithm – алгоритам свица
BA	bat algorithm – алгоритам слепог миша
KHA	krill herd algorithm – алгоритам крила
Loop BFA	loop bat family algorithm - циклични алгоритам фамилије слепих мишева
H–CS–FA	hibridni cuckoo search - firefly algorithm
MOO	multiobjective optimization – вишекритеријумска оптимизација
MKH	modified krill herd – модификовани алгоритам крила
HS	harmony search – хармонијска претрага
HBA	honey bee algorithm – алгоритам медоносних пчела
SI	swarm intelligence – интелигенција јата
BF	bat family – фамилија слепих мишева
PSOLVER	нови хибридни оптимизациони алгоритам оптимизације ројем честица
EHGA	ефективни хибридни генетски алгоритам са додатном флексибилном техником
TLBO	оптимизација заснована на предавању-учењу
HGSO	хибридни алгоритам свица и оптимизације ројем честица
C-EAPSO	chaos-enhanced accelerated particle swarm optmization – хаосом побољшани убрзани алгоритам оптимизације ројем честица
MBA	mine blast algorithm - алгоритам експлозије мине
HDE&N-MA	хибридна диференцијална еволуција и Nelder-Mead алгоритам
B-IA	биолошки инспирисан алгоритам заснован на мембранском прорачуну
MHSO	модификона оптимизација хармонијским претраживањем
FSA	филтер симулирано каљење
MFA	измењени алгоритам свица
IHSA	унапређени алгоритам хармонијског претраживања
TIHSA	двапут унапређени алгоритам хармонијског претраживања
GeneAS	комбиновани генетско прилагодљиво претраживање
EO	еволуциона оптимизација
DE	диференцијална еволуција
KanGAL	Kanpur genetic algorithms laboratory – генетски алгоритам лабораторије Канпур
WCA	water cycle aglorithm – алгоритам воденог циклуса
ABC	artificial bee colony - алгоритам вештачке колоније пчела

SSO	swallow swarm optimization – оптимизациони алгоритам јата ластавица
IGA	immune genetic algorithm – имуни генетски алгоритам
FMO	fuzzy multi-objective optimization – фази вишекритеријумска оптимизација
WNNC	wide normalized normal constraint–метод широке нормализација нормалних ограничења

Поглавље

1

УВОД

1. УВОД

Поље деловања теорије оптимизације је развој модела и метода којима је могуће пронаћи оптимална решења проблема који се истражује. При томе, предмет истраживања се мора трансформисати из реалног модела у математички модел. То у ствари значи да, предмет оптимизације (компонента/производ/процес) мора бити описан на одговарајући, математички начин, односно мора се извршити његова анализа преко математичких функција, које могу бити линеарне или нелинеарне. Процес тражења оптимума је уствари процес тражења минимума или максимума функције која описује дати проблем. Оптимално решење у принципу значи најбоље решење за познате услове. Да би знали да ли је добијено решење оптимално, односно најбоље, оно мора имати меру којом ће се одредити квалитет решења, односно могућност упоређивања са другим решењима. Зато је потребно да у математичком моделу посматраног проблема, постоји функција којој се, при сваком израчунавању придружује добијена нумеричка вредност. Та функција се зове функција циља или критеријумска функција.

У већини оптимизацијских проблема, процес добијања оптималног решења је веома сложен и као такав захтева доста времена, а у данашње време, и рачунарских ресурса за његово спровођење. Како је неисцрпна претрага простора могућих решења практично неизводљива, као и немогућност аналитичког решавања сложених оптимизационих проблема, у пракси се често користе разне методе које нуде приближна, односно, довољно добра решења.

Постоји више критеријума за класификовање оптимизационих метода, међутим у принципу оне се могу сврстати у три методе: градијентне методе, методе директног претраживања и савремене методе оптимизације. Прве две методе карактерише недовољна ефикасност и као такве се могу применити на оптимизационе проблеме мање сложености. При томе, градијентна метода подразумева да функција циља мора бити бар два пута диференцијабилна и непрекидна. Друга ствар која ограничава примену градијентне методе, на сложене оптимизационе проблеме, је задавање почетних вредности које морају бити блиске жељеним вредностима. Са друге стране, методе директног претраживања не захтевају да функција циља буде диференцијабилна и непрекидна, и не зависе од почетног избора пројектних параметара, али су ове методе ефикасне само при налажењу решења у пољу локалних минимума.

Средином прошлог века, а нарочито, крајем XX и почетком XXI века, појавиле су се методе које на ефикасан начин решавају сложене оптимизационе проблеме. Карактеристика ових метода је та што су инспирацију нашле у одговарајућим појавама у природи. Због тога се те методе називају биолошки инспирисане методе. Међу најпознатијим, најпопуларнијим, методама су: генетски алгоритми (Genetic Algorithm – GA, John Holland 1962. године),

диференцијала еволуција (Differential Evolution – DE, R. Storn и K. Price крајем 90тих), оптимизација појем честица (Particle Swarm Optmization – PSO, J. Kennedy и R. Eberhart 1995. године), оптимизација инспирисана кретањем мрава (Ant Colony Optimization – ACO M. Dorigo крај 90тих прошлог века), кукавичја претрага (Cuckoo Search – CS, Yang, X. S. и Suash Deb, 2007. године), алгоритам свица (Firefly Algorithm – FA, Yang, X. S., 2008. године), алгоритам слепог миша (Bat Algorithm – BA, Yang, X. S., 2010. године), оптимизација инспирисана кретањем арктичког крила (Krill Herd Algorithm – KHA, A.H. Gandomi и A.H. Alavi, 2012. године) итд. Предност ових алгоритама је тај што се могу применити на велики број оптимизационих проблема. Следећа предност је та да није потребно искуство при одређивању почетних вредности пројектних променљивих, јер дају могућност постављања широког опсега за почетне вредности пројектних променљивих. Даље, функција која се оптимизира овим методама не мора бити диференцијабилна и непрекидна, нема ограничења у односу на број променљивих који се оптимизира. И можда најбитнија предност ових метода је та да су све оне алгоритамски структуриране и да се као такве могу надограђивати једноставним модификацијама чиме се постиже велика ефикасност при проналажењу оптималног решења.

У овом докторату фокус је усмерен на четири од горе поменутих метода: кукавичја претрага (Cuckoo Search – CS), алгоритам свица (Firefly Algorithm – FA), алгоритам слепог миша (Bat Algorithm – BA), оптимизација инспирисана кретањем арктичког крила (Krill Herd Algorithm – KHA) и њихова примена на проблеме оптимизације из области примењене механике. У оквиру рада направљене су одређене модификације и хибридизације поменутих метода у циљу добијања универзалног алгоритма који може да се примени при решавању различитих оптимизационих проблема у машинству, односно примењеној механици у циљу добијања најбољих решења.

Како је већ раније написано, биолошки инспирисани алгоритми се све више користе за решавање нелинеарних проблема оптимизације. У другом поглављу дат је преглед основних појмова оптимизације, са посебним освртом на метахеуристику и као посебну класа метахеуристичких алгоритама, дат је преглед оптимизационих алгоритама који су нашли инспирацију у природи. Наведени су најзначајнији алгоритми са временом појављивања и ауторима који су их предложили, као и прегледна литература за сваки од поменутих алгоритама. Као увод у следећа поглавља, објашњени су појмови интнезификације, диверзификације и рандомизације, као основне карактеристике метахеуристике, односно природом инспирисаних алгоритама.

У трећем поглављу приказана је модификација стандардног BA алгоритма, у циклични алгоритам фамилије слепих мишева Loop BFA, која се одвија у три корака. Први корак подразумева начин израчунавања фреквенције за сваког слепог миша у оквиру једне фамилије. Други корак представља увођење фамилије

слепих мишева, како би се континуално понављала претрага простора могућих решења у циљу добијања оптималног решења. Следећа модификација иде у правцу финог цикличног претраживања простора решења. За сваког слепог миша, у фамилији, финим претраживањем по кораку Левијевог лета тражи се побољшано решење све док се не задовоље постављена ограничења. Исправност и ефикасност предложених модификација су верификовани кроз добре резултате добијене тестирањем на пет различитих инжењерских оптимизационих проблема из области примењене механике (суд под притиском, заварени носач, редуктор, конусна опруга, и ламеласта кочница са више дискова). Вредности функције циља, која је добијена модификацијом стандардног BA, односно применом Loop BFA, у неким примерима има најмању вредност у односу на примењене различите оптимизационе методе или има исту најмању вредност у поређењу са другим методама. Стандардно одступање у свим примерима има такође задовољавајуће малу вредност.

У циљу добијања што бољих резултата и ефектнијих алгоритама, многи аутори су поред модификације стандардних оптимизационих алгоритама, прибегавали и комбиновању два и више различитих оптимизационих алгоритам. У поглављу 4, предложени хибридни алгоритам, H–CS–FA алгоритам, за основу има алгоритам кукавичје претраге – CS, у који је инкорпориран део алгоритма свица – FA, што је приказано у Алгоритму 4.3 и на Слици 4.1. Наиме, уместо да се испразне гнезда када се достигне вероватноћа, p_a, проналажења „лоших“ гнезда, у алгоритам је додат део FA алгоритма у коме се проналази свитац који је са највећим интезитетом светлости. Усвојени концепт хибридног алгоритма тестиран је на пет оптимизационих модела: опруге, конусног квачила, I профила, мењача и носача са три промењљиве које се оптимизирају. Модели I профила и мењача спадају у групу вишекритеријумске оптимизације (multiobjective optimization - MOO). Упоређујући резултате, за свих пет примера, из цитиране литературе и H-CS-FA алгоритма, може се закључити да у свим случајевима, примењени хибридизовани алгоритам даје приближне и боље резултате, од алгоритама од којих је формиран.

Један од новијих алгоритама који је стекао популарност је алгоритам крила – KH, који су увели A.H. Gandomi и A.H. Alavi, 2012. године. Модификација овог алгоритма у модификовани алгоритам крила – MKH , и његова примена у димензионалној синтези четворочланих механизама као генератора путање, приказана је у поглављу пет. Модификације су се односиле на: иницијално одређивање позиције плена пре почетка итеративног процеса и замену генетског оператора укрштања са случајним комбиновањем добијених колона који су добијени у једној итерацији. MKH акгоритам је тестиран на четири референта примера из синтезе зглобног четвороугаоног механизма. Резултати добијени овим алгоритмом значајно су бољи у поређењу са резултатима добијеним у цитираној литератури. Грешка, која представља квадрат одступања између задатих тачака и

стварних тачака на путањи, у сва четири примера је мања у поређењу са резултатима из цитиране литературе.

Примена и ефикасност модификованих алгоритама: Loop BFA, H–CS–FA и МКН, проверена је у поглављу шест, на два нова модела: модел тела стругарског ножа и модел ексцентра, као дела стезног прибора. На основу добијених резултата, може се закључити да наведени алгоритми исправно функционишу. Конвегренција је добра, уз високу вредност стандардне девијације за оба примера и сва три алгоритма. Ово се може објаснити чињеницом да је у питању вишекритеријумска оптимизација, две противуречне функције циља.

Програми коришћени у поступку оптимизације написани су у програмском пакету MATLAB – Release 2009b. Симулација добијених механизама, у поглављу пет, односно дијаграми конвергенције (поглавље 3, 4, 5 и 6) такође су урађени у истом програму.

Поглавље

2

БИОЛОШКИ ИНСПИРИСАНИ ОПТИМИЗАЦИОНИ АЛГОРИТМИ

2. БИОЛОШКИ ИНСПИРИСАНИ ОПТИМИЗАЦИОНИ АЛГОРИТМИ

Концепт инжењерства потиче још од времена фундаменталних цивилизацијских иновација као што су точак и полуга. Свака од наведених иновација представља основу за модерну дефиницију инжењерства која гласи: "коришћење основних математичких принципа за развијање корисних алата или објеката". Захваљујући њима људи су успели направити грандиозне грађевине као што су на пример египатске пирамиде, или пак машине као што је парна машина [1].

Инжењерство је у много чему специфично. Једна од специфичности произилази из примене научних достигнућа из више различитих области, при пројектовању, анализи и/или конструисању разних врста производа и/или технологија. Друга специфичност је та да се та, научна, достигнућа остварују кроз знање, математички апарат и искуство при пројектовању употребљивих објеката (производ, грађевине и слично) или процеса. У ствари, може се рећи, да је инжењерство примена науке, технологије и искуства у решавању практичних проблема.

Многи сматрају да је инжењерство истовремено дисциплина, уметност и професија концентрисана на примену решења практичних проблема. Док научник увек поставља питање "зашто" и врши истраживања да нађе одговор на то питање, инжењер жели да "зна како" да реши проблем и како да примени решење. Научник истражује феномене који већ постоје улазећи у непознато, док инжењер креира оно што никада није постојало користећи оно што је познато.

У машинству, основни задатак инжењера је да пројектује, на бази познатих научних пронципа и постулата, нови технолошки процес обраде/мотаже, носећу конструкцију (машине, моста, грађевине), нови алат, нови уређај, транспортна средства, грађевинске и рударске машине, нову организациону шему предузећа, распоред машина у погону, нову фабрику.

Већина од наведених задатака подразумева примену знања из области примењене механике. Ово је због тога што при пројектовању увек утврђује (проверава) на који начин појединачни део (тело) или склоп делова (систем тела) реагује на дејство спољних сила, који су отпори ослонаца, какво је напонско стање, који је утицај кретања једне компоненте (тела) на остале компоненте у монтажном склопу (систему тела), какав је енергетски биланс и слично. При томе, у процесу пројектовања, решење које је пројектовано није и најбоље, ако не задовољава основне критеријуме сигурности, поузданости и ефективности и ефикасности, а да при његовој реализацији не изискује најмање трошкова, без обзира да ли је реч о количини материјала која се уградила, времену и

трошковима израде. Последње речено у ствари осликава и једну посебну грану инжењерства која се бави оптимизацијом.

Оптимизација, у инжењерском смислу, је поступак максимизације или минимизације, неког или више циљева у оквирима расположивих ресурса уз задовољавање постојећих ограничења. У овом смислу свака донета одлука је резултат неког свесног или несвесног мисаоног процеса оптимизирања. У начелу такву оптимизацију формално не доживљавамо као спроведени поступак, јер се догађа интуитивно или искуствено, на бази раније одређених најбољих решења. Класични поступак генерисања решења, на бази методе покушаја и погрешака, може се сматрати поступком оптимизације, јер се заснива на генерисању нових, непознатих, решења посматраног проблема, односно сукцесивном одбацивању оних која не задовољавају ограничења или оних која су лошија од раније добијених решења [2].

Инжењерска оптимизација подразумева системско тражење оптималног решења инжењерског проблема. При томе се води рачуна о критеријуму оптималности, да ли је у питању максимум или минимум, а у условима задатих ограничења.

Савремени развој производа и/или процеса, не подразумева само проналажење неког решења које ће задовољити постављене пројектне услове. Поред задовољења тих услова, пронађено решење треба да задовољи минимум трошкова свих ресурса, односно да максимизира критеријуме ефективности пројекта.

У том смислу процес оптимизације је знатно шири од скупа нумеричких алата. У ствари, овде се говори о новом приступу у инжењерској синтези односно развоју производа и/или процеса на бази концепта оптималности. Због овога се, оптимизација може схватити и као начин постављања инжењерских задатака, али и као специјализовани алат односно начин подршке доношењу одлука у најширем смислу.

Уопштено, проблем оптимизације може се написати на следећи начин:

Minimum	$f(x)$	функција циља,
Ograničenja		
	$h_j(x)=0,\ \ j=1,2,...,n_h,$	функције ограничења у облику једначина,
	$g_k(x)<0,\quad k=1,2,...,n_k$	функције ограничења у облику неједначина,
	$x_i^d \le x_i \le x_i^g \quad i=1,2,...,n$	границе,

где је: $f(x)$ функција циља;

$x = x_1, x_2, ..., x_n$ независни параметри које треба одредити током оптимизационог поступка. Независни параметри којим се описује дати проблем често се зову пројектни параметри или пројектне променљиве;

n_h је број једначина ограничења;

n_k је број неједначина ограничења. Ова ограничења, представљају област у којој се мора наћи решење оптимизационог проблема;

x_i^d и x_i^g представљају доњу и горњу границу за $i-$ту пројектну променљиву и

x_i $i-$та пројектна променљива.

Како се у овој дисертацији, на решавање оптимизационих проблема, користе биолошки–инспирисане методе, које припадају групи метахеуристичких метода, важно је напоменути да се при оптимизацији, овим методама, не добијају тачна, већ приближна решења која их са довољном тачношћу замењују.

2.1. ПОДЕЛА ОПТИМИЗАЦИОНИХ АЛГОРИТАМА

Постоји више критеријума за класификацију оптимизационих алгоритама. Једна од подела алгоритама је на детерминистичке (deterministic) и стохастичке (stohastic).

Код детерминистичких алгоритама ће се, при сваком извршавању, под било којим условима, од истог улаза доћи до истог излаза, следећи сваки пут исту путању. Међутим, за вишедимензионалне проблеме и сложене функције (функције са више локалних минимума), детерминистички алгоритми не могу да пронађу оптимално решење, зато што је за његово проналажење потребно доста времена и изузетан рачунарски ресурс.

Супротно од детерминистичких алгоритама, стохастички оптимизациони алгоритми се односе на алгоритме који претраживање простора могућих решења базирају на случајности. То значи да вредност пројектних променљивих, утиче на избор наредног корака у итеративном процесу одређивања решења. При сваком новом покретању, генерално посматрано, стохастички алгоритам кретаће се различитим путањама, у зависности од случајно изабране вредности пројектних променљивих. Случајна вредност пројектних променљивих, који одређује ток кретања алгоритма, се симулира помоћу генератора псеудо-случајних бројева.

Даља подела према [4], оптимизациони алгоритми се могу поделити на егзактне и на хеуристичке. Егзактни алгоритми, као што им само име и каже,

одговарају класи детерминистичких, а хеуристички су итеративни алгоритми. Хеуристичке методе су карактеристичне по томе да, најчешће, проналазе задовољавајућа решења у релативно кратком временском периоду.

За решавање многих практичних и референтних проблема није могуће користити егзактне методе зато што би за њихово решавање било потребно много времена. За разлику од егзактних метода оптимизације које гарантују проналажење оптималног решења, хеуристички алгоритми покушавају да пронађу што боље решење, али не могу да гарантују да је пронађено решење и оптимално [5, 24].

Једна од основних карактеристика хеуристике је та да се примењују за решавање конкретних и специфичних проблема тако што користе својства самих тих проблема при њиховом решавању. Реч хеуристика потиче од старогрчке речи *heuriskein*, што значи уметност проналажења новог начина (правила) у решавању проблема. Може се рећи да је примена хеуристике, почела 40тих година прошлог века, али да је крајем XX и почетком XXI века дошло до наглог развоја хеуристичких метода за решавање разних оптимизационих проблема.

Како су се хеуристичке методе и алгоритми показали као ефикасни оптимизациони апарат, при решавању великог броја проблема, многи истраживачи широм света, током поменутог периода, су показали велико интересовање за њихов развој и унапређивање. Учињени напор је довео до настанка метахеуристика, као универзалних хеуристика за широк спектар проблема, [24] .

Према Е. Г. Талби, у [6], метахеуристика представља скуп алгоритамских концепата који се користе за дефинисање хеуристичких метода које су применљиве на широк спектар проблема. Такође, метахеуристику можемо дефинисати и као универзалну хеуристику, чији је главни задатак усмеравање проблемски специфичних хеуристика према подручју простора претраживања у којем се налазе добра решења.

У литератури постоји велики број критеријума за поделу метахеуристика. Према [6], метахеуристике можемо поделити на:

- биолошки инспирисане метахеуристике (bio – inspired) и на оне које нису биолошки инспирисане (non bio – inspired). Настанак многих метахеуристичких метода је инспирисан процесима који се дешавају у природи, као што су на пример еволутивни алгоритми и вештачки имуни системи. Такође су и многи животињски системи послужили као инспирација за развој ових метода, као што су, на пример, колоније мрава и пчела, јата птица и риба, итд. Осим поменутог, у литератури се проналазе и метахеуристике које су инспирисане физичким процесима,

као што је алгоритам симулираног каљења;

- метахеуристике базиране на популацији решења (population-based) и метахеристике које користе само једно потенцијално решење (single-solution based). У другу групу спадају методе које у току процеса претраге манипулишу и врше трансформацију само једног решења. С друге стране, популационе метахеуристике, у које спадају, на пример, оптимизација ројевима честица и еволутивни алгоритми, врше претрагу коришћењем популације потенцијалних решења проблема. Метахеуристике које користе једно потенцијално решење углавном користе интензификацију у претраживању простора. Са друге стране, популационе метахеуристике су оријентисане ка диверзификацији. Због комплементарних карактеристика, ове метахеуристике се најчешће користе заједно;
- детерминистичке и стохастичке метахеуристике. Детерминистичке метахеуристике приступају решавању проблема оптимизације тако што доносе детерминистичке одлуке (примери су локална и табу претрага). Стохастички методи примењују случајна правила у процесу претраге (примери су симулирано каљење и еволутивни алгоритми). Код детерминистичких метахеуристика иста почетна популација увек генерише исто коначно решење, док код стохастичких метахеуристика могу да се генеришу другачија коначна решења помоћу исте почетне популације;
- метахеуристике које користе меморију (имају могућност памћења претходних решења (memory usage methods)) и на оне које не користе меморију (немају могућност памћења претходних решења (memoryless methods)). Алгоритми који не користе меморију немају могућност динамичког издвајања информација током процеса претраге. Један од представника ове врсте је, претходно поменута, метахеуристика симулираног каљења. С друге стране, многи алгоритми, током претраге, користе систем активног издвајања меморијских података, као што је на пример краткорочна и дугорочна меморија које поседују јединке табу претраге.

2.2 ИСТОРИЈСКИ РАЗВОЈ МЕТАХЕУРИСТИКА

Упркос чињеници да смо део природе и да живимо по природним законима, метахеуристика је модерна метода која се користи за решавање оптимизационих проблема. Како се не може са сигурношћу утврдити када је први пут коришћен метод метахеуристике, претпоставља се да је Alan Turing, први користио хеуристичке алгоритме за разбијање шифри током II светског рата, [8].

Средина XX века, 1960-те и 1970-те године, представља значајан период за развој еволуционих алгоритама [8]. Наиме, на Универзитету у Мичигену, John

Holland и његови сарадници су, током поменутих година, развили генетски алгоритам. Идеја је настала услед проучавања адаптивних система, где је Holland, први (1962. године), применио укрштање (crossover) и рекомбиновање (recombination) за моделирање таквих система. De Jong је 1975. године у својој дисертацији показао какав је потенцијал генетских алгоритама. За овај период је значајан развој еволутивне стратегије из 1963. године. Ingo Rechenberg и Hans-Paul Schwefel са Техничког универзитета у Берлину, развили су ову технику претраживања за решавање проблема аеронаутичког инжењерства, [8].

Значајна година за развој метахеуристике је 1983. Те године је развијен метод симулираног каљења од стране S. Kirkpatrick, C. D. Gellat и M. P. Vecchi инспирисан процесом каљења метала, [9].

До појаве Tabu претраживања, развијеног од стране Fred Glover, 1986-те, [10], већина метахеуристика није меморисала претходна решења, осим код селекције најбољег избора.

Почетком 90-тих година прошлог века, Marco Dorigo је у својој докторској тези о оптимизацији и природним алгоритмима, описао алгоритам оптимизације колоније мрава (ant colony optimisation – ACO), [11]. Ова техника претраживања је инспирисана интелигенцијом роја мрава, односно њиховим друштвеним понашањем при чему користе феромоне као хемијске носаче порука.

Алгоритам вештачког имуног система, инспирисан је карактеристикама имуног система сисара и користити меморију и учење као нов приступ решавању проблема. Овај алгоритам су, 1986. године, први предложили Farmer и остали у [12]. На основу овог алгоритма, Bersini и Varela су 1990. године развили имуну мрежу, као адаптивни систем са високим потенцијалом, [13], односно развијено је много других варијанти укључујући алгоритам селекције клона, алгоритам негативне селекције и друге.

Значајан напредак је направљен развојем отимизационог алгоритмаројем честица (particle swarm optimisation – PSO) 1995. године, од стране J. Kennedy и R. Eberhart [14]. Овај алгоритам је инспирисан интелигенцијом јата риба и птица, па чак и људског понашања. Вишеструки агенти, честице, окупљају се у рој око простора за претраживање, почевши од неког случајног иницијалног претпостављеног решења. Рој проналази тренутно најбоље и упоређује га са тренутно глобално најбољим решењем, како би се изабрао најквалитетнија решења.

R. Storn и K. Price крајем 90тих, прошлог века, развили су еволуциони алгоритам: диференцијална еволуција (differential evolution - DE), [15]. У то време, појављује се још један интересантан метод, укрштена ентропија (cross-

entropy), [16], која представља уопштен метод Монте Карло (Monte Carlo method). Овај метод, који се састоји од две фазе: генерисање случајног примера и ажурирање параметара, развио је Rubinstein.

Почетком 21-вог века развијају се нове оптимизационе методе. Zong Woo Geem и остали у [17], представљају алгоритам хармонијске претраге (harmony search – HS), који налази широку примену у решавању различитих оптимизационих проблема, као што су дистрибуција воде, моделирање транспорта и распоређивања. Алгоритам инспирисан понашањем бактерије Escherichia coli при њеном трагању за храном (оптимизација бактеријске потраге за храном), развио је K. M. Passino 2002. године, [18].

За оптимизацију интернет хостинг центра, развијен је алгоритам медоносних пчела (honey bee algorithm – HBA), 2004. године, од аутора S. Nakrani и C. Tovey [19], што је узроковало развој новог алгоритма пчела, 2005. године од стране D. T. Pham и осталих, [20] и вештачке колоније пчела од стране D. Karaboga [21].

У 2008. години, Mucherino и Seref, предложили су алгоритам претраге мајмуна, заснованог на понашању мајмуна приликом потраге за храном, [22]. Исте године, Yang, X. S. развија алгоритам свица (firefly algorithm – FA), [23], да би исти аутор, 2009. године са Suash Deb представио алгоритам кукавичје претраге (cuckoo search – CS), [61], односно 2010. године, алгоритам слепог миша (bat algorithm - BA), [30].

Један од новијих алгоритама који се појавио, а такође припада групи метахеуристичких метода је алгоритам крила (krill herd algorithm – KHA). Овај алгоритам развили су A.H. Gandomi и A.H. Alavi, 2012. године, [114].

2.3 ИНТЕНЗИФИКАЦИЈА И ДИВЕРЗИФИКАЦИЈА МЕТАХЕУРИСТИКА

Као што смо већ поменули, метахеуристика може на довољно ефикасан начин да пронађе прихватљива решења сложених проблема, путем покушаја и грешке у разумно изводљивом времену. Комплексност изучаваног проблема нам не дозвољава да истражимо свако могуће решење или комбинацију решења, у циљу да се пронађе добро решење у разумном временском интервалу. При томе, не постоји гаранција да је пронађено решење уједно и најбоље решење. Због тога, основни циљ је пронаћи ефикасан али и практичан алгоритам, који ће радити довољно дуго и за то време давати прихватљиве резултате. Међу тим пронађеним добрим решењима, очекује се да је најбоље међу њима близу оптимума, али се оптималност не гарантује.

Међутим, многи метахеуристички алгоритми обично имају особину глобалне конвергенције, чиме у пракси, они могу да пронађу глобални оптимум у коначном броју итерација. Ово особина их практично и препоручује за решавање проблема глобалне оптимизације. На пример, алгоритам кукавичје претраге [61], поред технике претраживања са добром конвергенцијом, користи и рандомизационе технике у циљу повећања ефикасности Левијевог лета.

Да би метахеуристички алгоритми били ефикасни, они морају да имају неке специфичне карактеристике. Једна од тих карактеристика је да имају способност да генеришу нова решења, која су боља у односу на претходна, претражујући простор где лежи глобални оптимум. Друга карактеристика је та да метахеуристички алгоритам мора да препозна замку уласка у локални оптимум, и да има механизам који ће га извести из тог подручја претраживања. Добра комбинација ових карактеристика, доводи под одређеним условима, доброј ефикасности алгоритма, што опет захтева добро уравнотежење две главне компоненте било које метахеуристике: истраживања (exploration) и експлоатације (exploitation), односно интнезификације и диверзификације, [24].

Диверзификација значи да се генеришу различита решења, претражуји простор могућих решења, на глобалном нивоу, док интензификација значи фокусирање на претраживање локалне области користећи информације о добрим решењима нађеним у том региону.

Истраживање, интензификација, често користи рандомизацију, која алгоритму даје способност да напусти простор локалног минимума. Она се такође може искористити за претраживање локалног простора око тренутно најбољег решења уколико је корак ограничен на локалном подручју. Уколико је корак довољно велики, рандомизација може претраживати простор на глобалном нивоу.

2.4 БИОЛОШКИ ИНСПИРИСАНИ ОПТИМИЗАЦИОНИ АЛГОРИТМИ

Биолошки инспирисани оптимизациони алгоритми (bio-inspired algorithms) припадају групи метахеуристичких оптимизационих метода које симулирају природне процесе и системе када изводе активности претраге простора потенцијалних решења, [7]. Међу овим алгоритмимима, специјална класа алгоритама која се развила, била је инспирисана интелигенцијом појева (swarm intelligence – SI). Анализирајући доступну литературу, може се закључити да био-инспирисани алгоритми засновани на интелигенцији појева спадају међу најпопуларније. Добри примери за ове алгоритме су: алгоритам мравље колоније (ant colony optimization – ACO), алгоритам роја честица (partical swarp optimization – PSO), алгоритам кукавичје претраге (cuckoo search – CS), алгоритам слепог миша (bat algorithm – BA), алгоритам свица (firefly algorithm – FA) и алгоритам

крила (krill herd algorithm – KH).

SI се односи на колективно понашање више, међусобно повезаних агената који прате нека једноставна правила. Иако се сваки агент може сматрати неинтелигентним, цео систем вишеструких агената, може показивати неко самоорганизовање и тај начин се може сматрати да се понаша као нека врста колективне интелигенције.

Сви SI алгоритми који користе мулти-агенте, инспирисани су друштевним понашањем инсеката, као што су: мрави, теримити, пчеле, осице, односно других животиња као што су птице или рибе. Класични алгоритам оптимизације ројем честица – PSO, користи понашање јата птица или риба, док рецимо FA алгоритам за инспирацију има светлуцање свитаца при размножавању и заштити од предатора. CS алгоритам је искористио паразитско понашање кукавица приликом полагања јаја, док је BA алгоритам за инспирацију искористио сонар микро слепих мишева при оријентацији у току лета, односно при лову. Са друге стране, KH алгоритам је нашао инспирацију из понашања арктичког крила при окупљању у велика јата и одржавању густине мега јата приликом одбране од грабљиваца и приликом храњења.

Једна од главних карактеристика SI алгоритама је та да мулти-агенти деле информације између себе, тако да смоорганизовање, ко-еволуција и учење током итеративног процеса обезбеђују високу ефикасност већине поменутих SI алгоритама. Друга карактеристика је та да се мулти-агенти могу паралелно кретати кроз алгоритам, што је веома практично када је у питању оптимизација великих размера.

У класи био – инспирисаних алгоритама, постоје алгоритми који нису засновани на интелигенцији ројева. На пример, генетски алгоритам је био-инспирисани алгоритам, али није SI алгоритам. Даље, неке алготитме је веома тешко класификовати. Такав алгоритам је диференцијална еволуција (differential evolution – DE). Уколико стриктно посматрамо, DE није био – инспирисан алгоритам, јер не постоји веза са било каквим биолошким понашањем. Међутим, како постоје одређене сличности са генетским алгоритмом, а и због речи еволуција, може се условно класификовати у категорију био–инспирисаних алгоритама.

Међу класом био–инспирисаних алгоритама који нису и SI алгоритми спадају, [7]: атмоферски модел облака (Yan and Hao), биогеографски заснована оптимизација (Simon), Brain Storm оптимизација (Shi), диференцијална еволуција (Storn and Price), алгоритам цвећа, алгоритам опрашивања цвећа (Xin-She Yang), алгоритам ехолокатора делфина (Kaveh and Farhoudi), алгоритам колоније термита (Hedayatzadeh и други).

Поглавље

3

МОДИФИКОВАНИ АЛГОРИТАМ СЛЕПОГ МИША

3. МОДИФИКОВАНИ АЛГОРИТАМ СЛЕПОГ МИША ЗА ОГРАНИЧЕНУ ОПТИМИЗАЦИЈУ

3.1. ПОНАШАЊЕ СЛЕПИХ МИШЕВА

Слепи мишеви припадају групи сисара са могућношћу летења. Поред ове карактеристике, јединствене међу сисарима, слепи мишеви имају способност коришћења ехолокатора.

Постоји око 1100 врста слепих мишева широм света. Распрострањени су готово по целој планети, нема их само у поларним подручјима и на острвима који су јако удаљени од копна. Величина им варира, од минијатурног, бумбар слепог миша, тежине од 1.5 до 2g, до гигантског слепог миша са распоном крила од око 2m, и тежине око 1kg, [25, 26].

Већина слепих мишева су инсектоједи. Приликом потраге за пленом, слепи мишеви користе сонар – ехолокатор. Њиме се служе приликом летења за лоцирање плена и препрека. Механизам ехолокатора је „једноставан“. Наиме, приликом летења, слепи миш испушта врло гласан звучни импулс и онда ослушкује одјек који се одбија назад од околних објеката (препрека или плен). У зависности од врсте, импулси које емитују могу да имају промењиве карактеристике, што зависи од стратегије која се примењује у лову. Приликом ехолокације, већина слепих мишева користе фреквенто модулисане сигнале које померају до једне октаве, док други чешће користе сигнале константне фреквенције. Ширина опсега сигнала варира у зависности од врсте и може се повећати коришћењем више хармоника, [28].

3.2. АКУСТИКА ЕХОЛОКАТОРА

Иако сваки импулс траје само неколико хиљадитих делова секунде (до око 8 до 10ms), он се емитује константном фреквенцијом која је обично у опсегу од 25kHz до 150kHz. Типичан опсег фреквенција за већину врста слепих мишева је између 25kHz и 100 kHz, мада неке врсте могу емитовати и више фреквенције, до 150kHz. Сваки ултразвучни рафал обично траје од 5 до 20ms. Слепи мишеви, емитују између 10 и 20 таквих звучних рафала у секунди. Када је у потрази за пленом, стопа емитовања импулса се може убрзати и до 200 импулса у секунди, поготово када су у близини плена. Веома кратко време и број импулса у том времену, подразумевају да слепи мишеви имају невероватну способност обраде звучних сигнала. У ствари, студије показују да је време обраде сигнала у уху слепог миша око 300 до 400μs, [28].

Како је брзина звука, $v = 340m/s$, дужина таласа λ, ултразвучног рафала са

константном фреквенцијом, f , је дата једначином (3.1). Вредност дужине таласа се креће у опсегу од 2 до 14mm, за опсег фреквенција од 25kHz до 150kHz, при чему наведени опсег одговара габаритима плена који слепи мишеви лове.

$$\lambda = \frac{v}{f} \tag{3. 1}$$

Емитовани импулси имају јачину до 110dB, који су у ултразвучном подручју. Јачина звука може да варира од највећих вредности када су у потрази за пленом, до нижих када се враћа са пленом. Домет емитованих импулса иде и до неколико метара у зависности од фреквенције емитовања [28]. При томе, су слепи мишеви у стању да идентификују препреку дебљине људске косе.

Студије показују да микро слепи мишеви користе време кашњења од емитовања ултразвучног таласа до детекције еха, временску разлику пријема између два уха, и варијацију јачине одјека да би изградили тродимензионалну слику простора који их окружује. Такође, могу да детектују удаљеност до мете и оријентацију објекта или плена у простору, чак и брзину којом се крећу инсекти које лове. Ово потврђују и студије којима се показује да су слепи мишеви у стању да селектују циљеве на основу различитих варијација Доплеровог ефекта проузрукованог вибрацијама крила циљаног плена, [28].

Овакво понашање слепих мишева, односно примена ехолокатора, искоришћено је да се повеже са функцијом циља која се оптимизира, односно за формулисање оптимизационог алгоритма. У даљем тексту, дат је основни алгоритам слепог миша (BA алгоритам), односно модификовани алгоритам (Loop BFA).

3.3. СТАНДАРДНИ АЛГОРИТАМ СЛЕПОГ МИША (BAT ALGORITHM – BA)

Да би се формирао алгоритам на основу понашања слепих мишева, у [30], идеализоване су неке од карактеристика ехолокатора код слепих мишева. Идеализација се огледа у следећем:

- Слепи мишеви користе ехолокатор да процене раздаљину, при чему разликују плен (храну) од препреке,
- Слепи мишеви лете насумично брзином v_i, на позицију x_i, са устаљеном фреквенцијом f_{min}, варирајући таласну дужину λ и јачину звука A_o, при тражењу плена. Такође, могу аутоматски да подесе таласну дужину (или фреквенцију) емитованих импулса и прилагоде брзину емисије импулса r у опсегу [0, 1] у зависности од близине плена.
- Иако јачина звука може да се мења на различите начине, претпоставка је да се мења од највеће (позитивне) A_o, до минималне константне вредности A_{min}.

Узимајући у обзир наведене, поједностављене претпоставке, због једноставности се уводе следеће апроксимације. Фреквенција се бира из опсега од [0, f_{max}], а брзина импулса r, се налази у опсегу [0, 1], где 0 значи да нема импулса, а 1 означава максималну брзину импулса.

На основу ових апроксимација и идеализације, формиран је стандардни BA алгоритам. Стандардни BA алгоритам са основним корацима, сажето је представљен кроз псеудо код дат у алгоритму 3.1.

Алгоритам 3. 1. Псеудо код BA алгоритма

```
Функција циља f(X), X=(x1, x2, ..., xd)^T
Иницијализација популације слепих мишева xi (i=1,2,...n) и vi
Дефинисање фреквенције импулса fi за xi
Иницијализација опсега импулса ri и јачине звука Ai
while (t < Max броја итерација)
        Генерисање новог решења прилагођавањем фреквенције,
       и ажурирање брзине и локације / решења (једначине (3.2.) до (3.4))
       if (rand>ri)
          Избор решења између најбољих решења
          Генерисање најбољег решења у околини изабраног најбољег решења
       end if
       Генерисање најбољег решења насумичним летом
       if (rand<Ai & f(xi)<f(x*))
          Прихватање новог решења
          Повећати ri и смањити Ai
       end if
      Рангирање слепих мишева и проналажење тренутно најбољег x*
  end while
  Постпроцесирање и приказивање резултата
```

3.3.1. Кретање слепих мишева

Виртуелни слепи мишеви, као и у природи се крећу са позиције и брзине , до нове позиције. Све се то дешава у d - димензионалном простору, где d представља број променљивих које се оптимизирају. Ажурирање нове позиције x_i^{new} и брзине v_i^{new}, одвија се кроз следеће кораке, [28]:

1. $f_i = f_{min} + (f_{max} - f_{min}) \cdot \beta$ – ажурирање фреквенције i-тог слепог миша, (3. 2)

2. $v_i^{new} = v_i^{new-1} + \left(x_i^{new} - x^{CB}\right) \cdot f_i$ – ажурирање брзине i-тог слепог миша, (3. 3)

3. $x_i^{new} = x_i^{new-1} + v_i^{new}$ – ажурирање позиције i-тог слепог миша, (3. 4)

где је: $\beta \in [0,1]$- случајни вектор преузет из униформне дистрибуције.

x^{CB} - најбоља глобална позиција – решење, добијена након поређења позиција свих слепих мишева у популацији.

Како је производ таласне дужине и фреквенције, $\lambda_i f_i$, величина за коју се мења брзина кретања виртуелног слепог миша, мењањем једне од величина λ_i или f_i, уз задржавање константне друге величине, утиче се на вредност промене брзине.

Након избора најбољег глобалног решења , коришћењем случајног корака, генерише се ново локално решење:

$$x_{new} = x_{old} + \varepsilon \cdot A^{av}, \tag{3. 5}$$

где је ε случајни број из опсега [0,1], док је A^{av} – просечна јачина звука од свих слепих мишева у истом кораку.

Ажурирање брзине и позиције слепих мишева има сличности са процедурама у стандардном алгоритму оптимизацијом роjем честица (Particle Swarm Optimization – PSO), [28], где је f_i у суштини регулише темпо и опсег хаотичног кретања честица. До одређене мере, може се рећи да BA алгоритам, представља уравнотежену комбинацију стандардног PSO алгоритма и интезивне локалне претраге управљане променом јачине и опсега звучног импулса.

3.3.2. Јачина звука и емисија импулса

Поред прилагођавања брзине избором било променом фреквенције f_i, било променом таласне дужине, λ_i, јачина звука A_i, и брзина звучног импулса r_i, морају да се ажурирају током итеративног поступка. Како у природи јачина звука опада када слепи миш локализује плен, јачина се може изабрати као било која погодна вредност, у зависности од проблема који се оптимизира. На пример, може се за почетну вредност усвојити $A_0 = 100$, а за крајњу вредност $A_{min} = 0$, или једноставније, може се усвојити $A_0 = 1$, а $A_{min} = 0$, где значи да је слепи миш пронашао плен и престао да емитује ултразвучне таласе. Током итеративног поступка вредности A_i, морају се налазити у границама $[A_0, A_{min}]$. Једначином (3.6) је дат начин одређивања вредности A_i и r_i.

$$A_i^{new} = \alpha \cdot A_i^{new-1}; \quad r_i^{new+1} = r_i^0 \cdot \left[1 - \frac{1}{e^{-\gamma t}}\right], \tag{3. 6}$$

У једначини (3.6) α и γ су константе и усвајају се у зависности од природе проблема који се оптимизира. Константа α се бира у распону (0,1), а γ је позитиван број, односно $\gamma > 0$. Уколико се поједностави, може се усвојити да су ове константе једнаке, $\alpha = \gamma$.

Да би се извршио избор параметара, BA алгоритма, потребно је експериментисати са различитим величинама, чији опсег зависи од проблема који се разматра. На пример, иницијална вредност јачине звука A_i^0 се може усвојити у опсегу [1,2], док почетна вредност брзине звучног импулса може бити 0 или било која вредност у распону [0,1]. Нове вредности за A_i и r_i се ажурирају, само уколико се у итеративном кораку добило боље решење, што у ствари значи да се слепи миш креће ка оптималном решењу.

3.4. МОДИФИКАЦИЈЕ СТАНДАРДНОГ BA АЛГОРИТМА

Многи аутори, су спроводили експерименте у циљу унапређења BA алгоритма. Тако, у [67] предложен је хибридни алгоритам, као комбинација BA и алгоритам хармонијске претраге, где се предлаже коришћење фиксних уместо променљивих вредности за фреквенцију *f* и јачину звука *A*. У алгоритам се уводи оператор мутације да би се повећала разноврсност популације, а у циљу повећања ефикасности претраживања и повећавања брзине конвергенције ка оптимуму. У [31], фактор инерције се уводи уместо случајног броја*: rand* $\in$ *[0,1]*. У свакој случајно генерисаној фреквенцији користили су растојања између тренутне позиције *j*-тог слепог миша, у *i*-тој итерацији, и најбољег решења у *i*-тој итерацији.

У неким истраживањима, као што је представљено у [32], јачина звука и опсег емитовања импулса се узимају у зависности од опсега пројектних променљивих.

У следећем поглављу описује се унапређење стандардног BA алгоритма.

3.5. ЦИКЛИЧНИ АЛГОРИТАМ ФАМИЛИЈА СЛЕПИХ МИШЕВА (Loop BFA)

Основни задатак у примени метахеуристичких метода за оптимизацију инжењерских проблема је изналажење универзалног алгоритма, који ће за већину проблема дати глобална решења у релативно кратком временском року и са довољним бројем понављања у итеративном циклусу. Отежавајућа околност су и ограничења која се морају задовољити за дати инжењерски проблем.

Истражујући на који начин решити овај задатак, први приступ који је примењен је формирање фамилија слепих мишева (Bat Family - BF). Наиме, да би се избегло узастопно покретање алгоритма, уводи се нова променљива *NumF* која се односи на број фамилија слепих мишева. Свака од фамилија има одређен број јединки слепих мишева (од 20 до 40), и за сваку фамилију се бележи оптимално

решење, од којих се поново бира најбоље, Алгоритам 3.2, [164].

На овај начин се избегава узастопно покретање алгоритма, и практично, свака следећа фамилија, има за „задатак“, да достигне до тада пронађено најбоље решење. Сада је остало да се пронађе начин како задовољити и постављена ограничења.

Алгоритам 3. 2. Псеудо код алгоритма фамилије слепих мишева (BFA)

```
Функција циља f(X), X=(x1, x2, ..., xd)^T
Иницијализација фамилије слепих мишева (j=1,2,…,NumF)
while k ≤ NumF %% Прва модификација
        Иницијализација популације слепих мишева x_i (i=1,2,...n) и v_i
        Дефинисање фреквенције импулса f_i за x_i
        Иницијализација опсега импулса r_i и јачине звука A_i
        while (t < Max броја итерација)
                Генерисање новог решења прилагођавањем фреквенције, и
                ажурирање брзине и локације/решења (једначине (3.2.) до (3.4))
                if (rand>r_i)
                        Избор решења између најбољих решења
                        Генерисање најбољег решења у околини изабраног && најбољег решења
                end if
                Генерисање најбољег решења насумичним летом
                if (rand<A_i & f(x_i)<f(x*))
                        Прихватање новог решења
                        Повећати r_i и смањити A_i
                end if
                Рангирање слепих мишева и проналажење тренутно && најбољег у фамилији x*_j
        end while
        Рангирање најбољих решења фамилије ( x*_1, x*_2, ..., x*_j,..., x*_NumF ) и && проналажење глобално најбољег решења x*
  end while %% Крај прве модификације
  Постпроцесирање и приказивање резултата
```

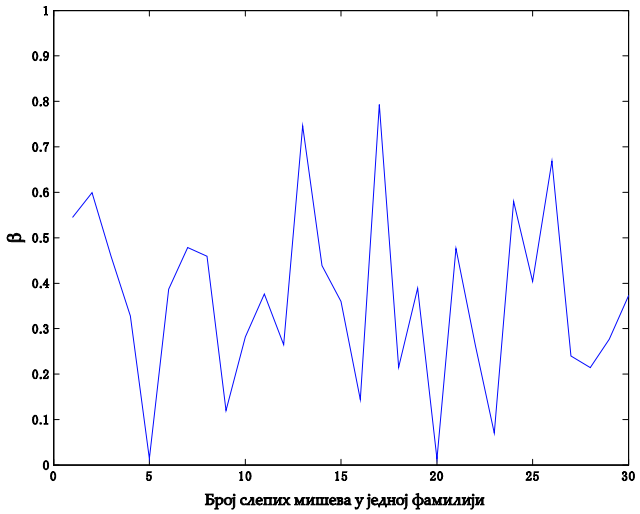
Истраживања су ишла у три корака. Први корак подразумева начин израчунавања фреквенције за сваког слепог миша, једначина (3.7). За израчунавање фреквенције уместо параметра β , у [33] користе се мапе хаоса, које такође генеришу случајан број у интервалу од [0, 1] . Водећи се тим искуством, у примерима који следе, а након испитивања различитих функција које у својој природи имају таласне особине, дошло се до израза за израчунавање фреквенције који је дат у једначини (3.7), при чему се параметар β, израчунава на следећи начин: $\beta = e^{1-rand} \cdot rand$.

$$f_i = f_{min} + \left(f_{max} - f_{min}\right) \cdot e^{1-rand} \cdot rand \qquad (3.\ 7)$$

где је:

f_{max} - максимална фреквенција и f_{min} - минимална фреквенција, које се бирају у зависности од проблема који се оптимизира.

Као што се може уочити, са Слике 3.1. новоизрачунати параметар β, се такође налази у опсегу од [0, 1]. Дистрибуција, приказана на слици, је урађена за једну фамилију од 30 слепих мишева.



