



Универзитет у Београду Машински факултет

Мастер академске студије

Индустрија 4.0

ТЕРМИНИРАЊЕ ТЕХНОЛОШКИХ СИСТЕМА И ПРОЦЕСА

ПРОЈЕКАТ

Оцена проектног задатка:	Предметни наставник:	проф. др Милица Петровић	
	Предметни сарадник:	Александар Јокић, мастер.инж.маш.	
Група: 1			
Потпис наставника:	РБ	Презиме и име:	Бр.инд.
	1.	Дејан Благојевић	4004/2019
	2.	Ивана Дракулић	4008/2020
	3.	Марко Микавица	4007/2020
	4.	Стефан Ковач	4003/2020
	5.	Ана Пауновић	4009/2020
	6.	Невена Стефановић	4003/2019

Школска година: 2021/2022.

**Дејан Благојевић¹, Ивана Дракулић², Марко Микавица³, Стефан Ковач⁴, Ана Пауновић⁵,
Невена Стефановић⁶**

РЕЗИМЕ

Индустрија 4.0 је концепт који се заснива на интелигентним системима. Циљ похађања предмета Терминирање технолошких система и процеса је стицање нових знања из биолошки инспирисаних алгоритама, као и њихова примена у реалним проблемима пре свега у терминирању технолошких система и процеса.

Пројектом је обухваћена анализа одабраних биолошки инспирисаних алгоритама оптимизације и, применом генетичког алгоритма, предложено је решење преоблема пројектовања технолошких процеса и планирања терминирања избраних репрезентативних делова. За имплементацију решења коришћен је софтверски пакет *Matlab*.

Стечено знање ће у многоме унапредити даље усавршавање у правцу концепта индустрије 4.0.

Кључне речи: оптимизација, Генетички алгоритми, Алгоритам симулираног жарења, *Matlab*,

¹ **Дејан Благојевић 4004/2019**, Универзитет у Београду – Машински факултет, студент друге године Мастер академских студија.

Е-пошта: dejan.blagojevic96@gmail.com

² **Ивана Дракулић 4008/2020**, Универзитет у Београду – Машински факултет, студент друге године Мастер академских студија.

Е-пошта: ivana.drakulic@gmail.com

³ **Марко Микавица 4007/2020**, Универзитет у Београду – Машински факултет, студент друге године Мастер академских студија.

Е-пошта: marko_mikavica@yahoo.com

⁴ **Стефан Ковач 4003/2020**, Универзитет у Београду – Машински факултет, студент друге године Мастер академских студија.

Е-пошта: 4003-2020@studenti.mas.bg.ac.rs

⁵ **Ана Пауновић 4009/2020**, Универзитет у Београду – Машински факултет, студент друге године Мастер академских студија.

Е-пошта: apanovic19@gmail.com

⁶ **Невена Стефановић 4003/2019**, Универзитет у Београду – Машински факултет, студент друге године Мастер академских студија.

Е-пошта: nevena.stefanovic88@gmail.com

СПИСАК СЛИКА

Слика 1. Генетички алгоритам.....	7
Слика 2. График функције Ackley	8
Слика 3. Ackley функција у Optimization Toolbox-у	9
Слика 4. Оптимално решење Ackley функција	9
Слика 5. Резултати оптимизације за различите улазне параметре	16
Слика 6. Резултати оптимизације за различите улазне параметре	18
Слика 7. Резултати оптимизације Трговачки путник	20
Слика 8. Алгоритам симулираног жарења [2].....	21
Слика 9. Поређење класичног и СА алгоритма [2].....	22
Слика 10. Schwefel функција	23
Слика 11. Schwefel функција у Optimization Toolbox-у	24
Слика 12. Графици функције Schwefel.....	26
Слика 13. Michalewicz функција.....	27
Слика 14. Оптимално решење Michaelwicz функције - SA.....	31
Слика 15. Промена стандардне девијације	31
Слика 16. Приближавање најбољем агенту [2]	33
Слика 17. Спирално приближавање плену [2].....	34
Слика 18. Компоненте вектора брзине [2]	36
Слика 19. Конусна замка коју мраволовац образује у песку [3].....	37
Слика 20. Ограничења кретања мрава око изабраног мраволовца [3].....	39
Слика 21. График функције „Xin-She Yang“	42
Слика 22. График резултата експеримента за функцију „Xin-She Yang“	43
Слика 23. Boxplot резултати експеримента за функцију „Xin-She Yang“	44
Слика 24. Крива конвергенције јединке са најбољим резултатом за функцију „Xin-She Yang“ ...	44
Слика 25. График функције „Rastrigin“	44
Слика 26. График резултата експеримента за функцију „Rastrigin“	45
Слика 27. Boxplot резултати експеримента за функцију „Rastrigin“	46
Слика 28. Крива конвергенције јединке са најбољим резултатом за функцију „Rastrigin“	46
Слика 29. Подстринг за план терминирања	52
Слика 30. Помочни подстринг за технолошки процес.....	52
Слика 31. Два случајно изабрана хромозама.....	53
Слика 32. Укрштање помоћног подстринга.....	53
Слика 33. Укрштање главног подстринга.....	54
Слика 34. Мутација (двопозициона замена).....	54
Слика 35. Мутација једног алтернативног технолошког процеса	54
Слика 36. Гантов дијаграм за терминирање проблема бр. 6.....	56
Слика 37. Минимална вредност ФЦ кроз итерације за проблем 6	56
Слика 38. Средња вредност ФЦ кроз итерације за проблем 6.....	56
Слика 39. Гантов дијаграм за терминирање проблема бр. 13.....	57
Слика 40. Минимална вредност ФЦ кроз итерације за проблем 13	57
Слика 41. Минимална вредност ФЦ кроз итерације за проблем 13	57
Слика 42. Гантов дијаграм за терминирање проблема бр. 23.....	58
Слика 43. Минимална вредност ФЦ кроз итерације за проблем 23	58
Слика 44. Минимална вредност ФЦ кроз итерације за проблем 23	58

САДРЖАЈ

1. УВОД.....	5
2. ГЕНЕТИЧКИ АЛГОРИТМИ.....	6
2.1 Задатак 1.....	7
2.2 Задатак 2	12
2.3 Задатак 3.....	19
3. АЛГОРИТАМ СИМУЛИРАНОГ ЖАРЕЊА.....	21
3.1 Задатак 1.....	23
3.2 Задатак 2	27
4. АЛГОРИТАМ ИНСПИРИСАН ИНТЕЛИГЕНЦИЈОМ ЈАТА КИТОВА (WOA)	32
5. АЛГОРИТАМ ИНСПИРИСАН ИНТЕЛИГЕНЦИЈОМ РОЈА ЧЕСТИЦА (PSO)	35
6. АЛГОРИТАМ ИНСПИРИСАН ИНТЕЛИГЕНЦИЈОМ МРАВОЛОВАЦА (ALO)	37
7. ЕКСПЕРИМЕНТАЛНО ПОРЕЂЕЊЕ РЕЗУЛТАТА ALO, PSO И WOA.....	42
7.1 Xin-She Yang функција	42
7.2 Rastrigin функција	44
8. ТЕРМИНИРАЊЕ ТЕХНОЛОШКИХ ПРОЦЕСА.....	47
8.1 Флексибилни технолошки процеси.....	47
8.2 Терминирање	50
8.3 Примена генетичких алгоритама у оптимизацији плана терминирања	51
8.3.1 Генерирање јединки у иницијалну популацију	51
8.3.2 Евалуација функције циља и иницијализација параметара генетичких алгоритама ...	52
8.3.3 Селекција.....	53
8.3.4 Укрштање.....	53
8.3.5 Мутација	54
8.4 Експериментални резултати	55
8.4.1 Експериментални резултати за проблем бр. 6 (делови 3-6-9-12-15-18)	56
8.4.2 Експериментални резултати за проблем бр. 13 (делови 2-3-6-9-11-12-15-17-18)	57
8.4.3 Експериментални резултати за проблем бр. 23 (делови 1-4-5-6-7-8-9-11-12-13-14-15-16-17-18)	58
9. ЗАКЉУЧАК	59
10. ЛИТЕРАТУРА	60

1. УВОД

Један од највећих проблема данашњег времена јесте правилно коришћење ресурса, било то време, природни ресурс или искоришћеност машина и алата у производном систему, а све то у сврху остваривање задатог циља. Оне организације које успеју да пронађу начин да своје ресурсе користе на најбољи могући начин биће успешније у динамичном тржишном окружењу данашњице.

Термин оптимизација потиче од латинске речи *optimum* чије је значење „оно што је најбоље“, а процес оптимизације представља одабир таквих улаза који дају најбољи могући излаз из система. Може се рећи да оптимизација представља процес којим се остварује најбољи резултат у одређеним околностима и под датим ограничењима система. Приликом пројектовања, изградње, рада као и одржавања система неопходно је доносити одлуке чији је крајњи циљ минимизација трошкова путем минимизације улазних ресурса, односно максимизација добити која се остварује.

Потребни улази у систем и жељени излаз из система могу се изразити као функција одређених променљивих те се оптимизација може дефинисати као процес проналажења оних услова који дају максималну или минималну вредност функције.

За решавање различитих проблема оптимизације развијен је велики број метода, неке од тих метода су [1]:

- Градијентне методе (енгл. *Gradient Based Methods*);
- Њутнов метод (енгл. *Newton's Method*);
- Градијентни метод најбржег спуста (енгл. *Steepest Descent Method*);
- Линијска претрага (енгл. *Line Search*);
- Конјуговани градијентни метод (енгл. *Conjugate Gradient Method*);
- Стохастички градијентни метод (енгл. *Stochastic Gradient Descent*).

Последњих деценија развијају се посебне методе оптимизације које су инспирисане појавама које се могу наћи у природи. Ове методе, називају се и еволуциони алгоритми или еволуционо програмирање, су се показале као изузетно успешне приликом решавања различитих проблема оптимизације. Најпознатије оптимизационе методе инспирисане природним процесима су:

- Генетички алгоритми (енгл. *Genetic algorithms GA*);
- Алгоритам симулираног жарења (енгл. *Simulated annealing SA*);
- Табу претрага (енгл. *Tabu search*);
- Различити алгоритми инспирисани интелигенцијом роја.

2. ГЕНЕТИЧКИ АЛГОРИТМИ

Један од најпознатијих и најстаријих биолошки инспирисаних алгоритама је генетички алгоритам. Генетички алгоритам (енгл. *Genetic Algorithm* – GA) је претраживачка метахеуристика која опонаша процес природне селекције. Уз помоћ генетичког алгоритма генеришу се решења за оптимизацију проблема коришћењем техника које су инспирисане природном еволуцијом, као што су наслеђивање, укрштање, мутација и селекција. Користи се за решавање различитих проблема, међутим најчешћа примена је у оптимизацији и проблемима претрага.

Како би се генетички алгоритми могли прецизно објаснити потребно је додатно појашњење терминологије која стоји иза њих.

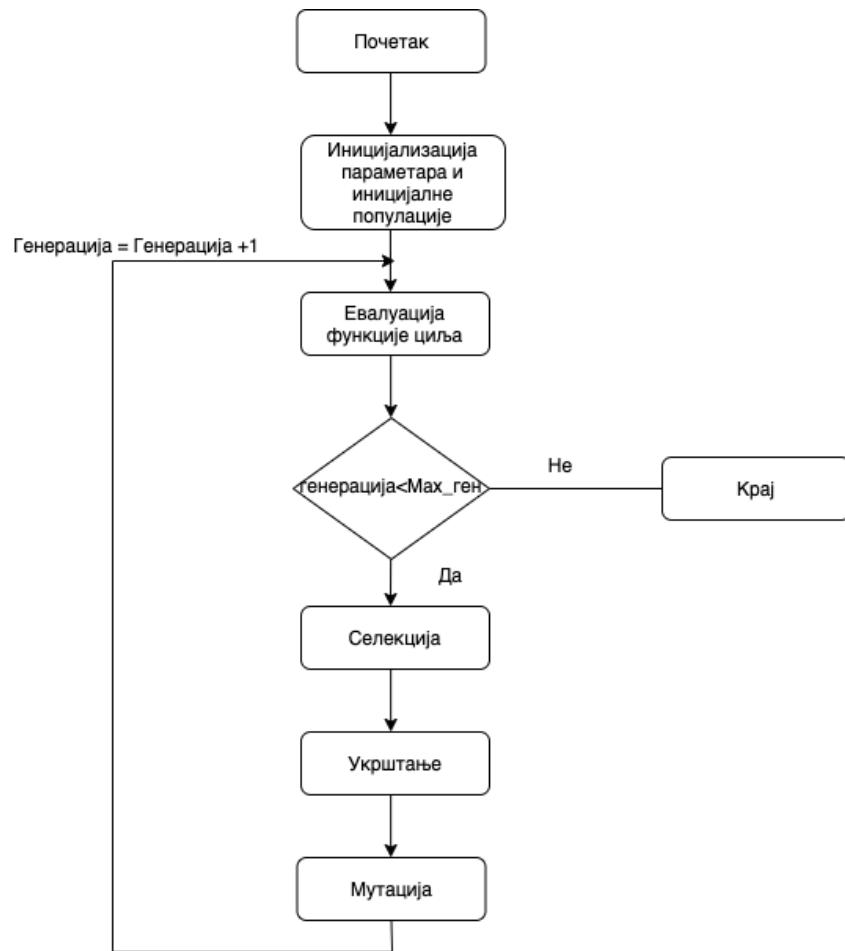
- **Јединке** које представљају могуће решење оптимизационог проблема, се кодирају као хромозоми.
- **Ген** је основни елемент хромозома који се традиционално кодира као бинарни стринг фиксне дужине.
- **Популација** је група свих јединки на основу којих се врши оптимизација.

Генетички алгоритми се примењују у виду компјутерске симулације процеса еволуције популације јединки кроз генерације и то пролазећи кроз следеће процесе:

1. **Иницијална популација** се генерише на случајан начин, али тако да покрива цео простор могућих решења.
2. Врши се **евалуација функције циља** за генерисане јединке. Циљ евалуације је да се додели вредност функције циља, на основу које се рангирају а потом и одабиру јединке за репродукцију.
3. **Селекција** је одабир два родитеља (енгл. *parents*) из тренутне популације на основу вредности функције циља. У оквиру овог пројекта коришћена је *Рулет селекција*, која се заснива на приступу по којем је одабир појединача (из целокупне популације) у директној сразмери са њиховом функцијом циља. Након селекције следи укрштање.
4. **Укрштање** је суштинска компонента у генетичким алгоритмима. Представља основни оператор за стварање нових хромозома тј. потомака (енгл. *offsprings*). Омогућавају спајање јединки које се налазе у различитим деловима простора претраге, тиме се постиже да претрага не буде локална.
5. **Мутације** се примењују случајно са ниском вероватноћом, и то у облику насумичних варијација неких, насумично одабраних гена.
6. **Размена генерација** значи да су јединке тренутне генерације замењене једнаким бројем генерисаних потомака насталих применом селекције, укрштања и мутације.
7. **Елитизам** служи за обезбеђивање конвергенције ка оптималном решењу. Функционише тако што се једном или већем броју јединики са највећом функцијом циља дозволи да опстане непромењена у следећој генерацији.