Слика 3. 1. Дистрибуција параметра β (једна фамилија 30 – јединки)

Након одређивања фреквенције, приступа се тражењу решења и провере да ли решења задовољавају постављена ограничења. То доводи до другог корака у истраживању, односно, увођење цикличног претраживања.

Увођењем цикличног претраживања (Loop Searching), [165], Алгоритам 3.3, проблем задовољења граничних вредности промењљивих се ефикасно решава. Наиме, алгоритам је модификован тако да је уведен непрекидни циклус којим се траже решења која задовољавају постављена ограничења. Када се ограничења задовоље напушта се непрекидни циклус. На овај начин се у сваком итеративном кораку добија неко решење које задовољава граничне услове. У оквиру финог претраживања уводи се корак којим се модификују добијена решења у тренутној итерацији. Корак се рачуна по принципу Левијевог лета (Lévy-flight) [34, 35].

Левијевим (Lévy) летом, дефинише се случајни корак, којим неке птице/животиње, претражују животни простор за насељавање, односно претражују простор при потрази за храном. Величина корака је одређена по правилима Левијеве дистрибуције.

Општа једначина која приказује генерисање новог решења је:

$$\mathbf{X}_{i,new} = \mathbf{X}_{i,old1} + \alpha \otimes \text{Lévy}(\beta)\,, \tag{3. 8}$$

где је: Lévy(β) – случајни број усвојен по принципу Левијеве дистрибуције,

оператор $\otimes$, представља Кронекеров (Kronecker) производ, [163].

α – величина корака којом се управља опсегом претраживања и која зависи од оптимизационог проблема:

$$\alpha = \frac{u}{|z|^{\frac{1}{\beta}}}\,, \tag{3. 9}$$

где се u, z одређују по принципу нормалне расподеле, тако да је $u \sim N(0,\sigma^2)$ и $z \sim N(0,1)$, при чему се σ рачуна по једначини (3.10). У једначини (3.10), Г означава гама функцију, док је за вредност коефицијента β усвојено 1,5.

$$\sigma = \left(\frac{\Gamma(1+\beta)\sin(\frac{\pi\beta}{2})}{\Gamma(\frac{1+\beta}{2})\beta \cdot 2^{\frac{\beta-1}{2}}} \right), \tag{3. 10}$$

Алгоритам 3. 3. Псеудо код цикличног претраживања (Loop Serching)

```
%% Почетак цикличног претраживања – Друга модификација
if ограничења нису задовољена
        %% генерисање локалног решења све док се не заовоље ограничења
        Почетак циклуса
                Корак = f(Lévy-лет)
                f(Korak) %% Генерисање новог решења случајним летом
                if (rand < r_i)
                        Избор решења између најбољих решења
                        Генерисање најбољег решења у околини изабраног && најбољег решења
                        Избор решења од најбољих решења
                end if
        Крај циклуса
 end if
```

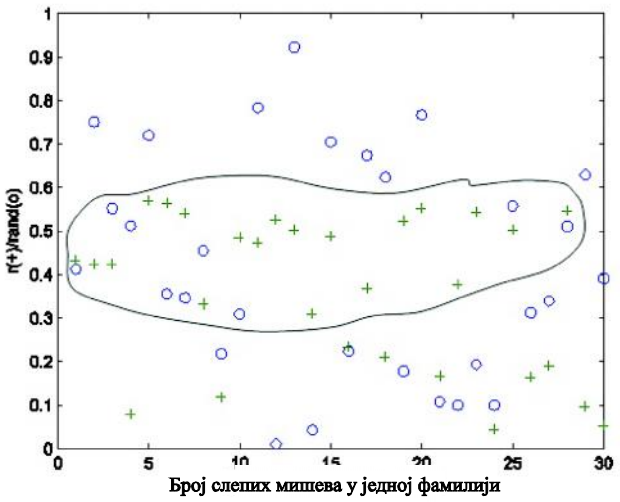
Корак Левијевог лета, се примењује у генерисању новог решења једино у случају достизања вероватноће да се плен налази у зони претраживања слепог миша, једначина (3.11).

$$\mathbf{X}_{i,new} = \begin{cases} X_{Best} + \alpha \times (\mathbf{X}(i) - X_{Best}), & rand_i > p_a \\ \mathbf{X}(i) & else \end{cases}. \qquad (3.\ 11)$$

Пронађено решење у оквиру једног пролаза код цикличног претраживања, не значи да смо нашли и најбоље. Сада се по изворном алгоритму, Алгоритам 3.1., испитује опсег звучног импулса, како би се пришло сужавању решења, односно ушло у процедуру тражења глобалног оптимума. Опсег звучног импулса *r*, дат је једначином (3.12):

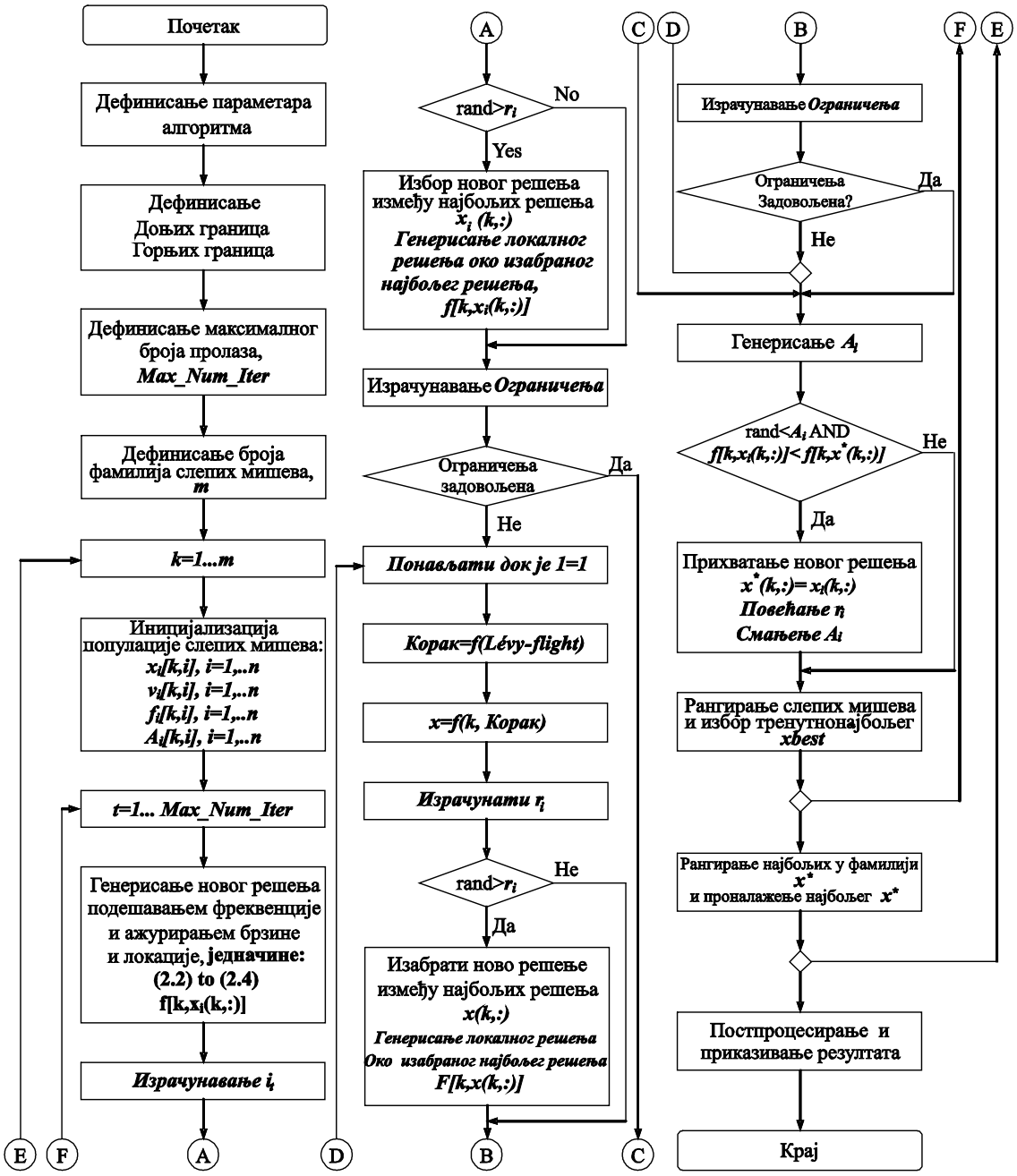
$$r = r_o \cdot \left(1 - e^{-\sin(rand)}\right). \qquad (3.\ 12)$$

На дијаграму приказаном на слици 3.2, јасно се уочава да, уважавајући принцип случајности, изабрани облик функције, омогућава алгоритму да улази у процес итерације, *rand* > *r* (уоквирена област), којим се сада тражи оптимално решење са финијим кораком претраживања.



Слика 3. 2. Расподела опсега импулса за за једну фамилију слепих мишева (30 јединки) + - израчуната вредност r, o – случајни број ∈ [0,1]

Коначно, трећи корак истраживања је омогућио да итеративни процес траје онолико пута, колико се унапред дефинише. Односно, уводе се фамилије слепих мишева, Слика 3.3. и Алгоритам 3.4, где се у оквиру сваке фамилије понавља циклично претраживање простора могућих решења. На овај начин пронађени најбољи резултат служи као референтна вредност за упоређивање са резултатима добијеним из осталих итерација (за сваку фамилију), што омогућава добијање глобалног решења.



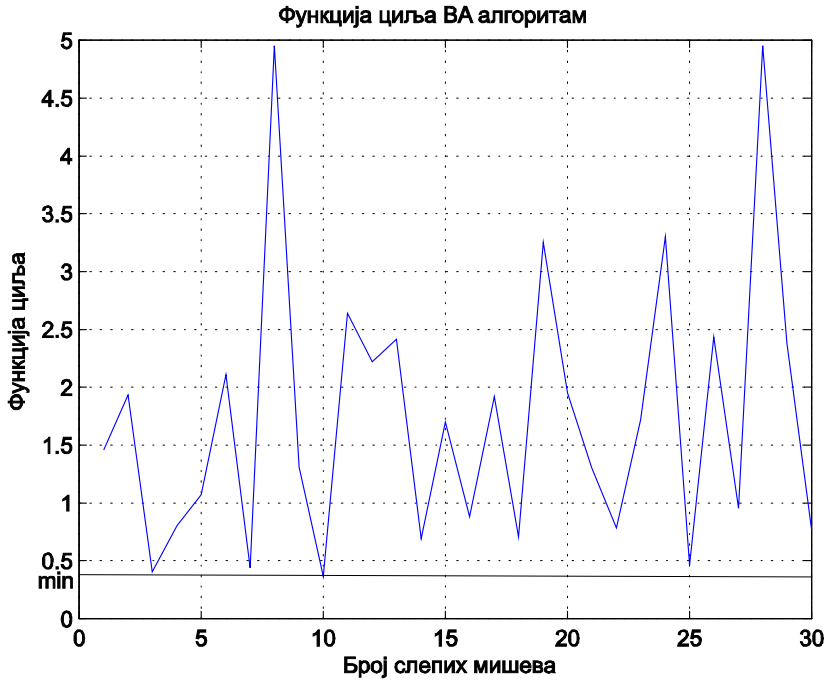
Слика 3. 3. Дијаграм тока цикличног алгоритма фамилије слепих мишева

Предност увођења ових модификација се може видети када се упореде вредности функције циља добијене током итеративног процеса, Слика 3.4 и Слика 3.5. Наиме, у класичном BA алгоритму, може се догодити, да се вредности функције циља значајно разликују од пронађеног оптималног решења, Слика 3.4, и итеративни поступак се мора понављати у циљу добијања оптималног решења. У цикличном алгоритму фамилије слепих мишева, Слика 3.5, може се уочити, да како расте број фамилија, свако следеће решење се мање разликује од оптимално пронађеног решења. За све бенчмарк примере, наведене у поглављу 3.6., процес проналажења оптималне вредности се понаша истоветно, као што је приказано на

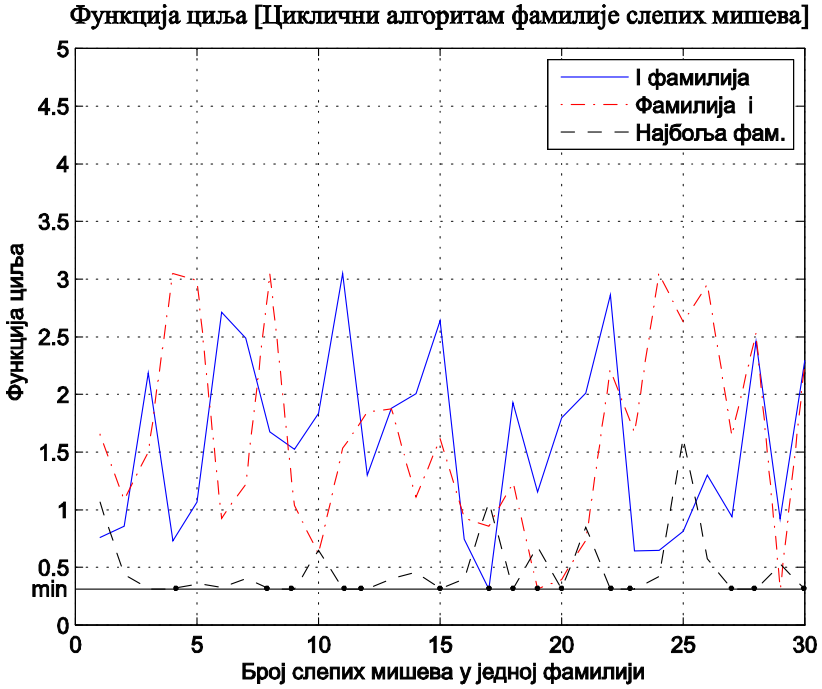
Сликама 3.4., и 3.8., које приказују вредности функције циља, за пример пројектовања ламеласте кочнице са више дискова.

Алгоритам 3. 4. Псеудо код цикличног алгоритма фамилије слепих мишева

```
1:  Функција циља f(X), X=(x1, x2, ..., xd)^T
2:  Иницијализација фамилије слепих мишева (j=1,2,...,m)
3:  while k ≤ m
4:      Иницијализација популације слепих мишева x_i (i=1,2,...n) и v_i
5:      Дефинисање фреквенције импулса f_i за x_i
6:      Иницијализација опсега импулса r_i и јачине звука A_i
7:      while (t < Max броја итерација)
8:          Генерисање новог решења прилагођавањем фреквенције, и &&
9:          ажурирање брзине и локације / решења (једначине (3.2.) до (3.4))
10:         if (rand>r_i)
11:             Избор решења између најбољих решења
12:             Генерисање најбољег решења у околини изабраног &&
                најбољег решења
13:         end if
15:         if ограничења нису задовољена %% Почетак цикличног
                                                претраживања
16:             %% генерисање локалног решења све док се не
                    задовоље ограничења
17:             Почетак циклуса
18:             Корак = f(Lévy-лет)
19:             f(Korak) %%Генерисање новог решења случајним
                                летом
20:             if (rand < ri)
21:                 Избор решења између најбољих решења
22:                 Генерисање најбољег решења у околини &&
                    изабраног најбољег решења
23:                 Избор решења између најбољих решења
24:             end if
25:             Крај циклуса
26:         end if %% Крај цикличног претраживања
27:         if (rand<A_i & f(x_i)<f(x^*))
28:             Прихватање новог решења
29:             Повећати r_i и смањити A_i
30:         end if
31:         Рангирање слепих мишева и проналажење тренутно најбољег
            у фамилији x_j^*
32:     end while
33:     Рангирање најбољих решења фамилије ( x_1^*, x_2^*, ..., x_j^*,..., x_NumF^* ) и &&
        проналажење глобално најбољег решења x^*
34: end while
35: Постпроцесирање и приказивање резултата
```



Слика 3. 4. Вредности функције циља за једну фамилију од 30 јединки, стандардни ВА алгоритам



Слика 3. 5. Вредности функције циља за прву фамилију, *i*-ту и најбољу фамилију у цикличном алгоритму фамилије слепих мишева

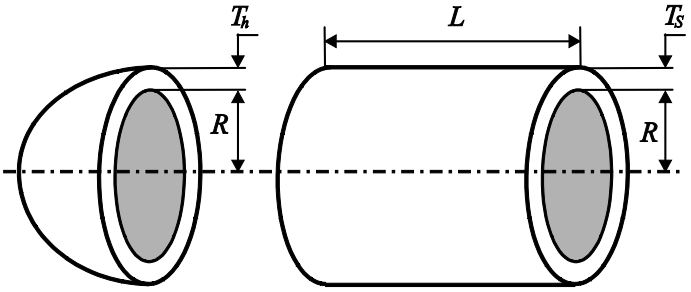
Применом ових модификација, добијање оптималних решења се постиже на ефикасан начин. Уколико се деси да се гранични услови не задовоље, итеративни процес се не напушта, већ се цикличним поступком поново генеришу нова решења која се константно проверавају. На крају, као последицу имамо добијање глобалног минимума.

3.6. РЕФЕРЕНТНИ ИНЖЕЊЕРСКИ ПРИМЕРИ

У овом поглављу, дају се резултати за 5 референтних примера, из области примењене механике, познатих у литератури. Такође, даје се компаративна анализа резултата преузетих из литературе, и резултата добијених предложеним цикличним алгоритмом фамилије слепих мишева.

3.6.1. Суд под притиском

Први пример односи се на минимизирање трошкова израде суда под притиском, Слика 3.6. Минимизирање укупних трошкова зависи од трошкова материјала, формирања и заваривања цилиндричне посуде под притиском. Овај проблем укључује четири пројектне променљиве: дебљину зида цилиндра T_s, дебљину данца T_h, унутрашњи полупречник цилиндра R, дужина цилиндричног дела посуде без данца L, односно: $[x_1 \;\; x_2 \;\; x_3 \;\; x_4] = [T_s \;\; T_h \;\; R \;\; L]$. Математичка формулација функције циља дата је једначинама од (3.13) до (3.17), [36]. Променљиве x_1 и x_2 су доступне дебљине ваљаних челичних лимова и као такве дефинишу се тако што се цели број помножи са 0.625in.



Слика 3. 6. Суд под притиском са пројектним променљивима

$$\text{Minimize } f(\mathbf{X}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \qquad (3.\ 13)$$

Функције ограничења:

$$g_1(\mathbf{X}) = -x_1 + 0.0193x_3 \le 0 \qquad (3.\ 14)$$

$$g_2(\mathbf{X}) = -x_2 + 0.00954x_3 \le 0 \qquad (3.\ 15)$$

$$g_3(\mathbf{X}) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0 \qquad (3.\ 16)$$

$$g_4(\mathbf{X}) = x_4 - 240 \le 0 \qquad (3.\ 17)$$

где је: $\mathbf{X} = (x_1, x_2, x_3, x_4)^T$

У радовима објављеним у референтним часописима, граничне вредности пројектних променљивих су се различито третирале. Узимајући у обзир ову чињеницу, приликом оптимизације суда под притиском, разматрана су четири случаја:

Случај a) $1 \cdot 0.0625 \le x_1, x_2 \le 99 \cdot 0.0625, 10 \le x_3 \le 200, 10 \le x_4 \le 200$;

Случај b) $1 \cdot 0.0625 \le x_1, x_2 \le 99 \cdot 0.0625, 10 \le x_3 \le 200, 10 \le x_4 \le 240$;

Случај c) $1 \le x_1, x_2 \le 99, 10 \le x_3 \le 200, 10 \le x_4 \le 200$ и

Случај d) $1 \le x_1, x_2 \le 99, 10 \le x_3 \le 200, 10 \le x_4 \le 240$.

Добијени резултати, применом цикличног алгоритма слепих мишева, упоређивани су са резултатима из цитиране литературе за сва четири случаја.

Приликом покретања алгоритма, за сва четири случаја, усвојени су следећи параметри алгоритма: број слепих мишева 30, број фамилија 50 и број итерација по свакој фамилији 1000.

Упоредни резултати, добијени коришћењем различитих метода из цитиране литературе приказани су у Табелама 3.1 до 3.4, за случајеве од a) до d) респективно.

Вредности пројектних променљивих као и статистичке вредности функције циља за случај a) дати су у Табели 3.1. Вредност функције циља добијена предложеним алгоритмом је иста као и у другим цитираним радовима, а стандардна девијација је нижа у поређењу са стандардним алгоритмом слепог миша предложеног у [36], док је најнижа вредности стандардне девијације добијена алгоритмом HGSO предложеним у [46].

Упоредни приказ резултата, за случај c), дат је у Табели 3.3. Из ове табеле може се видети да је вредност функције циља добијена цикличним алгоритмом слепог миша најнижа, при чему је стандардна девијација најнижа у односу на упоредне алгоритме, осим у случају резултата добијених алгоритмом B-IA приказаним у [45].

Табела 3. 1. Резултати оптимизације суда под притиском за Случај а)

	BA [36]	PSOLVER [37]	EHGA [38]	ABC [39]	TLBO [40]
x_1	0.8125	0.8125	0.8125	0.8125	0.8125
x_2	0.4375	0.4375	0.4375	0.4375	0.4375
x_3	42.0984456	42.0984	42.0984455959	42.098446	42.0984456
x_4	176.6365958	176.6366	176.636595842	176.636596	176.6365958
g_1	Н.П.	Н.П.	Н.П.	0.000000	-0.00000
g_2	Н.П.	Н.П.	Н.П.	-0.035881	-0.035880829
g_3	Н.П.	Н.П.	Н.П.	-0.000226	-0.00000
g_4	Н.П.	Н.П.	Н.П.	-63.363404	-63.36340416
Најбоље	6,059.7143348	6059.7143	6,059.7143	6,059.714339	6,059.714335
Средња вредност	6,179.13	6059.7143	6,059.7143	Н.П.	6,059.71434
Најлошије	6,318.95	6059.7143	6,059.7143	Н.П.	Н.П.
С.Д.	137.223	4.625E-12	2.8E-12	Н.П.	Н.П.

	CS [41]	HGSO [42]	**Loop BFA**
x_1	0.8125	0.8125	**0.8125**
x_2	0.4375	0.4375	**0.4375**
x_3	42.0984456	42.0984455958	**42.0984455958**
x_4	176.6365958	176.6365958424	**176.6365958429**
g_1	Н.П.	Н.П.	**-1.0599e-012**
g_2	Н.П.	Н.П.	**-0.035880829016**
g_3	Н.П.	Н.П.	**-0.000000**
g_4	Н.П.	Н.П.	**-23.3634041571**
Најбоље	6,059.714	6,059.714335048	**6,059.71433505**
Средња вредност	6,447.736	6,059.714335048	**6,059.71433506**
Најлошије	6,495.347	6,059.714335048	**6,059.71433518**
С.Д.	502.693	9,2504E-13	**2.59429E-08**

BA bat algorithm – стандардни алгоритам слепог миша, PSOLVER – нови хибридни оптимизациони алгоритам оптимизације ројем честица, EHGA ефективни хибридни генетски алгоритам са додатом флексибилном техником, ABC алгоритам artificial bee colony, TLBO оптимизација заснована на предавању-учењу, CS cuckoo search – претраживање кукавице, HGSO хибридни алгоритам оптимизације ројем честица и алгоритма свица.

Вредности пројектних променљивих као и статистичке вредности за случај b) су дате у Табели 3.2. Вредност функције циља добијена предложеним алгоритмом је иста као и у цитираној литератури, док је стандардна девијација најнижа.

Табела 3. 2. Резултати оптимизације суда под притиском за Случај b)

	C-EAPSO [43]	R-I-D-CPSO [44]	FA [45]	**Loop BFA**
x_1	0.75	0.75	0.75	**0.75**
x_2	0.375	0.375	0.375	**0.375**
x_3	38.86010	38.8600099	38.86010	**38.8601036269**
x_4	221.3655	221.365548	221.36547	**221.3654713566**
g_1	-3.22E-9	0	-0.0000000	**-8.30002733e-013**
g_2	-0.00427	-0.004275	-0.0043	**-0.004274611399**
g_3	-0.06867	-0.00019	-0.0134	**-0.0000000184**
g_4	-18.6345	-18.634452	-18.6345	**-18.6345286434**
Најбоље	5,850.383	5,850.38376	5,850.38306	**5,850.38306033**
Средња вредност	Н.П.	6059.7143	5,937.33790	**5,850.38307449**
Најлошије	Н.П.	6059.7143	6,258.96825	**5,850.38347513**
С.Д.	Н.П.	4.625E-12	164.54747	**5.91725E-05**

C-EAPSO (Chaos-enhanced accelerated) оптимизацијаројем честица, R-I-D-CPSO реална-целобројна-дискретна-кодирана оптимизацијаројем честица, FA firefly algorithm – алгоритам свица.

Табела 3. 3. Резултати оптимизације суда под притиском за Случај c)

	MBA [46]	HDE & N-MA [47]	B-IA [48]	**Loop BFA**
x_1	0.7802	0.778186	0.827599	**0.7781764127**
x_2	0.3856	0.384660	0.413794	**0.3846533211**
x_3	40.4292	40.3205	42.703137	**40.3200211815**
x_4	198.4964	199.989	169.965254	**199.9944019471**
g_1	0	Н.П.	-0.003429	**-3.9E-9**
g_2	0	Н.П.	-0.006406	**-3.191E-6**
g_3	-86.3645	Н.П.	-3,897.842439	**-0.0223299162**
g_4	-41.5035	Н.П.	-70.034746	**-40.0055980529**
Најбоље	5,889.3216	5,885.39	6,029.181059	**5,885.3470952**
Средња вредност	6,200.64765	5,885.39	6,136.7807	**6,030.495949**
Најлошије	6,393.5062	Н.П.	6,288.2630	**6,280.464743**
С.Д.	160.34	Н.П.	56.76628	**132.128474**

MBA алгоритам експлозије мине, HDE & N-MA хибридна диференцијална еволуција и Nelder-Mead алгоритам, , B-IA биолошки инспирисан алгоритам заснован на мембранском прорачуну.

За случај d), упоредне врености добијених резултата из цитиране литературе и модификованог алгоритма дате су у Табели 3.4. Као што се може видети, из дате табеле, вредност функције циља као и стандардна девијација добијени цикличним алгоритмом фамилије слепих мишева имају најмање вредности у односу на упоредне резултате преузете из цитиране литературе.

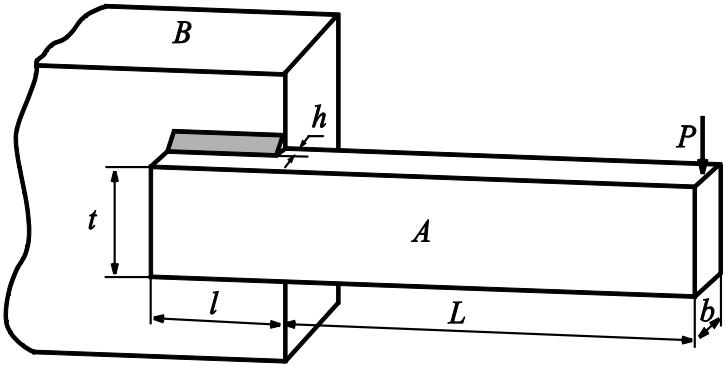
Табела 3. 4. Резултати оптимизације суда под притиском за Случај d)

	MHSO [63]	FSA [49]	MFA [50]	**Loop BFA**
x_1	0.744060	0.768325709391	0.7902	**0.7276421388**
x_2	0.367789	0.379783796302	0.3828	**0.3596787366**
x_3	38.552312	39.80962224818	39.9187	**37.6995280446**
x_4	226.155260	207.2255595186	206.8192	**239.9922765345**
g_1	Н.П.	Н.П.	Н.П.	**-4.12475E-5**
g_2	Н.П.	Н.П.	Н.П.	**-2.52390E-5**
g_3	Н.П.	Н.П.	Н.П.	**-4.0765388480**
g_4	Н.П.	Н.П.	Н.П.	**-7.7234655E-3**
Најбоље	5,829.54746	5,868.764836	6,048.5142	**5,804.79478323**
Средња вредност	Н.П.	6,164.585867	7,255.0734	**5,882.71719642**
Најлошије	Н.П.	6,804.328100	9,010.8501	**5,980.14621424**
С.Д.	Н.П.	257.473670	393.6952	**50.64215283**

MHSO модификона оптимизација хармонијским претраживањем, FSA филтер симулирано каљење, MFA измењени алгоритам свица.

3.6.2. Заварени носач

Основни задатак код оптимизације завареног носача, Слика 3.7., је пројектовање носача за минималне трошкове производње. Ограничења при решавању овог оптимизационог проблема се односе на дозвољени тангенцијални напон τ, дозвољени нормални напон σ, на максимално оптерећење, P_c, као и на дозвољени угиб носача, δ. На Слици 3.7, дате су пројектне промењљиве којима се дефинише функција циља, односно наведена ограничења: h, l, t и b, $\begin{bmatrix} x_1 & x_2 & x_3 & x_4 \end{bmatrix} = \begin{bmatrix} h & l & t & b \end{bmatrix}$.



Слика 3. 7. Приказ завареног носача

Математичка формулација функције циља, [36], представљена је једначинама (3.18) – (3.25) и представља укупне трошкове производње. Ти трошкови се углавном односе на трошкове монтаже, заваривања, рада и материјала.

$$\text{Minimize } f(\mathbf{X}) = 1.10471 x_1^2 x_2 + 0.04811 x_3 x_4 (L + x_2) \tag{3. 18}$$

Функције ограничења:

$$g_1(\mathbf{X}) = \tau(\mathbf{X}) - \tau_{\max} \le 0 \tag{3. 19}$$

$$g_2(\mathbf{X}) = \sigma(\mathbf{X}) - \sigma_{\max} \le 0 \tag{3. 20}$$

$$g_3(\boldsymbol{x}) = x_1 - x_4 \le 0 \tag{3. 21}$$

$$g_4(\boldsymbol{x}) = 0.1047 x_1^2 + 0.04811 x_3 x_4 (L + x_2) - 5 \le 0 \tag{3. 22}$$

$$g_5(\mathbf{X}) = 0.125 - x_1 \le 0 \tag{3. 23}$$

$$g_6(\mathbf{X}) = \delta(\mathbf{X}) - \delta_{\max} \le 0 \tag{3. 24}$$

$$g_7(\mathbf{X}) = P - P_C(\mathbf{X}) \le 0 \tag{3. 25}$$

где је: $\tau(\boldsymbol{x}) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (t'')^2}$, $\tau' = \frac{P}{\sqrt{2}x_1 x_2}$,

$\tau'' = \frac{MR}{J}$, $R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}$,

$J = 2\sqrt{2}x_1 x_2 \left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]$, $M = P\left(L + \frac{x_2}{2}\right)$,

$\sigma(\boldsymbol{x}) = \frac{6PL}{x_4 x_3^2}$, $\delta(\boldsymbol{x}) = \frac{4PL^3}{E x_3^3 x_4}$,

$P_C(\boldsymbol{x}) = \frac{4.013E\sqrt{\frac{x_3^2 x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right)$.

Такође је дато: $P = 6000lb$, $L = 14in$, $\delta_{max} = 0.25in$, $E = 30\times10^6 psi$, $G = 12\times10^6 psi$, $\tau_{max} = 13600psi$, $\sigma = 30000psi$, док су граничне вредности за пројектне променљиве: $0.125 \le x_1 \le 10$, $0.1 \le x_2 \le 10$, $0.1 \le x_3 \le 10$, $0.1 \le x_4 \le 5$.

У овом примеру, параметри алгоритма су: број слепих мишева у фамилији је 30, број фамилија слепих мишева је 50 и број итерација по фамилији је 1000.

Упоредни резултати, добијени модификованим алгоритмом и добијени у цитираној литератури дати су у Табели 3.5. Вредност функције циља добијена Loop BFA, је нижа од вредности добијених применом 5 различитих

оптимизационих алгоритама: BA, S-APA, EHGA, FA и B-IA, док је вредност индетична са резултатитима добијеним применом алгоритама: ABC, TLBO, MBA, IHSA и TIHSA. Најбоља вредност је добијена применом алгоритма PSOLVER, док је најнижа стандардна девијација добијена применом MBA.

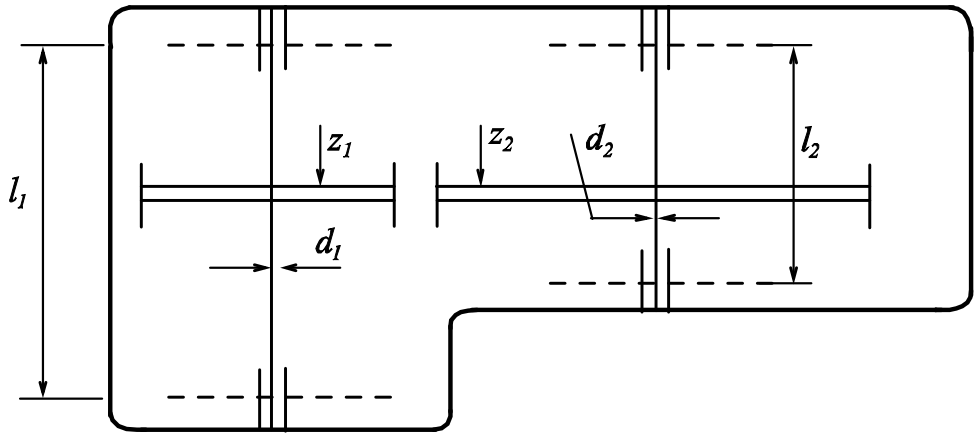
Табела 3. 5. Упоредни резултати оптимизације завареног носача добијени различитим методама

	BA [36]	S-APA [51]	PSOLVER [37]	EHGA [38]	ABC [39]	TLBO [40]
x_1	0.2015	0.2088	0.205830	0.2443689758	0.205730	0.20572963
x_2	3.562	3.4205	3.468338	6.2175197152	3.470489	3.47048834
x_3	9.0414	8.9975	9.036624	8.2914713905	9.036624	9.036625372
x_4	0.2057	0.2100	0.205730	0.2443689758	0.205730	0.205729634
g_1	Н.П.	-0.337812	Н.П.	Н.П.	0.000000	-5.64517E-05
g_2	Н.П.	-353.902604	Н.П.	Н.П.	-0.000002	-0.008826575
g_3	Н.П.	-0.00120	Н.П.	Н.П.	0.000000	-4.1E-09
g_4	Н.П.	-3.411865	Н.П.	Н.П.	-3.432984	-3.432983608
g_5	Н.П.	-0.08380	Н.П.	Н.П.	-0.080730	-0.08072963
g_6	Н.П.	-0.235649	Н.П.	Н.П.	-0.235540	-0.235540329
g_7	Н.П.	-363.232384	Н.П.	Н.П.	0.000000	-1.111156E-04
Најбоље	1.7312065	1.74830941	1.724717	2.38095658	1.724852	1.724852455
Средња вредност	1.8786560	Н.П.	1.724717	2.38095658	1.741913	1.7248561
Најлошије	2.3455793	Н.П.	1.724717	2.38095658	Н.П.	Н.П.
С.Д.	0.2677989	Н.П.	1.62E-11	1.1E-15	3.1E-02	Н.П.
	MBA [46]	IHSA [52]	FA [45]	B-IA [48]	TIHSA [53]	**Loop BFA**
x_1	0.205729	0.20573	0.2015	0.205675	0.20573	**0.2057295391**
x_2	3.470493	3.47049	3.5620	3.470993	3.47049	**3.4704915585**
x_3	9.036625	9.03662	9.0414	9.040587	9.03662	**9.0366228948**
x_4	0.205729	0.20573	0.2057	0.205728	0.20573	**0.2057296919**
g_1	-0.001614	Н.П.	Н.П.	-2.640579	0.00000	**-0.0010452872**
g_2	-0.016911	Н.П.	Н.П.	-26.062354	0.00000	**-0.0008538177**
g_3	-2.4E-07	Н.П.	Н.П.	-0.000053	-5.55E-17	**-1.53E-07**
g_4	-3.432982	Н.П.	Н.П.	-3.432268	-3.4329	**-3.4329833109**
g_5	-0.080729	Н.П.	Н.П.	-0.080675	-0.0807	**-0.0807295391**
g_6	-0.235540	Н.П.	Н.П.	-0.235559	-0.2355	**-0.2355403214**
g_7	-0.001464	Н.П.	Н.П.	-1.588096	-9.09E-13	**-0.0041145717**
Најбоље	1.724853	1.7248	1.7312065	1.725507	1.7248	**1.72485276**
Средња вредност	1.724853	Н.П.	1.8786560	1.726594	Н.П.	**1.72490284**
Најлошије	1.724853	Н.П.	2.3455793	1.728121	Н.П.	**1.72518446**
С.Д.	6.94E-19	Н.П.	0.2677989	7.246E-04	Н.П.	**6.555E-05**

IHSA унапређени алгоритам хармонијског претраживања, TIHSA двапут унапређени алгоритам хармонијског претраживања.

3.6.3. Редуктор

Циљ оптимизације једностепеног редуктора, приказаног на Слици 3.8, је смањење тежине редуктора са ограничењима која се односе на напон савијања зуба, површински напон, трансверзални угиб вратила и напоне у вратилу.



Слика 3. 8. Једностепени редуктор

Пројектне промењљиве за овај пример оптимизације су: ширина зуба b, модул зупчаника m, број зуба зупчаника z, дужина, између лежајева, првог вратила l_1, дужина, између лежајева, другог вратила l_2 и пречници првог d_1 и другог вратила d_2, $[x_1 \;\; x_2 \;\; x_3 \;\; x_4 \;\; x_5 \;\; x_6 \;\; x_7] = [b \;\; m \;\; z \;\; l_1 \;\; l_2 \;\; d_1 \;\; d_2]$. Трећа промењљива, која се односи на број зубаца зупчаника мора бити целобројна вредност, док остале промењљиве могу бити било који ненегативан број.

Једначинама (3.26) до (3.37), [28], дата је математичка формулација оптимизационог проблема једностепеног редуктора:

Minimize

$$f(\mathbf{X}) = 0.7854x_1x_2^2\left(3.3333x_3^2 + 14.9334x_3 - 43.0934\right) - 1.508x_1\left(x_6^2 + x_7^2\right) + 7.477\left(x_6^3 + x_7^3\right) + 0.7854\left(x_4x_6^2 + x_5x_7^2\right) \tag{3. 26}$$

Функције ограничења:

$$g_1(\mathbf{X}) = \frac{27}{x_1x_2^2x_3} - 1 \le 0, \tag{3. 27}$$

$$g_2(\mathbf{X}) = \frac{397.5}{x_1x_2^2x_3^2} - 1 \le 0, \tag{3. 28}$$

$$g_3(\mathbf{X}) = \frac{1.93x_4^3}{x_2x_3x_6^4} - 1 \le 0, \tag{3. 29}$$

$$g_4(\mathbf{X}) = \frac{1.93x_5^3}{x_2x_3x_7^4} - 1 \le 0, \tag{3. 30}$$

$$g_5(\mathbf{X}) = \frac{\left(\left(\frac{745x_4}{x_2x_3}\right)^2 + 16.9\times10^6\right)^{\frac{1}{2}}}{110x_6^3} - 1 \le 0, \tag{3. 31}$$

$$g_6(\mathbf{X}) = \frac{\left(\left(\frac{745x_5}{x_2x_3}\right)^2 + 157.5\times10^6\right)^{\frac{1}{2}}}{85x_7^3} - 1 \le 0, \tag{3. 32}$$

$$g_7(\mathbf{X}) = \frac{x_2x_3}{40} - 1 \le 0, \tag{3. 33}$$

$$g_8(\mathbf{X}) = \frac{5x_2}{x_1} - 1 \le 0, \tag{3. 34}$$

$$g_9(\mathbf{X}) = \frac{x_1}{12x_2} - 1 \le 0, \tag{3. 35}$$

$$g_{10}(\mathbf{X}) = \frac{15x_6 + 1.9}{x_4} - 1 \le 0, \tag{3. 36}$$

$$g_{11}(\mathbf{X}) = \frac{1.1x_7 + 1.9}{x_5} - 1 \le 0, \tag{3. 37}$$

Распон пројектних променљивих је: $2.6 \le x_1 \le 3.6$; $0.7 \le x_2 \le 0.8$; $17 \le x_3 \le 28$; $7.3 \le x_4 \le 8.3$; $0.7 \le x_5 \le 0.8$; $2.9 \le x_6 \le 3.9$; $5.0 \le x_7 \le 5.5$.

Параметри алгоритма за овај пример оптимизације су следећи: број слепих мишева у једној фамилији 30, број фамилија слепих мишева 50 и број итерација по свакој фамилији је 1500.

Резултати добијени различитим оптимизационим алгоритмима представљени су Табелом 3.6. Резултат функције циља и ограничења добијена цикличним алгоритмом фамилије слепих мишева су бољи него вредности добијене применом алгоритама ABC, TLBO, MBA и CS. Овде је важно напоменити да је најнижа вредност функције циља добијена стандардним BA алгоритмом, али да је вредност стандардне девијације знатно нижа код предложеног модификованог алгоритма.

Табела 3. 6. Упоредни резултати оптимизације једностепеног редуктора добијених различитим алгоритмима

	ABC [39]	TLBO [40]	MBA [46]	CS [41]	BA [28]	**Loop BFA**
x_1	3.499999	3.5	3.5	3.5015	3.5	**3.5**
x_2	0.7	0.7	0.7	0.7	0.7	**0.7**
x_3	17	17	17	17	17	**17**
x_4	7.3	7.3	7.300033	7.6050	7.30001	**7.3**
x_5	7.8	7.8	7.715772	7.8181	7.71532	**7.7153199132**
x_6	3.350215	3.350214666	3.350218	3.3520	3.35021	**3.3502146664**
x_7	5.287800	5.28668323	5.286654	5.2875	5.2875	**5.2866544650**
g_1	-0.073915	-0.07391528	Н.П.	Н.П.	-0.074	**-0.0739152804**
g_2	-0.197999	-0.952823443	Н.П.	Н.П.	-0.198	**-0.1979985271**
g_3	-0.499172	-0.499172248	Н.П.	Н.П.	-0.499	**-0.4991722483**
g_4	-0.901555	-0.901471698	Н.П.	Н.П.	-0.905	**-0.9046439045**
g_5	0.000000	-0.00000	Н.П.	Н.П.	0.000	**-2E-10**
g_6	0.000000	-0.00000	Н.П.	Н.П.	0.000	**-2.02E-12**
g_7	-0.702500	-0.7025	Н.П.	Н.П.	-0.703	**-0.7025000000**
g_8	0.000000	-0.00000	Н.П.	Н.П.	0.000	**-5.39E-13**
g_9	-0.583333	-0.583333	Н.П.	Н.П.	-0.583	**-0.5833333333**
g_{10}	-0.051326	-0.051325754	Н.П.	Н.П.	-0.051	**-0.0513257535**
g_{11}	-0.010695	-0.010852365	Н.П.	Н.П.	0.000	**-2E-10**
Најбоље	2,997.058412	2,996.348165	2,994.482453	3,000.9810	2,994.4671	**2,994.47106627**
Средња вредност	2,997.058412	Н.П.	2,996.769019	3,007.1997	2,994.4671	**2,994.471833**
Најлошије	Н.П.	Н.П.	2,999.652444	3,009.00	4,973.8644	**2,994.473622**
С.Д.	0	Н.П.	1.56	4.9634	721.51803	**6.8E-04**

CS cuckoo search – претраживање кукавице.

3.6.4. Конусна опруга

Код оптимизације конусне опруге циљ је минимизирати тежину опруге, Слика 3.9, уз задовољење ограничења датих једначинама од (3.38) до (3.45). Овај проблем има четири пројектне промењљиве: спољни пречник опруге D_e, унутрашњи пречник опруге D_e, дебљину опруге t и висину опруге h, $\left[x_1 \quad x_2 \quad x_3 \quad x_4\right]=\left[\, t \quad h \quad D_e \quad D_l\right]$.

$$\text{Minimize } f(\mathbf{X}) = 0.07075\pi\left(x_4^2 - x_3^2\right)x_1 \tag{3. 38}$$

Функције ограничења:

$$g_1(\mathbf{X}) = S - \frac{4E\delta_{\max}}{\left(1-\mu^2\right)\alpha x_4^2}\left[\beta\left(x_2 - \frac{\delta_{\max}}{2}\right) + \gamma x_1\right] \geq 0, \tag{3. 39}$$

$$g_2(\mathbf{X}) = \left(\frac{4E\delta_{\max}}{(1-\mu^2)\alpha x_4^2} \left[\left(x_2 - \delta\right)\left(x_2 - \frac{\delta}{2} \right) x_1 + x_1^3 \right] \right)_{\delta=\delta_{\max}} - P_{\max} \geq 0, \tag{3. 40}$$

$$g_3(\mathbf{X}) = \delta_1 - \delta_{\max} \geq 0, \tag{3. 41}$$

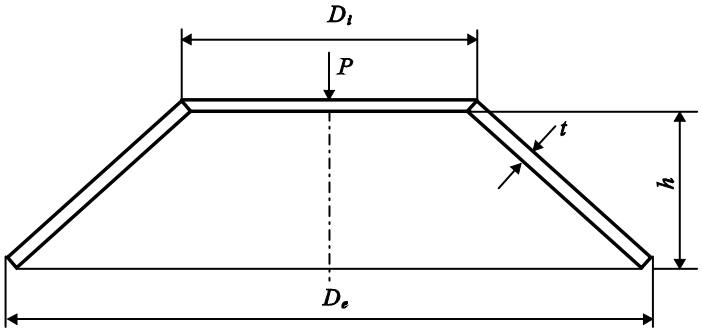
$$g_4(\mathbf{X}) = H - x_2 - x_1 \geq 0, \tag{3. 42}$$

$$g_5(\mathbf{X}) = D_{\max} - x_4 \geq 0, \tag{3. 43}$$

$$g_6(\mathbf{X}) = x_4 - x_3 \geq 0, \tag{3. 44}$$

$$g_7(\mathbf{X}) = 0{,}3 - \frac{x_2}{x_4 - x_3} \geq 0, \tag{3. 45}$$

где је: $\alpha = \frac{6}{\pi \ln K}\left(\frac{K-1}{K}\right)^2$, $\beta = \frac{6}{\pi \ln K}\left(\frac{K-1}{\ln K} - 1\right)$, $\gamma = \frac{6}{\pi \ln K}\left(\frac{K-1}{2}\right)$, $K = \frac{x_4}{x_3}$, $\delta_1 = f(a)a$, $a = \frac{x_2}{x_1}$.



Слика 3. 9. Конусна опруга

Вредности константи су: $P_{max} = 5400$lb, $E = 30 \times 10^6$psi, $\delta_{max} = 0.2$in, $\mu = 0.3$, $S = 200$ KPsi, $H = 2$in, $D_{max} = 12.01$in.

Вредности које се рачунају у зависности од параметра a, дати су у Табели 3.7.

Табела 3. 7. Вредности $f(a)$ у зависности од параметра a

$a \leq$	1.4	1.5	1.6	1.7	1.8	1.9	2.0	2.1	2.2	2.3	2.4	2.5	2.6	2.7	≥2.8
$f(a)$	1	0.85	0.77	0.71	0.66	0.63	0.6	0.58	0.56	0.55	0.53	0.52	0.51	0.51	0.5

Параметри алгоритма, за овај пример су: број слепих мишева у једној фамилији је 30, број фамилија слепих мишева је 50 и број итерација по фамилији је 1000.

Граничне вредности пројектних променљивих су: $0.01 \leq x_1 \leq 0.6$; $0.05 \leq x_2 \leq 0.5$; $5 \leq x_3$; $x_4 \leq 15$.

У Табели 3.8. приказани су најбољи резултати добијени како предложеном модификацијом стандардног BA алгоритма, тако и другим оптимизационим алгоритмима. Вредност функције циља, односно граничне вредности добијене цикличним алгоритмом фамилије слепих мишева је боља од резултата добијених алгоритмима GeneAS, EO и DE. Најнижа вредност функције циља је добијена алгоритмима MBA и TLBO, али је вредност стандардне девијације већа него код предложене модификације стандардног BA алгоритма.

Табела 3. 8. Упоредни резултати оптимизације конусне опруге добијених различитим алгоритмима

	GeneAS [54]	EO [56]	MBA [46]	DE [56]	TLBO [40]	**Loop BFA**
x_1	0.210	0.208	0.204143	0.204262	0.204143	**0.2041433608**
x_2	0.204	0.200	0.2	0.200021	0.2	**0.2000000004**
x_3	9.268	8.751	10.0304732	10.003783	10.03047	**10.0304718573**
x_4	11.4999	11.067	12.01	11.990978	12.01	**12.0099989020**
g_1	1988.370	2145.4109	4.58E-04	159.9270601084	1.77E-06	**0.0011704174**
g_2	197.726	39.75018	3.04E-07	0.1924019464	7.46E-08	**0.0002442483**
g_3	0.004	0.000	9.24E-10	2.1E-05	5.8E-11	**4.0E-10**
g_4	1.586	1.592	1.595856	1.595717	1.595857	**1.5958566388**
g_5	0.511	0.943	0	0.0190222	2.35E-09	**1.09E-06**
g_6	2.230	2.316	1.979526	1.987195	1.979527	**1.9795270447**
g_7	0.208	0.21364	0.198965	0.1993450567	0.198966	**0.1989657651**
Најбоље	2.16256	2.121964	1.9796747	1.984374	1.979675	**1.97967493**
Средња вредност	Н.П.	Н.П.	1.984698	Н.П.	1.97968745	**1.985862417**
Најлошије	Н.П.	Н.П.	2.005431	Н.П.	1.979757	**2.0004935**
С.Д.	Н.П.	Н.П.	7.78E-03	Н.П.	0.45	**7.45E-03**

GeneAS комбиновани генетско прилагодљиво претраживање, EO еволуциона оптимизација, DE диференцијална еволуција.

3.6.5. Ламеласта кочница са више дискова

У овом оптимизационом примеру, циљ је добити кочницу, Слика 3.10, са што мањом тежином. Као што се са поменуте слике види, број пројектних променљивих је 5 и оне се односе на:

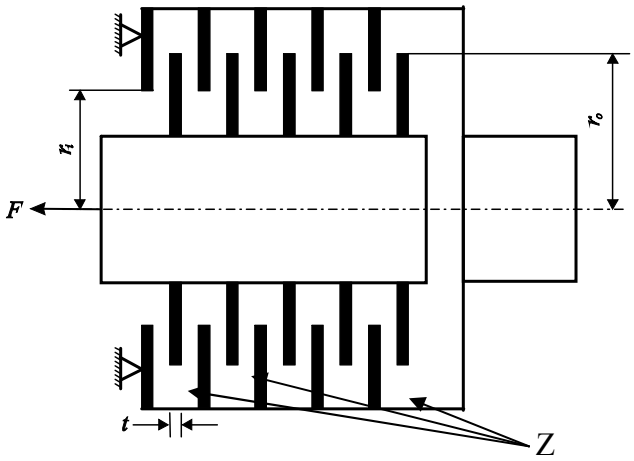
унутрашњи радијус , $r_i \equiv x_1$ (r_i = 60, 61, 62, ..., 80)

спољашњи радијус, $r_o \equiv x_2$ (r_o = 90, 91,92, ..., 110)

дебљина диска, $t \equiv x_3$ (t= 1, 1.5, 2, 2.5, 3)

сила the actuating, $F \equiv x_4$ (F = 600, 610, ..., 1000)

број фрикционих површина, $Z \equiv x_5$ (Z = 2, 3, ..., 9)



Слика 3. 10. Ламеласта кочница са више ламела

Функција циља, представљена једначином (3.46), мора да задовољи ограничења дата једначинама од (3.47) до (3.54): минимална разлика између спољног и унутрашњег пречника, максимална вредност дужине кочења, максимални дозвољени притисак, температура, релативна брзина клизања ламеле, време заустављања, вредност генерисаног момента и позитивно заустављно време.

$$\text{Minimize}: f(\mathbf{X}) = \pi\left(x_2^2 - x_1^2\right)x_3\left(x_5 + 1\right)\rho \tag{3. 46}$$

Функције ограничења:

$$g_1(\mathbf{X}) = x_2 - x_1 - \Delta r \geq 0 \tag{3. 47}$$

$$g_2(\mathbf{X}) = l_{\max} - \left(x_5 + 1\right)\left(x_3 + \delta\right) \geq 0 \tag{3. 48}$$

$$g_3(\mathbf{X}) = p_{\max} - p_{rz} \geq 0 \tag{3. 49}$$

$$g_4(\mathbf{X}) = p_{\max} v_{sr\max} - p_{rz} v_{sr} \geq 0 \tag{3. 50}$$

$$g_5(\mathbf{X}) = v_{sr\max} - v_{sr} \geq 0 \tag{3. 51}$$

$$g_6(\mathbf{X}) = T_{\max} - T \geq 0 \tag{3. 52}$$

$$g_7(\mathbf{X}) = M_h - sM_s \geq 0 \tag{3. 53}$$

$$g_8(\mathbf{X}) = T \geq 0 \tag{3. 54}$$

где се p_{rz}, v_{sr}, T и M_h рачунају на следећи начин:

$$p_{rz} = \frac{x_4}{\pi(x_2^2 - x_1^2)};\ v_{sr} = \frac{2\pi n(x_2^3 - x_1^3)}{90(x_2^2 - x_1^2)};\ T = \frac{I_z \pi n}{30(M_h + M_f)};\ M_h = \frac{2}{3}\mu x_4 x_5 \frac{x_2^3 - x_1^3}{x_2^2 - x_1^2}$$

Вредности параметара датим у претходним једначинама ограничењ су: Δr = *20mm*, минимална разлика између спољашњег и унутрашњег радијуса, δ = *1.5mm* растојање између дискова без оптерећења, максимална дужина l_{max} = *30mm*, максимални број дискова Z_{max} = *10*, максимална брзина of the slip stick v_{srmax} = 10m/s, коефицијент трења μ = *0.5*, степен сигурности *s* = *1.8*, статички момент M_f = *40Nm*, моменат отпора трења M_h = *40Nm*, улазна брзина *n* = *250rpm*,

максимално дозвољени притисак на диску $p_{max} = 1MPa$, масени момент инерције диска $I_z = 55kgm^3$, максимално време заустављања $T_{max} = 15s$.

Параметри алгоритма за овај проблем оптимизације су: број слепих мишева 30, број фамилија 50, и број итерација по фамилији је 1000.

У Табели 3.9., је дато поређење резултата функције циља и ограничења са другим методама. Вредност функције циља добијена применом предложених модификација стандардног BA алгоритма укључених у Loop BFA, је боље од решења добијених применом алгоритма KanGAL. Минимална вредност функције циља добијена предложеним алгоритмом је иста као и вредности добијене применом алгоритма TLBO односно WCA, док је вредност стандардне девијације нижа од вредности добијене применом алгоритма WCA.

Табела 3. 9. Упоредни резултати оптимизације конусне опруге добијених различитим алгоритмима

	KanGAL [57]	TLBO [40]	WCA [58]	**Loop BFA**
x_1	70	70	70	**70**
x_2	90	90	90	**90**
x_3	1.5	1	1	**1**
x_4	1000	810	910	**880**
x_5	3	3	3	**3**
g_1	0.00000	0.00000	0.00000	**0.00000**
g_2	22.00000	24.00000	24.00000	**24.00000**
g_3	0.9005	0.91942781	0.909480	**0.91246**
g_4	9.7906	9,830.371094	9.809429	**9,815.71181**
g_5	7.8947	7,894.69659	7.894696	**7,894.69659**
g_6	3.3527	0.702013203	2.231421	**1.81651**
g_7	60.6250	37,706.25	49.768749	**6,150.00000**
g_8	11.6473	14.2979868	12.768578	**13.18349**
Најбоље	0.4704	0.313657	0.313656	**0.31365661**
Средња вредност	Н.П.	0.3271662	0.313656	**0.31365661**
Најлошије	Н.П.	0.392071	0.313656	**0.31365661**
С.Д.	Н.П.	0.67	1.69E-16	**2.22045E-16**

KanGAL Kanpur genetic algorithms laboratory, WCA Water cycle aglorithm.

3.7. ЗАКЉУЧНЕ НАПОМЕНЕ УЗ ТРЕЋЕ ПОГЛАВЉЕ

У овом поглављу приказана је модификација стандардног BA алгоритма, која се одвијала у три корака. Први корак подразумева начин израчунавања фреквенције за сваког слепог миша у оквиру једне фамилије. Други корак представља увођење фамилије слепих мишева, како би се континуално понављала претрага простора могућих решења у циљу добијања оптималног решења. Следећа модификација иде у правцу финог цикличног претраживања простора решења. За сваког слепог миша, у фамилији, финим претраживањем по кораку Левијевог лета тражи се побољшано решење све док се не задовоље постављена

ограничења. Таква решења се упоређују за сваку фамилију слепих мишева и на крају се бира најбоље. Примењени циклични алгоритам фамилија слепих мишева, тестиран је на 5 бенчмарк инжењерских примера из области примењене механике. У сваком примеру коришћено је 50 фамилија где у свакој фамилији има по 30 јединки. Број итерација је 1000, односно 1500 у неким примерима, по једној фамилији. Вредности функције циља, која је добијена модификацијом стандардног BA, односно применом Loop BFA, у неким примерима има најмању вредност у односу на примењене различите оптимизационе методе или има исту најмању вредност у поређењу са другим методама. Стандардно одступање у свим примерима има такође задовољавајуће малу вредност.

Исправност и ефикасност предложених модификација су верификовани кроз добре резултате добијене тестирањем на пет различитих инжењерских оптимизационих проблема (суд под притиском, заварени носач, редуктор, конусна опруга, и ламеласта кочница са више дискова). Најмања средња и најгора вредност функције циља као и вредност стандардне девијације дате су за сваки од примера. Добијена вредност функције циља је нижа, у поређењу са вредностима у 21 цитираној литератури, исте вредности су добијене као у 15 цитираних референци, а већа вредност је у односу на 4 цитиране референце. Када је у питању стандардна девијација, вредност је нижа у поређењу са 16 наведених извора, мања у поређењу са 9 цитираних резултата. У 15 цитираних референци вредност стандардне девијације није дата.

Поглавље

4

ХИБРИДНИ АЛГОРИТАМ КУКАВИЧЈЕ ПРЕТРАГЕ (Cuckoo search–CS) И АЛГОРИТМА СВИЦА (Firefly algorthm – FA)

4. ХИБРИДИЗАЦИЈА АЛГОРИТАМА

У циљу добијања што бољих резултата и ефектнијих алгоритама, многи аутори су поред модификације стандардних оптимизационих алгоритама, прибегавали и комбиновању два и више различитих оптимизационих алгоритама.

Тако, у [65], представљени алгоритам представља комбинацију Learning Automata шеме и генетског алгоритма у чијој основи је алгоритам свица (Firefly algorithm - FA). Предложени алгоритам има три класе: у првој класи додаје се адаптивни параметара Learning Automata у алгоритма свица (FA), у другој класи додаје се генетски оператор из генетског алгоритма (GA) у FA, и на крају, трећа класа примењује Гаусову дистрибуцију и директно померање за сваког свица у пројектном простору.

A. Kaveh и S. Talatahari у [66], разматрали су оптимизациони алгоритам који је добијен комбиновањем три оптимизациона алгоритма: оптимизација појем честица (Particle swarm optimisation PSO), алгоритам колоније мрава (Ant colony optimization ACO) и алгоритма хармонијске претраге (Harmony search HS). Наиме, аутори су у хибридизован алгоритам PSACO: PSO и ACO, уградили део алгоритма хармонијске претраге (HS). Механизам из алгоритма HS омогућава враћање променљиве у задате границе, јер код PSO алгоритма постоји опасност да у неком тренутку, појединачна честица прекорачи задате границе. HS механизам је ту, да уместо усвајања граничних вредности (најнижа или највиша гранична вредност пројектне променљиве) поново, на случајан начин, генерише вредности пројектних променљивих

M. Kefayat, и остали у [76], комбинују алгоритам колоније мрава (ant colony optimization ACO) и алгоритам вештачке колоније пчела (artificial bee colony ABC) у хибридни алгоритам ACO – ABC, за проналажење оптималне локације и снаге дистрибутивног извора снаге (турбине, гориве ћелије, енергија ветра). Овај алгоритам комбинује стратегију засновану на дискретним (оптимизација локације) и континураним (оптимизација снаге) структурама у циљу достизања предности глобалне и локалне претраге ABC и ACO алгоритма, респективно. Такође, ABC алгоритам се користи за добијање скупа недоминантних решења, који се чувају у спољашњој архиви.

Комбиновање алгоритма хармонијске претраге (HS) са другим алгоритмом, је такође урађено у [67]. Наиме, G. Wang., L. Guo, формирају хибридни алгоритам HS/BA, комбинујући поменути HS са алгоритмом слепог миша (Bat algorithm BA) у циљу повећања разноликости популације слепих мишева, у BA алгоритму, а да би се избегла замка уласка у локални минимум. Наиме, у основни BA алгоритам се уводи оператор постепеног прилагођавања из HS алгоритма у циљу повећавања

конвергенције, што хибридизовани алгоритам чини применљивим за шири опсег проблема, уз задржавање карактеристика BA.

Аутори X. Yuan и остали су такође користили HS алгоритам за хибридизацију. Наиме, они су у [77], HS алгоритам користили да побољшају паралелни оптимизациони алгоритам хаоса (parallel chaos optimization algorithm PCOA), јер способност претраживања PCOA алгоритма, уобичајено, зависи од иницијалних, стартних вредности.

У [74], A. Kaveh и остали, комбинују оптимизациони алгоритам јата ластавица (Swallow Swarm Optimization – SSO) и оптимизациционi алгоритам роја честица (PSO). Пошто је популација честица подељена у подколоније, честице не уче само од честице са глобално најбољим искуством, већ и од најбољих честица из сваке подколоније. У овом алгоритму, укључује се елитизам, тако што се бирају и чувају најбоље честице (тј. глобални и локални лидери) у процесу ажурирања популације, правећи добру равнотежу између глобалног и локалног претраживања, тако да честица – истраживач, има способност учења од сопственог искуства и искуства популације, користећи бесцињну честицу да је додатно прилагоди бољем локалном претраживању.

Модел, који комбинује, имуни генетски алгоритам (Immune genetic algorithm IGA) и векторски контролисан PSO, за решавање двонивовских проблема, предложен је у [75]. Оператори IGA, укључујући укрштање и клонирање се примењују први, следећи правило PSO ажурирања. Способност претраживања IGA, се унапређује лолакном претрагом из PSO.

Хибридизација конвенционалног алгоритма кукавичје претраге, CS, и аритметичког оператора укрштања, HCSA, представљен је од стране M. Balasubbareddy и осталих у [78]. Овај алгоритам, је калибрисан у смислу конвергенције стопе и броја итерација узетих за коначну конвергенцију.

W. Long и остали, у [89] као основни алгоритам за хибридизацију користе алгоритам кукавичје претраге (Cuckoo search – CS). Наиме, CS алгоритам, као и већина алгоритама заснована на популацијама, веома добро идентификује обећавајућу зону простора претраге, али са недостатком финијег претраживања при одређивању минимума. За побољшање финијег претраживања, приликом уласка у обећавајућу зону, аутори CS алгоритму, додају Солис (Solis) и Ветс (Wets) технике локалног претраживања.

У овој овом поглављу за истраживање ефеката хибридизације изабрана су два оптимизаационa алгоритма, алгоритам кукавичје претраге (Cuckoo Search – CS) и алгоритам свица (Firefly – FA).

Изабрани алгоритми имају различити приступ и оптимизациони ток, али им је заједничко што је и код једног и код другог алгоритма, за добијање решења уграђен механизам Левијевог (L´evy) лета [34, 35]. За разлику од претходног поглавља, који је имао за циљ модификацију алгоритма у циљу боље ефикасности и ефективности, овде се разматра могућност инкорпорирања делова оптимизационих алгоритама и истраживање на понашање тока оптимизације у смислу брзе дивергенције решења, смањења стандардног одступања и слично.

4.1. АЛГОРИТАМ КУКАВИЧЈЕ ПРЕТРАГЕ (CUCKOO SEARCH – CS)

4.1.1. Кукавице и њихов посебан начин живота

На земљи постоји преко 10000 врста птица. За ову врсту животиња заједничко је то да све полажу јаја и ниједна птица не рађа младунче, [61]. Велика већина птица гради гнезда, на дрвећу, на земљи, у шупљинама стена, како би положила јаја из којих се излежу птићи. При томе птице инстиктивно граде неупадљива гнезда, због што мање уочљивости, нарочито од стране грабљиваца. Гледајући гнезда различитих врста птица, као што је то рецимо случај код птице ткача (Euplectes franciscanus), не може, а да се не закључи да су гнезда птица дело направљено по комплексним плановима и проверено инжењерским методама. Наравно ни једно од овога није истина, већ је то урођени инстикт да баш тако граде гнезда.

Постоје и друге птице које одскачу од сваке конвенције прављења гнезда и родитељства и прибегавају посебним вештинама у подизању својих породица. То су „птице паразити“, птице која никада не праве сопствена гнезда и уместо тога полажу своја јаја у гнезда других врста, остављајући праве власнике гнезда да брину о њиховим младунцима. Кукавица је најпознатија „птица паразит“,[60]. Њена стратегија укључује тајност, изненађење и брзину. Мајка уклања једно јаје положено од стране мајке домаћина, полаже своје јаје и лети са јајем птице домаћина у свом кљуну. Цели процес траје само 10 секунди. Кукавице освајају гнезда широког спектра разних птица и пажљиво мимиком боја и умножавањем својих сопствених јаја ради уклапања са оним код домаћина. Свака женка кукавице је специјализована за посебну врсту птице домаћина.

Мноштво врста птица учи како да препозна туђе јаје положено у сопствено гнездо и да га избаци ван свог гнезда. Дакле, кукавица константно импровизује мимикрију јаја домаћина, док птице домаћини покушавају да избаце паразитско јаје. Отимање између домаћина и паразита је сродно оружаној бици, свака страна покушава да надјача другу . Инстинкт кукавицу приморава да на основу богатства извора хране (нарочито са инсектима и гусеницама), бира област где ће положити јаје у гнездо домаћина. Са друге стране, младунчад кукавице, које се излегу у

гнезду домаћина, инстиктивно, прихватају навике птица домаћина. Кукавице постоје у широком спектру станишта. Оне насељавају шуме, шумовите пределе, зимзелене шуме тропског појаса, па чак се могу наћи и у неплодним срединама као што су пустиње. Због различитих станишта, кукавице су развиле велику прилагодљивост у циљу искоришћавања пуног потенцијала излегања, без обзира да ли се ради о гнездима птица домаћина у мочварном подручју, шумском подручју или пак у пустињском подручју.

4.1.2. Алгоритам кукавичје претаге, CS

Алгоритам кукавичје претраге (Cuckoo Search CS) је метахеуристички алгоритам, који је биолошки инспирисан понашањем кукавице приликом претраге простора за погодно гнездо у које ће положити своје јаје. Овај алгоритам је предложен од стане Yang и Deb у [61, 62].

Као што је у претходном поглављу речено, кукавице полажу своја јаја у гнезда других птица, при чему птица домаћин у потпуности преузима бригу о птићу кукавице када се оно излегне. Уколико је у могућности, кукавица полаже јаје у гнездо где је то већ раније радила, како би била сигурна да ће се њено јаје пре излећи од јаја птице домаћина. Наравно, уколико није у могућности то да уради, кукавица леже јаје које имитира јаје птице чије је гнездо. Како се птић кукавице прво излегне, оно инстиктивно гура остала јаја ван гнезда, како би остало само и примало сву храну коју птице домаћини доносе. Међутим, уколико птице домаћини схвате да то није њихово јаје или птић, оне уклањају кукавичје јаје или напуштају гнездо.

У намери да се оптимизациони проблем реши, потребно је да вредност проблемских променљивих буде формирана као распоред. У GA (Genetic Algorithm) и PS (Particle Swarm) овај скуп се назива „хромозом“ и „позиција честице“, респективно. Али овде у CS оптимизационом алгоритму овај скуп се назива „станиште“. У d - димензионалном проблему, где је d укупан број пројектних променљивих које се оптимизирају, станиште, димензије $1\times d$, представља тренутну животну позицију кукавице. Овај низ дефинише се на следећи начин: станиште $=\left(x_1,x_2,...,x_d\right)$. Свака од вредности $\left(x_1,x_2,...,x_d\right)$, представља децимални број са покретном тачком (floating point number). Вредност функције циља, f, на станишту: $\left(x_1,x_2,...,x_d\right)$, се означава као $f\left(x\right)=f\left(x_1,x_2,...,x_d\right)$.

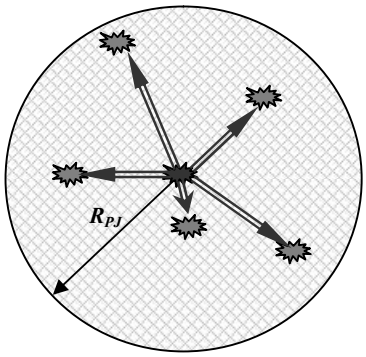
Да би се започео оптимизациони алгоритам, потребно је генерисати матрицу станишта величине $n\times d$, где је n број кукавица, [80, 87, 88]. Након тога насумично произведен број јаја је потребан за сваку од ових кукавица геерисаних у станишту. У природи свака кукавица полаже од 5 до 20 јаја. Још један обичај кукавица је да оне полажу јаја у оквиру максималног растојања од њихових

станишта – радијус полагања јаја R_{PJ}. У оптимизационом проблему са горњом границом *GG* и доњом границом *DG*, за пројектне променљиве, свака кукавица има радијус полагања јаја који пропорционалан укупном броју јаја, S_{ue}, броју актуелних јаја кукавица, S_{ae}, и који се налази у границама променљивих *DG* и *GG*, односно R_{PJ} се може дефинисати као што је дато у једначини (4.1).

$$R_{PJ} = \alpha \times \frac{S_{ae}}{S_{ue}} \times (GG - DG), \tag{4. 1}$$

где је α, цео случајни број којим се дефинише максимална вредност R_{PJ}.

Свака кукавица полаже јаја насумично у гнезда других птица унутар њеног R_{PJ}, Слика 4.1.



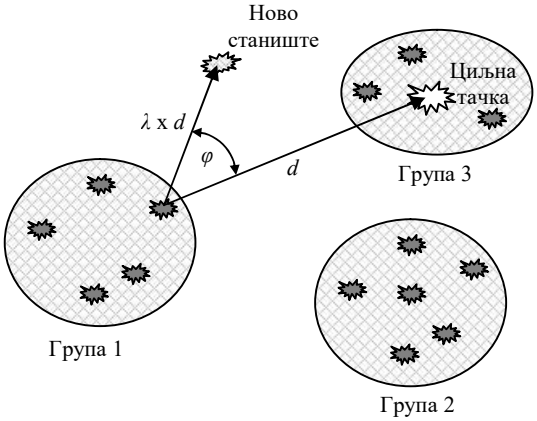
Слика 4. 1. Насумично полагање јаја у R_{PJ} - радијус полагања јаја, централни облак је иницијално станиште кукавице са 5 јаја, остали облаци су јаја у новим гнездима

Након што су сва кукавичја јаја положена у гнезда домаћина, нека од њих која су слична јајима домаћина су откривена од стране домаћина и избачена из гнезда. Дакле после процеса полагања јаја, $p_a\%$ од свих јаја (обично 10%), са мање добитним вредностима, биће уништена. Ова јаја немају шансе да одрасту. Остала јаја одрастају у туђем гнезду, скривена и храњена од стране птице домаћина. Још једна занимљива ствар везана за полагање кукавичјих јаја је та да само једно јаје у гнезду има шанси да одрасте. Ово је због тога када се кукавичје јаје излегне и испили, она избацује јаја птице домаћина из гнезда. У случају да се јаја птице домаћина излегну пре, а јаја кукавице касније, младунче кукавице поједе већину хране коју птица домаћин донесе у гнездо (због 3 пута већег тела, она одгурне осталу младунчад и поједе више). После неколико дана младунчад птица домаћина умиру од глади и једино младунче кукавице преостаје у гнезду.

Када младе кукавице порасту и постану одрасле, оне живе у њиховим сопственим територијама и групама за одређени период. Али када дође време за

полагање јаја, оне емигрирају у нова боља станишта са више сличности јајима домаћих птица и такође са више хране за нове младунце. Након што су формиране групе кукавица на различитим територијама, група (друштво) са најбољом вредношћу функције циља, је изабрана као циљна тачка за имиграцију осталих птица. Када одрасле кукавице живе широм окружења, тешко је препознати која кукавица припада којој групи. Да би решили овај проблем груписање кукавица се врши k-знаком методе груписања. Онда максимална вредност ових просечних вредности функције циља, одређује циљну групу и самим тим то најбоље станиште је ново одредиште за имигрционе кукавице.

Приликом премештања циљне групе, кукавице не лете цели пут до планираног станишта. Оне лете само део пута и такође имају одступање. Ово померање дато је на слици 4.2. Свака кукавица лети само $\lambda[\%]$ од свих дистанци према циљном станишту и такође имају одступање од $\varphi[\%]$ радијана. Ова два параметра λ и φ помажу кукавицама да претраже више позиција у свим окружењима. За сваку кукавицу, λ и φ су дефинисани као: $\lambda \in U(0,1); \varphi \in (-\omega, +\omega)$, ω је параметар који ограничава одступање од циљног станишта.



Слика 4. 2. Имиграција узорка кукавица према циљном станишту

Према чињеници да постоји равнотежа у популацији птица тако број $N_{\max}$ контролише и ограничава максимални број живих кукавица у окружењу. Овај баланс је због ограничених извора хране, опасности од предатора и неспособности да нађу погодно гнездо за своја јаја.

У овом оптимизационом алгоритму свако гнездо представља могуће решење. Процес репродукције кукавица у овом алгоритму се поједностављује са три правила [61]:

1. Свака кукавица полаже јаје у случајно изабрано гнездо;
2. Најбоље гнездо је носилац следеће генерације кукавица;

3. Број доступних гнезда за полагање јаја је фиксан (ограничен) и вероватноћа откривања кукавичјег јаја од стране домаћина је дата вероватноћом $p_a \in [0,1]$. Птице, домаћини, могу да открију најлошије гнездо које се, по откривању, брише из популације.

На основу ових правила и идеализације понашања кукавице, формиран је стандардни CS алгоритам. Стандардни CS алгоритам са основним корацима, сажето је представљен кроз псеудо код дат у алгоритму 4.1.

Алгоритам 4. 1. Псеудо код CS алгоритма

1: Функција циља $f(\mathbf{X}), \mathbf{X} = (x_1, x_2, ..., x_d)^T$
2: Иницијализација популације од *n* гнезда x_i *(i=1,2,…n)*
3: **while** *(t < Max броја итерација*)
4: Израчунавање квалитета избора кукавице Левијевим летом, F_i
5: $f_{new} = \min(\mathbf{F})$;
6: Случајан избор гнезда међу *n* генерисаних (*j-то* гнездо)
7: **if (**$F_i > F_j$)
8: Замени *j-то* гнездо са новим решењем
9: **end if**
10: Израчунавање вероватноће откривања гнезда, $p_a = f(p_a, t)$
11: **if (***rand* >= p_a)
12: Напуштање лошег гнезда и увођење новог
13: Израчунавање квалитета избора кукавице Левијевим летом, F_i
14: $f_{new} = \min(\mathbf{F})$;
15: **end if**
16: **if (**$f_{new} < f_{min}$*)*
17: Прихватање новог решења, $f_{min} = f_{new}$
18: **end if**
19: Рангирање решења и проналажење тренутно најбољег
20: **end while**
21: Постпроцесирање и приказивање резултата

Када се генерише ново решење $x(t+1)$, примењује се Левијев лет (L´evy flight), једначина (2.9) за *i*-ту кукавицу, са кораком, $\alpha > 0$, чија се вредност узима од опсега проблема који се оптимизује. У већини случајева усваја се $\alpha = 1$, мада може да се и рачуна, као што је приказано у поглављу 2, једначина (2.10). Поменута једначина (2.9) је у суштини стохастичка једначина за случајни корак. У суштини, случајни корак представља Марковљев низ чија следећа локација/статус зависе једино од тренутне локације $X_{i,old}$, и вероватноће транзиције $\alpha \otimes \text{Lévy}(\beta)$.

Левијев лет обезбеђује случајни корак, док се величина случајног корака усваја из Левијеве дистрибуције), [34, 35],, једначина (4.2).

$$L'evy \sim u = t - \lambda, (1 < \lambda \leq 3), \tag{4. 2}$$

На овај начин је могуће добити нова решења у околини најбољег решења врло брзо, што доводи до убрзања локалне претраге. Међутим, значајан део нових решења треба да се генерише у широком пољу рандомизације и чије локације би требало да буде довољно далеко од тренутно најбољег решења, што обезбеђује да систем не упадне у замку локалног минимума.

4.2. АЛГОРИТАМ СВИЦА (FIREFLY ALGORTHM –FA)

4.2.1. Понашање свитаца

Свитац или светлећа буба (Lampyris Noctiluca) је крилати инсект који сјаји жуто, зелено или црвено захваљујући хемијски произведеној светлости (биолуминисценција) коју емитује из стомака [59]. У природи постоји око 2000 врста свитаца и већина од њих производи кратко и ритмичко треперење светлости. Шаблон треперења је јединствен за сваку од врста.

Постоји више разлога што свитац светли. Пре свега да би привукао партнера за парење. У оквиру различитих врста постоје и различити шаблони којим свици светле. Најчешће мужјак покушава да импресионира женку својим светлом правећи одређене путање у ваздуху док женка обично стрпљиво чека на грани или у трави. Када је партнер заинтересује она такође отпочиње са емитовањем светла. Други разлог је што својим светлом одбијају предаторе који их лове. Предатори знају да су свици непријатног укуса те ће их избегавати када виде ове њихове сигнале.

4.2.2. Алгоритам свица

Алгоритам свица (FA) први је увео X.S. Yang, у [73]. За формирање алгоритма свица (FA), потребно је идеализовати неке карактеристике трепераве светлости свица. При томе користе се три идеализована правила, [69]:

1. сви свисци су унисексуални, тако да привлаче једни друге без обзира на пол;
2. привлачност је пропорционална сјају треперења, тако да ће свитац са мање сјајном светлошћу кренути у правцу свица који трепери светлост са већим интезитетом. Привлачност и интезитет треперења светлости (сјај) се смањују како се растојање између свитаца повећава. Уколико у близини појединачног свица нема јачег интезитета треперења светлости, он ће се кретати насумице;

3. Интезитет треперење светлости свица је под утицајем или се одређује у зависности од простора дефининсаног функцијом циља. Код проблема тражења максимума, интезитет светлости једноставно се може посматрати као пропорционална вредност функције циља.

На основу, претходно наведена, три правила основни кораци алгоритма свица могу се сумирати у облику псеудо кода, приказаном у алгоритму 4.2.

Алгоритам 4. 2. Алгоритам свица

```
1: begin
2:    Функција циља f(X), X = (x_1, x_2, ..., x_d)^T
3:    Дефинисање укупног броја свитаца у популацији n
4:    Генерисање иницијалне популације свитаца x_i (i = 1,...,n)
5:    Дефинисање броја променљивих d
6:    Интезитет светлости I_i на x_i се одређује помоћу f(x_i)
7:    Дефинисање коефицијента абсорпције светлости γ
8:    Дефинисање броја понављања BP
9:    while (t < BP)
10:         for i=1:n  %%свих n свитаца
11:           for j=1:n  %%свих n свитаца
12:              if (I_j>I_i)
13:                 Померање свица i у правцу j у d-димензионалном простору
14:                 Варирање привлачности са растојањем r преко exp[-γr]
15:                 Израчунавање новог решења и ажурирање јачине светлости
16:              end if
17:           end for j
18:         end for i
19:         Рангирање свитаца и проналажење тренутно најбољег решења
20:   end while
21:   Пострпоцесирање и приказивање резултата
```

У одређеном смислу, може се уочити одређена концептуална сличност између FA и алгоритма потраге за храном бактерија (BFA) [18]. У BFA, привлачност између бактерија зависи делимично од вредности функције циља, а делимично од растојања између њих, док је у FA, привлачност повезана са вредношћу функције циља и монотоно опада са променом растојања. Међутим, механизам уграђен у FA има прилагодљиву видљивост и већу свестраност у варијацији привлачности, што, уобичајено доводи до веће мобилности, чиме се добија ефикасније истраживање простора претраге, [69].

4.2.3. Интезитет треперења светлости, привлачност, померање

У FA агоритму, од посебног значаја су: варијација интезитета светлости и формулација привлачности. Због једноставности, увек се може претпоставити да је привлачност појединачног свица одређена јачином интезитета треперења светлости, која је, заузврат, повезана са вредношћу функције циља, [73, 34, 35].

У поједностављеном случају, код проблема тражења максимума, јачина треперења светлости, *I*, свица, на одређеној локацији x, може бити изабрана као $I(x) \propto f(x)$. Међутим, привлачност, β, је релативна, и зависи од утиска посматрача, или од процене других свитаца. Дакле, варираће са растојањем, $r_{i,j}$, између *i*-тог и *j*-тог свица. Поред тога, интезитет треперења светлости опада са повећањем растојања од његовог извора, и светлост се апсорбује, тако да би требало да се омогући, да се привлачност мења са степеном апсорбције. У најједноставнијем облику, интезитет треперења светлости $I(r)$, се мења обрнуто пропорционално квадрату растојања од извора, $I(r) = I_s / r^2$, где је, I_s, интезитет јачине треперења светлости извора. За дату средину са фиксним коефицијентом апсорбције светлости, γ, интезитет треперења светлости, *I*, се мења са променом растојања, *r*, једначина (4.3).

$$I(r) = I_0 e^{-\gamma r} \quad , \tag{4. 3}$$

где је I_0 иницијални интезитет треперења светлости.

У циљу обезбеђења сингуларитета, при $r = 0$, у изразу I_s / r^2, комбиновани ефекат и обрнуте пропорционалности квадрата растојања и апсорпције може се апроксимирати једначином (4.4).

$$I(r) = I_0 e^{-\gamma r^2} \tag{4. 4}$$

Како је привлачност свица пропорционална интезитету треперења светлости, привлачност свица, β, се може дефинисати као што је дато једначином (4.5).

$$\beta(r) = \beta_0 e^{-\gamma r^2}, \text{ или}$$
$$\beta = \frac{\beta_0}{1 + r^2} \tag{4. 5}$$

где је β_0 привлачност при *r* = 0.

Једначина (4.5), дефинише карактеристично растојање $\Gamma = 1/\sqrt{\gamma}$, преко чије вредности се привлачност драстично мења од β_0 до $\beta_0 e^{-1}$, односно $\beta_0 / 2$, у зависности од тога који је облик једначине (4.5) примењен.

Растојање између било која два свица i и j, при x_i и x_j, респективно, представља растојање у Декартовим координатама, једначина (4.6).

$$r_{ij} = \left\| x_i - x_j \right\| = \sqrt{\sum_{k=1}^{d} \left(x_{i,k} - x_{j,k} \right)^2} \,, \tag{4. 6}$$

где је $x_{i,k}$ *k-та* компонента просторне координате x_i *i-тог* свица. У 2-D случају, то је: $r_{ij} = \sqrt{\left(x_i - x_j \right)^2 + \left(y_i - y_j \right)^2}$.

Померање *i-тог* свица, у смеру привлачнијег (сјајнијег), *j-тог*, свица је одређено једначином (4.7).

$$x_i = x_i + \beta_0 e^{-\gamma r_{ij}^2} \left(x_i - x_j \right) + \alpha \left(rand - \frac{1}{2} \right) \tag{4. 7}$$

где други сабирак представља утицај привлачности, а трећи представља рандомизацију, са параметром *α*. *rand* је случајни број генерисан по равномерној расподели у опсегу [0, 1].

Параметар γ карактерише варијацију привлачности јер његова вредност је веома важна у одређивању брзине конвергенције и одређује како ће се FA алгоритам понашати. У теорији $\gamma \in [0, \infty]$, али у пракси, $\gamma = O(1)$ је одређена карактеристичном дужином, Г, система који се оптимизује. Тако, у већини примена, овај параметар варира од 0.1 до 10.

4.3. ХИБРИДНИ АЛГОРИТАМ КУКАВИЧЈЕ ПРЕТРАГЕ И АЛГОРИТМА СВИЦА - H-CS-FA

Хибридни алгоритам који је примењен у овој дисертацији подразумева комбинацију анализираних оптимизационих алгоритама (алгоритам кукавичје претраге CS и алгоритам свица FA). И код једног и код другог алгоритма, генерисање нових решења се одвија по механизму Левијевог лета. Основна разлика у ова два алгоритма је та што код CS алгоритма, постоји механизам одбацивања „лошег“ гнезда, линије 11-15, алгоритам (4.1), што представља опасност да алгоритам у неком тренутку напусти добар правац у тражењу оптималног решења, што пак може да доведе до повећања времена проналаска оптимума.

Са друге стране у FA алгоритму нема механизма за одбацивање „лоших“ свитаца. Алгоритам је конципиран да у току итерације непрекидно сортира решења, од најмањег ка највећем, линије 12-16, алгоритам (4.2). Оваквим начином постоји опасност да се упадне у зону локалних оптимума, што опет продужава процес тражења најбољег решења.

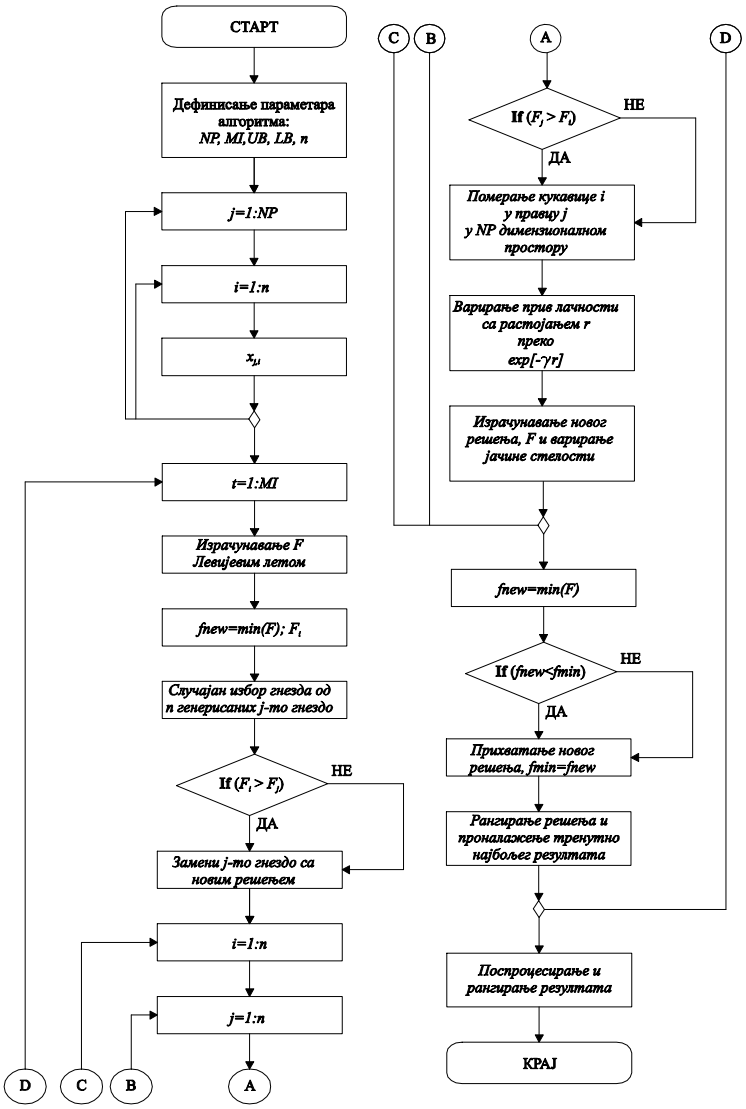
Предложени хибридни алгоритам за основу има CS алгоритам у који је инкорпориран део FA алгоритма, што је приказано у алгоритму 4.3 и на слици 4.3. Наиме, уместо да се испразне гнезда када се достигне вероватноћа p_a проналажења „лоших“ гнезда, у алгоритам је додат део FA алгоритма у коме се проналази свитац који са највећим интезитетом светлости.

Алгоритам 4. 3. Псеудо код H-CS-FA алгоритма

```
1:  Функција циља F (X), X=(x1, x2, ..., xNP)^T
2:  Дефинисање броја параметара који се оптимизираји NP
3:  Дефинисање броја итерација MI (i = 1,...,MI)
4:  Дефинисање граница пројектних променљивих %%горња UB и доња
    граница LB
5:  Дефинисање броја гнезда, n
6:  Иницијализација популације од n гнезда x_{j,i} (i=1,2,...n), j=1,2,..,NP
7:  while (t < MI)
8:      Израчунавање квалитета избора кукавице Левијевим летом, F_i
9:      f_new = min(F)
10:     Случајан избор гнезда међу n генерисаних (j-то гнездо)
11:     %% уметнут део FA
12:     if (F_i > F_j)
13:         Замени j-то гнездо са новим решењем
14:     end if
15:     for i=1:n %%свих n гнезда
16:         for j=1:n %%свих n гнезда
17:             if (F_j>F_i)
18:                 Померање кукавице i у правцу j у NP-димензионалном
                    простору
19:                 Варирање привлачности са растојањем r преко exp[-γr]
20:                 Израчунавање новог решења и ажурирање јачине светлости
21:             end if
22:         end for j
23:     end for i
24:     f_new = min(F)
25:     if (f_new< f_min)
26:         Прихватање новог решења, f_min= f_new
27:     end if
28:     Рангирање решења и проналажење тренутно најбољег резултата
29: end while
30: Постпроцесирање и приказивање резултата
```

Усвојени концепт хибридног алгоритма тестиран је на пет оптимизационих модела: опруге, конусног квачила, I профила, мењача и носача са три променљиве које се оптимизирају. Модели I профила и мењача спадају у групу вишекритеријумске оптимизације (multiobjective optimization - MOO).

МОО се користи за решавање оптимизационих проблема који укључују две или више комплексних, нелинеарних, супростављених функција циља, које треба да се оптимизују истовремено: $\min F(x) = \min\{f_1(x), f_2(x), ..., f_m(x)\}$, где је x вектор пројектних променљивих: $x = (x_1, x_2, ..., x_{NP})$.Код једнокритеријумске оптимизације, оптимизације са једном функцијом циља, оптимизациони алгоритам, генерише ново решење тражећи га у простору претраге дефинисаном границама пројектних променљивих. Током итеративног процеса, лошија решења се одбацују и замењују бољим. Пронађено решење је за тај итеративни циклус и најбоље за дати оптимизациони проблем. Међутим, код нетривијалних МОО проблема, не постоји једнинствено решење које ће у исто време да оптимизује сваку од функција циља. Пронађена оптимална решења не могу бити побољшана по једном критеријуму, а да не буду погоршана по неком другом, што у многоме отежава претрагу и проналажење оптималног решења по свим критеријумима.



Слика 4. 3. Дијаграм тока хибридног алгоритма H-CS-FA

Да би се превазишао овај проблем, из матрице решења $n \times NP$; n-број гнезда, NP-број пројектних променљивих, бира се само оно решење код којих је испуњено да су сви елементи новогенерисаног реда матрице решења мањи од одговарајућих елемената до тада најбољег решења, осенчени редови алгоритма 4.4.

Алгоритам 4. 4. Псеудо код H-CS-FA алгоритма код MOO

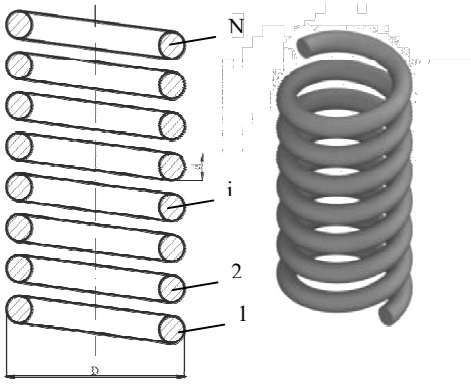
1: Функција циља F (**X**), **X**=$(x_1, x_2, ..., x_{NP})^T$
2: Дефинисање броја параметара који се оптимизирају NP
3: Дефинисање броја итерација MI $(i = 1,...,MI)$
4: Дефинисање граница пројектних променљивих %%горња UB и доња граница LB
5: Дефинисање броја гнезда, n
6: Иницијализација популације од n гнезда $x_{j,i}$ $(i=1,2,...n)$, $j=1,2,..,NP$
7: **while** $(t < MI)$
8: Израчунавање квалитета избора кукавице Левијевим летом, F_i
9: $f_{new} = \min(\mathbf{F}),\ f_{new} = f_{new1}, f_{new2}, ..., f_{newNP}$
10: Случајан избор гнезда међу n генерисаних (*j-то* гнездо)
11: %% *уметнут део FA*
12: **if** $\left(f_{newi1} > f_{newj1} \wedge f_{newi2} > f_{newj2} \wedge ... \wedge f_{newiNP} > f_{newjNP}\right)$
13: Замени *j-то* гнездо са новим решењем
14: **end if**
15: **for** i=1:n %%*свих n гнезда*
16: **for** j=1:n %%*свих n гнезда*
17: **if** $\left(f_{j_1} > f_{i1} \wedge f_{j2} > f_{i2} \wedge ... \wedge f_{jNP} > f_{iNP}\right)$
18: *Померање кукавице i у правцу j у NP-димензионалном простору*
19: *Варирање привлачности са растојањем r преко exp[-γr]*
20: *Израчунавање новог решења и ажурирање јачине светлости*
21: **end if**
22: **end for** *j*
23: **end for** *i*
24: $f_{new} = \min(\mathbf{F}),\ f_{new} = f_{new1}, f_{new2}, ..., f_{newNP}$
25: **if** $\left(f_{new1} \le f_{\min 1} \wedge f_{new2} \le f_{\min 2} \wedge ... \wedge f_{\min NP} \le f_{newNP}\right)$
26: Прихватање новог решења, $\left(f_{\min 1} = f_{new1}, f_{\min 2} = f_{new2}, ..., f_{\min NP} = f_{newNP}\right)$
27: **end if**
28: Рангирање решења и проналажење тренутно најбољег резултата
29: **end while**
30: Постпроцесирање и приказивање резултата

4.4. ОПТИМИЗАЦИОНИ МОДЕЛИ ИЗ ОБЛАСТИ ПРИМЕЊЕНЕ МЕХАНИКЕ

4.4.1. Модел опруге

Овај проблем је описао Arora [80] и Belegundu [81], а састоји се од минимизирања тежине затегнуте или сабијене опруге, Слика 4.4., које имају ограничења везана за минимални угиб, трење, фреквенције угиба, ограничења спољег пречника и пројектних променљивих, пре чему су пројектне променљиве: пречник опруге *D*, пречник жице *d*, број активних намотаја *N*, $[x_1 \quad x_2 \quad x_3] = [d \quad D \quad N]$.

Математичка формулација овог проблема је описана једначинама (4.8) – (4.12).



Слика 4. 4. Шематски приказ опруге са пројектним променљивама

$$\text{Минимизирати } f(\mathbf{X}) = (x_3 + 2)x_2 x_1^2 \tag{4. 8}$$

$$\text{Ограничења } g_1(\mathbf{X}) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \le 0 \tag{4. 9}$$

$$g_2(\mathbf{X}) = \frac{4x_2^2 - x_2 x_1}{12566\left(x_2 x_1^3 - x_1^4\right)} + \frac{1}{5108 x_1^2} - 1 \le 0 \tag{4. 10}$$

$$g_3(\mathbf{X}) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \le 0 \tag{4. 11}$$

$$g_4(\mathbf{X}) = \frac{x_1 + x_2}{1.5} - 1 \le 0 \tag{4. 12}$$

Где је $\mathbf{X} = (x_1, x_2, x_3)$, при чему су ограничења пројектних променљивих: $0.05 \le x_1 \le 2$, $0.25 \le x_2 \le 1.3$, $2 \le x_3 \le 15$.