Представљена процедура извршава се док се не постигне оптимално решење или док се не оствари задати број генерација.

Генетички алгоритам се може приказати дијагромом тока датим на слици 1.



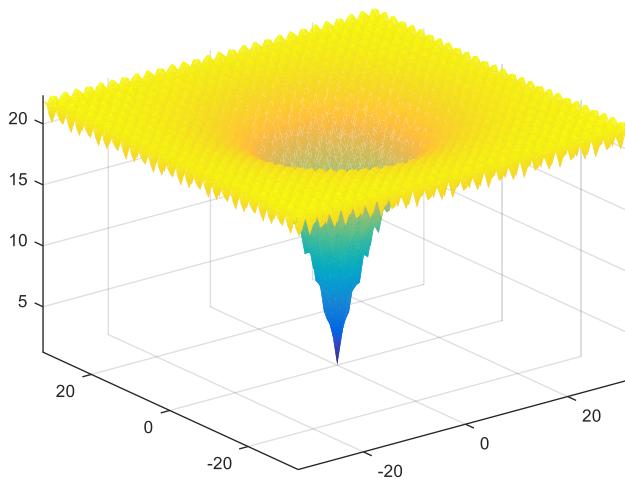
Слика 1. Генетички алгоритам

2.1 Задатак 1

Ackley функција се често користи за тестирања различитих алгоритама који се користе за решавање проблема оптимизације. Објавио је Дејвид Ацкли у својој докторској дисертацији „A connectionist machine for genetic hillclimbing“ 1987. године. Функција чији је минимум потребно пронаћи дата је једначином (1), њен график дат је сликом 2.

$$f(\mathbf{x}) = -20 \cdot e^{\left[-\frac{1}{5} \sqrt{\frac{1}{n} \sum_{i=1}^n x_i^2} \right]} - e^{\left[\frac{1}{n} \sum_{i=1}^n \cos(2\pi x_i) \right]} + 20 + e \quad (1)$$

$$n = 2, \quad -32.768 \leq x_i \leq 32.768$$



Слика 2. График функције Ackley

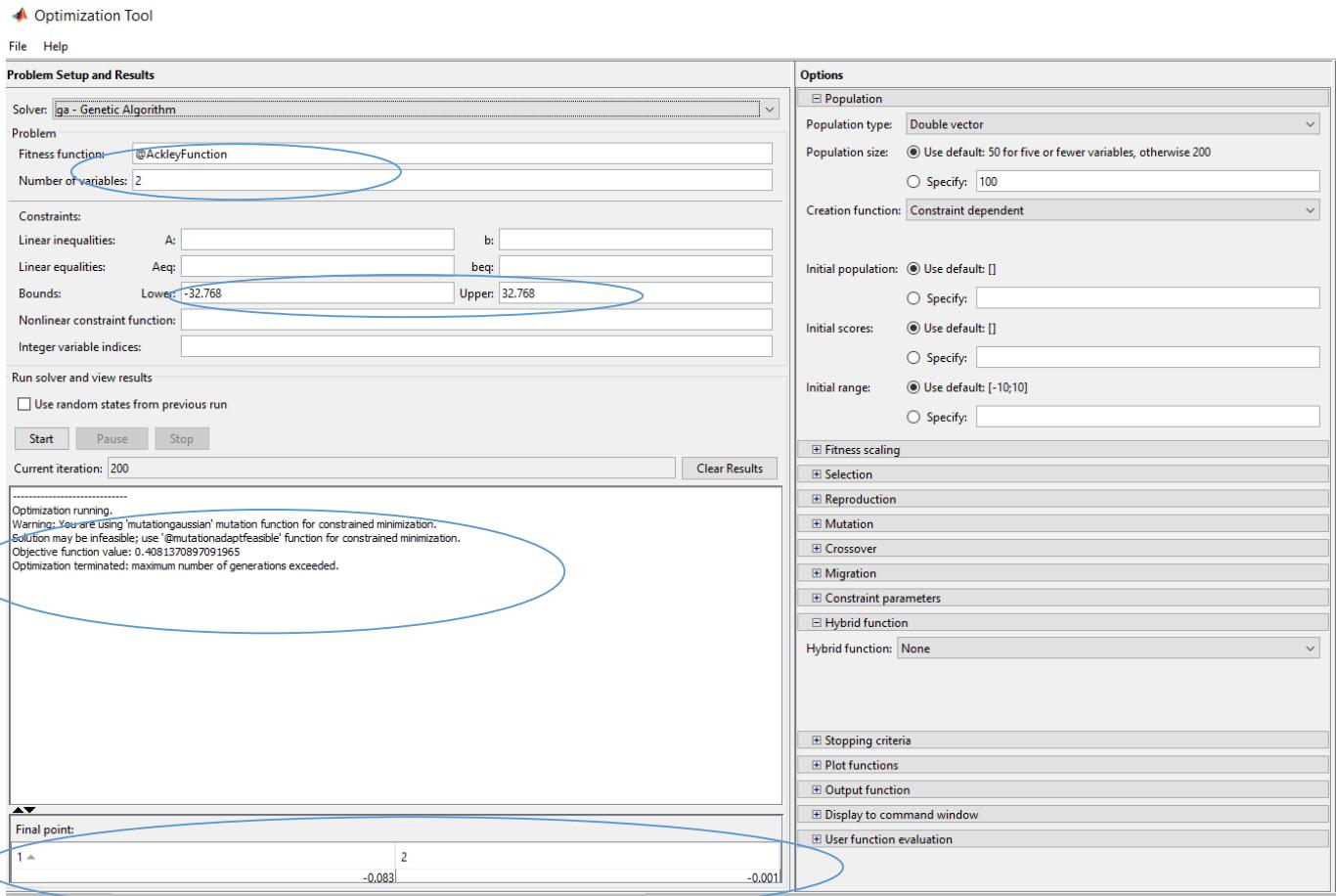
Минимизација функције извршена је применом коришћења софтверског алата *Optimization Toolbox*-а. Функција циља је дата у скрипти испод.

```
function [f] = AckleyFunction(ulaz)

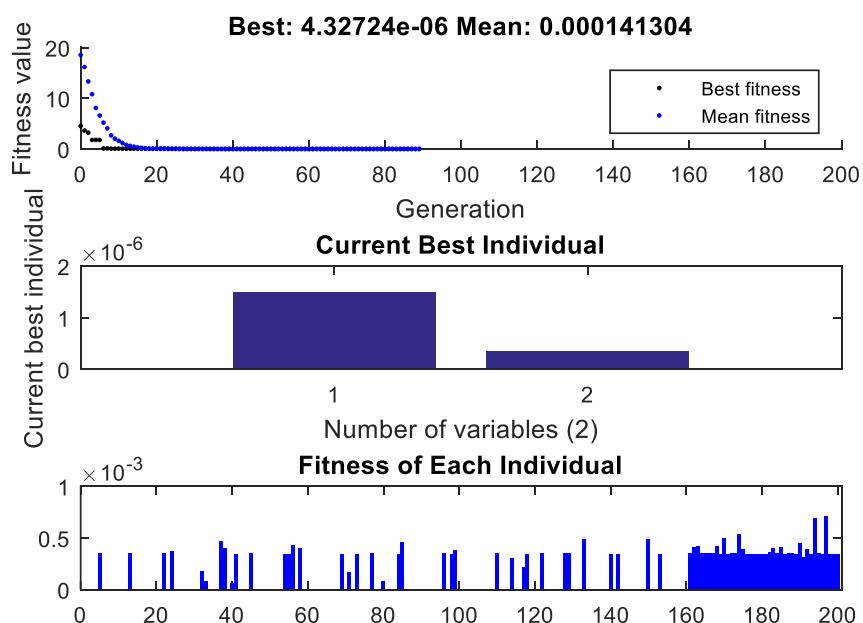
n = 2;
x = ulaz;
f = -20*exp(-0.2*sqrt(sum(x.^2, 2)/n)) - exp(sum(cos(2*pi*x), 2)/n) + exp(1) + 20;

end
```

На слици 3 приказан је *Optimization Toolbox* за примену генетичког алгоритма, *toolbox* је алат који је врло једноставан за коришћење. Након што се дефинише функција циља (енгл. *Fitness function*) уносе се различите опције којима се између осталог могу дефинисати максимални број генерација, величина популације, вероватноћа укрштања, вероватноћа мутације и број елитних хромозома. Резултати извршавања оптимизације су дати у празном прозору као *Objective function value*, а вредности x_1 и x_2 у делу *Final point*. Дијаграми добијених вредности дати су на слици 4.



Слика 3. Ackley функција у Optimization Toolbox-у



Слика 4. Оптимално решење Ackley функција

Промена параметара варира и вредност решења функције. Табелом 1 дата су решења добијена изменом параметара:

- Максимални број генерација,
- Величина популације,
- Вероватноћа укрштања,
- Вероватноћа мутације,
- Број елитних хромозома.

Најбоље решење добијено је са пареметрима датим под редним бројем 63 и у табели је приказано жутом бојом.

Табела 1. Табела са резултатима оптимизације за различите параметре Ackley функције

P. Бр.	Број ген.	Величина поп.	Вер. укрштања	Функција мутације	Вер. мутације	Бр. елит. хромозом	x1	x2	f(x)
1	100	30	0.8	@mutationuniform	0.05	1	-0.8370344	-0.0644141	2.9036097
2	100	30	0.8	@mutationuniform	0.05	5	0.1722905	-0.3841276	3.0037883
3	100	30	0.8	@mutationuniform	0.1	1	0.3359916	0.2188439	2.9686663
4	100	30	0.8	@mutationuniform	0.1	5	-0.1636119	0.1437158	1.5602861
5	100	30	0.8	@mutationadaptfe asible	0.05	1	-0.0000001	0.0000001	0.0000005
6	100	30	0.8	@mutationadaptfe asible	0.05	5	0.0000035	0.0000005	0.0000100
7	100	30	0.8	@mutationadaptfe asible	0.1	1	0.0000086	0.0000019	0.0000248
8	100	30	0.8	@mutationadaptfe asible	0.1	5	0.0000030	-0.0000030	0.0000120
9	100	30	0.6	@mutationuniform	0.05	1	-0.0165909	0.2789376	1.9911005
10	100	30	0.6	@mutationuniform	0.05	5	0.9190711	-0.0476336	2.6634608
11	100	30	0.6	@mutationuniform	0.1	1	0.4321886	0.1724636	3.1905275
12	100	30	0.6	@mutationuniform	0.1	5	0.0888237	0.0325865	0.4906006
13	100	30	0.6	@mutationadaptfe asible	0.05	1	-0.0000011	-0.0000027	0.0000083
14	100	30	0.6	@mutationadaptfe asible	0.05	5	0.0000030	-0.0000006	0.0000087
15	100	30	0.6	@mutationadaptfe asible	0.1	1	-0.0000037	0.0000001	0.0000104
16	100	30	0.6	@mutationadaptfe asible	0.1	5	0.0000014	0.0000056	0.0000162
17	100	100	0.8	@mutationuniform	0.05	1	-0.1661513	0.0849228	1.2615305
18	100	100	0.8	@mutationuniform	0.05	5	-0.4400818	0.0055622	2.8898500
19	100	100	0.8	@mutationuniform	0.1	1	0.1660315	0.0389872	1.1056546
20	100	100	0.8	@mutationuniform	0.1	5	0.1343333	0.0105446	0.7998888
21	100	100	0.8	@mutationadaptfe asible	0.05	1	0.0000062	0.0000004	0.0000174
22	100	100	0.8	@mutationadaptfe asible	0.05	5	0.0000017	-0.0000013	0.0000062
23	100	100	0.8	@mutationadaptfe asible	0.1	1	0.0000033	0.0000018	0.0000107
24	100	100	0.8	@mutationadaptfe asible	0.1	5	0.0000027	0.0000018	0.0000091
25	100	100	0.6	@mutationuniform	0.05	1	0.3560985	-0.0352510	2.5096628
26	100	100	0.6	@mutationuniform	0.05	5	-0.8664159	-0.0614216	2.8108002
27	100	100	0.6	@mutationuniform	0.1	1	-0.0824881	0.0154155	0.4146703
28	100	100	0.6	@mutationuniform	0.1	5	0.1421270	0.0092159	0.8631135

29	100	100	0.6	@mutationadaptfe asible	0.05	1	-0.0000026	0.0000009	0.0000077
30	100	100	0.6	@mutationadaptfe asible	0.05	5	0.0000016	-0.0000002	0.0000044
31	100	100	0.6	@mutationadaptfe asible	0.1	1	0.0000004	0.0000008	0.0000026
32	100	100	0.6	@mutationadaptfe asible	0.1	5	0.0000001	0.0000007	0.0000021
33	200	30	0.8	@mutationuniform	0.05	1	-0.1558074	0.1188651	1.3573431
34	200	30	0.8	@mutationuniform	0.05	5	-0.2575232	0.3862629	3.3197169
35	200	30	0.8	@mutationuniform	0.1	1	0.9250589	-0.0041790	2.5969646
36	200	30	0.8	@mutationuniform	0.1	5	-0.8441341	-0.0426699	2.8312964
37	200	30	0.8	@mutationadaptfe asible	0.05	1	0.0000015	0.0000023	0.0000078
38	200	30	0.8	@mutationadaptfe asible	0.05	5	0.0000026	0.0000005	0.0000075
39	200	30	0.8	@mutationadaptfe asible	0.1	1	0.0000040	0.0000005	0.0000114
40	200	30	0.8	@mutationadaptfe asible	0.1	5	0.0000036	0.0000175	0.0000506
41	200	30	0.6	@mutationuniform	0.05	1	0.3142883	-0.5807496	3.9707320
42	200	30	0.6	@mutationuniform	0.05	5	-0.7096876	0.0804801	3.2734391
43	200	30	0.6	@mutationuniform	0.1	1	0.0829176	-0.0710187	0.6024290
44	200	30	0.6	@mutationuniform	0.1	5	-0.0167679	-0.0207083	0.0941822
45	200	30	0.6	@mutationadaptfe asible	0.05	1	-0.0000016	0.0000024	0.0000080
46	200	30	0.6	@mutationadaptfe asible	0.05	5	0.0000028	0.0000028	0.0000111
47	200	30	0.6	@mutationadaptfe asible	0.1	1	-0.0000025	-0.0000027	0.0000103
48	200	30	0.6	@mutationadaptfe asible	0.1	5	-0.0000044	0.0000117	0.0000354
49	200	100	0.8	@mutationuniform	0.05	1	0.1497591	-0.0732504	1.0831714
50	200	100	0.8	@mutationuniform	0.05	5	-0.0129524	0.1244876	0.7224139
51	200	100	0.8	@mutationuniform	0.1	1	0.0986375	0.0189581	0.5322243
52	200	100	0.8	@mutationuniform	0.1	5	-0.0869566	0.1294945	0.9934876
53	200	100	0.8	@mutationadaptfe asible	0.05	1	0.0000007	0.0000040	0.0000114
54	200	100	0.8	@mutationadaptfe asible	0.05	5	0.0000009	0.0000004	0.0000029
55	200	100	0.8	@mutationadaptfe asible	0.1	1	0.0000027	0.0000202	0.0000577
56	200	100	0.8	@mutationadaptfe asible	0.1	5	0.0000020	-0.0000027	0.0000095
57	200	100	0.6	@mutationuniform	0.05	1	-0.0420790	-0.0536589	0.3129144
58	200	100	0.6	@mutationuniform	0.05	5	-0.0326256	0.0129605	0.1318098
59	200	100	0.6	@mutationuniform	0.1	1	0.0287654	0.0084155	0.1085241
60	200	100	0.6	@mutationuniform	0.1	5	-0.0295236	-0.0063625	0.1095349
61	200	100	0.6	@mutationadaptfe asible	0.05	1	-0.0000008	0.0000001	0.0000024
62	200	100	0.6	@mutationadaptfe asible	0.05	5	0.0000002	-0.0000008	0.0000022
63	200	100	0.6	@mutationadaptfe asible	0.1	1	0.0000003	0.0000002	0.0000010
64	200	100	0.6	@mutationadaptfe asible	0.1	5	0.0000064	-0.0000006	0.0000181

Након спроведеног експеримента може се закључити:

- Мутација типа *Adaptive feasible* даје најбоље резултате;
- Повећањем величине популације резултат је све бољи;
- Проценат елитних хромозома утиче на резултат, када је проценат елитних хромозома нижи резултат је бољи;
- Повећањем максималног броја генерација није се значајно утицало на резултат.