4.4.1.1. Резултати оптимизације опруге

Упоредни резултати примене CS, FA и H-CS-FA алгоритма, приказани су у Табели 4.1, а поређење резултата добијених применом H-CS-FA алгоритма, са другим оптимизационим алгоритмима, приказани су у Табели 4.2. На Сликама 4.5-4.7 , приказано је генерисање резултата током итеративног процеса.

Параметри алгоритма, за сва три случаја су били исти: број гнезда, односно свитаца $n = 55$, максималан број итерација $MI = 1000$.

Табела 4. 1. Упоредни резултати за оптимизацију опруге FA, CS, **H-CS-FA**.

	FA	CS	**H-CS-FA**
x_1	0.054393268016	0.051853127460	**0.0519552018**
x_2	0.424405194347	0.360666961732	**0.363151422**
x_3	8.599906562289	11.064343776340	**10.921685894**
g_1	-0.046216318389	-0.000256123019	**-0.0043100382**
g_2	-0.001799796129	-0.000023755605	**-0.0055615792**
g_3	-3.931871836250	-4.060090859947	**-4.0187447769**
g_4	-0.680801025092	-0.724986607205	**-0.7216682829**
Најбоље	0.013309846067	0.012669044810	**0.0126667443**
Средња вредност	0.013415128585	0.0128645217324	**0.01310808975**
Најлошије	0.018539299327	0.019047374978	**0.0205215144**
С.Д.	0.000378173921	0.000054314649	**0.0005174367**

CS – Алгоритам кукавичје претраге, FA – алгоритам свица, **CS-FA** – хибридни алгоритам

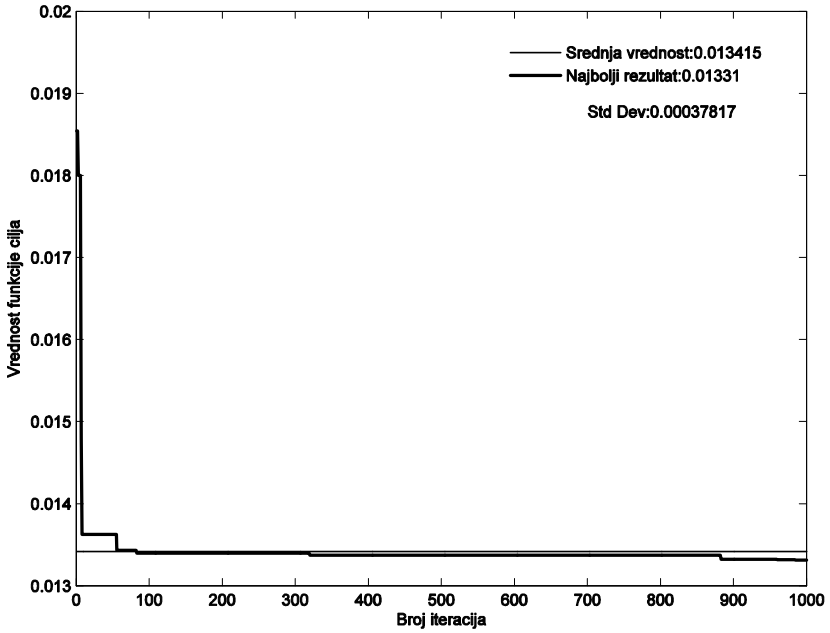
Упоређујући резултате добијене основним алгоритмима CS и FA, са H-CS-FA алгоритмом, Табела 4.1, примећује се да је најбољи резултат добијен применом H-CS-FA алгоритма. Најмања вредност SD, се добија применом FA али је резултат најлошији. Применом сва три алгоритма, испуњена су ограничења, $g_1 \le 0, g_2 \le 0, g_3 \le 0, g_4 \le 0.$

Табела 4. 2. Упоредни резултати оптимизације опруге

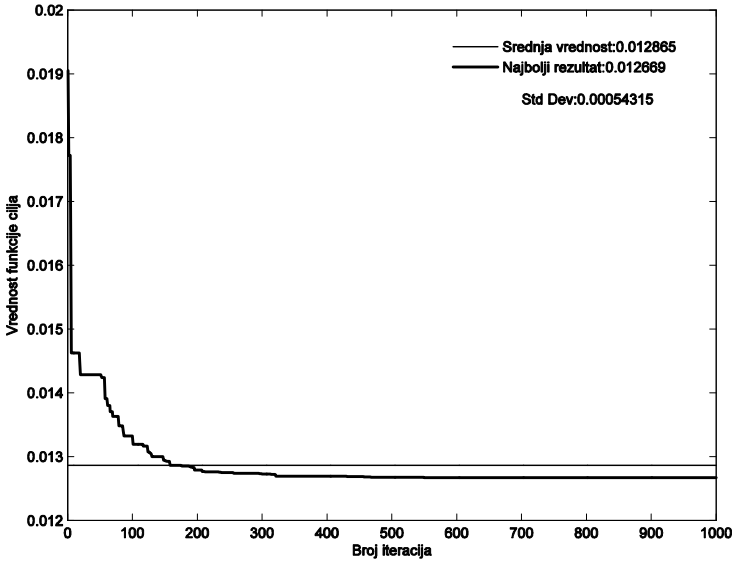
	Mahdavi и други (IHS) [83]	Lobato и Valder (FSO) [84]	Zou и други (DSO) [85]	Mun и Cho (MHS) [63]	Bulatović и други (ICS) [82]	**H-CS-FA**
x_1	0.05115438	0.051744	0.051711791	0.05171296	0.050000	**0.0519552018**
x_2	0.34987116	0.357754	0.357264808	0.35729285	0.489169	**0.363151422**
x_3	12.0764321	11.56132	11.25696483	11.2553284	3.832967	**10.921685894**
g_1	-0.0521995	-0.028697	-5.1563e-009	Н/Д	0.000000	**-0.0043100382**
g_2	0.0136707	-0.000645	-2.8087e-010	Н/Д	0.430373	**-0.0055615792**
g_3	-3.8601496	-3.911391	-4.0549	Н/Д	-6.656629	**-4.0187447769**
g_4	-0.7326496	-0.727001	-0.7273	Н/Д	-0.960830	**-0.7216682829**
Најбоље	0.0128874	0.012789	0.12665	0.0126652	0.0071332	**0.0126667443**

IHS – унапређени алгоритам хармонијске претраге (improved harmony serch algorithm), FSO – оптимизациони алгоритам јата риба (Fish Swarm Optimization Algorithm), DSO – оптимизациони алгоритам директне претраге (Directed searching optimization algorithm), MHS – модификовани алгоритам хармонијске претраге (Modified harmony search), ICS – унапређени алгоритам кукавичје претраге (Improved Cuckoo Search algorthm).

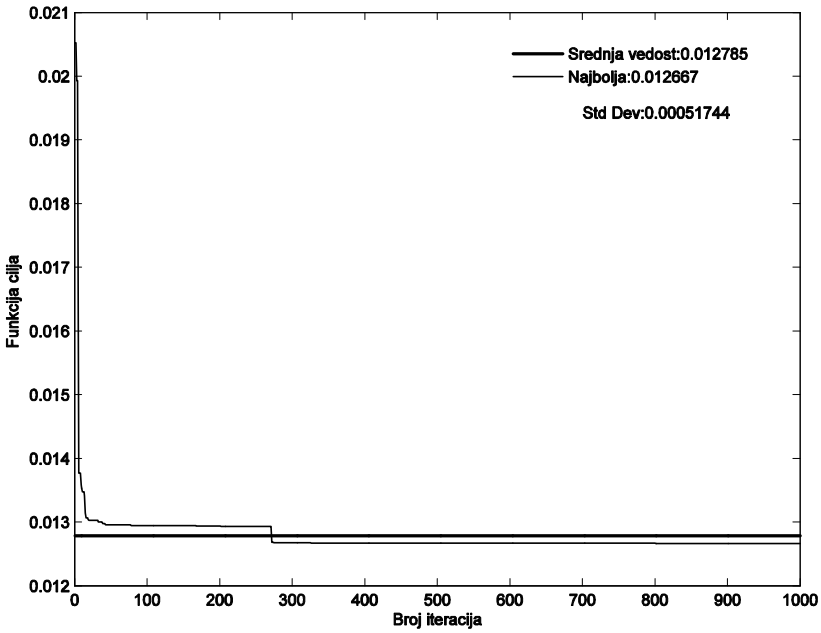
Са друге стране, упоређујући H-CS-FA са резултатима добијених применом других алгоритама, Табела 4.2, може се видети да је најбољи резултат добијен применом ICS. Међутим, ограничење g_2, није задовољено. MHS, алгоритам је дао бољи резултат, али нису наведени гранични услови. H-CS-FA даје боље резултате од преостала три наведена алгоритма: IHS, FSO, DSO.



Слика 4. 5. Најбоља и средња вредност функције циља при оптимизацији опруге применом FA алгоритма



Слика 4. 6. Најбоља и средња вредност функције циља при оптимизацији опруге применом CS алгоритма



Слика 4. 7. Најбоља и средња вредност функције циља при оптимизацији опруге применом H-CS-FA алгоритма

4.4.2. Модел конусног квачила

Овај пример је преузет из [68]. Циљ је наћи минималну запремину конусног квачила, Слика 4.8., како би оно преносило минимални обртни момент. Спољашњи и унутрашњи радијус квачила, $R1 \equiv x_1$ и $R_2 \equiv x_2$ су пројектне променљиве, а функција циља је дата једначином (4.13).

$$f(R_1, R_2) = \frac{1}{3}\pi h\left(R_1^2 + R_1 R_2 + R_2^2\right), \tag{4. 13}$$

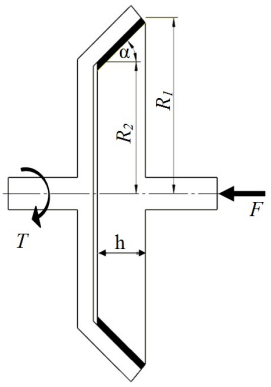
где је:

h – аксијална дебљина, $h = \dfrac{R_1 - R_2}{tg\ \alpha}$, (4. 14)

α – половина угла конуса

Када једначину (4.14), уврстимо у (4.13) добије се функција циља приказана једначином (4.15).

$$f(R_1, R_2) = \frac{1}{3}\pi \frac{R_1 - R_2}{tg\alpha}\left(R_1^2 + R_1 R_2 + R_2^2\right) = \frac{\pi}{3tg\alpha}\left(R_1^3 - R_2^3\right), \tag{4. 15}$$



Слика 4. 8. Конусно квачило

Како је угао конуса α, константан, то се може израз: $\frac{\pi}{3tg\alpha}$, сматрати константом, $k_1 = \frac{\pi}{3tg\alpha}$. Сила F, која се јавља на додирним повшинама конуса, дата је једначином (4.16), а обртни момент T, једначином (4.17).

$$F = \int p \sin\alpha dA = \int_{R_2}^{R_1} p \frac{2\pi r dr}{\sin\alpha} \sin\alpha = 2\pi p \int_{R_2}^{R_1} r dr = \pi p\left(R_1^2 - R_2^2\right), \tag{4. 16}$$

$$T = \int rfpdA = \int_{R_2}^{R_1} rfp \frac{2\pi r}{\sin\alpha} dr = \frac{2\pi pf}{\sin\alpha} \int_{R_2}^{R_1} r^2 dr = \frac{2\pi pf}{3\sin\alpha}\left(R_1^3 - R_2^3\right), \tag{4. 17}$$

где је p, pritisak, f, коефицијент трења и A површина контакта.

Како је, $\left(R_1^3 - R_2^3\right) = \left(R_1 - R_2\right)\left(R_1^2 + R_1R_2 + R_2^2\right)$, а из једначине (4.16) производ πp се може изразити у функцији силе, F и радијуса R_1, R_2, $\pi p = \frac{F}{\left(R_1^2 - R_2^2\right)}$ и када се ове трансформације уврсте у једначину (4.17), добиће се нови израз за обртни моменат, у функцији силе , F , и радијуса R_1 и R_2, једначина (4.18).

$$T = \frac{2Ff}{3\sin\alpha\left(R_1^2 - R_2^2\right)}\left(R_1 - R_2\right)\left(R_1^2 + R_1R_2 + R_2^2\right) = \frac{k_2\left(R_1^2 + R_1R_2 + R_2^2\right)}{\left(R_1 + R_2\right)}, \tag{4. 18}$$

где је $k_2 = \frac{2Ff}{3\sin\alpha}$.

Пошто је k_1 константа, функција циља се може усвојити као $f(R_1, R_2) = \left(R_1^3 - R_2^3\right)$.

По претпоставци минимални момент који се преноси је $5k_2$. Такође се претпоставља да је спољни пречник, $R_1 \equiv x_1$, два пута већи од унутрашњег, $R_2 \equiv x_2$. Тако оптимизациони проблем постаје:

$$\text{Минимизирати } f(\mathbf{X}) = \left(x_1^3 - x_2^3\right) \tag{4. 19}$$

$$\text{Ограничења: } g_1(\mathbf{X}) = \frac{x_1}{x_2} \geq 2 \tag{4. 20}$$

$$g_2(\mathbf{X}) = \frac{\left(x_1^2 + x_1 x_2 + x_2^2\right)}{\left(x_1 + x_2\right)} \geq 5 \tag{4. 21}$$

где је $\mathbf{X} = (x_1, x_2)$, при чему су ограничења пројектних променљивих: $1 \leq x_1$, $x_2 \leq 10$.

4.4.2.1. Резултати оптимизације конусног квачила

Упоредни резултати примене CS, FA и H-CS-FA алгоритма, приказани су у Табели 4.3, а поређење резултата добијених применом H-CS-FA алгоритма, са другим оптимизационим алгоритмима, приказани су у Табели 4.4. На Сликама 4.10-4.11, приказано је генерисање резултата током итеративног процеса.

Параметри алгоритма, за сва три случаја су били исти: број гнезда, односно свитаца n = 55, максималан број итерација MI = 1000.

Табела 4. 3. Упоредни резултати за оптимизацију конусног квачила FA, CS, **H-CS-FA**

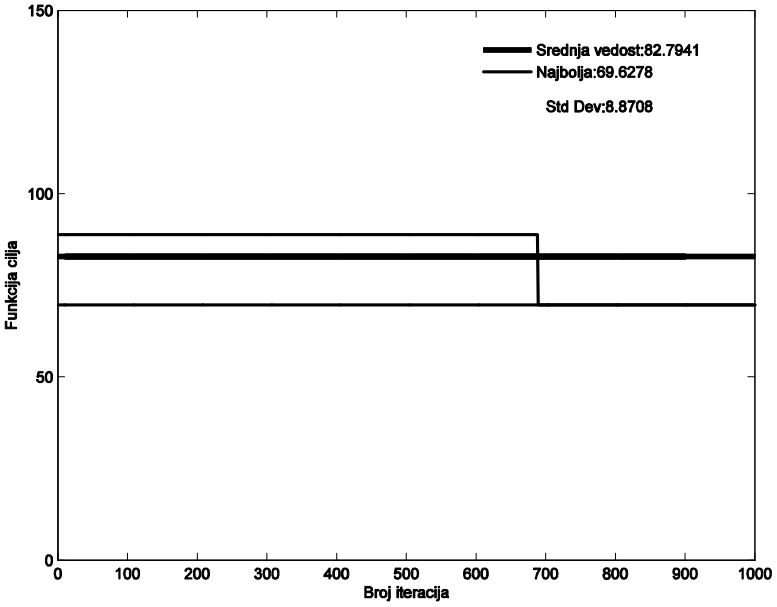
	FA	CS	**H-CS-FA**
x_1	4.2987150371	4.2858782136	**4.2858018767**
x_2	2.1405496597	2.1428154509	**2.1428730927**
g_1	2.0082295301	2.0001154144	**1.8677919002**
g_2	5.0102796429	2.0001225526	**5.0114689631**
Најбоље	69.627846804	68.8871584429	**68.8821578422**
Средња вредност	82.794140698	76.5070814752	**70.2565930332**
Најлошије	88.7649018821	315.0993333692	**110.483803939**
С.Д.	8.870822284	11.9088065757	**4.9081240612**

CS – Алгоритам кукавичје претраге, FA – алгоритам свица, **H-CS-FA** – хибридни алгоритам

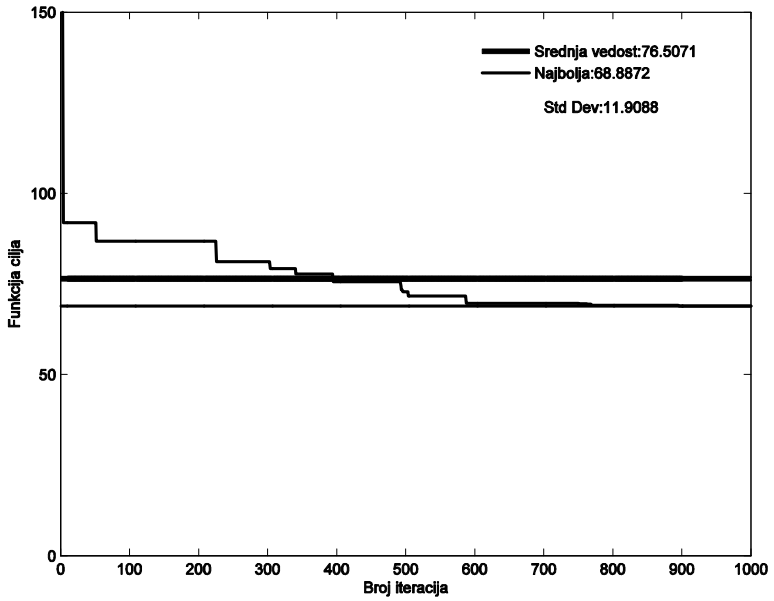
Табела 4. 4. Упоредни резултати за оптимизацију конусног квачила

	PSO [68]	ABC [68]	HEA_M [68]	HPABC [68]	**H-CS-FA**
Најбоље	68.877551	68.877551	68.88543	68.877551	**68.8821578422**

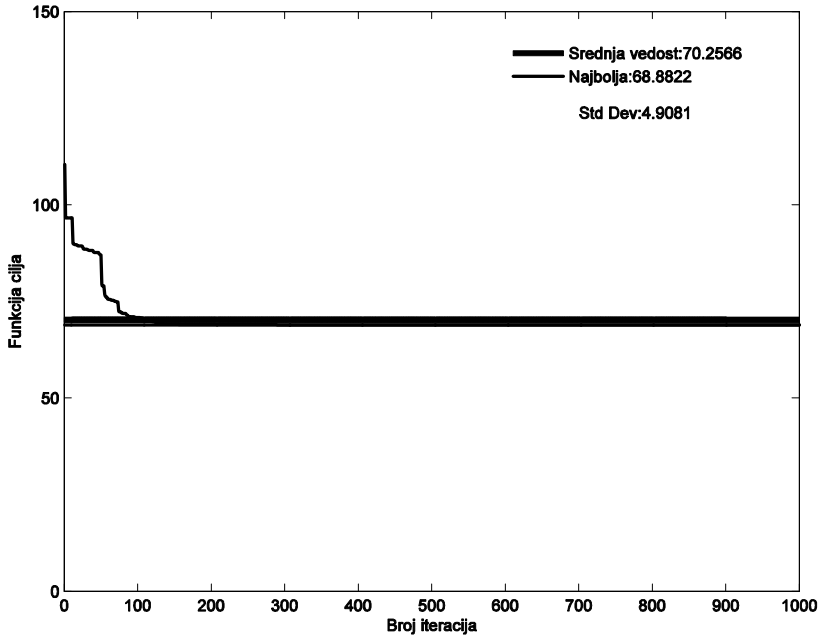
PSO –оптимизација роја честица, ABC – алгоритам вештачке колоније пчела, HEA – алгоритам хармонијских елемената (Harmony Elements Algorithm), HPABC – хибридни алгоритам оптимизације роја честица и вештачке колоније плчела (Hybrid Particle swarm +Artificial Bee Colony), **H-CS-FA** – хибридни алгоритам



Слика 4. 9. Најбоља и средња вредност функције циља при оптимизацији конусног квачила применом FA алгоритма



Слика 4. 10. Најбоља и средња вредност функције циља при оптимизацији конусног квачила применом CS алгоритма



Слика 4. 11. Најбоља и средња вредност функције циља при оптимизацији конусног квачила применом H-CS-FA алгоритма

Из Табеле 4.4, може се видети да хибридни алгоритам H-CS-FA, даје бољи резултат од алгоритма HEA_M, док су резултати добијени осталим наведеним алгоритмима бољи.

4.4.3. Модел I профила

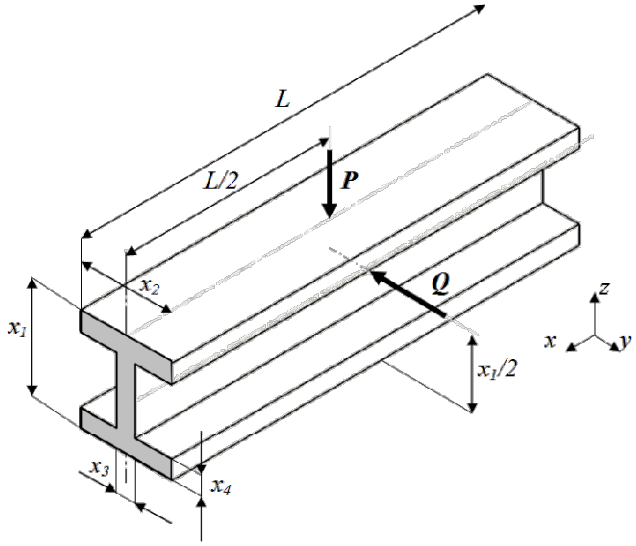
Оптимизацију I профила први су описали Hajela and Shih (1990). Овај проблем се састоји од четири пројектне променљиве $\mathbf{X}(x_1, x_2, x_3, x_4)$ које су различитих димензија, као што је приказано на Слици 4.12. На профил делује систем сила P (дуж z осе, осе профила) и Q (попречно на профил, дуж y осе). Циљ овог модела је минимизирање укупног попречног пресека профила.

Познате вредности су:

$\boldsymbol{P}$ = 600 [kN]; $\boldsymbol{Q}$ = 600 [kN]; Јунгов модул еластичности: $E = 2 \cdot 10^4 \left[\frac{kN}{cm^2} \right]$;

максимално дозвољени напон: $\sigma = 16 \left[\frac{kN}{cm^2} \right]$; дужина профила: L = 200 [cm].

Математички модел оптимизације I профила исказан је једначинама (4.22) – (4.25).



Слика 4. 12. I профил

$$\text{Минимизација } \left[f_1(\mathbf{X}), f_2(\mathbf{X})\right], \tag{4. 22}$$

где је:

$$f_1(\mathbf{X}) = 2x_2x_4 + x_3(x_1 - 2x_4), \tag{4. 23}$$

$$f_2(\mathbf{X}) = \frac{PL^3}{48EI}, \qquad I = \frac{x_3(x_1 - 2x_4)^3 + 2x_2x_4\left[4x_4^2 + 3x_1(x_1 - 2x_4)\right]}{12} \tag{4. 24}$$

уз функцији ограничења:

$$g_1(\mathbf{X}) = \frac{18000x_1}{x_3(x_1 - 2x_4)^3 + 2x_2x_4\left(4x_4^2 + 3x_1(x_1 - 2x_4)\right)} + \frac{15000x_2}{(x_1 - 2x_4)x_3^3 + 2x_2^3x_4} \le 16, \tag{4. 25}$$

При чему су ограничења пројектних променљивих: $10 \le x_1 \le 80$, $10 \le x_2 \le 50$, $0.9 \le x_3 \le 5$, $0.9 \le x_4 \le 5$.

4.4.3.1. Резултати оптимизације I профила

Упоредни резултати примене CS, FA и H-CS-FA алгоритма, приказани су у Табели 4.5, а поређење резултата добијених применом H-CS-FA алгоритма, са другим оптимизационим алгоритмима, приказани су у Табели 4.6. На Сликама 4.13-4.15, приказано је генерисање резултата током итеративног процеса.

Параметри алгоритма, за сва три случаја су били исти: број гнезда, односно свитаца $n = 55$, максималан број итерација $MI = 1000$.

Табела 4. 5. Упоредни резултати за оптимизацију I профила FA, CS, **H-CS-FA**

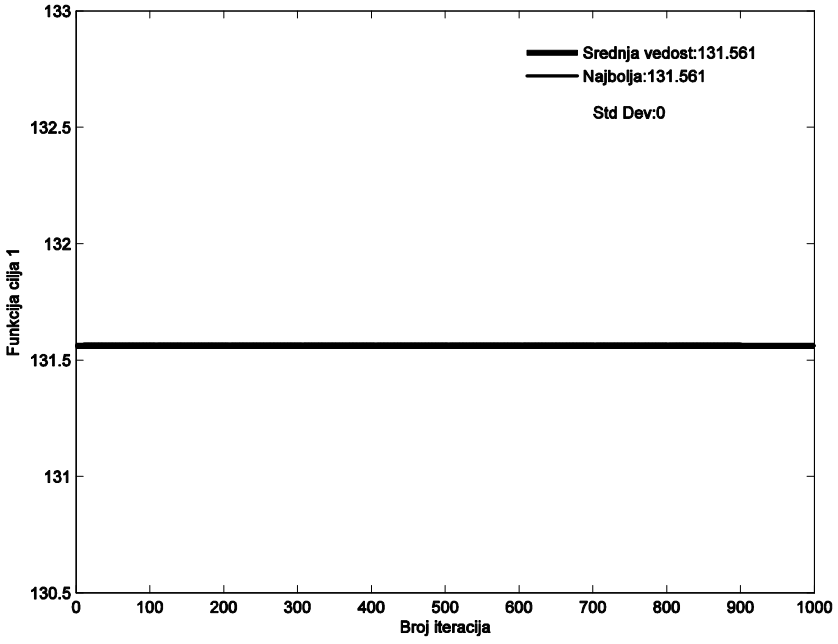
	FA		CS		**H-CS-FA**	
x_1	48.2862145205		79.9785455319		**65.8512128220**	
x_2	34.9952945810		28.1795967907		**41.4181464655**	
x_3	1.3677938109		0.9022339456		**3.1687356284**	
x_4	0.9741337310		0.900000000		**2.8581910417**	
g_1	6.1367887590		15.7438971095		**15.1948428986**	
	f_1	f_2	f_1	f_2	f_1	f_2
Најбоље	131.56095095	0.1310104882	121.25861182	0.063045528	**114.49146904**	**0.0212728623**
Средња вредност	131.56095095	0.1310104882	121.31428016	0.063053371	**133.082073155**	**0.077.9278132**
Најлошије	131.56095095	0.1310104882	176.926956372	0.070888152	**427.31333105**	**0.0842820875**
С.Д.	0.0000000000	0.0000000000	1.76038762355	248 e-006	**54.201501478**	**0.0117006197**

CS – Алгоритам кукавичје претраге, FA – алгоритам свица, **H-CS-FA** – хибридни алгоритам

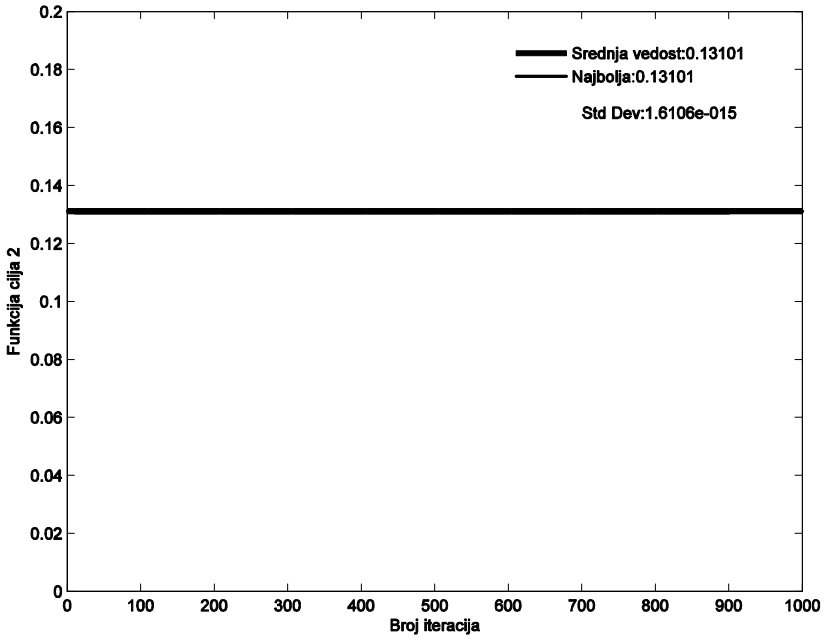
Табела 4. 6. Упоредни резултати за оптимизацију I профила.

	FMO [86]		WNNC[86]		**H-CS-FA**	
x_1	80		80		**65.8512128220**	
x_2	26.1303		50		**41.4181464655**	
x_3	1.4637		0.9		**3.1687356284**	
x_4	4.7086		2.8160		**2.8581910417**	
g_1	Н/Д		Н/Д		**15.1948428986**	
	f_1	f_2	f_1	f_2	f_1	f_2
Најбоље	349.3860	0.0128	348.5352	0.0111	**114.49146904**	**0.0212728623**

FMO – фази вишекритеријумска оптимизација (fuzzy multi-objective optimization), WNNC – метод широке нормализација нормалних ограничења (wide normalized normal constraint) CS-FA – хибридизовани CS+FA алгоритам

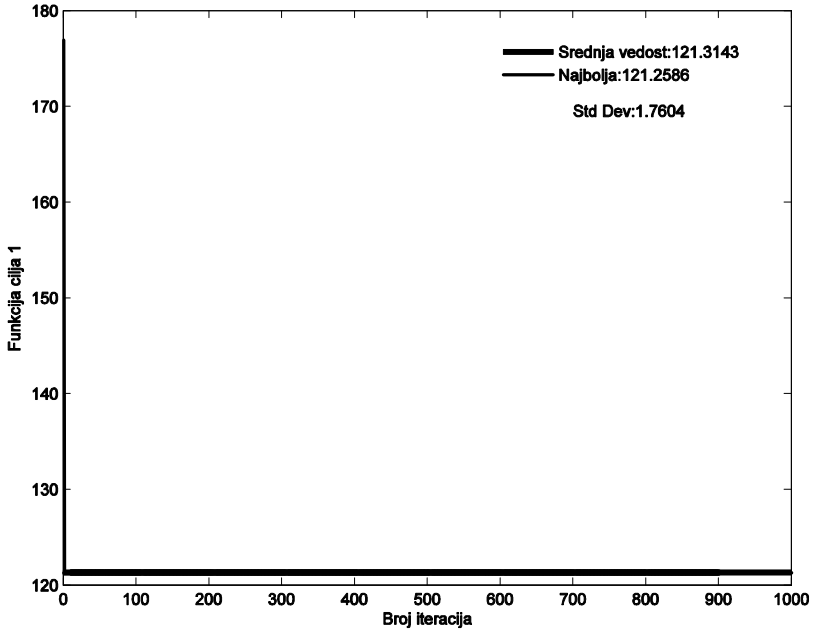


Функција циља f_1, једначина (4.23)

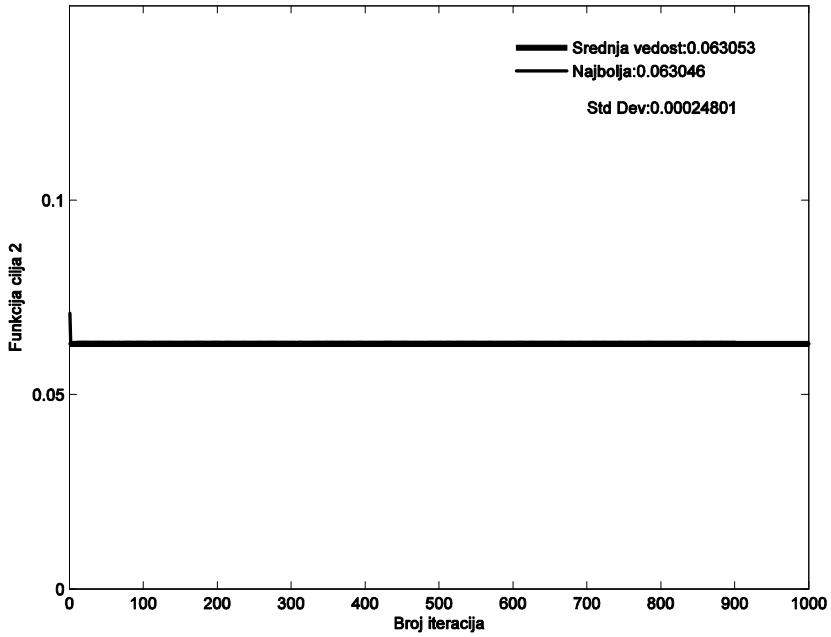


Функција циља f_2, једначина (4.24)

Слика 4. 13. Најбоља и средња вредност функције циља: f_1, f_2; при оптимизацији I профила применом FA алгоритма

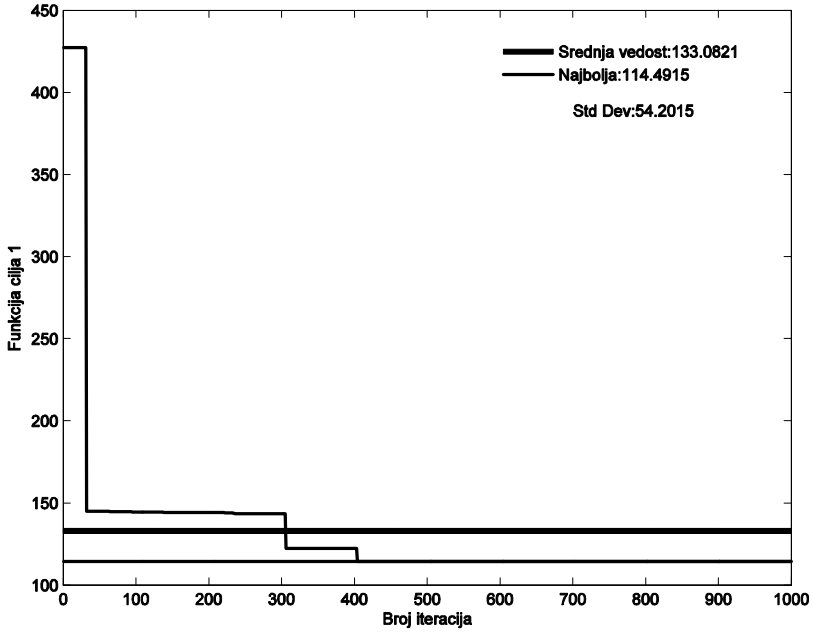


Функција циља f_1, једначина (4.23)

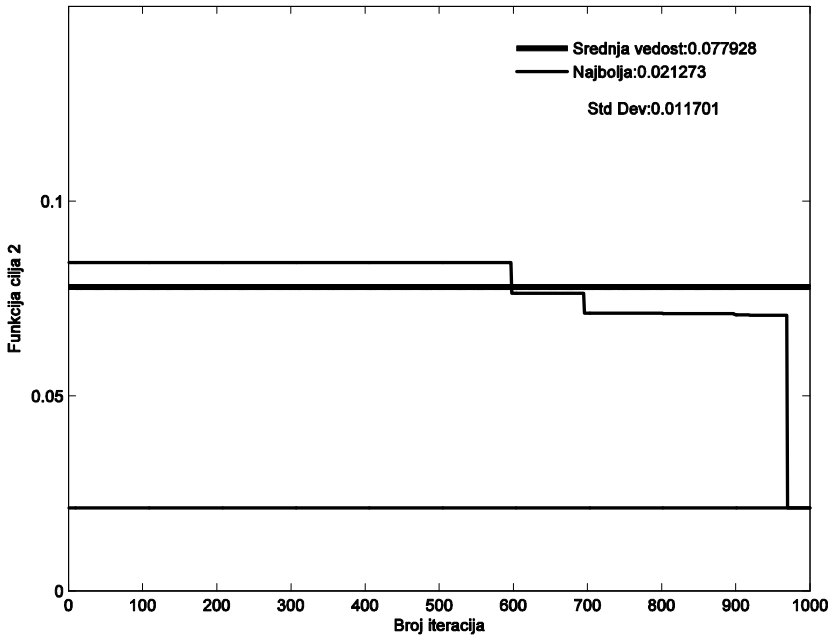


Функција циља f_2, једначина (4.24)

Слика 4. 14. Најбоља и средња вредност функције циља: f_1, f_2; при оптимизацији I профила применом CS алгоритма



Функција циља f_1, једначина (4.23)



Функција циља f_2, једначина (4.24)

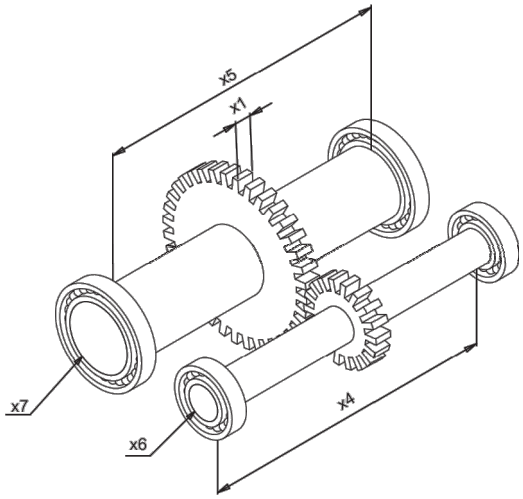
Слика 4. 15. Најбоља и средња вредност функције циља: f_1, f_2; при оптимизацији I профила применом H-CS-FA алгоритма

Анализирајући Табелу 4.5 и Слике 4.13-4.15., уочава се да се добри резултати добијају применом H-CS-FA алгоритма. Наиме, са Слике 4.13, се уочава да FA и CS алгоритам, одмах упадну у зону локалног минимума, а да H-CS-FA алгоритам, током итеративног поступка, претражује комплетан простор дефинисан пројектним променљивим. Такође, за обе функције циља, H-CS-FA алгоритам даје најбоље резултате.

Када се H-CS-FA алгоритам упореди са резултатим из [86], може се уочити да овај алгоритам даје далеко најбољу вредност функције циља f_1 у поређењу са FMO и WNNC алгоритмима, док је вредност функције циља f_2, лошија у поређењу са поменутим алгоритмима.

4.4.4. Модел мењача

Модел мењача, Слика 4.16, који се користи у овом поглављу је преузет из Kurapati и Azarm (2000), али са три функције циља (вишекритеријумски модел оптимизације) како је урађено по Huang et al. (2006). Као што се види са слике, проблем се састоји од седам пројектних променљивих, $\mathbf{X}(x_1, x_2, x_3, x_4, x_5, x_6, x_7)$.



Слика 4. 16. Мењач и пројектне промењљиве

Пројектне промењљиве представљају:

x_1 - ширина зупчаника (cm)

x_2 - модул зупца зупчаника (cm)

x_3 - број зубаца на зупчанику

x_4 - осно растојање I (cm)

x_5 - осно растојање II (cm)

x_6 - пречник вратила I (cm)

x_7 - пречник вратила II (cm)

Математичка формулација проблема дата је једначинама (4.26) – (4.40).

$$\text{Минимизација } \left[f_1(\mathbf{X}), f_2(\mathbf{X}), f_3(\mathbf{X})\right], \tag{4. 26}$$

где је:

$$f_1(\mathbf{X}) = 0.7854x_1x_2^2\left(\frac{10}{3} + 14.9334x_3 - 4302934\right) - 1.508x_1\left(x_6^2 + x_7^2\right) + 7.4777\left(x_6^3 + x_7^3\right) + 0.7854\left(x_4x_6^2 + x_5x_7^2\right), \tag{4. 27}$$

$$f_2(\mathbf{X}) = \frac{\sqrt{\left(745x_4x_2^{-1}x_3^{-1}\right)^2 + 1.69e^7}}{0.1x_6^3}, \tag{4. 28}$$

$$f_3(\mathbf{X}) = \frac{\sqrt{\left(745x_5x_2^{-1}x_3^{-1}\right)^2 + 1.575e^8}}{0.1x_7^3}, \tag{4. 29}$$

уз функције ограничења:

$$g_1(\mathbf{X}) = 27x_1^{-1}x_2^{-2}x_3^{-3} - 1 \le 0, \tag{4. 30}$$

$$g_2(\mathbf{X}) = 397.5x_1^{-1}x_2^{-2}x_3^{-2} - 1 \le 0, \tag{4. 31}$$

$$g_3(\mathbf{X}) = 1.93x_2^{-1}x_3^{-1}x_4^3x_6^{-4} - 1 \le 0, \tag{4. 32}$$

$$g_4(\mathbf{X}) = 1.93x_2^{-1}x_3^{-1}x_5^3x_7^{-4} - 1 \le 0, \tag{4. 33}$$

$$g_5(\mathbf{X}) = x_2x_3 - 40 \le 0, \tag{4. 34}$$

$$g_6(\mathbf{X}) = x_1x_2^{-1} - 12 \le 0, \tag{4. 35}$$

$$g_7(\mathbf{X}) = 5 - x_1x_2^{-1} \le 0, \tag{4. 36}$$

$$g_8(\mathbf{X}) = 1.9 - x_4 + 1.5x_6 \le 0, \tag{4. 37}$$

$$g_9(\mathbf{X}) = 1.9 - x_5 + 1.5x_7 \le 0, \tag{4. 38}$$

$$g_{10}(\mathbf{X}) = f_2(x) - 1300 \le 0, \tag{4. 39}$$

$$g_{11}(\mathbf{X}) = f_3(x) - 850 \le 0, \tag{4. 40}$$

При чему су ограничења пројектних променљивих: $2.6 \le x_1 \le 3.6$, $0.7 \le x_2 \le 0.8$, $17 \le x_3 \le 28$, $7.3 \le x_4 \le 8.3$, $7.3 \le x_5 \le 8.3$, $2.9 \le x_6 \le 3.9$, $5 \le x_5 \le 5.5$.

4.4.4.1. Резултати оптимизације мењача

Упоредни резултати примене CS, FA и H-CS-FA алгоритма, приказани су у Табели 4.4, а упоређење са резултаима из [86], у Табели 4.8. Генерисање резултата током итеративног процеса приказан је на Сликама 4.17-4.19.

Параметри алгоритма, за сва три случаја су били исти: број гнезда, односно свитаца $n = 55$, максималан број итерација $MI = 1000$.

Табела 4. 7. Упоредни резултати за оптимизације мењача FA, CS, H-CS-FA

	FA			CS			**H-CS-FA**		
x_1	3.5664072552			3.6000000000			**3.5999940212**		
x_2	0.7132814383			0.7194691170			**0.7000000000**		
x_3	24.0000000000			19.0000000000			**17.0000000000**		
x_4	7.5700986867			8.0801567898			**8.2634465748**		
x_5	7.9253865527			8.1880339478			**8.0065270475**		
x_6	3.3222141324			3.7327958994			**3.9000000000**		
x_7	5.4481855530			5.5000000000			**5.5000000000**		
g_1	-0.9991535076			-0.0099788760			**-0.9993023671**		
g_2	-0.6759815046			-0.0040911479			**-0.7124202312**		
g_3	-0.6292730401			-0.0061636878			**-0.6426211813**		
g_4	-0.9411959492			-0.0091530072			**-0.9416266136**		
g_5	-21.4537278135			-0.26330086777			**-20.4000000000**		
g_6	-6.9999909289			-0.06996310592			**-6.8599348933**		
g_7	-0.0000090711			-0.00003689407			**-0.1400651067**		
g_8	-0.6872455369			-0.00580962940			**-1.0831183220**		
g_9	-0.0325781341			-0.00238033947			**-0.2055790773**		
g_{10}	-175.4930971373			-5.05089764458			**-371.55639952**		
g_{11}	-73.6456619819			-0.95209226933			**-95.4558430116**		
	F_1	F_2	F_3	F_1	F_2	F_3	F_1	F_2	F_3
Најбоље	4700.3739	1124.736	776.3339	4854.4597	1051.065	1051.065	**3356.4753**	**698.4919**	**754.9155**
Средња вредност	4793.7349	1217.89	868.8135	17623.164	13431.114	13431.113	**3380.2114**	**725.2296**	**778.5494**
Најлошије	18965.4005	15868.379	15326.765	55782.152	50695.149	50695.14	**26869.4501**	**24757.4370**	**24364.5281**
С.Д.	646.6540	644.5198	641.9742	13200.625	12872.222	12872.22	**743.5385**	**760.7298**	**746.6007**

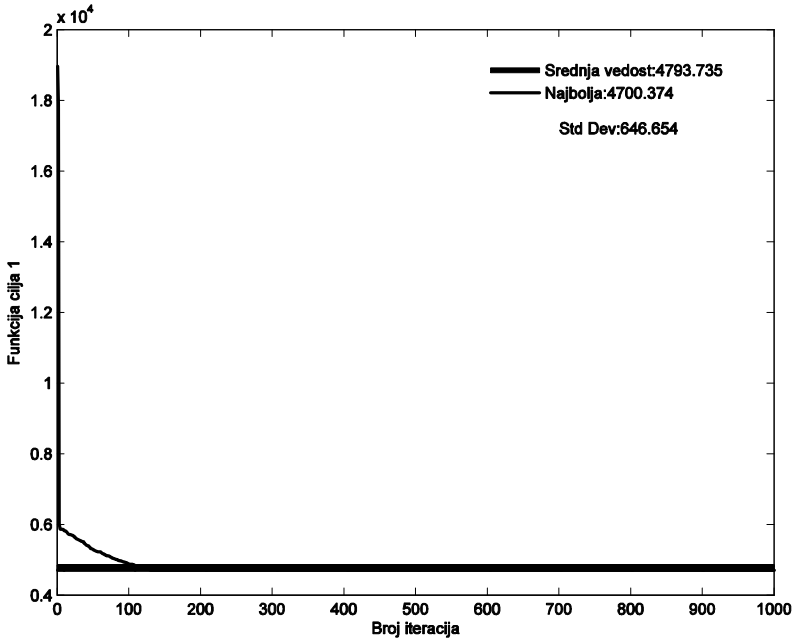
CS – Алгоритам кукавичје претраге, FA – алгоритам свица, **H-CS-FA** – хибридни алгоритам

Табела 4. 8. Упоредни резултати за оптимизације мењача

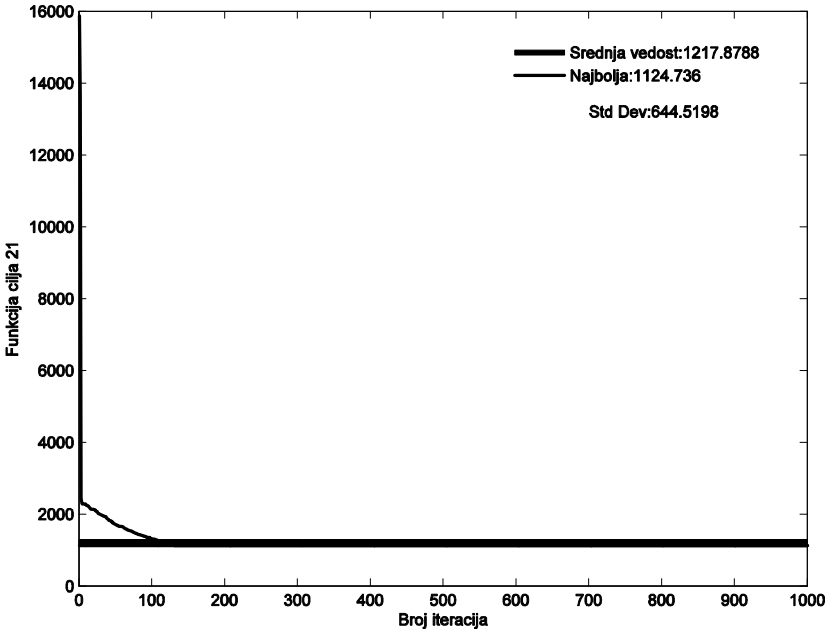
FMO [86]			WNNC[86]			**H-CS-FA**		
F_1	F_2	F_3	F_1	F_2	F_3	F_1	F_2	F_3
3425.0	879.8	797.6	3412.1	829.4	754.7	**3356.4753**	**698.4919**	**754.9155**

FMO – фази вишекритеријумска оптимизација (fuzzy multi-objective optimization), WNNC – метод широке нормализација нормалних ограничења (wide normalized normal constraint) H-CS-FA – хибридизовани CS+FA алгоритам

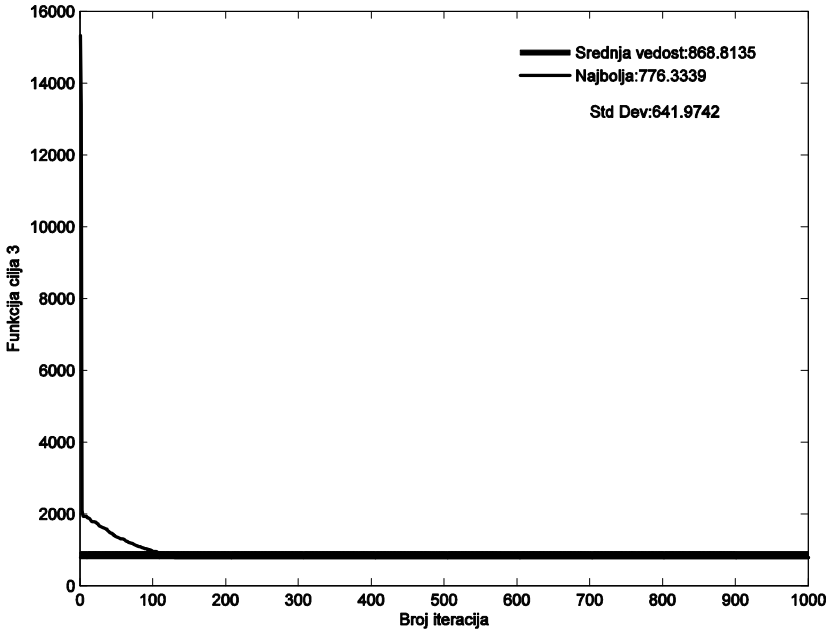
На основу добијених резултата, Табеле 4.7-4.8, хибридни алгоритам H-CS-FA, даје најбоље резултате за све три функције циља, међутим у овом примеру може се уочити да је стандарна девијација изузетно висока, што значи да алгоритам у неком тренутку може да упадне у замку локалног минимума.



Функција циља f_1, једначина (4.25)

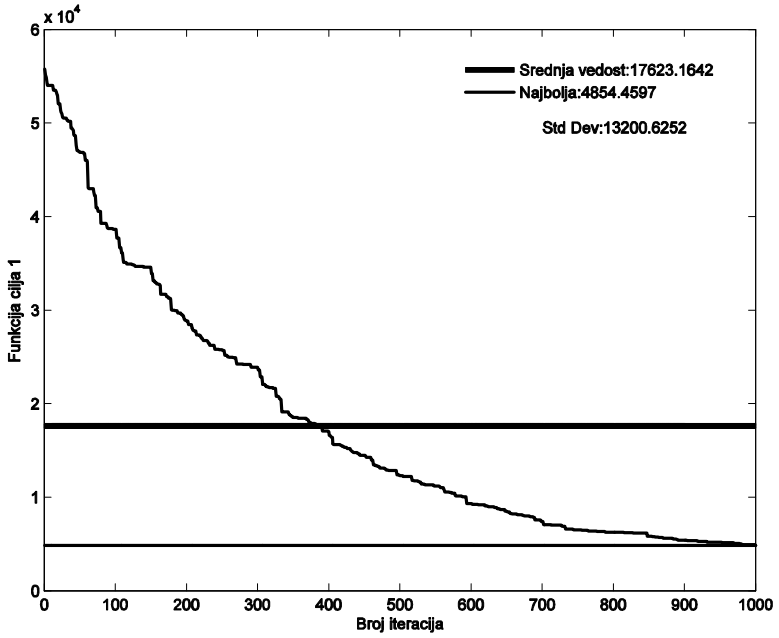


Функција циља f_2, једначина (4.26)

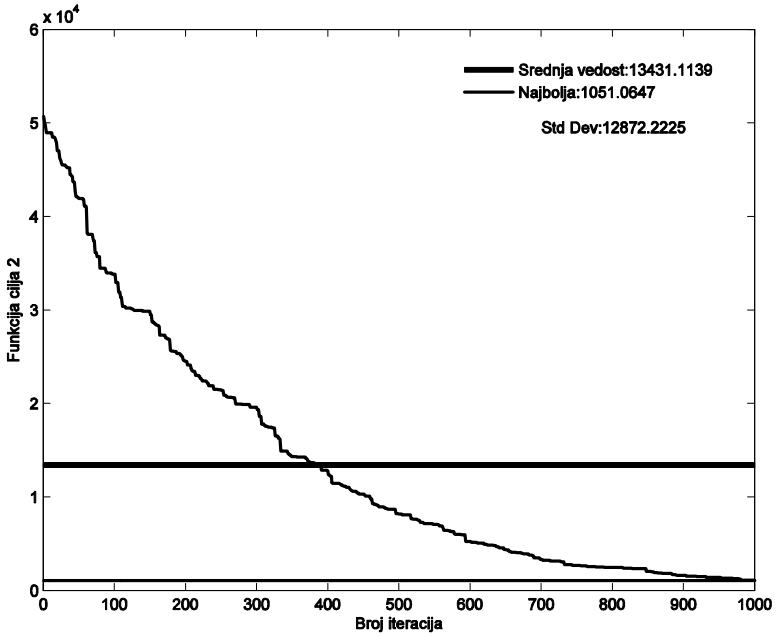


Функција циља f_3, једначина (4.27)

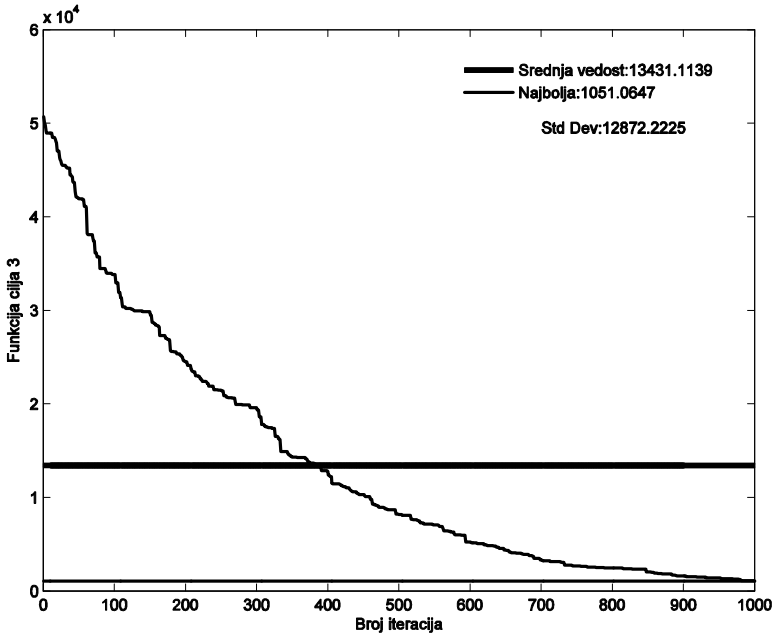
Слика 4. 17. Најбоља и средња вредност функције циља при оптимизацији мењача применом FA алгоритма



Функција циља f_1, једначина (4.25)

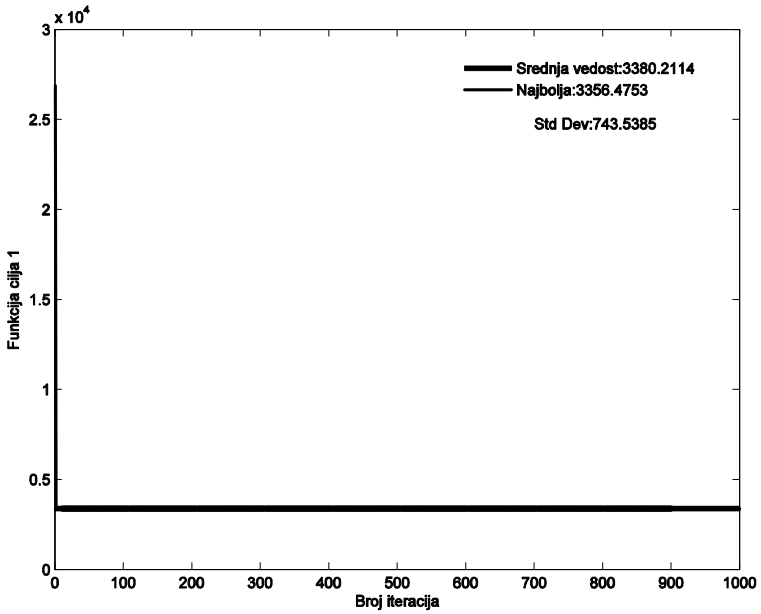


Функција циља f_2, једначина (4.26)

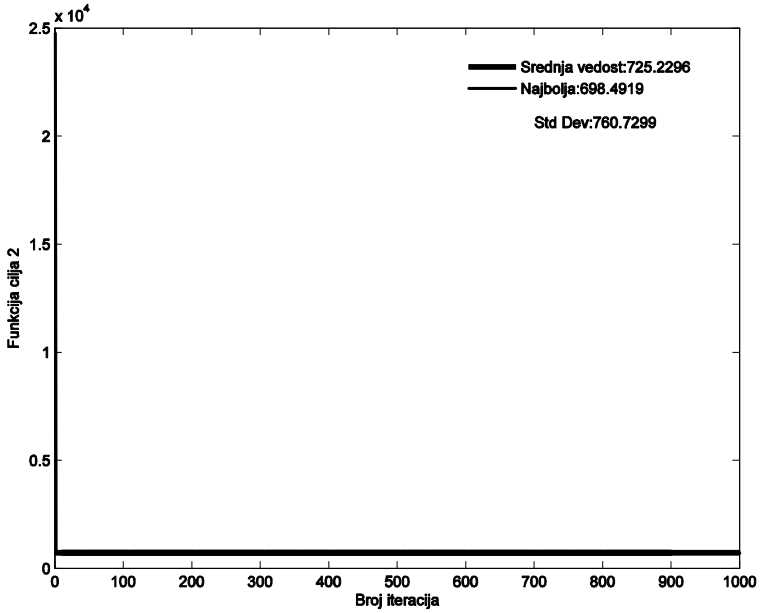


Функција циља f_3, једначина (4.27)

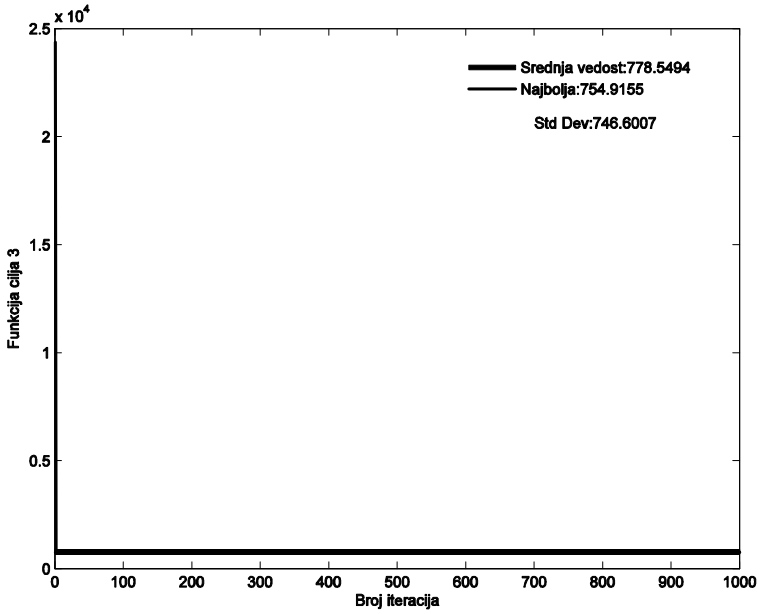
Слика 4. 18. Најбоља и средња вредност функције циља при оптимизацији мењача применом CS алгоритма



Функција циља f_1, једначина (4.25)



Функција циља f_2, једначина (4.26)



Функција циља f_3, једначина (4.27)

Слика 4. 19. Најбоља и средња вредност функције циља при оптимизацији мењача за функцију F_1 применом H-CS-FA алгоритма

4.4.5. Оптимизација носача са три променљиве

У моделу који следи, [79] идеја је да се изврши минимизирање висина попречних пресека h_1, h_2, h_3, h_4: $\begin{bmatrix} x_1 & x_2 & x_3 & x_4 \end{bmatrix} = \begin{bmatrix} h_1 & h_2 & h_3 & h_4 \end{bmatrix}$ свих елемената носача датог на Слици 4.20. Овај оптимизациони проблем одређен је са 3 независне и 2 зависне променљиве. Вертикално померање, u_A (g_1) тачке А на слободном крају горњег члана носача, је унапред дефинисано са максимално дозвољеном вредношћу. Носач је оптерећен континуалним оптерећењем (q_1, q_2) на хоризонталним члановима носача, односно хоризонталном силом F, која делује на вертикални елемент носача, Слика 4.20.

Познате вредности су:

$$F = 30 \cdot 10^3 \left[N \right]; q_1 = 3 \cdot 10^3 \left[N/m \right]; q_2 = 4 \cdot 10^3 \left[N/m \right];$$

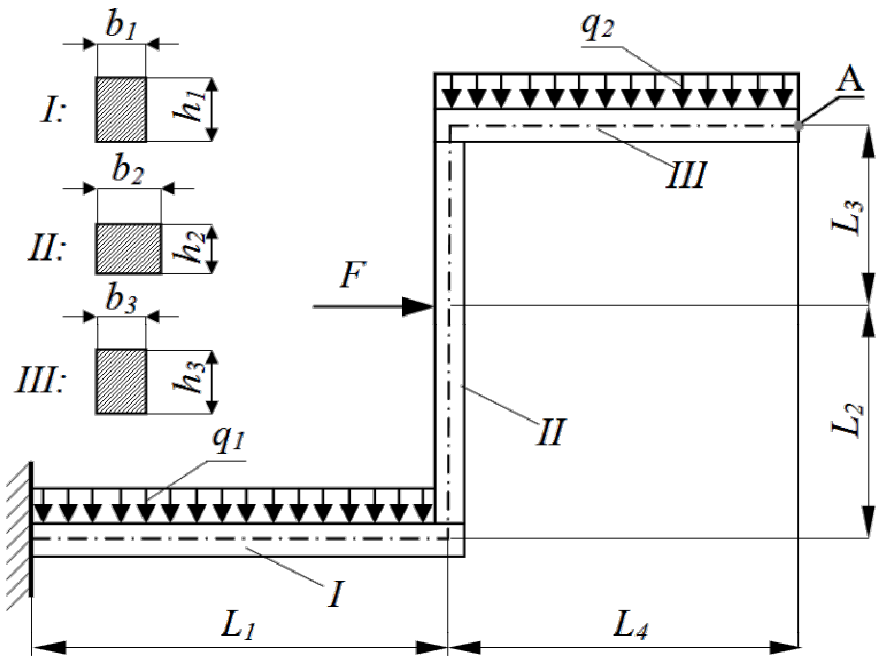
Јунгов модул еластичности: $E = 20 \cdot 10^9 \left[Pa \right];$

дужине: $L_1 = 4\left[m \right];\ L_2 = 4\left[m \right];\ L_3 = 2\left[m \right]; L_4 = 4\left[m \right],$

ширине профила: $b_1 = b_2 = b_3 = 0.2\left[m \right];$

главни момент инерције профила:

$$I_1 = 1.666\overline{6} . \cdot 10^{-2} \cdot h_1^3 \left[m^4 \right];\ I_2 = 1.666\overline{6} . \cdot 10^{-2} \cdot h_2^3 \left[m^4 \right];\ I_3 = 1.666\overline{6} . \cdot 10^{-2} \cdot h_3^3 \left[m^4 \right]$$



Слика 4. 20. Оптимизациони модел са пројектним промењљивима

Математичка формулација проблема дата је једначинама (4.41) – (4.42).

$$\text{Минимизација } f(\mathbf{X}) = 0.8x_1 + x_2 + 0.8x_3, \tag{4. 41}$$

при ограничењу:

$$u_A(\mathbf{X}) = \left[\frac{11.2480 \cdot 10^{-3}}{x_1^3} + \frac{3.5399 \cdot 10^{-3}}{x_2^3} + \frac{0.3840 \cdot 10^{-3}}{x_3^3} \right] \le 0.05[m], \tag{4. 42}$$

Где је $\mathbf{X} = (x_1, x_2 x_3)$, при чему су ограничења пројектних промењљивих: $0.1 \le x_1 \le 0.9$, $0.1 \le x_2 \le 0.9$, $0.1 \le x_3 \le 0.9$.

4.4.5.1. Резултати оптимизације тродимензионалног носача

Упоредни резултати примене CS, FA и H-CS-FA алгоритма, приказани су у Табели 4.9, док је генерисање резултата током итеративног процеса приказни на Сликама (4.21)-(4.23).

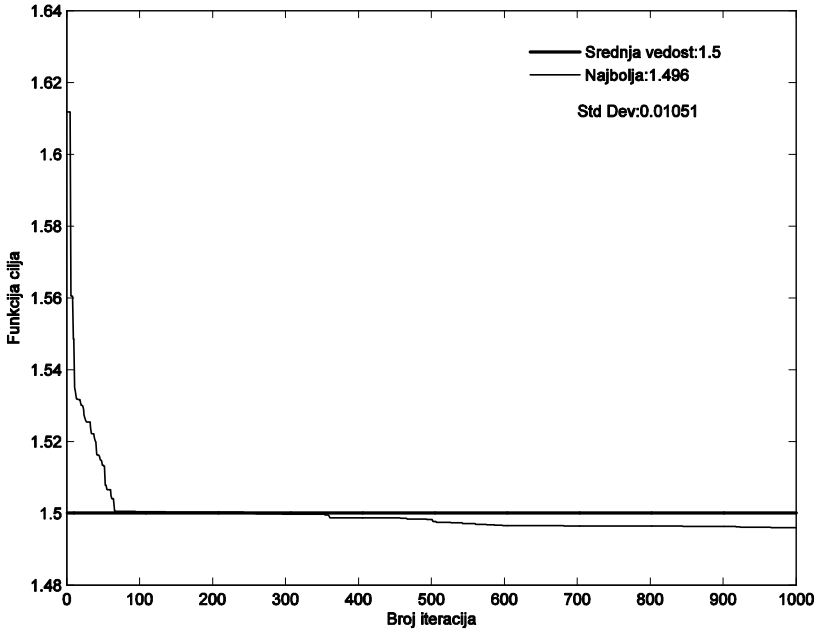
Параметри алгоритма, за сва три случаја су били исти: број гнезда, односно свитаца n = 55, максималан број итерација MI = 1000.

Табела 4. 9. Упоредни резултати за оптимизацију тродимезионалног носача.

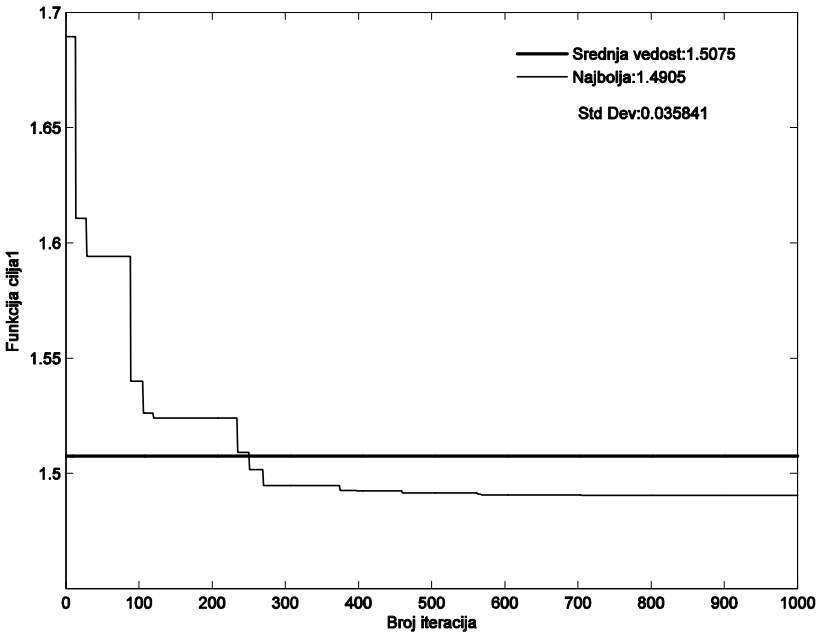
	FA	CS	Графичка метода [79]	ANSYS/Design Optimization [79]	**H-CS-FA**
x_1	0.7646629023	0.8048012636	0.80458	0.80458	**0.8907125203**
x_2	0.5682112729	0.5683274364	0.56993	0.44054	**0.5619783280**
x_3	0.5397788866	0.3479183333	0.34585	0.51398	**0.3244493099**
g_1	0.0499809669	0.049998251	0.05	0.06558	**0.0500000000**
Најбоље	1.4960387043	1.4905031139	1.4903	1.4618	**1.4900914500**
Средња вредност	1.5000356637	1.50753683611	Н.Д	Н.Д	**1.4904764814**
Најлошије	1.6117647041	1.6894202837	Н.Д	Н.Д	**1.5341077925**
С.Д.	0.0105095924	0.0358408060	Н.Д	Н.Д	**0.0022841807**

CS – Алгоритам кукавичје претраге, FA – алгоритам свица, **H-CS-FA** – хибридни алгоритам

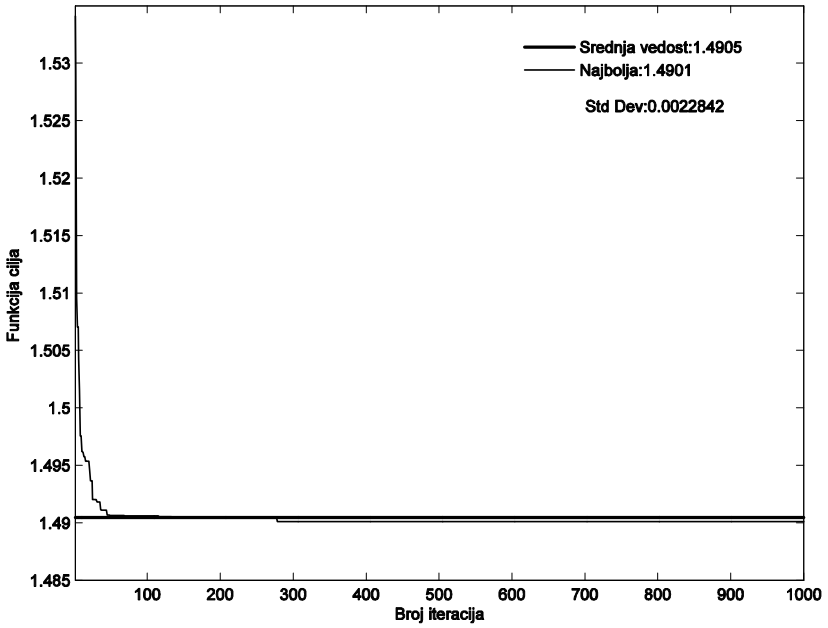
H-CS-FA, алгоритам даје бољи резултат у поређењу са CS и FA алгоритмом, Табела 4.9. Метода ANSYS/Design Optimization - Subproblem Approximation Method, даје бољи резултат, међутим из дате табеле може се видети да је нарушено ограничење $g_1 \le 0.05$.



Слика 4. 21. Најбоља и средња вредност функције циља тродимензионалног носача применом FA алгоритма



Слика 4. 22. Најбоља и средња вредност функције циља циља тродимензионалног носача применом CS алгоритма



Слика 4. 23. Најбоља и средња вредност функције циља циља тродимензионалног носача применом H-CS-FA алгоритма

4.5 ЗАКЉУЧНЕ НАПОМЕНЕ УЗ ЧЕТВРТО ПОГЛАВЉЕ

У овом поглављу урађена је хибридизација два основна алгоритма, CS (кукавичје претраге) и FA (алгоритам свица). За све примере коришћени су исти параметри алгоритма, као и исти број итерација. Број кукавица, односно свитаца је такође био исти и износио је 55.

Предложени хибридни алгоритам за основу има CS алгоритам у који је инкорпориран део FA алгоритма, што је приказано у Алгоритму 4.3 и на Слици 4.1. Наиме, уместо да се испразне гнезда када се достигне вероватноћа, p_a, проналажења „лоших“ гнезда, у алгоритам је додат део FA алгоритма у коме се проналази свитац који светли са највећим интезитетом светлости.

Као што се може приметити, из приложених табела и дијаграма, резулати добијени хибридизованим оптимизационим алгоритмом H-CS-FA, су бољи од резултата добијених основним CS, односно FA алгоритмом.

Алгоритам кукавичје претраге даје нешто боље резултате од резултата добијених алгоритмом свица. Међутим конвергенција је знатно боља код примене алгоритма свица. Хибридизацијом ова два поменута алгоритма, добили смо најбоље карактеристике од оба. Брзи улазак у област оптимума (FA) и боље претраживање тог простора (CS), који је резултирао најбољим решењима.

Упоређујући резултате, за свих пет примера примењене механике, из цитиране литературе и H-CS-FA алгоритма, може се закључити да у свим случајевима, примењени хибридизовани алгоритам даје приближне и боље резултате, у односу на резултате добијене алгоритмом FA, односно CS.

Поглавље

5

МОДИФИКОВАНИ KRILL HERD (KH) АЛГОРИТАМ

5. МОДИФИКОВАНИ KRILL HERD – (KH) АЛГОРИТАМ

5.1. ПОНАШАЊЕ АНТАРКТИЧКОГ КРИЛА

Антарктички крил (енг. Krill, латински *Euphausia superba*) је једна од најбоље проучених врста које живе у океану. Ове раколике животиње припадају групи зоопланктона и могу да нарасту од 4 до 5cm. Станиште ових животињица су горњи слојеви воденог стуба у оквиру кога могу да мењају своју позицију, али не и да се супротставе морским струјама. Због ове чињенице, не могу да мигрирају као што то чине остале животиње.

Хране се искључиво микроскопским фитопланктоном (фито – биљка), којим је иначе богата вода Антарктика. Крил се окупља у великим јатима, за које се процењује да могу да буду тешка и 2 милиона тона и да се простиру на преко 450 квадратних колиметара.

Јата крила, се подижу или спуштају у воденом стубу у зависности од доба дана. Генерално се подижу ноћу или по дану са мањим интезитетом светла да би се хранили. Током дана, понирући, враћају се у дубље воде и до неколико стотина метара [90].

Као што је већ напоменуто, једна од главних особина антарктичког крила је окупљање у огромна јата, [114]. При нападу предатора, као што су фоке, пингвини или морске птице, уклања се појединачни крил. Резултат тих напада је смањење густине јата. Формирање јата крила после напада зависи од више параметара. Окупљање индивидуалних крилова у јато је вишекритеријумски процес у који су укључена два главна циља:

1. повећање густине јата, и
2. храњење.

Процес окупљања крила у јато је узет, [114], у разматрање при формирању метахеуристичког алгоритма за решавање проблема глобалне оптимизације. Густина, зависна од привлачности крила (повећање густине) и потрага за храном (област са високом концетрацијом хране) су искоришћени као циљеви који на крају доводе до окупљања крила у јато око глобалног минимума. У овом процесу, индивидуални крил, при образовању највеће густине јата и при потрази за храном, се померају ка најбољем решењу. Другим речима, што је мање растојање између крила (већа густина) и што је мање растојање од плена (хране), то функција циља има мању вредност.

5.2. ЛАГРАНЖЕВ МОДЕЛ ОКУПЉАЊА АНТАРТИЧКОГ КРИЛА У ЈАТО

У природном окружењу, подешавање индивидуалног крила у простору, предпоставља се да је комбинација растојања до хране и растојања између крила којим се добија највећа густина јата. Зато се положај (виртуално растојање) узима за вредност функције циља. Временски зависна позиција појединачног крила у 2D простору регулише се преко три главне активности, [114]:

1. Померање проузроковано кретањем других индивидуа N_i;
2. Потрага за храном F_i; и
3. Случајна дифузија D_i.

Како је познато да оптимизациони алгоритми морају да буду у стању да претражују простор произвољних димензија, Лагранжев модел, једначина (5.1) је генерализован за n - димензиони простор, [114].

$$\frac{dX_i}{dt} = N_i + F_i + D_i \tag{5. 1}$$

где је: N_i - померање индивидуалног крила услед померања осталих крила приликом окупљања у јато; F_i - померање у потрази за храном, и D_i – физичка дифузија i – тог крила.

5.2.1. Померање индивидуалног крила, N_i

Померање појединачног крила је дато једначином (5.2), [114]. Ово померање крила у простору је у последица тежње да се одржи максимална густина јата, односно међусобног утицаја суседних крила.

$$N_i^{new} = N_{max} \cdot \alpha_i + N_i^{old} \cdot \omega_n \tag{5. 2}$$

Проузроковани правац кретања α_i, једначина (5.3), се процењује на основу локалне густине јата – локални ефекат, жељене густине јата – циљни ефекат и одбојне густине – одбојни ефекат. Коефицијент инерције – ω_n се одређује у опсегу [0,1]. N_{max} – представља максимално индуковану силу, где N_i^{old}, представља последњи индуковани померај.

$$\alpha_i = \alpha_i^{local} + \alpha_i^{target} \tag{5. 3}$$

где је:

α_i^{local} - утицај суседног крила на померање i – тог крила, једначина (5.4) и

α_i^{target} - утицај крила са најбољом вредношћу функције циља на померање *i* – тог крила, једначина (5.5).

$$\alpha_i^{local} = \sum_{j=1}^{NN} \hat{K}_{ij} \hat{X}_{ij} \tag{5. 4}$$

$$\alpha_i^{target} = C^{best} \cdot \hat{K}_{i,best} \cdot \hat{X}_{i,best} \tag{5. 5}$$

где је:

$$\hat{\mathbf{X}}_{ij} = \frac{\mathbf{X}_j - \mathbf{X}_i}{\left\| \mathbf{X}_j - \mathbf{X}_i \right\| + \varepsilon} \tag{5. 6}$$

$$\hat{K}_{ij} = \frac{K_i - K_j}{K^{worst} - K^{best}} \tag{5. 7}$$

K^{worst}, K^{best} – су најлошија и најбоља вредност функције циља,

K_i – вредност функције циља *i*-тог крила, *i = 1,..,NK*; где је *NK* број крила,

K_j – вредност функције циља *j*-тог суседног крила, *j = 1,..,NN*; где је *NN* број суседних крила *i*-тог крила,

$\boldsymbol{X_i}$ и $\boldsymbol{X_j}$ – представљају позиције, вредности пројектних променљивих, *i*-тог и његовог *j*-тог суседног крила,

ε – позитивни мали број, који се додаје количнику у једначини (5.6) да би се избегао сингуларитет,

C^{best} - случајни коефицијент који се рачуна у зависности од случајног броја: $rand \in [0,1]$ и односа тренутне итерације *I*, и укупног броја итерација I_{max}, који се рачуна по једначини (5.8).

$$C^{best} = 2 \cdot (rand + \frac{I}{I_{max}}) \tag{5. 8}$$

rand – случајни број из опсега [0,1]

I – актуелни број итерације,

I_{max} – укупни број итерација.

$\hat{K}_{i,best}$ - се израчунава на потпуно исти начин као и $\hat{K}_{ij}$, једначина (5.7), где се вредност функције циља, за *j*-тог суседног крила, замењује најбољом

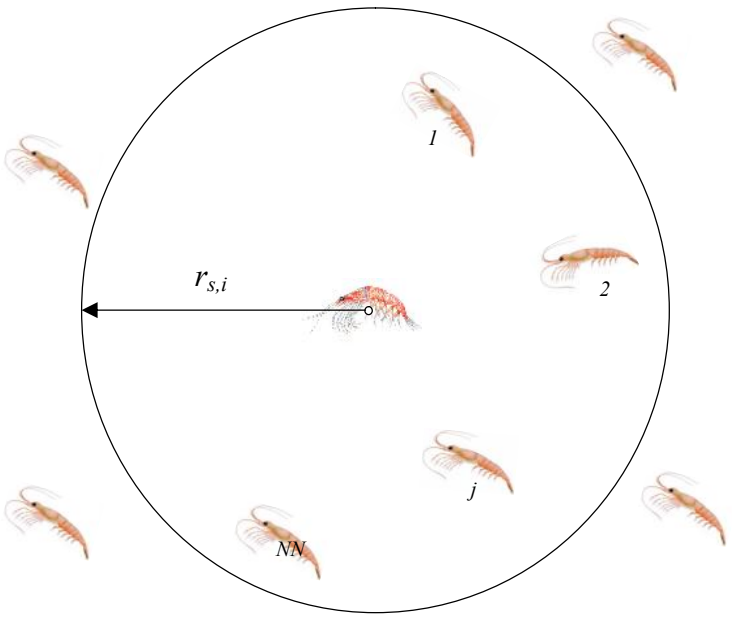
вредношћу функције циља K_{best} и $\hat{X}_{i,best}$ - се израчунава као и $\hat{X}_{ij}$, једначина (5.6), при чему се X_j замењује са позицијом крила који даје најбољу вредност функције циља, X_{best}.

Променљива *NN*, у једначини (5.4) представља, као што је већ и напоменуто, број суседних крила.

Десне стране једначина: (5.4), (5.6) и (5.7) садрже и векторске и нормализоване вредности функције циља. Вектори показују правац померања крила узрокован од стране суседног и свака вредност, у оквиру вектора, представља ефекат суседног крила, [114]. Вектор суседа може бити или привлачни или одбојни, док нормализована вредност може бити позитивна или негативна.

За избор суседних крила, могу се усвојити различите стратегије. На пример, коефицијент суседства се може дефинисати у смислу проналажења броја најближих крила. Узимајући у обзир понашање крила у природном окружењу: одржавање оптималне густине јата и потрага за пленом, може се дефинисати критично растојање $\boldsymbol{r_{s,i}}$, једначина (5.9), за сваког, *i*-тог, крила, које представља полупречник круга описаног око њега, у коме се могу пронаћи његови најутицајнији суседи, Слика 5.1, [114].

$$r_{s,i} = \frac{1}{5N}\sum_{j=1}^{NN}\left\|\mathbf{X}_i - \mathbf{X}_j\right\| \tag{5. 9}$$



Слика 5. 1. Шематски приказ растојања за дефинисање суседних крила

5.2.2. Потрага за храном, F_i

Потрага за храном је условљена са два параметра. Први параметар се односи на локацију плена, а други се односи на информацији о ранијим локацијама плена. Померање *i*-тог крила у потрази за храном је дато једначином (5.10), [114].

$$F_i = V_f \beta_i + \omega_f F_i^{staro} \tag{5. 10}$$

где је:

$\beta_i = \beta_i^{plena} + \beta_i^{best}$

V_f – брзина плена,

ω_f – коефицијент инерције кретања у потрази за храном, $\omega_f \in [0,1]$

F_i^{staro} - последње кретање у потрази за пленом,

β_i^{plena} - коефицијент који узима у обзир привлачност плена, и

β_i^{best} - утицај најбољег решења на кретање *i*-тог крила.

Утицај хране се дефинише у зависности од локације плена. Прво се налази центар масе плена, па се у зависности од тога одређује атрактивност, односно привлачност хране. У овом случају се врши само процена привлачности хране. У [25], виртуални центар концетрације плена, се процењује на основу дистрибуције вредности функције циља, што је у ствари инспирисано налажењем центра масе – тежишта. Виртуелни центар плена, за сваку итерацију, се рачуна по формули (5.11):

$$\mathbf{X}^{plena} = \frac{\sum_{i=1}^{N} \frac{1}{K_i} \mathbf{X}_i}{\sum_{i=1}^{N} \frac{1}{K_i}} \tag{5. 11}$$

тако да се привлачност хране, β_i^{plena}, за сваког крила у јату, може одредити по формули (5.12):

$$\beta_i^{plena} = C^{plena} \cdot \hat{K}_{i,plena} \cdot \hat{X}_{i,plena} \tag{5. 12}$$

где је C^{plena} – коефицијент плена, који се рачуна по једначини (5.13).

$$C^{plena} = 2 \cdot \left(1 - \frac{I}{I_{max}}\right) \tag{5. 13}$$

Привлачност хране, дефинисана једначином (5.12), подразумева могућност приближења јата крила глобалном оптимуму. Узимајући у обзир претходну констатацију, индивидуални крил се окупља у јато око глобалног минимума после извесног броја понављања. И баш та чињеница је уграђена у KH алгоритам ради побољшања глобалности алгоритма.

5.2.3. Физичка дифузија, D_i

Физичка дифузија индивидуалних крилова се посматра као случајни процес. Као таква, може се дефинисати на основу максималне брзине дифузије, D^{max} и случајног вектора правца, δ, који има случајну вредност из опсега [-1,1], једначина (5.14).

$$D_i = D^{max} \cdot \delta \tag{5. 14}$$

Максимална брзина дифузије, D^{max}, се такође усваја као случајан број, тако да је у [25], за опсег могућих вредности усвојено: $D^{max} \in [0.002, 0.010]$ m/s. Међутим, једначина (5.14) не гарантује да ће се смањивати кретање крила како се они позиционирају ка глобалном минимуму. Зато је, у цитираној литератури, модификована једначина (5.14) увођењем односа тренутне итерације I и максималног броја итерација I_{max}, једначина (5.15), како би се на тај начин смањивала вредност D_i како итеративни процес одмиче.

$$D_i = D^{max} \cdot \left(1 - \frac{I}{I_{max}}\right) \cdot \delta \tag{5. 15}$$

5.2.4. Процес кретања у KH алгоритму

Уопштено, кретање индивидуалних крила подразумева промену његове позиције ка најбољем решењу. Потрага за храном и кретање крила проузроковано кретањем суседних крила садржи две глобалне и две локалне стратегије. У складу са датим формулацијама поменутих кретања i-тог крила, ако је одговарајућа вредност функције циља сваког од претходно поменутих фактора (K_j, K^{best}, K^{plena} или K_i^{best}) је бољa (мања) од вредности функције циља посматраног крила, добија се привлачни ефекат, у супротном је одбојни ефекат. Из претходно реченог, види се да на кретање крила највећи утицај има најбоље решење. Сам вектор положаја, X_i, индивидуалног крила у интервалу од t до $t+\Delta t$, се може израчунати на основу једначине (5.16).

$$X_i(t+\Delta t) = X_i(t) + \Delta t \cdot \frac{dX_i}{dt} \tag{5. 16}$$

Може се приметити да је најважнија константа, за успешност KH алгоритма, Δt, тако да се при одређивању ове вредности, у току итеративног поступка тражења најбољег решења, мора бити посебно опрезан. Ово је из разлога јер Δt фактор увећања вектора брзине, као што се види из једначине (5.16). Константа Δt комплетно зависи од простора претраживања, тако да се њена вредност може израчунати на основу једначине (5.17).

$$\Delta t = C_t \sum_{j=1}^{NV} \left(\mathbf{UB}_j - \mathbf{LB}_j \right) \tag{5. 17}$$

где је:

NV број пројектних променљивих,

$\mathbf{UB}_j$ и $\mathbf{LB}_j$ горња и доња граница *j*-те променљиве ($j = 1,..,NV$)

C_t је емпиријски одређена константа и креће се у опсегу [0,2]. Што је вредност C_t мања појединачни крил боље претражује простор.

5.2.5. Генетски опеаратори - укрштање и мутација

У циљу побољшања перфомнаси KH алгоритма уграђен је генетски репродукциони механизам. Уведени адаптивни генетски механизми, укрштање и мутација, су инспирисани класичним DE алгоритмом.

Оператор укрштања је први пут употребљен у GA алгоритму. Векторизована верзија је такође коришћена у DE алгоритму. У оригиналном KH алгоритму, примењена је адаптивна векторизована шема укрштања, [25].

Укрштањем се управља преко вероватноће укрштања, C_r, једначина (5.18).

$$x_{i,m} = \begin{cases} x_{r,m} & rand_{i,m} < C_r \\ x_{r,m} & else \end{cases} \tag{5. 18}$$

где је:

$x_{i,m}$ *m*-та компонента од $\boldsymbol{X}_i$, односно $x_{r,m}$ *m*-та компонента од $\boldsymbol{X}_r$

$r \in \{1,2,3,...,i-1,i+1,...,N\}$

$C_r = 0.2 \cdot \hat{K}_{i,best}$, $\hat{K}_{i,best}$ - се израчунава на потпуно исти начин као и $\hat{K}_{ij}$, једначина (5.7)

На овај начин, вероватноћа укрштања је једнака 0 за глобално најбоље решење и повећава се са смањењем вредности функције циља.

Мутација је веома значајна у алгоритмима као што су ES и DE. Управљање мутацијом се изводи помоћу вероватноће мутације, M_u, тако да је у изворном KH алгоритму [25], примењена адаптивна мутациона шема представљена једначином (5.19).

$$x_{i,m} = \begin{cases} x_{gbest,m} + \mu\left(x_{p,m} - x_{q,m}\right) & rand_{i,m} < M_u \\ x_{r,m} & else \end{cases} \tag{5. 19}$$

где је:

$M_u = \dfrac{0.05}{\hat{K}_{i,best}}$, $\hat{K}_{i,best}$ - се израчунава на потпуно исти начин као и $\hat{K}_{ij}$, једначина (5.7)

$$p, q \in \{1,2,3,...,i-1,i+1,...,K\}$$

$$\mu \in [0,1]$$

На овај начин, вероватноћа мутације је једнака 0 за глобално најбоље решење и повећава се са смањењем вредности функције циља.

5.2.6. Методологија KH алгоритма

Уважавајући понашање арктичких крила приликом окупљања у јато, односно активности индивидуалних крила (кретање проузроковано кретањем суседних крила, кретање у потрази за храном и физичка дифузија) које су описане у претходним поглављима могуће је идеализовати карактеристике наведених кретања. Идеализација урађена у основном KH алгоритму, са уведеним бројем понављања, представљена је псеудокодом датим у алгоритму 5.1.

У Алгоритму 5.1., су:

$\mathbf{K}(x)$ - функција циља,

$\mathbf{X}(:, nr)$ - популација – позиција крила у nr-том понављању,

K_{best}, X_{best} - најбоље решење функције циља и одговарајућа позиција крила у nr-том понављању,

$\mathbf{X}_j$ - позиција суседног крила,

$\mathbf{X}(:,i)$, $\mathbf{K}(i)$ - позиција крила и израчуната вредност функције циља у i-тој итерацији.

Алгоритам 5. 1. KH алгоритам са NR бројем понављања

1: **begin**
2: Функција циља $\mathbf{K}(\boldsymbol{X})$, $\boldsymbol{X} = (x_1, x_2, ..., x_d)^T$
3: Дефинисање броја параметара који се оптимизираји NP
4: Дефинисање броја итерација MI $(i = 1, ..., MI)$
5: Дефинисање брзине плена V_f
6: Дефинисање броја крила NK $(nk = 1, ..., NK)$
7: Дефинисање граница пројектних променљивих %Горња **UB**, доња **LB** граница
8: Дефинисање максималне дифузије и максималне проузруковане брзине: && $D_{\max}$, $N_{\max}$
9: Дефинисање броја понављања: NR $(nr = 1, ..., NR)$
10: **while** $(nr < NR)$
11: **while** $j<NK$
12: Иницијализација $\mathbf{X}_j(:, nr)$
13: **end while**
14: Израчунавање функције циља за иницијалну популацију K_{best}, $\mathbf{X}_{best}$
15: **while** $(i < MI)$
16: % Израчунавање позиције на бази поменутих кретања: N_i, F_i, D_i
17: Позиција плена F_i
18: **while** $(j < NN)$
19: позиција суседног крила $\mathbf{X}_j$
20: **end while**
21: Дифузија D_i
22: % крај израчунавања позиције крила
23: Имплементација генетског оператора
24: Ажурирање позиције крила $\mathbf{X}(:,i)$ и израчунавање функције циља $\mathbf{K}(i)$
25: **if** $(\mathbf{K}(i) < K_{best})$
26: % Прихвати ново решење
27: $K_{best} = \mathbf{K}(i)$
28: $\mathbf{X}_{best}(nr) = \mathbf{X}(:,i)$
29: **end if**
30: **end while**
31: Рангирање крила и тражење тренутно најбољег $\mathbf{X}_{best}$
32: **end while**
33: Пострпоцесирање и приказивање резултата
34. **end begin**

5.3. МОДИФИКОВАНИ КН АЛГОРИТАМ - МКН

У циљу побољшања перфоманси стандардног КН алгоритма, извршена је модификација, да би се добили ефективни резултати у синтези четворочланог механизама. Наиме, анализирајући идеализовано понашање крила, кроз њихове три главне активности (померање проузроковано кретањем других крила, потрага за пленом и случајна дифузија), закључује се да је циљ, виртуелних крила, да имају оптималну густину и позицију у односу на плен. Када се покрене итеративни процес, једино се иницијализује позиција крила, док се позиција хране не иницијализује, већ се израчунава, линија 17, Алгоритам 5.1.

Прва модификација, видети Алгоритам 5.2, линије 13-21, [165], што доводи до ефикасног КН алгоритма, је иницијализација локације хране. Наиме, после генерисања почетне позиције крила *X*, израчунава се функција циља *K*, за сваког крила, од 1,..., *NK*, па се затим тражи рачић односно позиција *A*, $A \in [1,\mathrm{NK}]$, са најмањом вредношћу функције циља, K_{min}^{init}. Ови прорачуни се изводе на почетку сваког корака понављања *nr*=1,...,*NR*, а пре итеративног поступка проналажења оптималног решења. Када почне *i*-та итерација, *i*=1,...,*MI*, иницијализује се позиција плена (food) *F*, линија 15, Алгоритам 5.2 . За проналажење виртуалног центра концентрације хране коришћен је принцип центра маса, [114], где се узима у обзир вредност функције циља за сваког од крила, односно вредности пројектних променљивих за сваког од њих, једначине (5.20)-(5.21).

$$F = \frac{\mathbf{X}_f}{\sum_{nk=1}^{NK} \frac{1}{\mathbf{K}(nk)}} \tag{5. 20}$$

где је:

$$\mathbf{X}_f = \sum_{np=1}^{NP} \frac{\mathbf{X}(np)}{\mathbf{K}} \tag{5. 21}$$

Када се израчуна позиција плена (food), **F**, проверава се да ли се израчунате вредности налазе у границама пројектних променљивих, **LB≤F≤UB**. Уколико се вредности **F** не налазе у датим границама, нове вредности се рачунају по једначини (5.22).

После иницијализације локације хране, линија 15, Алгоритам 5.2., рачуна се вредност функције циља у зависности од локације хране, F_i. Затим се проналази најбоља вредност функције циља K_f^{best}, која одговара локацији хране, F_{best}.

$$\mathbf{F} = \begin{cases} \mathbf{UB} + rand \cdot \left(K_{\min}^{init} - \mathbf{UB} \right), \mathbf{F} > \mathbf{UB} \\ \mathbf{LB} + rand \cdot \left(K_{\min}^{init} - \mathbf{LB} \right), \mathbf{F} < \mathbf{LB} \end{cases} \quad (5.\ 22)$$

Ако је вредност функције циља K_f^{best}, добијене на основу вредности $\mathbf{F}_{best}$, боља, односно мања од вредности функције циља K_{best}, израчунате на бази позиције крила $X(:,nr)$, тада се најбоља позиција крила, $\mathbf{X}_{best}$, замењује са најбољом локацијом хране, $\mathbf{F}_{best}$. Иницијалне вредности се регрупишу, пре почетка итеративног процеса тражења оптимума, како би се задовољила оба услова: и оптимална густина и најбоља позиција хране.

Следећа модификација се односила на замену генетског оператора. У оригиналном алгоритму, Гандоми и Алави у [114], користе оператор укрштања за проналажење позиције вредности функција циља. Уместо поменутог генетског оператора, уводи се комбиновање позиција вредности функција циља (линије 29-41 у алгоритму 5.2.), [165]. Наиме, после израчунавања вредности функција циља, на бази поменутих активности крила (померање проузроковано кретањем других крила, потрага за пленом и случајна дифузија), отпочиње процес случајног комбиновања позиција вредности функције циља. Овом модификацијом, новонастали, низ решења се коригује за вредност физичке дифузије, D_i и са таквим решењима, се израчунава вредност функције циља, да би се у истој итерацији поново претражио простор решења, у циљу побољшања пронађеног оптимума. На овај начин, простор локалног минимума је ефикасно избегнут.