Adaptive feasible мутација најбоље генерише правце који се прилагођавају у односу на последњу генерацију. Дужина корака се бира тако да су задовољена сва ограничења као и постављене границе.

2.2 Задатак 2

Rosenbrock функција је функција уведена од стране Houarda Rosenbrock 1960. године. Ова функција је највећу примену пронашла за тестирање различитих алгоритама оптимизације. Функција чији минимум тражимо дата је једначином (2), а график је дат на слици 5.

$$f(\mathbf{x}) = \sum_{i=1}^{n-1} \left[(x_i - 1)^2 + 100(x_{i+1} - x_i^2)^2 \right] \quad (2)$$

$$n = 2, -5 \leq x_i \leq 5$$

Извршено је 216 покретања алгоритма оптимизације са варирањем следећих параметара:

- број генерација - 10, 50, 100 или 500;
- величина популације - 5, 10 или 20;
- број елитних - 1, 2 или 3;
- вењроватноћа мутације 0.1 и 0.2;
- вероватноћа укрштања 0.9, 0.7 и 0.5.

У Табели 2 су експериментални подаци добијени применом различитих параметара, најбољи резултати су обележени жутом бојом, на сликама 5 и 6 дати су резултати извршавања алгоритма.

Табела 2. Табела са резултатима оптимизације за различите параметре *Rosenbrock* функција

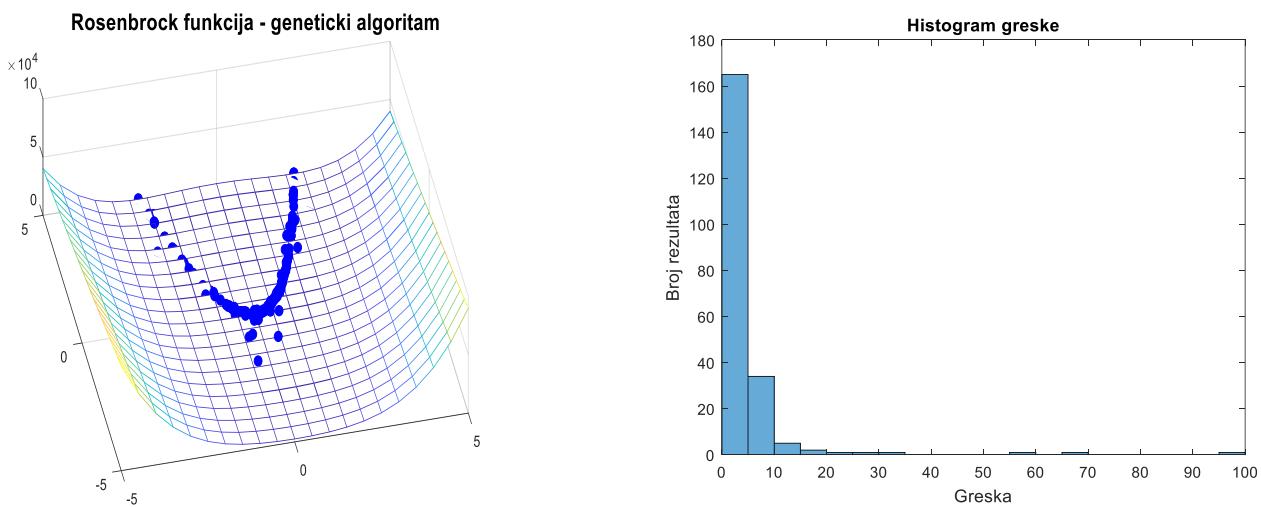
P6	Величина популације	Број генерација	Број елитних	Вероватноћа мутације	Вероватноћа укрштања	X1	X2	Y
1	5	10	1	0,1	0,9	0,52	0,36	1,033216
2	5	10	1	0,1	0,7	1,04	1,28	3,937856
3	5	10	1	0,1	0,5	0,44	-1	142,781696
4	5	10	1	0,2	0,9	1,6	2,6	0,52
5	5	10	1	0,2	0,7	-0,32	-1,8	363,654976
6	5	10	1	0,2	0,5	2,36	3,52	421,935616
7	5	10	2	0,1	0,9	-1,96	4,32	31,648256
8	5	10	2	0,1	0,7	-1,64	2,56	8,649216
9	5	10	2	0,1	0,5	1,36	2,2	12,407616
10	5	10	2	0,2	0,9	-0,24	-0,76	68,384576
11	5	10	2	0,2	0,7	-1,68	2,88	7,514176
12	5	10	2	0,2	0,5	0,88	0,72	0,310336
13	5	10	3	0,1	0,9	-1,16	1,72	18,683136

14	5	10	3	0,1	0,7	0,68	0,48	0,133376
15	5	10	3	0,1	0,5	-1,52	2,16	8,612416
16	5	10	3	0,2	0,9	0,68	-0,04	25,342976
17	5	10	3	0,2	0,7	-2,16	2,88	328,822336
18	5	10	3	0,2	0,5	0,4	0,2	0,52
19	10	10	1	0,1	0,9	0,72	0,56	0,251456
20	10	10	1	0,1	0,7	-1,72	3,04	8,064256
21	10	10	1	0,1	0,5	1,72	2,2	58,035456
22	10	10	1	0,2	0,9	1,32	1,76	0,133376
23	10	10	1	0,2	0,7	0,12	0	0,795136
24	10	10	1	0,2	0,5	0,76	0,64	0,446976
25	10	10	2	0,1	0,9	0,08	0,04	0,959296
26	10	10	2	0,1	0,7	-0,08	-0,24	7,237696
27	10	10	2	0,1	0,5	1,52	2,72	17,047616
28	10	10	2	0,2	0,9	-1,72	2,96	7,398656
29	10	10	2	0,2	0,7	1,04	1,08	0,001856
30	10	10	2	0,2	0,5	-0,32	0,12	1,773376
31	10	10	3	0,1	0,9	-0,36	-0,84	95,862016
32	10	10	3	0,1	0,7	-1,24	1,6	5,406976
33	10	10	3	0,1	0,5	0,28	0,08	0,518656
34	10	10	3	0,2	0,9	-1,56	2,44	6,557696
35	10	10	3	0,2	0,7	-2,08	4,2	11,084096
36	10	10	3	0,2	0,5	0,48	0,28	0,516416
37	20	10	1	0,1	0,9	-0,96	1,04	5,243456
38	20	10	1	0,1	0,7	-0,84	0,72	3,406336
39	20	10	1	0,1	0,5	1,64	2,68	0,418816
40	20	10	1	0,2	0,9	-0,68	0,44	2,872576
41	20	10	1	0,2	0,7	1	1	0
42	20	10	1	0,2	0,5	2	4	1
43	20	10	2	0,1	0,9	0,72	0,48	0,225856
44	20	10	2	0,1	0,7	-0,52	0,32	2,556416
45	20	10	2	0,1	0,5	1,8	3,2	0,8
46	20	10	2	0,2	0,9	1,12	1,28	0,079936
47	20	10	2	0,2	0,7	0,08	0	0,850496
48	20	10	2	0,2	0,5	-0,96	0,92	3,841856
49	20	10	3	0,1	0,9	1,2	1,44	0,04
50	20	10	3	0,1	0,7	-2,24	5	10,528576
51	20	10	3	0,1	0,5	0,88	0,76	0,035136
52	20	10	3	0,2	0,9	-1,84	3,28	9,180736
53	20	10	3	0,2	0,7	-1,04	1,12	4,309056
54	20	10	3	0,2	0,5	0,72	0,52	0,078656
55	5	50	1	0,1	0,9	-0,52	0,36	3,113216
56	5	50	1	0,1	0,7	-0,36	0,16	1,942016
57	5	50	1	0,1	0,5	2,16	4,68	1,366336
58	5	50	1	0,2	0,9	2,16	4,68	1,366336
59	5	50	1	0,2	0,7	1,72	2,92	0,665856
60	5	50	1	0,2	0,5	-1,84	3,32	8,495936
61	5	50	2	0,1	0,9	0,2	0	0,8
62	5	50	2	0,1	0,7	0,04	-0,24	6,758656
63	5	50	2	0,1	0,5	-1,16	1,64	13,332736
64	5	50	2	0,2	0,9	0,84	0,6	1,140736
65	5	50	2	0,2	0,7	0,04	0	0,921856
66	5	50	2	0,2	0,5	2	4	1
67	5	50	3	0,1	0,9	0,84	0,72	0,046336
68	5	50	3	0,1	0,7	1,68	3,04	5,197376
69	5	50	3	0,1	0,5	2,24	5	1,568576

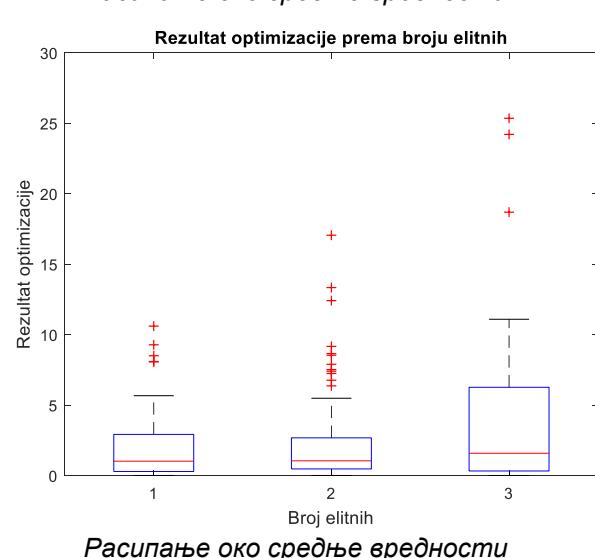
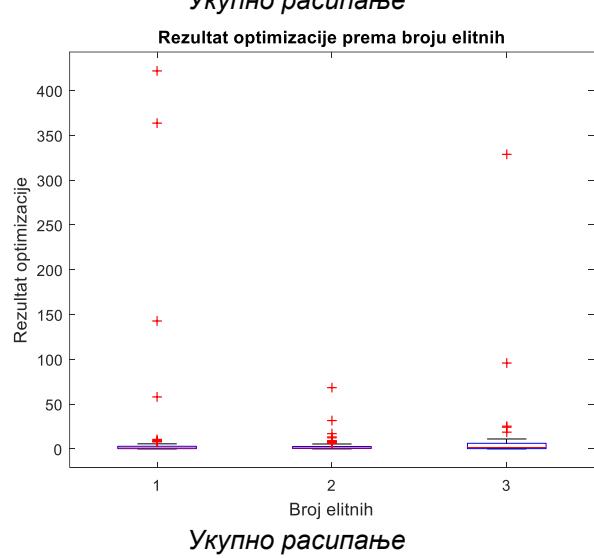
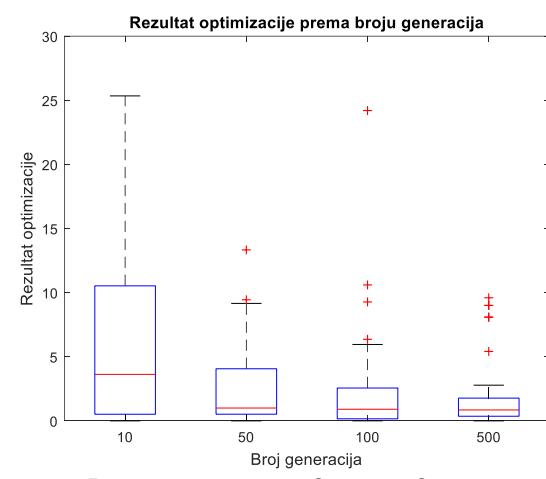
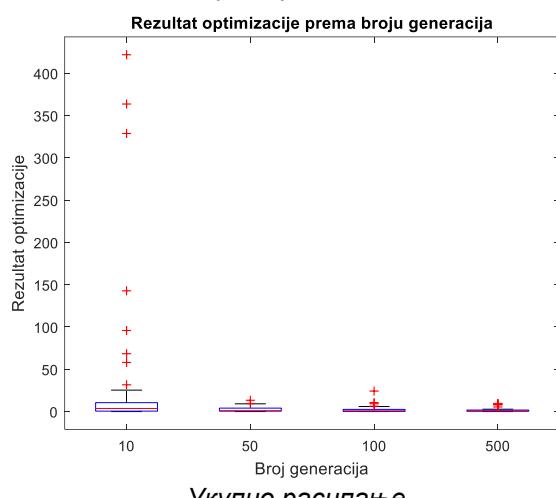
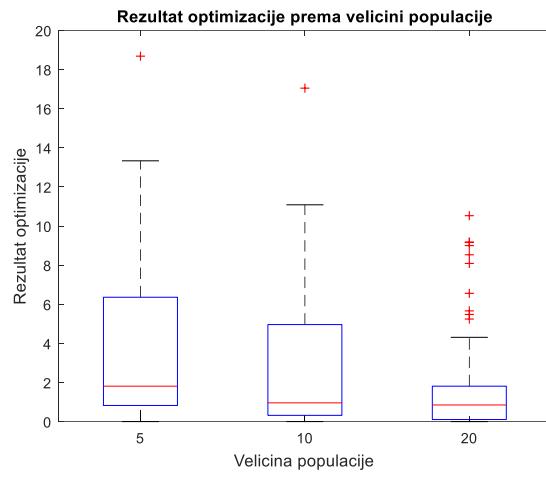
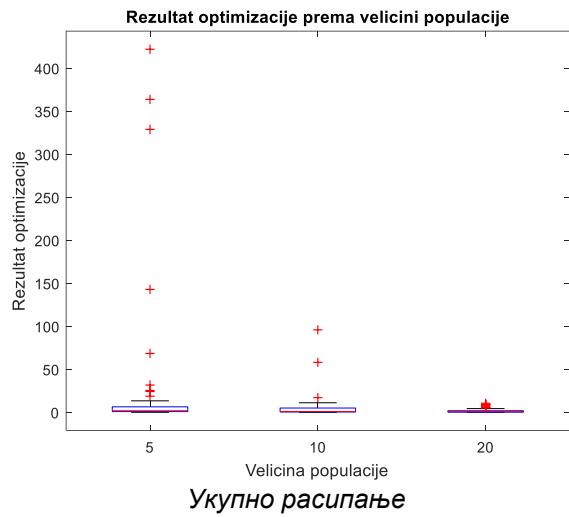
70	5	50	3	0,2	0,9	1,88	3,24	9,441536
71	5	50	3	0,2	0,7	1,8	3,24	0,64
72	5	50	3	0,2	0,5	0,44	0	4,061696
73	10	50	1	0,1	0,9	0,52	0,24	0,322816
74	10	50	1	0,1	0,7	-1,04	1,08	4,161856
75	10	50	1	0,1	0,5	0,24	0,04	0,608576
76	10	50	1	0,2	0,9	0,64	0,44	0,222016
77	10	50	1	0,2	0,7	-0,4	0,16	1,96
78	10	50	1	0,2	0,5	1,48	2,2	0,239616
79	10	50	2	0,1	0,9	-0,56	0,08	7,890496
80	10	50	2	0,1	0,7	1,2	1,4	0,2
81	10	50	2	0,1	0,5	-0,12	0	1,275136
82	10	50	2	0,2	0,9	0,32	0,08	0,512576
83	10	50	2	0,2	0,7	0,64	0,44	0,222016
84	10	50	2	0,2	0,5	0	0,12	2,44
85	10	50	3	0,1	0,9	-1,84	3,4	8,086336
86	10	50	3	0,1	0,7	0	0	1
87	10	50	3	0,1	0,5	0,92	0,76	0,752896
88	10	50	3	0,2	0,9	-0,4	0,16	1,96
89	10	50	3	0,2	0,7	-0,24	0,12	1,926976
90	10	50	3	0,2	0,5	0,28	0,04	0,665856
91	20	50	1	0,1	0,9	-1,36	1,88	5,662016
92	20	50	1	0,1	0,7	0,56	0,32	0,197696
93	20	50	1	0,1	0,5	-0,28	0	2,253056
94	20	50	1	0,2	0,9	0,08	0	0,850496
95	20	50	1	0,2	0,7	1,72	2,96	0,518656
96	20	50	1	0,2	0,5	0,08	0	0,850496
97	20	50	2	0,1	0,9	0,24	0,04	0,608576
98	20	50	2	0,1	0,7	0,28	0,08	0,518656
99	20	50	2	0,1	0,5	-2	4,04	9,16
100	20	50	2	0,2	0,9	1	1	0
101	20	50	2	0,2	0,7	-1,92	3,68	8,530496
102	20	50	2	0,2	0,5	1,64	2,68	0,418816
103	20	50	3	0,1	0,9	0,08	0,04	0,959296
104	20	50	3	0,1	0,7	-1,84	3,4	8,086336
105	20	50	3	0,1	0,5	0,48	0,24	0,279616
106	20	50	3	0,2	0,9	0,72	0,52	0,078656
107	20	50	3	0,2	0,7	-1,56	2,44	6,557696
108	20	50	3	0,2	0,5	1	1	0
109	5	100	1	0,1	0,9	-1,92	3,6	9,272896
110	5	100	1	0,1	0,7	1,12	1,24	0,035136
111	5	100	1	0,1	0,5	1,28	1,72	0,744256
112	5	100	1	0,2	0,9	-0,72	0,52	2,958656
113	5	100	1	0,2	0,7	-0,36	0,12	1,858816
114	5	100	1	0,2	0,5	-1,24	1,56	5,067776
115	5	100	2	0,1	0,9	-1,28	1,68	5,371456
116	5	100	2	0,1	0,7	2,24	5	1,568576
117	5	100	2	0,1	0,5	1,24	1,52	0,088576
118	5	100	2	0,2	0,9	-0,64	0,44	2,782016
119	5	100	2	0,2	0,7	-0,28	0,12	1,811456
120	5	100	2	0,2	0,5	1,4	1,92	0,32
121	5	100	3	0,1	0,9	1,96	3,76	1,587456
122	5	100	3	0,1	0,7	0	-0,08	1,64
123	5	100	3	0,1	0,5	-1,44	2,08	5,957696
124	5	100	3	0,2	0,9	-1,2	1	24,2
125	5	100	3	0,2	0,7	0,24	0,08	0,627776