За разлику од предложене формуле за израчунавање физичке дифузије, D_i, једначина (5.15), у модификованом КН алгоритму ова вредност се израчунава по предложеној једначини (5.23), као функција претходне вредности функције циља, K_i, најбоље вредности функције циља K_{best}, најгоре вредности функције циља, K_{worst} и случајног вектора, $rand \in [-1,1]$. Једначина (5.23) је последица великог броја експериментисања током процеса истраживања примене КН алгоритма на проблеме синтезе механизама, где се узимало у обзир да је физичка дифузија случајна вредност.

$$D_i = D_{max} \cdot (1 - \frac{i}{MI}) \cdot floor(rand + \frac{K_i - K_{best}}{K_{worst}}) \quad (5.\ 23)$$

где је:

MI – укупан број итерација

i – та итерација, i=1,..,MI;

$rand \in [-1,1]$

floor – заокружена негативна приближна вредност

D_{max} – максимална брзина дифузије, параметар КН алгоритма.

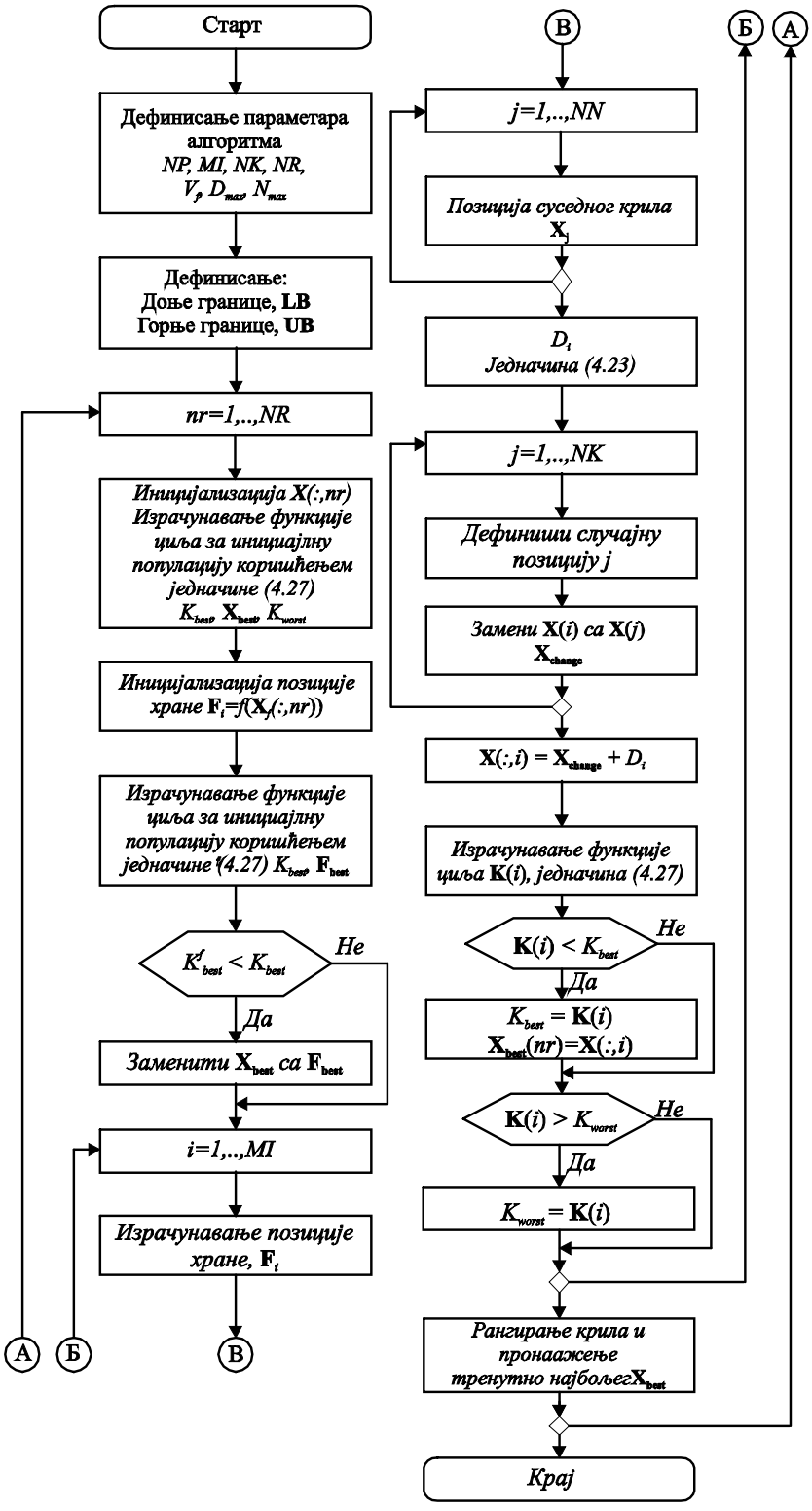
Ове модификације су резултирале добијањем модификованог КН алгоритма, МКН, који је представљен у алгоритму 5.2. Ради бољег разумевања, дат је и дијаграм тока новонасталог алгоритма, Слика 5.2.

Алгоритам 5. 2. МКН

1: **begin**
2: Дефинисње броја параметара који се оптимизирају NP
3: Дефинисање броја итерација $MI\ (i = 1,...,MI)$
4: Дефинисање брзине хране V_f
5: Дефинисање броја крила $NK\ (nk = 1,...,NK)$
6: Дефинисање броја пројектних параметара % Горња **UB**, доња **LB** граница
7: Дефинисање максималне дифузије: $D_{\max}$,
8. Дефинисање максималне индуковане брзине: $N_{\max}$,
9: Дефинисање броја понављања: $NR\ (nr = 1,...,NR)$
10: **while** ($nr < NR$)
11: *Иницијализација* $\mathbf{X}(:,nr)$
12: *Израчунавање функције циља за иницијалну популацију* K_{best} , $\mathbf{X}_{\mathbf{best}}$, K_{worst} *%% једначина (5.27)*
13: ***% 1. модификација***
14: *% Иницијализација хране*
15: *Позиција хране* $\mathbf{F}_i = f(\mathbf{X}_f(:,nr))$
16: *% Завршетак иницијализације хране*
17: *Израчунавање вредности функције циља иницијалне популације хране:* $K_{best}^f, \mathbf{F}_{\mathbf{best}}$ *%% једначина (5.27)*
18: **if** $K_{best}^f < K_{best}$
19: *замени* $\mathbf{X}_{\mathbf{best}}$ *са* $\mathbf{F}_{\mathbf{best}}$
20: ***end if***
21: ***% Завршетак 1. модификације***
22: **while** ($i < MI$)
23: *% Израчунавање кретања*
24: *Позиција хране* $\mathbf{F}_i$
25: **while** ($j < NN$)
26: *позиција суседног крила* $\mathbf{X}_j$
27: **end while**
28: % *Завршетак прорачуна кретања*
29: ***% 2. modification***
30: *% Дифузија %% једначина (5.23)*
31: D_i
32: *% комбиновање позиција крила*
33: **while** ($j < NK$)
34: *дефиниши случајну позицију j*

35: *замени* $\mathbf{X}(i)$ *with* $\mathbf{X}(j) \rightarrow \mathbf{X}_{\mathbf{change}}$
36: **end while**
37: *% Завршетак комбиновања позиција крила*
38: *% Ажурирање позиције* **X**
39: $\mathbf{X} = \mathbf{X}_{\mathbf{change}} + D_i$
40: *% Завршетак ажурирање позиције* **X**
41: ***% Крај 2. модификације***
42: *Израчунавање функције циља* $\mathbf{K}(i)$ *%% једначина (5.27)*
43: **if** ($\mathbf{K}(i) < K_{best}$)
44: *%Прихвати ново решење*:
45: $K_{best} = \mathbf{K}(i)$
46: $\mathbf{X}_{\mathbf{best}}(nr) = \mathbf{X}(:,i)$
47: **end if**
48: **if** ($\mathbf{K}(i) > K_{worst}$)
49: $K_{worst} = \mathbf{K}(i)$
50: **end if**
51: **end while**
52: *Рангирати крил и пронаћи тренутно најбоље решење*
53: **end while**
54: *Постпроцесирање резултата и визуализација*
55: **end begin**

У примерима који следе, у којима је МКН алгоритам тестиран, параметри алгоритма су: брзина хране – V_f, максимална брзина дифузије – D_{max}, и максимална индукована брзина N_{max}, су усвојени из цитиране литературе, као и на основу великог броја изведених експеримената. Усвојене вредности параметара алгоритма могу да варирају од случаја до случаја, тако да су за представљене примере усвојене вредности: , V_f =0.02, D_{ma}=0.005 і N_{max} =0.01. Остале вредности, као што је број пројектних променљивих-*NP*, број итерација–*MI*, број крила–*NK* и број понављања у итеративном процесу–*NR*, се разликују од примера до примера у зависности од природе путање посматраног механизма.



Слика 5. 2. Дијаграм тока МКН алгоритма

5.4. ФОРМУЛАЦИЈА ПРОБЛЕМА ЗГЛОБНОГ ЧЕТВОРОУГАОНОГ МЕХАНИЗМА

5.4.1. Једначине положаја

Слика 5.3. представља све геометријске параметре зглобног четвороугаоног механизма, где је тачка на спојки, *C*, тачка механизма која треба да прође кроз задате тачке путање. Мала слова означавају одговарајуће дужине чланова механизма, а углови су означени грчким словима:

a – дужина кривање,

b – дужина спојке,

c – дужина шеталице,

d – дужина постоља,

e и *f* – представљају пројекцију тачке *C* на правац спојке, односно правац који је нормалан на спојку,

x_O и y_O – координате непокретне тачке *O* на кривању у односу на глобални координатни систем xO_1y,

θ_2 – угао кривање у односу на релативни координатни систем x_rOy_r,

θ_3 – угао кривање у односу на релативни координатни систем x_rOy_r,

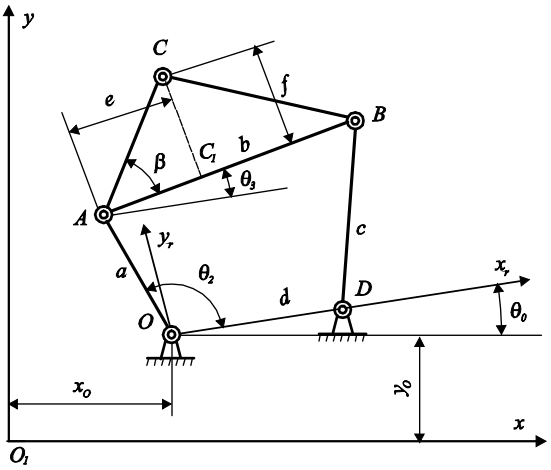
θ_0 – угао формиран између постоља и координатног система xO_1y.

Углови , θ_2 и θ_3, су израчунати на основу Freudenstein-ових једначина [121]. Позиција тачке C, у односу на глобални координатни систем xO_1y, може се представити као:

$$\begin{bmatrix} C_x \\ C_y \end{bmatrix} = \begin{bmatrix} cos\,\theta_0 & -sin\,\theta_0 \\ sin\,\theta_0 & cos\,\theta_0 \end{bmatrix} \begin{bmatrix} C_{xr} \\ C_{yr} \end{bmatrix} + \begin{bmatrix} x_o \\ y_o \end{bmatrix}, \tag{5. 24}$$

где је

$$\begin{aligned} C_{xr} &= a \cdot cos\,\theta_2 + e \cdot cos\,\theta_3 - f \cdot sin\,\theta_3 \\ C_{yr} &= a \cdot sin\,\theta_2 + e \cdot sin\,\theta_3 + f \cdot cos\,\theta_3 \end{aligned}. \tag{5. 25}$$



Слика 5. 3. Четворочлани механизам у глобалном координатном систему

5.4.2. Пројектни параметри

У свим примерима синтезе механизама, као генератора путање са прописаним временом, који се даље разматрају, следећих девет променљивих се оптимизирају: a,b,c,d,e,f,x_o,y_o и θ_o. У случају синтезе механизама као генератора путање без прописаног времена, улазни углови кривање θ_2^i (i = 1,..,N), који одговарају датим тачкама на путањи, се такође оптимизирају. У општем случају, вектор пројектних променљивих је дат као:

$$\mathbf{X} = [a,b,c,d,e,f,x_o,y_o,\theta_o,\theta_2^1,\theta_2^2,...,\theta_2^N], \tag{5. 26}$$

где је N број датих тачака.

5.4.3. Функција циља и ограничења

Fункција циља је подељена у два дела. Први део се односи на грешку положаја између задатих тачака и актуелних (генерисаних) тачака које описује тачка C на спојки четворочланог механизма током њеног кретања. Други део функције циља се односи на ограничења.

За дефинисање оптимизационог проблема користе се два ограничења, у којима се уводе казнене функције, тако да имамо:

$$\min\left\{\sum_{i=1}^{N}\left[\left(C_{xd}^i - C_x^i\right)^2 + \left(C_{yd}^i - C_y^i\right)^2\right] + M_1 h_1(\mathbf{X}) + M_2 h_2(\mathbf{X})\right\}, \tag{5. 27}$$

са границама:

$$x_j \in [x_j^{lb}, x_j^{ub}],\ \forall x \in \mathbf{X}, j = 1,...,NP.$$

где:

N – број тражених циљних тачака,

$\left(C_{xd}^{i} - C_{x}^{i}\right)$ – координате задатих тачака у односу на глобални координатни систем,

$\left(C_{yd}^{i} - C_{y}^{i}\right)$ – координате које генерише тачка C на спојки,

$h_1(\mathbf{X})$ – односи се на услове Grashof-a. $h_1(\mathbf{X})$=0, показује да је услов Grashof-a тачан, $h_1(\mathbf{X})$=1, показује је услов Grashof-a нетачан.

$h_2(\mathbf{X})$ – односи се на улазни угао кривање. Улазни углови, θ_2^i (i=1,..,N) су поређани од највеће до најмање или од најмање до највеће вредности. $h_2(X)$=0, значи да су секвенцијални услови за θ_2^i тачни и $h_2(\mathbf{X})$=1, значи да су секвенцијални услови за θ_2^i нетачни. M_1 и M_2 су највише вредности које кажњавају функцију циља, уколико се не задовоље ограничења. Свака пројектна променљива x_j, мора бити дефинисана са њеном најмањом, x_j^{lb}, односно највећом, x_j^{ub}, граничном вредности, при чему је NP број пројектних променљивих. Ограничења која се односе на пројектне променљиве се директно дају у МКН алгоритму, чиме се омогућава да те променљиве буду у оквиру дефинисаних граница током итеративног процеса.

5.5. ПРИМЕРИ

У овом поглављу, разматрају се познати примери синтезе четворочланих механизама коришћењем модификованог КН алгоритма, МКН. У сваком примеру, проблем синтезе као генератора путање се дефинише преко скупа датих тачака на путањи. Узети су у обзир проблеми са и без прописаног времена.

Параметри МКН алгоритма у свим примерима су: NP – број параметара који се оптимизирају, MI – максималан број итерација, NK – број крила, NR – број понављања у свакој итерацији, V_f – брзина хране (плена), N_{max} – максимална индукована брзина, D_{max} – максимална дифузија.

5.5.1. Пример 1

У овом примеру се разматра проблем синтезе четворочланог механизма, као генератора путање, без прописаног времена, где тачка спојке мора да прође кроз шест задатих тачака које леже на верикалној праволинијској путањи.

Пројектне промењљиве су:

$$\mathbf{X} = [a, b, c, d, e, f, x_o, y_o, \theta_o, \theta_2^1, \theta_2^2, \theta_2^3, \theta_2^4, \theta_2^5, \theta_2^6].$$

Жељене, односно задате, тачке путање су:

$$\{C_d^i\} = \{(20,20); (20,25); (20,30); (20,35); (20,40); (20,45)\},\ i = 1,..,6.$$

Границе пројектних промењљивих:

$0 \le a,b,c,d \le 60$; $-60 \le e,f,x_o,y_o \le 60$; $0 \le \theta_2^1, \theta_2^2, \theta_2^3, \theta_2^4, \theta_2^5, \theta_2^6 \le 2\pi$.

Параметри алгоритма NP=15, MI=1000, NK=25, NR=50, V_f=0.02, N_{max}=0.01, D_{max} =0.005, док су вредности пројектних промењљивих механизма, као и добијених грешака, дати у Табели 5.1.

Табела 5. 1. Упоредни резултати за пример 1.

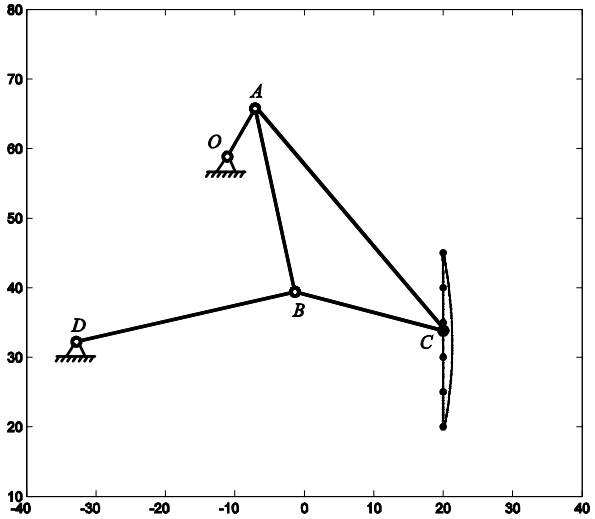
	Cabrera и остали [99] (GA)	Acharyya и Mandal [112] (EAs)	Cabrera и остали [103] (MUMSA)	Ortiz и остали [104] (IOA)	**МКН алгоритам**
a	8.562912	6.404196	8.2046468	18.730993	**7.97727842**
b	19.09486	31.60722	24.932131	31.223101	**26.95316363**
c	47.83886	50.59949	31.385926	42.223736	**32.28634466**
d	39.46629	35.02074	31.788264	54.715815	**34.36457801**
e	13.38556	20.80324	34.193719	-27.29874	**36.96182480**
f	12.21961	41.54364	14.415668	31.650513	**19.66792583**
x_o	29.7225	60.00000	-6.366519	43.070863	**-11.09433942**
y_o	23.4545	18.07791	56.83676	27.417061	**58.82813230**
θ_0	6.20163	0.00000	4.015959	5.977455	**4.02771413**
θ_2^1	6.11937	6.283185	1.366547	6.424111	**1.50348386**
θ_2^2	0.19304	0.264935	2.330773	6.534955	**2.38565679**
θ_2^3	0.44083	0.500377	2.871039	0.362302	**2.91751288**
θ_2^4	0.68467	0.735321	3.394591	0.469063	**3.42256857**
θ_2^5	0.95835	0.996529	3.97096	0.577652	**3.96922588**
θ_2^6	1.35533	1.333549	4.96349	0.690469	**5.18614585**
Грешка	0.035142	0.0178414	2.057e-0004	2.3712e-004	**2.3667e-005**

У Табели 5.2., дате су вредности датих тачака, као и тачака које генерише тачка С на спојки приликом кретања дуж задате путање.

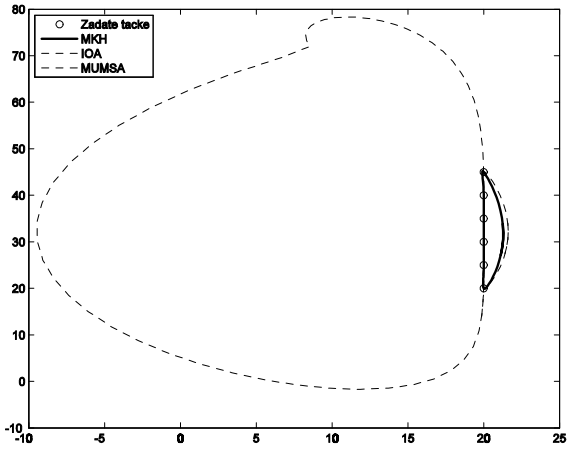
Табела 5. 2. Пример 1 – Грешка – 2.3667e-005

i	1	2	3	4	5	6
C_{xd}^i	20	20	20	20	20	20
C_x^i	20.000932	20.000001	20.000002	19.999996	20.000003	20.000004
C_{yd}^i	20	25	30	35	40	45
C_y^i	19.995225	24.999998	30.000003	35.000002	39.999996	45.000000

Најбољи механизам за Пример1., као и путања коју он генерише, дати су на Слици 5.5. Слика 5.6., представља путању тачке C, добијеног механизма, као и путање механизама добијених од других аутора (Cabrera и остали [103] – MUMSA алгоритам и Ortiz и остали [104] – Оптимизациони алгоритам са самоприлагодљивом техником IOA).



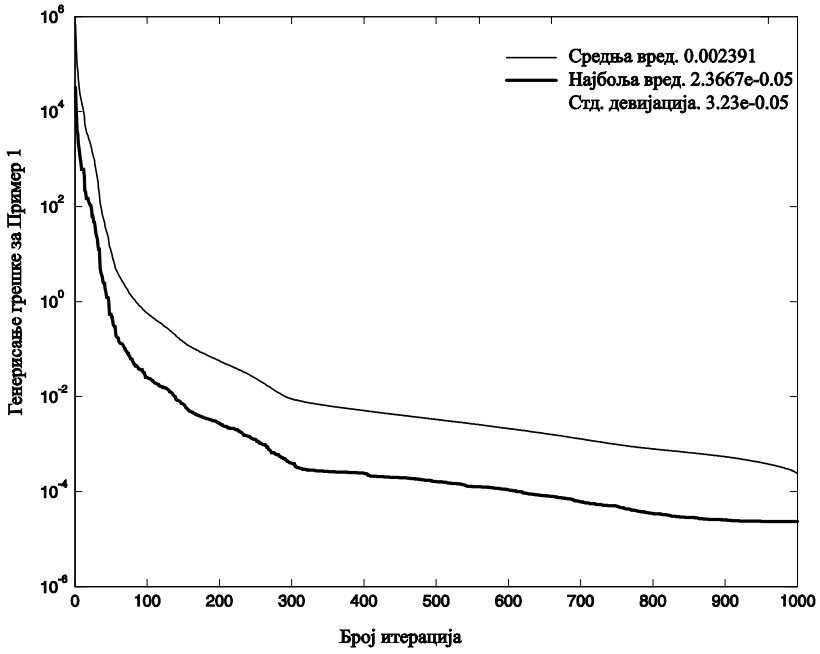
Слика 5. 4. Најбоље добијени механизам у примеру 1.



Слика 5. 5. Путања спојке.

Статистички резултати оптималне синтезе су приказани на Слици 5.6: најбоља вредност, просечна вредност, као и стандардна девијација. У процесу синтезе, коришћењем МКН алгоритма, број понављања у свакој итерацији је био

NR=50, а максимални број итерација је био *MI*=1000. Ова слика и све слике које показују развој грешке, приказане су у логаритамској скали за Y осу.



Слика 5. 6. Најбоља и средња вредност грешке у Примеру 1.

5.5.2. Пример 2

Овај пример разматра проблем синтезе четворочланог механизма, као генератора путање, са прописаним временом, где тачка спојке мора да прође кроз осамнаест датих тачака.

Пројектне промењљиве су:

$$\mathbf{X} = [a, b, c, d, e, f, x_o, y_o, \theta_o, \theta_2^1].$$

Жељене, односно задате, тачке путање су:

$$\left\{C_d^i\right\} = \left\{\begin{array}{l}(0.5,1.1);(0.4,1.1);(0.3,1.1);(0.2,1.0);(0.1,0.9);(0.05,0.75);\\(0.02,0.6);(0,0.5);(0,0.4);(0.03,0.3);(0.1,0.25);(0.15,0.2);\\(0.2,0.3);(0.3,0.4);(0.4,0.5);(0.5,0.7);(0.6,0.9);(0.6,1.0)\end{array}\right\}$$

$$\left\{\theta_2^i\right\} = \left\{\theta_2^1, \theta_2^1 + 20 \cdot i\right\},\ i = 1,...,17$$

Границе пројектних промењљивих:

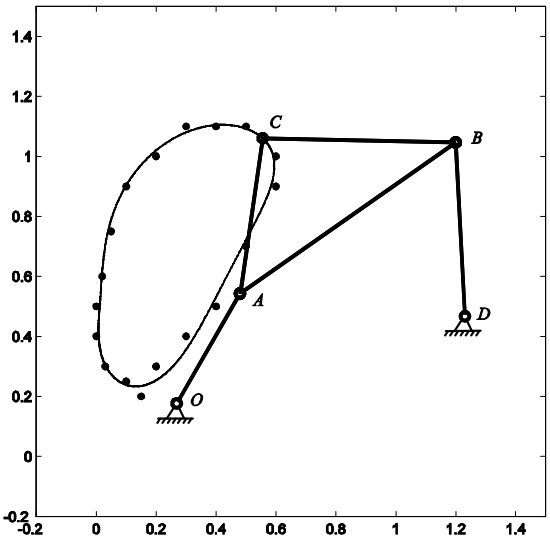
$0 \le a,b,c,d \le 50$; $-50 \le e,f,x_o,y_o \le 50$; $0 \le \theta_0, \theta_2^1 \le 2\pi$.

Параметри алгоритма NP=10, MI=50, NK=25, NR=50, V_f =0.02, N_{max}=0.01, D_{max}=0.005.

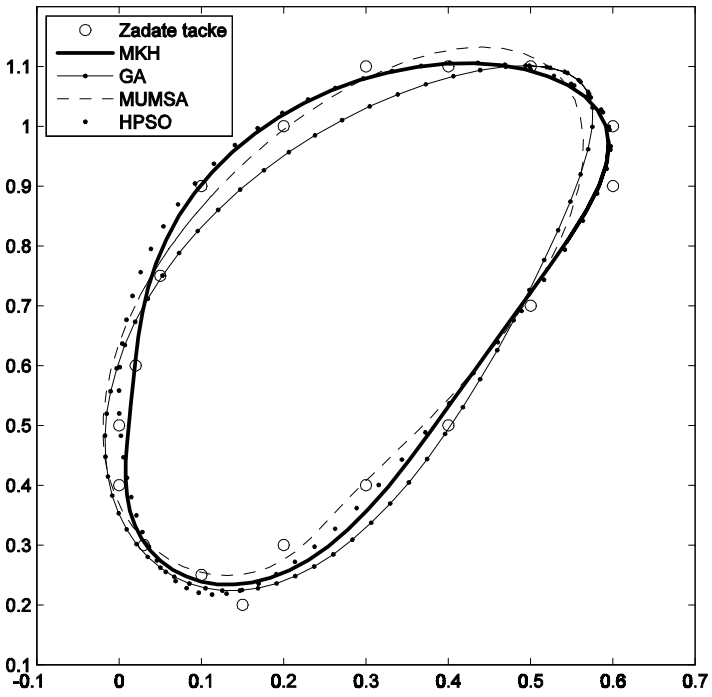
Вредности пројектних променљивих механизма, као и добијених грешака, дати су у Табели 5.3. У истој табели су дати резултати, за овај пример, од других аутора.

Табела 5. 3. Упоредни резултати за пример 2.

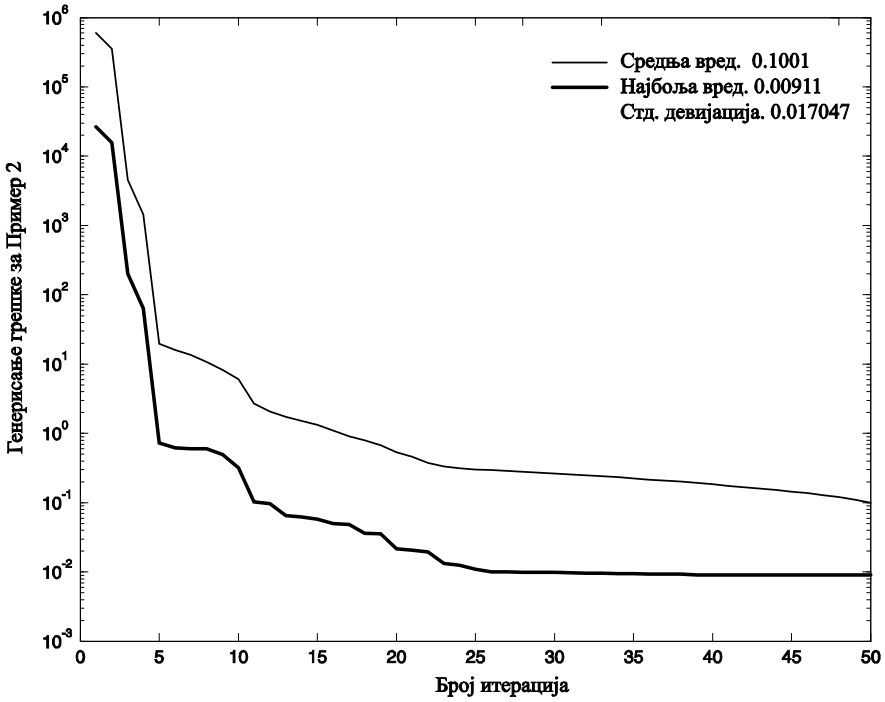
	Kunjur и Krishnamurt [98] (GA)	Ortiz и остали [104] (IOA)	Cabrera и остали [99] (GA)	Cabrera и остали [103] (MUMSA)	Lee и Lee [111] (HPSO)	**МКН алгоритам**
a	0.274853	0.245216	0.237803	0.297057	0.3700	**0.42180**
b	1.180253	6.38294	4.828954	3.913095	2.9048	**0.87821**
c	2.138209	2.620532	2.056456	0.849372	0.5000	**0.58013**
d	1.879660	4.040435	3.057878	4.453772	2.8500	**1.00429**
e	-0.833592	1.139106	0.767038	1.6610626	1.0205	**0.35907**
f	-0.378770	1.866109	1.850828	2.7387359	1.6894	**0.38081**
x_o	1.132062	1.891805	1.776808	2.806964	0.9400	**0.26886**
y_o	0.663433	-0.761339	-0.641991	4.853543	-1.1712	**0.17715**
θ_0	4.354224	1.187751	1.002168	-1.309243	0.7600	**0.29294**
θ_2^1	2.558625	0.000000	0.226186	4.853543	0.5134	**0.88595**
Грешка	0.043	0.0349	0.0337	0.0196	0.0110	**0.00911**



Слика 5. 7. Најбоље добијени механизам у примеру 2.



Слика 5. 8. Путања спојке



Слика 5. 9. Најбоља и средња вредност грешке у Примеру 2.

Слика 5.7, представља најбоље генерисани механизам применом MKH алгоритма за Пример 2, док Слика 5.8, представља путању тачке *C* добијеног механизма, као и путање механизама од других аутора за исти пример (Cabrera и остали [103] – MUMSA алгоритам, Cabrera и остали. [99] – Генетски алгоритам - GA, Lee и Lee [111] – Хибридна оптимизација ројем честица - HPSO). У овом примеру, број понављања у свакој итерацији је био *NR*=50, а максимални број итерација је био *MI*=50, а статистички резултати дати су на Слици 5.9.

5.5.3. Пример 3

У овом примеру разматра се проблем синтезе четворочланог механизма, као генератора путање, где тачка спојке мора да прође кроз дванаест тачака, правећи путању у облику броја осам.

Пројектне промењљиве су: $\mathbf{X} = [a, b, c, d, e, f, x_o, y_o, \theta_o, \theta_2^1]$.

Жељене, односно задате, тачке путање су:

$$\{C_d^i\} = \begin{Bmatrix} (4.15,2.21); (4.50,2.18); (4.53,1.83); (4.13,1.68); (3.67,1.58); (2.96,1.33); \\ (2.67,1.06); (2.63,0.82); (2.92,0.81); (3.23,1.07); (3.49,1.45); (3.76,1.87) \end{Bmatrix}$$

Границе пројектних промењљивих:

$0 \le a,b,c,d \le 50$; $-50 \le e,f \le 50$; $-10 \le x_o, y_o \le 10$; $0 \le \theta_0, \theta_2^1 \le 2\pi$, i=1,...,12.

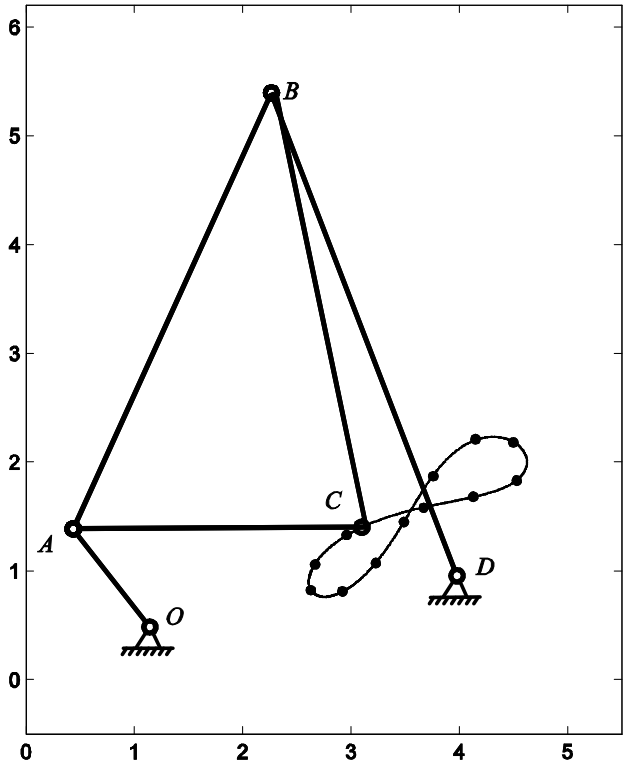
Параметри алгоритма NP=10, MI=1500, NK=25, NR=50, V_f=0.02, N_{max}=0.01, D_{max}=0.005.

Вредности пројектних промењљивих механизма, као и добијених грешака, дати су у Табели 5.4. Ради упоредне анализе добијених резултата са резултатима других аутора, у истој табели су приказани резултати од аутора Lee и Lee [111] (PSO – Оптимизација појем честица и HPSO – Хибридна оптимизација појем честица).

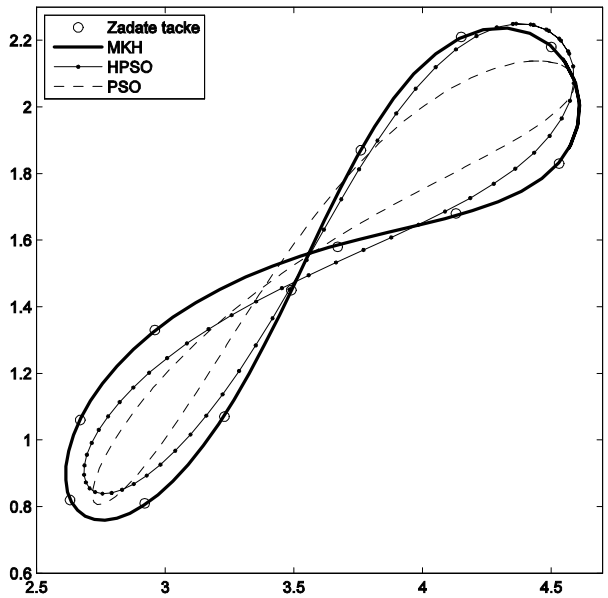
Табела 5. 4. Упоредни резултати за пример 3.

	Lee и Lee [111] (PSO)	Lee и Lee [111] (HPSO)	**МКН алгоритам**
a	1.1013	1.1133	**1.14644**
b	3.9558	14.7381	**4.40713**
c	3.9330	16.8017	**4.75773**
d	4.5503	4.5359	**2.87645**
e	3.3824	0.1958	**1.12238**
f	-2.0237	-3.9370	**-2.41886**
x_o	0.0000	0.0000	**1.14265**
y_o	0.0000	0.0000	**0.48237**
θ_0	0.0000	0.0000	**0.16502**
θ_2^1	-0.2014	-0.1816	**-0.31399**
Грешка	0.1716	0.0964	**1.5971e-004**

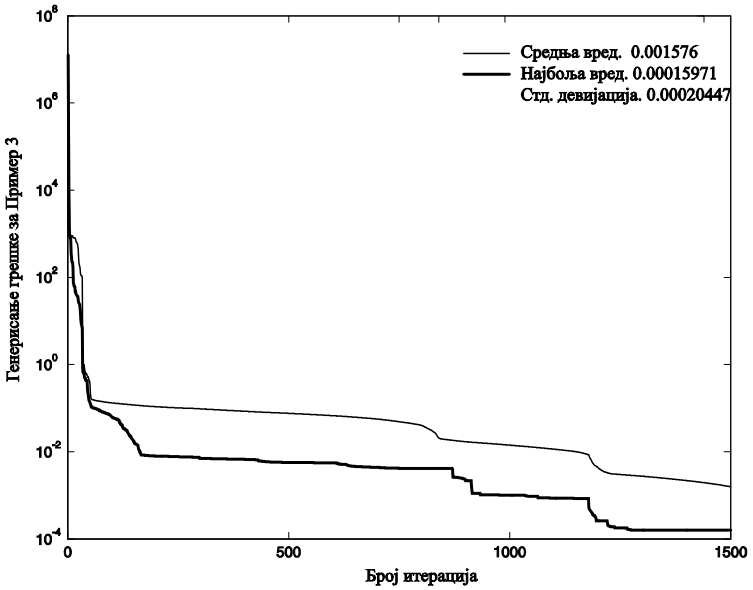
На Слици 5.10, приказан је механизам, генерисан применом МКН алгоритма, који даје најбоље резултате са путањом коју генерише. Слика 5.11, приказује криве коју генерише спојка применом МКН, PSO и HPSO алгоритма. Са слике се може приметити да МКН алгоритам генерише значајно мању грешку и самим тим даје боље резултате у односу на PSO и HPSO. У овом примеру, број понављања у свакој итерацији је био *NR*=50, а максимални број итерација је био *MI*=1500, а статистички резултати грешке су дати на Слици 5.12.



Слика 5. 10. Најбоље добијени механизам у примеру 3.



Слика 5. 11. Путања спојке



Слика 5. 12. Најбоља и средња вредност грешке у Примеру 3.

5.5.4. Пример 4

Овај пример разматра проблем механизма који генерише путању са 25 тачака, предложен од стране McGarva [122], као трајекторија дела машине за паковање. Такође се бави проблемима синтезе као генератора путање четворочланог механизма.

Пројектне промењљиве су: $\mathbf{X} = [a, b, c, d, e, f, x_o, y_o, \theta_o]$.

Жељене, односно задате, тачке путање су:

$$\left\{C_d^i\right\} = \left\{\begin{array}{l} (7.03,5.99);(6.95,5.45);(6.77,5.03);(6.40,4.60);(5.91,4.03); \\ (5.43,3.56);(4.93,2.94);(4.67,2.60);(4.38,2.20);(4.04,1.67); \\ (3.76,1.22);(3.76,1.97);(3.76,2.78);(3.76,3.56);(3.76,4.34); \\ (3.76,4.91);(3.76,5.47);(3.80,5.98);(4.07,6.40);(4.53,6.75); \\ (5.07,6.85);(5.05,6.84);(5.89,6.83);(6.41,6.80);(6.92,6.58) \end{array}\right\}$$

Границе пројектних промењљивих:

$0.1 \le a \le 5$; $1 \le b,c,d \le 10$; $1 \le e \le 14$; $1 \le f \le 12$; $-5 \le x_o, y_o \le 5$; $-\pi/2 \le \theta_0 \le \pi/2$, $0 \le \theta_2^i \le 2\pi$, i=1,...,25.

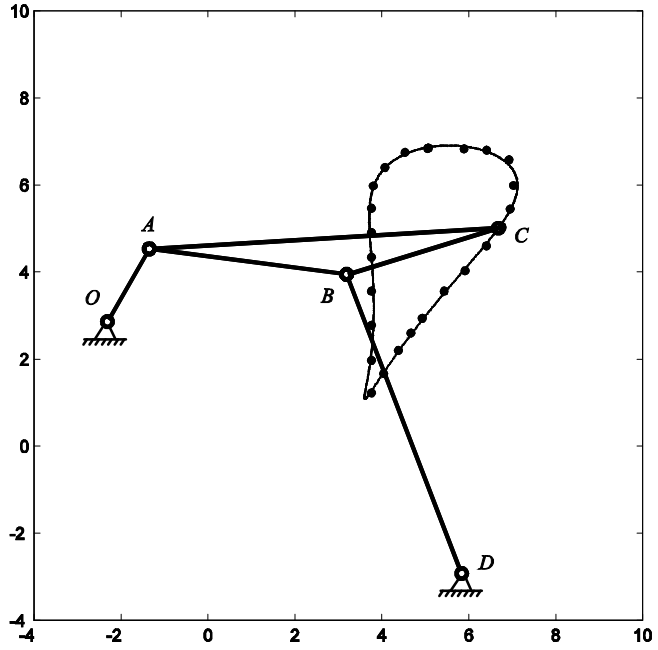
Параметри алгоритма NP=9, MI=1500, NK=25, NR=50, V_f=0.02, N_{max}=0.01, D_{max}=0.005.

Вредности пројектних променљивих, као и добијена грешка, дати су у Табели 5.5. Иста табела приказује геометријске параметре механизма из радова: Laribi и остали [100] – генетски алгоритам – метод фази логике, GA-FL и Smaili и Diab [94] – Мрављи – градијентни метод претраживања, Ant – gradient.

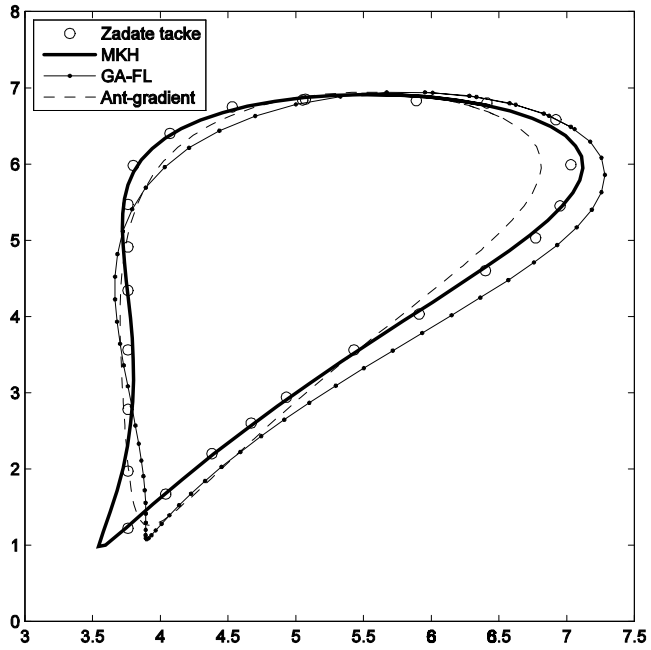
Табела 5. 5. Упоредни резултати за пример 5.

	Laribi и остали [100] (GA-FL)	Smaili и Diab [94] (Ant-gradient)	**МКН алгоритам**
a	3.01	1.89	**1.93027**
b	8.8	8.41	**4.57242**
c	8.8	6.75	**7.36674**
d	9	13.08	**9.99432**
e	8.626	14.177	**7.90168**
f	-6.985	2.794	**1.49860**
x_o	-2.4	-8.77	**-2.31301**
y_o	-4	1.20	**2.86189**
θ_0	0.489	-0.3815	**-0.61763**
Грешка	0.9022	0.5504	**0.03916**

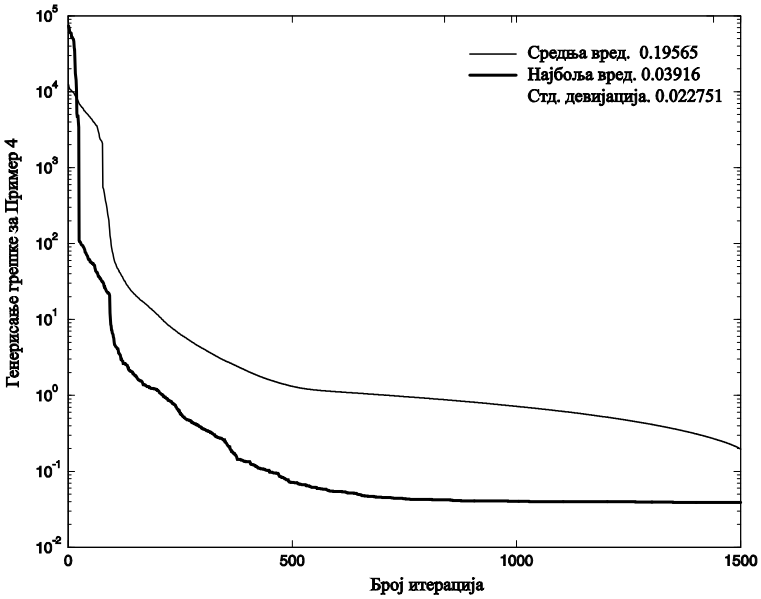
Слика 5.13, приказује најбољи генерисани механизам за овај пример, док на Слици 5.14 су дате путање добијене применом алгоритма МКН, GA – FL и Ant-Gradient. Статистички резултати генерисане грешке за овај пример су дати на Слици 5.15. Ови резултати су добијени са следећим параметрима: број понављања у свакој итерацији је био *NR*=50, а максимални број итерација је био *MI*=1500.



Слика 5. 13. Најбоље добијени механизам у примеру 4.



Слика 5. 14. Путања спојке



Слика 5. 15. Најбоља и средња вредност грешке у Примеру 4.

5.6. ЗАКЉУЧНЕ НАПОМЕНЕ УЗ ПЕТО ПОГЛАВЉЕ

МКН алгоритам је добијен модификацијама стандардног КН алгоритма у циљу добијања оптималних решења у димензионалној синтези четворочланих механизама као генератора путање.

Модификације су се односиле на:

- Иницијално одређивање позиције плена пре почетка итеративног процеса. Наиме, стандардни КН алгоритам подразумева иницијално одређивање позиције крила у јату узимајући у обзир оптималну густину. Иницијализацијом плена добија се почетно позиционирање крила и у односу на плен. Другим речима, пре почетка итеративног процеса, проверавају се позиције појединачних крила у односу на плен и упоређењем позиција, кроз функцију циља, формира се иницијално оптимално решење са којим се улази у итеративни поступак.
- Замену генетског оператора укрштања са случајним комбиновањем добијених колона који су добијени у једној итерацији. Пошто је КН алгоритам заснован на трима случајним активностима (кретање изазвано другим појединачним криловима, кретање при потрази за храном и случајна дифузија), постоји опасност да једно од кретања "поквари" позиције крила у јату и да решење одведе у правцу локалног минимума. Да би се ова опасност избегла уведена је поменута модификација случајног кобиновања колона вектора решења. Након одређивања нових вектора решења (комбиновањем колона) новонастали редослед крила, коригује се за вредност физичке дифузије, чиме се у истој итерацији финије претражује простор решења у циљу побољшања нађеног оптимума.

МКН акгоритам је тестиран на четири референтна примера из синтезе зглобног четвороугаоног механизма. Резултати добијени овим алгоритмом значајно су бољи у поређењу са резултатима добијеним у цитираној литератури. Грешка, која представља квадрат одступања између задатих тачака и стварних тачака на путањи, у сва четири примера је мања у поређењу са резултатима из цитиране литературе. Нарочито је то изражено у Примерима 3 и 4.

У поступку синтезе мора се водити рачуна да механизам који се добије буде изводљив односно да не дође до диспропорције дужина чланова механизма. Однос дужина најдужег и најкраћег члана добијених механизма у свим примерима је веома задовољавајући. МКН поред веома добрих резултата показује и изузетну ефикасност у примени која се огледа у веома брзој конвергенцији приказаној у развоју грешке дуж итеративног процеса, Слике 5.15, 5.12, 5.9 и 5.6. Такође се може уочити да је вредност стандардне девијације за сва четири

примера веома мала, што указује да се МКН алгоритмом избегава улазак у простор локалног минимума.