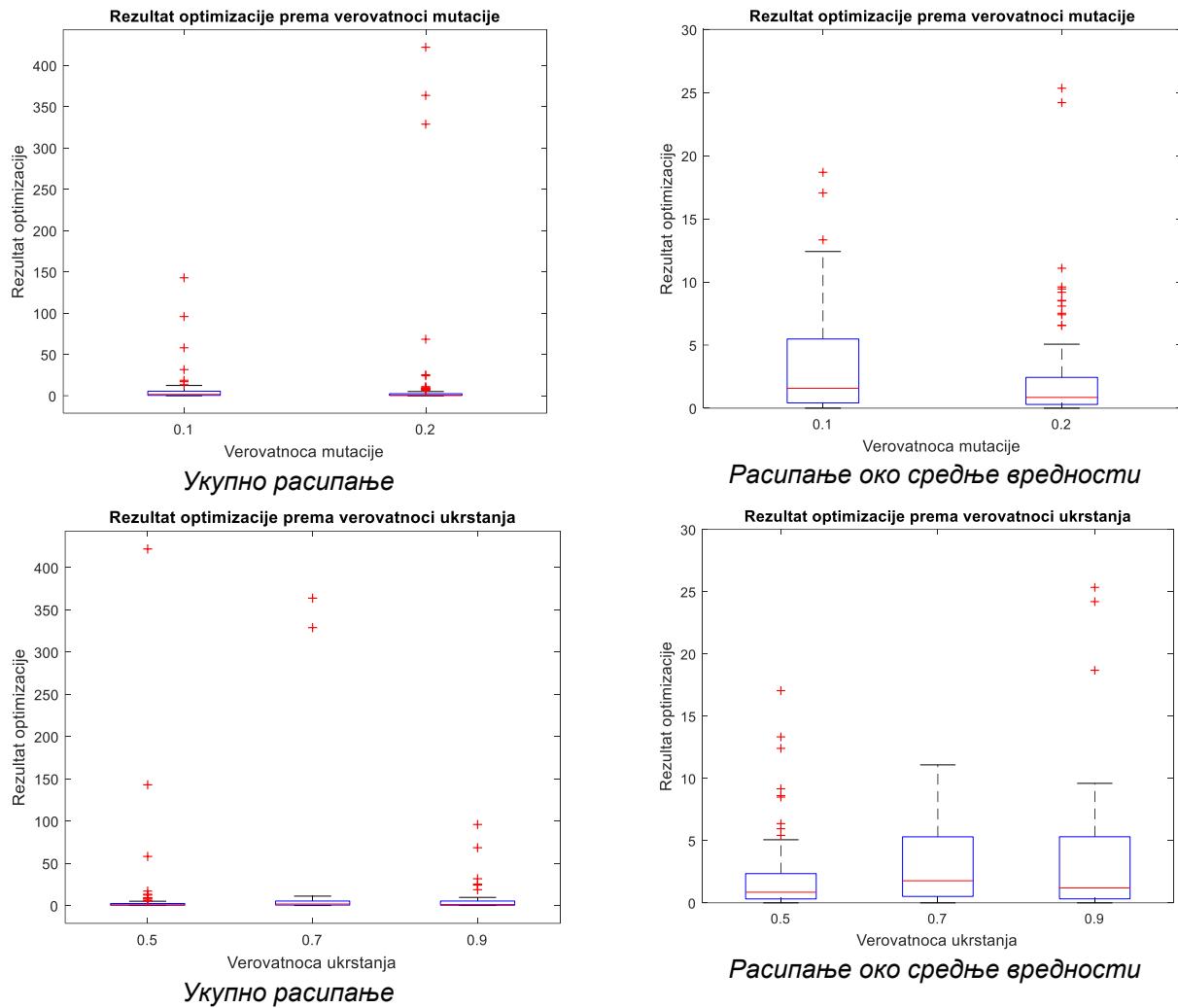
126	5	100	3	0,2	0,5	0,76	0,52	0,389376
127	10	100	1	0,1	0,9	0,44	0,2	0,317696
128	10	100	1	0,1	0,7	-1,2	1,2	10,6
129	10	100	1	0,1	0,5	-1,12	1,24	4,515136
130	10	100	1	0,2	0,9	0,08	0,12	2,136896
131	10	100	1	0,2	0,7	0,24	0,04	0,608576
132	10	100	1	0,2	0,5	-0,32	0,12	1,773376
133	10	100	2	0,1	0,9	1,64	2,68	0,418816
134	10	100	2	0,1	0,7	-1,2	1,36	5,48
135	10	100	2	0,1	0,5	-1,52	2,32	6,359616
136	10	100	2	0,2	0,9	1,24	1,52	0,088576
137	10	100	2	0,2	0,7	-0,6	0,36	2,56
138	10	100	2	0,2	0,5	0,56	0,32	0,197696
139	10	100	3	0,1	0,9	0,64	0,44	0,222016
140	10	100	3	0,1	0,7	1	1,04	0,16
141	10	100	3	0,1	0,5	0,96	0,92	0,001856
142	10	100	3	0,2	0,9	-0,6	0,36	2,56
143	10	100	3	0,2	0,7	1	0,96	0,16
144	10	100	3	0,2	0,5	1,2	1,44	0,04
145	20	100	1	0,1	0,9	1,72	2,96	0,518656
146	20	100	1	0,1	0,7	-0,32	0,12	1,773376
147	20	100	1	0,1	0,5	1,28	1,64	0,078656
148	20	100	1	0,2	0,9	2,08	4,32	1,170496
149	20	100	1	0,2	0,7	0,84	0,72	0,046336
150	20	100	1	0,2	0,5	-0,32	0,12	1,773376
151	20	100	2	0,1	0,9	0,08	0,04	0,959296
152	20	100	2	0,1	0,7	0,96	0,92	0,001856
153	20	100	2	0,1	0,5	0,96	0,92	0,001856
154	20	100	2	0,2	0,9	-0,56	0,32	2,437696
155	20	100	2	0,2	0,7	0,48	0,24	0,279616
156	20	100	2	0,2	0,5	0,08	0	0,850496
157	20	100	3	0,1	0,9	-1,2	1,36	5,48
158	20	100	3	0,1	0,7	-0,28	0,12	1,811456
159	20	100	3	0,1	0,5	1,32	1,76	0,133376
160	20	100	3	0,2	0,9	1,32	1,76	0,133376
161	20	100	3	0,2	0,7	1,4	1,96	0,16
162	20	100	3	0,2	0,5	0,08	0	0,850496
163	5	500	1	0,1	0,9	-0,28	0,12	1,811456
164	5	500	1	0,1	0,7	-0,56	0,32	2,437696
165	5	500	1	0,1	0,5	-0,64	0,44	2,782016
166	5	500	1	0,2	0,9	-0,32	0,12	1,773376
167	5	500	1	0,2	0,7	0,32	0,08	0,512576
168	5	500	1	0,2	0,5	0,96	0,92	0,001856
169	5	500	2	0,1	0,9	2,04	4,16	1,081856
170	5	500	2	0,1	0,7	-0,32	0,12	1,773376
171	5	500	2	0,1	0,5	0,08	0	0,850496
172	5	500	2	0,2	0,9	0,56	0,32	0,197696
173	5	500	2	0,2	0,7	-0,32	0,12	1,773376
174	5	500	2	0,2	0,5	0,08	0	0,850496
175	5	500	3	0,1	0,9	-0,56	0,28	2,546496
176	5	500	3	0,1	0,7	0,08	0	0,850496
177	5	500	3	0,1	0,5	-1,32	1,76	5,413376
178	5	500	3	0,2	0,9	-0,28	0,12	1,811456
179	5	500	3	0,2	0,7	1,76	3,08	0,608576
180	5	500	3	0,2	0,5	-0,28	0,12	1,811456
181	10	500	1	0,1	0,9	0,84	0,72	0,046336

182	10	500	1	0,1	0,7	-1,84	3,4	8,086336
183	10	500	1	0,1	0,5	1,76	3,08	0,608576
184	10	500	1	0,2	0,9	0,08	0	0,850496
185	10	500	1	0,2	0,7	-0,32	0,12	1,773376
186	10	500	1	0,2	0,5	1,08	1,16	0,010496
187	10	500	2	0,1	0,9	2,04	4,2	1,229056
188	10	500	2	0,1	0,7	0,08	0,04	0,959296
189	10	500	2	0,1	0,5	0,08	0	0,850496
190	10	500	2	0,2	0,9	2,04	4,2	1,229056
191	10	500	2	0,2	0,7	0,32	0,08	0,512576
192	10	500	2	0,2	0,5	0,24	0,04	0,608576
193	10	500	3	0,1	0,9	1,6	2,56	0,36
194	10	500	3	0,1	0,7	-2	4	9
195	10	500	3	0,1	0,5	0,64	0,4	0,138816
196	10	500	3	0,2	0,9	-2,08	4,36	9,599296
197	10	500	3	0,2	0,7	-1,84	3,4	8,086336
198	10	500	3	0,2	0,5	-0,32	0,12	1,773376
199	20	500	1	0,1	0,9	0,8	0,64	0,04
200	20	500	1	0,1	0,7	1,04	1,08	0,001856
201	20	500	1	0,1	0,5	0,72	0,52	0,078656
202	20	500	1	0,2	0,9	1	1	0
203	20	500	1	0,2	0,7	0,96	0,92	0,001856
204	20	500	1	0,2	0,5	0,84	0,72	0,046336
205	20	500	2	0,1	0,9	-0,28	0,12	1,811456
206	20	500	2	0,1	0,7	-0,32	0,12	1,773376
207	20	500	2	0,1	0,5	-0,32	0,12	1,773376
208	20	500	2	0,2	0,9	1,84	3,4	0,726336
209	20	500	2	0,2	0,7	1,4	1,96	0,16
210	20	500	2	0,2	0,5	-0,32	0,12	1,773376
211	20	500	3	0,1	0,9	-2	4	9
212	20	500	3	0,1	0,7	0,08	0	0,850496
213	20	500	3	0,1	0,5	0,08	0	0,850496
214	20	500	3	0,2	0,9	1,4	1,96	0,16
215	20	500	3	0,2	0,7	-0,32	0,12	1,773376
216	20	500	3	0,2	0,5	0,08	0	0,850496



Слика 5. Резултати оптимизације за различите улазне параметре





Слика 6. Резултати оптимизације за различите улазне параметре

2.3 Задатак 3

Проблем трговачког путника је могуће решити помоћу коришћења генетичких алгоритама. За дату листу градова и растојања између свака два од њих (Табела 3) потребно је пронаћи путању која укључује све градове само једном такву да је укупна дужина пређеног пута најкраћа.

Табела 3. Подаци за проблем трговачког путника

Дистанца у $\text{km} \cdot 10^{-3}$		1	2	3	4	5	6	7	8
		Варшава	Делхи	Мадрид	Москва	Шангај	Лондон	Сеул	Париз
1	Варшава	0	5,262	2,289	1,150	7,966	1,448	7,105	1,366
2	Делхи	5,262	0	7,273	4,341	4,242	6,710	4,686	6,758
3	Мадрид	2,289	7,273	0	3,440	10,254	1,262	9,996	0,719
4	Москва	1,150	4,341	3,440	0	6,815	2,500	6,606	2,759
5	Шангај	7,966	4,242	10,254	6,815	0	9,196	0,870	9,560
6	Лондон	1,448	6,710	1,262	2,500	9,196	0	8,858	0,618
7	Сеул	7,105	4,686	9,996	6,606	0,870	8,858	0	9,283
8	Париз	1,366	6,758	0,719	2,759	9,560	0,618	9,283	0

Извршено је 24 варијације алгоритма тако што су варирани следећи параметри:

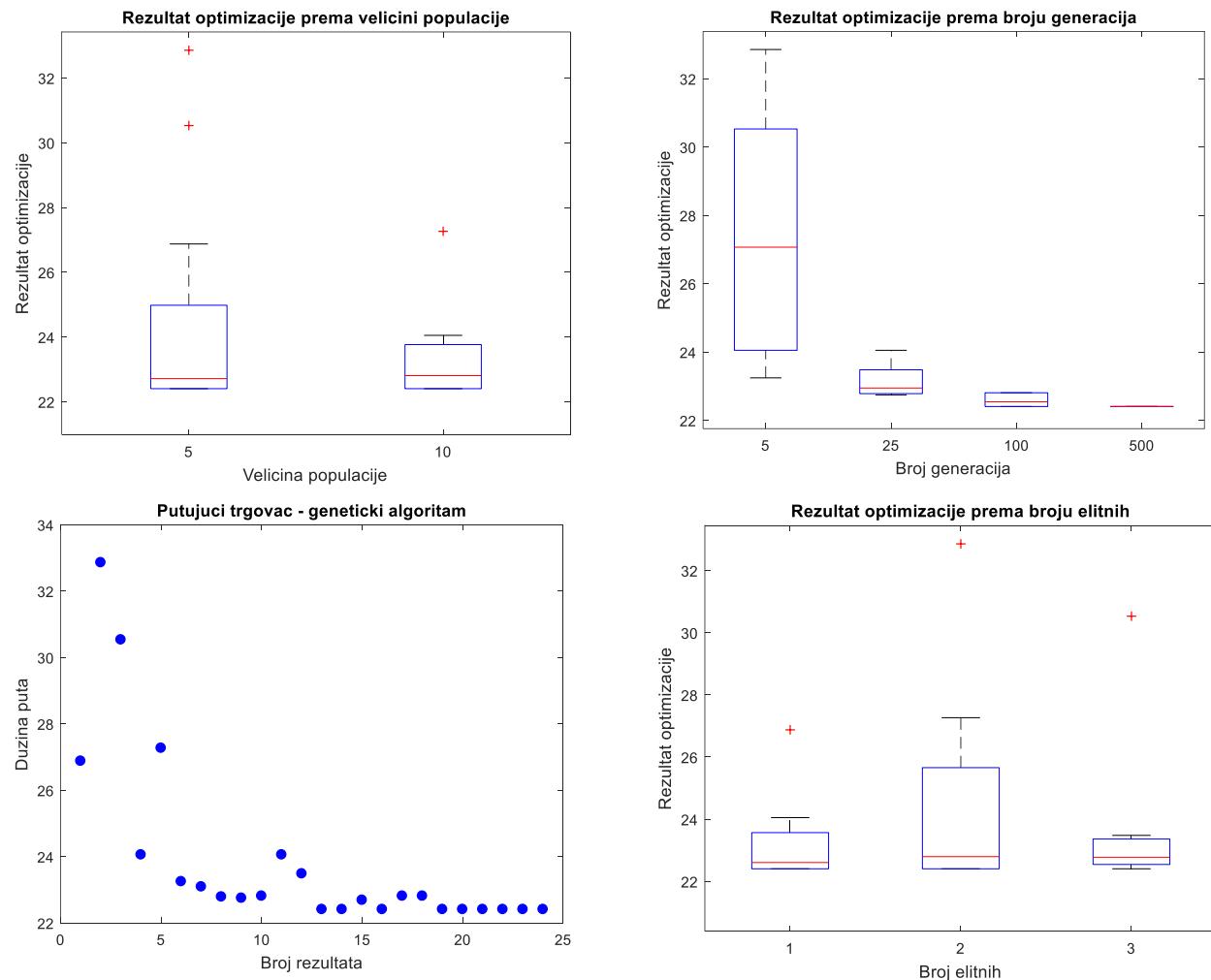
- број генерација - 10, 25, 100 или 500;
- величина популације - 5 или 10;
- број елитних - 1, 2 или 3

Након извршавања оптимизационог алгоритма резултати су приказани у Табели 4 и жутом бојом су приказана решења са најбољим резултатом, а на слици 7 су резултати оптимизације.

Табела 4. Табела са резултатима оптимизације за различите параметре - Трговачки путник

РБ	Величина популације	Број генерација	Број елитних	Дужина пута	Градови
1	5	5	1	26,874	Pariz-Sangaj-Seul-Delhi-Moskva-London-Madrid-Varsava
2	5	5	2	32,856	Moskva-Delhi-Pariz-Varsava-London-Madrid-Seul-Sangaj
3	5	5	3	30,53	Varsava-Delhi-Moskva-Sangaj-Seul-Madrid-London-Pariz
4	10	5	1	24,051	Madrid-London-Varsava-Sangaj-Seul-Delhi-Moskva-Pariz
5	10	5	2	27,265	Madrid-Sangaj-Seul-Delhi-Moskva-Pariz-London-Varsava
6	10	5	3	23,244	Sangaj-Seul-Varsava-Pariz-London-Madrid-Moskva-Delhi
7	5	25	1	23,087	Moskva-Delhi-Sangaj-Seul-London-Pariz-Madrid-Varsava
8	5	25	2	22,783	Delhi-Moskva-Madrid-Pariz-London-Varsava-Seul-Sangaj
9	5	25	3	22,746	Moskva-Pariz-Madrid-London-Varsava-Seul-Sangaj-Delhi

10	10	25	1	22,808	London-Madrid-Pariz-Varsava-Moskva-Delhi-Sangaj-Seul
11	10	25	2	24,051	Seul-Delhi-Moskva-Pariz-Madrid-London-Varsava-Sangaj
12	10	25	3	23,48	Seul-Sangaj-Moskva-London-Madrid-Pariz-Varsava-Delhi
13	5	100	1	22,405	Seul-Varsava-Pariz-Madrid-London-Moskva-Delhi-Sangaj
14	5	100	2	22,405	Delhi-Moskva-London-Madrid-Pariz-Varsava-Seul-Sangaj
15	5	100	3	22,684	Moskva-Delhi-Sangaj-Seul-Varsava-Madrid-Pariz-London
16	10	100	1	22,405	Varsava-Pariz-Madrid-London-Moskva-Delhi-Sangaj-Seul
17	10	100	2	22,808	Moskva-Varsava-Pariz-Madrid-London-Seul-Sangaj-Delhi
18	10	100	3	22,808	Pariz-Madrid-London-Seul-Sangaj-Delhi-Moskva-Varsava
19	5	500	1	22,405	Madrid-London-Moskva-Delhi-Sangaj-Seul-Varsava-Pariz
20	5	500	2	22,405	Pariz-Madrid-London-Moskva-Delhi-Sangaj-Seul-Varsava
21	5	500	3	22,405	London-Madrid-Pariz-Varsava-Seul-Sangaj-Delhi-Moskva
22	10	500	1	22,405	Varsava-Seul-Sangaj-Delhi-Moskva-London-Madrid-Pariz
23	10	500	2	22,405	Sangaj-Delhi-Moskva-London-Madrid-Pariz-Varsava-Seul
24	10	500	3	22,405	Moskva-Delhi-Sangaj-Seul-Varsava-Pariz-Madrid-London



Слика 7. Резултати оптимизације Трговачки путник

3. АЛГОРИТАМ СИМУЛИРАНОГ ЖАРЕЊА

Алгоритам симулираног жарења (енг. *Simulated annealing*, скраћено SA) је стохастички алгоритам који се примењује за проблем глобалне оптимизације, најчећше за проналазак апроксимације глобалног оптимума неке функције. Настао је 1953. године и предложила га је група аутора *Metropolis* и остали. Назив је добио на основу аналогије са процесом жарења челика који се користи у металургији. Челик се прво загрева до одређене температуре и на њој се задржи одређено време, а затим креће постепено хлађење које оплемењује челик и доводи до побољшања затезне чврстоће. У општем смислу, жарење, у науци о материјалима, представља процес који доводи до побољшања механичких, физичких или технолошких особина метала. Приликом процеса хлађења метала један од најбитнијих фактора је брзина хлађења, која се мора контролисати да не би довела до нежељених деформација у структури кристалне решете које могу довести до високе крости и лаког пуцања метала (аморфно стање). Приликом загревања атоми почињу да се крећу према вишим енергетским нивоима, а затим се приликом постепеног хлађења атоми распоређују у кристалној решетци тако да доводе до мање унутрашње енергије него пре загревања. Усклађена структура кристалних решетки које немају деформације доводи до стања минималне енергије, што значи већа стабилност и боље карактеристике [1].

На основу аналогије са горе наведеним, алгоритам симулираног жарења се може представити у два корака, тј. корак загревања и корак хлађења метала, а функција циља алгоритма се поистовећује са стањем минималне енергије.

У току израде овог алгоритма корак загревања се замењује неком иницијалном високом темературом, док се за почетно тренутно решење узима било које решење из простора могућих решења. У току извршавања алгоритма узима се насумично једно од околних решења и испитује се да ли је оно боље. Уколико је ново решење боље од тренутног онда оно постаје тренутно, али ако је пошије оно такође може постати тренутно са одговарајућом вероватноћом у зависности од тренутне температуре. Приликом извршавања итерација алгоритма температура се смањује за одговарајући температурни коефицијент, што омогућава да се за тренутно решење усвоји и неко решење које није боље од претходног све док је температура доволно висока. Ова особина представља основну разлику овог и класичног алгоритма јер боље решење не мора увек бити усвојено што омогућава наставак претраге уколико се алгоритам нађе у локалном оптимуму.

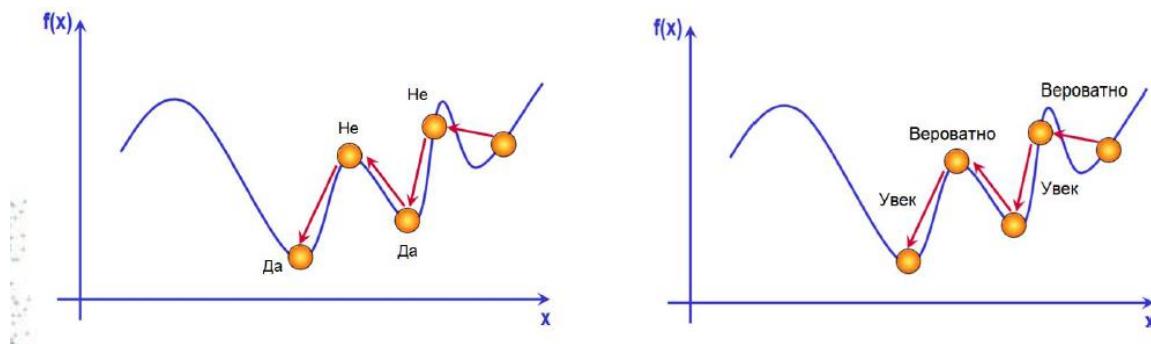


Слика 8. Алгоритам симулираног жарења [2]

Вероватноћа којом се усваја лошије решење као тренутно зависи од неколико фактора. Као што је горе и споменуто, прво се разматра на којој температури се тренутно алгоритам налази. Поред температуре разматра се и разлика између тренутног и новог решења. Математичка релација која описује утицај ових параметара на вероватноћу избора лошијег решења је дата једначином (3).

$$P(\Delta E) = e^{\left(\frac{-\Delta E}{k \cdot T}\right)} \quad (3)$$

Где је $\Delta E = \Delta E_1 - \Delta E_2 = f(\text{функција циља привременог решења}) - f(\text{функција циља тренутног решења})$, а k -Болцманова константа.



Слика 9. Поређење класичног и СА алгоритма [2]

За потребе пројекта, у Табели 5 ће бити приказан псеудокод који описује алгоритам симулираног жарења који је коришћен.