Алгоритам МКН је успешно примењен у синтези зглобног четвороугаоног механизма као генератора путање, и даје могућност његове примене и у другим проблемима синтезе равних полужних механизама код којих се полази са различитим захтевима.

Поглавље

6

НОВИ МОДЕЛИ

6. ПРИМЕНА АНАЛИЗИРАНИХ АЛГОРИТАМА НА НОВЕ МОДЕЛЕ

6.1. ОПТИМИЗАЦИЈА РЕЖИМА ОБРАДЕ

Избор параметара (режима) обраде (брзина резања- $v\,[\mathrm{m/s}]$, корак- $s\,[\mathrm{mm/o}]$, дубина резања- $a\,[\mathrm{mm}]$) су предмет рада технолога, који бира одговарајуће вредности, на основу врсте обраде, изабраног алата, материјала предмета обраде, препоручених вредности и на основу свог искуства, односно производне праксе предузећа где се изводи машинска обрада. Овакав начин постављања параметара обраде не гарантује оптималне перформансе посматраног метода обраде, где под перформансама продразумевамо остварење високог квалитета обраде, максималну производност и минималне трошкове.

Почевши још од 1907. године, када је F.W. Taylor објавио своју књигу „On the art of cutting metals“, па до данашњих дана основни мотив, у металопреради је избор оптималних режима, у циљу побољшања квалитета обраде, смањења трошкова и повећања ефикасности производње. Проблем избора оптималних режима обраде је у ствари проблем вишекритеријумске оптимизације, где су функције циља често супростављене једне другој. Наиме, јасно је да повећање квалитета обраде изискује повећање трошкова обраде, док повећање ефикасности (коришћење робуснијих режима обраде: већа брзина резања, већи корак и повећана дубина резања), може да доведе до смањења квалитета обраде.

За оптимизацију параметара обраде користе се различите методе, од конвенционалних оптимизационих метода који подразумевају: геометријско програмирање [123], динамичко програмирање [124-126], целобројно програмирање [127], детерминистичке технике [128-133], па до мета-хеуристичких алгоритама [134-156].

Мадић и Радовановић, у [138] примењују патерни алгоритам претраживања (pattern search algorithm) у циљу добијања оптималних параметара обраде. Овај алгоритам се карактерише низом истражних потеза који узимају у обзир понашање функције циља на низу тачака, при чему све оне леже на рационалној решетки. Алгоритам израчунава секвенцу тачака које могу или не могу имати прилаз ка оптималној тачки. Овај алгоритам користи групу вектора $\{v_i\}$, који представљају низ, за одређивање тачака, за претраживање простора решења, у свакој итерацији. Димензија низа је одређена бројем независних променљивих функције циља. Овај алгоритам, аутори користе за проналажење оптималне вредности квалитета обрађене површине, R_a, и то код поступка обраде воденим млазом, стругањем, глодањем и бушењем. За функцију циља R_a, користили су

једначине за предвиђање квалитета обрађене површине, добијене експерименталним путем.

Поред једнокритеријумске оптимизације, овај алгоритам аутори користе и за вишекритеријумску оптимизацију код одређивања оптималних режима обраде код електроерозионе обраде, односно електроерозионе обраде жицом. У првом случају, функција циља је одређивање максималне производности кроз количину скинутог материјала MRR, уз најбољи квалитет обрађене површине, SR, док је у другом случају функција циља било одређивање максималне брзине резања, V_m уз, такође, најбољи квалитет обрађене површине, R_a.

U. Zuperl и F. Cus, у [148], примењују неуро мреже за одређивање оптималних услова при обради стругањем. Модификовани неуро алгоритмом добијају се режими обраде који обезбеђују максималну производност – минимално време обраде, $\min T_P(v,f,a)$, смањење трошкова обраде, $\min C_P(v,f,a)$ и побољшање квалитета обраде, $\min R_a(v,f,a)$. Као што се види, аутори су користили предложени алгоритам за решавање проблема код вишекритеријумске оптимизације.

Алгоритам ради на основу „feedforward“ и радијално базних мрежа уз истовремену примену нових напредних алгоритама учења, који се аутоматски прилагођавају тренутним условима у тренажном процесу. Полазна тачка за коришћење неуронских мрежа је мноштво репрезентативних нумеричких података прикупљених експерименталним мерењем. Обука неуронских мрежа се може делимично одвијати и на основу резултата датих од стране постојећих аналитичких и емпиријских модела.

За решавање вишекртитеријумске оптимизације код одређивање оптималних параметара обраде стругањем (брзина резања - v, корак- s и дубина резања- a), K. Deb и R. Datta у [149] користе еволутивни вишекритеријумски оптимизациони (evolutionary multi-objective optimization EMO) алгоритам. EMO је модификован коришћењем процедуре локалне претраге у циљу добијања бољих својстава конвергенције.

У првом случају, вишекритеријумска оптимизација се односи на одређивање оптималних режима стругања у циљу добијања минималног времена обраде, $\min T_P(v,f,a)$ и минимизација дела животног века алата $\min \xi(MRR(v,f,a),T(v,f,a))$, где је MRR-количина уклоњеног материјала обратка, а T-време обраде. У другом случају се посматрају три функције циља: минимизирање времена обраде, $\min T_P(v,f,a)$; минимизација трошкова обраде, $\min C_P(v,f,a)$ и минимизација храпавости обрађене површине, R_a, $\min R_a(v,f,a)$.

A. R. Yildiz у [136], представља нови приступ хибридног оптимизационог алгоритма заснованог на алгоритму диференцијелане еволуције (DE) и рецептора за уређивање својстава имуног система, примењеног на оптимизацију процеса обраде глодањем. Након избора оператора у DE, уређивање рецептора се примењује на свим популацијама у циљу избегавања локалног минимума. У уређивању рецептора, нове индивидуа и 25% популације се производе случајно и замењују се индивидуама случајном изабраним из популације. Уколико индивидуа, изабрана из популације, представља најбоље решење, она се не замењује са новом, већ се уместо ње бира следећа индивидуа из популације. Вишекритеријумска оптимизација се састоји од две функције циља: минимизација трошкова обраде глодањем, $\min C_u(v,s)$ и минимизацијом времена израде $\min T_u(v,s)$.

У свим горе поменутим радовима, граничне функције узимају у обзир могуће брзине резања на изабраним машинама, могуће вредности корака, расположиву снагу машине, максимално могућу силу резања, као и граничне вредности дубине резања.

Поред оптимизације режима обраде мета-хеуристички алгоритми се користе и у оптимизацији прибора за стезање и позиционирање. Тако, N. Kaya, у [157], примењује генетски алгоритам (GA) за оптимално позиционирање и стезање радног предмета у процесу обраде. GA алгоритам кроз интеграцију са кодом коначних елемената израчунава вредности функције циља за сваку генерацију.

У примерима који следе, оптимизацији тела стругарског ножа и оптимизација ексцентра као дела стезног прибора, тестира се примена претходно анализираних оптимизационих алгоритама: цикличног алгоритма фамилије слепих мишева (Loop BFA), хибридног алгоритма кукавичје претраге и алгоритма свица (H-CS-FF) и модификованог алгоритма јата крила (MKH); на проблеме из области обраде метала резањем.

6.2. ОПТИМИЗАЦИЈА ТЕЛА СТРУГАРСКОГ НОЖА

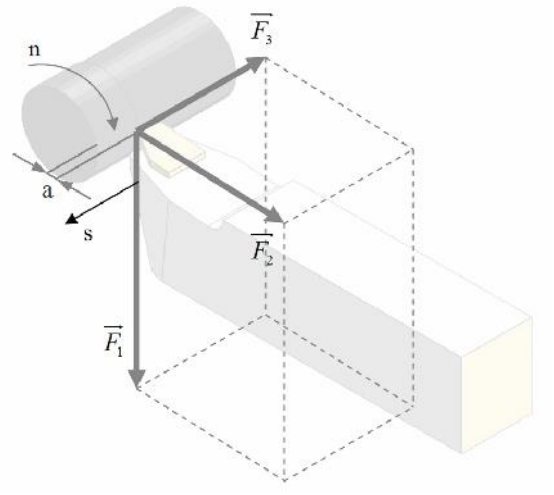
Код обраде метала стругањем, одвајање материјала (настанак струготине) настаје продирањем резног клина стругарског ножа у материјал предмета обраде. Спада у групу косог резања, код кога главно кретање, и то обртно, изводи обрадак – предмет обраде, док помоћно кретање је праволинијско и изводи га алат.

Приликом извођења стругарске обраде, јављају се три главна отпора резању, Слика 6.1, [158]:

F_1-главни отпор резању,

F_2 - отпор продирању резног клина алата у материјал обрадка,

F_3 - отпор помоћног кретања.

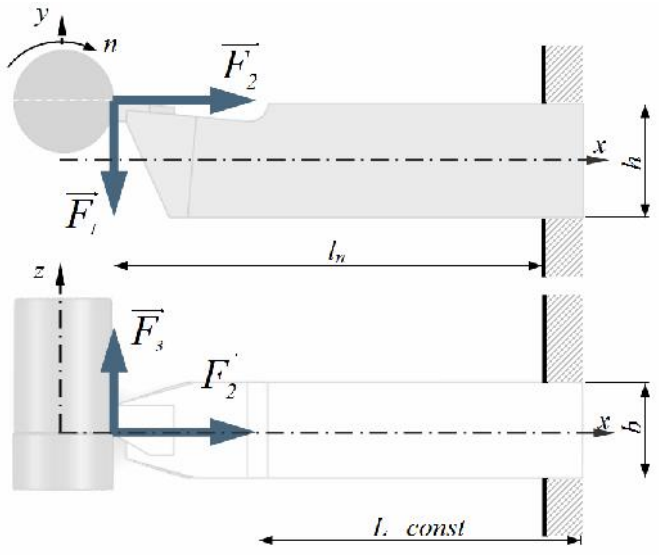


Слика 6. 1. Модел сила резања код обраде метала стругањем,
n - број обртаја [o/min] , s - корак [mm/o], a – дубина резања [mm]

Основни параметри, режими обраде, код стругања су, [158]:

- дубина резања, $a[mm]$;
- број обртаја обратка, $n[o/\min]$;
- корак алата (стругарског ножа), $s[mm/o]$

Главни отпори резања, Слика 6.2, се могу изразити у функцији од режима обраде што је и представљено једначинама (6.1) до (6.3).



Слика 6. 2. Главни отпори резању, у хоризонталној и вертикалној равни

$$F_1 = C_{k1} a^{x_1} s^{y_1}, \tag{6. 1}$$

$$F_2 = C_{k2} a^{x_2} s^{y_2}, \tag{6. 2}$$

$$F_3 = C_{k3} a^{x_3} s^{y_3}, \tag{6. 3}$$

где су: C_{ki}, x_i и y_i, $i = 1,2,3$; фактори који зависе од врсте материјала обратка, односно од врсте материјала алата, [158, 159].

Пошто је стругарски нож укљештен у носачу алата, мора се извршити провера отпорности тела ножа, на савијање и притисак, који се јавља под утицајем главних отпора резању. Изрази за напоне, који се јављају у телу стругарског ножа, дати су једначинама (6.4) – (6.6).

$$\sigma_x = -\frac{F_2}{A} = -\frac{F_2}{b \cdot h}, \tag{6. 4}$$

$$\sigma_y = \frac{M_y}{W_y}, \tag{6. 5}$$

$$\sigma_z = \frac{M_z}{W_z} \tag{6. 6}$$

где је:

σ_x [N/mm^2], притисак који трпи тело ножа од отпора продирања F_2 [N],

σ_y [N/mm^2], нормални напон услед момената савијања око y-ose M_y [Nmm], једначина (6.7).

W_y [mm^3], отпорни момент пресека тела ножа за y-осу, једначина (6.8)

σ_z [N/mm^2], нормални напон услед момената савијања око z-ose M_z [Nmm], једначина (6.9).

W_z [mm^3], отпорни момент пресека тела ножа за z-осу, једначина (6.10)

b, h [mm], димензије попречног пресека тела ножа.

$$M_y = -F_3 \cdot l_n; \ , \tag{6. 7}$$

$$W_y = \frac{b^2 \cdot h}{6}; \tag{6. 8}$$

$$M_z = -F_1 \cdot l_n + F_2 \cdot \frac{h}{2}; \tag{6. 9}$$

$$W_z = \frac{b \cdot h^2}{6}; \tag{6. 10}$$

Када се једначине (6.7) и (6.8) уврсте у једначину (6.5), односно када се једначине (6.9) и (6.10) уврсте у једначину (6.6) добиће се изрази за σ_y и σ_z представљени једначинама (6.11) и (6.12) респективно.

$$\sigma_y = -\frac{6 \cdot C_{k3} a^{x_3} s^{y_3} \cdot l_n}{b^2 \cdot h} \tag{6. 11}$$

$$\sigma_z = -\frac{6 \cdot C_{k1} a^{x_1} s^{y_1} \cdot l_n}{b \cdot h^2} + \frac{6 \cdot C_{k2} a^{x_2} s^{y_2} \cdot h}{2 \cdot b \cdot h^2} \tag{6. 12}$$

Укупан нормални напон представља збир апсолутних вредности напона σ_x, σ_y и σ_z и он мора да задовољи критеријум дозвољеног напона, једначина (6.13).

$$\sigma = |\sigma_x| + |\sigma_y| + |\sigma_z| \le \sigma_{doz} \tag{6. 13}$$

Ова анализа оптерећења тела стругарског ножа је била неопходна, због постављања ограничења који се односи на напонско стање тела ножа у процесу обраде стругањем.

Оптимизациони проблем се односи на одређивање максималне количине материјала приликом обраде стругањем, $MRR = f(v,s,a)$, при минималним димензијама попречног пресека тела стругарског ножа, $V = f(b,h,l_n)$. Математички модел функције циља са ограничењима дат је једначинама (6.14)-(6.20).

$$\max(MRR = v \cdot s \cdot a)\left[\mathrm{mm}^3/\min\right], \tag{6. 14}$$

$$\min(\mathrm{V} = \mathrm{b} \cdot \mathrm{h} \cdot \mathrm{l}_n) \tag{6. 15}$$

Ограничења:

$$g_1 = \sigma - \sigma_{doz} \le 0; \tag{6. 16}$$

$$g_2 = F_1 - 2300 \le 0; \tag{6. 17}$$

$$g_3 = P - P_{\max} \cdot \eta \le 0; \tag{6. 18}$$

$$g_4 = 1.6 - \frac{h}{b} \le 0; \tag{6. 19}$$

$$g_5 = 1.6 - \frac{l_n}{h} \le 0; \tag{6. 20}$$

Једначинама (6.14) и (6.15) дефинисана је вишекритеријумска оптимизација. Наиме, идеја је да се изаберу режими обраде брзина резања - v, корак - s, дубина резања - a, при којима ће се добити максимална количина обрађеног материјала

(MRR), а при томе да су димензије попречног пресека ножа минималне. У једначини (6.15) није узета у обзир укупна дужина тела ножа L, јер се она може изразити у функцији слободне дужине ножа, l_n и дужине која је неопходна за стезање: $L = l_n + const$.

Ограничење g_2 се односи на препоруку максималне силе резања, $F_{1\max} = 5000\,[\mathrm{N}]$, односно g_3 на максималну снагу резања, $P_{\max} = 10\,[\mathrm{kW}]$. Подаци за $F_{1\max}$ и $P_{\max}$, преузети из [149], се односе на обраду челичне шипке, са стругарским ножем са плочицом P20, на CNC стругу снаге мотора од 10kW и степена корисности $\eta = 75\%$. Ограничења g_4 и g_5 се односе на препоручене вредности односа димензија попречног пресека тела ножа, односно односа слободне дужине тела ножа l_n и висине h попречног пресека, [158, 159].

Пројектне промењљиве: брзина резања - v, корак - s, дубина резања - a, димензије тела пресека стругарског ножа - $b \times h$ се налазе у границама:

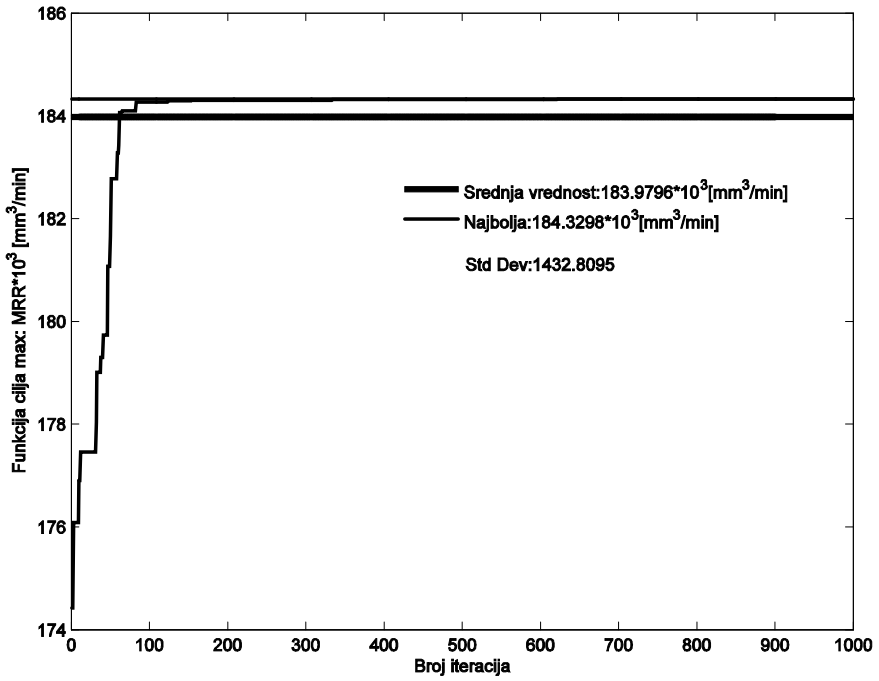
$$250 \le v \le 400\,;\, 0.15 \le s \le 0{,}55\,;\, 0.5 \le a \le 6\,,[149];$$
$$6 \le b \le 40\,;\, 6 \le h \le 50\,;\, 9 \le l_n \le 80\,[159].$$

6.2.1. Резултати оптимизације

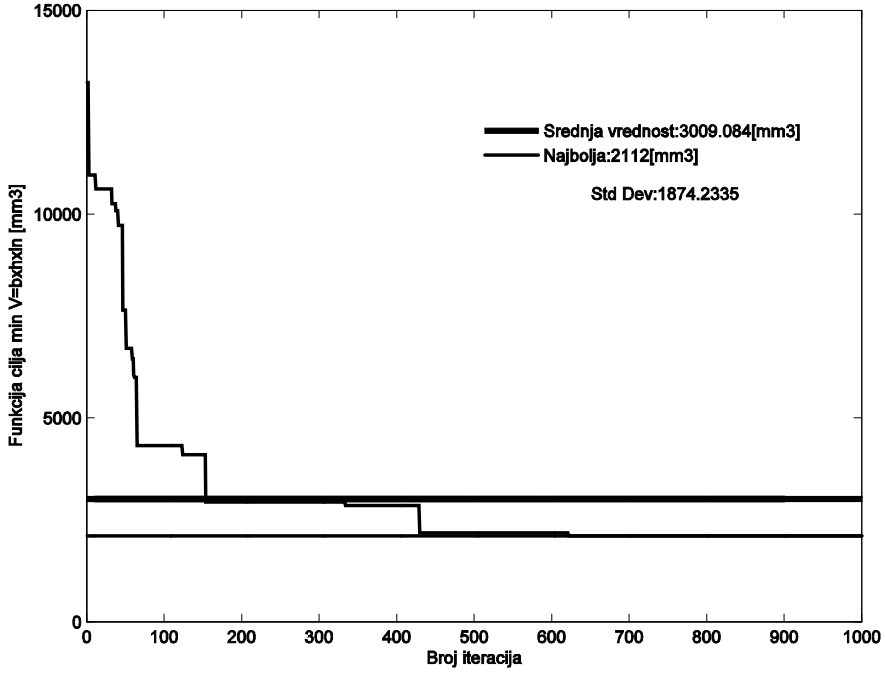
Упоредни приказ резултата, функције циља (6.14)-(6.18), добијених алгоритмима: циклични алгоритам фамилије слепих мишева (Loop BFA), хибридни алгоритам кукавичје претраге и алгоритма свица (H-CS-FF) и модификовани алгоритам јата крила (MKH), дат је у Табели 6.1. Ток генерисања вредности функција циља, приказан је на Сликама од 6.3 до 6.5.

Табела 6. 1. Резултати оптимизације тела стругарског ножа

	LoopFBA		H-CS-FF		MKH	
v	400.00		269.33896		396.54641	
s	0.55		0.15		0.4909	
a	0.8378628547		1.45836		1.55493	
$b \times h \times l_n$	$6 \times 11 \times 32$		$6 \times 49 \times 80$		$6 \times 10 \times 16$	
g_1	-4437.506512054349		-2500.859192700796524		-3796.969160138297	
g_2	-1.250043413696		- 0.578612316356966		-0.370862022226	
g_3	-25.219197149399		-190.310830811429		-22.737268160532	
g_4	-1.309090909091		- 0.0326530612244897		0	
g_5	-0.23333333		-6.56666666666667		-0.0666666	
	MRR	V	MRR	V	MRR	V
Најбоље	184329. 82803274	2112.00	168511.0910474	2288.00	302692.3929835	960.00
Средња вредност	183979. 56124019	3009.084	165766.029386	3479.672	299723.9669634	2322.4833
Најлошије	174427. 75962776	13230.00	162150.580446	4800.00	242608.23611	13182.434
С.Д.	1432. 809543	1874. 2335	3091.82371	1229.655	5751.79684722	1000.729

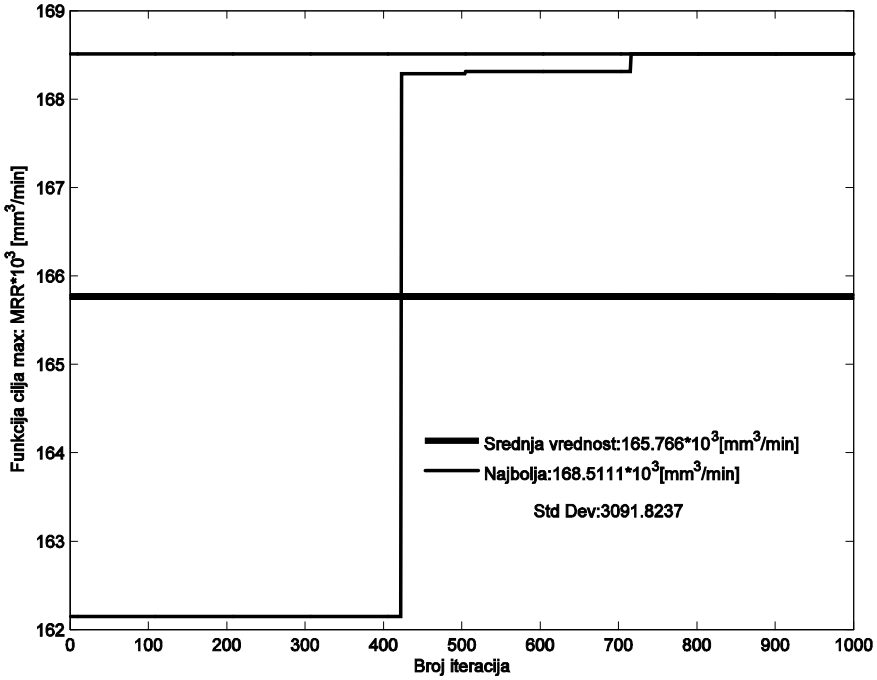


а)

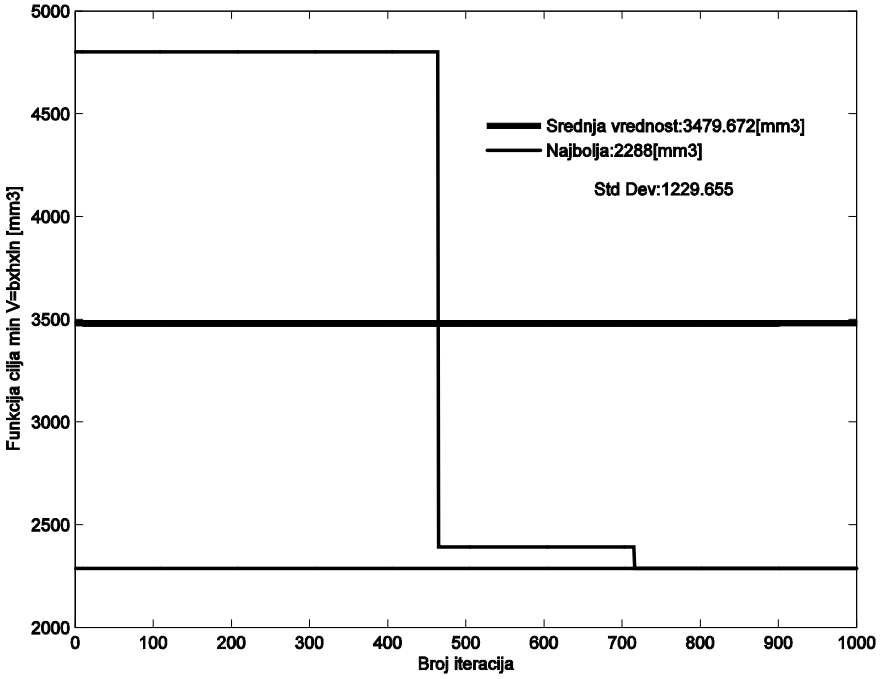


б)

Слика 6. 3. Најбоља и средња вредност функције циља: (а) $\max(MRR)$, б) $\min(V)$ применом алгоритма LoopFBA

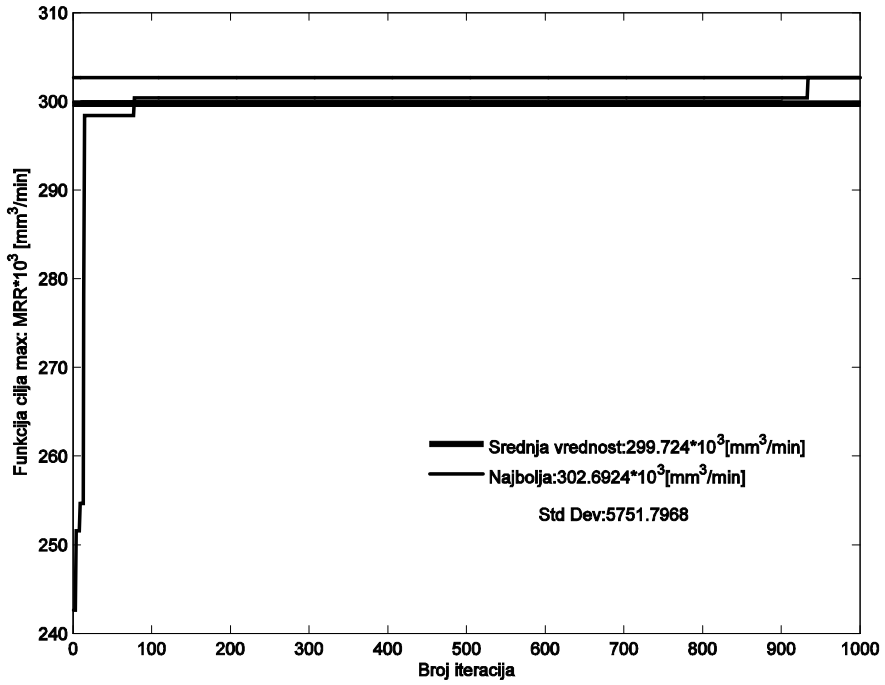


а)

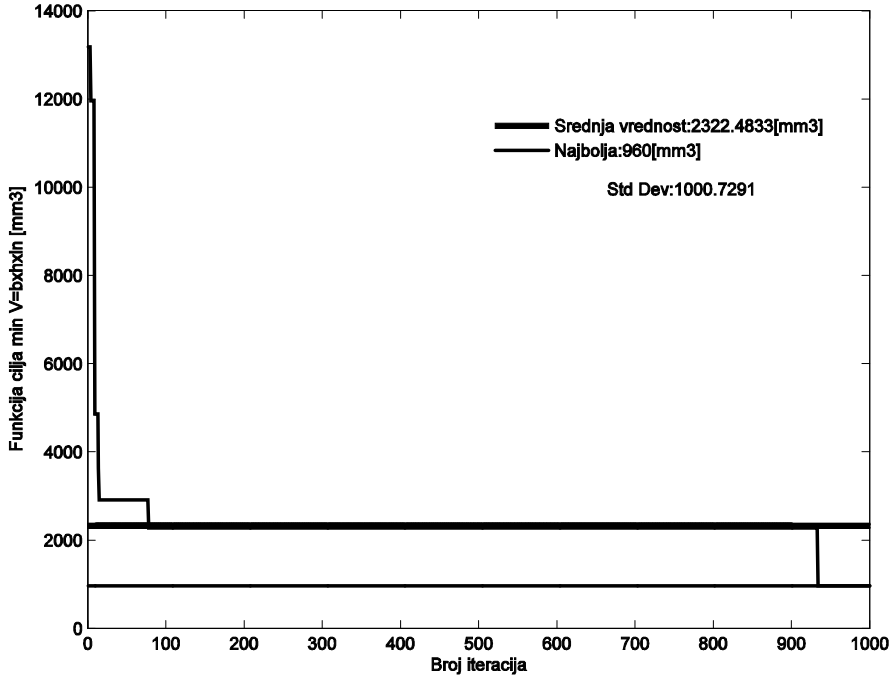


б)

Слика 6. 4. Најбоља и средња вредност функције циља: (а) max(*MRR*) , б) min(*V*) применом алгоритма H-CS-FF



а)



б)

Слика 6. 5. Најбоља и средња вредност функције циља: (а) $\max(MRR)$ **,** б) $\min(V)$ применом алгоритма МКН

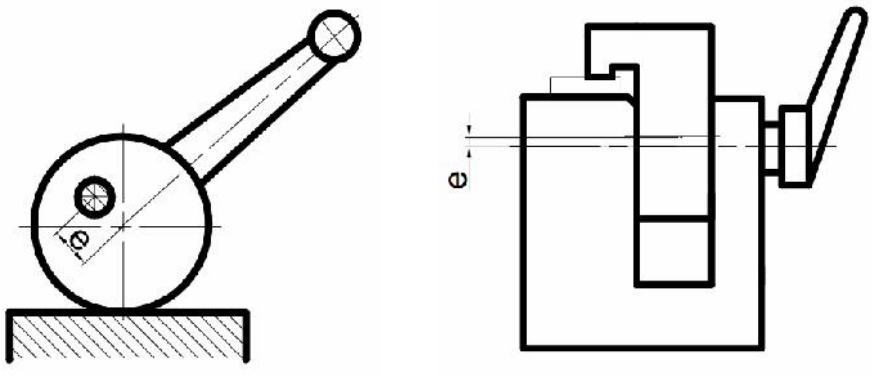
Из Табеле 6.1, може се видети да су вредности функције циља, $\max(MRR)$ и $\min(V)$ добијене модификованим алгоритмом крила – МКН, најбоље у односу на

друга два алгоритма. Такође, за сва три алгоритма, функције ограничења, g_i , $i = 1,..,6,$ су задовољене јер је $g_i \le 0, \forall\, i = 1,..,6.$

Са друге стране, када се анализирају слике конвергенције резултата, од 6.3 до 6.5, примећује се да применом цикличним алгоритмом слепог миша-LoopFBA добијамо најбржу конвергенцију, односно најбрже се долази до најбољих решења, док код МКН алгоритма имамо такође веома брзу конвергенцију, а затим учестано понављање вредности, блиске или једнаке средњим вредностима, функција циља, да би се тек при крају итеративног процеса добили најбољи резултати.

6.3. ОПТИМИЗАЦИЈА ЕКСЦЕНТРА КОД СТЕЗНОГ АЛАТА

Стезање ексцентром се базира на принципу клина – косе равни. Ексцентар је цилиндричан, Слика 6.6. а) или плочаст елемент Слика 6.6. б) обртан око осе која је ексцентрично постављена у односу на његову осу симетрије, [161].



а) б)

Слика 6. 6. Примери екцентра

Ексцентри се израђују од челика за цементацију и цементирају се на тврдоћу 55-60HRC, при чему се контактне површине ексентра брусе. Стезање ексентром се обавља у кратком временском интервалу, због краћег пута послуживања (покрет ручице надоле), међутим ексентри имају ужу примену. Они се примењују у операцијама обраде које карактерише висока стабилност стезног прибора, мање силе резања и обрада са незнатним вибрацијама. У принципу постоје две врсте ексцентара: кружни и криволинијски. За проблем оптимизације је изабран кружни ексентар, и даље у тексту се дају основне формуле за прорачун ексентра.

Један од основних услова које ексцентар треба да задовољи јесте обезбеђење самокочивости. Из тог разлога треба размотрити односе који треба да буду испуњени да би угао трења ρ био већи од угла пењања ексцентра, [161].

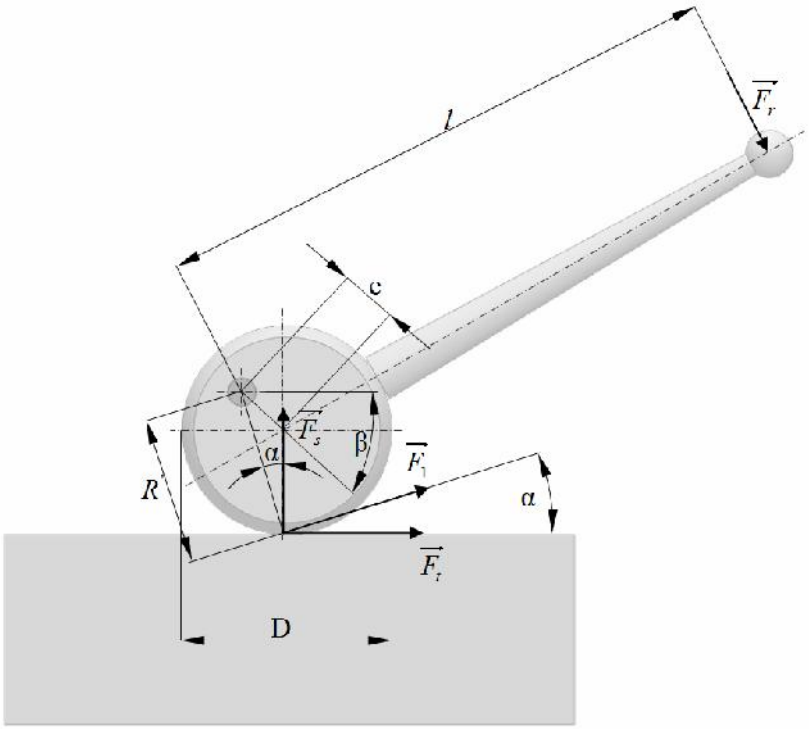
На основу Слике 6.7, може се успоставити зависност између угла пењања ексцентра - α и угла закретања ексцентра- β , једначина (6.21).

$$\tan\alpha = \frac{e\cdot\cos\beta}{R + e\cdot\sin\beta}; \qquad (6.\ 21)$$

где је e - ексцентритет,

$R = \frac{D}{2}$, полупречник ексцентра.

Сила стезања F_s, која се постиже при стезању кружним ексцентром може се, са практично довољном тачношћу, израчунати уз претпоставку да ексцентар функционише као клин. Проблем се у принципу своди на статичку равнотежу сила у односу на тачку обртања рукавца ексцентра, Слика 6.7.



Слика 6. 7. Анализа сила на кружном ексцентру

Полазећи од једначине равнотеже момената на тачку обртања рукавца ексентра, једначина (6.22), односно једначине расподеле сила код клина (односно косе равни), једначина (6.23), добиће се зависност силе стезања од силе на ручици полуге ексентра, једначина (6.24).

$$F_r\cdot l = F_1\cdot R' \text{ и } F_t = F_1\cdot\cos\alpha; \qquad (6.\ 22)$$

$$F_s = \frac{F_t}{\mathrm{tg}(\alpha + \rho_1) + \mathrm{tg}\rho_2}; \tag{6. 23}$$

$$F_s = \frac{F_r \cdot l}{\left[\mathrm{tg}(\alpha + \rho_1) + \mathrm{tg}\rho_2\right] \cdot R'}; \tag{6. 24}$$

где је:

F_r - сила на ручици ексцентра, може се усвојити $\approx 150\mathrm{N}$

l - крак силе, [mm]

α - променљиви угао пењања ексентра [rad]

ρ_1 - угао трења на контактној површини ексентра и предмета обраде [rad]

ρ_2 - угао трења рукавцу ексентра [rad]

R' - растојање обртне тачке ексентра од тачке контакта: $R' = \dfrac{\dfrac{D}{2} + e \cdot \sin\beta}{\cos\alpha}$

F_1 - резултујућа сила која делује на предмет обраде.

На основу ових анализа, сада можемо поставити оптимизациони модел кружног ексцентра. И у овом случају ради се о вишекритеријумској оптимизацији са две функције циља. Прва се односи на одређивање минималних димензија ексцентра, али, што је и друга функција циља, уз постизање максималне силе стезања.

Математички модел оптимизације ексентра и одговарајуће функције ограничења дате су једначинама од (6.25) до (6.32).

$$\mathrm{Minimize}\, V = \left(\frac{D}{2}\right)^2 \cdot \pi \cdot b; \tag{6. 25}$$

$$\mathrm{Maximize}\, F_s = \frac{F_r \cdot l}{\left[\mathrm{tg}(\alpha + \rho_1) + \mathrm{tg}\rho_2\right] \cdot R'}; \tag{6. 26}$$

Ограничења:

$$g_1 = F_R - F_s \le 0;; \tag{6. 27}$$

$$g_2 = p_{kontakt} - \left(p_{kontakt}\right)_{doz} \le 0; \tag{6. 28}$$

$$g_3 = \left(p - p_{doz}\right)_{rukavac} \le 0;; \tag{6. 29}$$

$$g_4 = 14 - \frac{D}{e} \le 0;; \tag{6. 30}$$

$$g_5 = \frac{18.65 \cdot F_s}{K \cdot D \cdot b} + 41.4 - HRC \le 0;; \tag{6. 31}$$

$$g_6 = \left(p - p_{doz}\right)_{ekscentar} \le 0;; \tag{6. 32}$$

Ограничење g_1 обезбеђује да сила стезања не буде мања од максималне силе резања. У овом примеру претпостављено је да је максимална сила резања $F_R = 2500\text{N}$.

Ограничење g_2 се односи на проверу да ли ће на месту контакта доћи до трајних деформација, [161]. Контактни површински притисак, $p_{kontakt}$, се рачуна по једначини (6.32), док се за вредност $\left(p_{kontakt}\right)_{doz}$, узима вредност за 20% већа од напона на граници течења слабијег материјала, R_e, $\left(p_{kontakt}\right)_{doz} = 1.2 \cdot R_e$.

$$p_{kontakt} = \frac{4}{\pi}\sqrt{\frac{E_e \cdot F_s}{R' \cdot b}}; \tag{6. 33}$$

где је

$E_e = \frac{2 \cdot E_1 \cdot E_2}{E_1 + E_2}$ - еквивалентни Јунгов модул еластичности; E_1 и E_2 Јунгов модул еластичности ексцентра, односно предмета обраде, респективно и

b - ширина ексцентра.

Даље је потребно проверити вредности притисака који се добијају на линији контакта ексентра и радног предмета, једначина (6.34), односно притисак у рукавцу ексентра једначина (6.35) у односу на дозвољене вредности притиска за материјале ексентра и радног материјала, p_{doz}. Пошто је у питању челик, дозвољена вредност притиска у увом случају је $p_{doz} = 150\left[\frac{\text{N}}{\text{mm}^2}\right]$.

$$p_{eksentra} = \frac{F_s}{D \cdot b}; \tag{6. 34}$$

$$p_{rukavac} = \frac{F_s}{D_r \cdot b}; \tag{6. 35}$$

где је D_r пречник рукавца ексентра.

Једначина (6.31) се односи на проверу деформације површине по Левину и Решетову, за линијски додир. У овој једначини K представља дозвољено контактно оптерећење, које зависи од врсте термичког третмана (потпуно каљен или цементиран па каљен; индукционо каљен или нитриран) и типа контакта (лопта, кратак ваљак, дуги ваљак) . За индукционо каљене површине и кратак ваљак вредност K износи $18\left[\frac{\mathrm{N}}{\mathrm{mm}^2}\right]$, док је вредност $HRC = 96.4$, за челик Č3130.

Пројектне промењљиве: пречник ексцентра - D, ексцентар - e, угао закретања ексентра - β, крак силе - l, ширина ексцентра - b и пречник рукавца ексентра - D_r се налазе у границама, [162]:

$$40 \le D \le 115\,;\, 4 \le e \le 10\,;\, 30 \le \beta \le 90\,;\; 100 \le l \le 180\,;\, 60 \le b \le 120\,;\, 15 \le D_r \le 35\,.$$

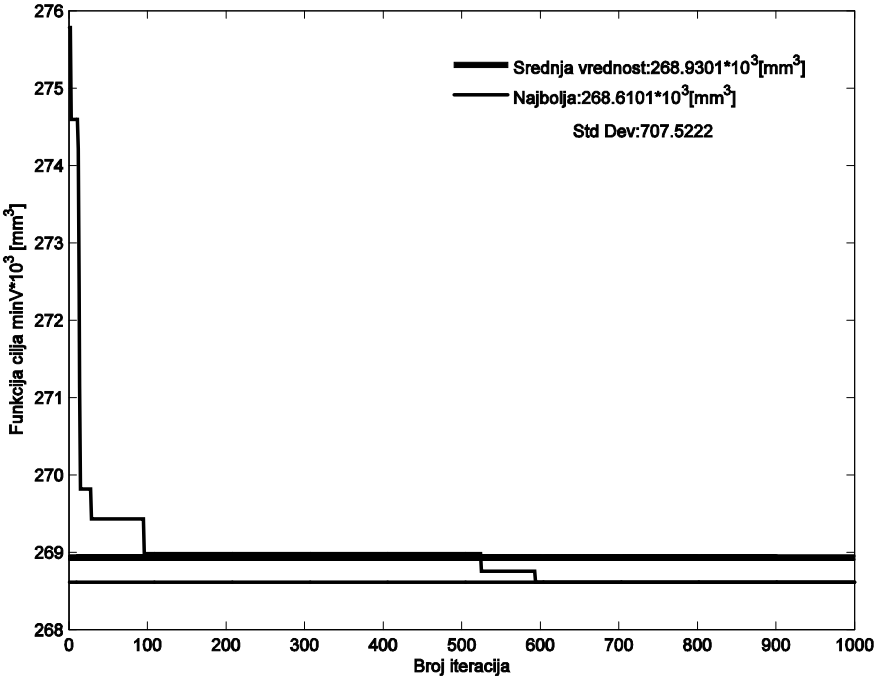
6.3.1. Резултати оптимизације

Упоредни приказ резултата, функције циља (6.25)-(6.32), добијених алгоритмима: циклични алгоритам фамилије слепих мишева (Loop BFA), хибридни алгоритам кукавичје претраге и алгоритма свица (H-CS-FF) и модификовани алгоритам јата крила (MKH), дат је у Табели 6.2. Ток генерисања вредности функција циља, приказан је на Сликама од 6.8 до 6.10.