Табела 5. Псеудокод алгоритма симулираног жарења

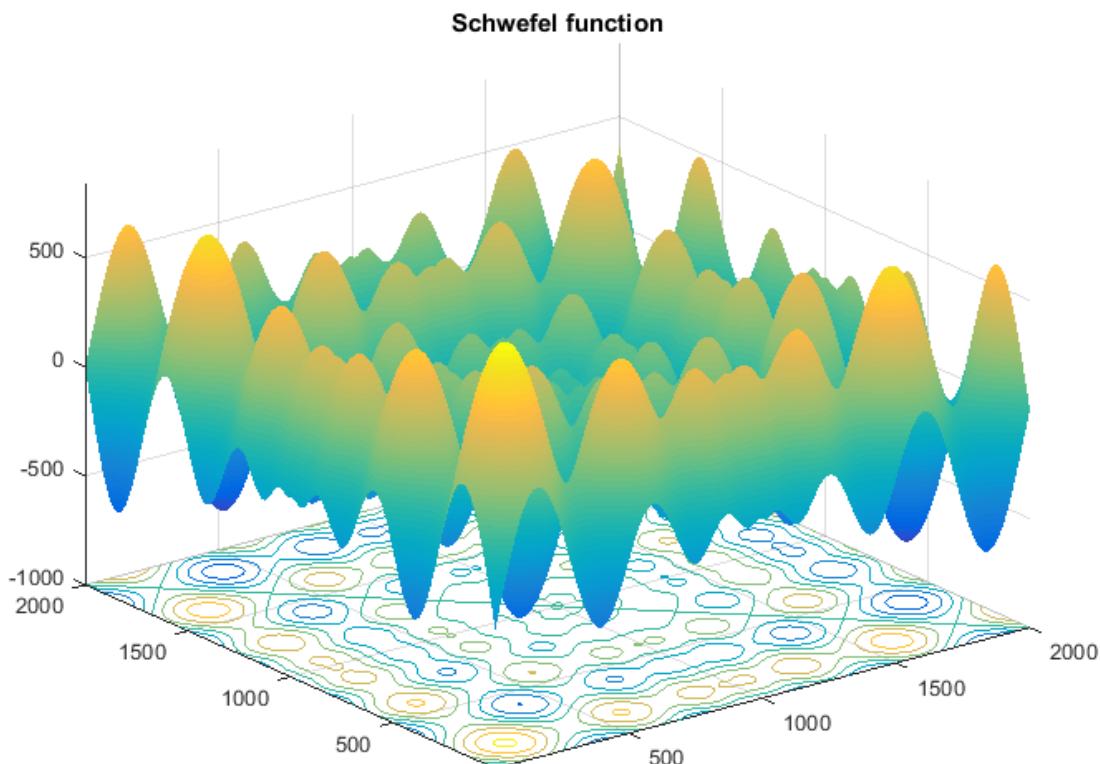
Корак	Опис псеудокода
1:	Одредити иницијалну тренутну температуру T_0 , фактор хлађења T_c и најнижу температуру T_{min} која када се усвоји представља последњу итерацију алгоритма.
2:	Случајно одабрати једно могуће решење из простора могућих решења и усвојити га као тренутно
3:	Евалуација функције циља
4:	Repeat:
5:	Променити тренутно решење и усвојити га као привремено
6:	Израчунати вредност функцију циља за тренутно и привремено решење
7:	За проблем минимизације: $\Delta E = f(\text{привремено решење}) - f(\text{тренутно решење})$ За проблем максимизације: $\Delta E = f(\text{тренутно решење}) - f(\text{привремено решење})$
8:	if $\Delta E < 0$
9:	Привремено решење поставити као тренутно
10:	Else
11:	Случајно генерисати r у интервалу $[0,1]$
12:	If $r < e^{-\Delta E/T}$
13:	Привремено решење поставити као тренутно
14:	Else
15:	Тренутно решење оставити непромењено
16:	end if
17:	end if
18:	$T = T \cdot T_c$ (Смањивање температуре за усвојени температурни коефицијент)
19:	Until $T \leq T_{min}$
20:	Output оптимално решење

3.1 Задатак 1

Schwefel функција се често користи за решавање проблема оптимизације. Функција чији се минимум тражи дата је једначином (4), а график је дат на слици 10.

$$f(\mathbf{x}) = -\sum_{i=1}^n x_i \sin(\sqrt{|x_i|}) \quad (4)$$

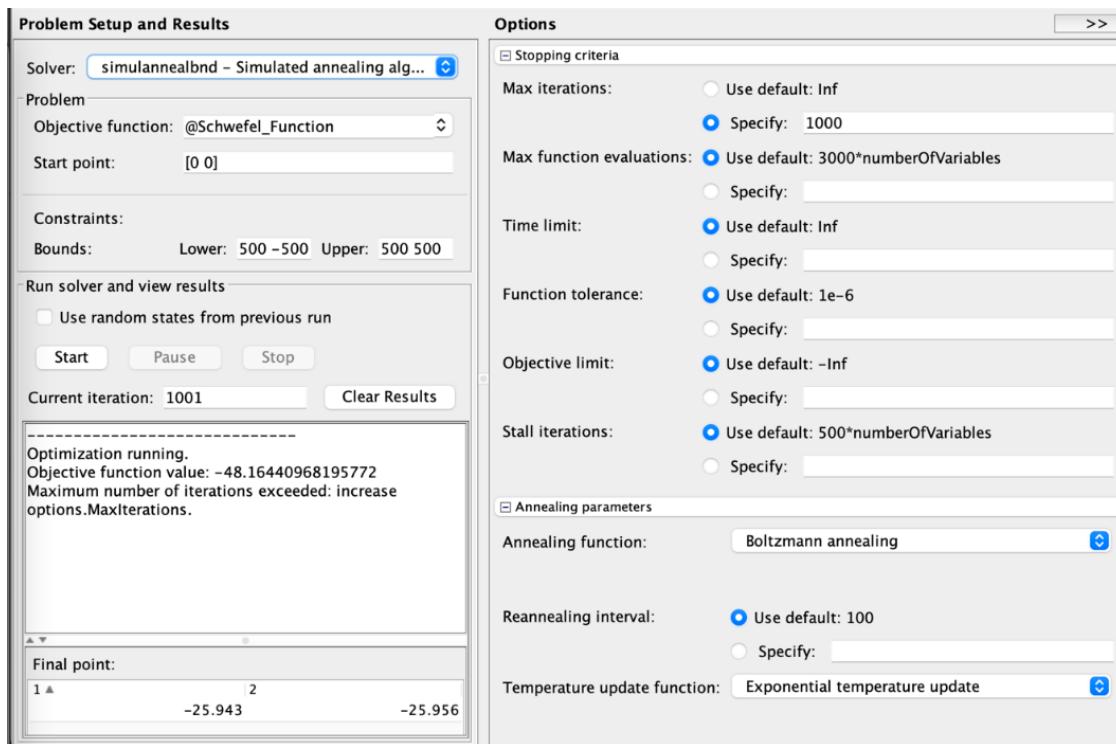
$$n=2, -500 \leq x_i \leq 500$$



Слика 10. Schwefel функција

Минимизација функције извршена је применом алгоритма симулираног жарења у оквиру Optimization Toolbox-а (слика 11). Функција циља:

```
function [y] = Schwefel_Function(x)
y = sum(-x.*sin(sqrt(abs(x))))
```



Слика 11. Schwefel функција у Optimization Toolbox-у

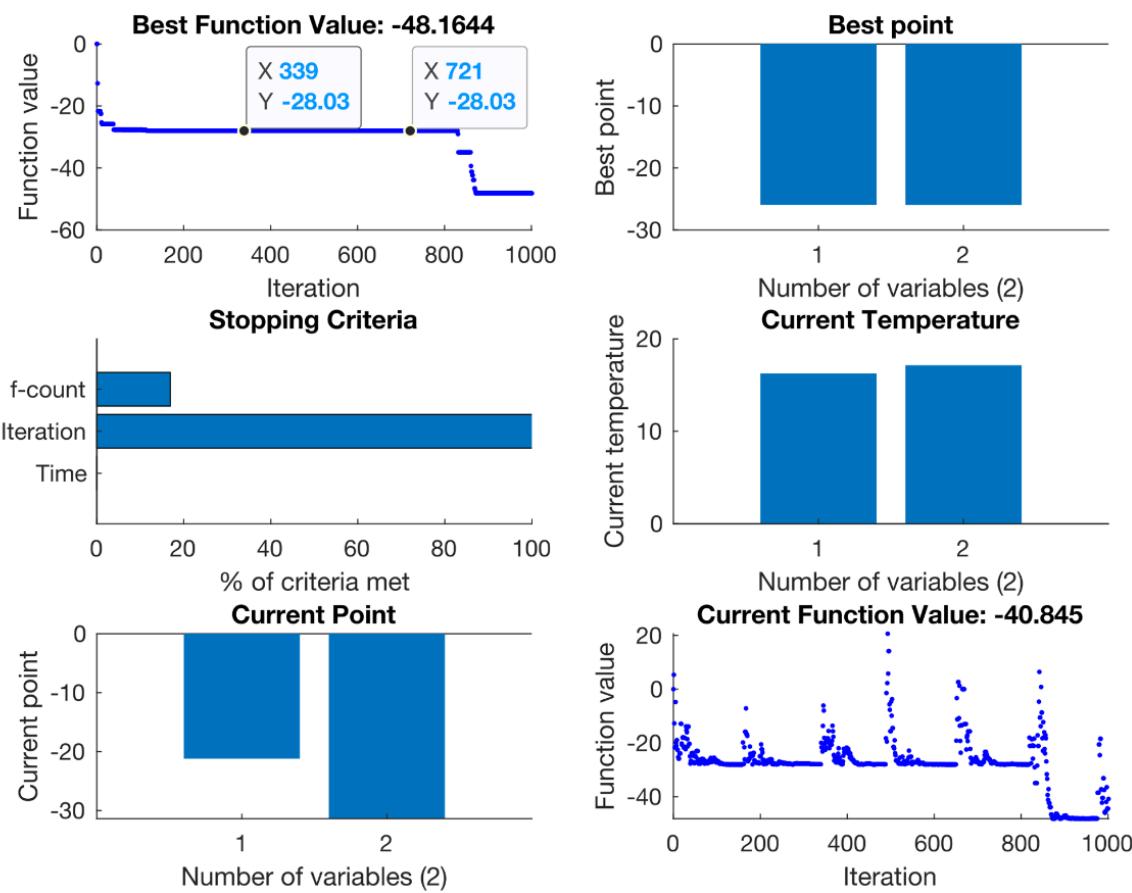
Добијени резултати приказани су Табелом 6 где је жутом бојом под редним бројем 29 приказано најбоље решење.

Табела 6. Табела са резултатима оптимизације за различите параметре Schwefel function

P. Бр	Максимални број итерација	Иницијална вредност температуре	Функција жарења	Почетна тачка	Остварен број итерација	x1	x2	f(x)
1.	1000	100	Болцманово жарење	[0 0]	1001	-25.94	-25.96	-48.16
2.	1000	1	Болцманово жарење	[0 0]	1001	5.24	5.23	7.89
3.	1000	50	Болцманово жарење	[0 0]	1001	-25.87	-25.87	-48.17
4.	1000	150	Болцманово жарење	[0 0]	1001	-25.87	65.51	-87.71
5.	1000	200	Болцманово жарење	[0 0]	1001	-25.89	-25.81	-48.16
6.	1000	1000	Болцманово жарење	[0 0]	1001	74.148	-123.9	-176.68
7.	1000	500	Болцманово жарење	[0 0]	1001	-25.65	65.68	-87.71
8.	1000	100	Болцманово жарење	[-5 1]	1001	-25.87	-25.84	-48.16
9.	1000	100	Болцманово жарење	[-50 50]	1001	-25.89	65.53	-87.72
10.	1000	100	Болцманово жарење	[-50 -100]	1001	-25.89	-124.85	-146.96
11.	1000	100	Болцманово жарење	[-100 -100]	1001	-124.82	-124.81	-124.75
12.	1000	1000	Болцманово жарење	[-100 -100]	1001	-130	-114.1	-227.89
13.	1000	100	Болцманово жарење	[-100 200]	1001	-124.83	203.84	-324.72
14.	1000	100	Болцманово жарење	[-100 100]	1001	-124.83	65.533	-186.5
15.	1000	100	Болцманово жарење	[-300 400]	1001	-302.52	420.85	-719.52
16.	1000	100	Болцманово жарење	[-500 500]	1001	-500	420.97	-599.57
17.	1000	1000	Болцманово жарење	[-500 500]	1001	-500	420.98	-599.57
18.	1000	100	Болцманово жарење	[-40 30]	1001	-25.87	65.57	-87.72
19.	1000	100	Брзо жарење	[-300 400]	1001	0.33	0.159	-719.53
20.	1000	100	Брзо жарење	[-400 300]	1001	-302.5	203.82	-502.39
21.	1000	100	Брзо жарење	[-500 500]	1001	-500	421.05	-599.57
22.	1000	100	Брзо жарење	[-5 1]	1001	-124.8	-124.81	-245.75
23.	1000	100	Брзо жарење	[-50 50]	1001	65.5	65.61	-127.26
24.	1000	100	Брзо жарење	[0 0]	1001	65.59	-124.82	-186.51
25.	1000	100	Брзо жарење	[-100 -100]	1001	-124.82	65.54	-186.51
26.	1000	100	Брзо жарење	[-100 100]	1001	65.55	203.81	-265.48
27.	1000	100	Брзо жарење	[-100 200]	1001	-124.69	203.82	-324.72
28.	1000	1000	Брзо жарење	[-300 400]	1001	-302.52	420.9	-719.53
29.	Inf	1000	Брзо жарење	[-300 400]	1505	421.08	421.03	-837.96
30.	Inf	100	Болцманово жарење	[-300 400]	1957	-302.54	420.96	-719.53
31.	Inf	100	Брзо жарење	[-5 1]	1364	-124.8	203.8	-324.71
32.	Inf	100	Болцманово жарење	[-5 1]	2270	-25.89	65.5	-87.72
33.	Inf	100	Болцманово жарење	[0 0]	2706	65.5	-25.86	-87.72
34.	Inf	100	Брзо жарење	[0 0]	1706	203.82	203.82	-403.68
35.	Inf	100	Болцманово жарење	[-100 100]	3562	-124.83	65.55	-186.51
36.	Inf	100	Брзо жарење	[-100 100]	1886	-124.81	203.809	-324.72
37.	Inf	100	Болцманово жарење	[-100 -100]	1153	-124.85	-124.86	-245.75
38.	Inf	100	Брзо жарење	[-100 -100]	2244	-124.83	-124.83	-245.75

Након спроведеног експеримента може се закључити:

- Повећање броја итерације доводи до мањих вредности функције циља;
- Брзо жарење се у већини случајева показало боље од Болцмановог, алгоритам који је користио брзо жарење је добијао мање вредности функције циља кроз мање итерације;
- Повећање иницијалне температуре доводи до мањих вредности функције циља (слика 12).



Слика 12. Графици функције Schwefel

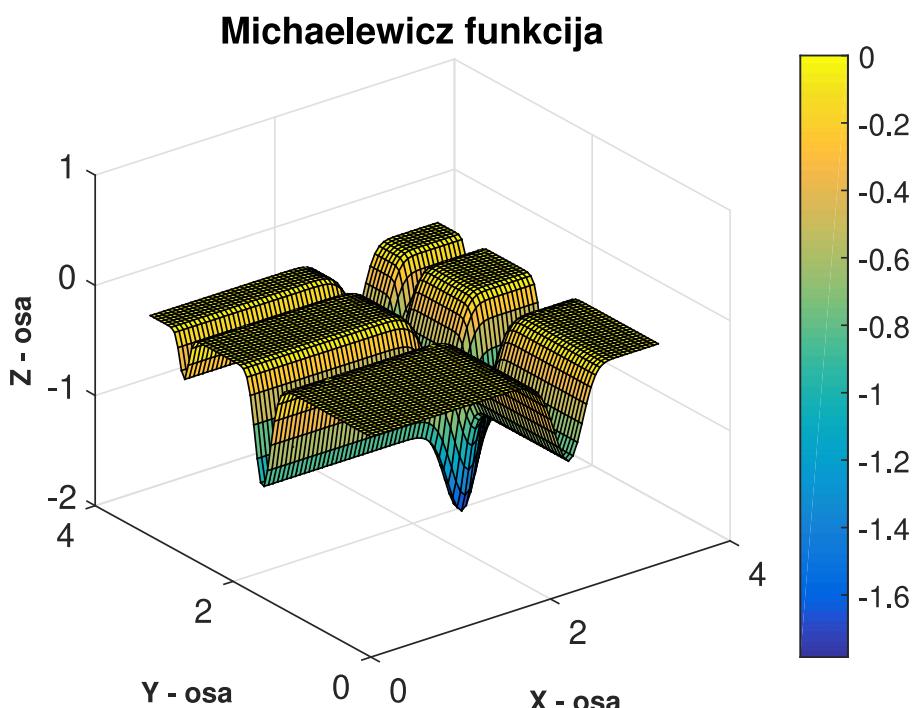
3.2 Задатак 2

Функција „Michaelwicz“ је мултимодална тест функција, корисна за процену карактеристика оптимизационог алгоритма, као што су: брзина конвергенције, тачност и генералне перформансе. Функција чији се минимум тражи дата је једначином (5), а график је дат на слици 13..

$$f(\mathbf{x}) = -\sum_{i=1}^n \sin(x_i) \cdot \left[\sin\left(\frac{ix_i^2}{\pi}\right) \right]^{20} \quad (5)$$

$$n=2, 0 \leq x_i \leq 5$$

Глобални минимум функције је $f(\mathbf{x}) = -1.8013$ за $\mathbf{x} = (2.20, 1.57)$.