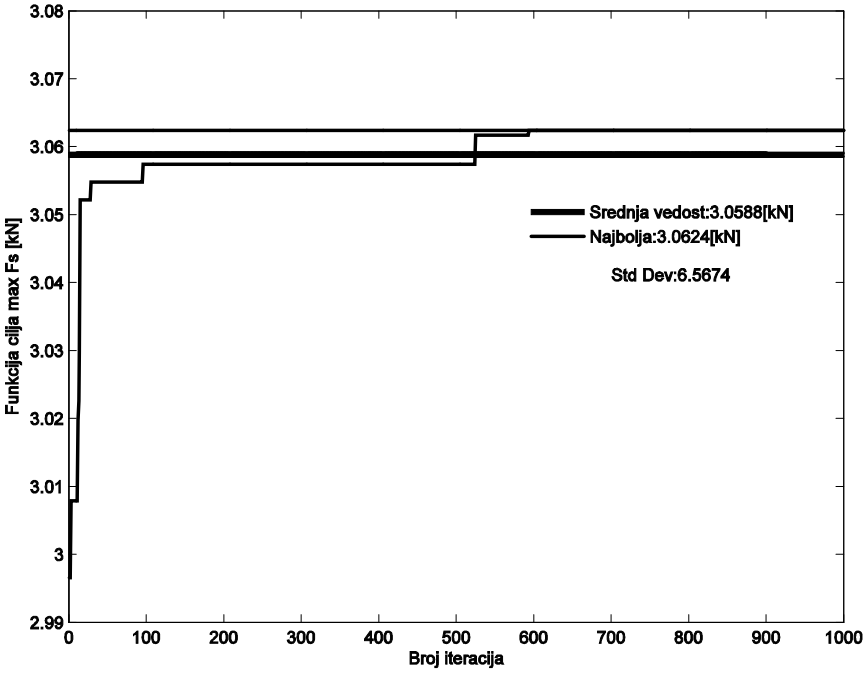
Табела 6. 2. Резултати оптимизације ексцентра

	LoopFBA		H-CS-FF		MKH	
D	56.00007347098426		68.86583		56.22869	
e	4		4.00000		4.01169	
β	90° (π/2)		90° (π/2)		85° (π·17/30)	
l	164.3		180.00000		174.13285	
b	10.039361		96.00000		110.47683	
D_r	31.0841824131		35.00000		20.12035	
g_1	-562.3928		-293.665770918547		-598.8392358535625	
g_2	-0.1095		-38.5151961538239		-1.3786958795177	
g_3	-149.0965		-149.168551853893		-148.6059051886784	
g_4	-0.0012		-3.2164575		-0.0162101259070	
g_5	-54.4804		-54.56216975549		-54.4831360914344	
g_6	-149.4985		-149.577429254628		-149.5011501150574	
	V	F_s	V	F_s	V	F_s
Најбоље	268610.06705	3062.39278	292348.91465	2975.81491	274332.3369382	3098.8395451
Средња вредност	268930.13266	3058.82352	323218.03655	2815.65655	277164.7488624	3039.7645054
Најлошије	275779.70696	2996.54577	423905.92624	2231.86118	295327.5310168	2933.7663861
С.Д.	707.52216565	6.56736132	45435.2027358	266.058509	2191.683067381	42.749816778

Из горње табеле, уочава се да се највећа сила стезања добија алгоритмом MKH, док је најмања запремина добијена применом алгоритма LoopFBA. Сва ограничења, g_i , $i = 1,..,6,$ су задовољена, јер је $g_i \le 0, \forall\, i = 1,..,6.$

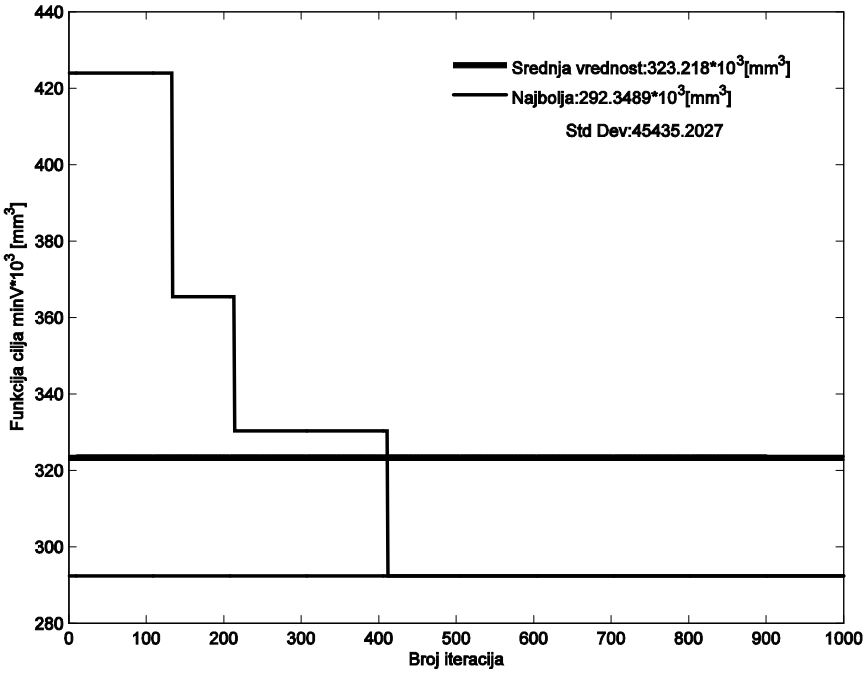


а)

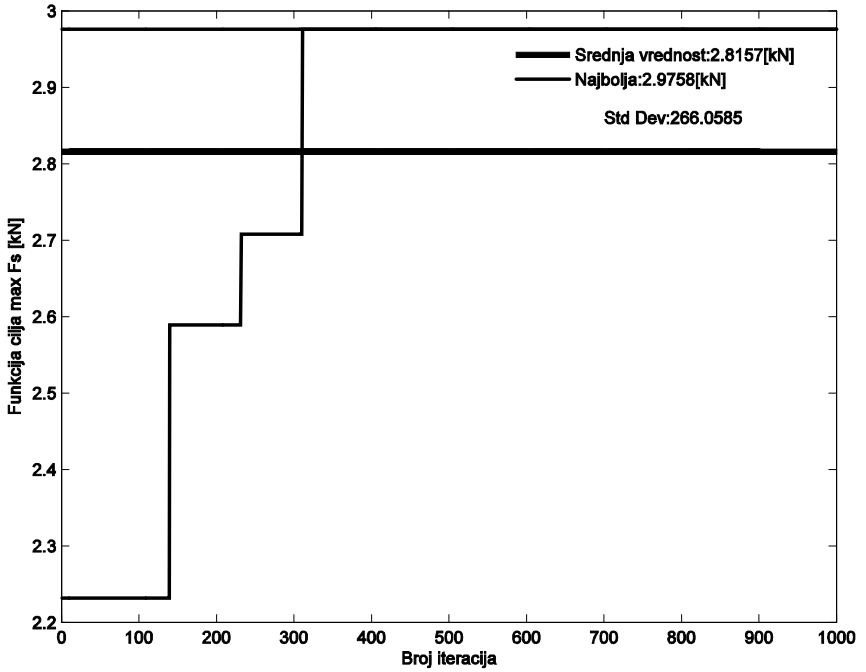


б)

Слика 6. 8. Најбоља и средња вредност функције циља: (а) $\min(V)$, б) $\max(F_s)$ применом алгоритма LoopFBA

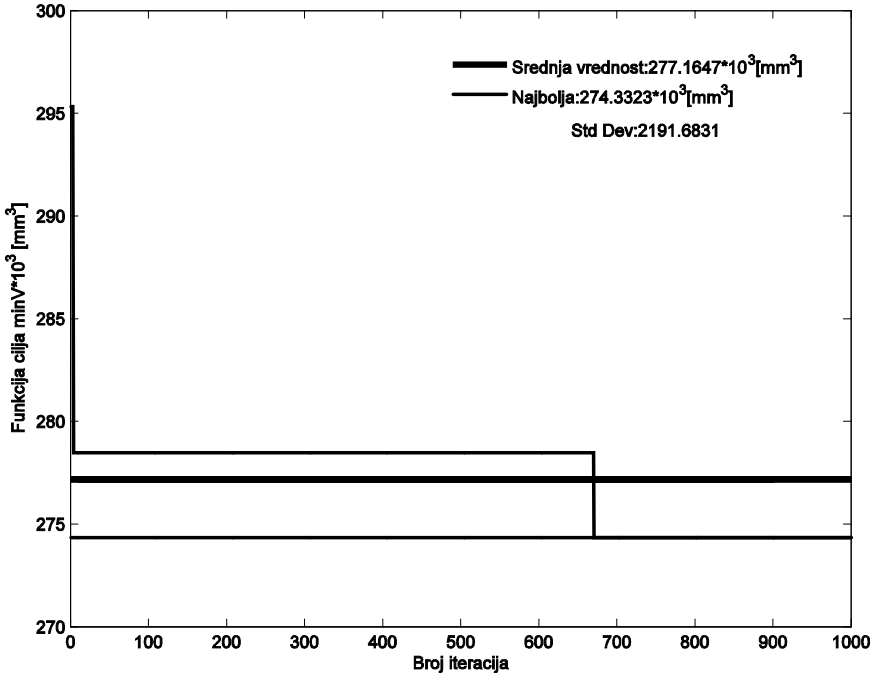


а)

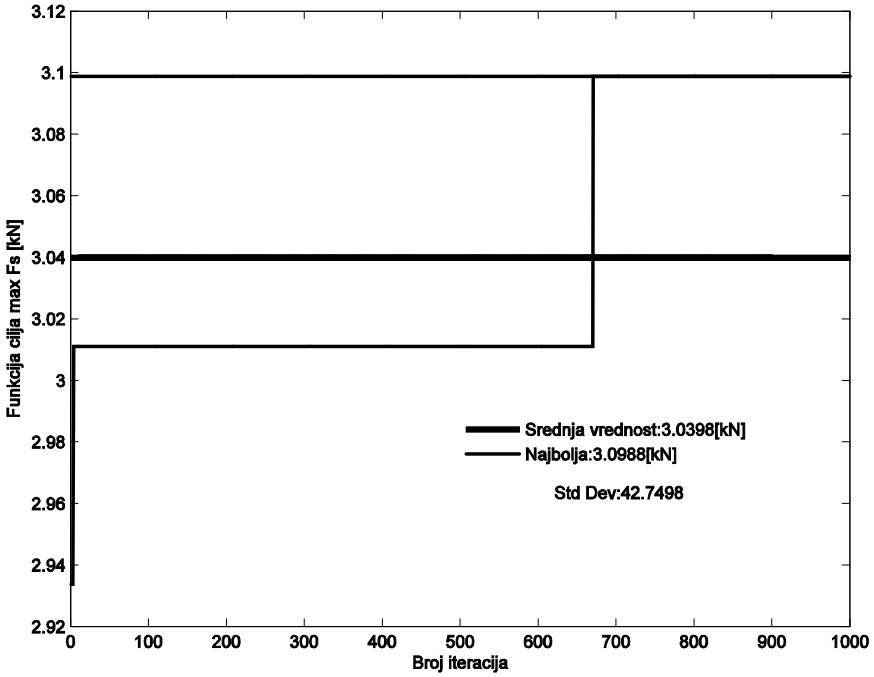


б)

Слика 6. 9. Најбоља и средња вредност функције циља: (а) $\min(V)$, б) $\max(F_s)$ применом алгоритма H-CS-FF



а)



б)

Слика 6. 10. Најбоља и средња вредност функције циља: (а) $\min(V)$, б) $\max(F_s)$ применом алгоритма МКН

Анализирајући слике конвергенције резултата, од 6.8 до 6.10, примећује се да применом цикличног алгоритма слепог миша-LoopFBA добијамо најбржу конвергенцију, односно најбрже се долази до најбољих решења, док код МКН алгоритма имамо такође брзу конвергенцију до решења приближним средњој

вредности, и њено учестано понављање, да би се тек при крају итеративног процеса добили најбољи резултати.

6.4. ЗАКЉУЧНЕ НАПОМЕНЕ УЗ ШЕСТО ПОГЛАВЉЕ

У овом поглављу, на примерима оптимизације тела стругарског ножа и ексцентра, тестирала се примена оптимизационих алгоритама: цикличног алгоритма фамилије слепих мишева (Loop BFA), хибридног алгоритма кукавичје претраге и алгоритма свица (H-CS-FF) и модификованог алгоритма јата крила (MKH).

На основу добијених резултата, из Табела 6.1 и 6.2, и дијаграма 6.5-6.7, односно 6.8-6.10, може се закључити да наведени алгоритми исправно функционишу. Конвегренција је добра, уз високу вредност стандардне девијације за оба примера и сва три алгоритма. Ово се може објаснити чињеницом да је у питању вишекритеријумска оптимизација, две противуречне функције циља. Наиме, у првом примеру су тражене минималне димензије тела стругарског ножа, уз постизање максималне количине скинутог материјала у току процеса обраде. Код другог примера, функције циља су биле максимална сила стезања, уз минималне димензије ексцентра.

Упоређујући резултате за оба примера, може се закључити да MKH алгоритам даје боље резултате од друга два алгоритма.

У примеру оптимизације тела стругарског ножа, MKH алгоритам је дао знатно боље резултате од Loop BFA, односно H-CS-FF алгоритма, Табела 6.1. За разлику од Loop BFA, MKH алгоритам је релативно брзо конвергирао, Слике 6.5 и 6.7, али ка вредности, блиској или једнакој средњој вредности функција циља, да би тек при крају итеративног процеса постигао најбољи резултат.

И за пример оптимизације ексцентра, може се извући сличан закључак. Применом MKH алгоритма добила се највећа сила стезања, док је Loop BFA дао најмањи ексцентар уз задовољавајућу силу стезања, Табела 6.2. Резултати добијени H-CS-FF алгоритмом су лошији од друга два поменута алгоритма. За разлику од Loop BFA, MKH алгоритам је релативно брзо конвергирао ка вредности блиској средњој вредности функције циља, да би тек у последњој трећини итеративног процеса постигао најбољи резултат, Слике 6.8 и 6.10.

Поглавље

7

ЗАКЉУЧАК

7. ЗАКЉУЧАК

Савремена оптимизација подразумева примену модерних метода оптимизације, а нарочито у савременом инжењерству, где се на решавање оптимизационих проблема примењене механике готово увек формирају комплексне, нелинеарне функције циља са великим бројем пројектних промењљивих и функција ограничења. Група најпопуларнијих метода које се данас примењују на пољу решавања оптимизационих проблема су биолошки инспирисани оптимизациони алгоритми.

Хипотеза од које се пошло је да се коришћењем савремених биолошко инспирисаних алгоритама, као и њиховим модификацијама и хибридизацијама, може постићи решење у пољу глобалних минимума за широк спектар оптимизационих проблема примењене механике. Постављена хипотеза је доказана на модификацијима и хибридизацији следећа четири алгоритма: кукавичја претрага (Cuckoo Search – CS), алгоритам свица (Firefly Algorithm – FA), алгоритам слепог миша (Bat Algorithm – BA), оптимизација инспирисана кретањем арктичког крила (Krill Herd Algorithm – KHA).

Приликом увођења модификација у основни оптимизациони алгоритам, у циљу ефикасности рада алгоритма, не треба се водити само ка добијању најбољих резултата већ и ка брзини конвергенције, односно времену које протекне док се не генеришу резултати. У трећем и петом поглављу извршена је модификација стандардног BA, односно KH алгоритма, док је у четвртом извршена хибридизација CS и FA алгоритма, при чему се водило рачуна о претходно поменутом. Наиме, модификовани, односно, хибридни алгоритам морао је да произведе најбоље резултате уз задовољавајућу конвергенцију и за најкраће време рада алгоритма.

Модификација алгоритма слепог миша одвијала се у три корака (поглавље 3). Први корак се односи на израчунавање фреквенције за сваког слепог миша у оквиру једне фамилије. Други корак је увео фамилије слепих мишева, како би се континуално понављала претрага простора могућих решења у циљу добијања оптималног решења. Следећа модификација је омогућила фино циклично претраживање простора решења. За сваког слепог миша, у фамилији, финим претраживањем по кораку Левијевог лета тражило се побољшано решење све док се не задовоље постављена ограничења. Таква решења се упоређују за сваку фамилију слепих мишева и на крају се бира најбоље. Овакав циклични алгоритам фамилија слепих мишева (Loop BFA), тестиран је на 5 референтних инжењерских примера из области примењене механике: суд под притиском, заварени носач, редуктор, конусна опруга и ламеласта кочница са више дискова. У сваком примеру коришћено је 50 фамилија где у свакој фамилији има по 30 јединки. Број итерација је 1000, односно 1500 у неким примерима, по једној фамилији.

Добијена вредност функције циља, Табеле 3.1–3.9., је нижа, у поређењу са вредностима из 21 цитиране референце, исте вредности су добијене као у 15 цитираних референци, а већа вредност је у односу на 4 цитиране референце. Када је у питању стандардна девијација, вредност је нижа у поређењу са 16 наведених извора, мања у поређењу са 9 цитираних резултата.

Хибридни алгоритам, H–CS–FA , предложен у овом раду, за основу има CS алгоритам у који је инкорпориран део FA алгоритма. Наиме, у стандарном CS алгоритму изводи се операција пражњења гнезда, уколико се у итеративном процесу достигне вероватноћа, p_a, проналажења „лоших“ гнезда. Уместо овог дела, у CS алгоритам, се додаје део FA алгоритма у коме се проналази свитац који светли са највећим интезитетом светлости.

Предложени H–CS–FA алгоритам, алгоритам 4.3, је дао боље резултате од резултата добијених стандарним CS, односно, FA алгоритмом, Табеле 4.1, 4.3, 4.5, 4.7 и 4.9. Исправност учињене хибридизације проверена је на следећим примерима примењене механике: модел опруге, модел конусног квачила, модел I профила, модел мењача и модел носача са три променљиве. Такође, новонастали алгоритам дао је боље или бар приближне резултате, за наведене примере, из цитиране литературе, Табеле 4.2, 4.4, 4.6, 4.8 и 4.9. При томе су сва постављена ограничења задовољена.

Хибридизацијом ова два поменута алгоритма, добијене су најбоље карактеристике од оба. Брзи улазак у област оптимума (FA) и боље претраживање тог простора (CS), који је резултирао најбољим решењима. Ово је илустровано и дијаграмима конвергенције за сваки пример, Слике 4.3 – 4.23.

За примену биолошки инспирисаних алгоритама у синтези четворочланих механизама као генератора путање извршена је модификација стандардног KH алгоритма. KH алгоритам, поглавље пет, се базира на понашању арктичког крила приликом окупљања у јато и приликом потраге за храном. У стандардном алгоритму се пре почетка итеративног поступка, дефинише иницијална популација крила и затим се поступак тражења изводи према предложеном алгоритму, алгоритам 5.1.

Модификовани KH алгоритам – MKH представљен је алгоритмом 5.2. Прва модификација која је учињена је да се иницијално одређује позиција плена пре почетка итеративног процеса. Наиме, стандардни KH алгоритам подразумева иницијално одређивање позиције крила у јату узимајући у обзир оптималну густину. Иницијализацијом плена добија се почетно позиционирање крила и у односу на плен. Другим речима, пре почетка итеративног процеса, проверавају се позиције појединачних крила у односу на плен и поређењем позиција, кроз функцију циља, формира се иницијално оптимално решење са којим се улази у итеративни поступак.

Друга модификација се односила на замену генетског оператора укрштања са случајним комбиновањем добијених колона који су добијени у једној итерацији. Пошто је KH алгоритам заснован на трима случајним активностима (кретање изазвано другим појединачним криловима, кретање при потрази за храном и случајна дифузија), постоји опасност да једно од кретања „поквари“ позиције крила у јату и да решење одведе у правцу локалног минимума. Да би се ова опасност избегла уведена је поменута модификација случајног кобиновања колона вектора решења. Након одређивања нових вектора решења (комбиновањем колона) новонастали редослед крила, коригује се за вредност физичке дифузије, чиме се у истој итерацији финије претражује простор решења у циљу побољшања нађеног оптимума.

Исправност и ефикасност предложеног MKH алгоритма је проверена на четири референтна примера из синтезе зглобног четвороугаоног механизма. Резултати добијени овим алгоритмом значајно су бољи у поређењу са резултатима добијеним у цитираној литератури. Грешка, која представља квадрат одступања између задатих тачака и стварних тачака на путањи, у сва четири примера је мања у поређењу са резултатима из цитиране литературе. Нарочито је то изражено у примерима 3 и 4.

У поступку синтезе мора се водити рачуна да механизам који се добије буде изводљив односно да не дође до диспропорције дужина чланова механизма. Однос дужина најдужег и најкраћег члана добијених механизма у свим примерима је веома задовољавајући. MKH поред веома добрих резултата показује и изузетну ефикасност у примени која се огледа у веома брзој конвергенцији приказаној у развоју грешке дуж итеративног процеса, Слике 5.15, 5.12, 5.9 и 5.6. Такође се може уочити да је вредност стандардне девијације за сва четири примера веома мала, што указује да се MKH алгоритмом избегава улазак у простор локалног минимума.

За протврду универзалности предложених биолошки инспирисаних алгоритама Loop BFA, H–CS–FA и MKH, формирана су два нова модела: модел стругарског ножа и модел ексцентра стезног прибора.

На основу добијених резултата, из Табела 6.1 и 6.2, и дијаграма 6.5-6.7, односно 6.8-6.10, може се закључити да наведени алгоритми исправно функционишу. Конвегренција је добра, уз високу вредност стандардне девијације за оба примера и сва три алгоритма. Ово се може објаснити чињеницом да је у питању вишекритеријумска оптимизација, две противуречне функције циља. Наиме, у првом примеру су тражене минималне димензије тела стругарског ножа, уз постизање максималне количине скинутог материјала у току процеса обраде. Код другог примера, функције циља су биле максимална сила стезања, уз минималне димензије ексцентра.

У примеру оптимизације тела стругарског ножа, МКН алгоритам је дао знатно боље резултате од Loop BFA, односно H-CS-FF алгоритма, Табела 6.1. За разлику од Loop BFA, МКН алгоритам је релативно брзо конвергирао, Слике 6.5 и 6.7, али ка вредности, блиској или једнакој средњој вредности, функција циља, да би тек при крају итеративног процеса постигао најбољи резултат.

И за пример оптимизације ексцентра, може се извући сличан закључак. Применом МКН алгоритма добила се највећа сила стезања, док је Loop BFA дао најмањи ексцентар уз задовољавајућу силу стезања, Табела 6.2. Резултати добијени H-CS-FF 0алгоритмом су лошији од друга два поменута алгоритма. За разлику од Loop BFA, МКН алгоритам је релативно брзо конвергирао ка вредности блиској средњој вредности функције циља, да би тек у последњој трећини итеративног процеса постигао најбољи резултат, Слике 6.8 и 6.10.

Примењени модификовани алгоритми, односно хибридизован алгоритам, показали су се ефикасним у решавању оптимизационих проблема примењене механике. Наиме, из приложених резултата може се видети да су добијене вредности функције циља боље од резултата из цитиране литературе, да је конвергенција добра и да је избегнута замка уласка у простор локалног минимума. Универзалност примене је показана при оптимизацији различитих референтних примера из литература као и код новоуведених модела.

ЛИТЕРАТУРА

ЛИТЕРАТУРА

[1] https://sr.wikipedia.org/sr/Инжењерство.

[2] D. Vučina, Metode inženjerske numeričke optimizacije - s primjerima primjene uprogramskom jeziku c i matlab, Fakultet elektrotehnike strojarstva i brodogradnje, Manualia universitatis studiorum spalatensis - Udžbenici sveučilišta u Splitu, Split, 2005.

[3] Р. Р. Булатовић, Истраживање метода синтезе раванских механизама са контролисаним кретањем чланова, докторска дисертација, Машински факултет Београд, Универзитет у Београду, Београд, 2007.

[4] F. Neumann and C. Witt, Bioinspired Computation in Combinatorial Optimization. Natural Computing Series, Springer-Verlag, November 2010.

[5] R. Martini and G. Reinelt, The Linear Ordering Problem Exact and Heuristic Methods in Combinatorial Optimization, vol. 175 of Applied Mathematical Sciences. Springer-Verlag Berlin Heidelberg, 2011.

[6] E. G. Talbi, Metaheuristics: From Design to Implementation. No. 1, Wiley Publishing, July 2009.

[7] I.Fister Jr., Xin-She Yang, I.Fister, J.Brest, D. Fister, A Brief Review of Nature-Inspired Algorithms for Optimization, Elektrotehniški vestnik 80(3), (2013) 116–122.

[8] Xin-She Yang, Review of Metaheuristics and Generalized Evolutionary Walk Algorithm, Int. J. Bio-Inspired Computation, Vol. 3, No. 2, (2011), pp. 77-84.

[9] S. Kirkpatrick, C. D. Gellat, M. P. Vecchi, Optimisation by simulated annealing, Science, 220, (1983), 671-680.

[10] F. Glover, M. Laguna , Tabu Search, Kluwer Academic Publishers, Boston, 1997.

[11] M. Dorigo, Optimisation, Learning and Natural Algorithms, PhD thesis, Politecnico di Milano, Italy, 1992.

[12] J.D. Farmer, N. Packard, A. Perelson, The immune system, adaptation and machine learning, Physica D, 2, (1986), 187-204.

[13] H. Bersini и F. J. Varela, Hints for adaptive problem solving gleaned from immune networks, Parellel Problem Solving from Nature, PPSW1, Dortmund, FRG, (1990).

[14] J. Kennedy, R. Eberhart, Particle swarm optimisation, in: Proc. of the IEEE Int. Conf. on Neural Networks, Piscataway, NJ, (1995), pp. 1942-1948.

[15] Storn, R. and Price, K., Differential evolution - a simple and efficient heuristic for global optimisation over continuous spaces', Journal of Global Optimisation, Vol. 11, (1997), 341-359.

[16] R.Y. Rubinstein, Optimisation of Computer simulation Models with Rare Events, European Journal of Operations Research, 99, (1997) , 89-112.

[17] Z. W. Geem, J. K. Kim, and G. V. Loganathan, A new heuristic optimisation: Harmony search, Simulation, Vol. 76(2), (2001), 60-68.

[18] K. M. Passino, Biomimicry of bacterial foraging for distributed optimisation and control, IEEE Control System Magazine, (2002), pp. 52-67.

[19] S. Nakrani, C. Tovey, On honey bees and dynamic server allocation in Internet hostubg centers, Adaptive Behavior, Vol. 12, C. (2004) , 223-240.

[20] D. T. Pham, A. Ghanbarzadeh, E. Koc, S. Otri, S. Rahim, M. Zaidi, The bees algorithm, Technical Note, Manufacturing Engineering Center, Cardiff University, (2005).

[21] D. Karaboga, An idea based on honey bee swarm for numerical optimisation, Technical Report, Erciyes Univers, (2005).
[22] A. Mucherino, O. Seref, Modelling and solving real-life global optimisation problems with metaheuristic methods', in: Advances in Modeling Agricultural Systems, Springer Optimisation and Its Applications Series Vol. 25 (Eds Papajorgji P. J. and Pardolos P. M.), (2008), pp. 403-420.
[23] X. S. Yang, Nature-Inspired Metaheuristic Algorithms, Luniver Press, (2008). Metaheuristic Algorithms: Optimal Balance of Intensification and Diversification
[24] Yang , X. S., Deb S., Fong, S., Metaheuristic Algorithms: Optimal Balance of Intensification and Diversification, Applied Mathematics & Information Sciences, 8, No. 3, (2014) 977-983.
[25] http://sr.wikipedia.org/Слепи_мишеви
[26] http://www.nhm.ac.uk/
[27] X.S. Yang, Engineering optimization: an introduction with metaheuristic applications, John Wiley & Sons, (2010).
[28] X. S. Yang and A. H. Gandomi, Bat Algorithm: a novel approach for global engineering optimization, Eng Computation. 29 (5) (2012) 464-483.
[29] X. S. Yang, A New Metaheuristic Bat-Inspired Algorithm, in Nature Inspired Cooperative Strategies for Optimization. J. R. Gonzales et al, Eds., Springer Press, 284 (2010) 65-74.
[30] X. S. Yang, BAT Algorithm for multi-objective optimization, Int J Bio-Inspir Com. 3 (5) (2011) 267-274.
[31] S. Yilmaz and E. U. Kucuksille, Improved Bat Algorithm (IBA) on Continuous Optimization Problems, Lec Notes Soft Engin. 1 (3) (2013) 279-283.
[32] O. Hasancebi and T. Teke, A bat-inspired algorithm for structural optimization, Comput Struct. 128 (2013) 77-90.
[33] A. H. Gandomi and X. S. Yang, Chaotic bat algorithm, J Comput Sci-neth. 5 (2) (2014) 224-232.
[34] P. Barthelemy, J. Bertolotti and D.S. Wiersma, A Lévy-flight for light, Nature. 453 (2008) 495-498.
[35] C. Brown, L. S. Liebovitch and R. Glendon, Lévy-flights in Dobe Ju/'hoansi foraging patterns, Hum Ecol. 35 (2007) 129-138.
[36] A. H. Gandomi, X. S. Yang, A. H. Alavi and S. Talatahari S, Bat algorithm for constrained optimization tasks, Neural Comput Appl. 22 (2013) 1239-1255.
[37] H. A. Kayhan, H. Ceylan, T. M. Ayvaz and G. Gurarslan, PSOLVER: A new hybrid particle swarm optimization algorithm for solving continuous optimization problems, Expert Sys Appl.
[38] Jia-ging Zhao, L. Wang, P. Zeng and Wen-hui Fan, An effective hybrid genetic algorithm with flexible allowance technique for constrained engineering design optimization, Ex-pert Sys Appl. 39 (5) (2012) 6041-6051.
[39] B. Akay and D. Karaboga, Artificial bee colony algorithm for large-scale problems and engineering design optimization, J Intell Manuf. 23 (2012) 1001-1014.
[40] V. R. Rao and J. V. Savsani, Teaching-learning-based op-timization: a novel method for constrained mechanical design optimization problems, Comput Aided Design. 43 (3) (2011) 303-315.

[41] A. H. Gandomi, X. S. Yang and A.H. Alavi, Cuckoo search algorithm: a metaheuristic to solve structural optimization problems, Eng Comput-Germany. 29 (2013) 17-35.
[42] Y. Zhou, G. Zhou and J. Zhang, A hybrid glowworm swarm optimization algorithm for constrained engineering de-sign problems, Appl Math Inform Sci. 7 (1) (2013)379-388.
[43] A. H. Gandomi, G. J. Yun, X. S. Yang and S. Talatahari, Chaos-enhanced accelerated particle swarm optimization, Com-mun Nonlinear Sci. 18 (2) (2013) 327-340.
[44] D. Datta and J. R. Fifuerira, A real-integer-discrete-coded particle swarm optimization for design problems, Appl Soft Comput. 11 (4) (2011) 3625-3633.
[45] A. H. Gandomi, X. S.Yang and A.H. Alavi, Mixed variable structural optimization using Firefly Algorithm, Comput Struct. 89 (23-24) (2011) 2325-2336.
[46] A. Sadollah, A. Bahreininejad, H. Eskandar and M. Hamdi, Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems, Appl Soft Comput. 13 (5) (2013) 2592-2612.
[47] Z. Gao, T. Xiao and W. Fan, Hybrid differential evolution and Nelder-Mead algorithm with re-optimization, Soft Comput. 15 (2011) 581-94.
[48] Jian-hua Xio, Yu-fang Huang and Z. Cheng, A bio-inspired algorithm based on membrane computing for engineer-ing design problem, I J Comput Sci. 10 (1) (2013) 580-588.
[49] A.R. Hedar, M. Fukushima, Derivate-Free Filter Simulated Annealing Method for Constrained Continuous Global Optimization, J. Global Optim. 35 (4) (2006) 521-549.
[50] M.J. Kazemzadeh-Parsi, A modified firefly algorithm for engineering design optimization problems, IJST-T Mech Eng, 38 (M2) (2014) 403-421.
[51] C.A.C. Coello, Use of a self-adaptive penalty approach for engineering optimization problems, Comput Ind. 41 (2) (2000) 113-127.
[52] M. Mahdavi, M. Fesanghary and E. Damangir, An im-proved harmony search algorithm for solving optimization prob-lems, Appl Math Comput. 188 (2) (2007) 1567-1579.
[53] M. Jaberipour and E. Khorram, Two improved harmony search algorithms for solving engineering optimization problems, Commun Nonlinear Sci. 15 (11) (2010) 3316-3331.
[54] K. Deb and M. Goyal, A combined genetic adaptive search (GeneAS) for engineering design, Comput Sci Inform. 26 (1996) 30-45.
[55] C.A.C. Coello, Treating constraints as objectives for sin-gle-objective evolutionary optimization, Eng Optimiz. 32 (3) (2000) 275-308.
[56] V. R. Rao and J.V. Savsani, Mechanical design optimizaiton using advanced optimization techniques: Springer-Verlag. London (2012).
[57] Deb and A. Srinivasan, Innovization: innovative design principles through optimization, Kanpur genetic algorithms laboratory (KanGAL). Indian Institute of Technology Kanpur
[58] H. Eskandar, A. Sadollah, A. Bahreininejad and M. Hamdi, Water cycle aglorithm – A novel metaheuristic optimization method for solving constrained engineering optimization problems, Comput Struct. 110-111 (2012) 151-166.
[59] http://www.kakopedija.com/8250/kako-svitac-svetli/
[60] https://en.wikipedia.org/wiki/Cuckoo/
[61] Yang, X.S., Deb, S., Cuckoo search via Levy flights, World Congres on Nature & Biologically Inspired Computing (NaBIC), IEEE Publications, (2009), 210-214.
[62] Yang, X.S., Deb, S., Engineering Optimisation by Cuckoo Search, International Journal of Mathematical Modelling and Numerical Optimisation 2(4), (2010), 330-343.

[63] Sungho M,, Yoon-on C., Modified harmony search optimization for constrained design problems, Expert Systems with Applications, 39, (2012), 419-423
[64] Ramin, R., Cuckoo Optimization Algorithm, Applied Soft Computing, 11, 8, (2011) 5508-5518
[65] Sh. Mashhadi, A. Abshouri, B. Nasiri, M. R. Meybodi, Some hybrid models to improve Firefly algorithm performance, Int. J. Artificial Intelligence, (2012) 8 (S12): 97–117.
[66] A. Kaveh, S. Talatahari, Engineering optimization with hybrid particle swarm and ant colony optimization, Asian journal of civil engineering (building and housing) vol. 10, no. 6, (2009) 611-628
[67] G. Wang., L. Guo., A novel hybrid bat algorithm with harmony search for global numerical optimization, Journal of Applied Mathematics 02/2013, (2013) DOI: 10.1155/2013/696491.
[68] R., V., Rao ,V., J., Savsani, Mechanical Design Optimization Using Advanced Optimization Techniques, Springer-Verlag London, 2012.
[69] X., S., Yang, Firefly Algorithm, L´evy Flights and Global Optimization, Research and Development in Intelligent Systems XXVI (2010), pp 209-218,
[70] E., Bonabeau, M., Dorigo, G., Theraulaz, Swarm Intelligence: From Natural to Artificial Systems, Oxford University Press, (1999)
[71] D. E., Goldberg, Genetic Algorithms in Search, Optimisation and Machine Learning, Reading, Mass.: Addison Wesley (1989).
[72] X., S., Yang, Engineering optimization: An Introduction with Metheuristic Aplications, John Wiley & Sons, Inc., 2010.
[73] X.S. Yang, Firefly algorithms for multimodal optimization, Stoch. Algorithms: Found. Appl. (2009) 169–178.
[74] A. Kaveh, T. Bakhshpoori, E. Afshari, An efficient hybrid Particle Swarm and Swallow Swarm Optimization algorithm, Computers and Structures 143 (2014) 40–59.
[75] R.J.Kuo, Y.H.Lee, F. E. Zulvia, F.C. Tien, Solving bi-level linear programming problem through hybrid of immune genetic algorithm and particle swarm optimization algorithm, Applied Mathematics and Computation 266 (2015) 1013–1026.
[76] M. Kefayat, A. Lashkar Ara, S.A. N. Niaki, A hybrid of ant colony optimization and artificial bee colony algorithm for probabilistic optimal placement and sizing of distributed energy resources, Energy Conversion and Management 92 (2015) 149–161.
[77] X. Yuan, J. Zhao, Y. Yang, Y. Wang, Hybrid parallel chaos optimization algorithm with harmony search algorithm, Applied Soft Computing 17 (2014) 12–22.
[78] M. Balasubbareddy, S. Sivanagaraju, C. V. Suresh, Multi-objective optimization in the presence of practical constraints using non-dominated sorting hybrid cuckoo search algorithm, article in press, Engineering Science and Technology, an International Journal (2015), http://dx.doi.org/10.1016/j.jestch.2015.04.005
[79] F. Fedorik, Using optimization's algorithms by designing of structures, doctoral thesis, Institute of Structural Mechanics, Faculty of Civil Engineering Brno, University of Technology, (2013), 119-140.
[80] S., Aurora, Introduction to optimum design, McGraw-Hill, New York, 1989.
[81] Ad., Belegundu, A study of mathematical programming methods for structural optimization, Ph.D Thesis, Department for Civil and Environmental Engineering, University of Iowa, 1982.

[82] R. R. Bulatović, G. Bošković, M, M. Savković, M. M. Gašić, Improved Cuckoo Search (ICS) algorthm for constrained optimization problems, Latin American Journal of Solids and Structures 11 (2014) 1349-1362.

[83] Mahdavi, M., Fesanghary, M., Damangir, E., An improved harmony serch algorithm for solving optimization problems. Applied Mathematic and Computation, (2007), 188(2), 1567-1579.

[84] Lobato S. F., Valder Steffen Jr., Fish Swarm Optimization Algorithm Applied to Engineering System Dewsign, Latin American Journal of Solids and Structures, (2014), 11, 143-156.

[85] Zou, D., Liu, H., Gao, L., Li., S., Directed searching optimization algorithm for constrained optimization problems. Expert Systems with Applications, (2011), 38(7), 8716-8723.

[86] M. Martínez-Iranzo, J. M. Herrero, J. Sanchis, X. Blasco, S. García-Nieto, Applied Pareto multi-objective optimization by stochastic solvers, Engineering Applications of Artificial Intelligence 22 (2009) 455–465.

[87] X. She Yang, Cuckoo Search and Firefly Algorithm Theory and Applications, Volume 516, Springer International Publishing Switzerland 2014, ISBN: 978-3-319-02140-9 (Print) 978-3-319-02141-6 (Online).

[88] S. Arora, A conceptual comparison of firefly algorithm, bat algorithm and cuckoo search, Control Computing Communication & Materials (ICCCCM), 2013 International Conference on , p.1-4.

[89] W. Long, X. Liang, Y. Huang, Y. Chen, An effective hybrid cuckoo search algorithm for constrained global optimization, Neural Computing and Applications, 2014, Volume 25, Issue 3-4, pp 911-926.

[90] http://www.coolantarctica.com/Antarctica%20fact%20file/wildlife/krill.php

[91] G. N. Sandor, A. G. Erdman, Advanced mechanism design, Analysis and Synthesis, Prentice-Hall, New Jersey, 1984.

[92] I. Ullah, S. Kota, Optimal synthesis of mechanisms for path generation using Fourier descriptors and global search methods, ASME Journal of Mechanical Design 119 (1997), 504-510

[93] A.A. Smaili, N.A. Diab, Naji A Atallah, Optimum synthesis of mechanisms using tabu-gradient search algorithm, ASME Journal of Mechanical Design 127 (2005), 917-923

[94] A.A. Smaili, N.A. Diab, Optimum synthesis of hibrid-task mechanisms using ant-gradient search method, Mech. Mach. Theory 42 (2007) 115-130.

[95] J.M. Hansen, Synthesis of mechanisms using time-varying dimensions, Multibody System Dynamics 7 (2002), 127-144

[96] J. Mariappan, S. Krishnamurty, A generalized exact gradient method for mechanism synthesis, Mech. Mach. Theory 31 (1996) 413-427.

[97] H. Zhou, H. E. Cheung, Optimal synthesis of crank-rocker linkages for path generation using the orientation structural error of the fixed link, Mech. Mach. Theory 36 (2001) 973-982.

[98] A. Kunjur, S. Krishnamurty, Genetic algorithms in mechanical synthesis, Journal of Applied Mechanisms and Robotics 4 (1997) 18-24.

[99] J.A. Cabrera, A. Simon, M. Prado, Optimal synthesis of mechanisms with genetic algorithms, Mech. Mach. Theory 37 (2002) 1165-1177.

[100] M.A. Laribi, A. Mlika, L. Romdhane, S. Zeghloul, A combined genetic algorithm-fuzzy logic method (GA-FL) in mechanisms synthesis, Mech. Mach. Theory 39 (2004) 717-735.
[101] R.R. Bulatović, S. R. Đorđević, On the optimum synthesis of a four-bar linkage using differential evolution and method of variable controlled deviations, Mech. Mach. Theory 44 (2009) 235-246.
[102] J.A. Cabrera, F. Nadal and J.P. Muñoz, A. Simon, Multiobjective constrained optimal synthesis of planar mechanisms using a new evolutionary algorithm, Mech. Mach. Theory 42 (2007) 791-806.
[103] J.A. Cabrera, A. Ortiz, F. Nadal, J.J. Castillo, An evolutionary algorithm for path synthesis of mechanisms, Mech. Mach. Theory (2011) 127-141.
[104] A. Ortiz, J.A. Cabrera, F. Nadal, A. Bonilla, Dimensional synthesis of mechanisms using Differential Evolution with auto-adaptive control parameters, Mech. Mach. Theory 64 (2013) 210-229.
[105] F. Peñuñuri, R. Peón-Escalante, C. Villanueva, D. Pech-Oy, Synthesis of mechanisms for single and hybrid tasks using differential evolution, Mech. Mach. Theory 52 (2011) 1335-1349.
[106] S.B. Matekar, G.R. Gogate, Optimum synthesis of path generating four-bar mechanisms using differential evolution and a modified error function, Mech. Mach. Theory 52 (2012) 158-179.
[107] G. R. Gogate and S. B. Matekar, Optimum synthesis of motion generating four-bar mechanisms using alternate error functions, Mech. Mach. Theory 54 (2012) 41-61.
[108] S.P. Shiakolas, D. Koladiya, J. Kebrle, On the optimum synthesis of six-bar linkages using differential evolution and the geometric centroid of precision positions technique, Mech. Mach. Theory 40 (2005) 319-335.
[109] W.Y. Lin, S.S. Wang, Dimensional synthesis of a five-point double-toggle mould clamping mechanis using a genetic algorithm-differential evolution hybrid algorithm, Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science 224 (2010) 1305-1313.
[110] When-Yi Lin, A GA-DE hybrid evolutionory algorithm for path synthesis of four-bar linkage, Mech. Mach. Theory 45 (2010) 1096-1107.
[111] C.T. Lee, C.C. Lee, On a hybrid particle swarm optimization method and its application in mechanism design, Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science (2014) (In Press) doi:10.1177/0954406214522206, pp.14.
[112] S. K. Acharyya, M. Mandal, Performance of EAs for four-bar linkage synthesis, Mech. Mach. Theory 44 (2009) 1784-1794.
[113] R.R. Bulatović, S. R. Đorđević, V.S. Đorđević, Cuckoo Search algorithm: A metaheuristic approach to solving the problem of optimum synthesis of a six-bar double dwell linkage, Mech. Mach. Theory 61 (2013) 1-13.
[114] A.H. Gandomi, A.H. Alavi, Krill Herd: a new bio-inspired optimization algorithm, Commun. Nonlinear Sci. 17 (2012) 4831–4845.
[115] G. Wang, L. Guo, A. H. Gandomi, L. Cao, A. H. Alavi, H. Duan, J. Li, Lévy-Flight Krill Herd Algorithm, Math. Probl. Eng. 2013 (2013) 1-14.
[116] G. Wang, L. Guo, H. Wang, H. Duan, L. Liu, J. Li, Incorpoprating mutation scheme into krill herd algorithm for global numerical optimization,

[117] S. Saremi, S. M. Mirjalili, S. Mirjalili, Chaotic krill herd optimization algorithm, Procedia Technology 12 (2014) 180-185.
[118] A. H. Gandomi, S. Talatahari, F. Tadbiri, A. H. Alavi, Krill herd algorithm for optimum design of truss structures, Int. J. Bio-Inspired Computation, 5 (2013) 281-288.
[119] J. Li, Y. Tang, C. Hua, X. Guan, An improved krill herd algorithm: Krill Herd with linear decreasing step, Appl. Math. Comput. 234 (2014) 356-367.
[120] W. Gai-Ge, A. H. Gandomi, A. H. Alavi, A chaotic particle-swarm krill herd algorithm for global numerical optimization. Kybernetes 42 (2013) 962 - 978.
[121] F. Freudenstein, An analytical approach to the design of four-link mechanisms, Transactions of the ASME 76 (1954) 483-492.
[122] R. J. McGarva, Rapid search and Selection of Path Generating Mechanisms from a Library, Mech. Mach. Theory 29 (1994) 223-235.
[123] P.G. Petropoulos, Optimal selection of machining rate variable by geometric programming, International Journal of Production Research 11 (4) (1973) 305–314.
[124] Y.C. Shin, Y.S. Joo, Optimization of machining conditions with practical constraints, International Journal of Production Research 30 (12) (1992) 2907–2919.
[125] J.S. Agapiou, The optimisation of machining operations based on a combined criterion Part 2: multipass operations, Journal of Engineering for Industry 114 (1992) 508–513.
[126] E.J.A. Armarego, A.J.R. Simith, J. Wang, Computer-aided constrained optimisation analyses strategies for multipass helical tooth milling operation, Annals of the CIRP 43 (1) (1994) 437–442.
[127] R. Gupta, J.L. Batra, J.K. Lal, Determination of optimal subdivision of depth of cut in multi-pass turning with constraints, International Journal of Production Research 33 (1995) 115–127.
[128] S.E. Kilic, C. Cogun, D.T. Sen, A computer-aided graphical technique for the optimization of machining conditions, Computers in Industry 22 (3) (1993) 319–326.
[129] E.J.A. Armarego, A.J.R. Smith, J. Wang, Constrained optimization strategies CAM software for single-pass peripheral milling, International Journal of Production Research 31 (9) (1993) 2139–2160.
[130] M.T. Rad, I.M. Bidhendi, On the optimization of machining parameters for milling operations, International Journal of Machine Tools & Manufacture 37 (1) (1997) 1–16.
[131] J. Wang, Computer-aided economic optimization of end-milling operations, International Journal of Production Economics 54 (3) (1998) 307–320.
[132] J. Wang, J.A. Armarego, Computer-aided optimization of multiple constraint single pass face milling operations, Machining Science and Technology 5 (1) (2001) 77–99.
[133] J. Wang, T. Kuriyagawa, X.P. Wei, D.M. Guo, Optimization of cutting conditions for single pass turning operations using a deterministic approach, International Journal of Machine Pass and Manufacture 42 (9) (2002) 1023–1033.
[134] Kilickap, E., Huseyinoglu, M., & Yardimeden, A. Optimization of drilling parameters on surface roughness in drilling of AISI 1045 using response surface methodology and genetic algorithm. International Journal of Advanced Manufacturing Technology, 52 (2011). , 79–88.
[135] Kolda, T.G., Lewis, R.M, & Torczon, V., Optimization by direct search: new perspectives on some classical and modern methods. SIAM Review, 45, 385–482.

[136] Ali R. Yildiz, A new hybrid differential evolution algorithm for the selection of optimal machining parameters in milling operations, Applied Soft Computing 13 (2013) 1561–1566
[137] M. Kovačević, M. Madić, & M. Radovanović, Software prototype for validation of machining optimization solutions obtained with meta-heuristic algorithms. Expert Systems with Applications, 40 (2013), 6985–6996.
[138] M. Madić, M. Radovanović, Optimization of machining processes using pattern search algorithm, International Journal of Industrial Engineering Computations 5 (2014) 223–234
[139] Lewis, R.M, Torczon, V., & Trosset, M.W. (2000). Direct search methods: then and now. Journal of Computational and Applied Mathematics, 124, 191–207.
[140] Maji, K., & Pratihar, D. K. (2011). Modeling of electrical discharge machining process using conventional regression analysis and genetic algorithms. Journal of Materials Engineering and Performance, 20, 1121–1127.
[141] Rao, R.V., & Pawar, P.J. (2009). Modelling and optimization of process parameters of wire electrical discharge machining. Proceedings of the Institution of Mechanical Engineers, Journal of Engineering Manufacture 223, 1431–1440.
[142] Rao, R.V., & Pawar, P.J. (2010). Parameter optimization of a multi-pass milling process using non-traditional optimization algorithms. Applied Soft Computing, 10, 445–456.
[143] Rao, R.V. (2011). Advanced modeling and optimization of manufacturing processes: international research and development. London: Springer-Verlag.
[144] Rao, R.V., & Kalyankar, V.D. (2013). Parameter optimization of modern machining processes using teaching–learning-based optimization algorithm. Engineering Applications of Artificial Intelligence, 26, 524–531.
[145] Samanta, S., & Chakraborty, S. (2011). Parametric optimization of some non-traditional machining processes using artificial bee colony algorithm. Engineering Applications of Artificial Intelligence, 24, 946–957.
[146] Savas, V., & Ozay, C. (2008). The optimization of the surface roughness in the process of tangential turn-milling using genetic algorithm. International Journal of Advanced Manufacturing Technology, 37, 335–340.
[147] Torczon, V. (1989). Multi-directional search: a direct search algorithm for parallel machines. PhD thesis, Rice University.
[148] U. Zuperl, F. Cus, Optimization of Cutting Conditions During Machining by Using Neural Networks, International Conference on Flexible Automation and Intelligent Manufacturing 2002, Dresden, Germany
[149] K. Deb, R. Datta, Hybrid Evolutionary Multi-Objective Optimization of Machining Parameters, 2011, KanGAL Report Number 2011005
[150] Torczon, V. (1997). On the convergence of pattern search algorithms. SIAM Journal on Optimization, 7, 1–25.
[151] Wang, Z.G., Wong, Y.S., & Rahman, M. (2004). Optimisation of multi-pass milling using genetic algorithm and genetic simulated annealing. International Journal Advanced Manufacturing Technology, 24, 727–732.
[152] Yildiz, A.R. (2009). An effective hybrid immune-hill climbing optimization approach for solving design and manufacturing optimization problems in industry. Journal of Materials Processing Technology, 209, 2773–2780.

[153] Yildiz, A.R., & Ozturk, F. (2006). Hybrid enhanced genetic algorithm to select optimal machining parameters in turning operation. Proceedings of the Institution of Mechanical Engineers Part B Journal of Engineering Manufacture, 220, 2041–2053.
[154] Yusup, N., Sarkheyli, A., Zain, A.M., Hashim, S.Z.M., & Ithnin, N. (2013). Estimation of optimal machining control parameters using artificial bee colony. Journal of Intelligent Manufacturing, DOI 10.1007/s10845-013-0753-y.
[155] Zain, A.M., Haron, H., & Sharif, S. (2011). Genetic algorithm and simulated annealing to estimate optimal process parameters of the abrasive waterjet machining. Engineering with Computers, 27, 251–259.
[156] Zhang, J.Y., Liang, S.Y., Yao, J., Chen, J.M., & Huang, J.L. (2006). Evolutionary optimization of machining processes. Journal of Intelligent Manufacturing, 17, 203–215.
[157] N. Kaya, Machining fixture locating and clamping position optimization using genetic algorithms, Computers in Industry 57 (2006) 112–120.
[158] М. Калајџић, Милисав Калајџић, Технологија обраде резањем – приручник, Машински факлутет, 2004, Београд
[159] Д. Миликић и други, Технологија обрде резањем, збирка решених и задатака за вежбу, Универзитет у Новом Саду, Факултет техничких наука, 2000, Нови Сад
[160] F. Cus, J. Balic, Optimization of cutting process by GA approach, Robotics and Computer Integrated Manufacturing 19, (2003) 113–121.
[161] Б. Тадић, Алати и прибори, скрипта, Крагујевац, 2008.
[162] Д. Пајић, Конструкција и примена стезних алата, Новинско издавачко предузеће: Техничка књига, Београд, 1967.
[163] https://en.wikipedia.org/wiki/Kronecker_product
[164] G. R. Miodragović, R. R. Bulatović, Loop bat family algorithm (Loop BFA) for constrained optimization, Journal of Mechanical Science and Technology 29 (8) (2015), 3329–3341.
[165] R. R. Bulatović, G. R. Miodragović, M. S. Bošković, Modified Krill Herd (MKH) algorithm and its application in dimensional synthesis of a four-bar linkage, Mechanism and Machine Theory 95 (2016) 1–21, DOI:10.1016 /j.mechmachtheory.2015.08.004