Слика 13. Michalewicz функција

Имплементација „Michaelwicz“ функције у MATLAB-у:

```
function [y] = michal(x)

d = 2;
sum = 0;
for i = 1:d
    new = sin(x(i)) * (sin((i)*x(i)^2/pi))^(20);
    sum = sum + new;
end
y = -sum;
end
```

Коришћен код алгоритма симулираног жарења за минимизацију функције, развијен у MATLAB-у. Функција napravi_privremeno_resenje прави привремена решења користећи нормалну расподелу са очекивањем вредности тренутног решења и константном стандардном девијацијом.

```
function [y] = napravi_privremeno_resenje(X,T0,T_tr)

std_dev=0.5;
i = round(rand())+1;

y = X;
y(i) = y(i) + randn(1,1)*std_dev;
%y(i) = y(i) + randn(1,1)*(0.4+1*(T_tr/T0));

if(y(i)<0)
    y(i)=0;
elseif(y(i)>5)
    y(i)=5;
end

end
```

Коришћена је и друга варијанта ове функције када стандардна девијација зависи од тренутне температуре.

```
function [y] = napravi_privremeno_resenje(X,T0,T_tr)

%std_dev=0.5;
koef_std=1.75;
min_std=0.25;
i = round(rand())+1;

y = X;
%y(i) = y(i) + randn(1,1)*std_dev;
y(i) = y(i) + randn(1,1)*(min_std+koef_std*(T_tr/T0));

if(y(i)<0)
    y(i)=0;
elseif(y(i)>5)
    y(i)=5;
end

end
```

Главни код који извршава алгоритам симулираног жарења приказан је у наставку.

```
T_tr = T0;
I_tr = Ir;
Y_tr = Yr;
Y_best = Yr;
I_best = I_tr;
iteracija = 0;
iter_best = 0;
figure(1);
subplot(2,1,1);
title('Menjanje Najbolje funkcije cilja kroz iteracije');
xlabel('Broj iteracija');
ylabel('Najbolja funkcija cilja')
%hold on;
```

```

%plot(iteracija, Y_best,:b+);

subplot(2,1,2);
title('Menjanje trenutne funkcije cilja kroz iteracije');
xlabel('Broj iteracija');
ylabel('Trenutna funkcija cilja')
%hold on;
%plot(iteracija, Y_tr,:r*);

Y_prev = Y_tr;
Y_best_prev = Y_best;

while (T_tr>T_min)
    %I_pr = napravi_privremeno_resenje(I_tr);
    I_pr = napravi_privremeno_resenje(I_tr,T0,T_tr);
    Y_pr = funkcija_cilja(I_pr);
    delta_e = Y_pr - Y_tr;
    iteracija = iteracija + 1;

    if (delta_e < 0)
        Y_tr = Y_pr;
        I_tr = I_pr;
        if (Y_tr < Y_best)
            Y_best_prev = Y_best;
            Y_best = Y_tr;
            I_best = I_tr;
            iter_best = iteracija;
        end
    else
        verovatnoca_promene = exp(-delta_e/T_tr);
        r = rand();
        if(r < verovatnoca_promene)
            Y_tr = Y_pr;
            I_tr = I_pr;
        end
    end
end

T_tr = T_tr * Tc;
subplot(2,1,1);
hold on;
plot([iteracija, iteracija+1], [Y_best_prev, Y_best],'b');
subplot(2,1,2);
hold on;
plot([iteracija, iteracija+1], [Y_prev, Y_tr],'r');
Y_best_prev = Y_best;
Y_prev = Y_tr;
end

disp('Najbolji broj je');
disp(I_best);
disp('Njegova funkcija cilja je');
disp(Y_best);

%resenje1 = [T0, T_min, Tc];
resenje1 = [round(T0, 3, 'significant'), round(T_min, 3, 'significant'), round(Tc, 3, 'significant')];

I_r_tabela = round(Ir, 3, 'significant');
Y_r_tabela = round(Yr, 3, 'significant');
I_best_tabela = round(I_best, 3, 'significant');
Y_best_tabela = round(Y_best, 3, 'significant');

%resenje2 = [round(Ir, 3, 'significant'), round(Yr, 3, 'significant'); ...
%    round(I_best, 3, 'significant'), round(Y_best, 3, 'significant')];

```

```

resenje3 = [iter_best, iteracija];
savefig('rbroj14.fig');

```

Приликом минимизације функције испитиван је утицај промене следећих параметара алгоритма симулираног жарења:

- иницијална вредност температуре,
- вредност минималне температуре,
- коефицијент хлађења

и резултати су приказани у Табелама 7 и 8. Такође, у обзир је узет и начин прављења привремених решења.

У првом случају, привремена решења се праве на исти начин током целог процеса извршавања алгоритма (Табела 7). У другом случају, стандардна девијација приликом прављења привременог решења се смањује пропорционално са опадањем температуре (Табела 8). Најбоље добијено решење приказано је на слици 14.

Табела 7. Симулирано жарење привремена решења се праве на исти начин током целог процеса извршавања алгоритма

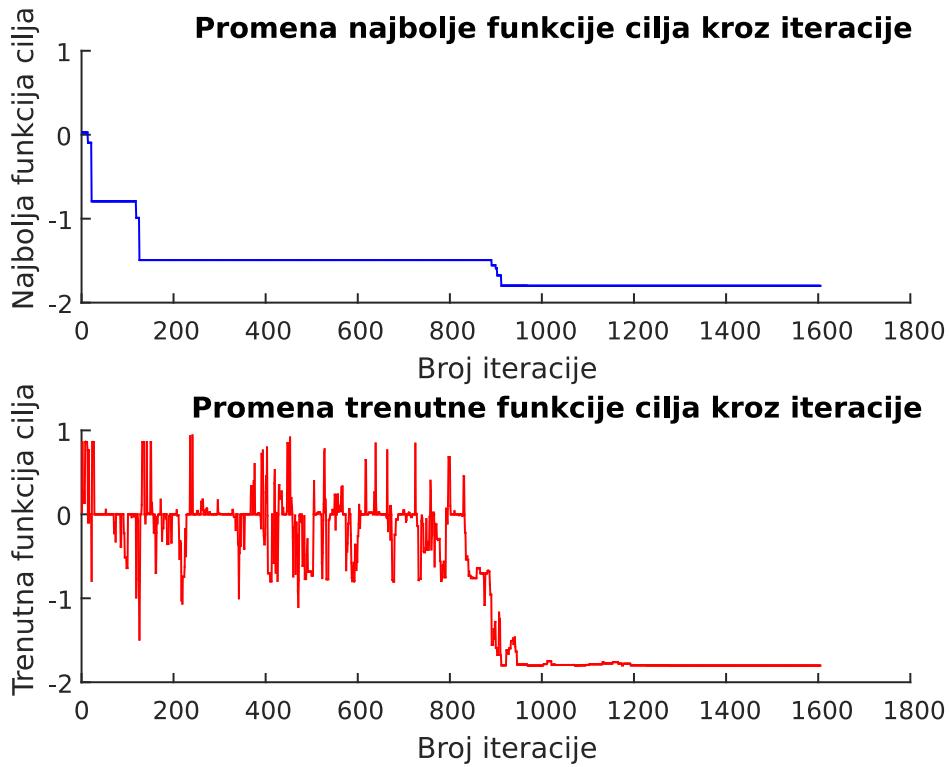
P. бр.	Иниц. темп.	Мин. темп.	Коef. хлађ.	Ir	Yr	I_best	Y_best	Iter_best	Бр. итер.	std_dev
1	500	0.0001	0.95	[0.354,4.414]	1.529e-16	[2.255,1.567]	-1.7560	209	301	1.5
2	1000	0.0001	0.95	[4.073,4.528]	0.025	[2.248,1.581]	-1.7635	242	315	1.5
3	1000	0.001	0.95	[1.708,4.664]	0.424	[2.196,1.566]	-1.7993	231	270	1.5
4	1000	0.01	0.95	[4.589,0.686]	1.692e-08	[2.216,1.536]	-1.7531	187	225	1.5
5	1000	0.0001	0.95	[2.586,4.952]	-0.019	[2.187,1.536]	-1.7519	311	315	2
6	1000	0.0001	0.95	[1.692,2.903]	-0.011	[2.210,1.581]	-1.7957	252	315	1
7	1000	0.0001	0.95	[4.432,2.575]	-0.043	[2.197,1.593]	-1.7784	246	315	0.5

Табела 8. Симулирано жарење - стандардна девијација приликом прављења привременог решења се смањује пропорционално са опадањем температуре

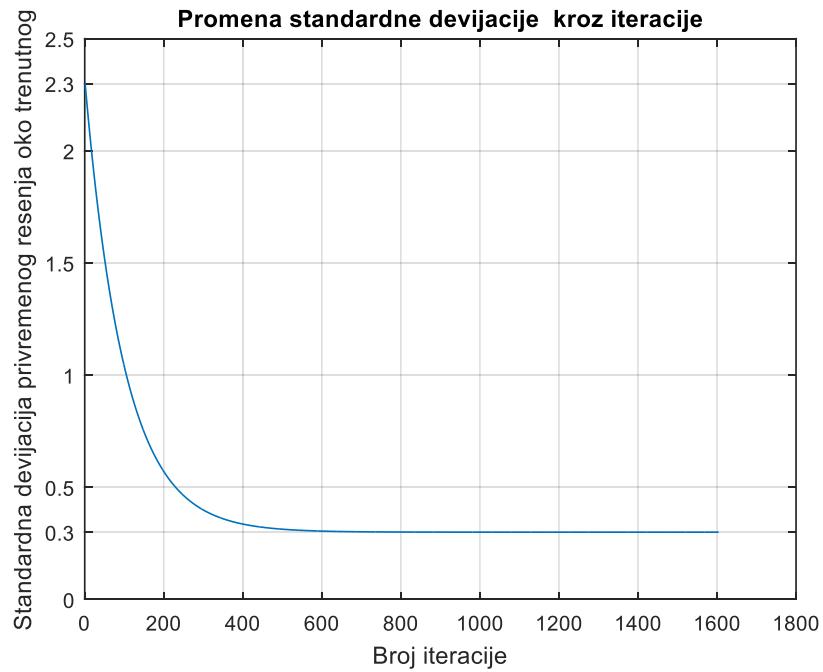
P. бр.	Ини. темп.	Мин. темп.	Коef. хлађ.	Ir	Yr	I_best	Y_best	Iter_b est	Бр. итер.	min_std	koef_s td
8	1000	0.0001	0.95	[0.306,3.305]	1.252e-05	[2.195,1.587]	-1.7886	276	315	0.4	1.5
9	1000	0.0001	0.99	[2.687,4.087]	0.216	[2.200,1.570]	-1.8012	1033	1604	0.4	1.5
10	1000	0.0001	0.99	[2.644,3.344]	-0.004	[2.211,1.571]	-1.8001	1575	1604	0.4	2
11	1000	0.0001	0.99	[2.275,4.896]	-0.004	[2.206,1.570]	-1.8011	1412	1604	0.3	2
12	1000	0.0001	0.99	[2.333,4.629]	-0.474	[2.048,1.633]	-1.3575	270	1604	0.2	2
13	1000	0.0001	0.99	[2.057,4.267]	-0.474	[2.206,1.572]	-1.8010	1581	1604	0.3	1.5
14	1000	0.0001	0.99	[2.504,1.723]	-0.437	[2.201,1.570]	-1.8012	1308	1604	0.25	1.75

Анализом добијених резултата може се закључити:

- повећање иницијалне вредности температуре и коефицијента хлађења даје боље резултате;
- смањивањем стандардне девијације приликом прављења привременог решења постиже се прецизнија конвергенција (слика 15);
- повећањем минималне температуре смањује се укупан број итерација што негативно утиче на проналажење оптималног решења.



Слика 14. Оптимално решење Michaelwicz функције - SA



Слика 15. Промена стандардне девијације

4. АЛГОРИТАМ ИНСПИРИСАН ИНТЕЛИГЕНЦИЈОМ ЈАТА КИТОВА (WOA)

Алгоритам инспирисан интелигенцијом јата китова спада у групу метахеуристичких алгоритама инспирисаних интелигенцијом роја. Алгоритам је оригинално предложен у раду професора Мирџалилија и Луиса са универзитета Торенс у Аустралији [3].

Аутори су инспирацију пронашли у јединственом начину лова грбавих китова који се назива метод лова међурићима. Ова врста кита се обично храни ситним рибама или шкампима при самој површини воде. Процес лова се може поделити на 3 фазе.

Прва фаза – Тражење плена

У првој фази лова китови траже свој плен. Са аспекта оптимизација овај део прве фазе представља процес експлорације простора могућих решења и проналажење положаја који су близу глобалног оптимума, док се притом избегавају локални оптимуми. Грбави китови претражују своју околину случајно, на основу позиције другог кита.

У овој фази користи се коефицијент A са случајним вредностима већим од 1 или мањим од -1 како би се агент за претрагу удаљио далеко од референтног кита. У оквиру експлорације ажурира се позиција агента према случајно изабраном агенту. Овај механизам и услов да је $|A| > 1$ су карактеристични за експлорацију и дозвољавају WOA алгоритму да изврши глобалну претрагу.

$$\vec{D} = \left| \vec{C} \cdot \vec{X}_{rand} - \vec{X} \right| \quad (6)$$

$$\vec{X}_{(t+1)} = \vec{X}_{rand} - \vec{A} \cdot \vec{D} \quad (7)$$

Где су:

\vec{A} - коефицијент који се рачуна као $\vec{A} = 2\vec{a}\vec{r} - \vec{a}$ где се a линеарно смањује са 2 на 0 током итерација, а r - случајни вектор у $[0, 1]$,

\vec{X} - тренутни агент,

t - тренутна итерација,

\cdot - множење сваког елемента вектора,

\vec{X}_{rand} - вектор случајног положаја (случајни кит) изабран из тренутне популације,

\vec{C} - коефицијент који се рачуна као $\vec{C} = 2\vec{r}$.

По завршетку експлорације, започиње процес експлоатације. Процес експлоатације је инспирисан тренутком када китови нађу свој плен и крену у фазу лова. Процес лова се у природи одвија на следећи начин: када уоче плен, китови прво зарањају и приближавају се мети, потом формирају спирале око плена, полако израњају и испуштају међуриће. Улога међура је да дезоријентише плен како би им се китови приближили.

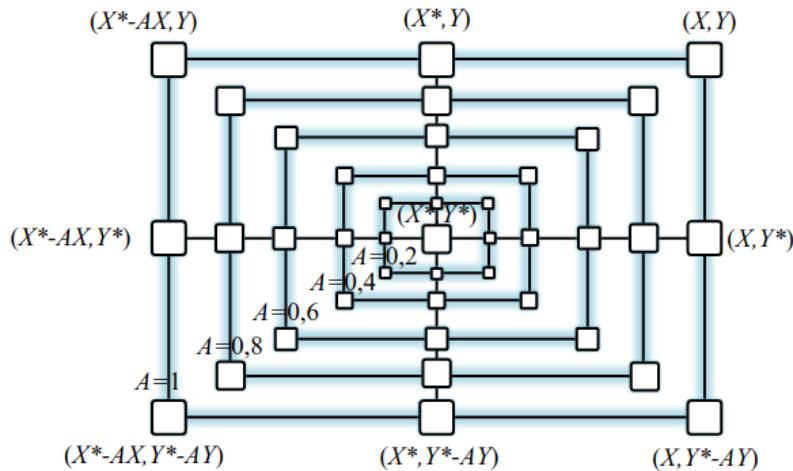
Приближавање и кружење око плена се у експлоатацији моделира кроз две фазе:

- приближавање плену,
- кружење око плена.

Друга фаза – Приближавање плену

Друга фаза WOA алгоритма се односи на приближавање плену. С обзиром да се тражи тачка у простору која има оптималну вредност фуџкије циља и која није позната, претпоставља се да се она налази близу тренутно најбољег агента и претрага се усмерава ка њему.

Процес приближавања се изводи по теменима квадрата око жељеног решења у n -димензионом простору (слика 16).



Слика 16. Приближавање најбољем агенту [2]

Математичко моделирање фазе приближавања плену може се извршити применом једначина (8) и (9):

$$\vec{D} = \left| \overrightarrow{\mathbf{C}} \cdot \overrightarrow{X}^*(t) - \overrightarrow{X}(t) \right| \quad (8)$$

$$\overrightarrow{X}(t+1) = \overrightarrow{X}^*(t) - \vec{A} \vec{D} \quad (9)$$

Где су X и X^* тренутни и најбољи агент, респективно.

X^* се мења током итерација. Коефицијенти A и C се рачунају за сваку итерацију на следећи начин:

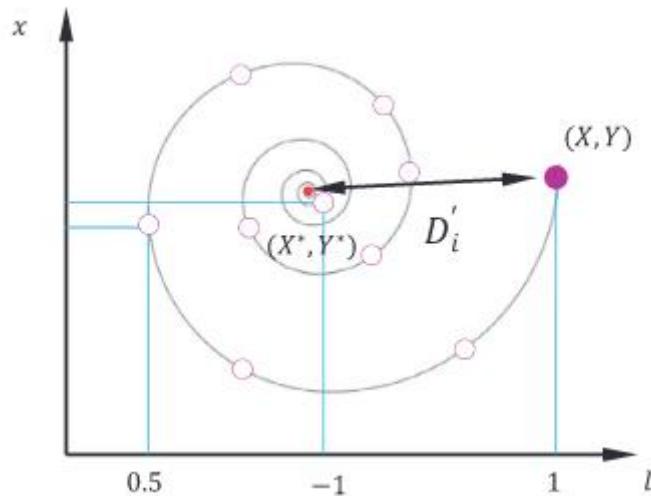
$$A = 2a \cdot r - a \quad (10)$$

$$C = 2 \cdot r \quad (11)$$

Где a линеарно опада од 2 до 0 од прве до последње итерације, а r је униформно изабран случајан број на интервалу $[0, 1]$.

Трећа фаза – Спирално приближавање плену

Трећа фаза WOA алгоритма је слична другој фази. И у овој фази се агенти приближавају најбољем агенту, с тим што се крећу по трајекторији спиралног облика (слика 17).



Слика 17. Спирално приближавање плену [2]

Математичко моделирање фазе спиралног приближавања плену може се извршити применом једначине (12):

$$\vec{X}_{(t+1)} = \vec{D}' \cdot e^{bl} \cdot \cos(2\pi l) + \vec{X}_{(t)}^* \quad (12)$$

где је $\vec{D}' = \left| \vec{X}_{(t)}^* - \vec{X}_{(t)} \right|$, b је константа која дефинише облик спирале и l је унiformно изабран случајан број из интервала $[-2, 1]$.

5. АЛГОРИТАМ ИНСПИРИСАН ИНТЕЛИГЕНЦИЈОМ РОЈА ЧЕСТИЦА (PSO)

Алгоритам оптимизације ројем честица (енгл. *Particle swarm optimization (PSO)*) је метахеуристички алгоритам заснован на интелигентном колективном понашању роја, подобан за решавање сложених математичких проблема који се јављају у инжењерским задацима.

Деведесетих година двадесетог века спроведено је неколико студија која се односе на социјално понашање група животиња. Ове студије су показале да су неке животиње које припадају одређеним групама, попут роја инсеката, способне да деле информације унутар својих група и на тај начин стекну велику предност опстанка. Мотивисани овим истраживањима, Кенеди и Еберхарт први пут су предложили овај алгоритам 1995. године [5].

Основна варијанта по којој PSO алгоритам функционише састоји се од популације јединки (енгл. *particle*) које се крећу кроз простор могућих решења. Кретање јединки вођено је њиховом сопственом најбољом позицијом у простору претраге, као и јединке са најбољом позицијом у целом роју. Сходно томе, откривањем нових побољшаних позиција усмерава се кретање роја.

Свака јединка је окарактерисана својим *положајем* и *брзином*. Положај јединке представља потенцијално решење проблема, док се брзина јединке динамички подешава током извршавања алгоритма, прилагођавајући на тај начин кретање јединке сходно свом сопственом искуству, што представља *когнитивни утицај*, као и информацијама прикупљени у интеракцији са осталим јединкама, *социјални утицај*.

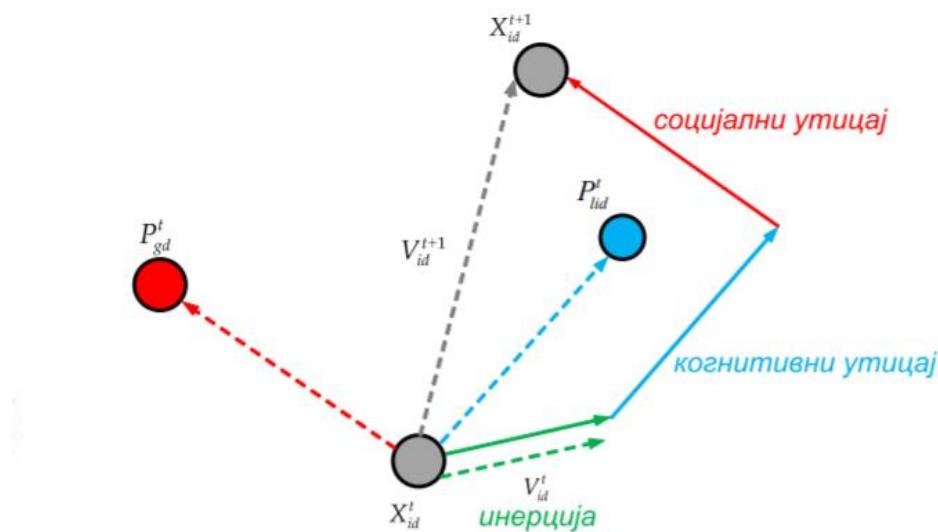
Наредна брзина i -те јединке дата је једначином (13). Она садржи три компоненте: *компонента инерције*, *когнитивна компонента* и *социјална компонента*.

$$\vec{V}_{id}^{t+1} = \overline{W} \cdot \vec{V}_{id}^t + C_1 \cdot rand() \cdot (\vec{P}_{lid}^t - \vec{X}_{id}^t) + C_2 \cdot Rand() \cdot (\vec{P}_{gd}^t - \vec{X}_{id}^t) \quad (13)$$

Компонента $\overline{W} \cdot V_{id}^t$ представља инерцију претходне ка следећој брзини, где је \overline{W} параметар инерције. Когнитивна компонента, $C_1 \cdot rand \cdot (\vec{P}_{lid}^t - \vec{X}_{id}^t)$, представља самоспознају сваке појединачне јединке. Коефицијент C_1 , који множи разлику локално најбољег и тренутног положаја, је позитивна константа убрзања и представља утицај когнитивне способности конкретне јединке на њено будуће понашање. Трећа компонента, $C_2 \cdot Rand \cdot (\vec{P}_{gd}^t - \vec{X}_{id}^t)$, је социјална компонента помоћу које све јединке у роју деле информацију о најбоље постигнутој позицији, где је коефицијент C_2 позитивна константа убрзања и представља утицај целог роја на понашање посматране јединке.

Наредни положај се сада рачуна као збир положаја у претходној итерацији и наредне брзине (једначина 14)

$$\vec{X}_{id}^{t+1} = \vec{X}_{id}^t + \vec{V}_{id}^{t+1} \quad (14)$$



Слика 18. Компоненте вектора брзине [2]

Иако не постоји универзални алгоритам оптимизације, који би давао најбоље резултате при решавању сваког проблема оптимизације, важно је истаћи неке предности које коришћење алгоритма PSO има у поређењу са осталим оптимизационим алгоритмима, које се огледају пре свега у малом броју параметара које треба подесити.

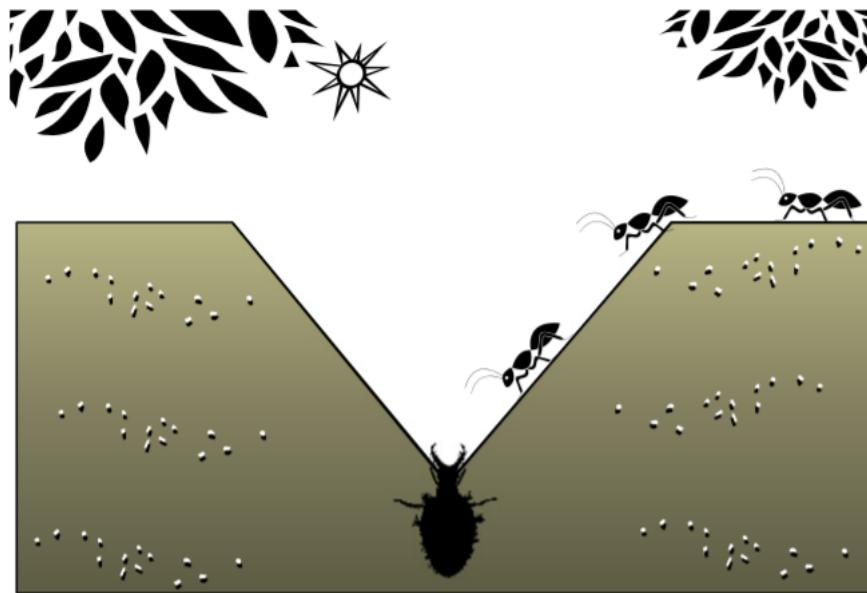
За потребе пројекта, у Табели 9 је приказан псеудокод који описује алгоритам инспирисан ројем честица који је коришћен.

Табела 9. Псеудокод алгоритма инспирисаног ројем честица

Псевдокод алгоритма PSO	
Корак 1	Иницијализација параметара алгоритма
Корак 2	Иницијализовати рој јединки са случајним брзинама и случајним положајима
Корак 3	Оредити функцију циља за сваку јединку
Корак 4	Иницијализовати глобално најбољи положај свих јединки (gbest)
Корак 5	Иницијализовати локално најбољи положај свих јединки (pbest)
Корак 6	Понављање итеративног поступка до максималног броја итерација
Корак 7	Генерисати нови рој израчунавањем брзине и положаја јединки
Корак 8	Оредити функцију циља за сваку јединку
Корак 9	Оредити gbest и pbest
Корак 10	Ажурирати gbest и pbest
Корак 11	Увећати број итерације за 1
Корак 12	Све док се не достigne максималан број итерација
Корак 13	Резултати оптимизације

6. АЛГОРИТАМ ИНСПИРИСАН ИНТЕЛИГЕНЦИЈОМ МРАВОЛОВАЦА (ALO)

Оптимизациони алгоритам ALO (енгл. *Ant Lion Optimization* – ALO) је још један од биолошки инспирисаних алгоритама оптимизације. Биолошка основа је заснована на врсти инсеката грабљиваца из породице *Myrmeleontidae* који се називају мраволовци (енгл. *Ant Lion*). Мраволовци у току свог биолошког развоја формирају две фазе: ларве и одрасле јединке. Нама од значаја је фаза ларве у којој мраволовци лове свој плен (мраве). Њихов механизам лова је заснован на постављању замке у облику конуса. Мраволовац копа конусну јamu у песку крећући се кружном путањом и избацујући песак. Након што направи замку, мраволовац се сакрије испод ње, на дну конуса, и чека да мрав (или неки други инсект) због свог хаотичног случајног кретања упадне у замку. Тада мраволовац хвата плен и избацује песак ка рубу јаме како би плен заробио у песку. Облик замке коју постављају мраволовци се може видети на слици 19 [3].



Слика 19. Конусна замка коју мраволовац образује у песку [3]

Лов мраволовца се може дефинисати у пет главних корака:

1. насумично кретање мрава,
2. изградња замке,
3. хватање мрава у замку,
4. хватање плена (затрпавање песком) и
5. поновна изградња замке.

ALO алгоритам оптимизације опонаша механизам лова мраволовца у природи и њихову интеракцију са мравима. Мрави се крећу простором могућих решења на основу свог случајног кретања док траже храну која се може моделирати на следећи начин:

$$X(t) = [0, \text{cumsum}(2r(t_1)-1), \text{cumsum}(2r(t_2)-1), \dots, \text{cumsum}(2r(t_n)-1)] \quad (15)$$

Где је:

- *cumsum* - кулмулативна сума,
- *n* - максималан број итерација,

- t - корак случајног лутања,
 - $r(t)$ - стохастичка функција дефинисана на основу једначине (16).
- $$r(t) = \begin{cases} 1 & \text{if } rand > 0,5 \\ 0 & \text{if } rand \leq 0,5 \end{cases} \quad (16)$$
- $rand$ представља случајано генерисан број из интервала $[0,1]$ на основу унiformне расподеле.

Позиције мраволовца и мрава могу се представити у облику матрица (једначина 17 и 18) за чување сваке позиције мрава и мраволовца у d димензија за n мрава или мраволовца.

$$M_{mrv} = \begin{bmatrix} A_{1,1} & A_{1,2} & \cdots & A_{1,d} \\ A_{2,1} & A_{2,2} & \cdots & A_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n,1} & A_{n,2} & \cdots & A_{n,d} \end{bmatrix} \quad (17)$$

$$M_{mrvolovac} = \begin{bmatrix} AL_{1,1} & AL_{1,2} & \cdots & AL_{1,d} \\ AL_{2,1} & AL_{2,2} & \cdots & AL_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ AL_{n,1} & AL_{n,2} & \cdots & AL_{n,d} \end{bmatrix} \quad (18)$$

Да би се могао испитати квалитет тренутне позиције мрава, за оптимизациони проблем, потребно је дефинисати функцију циља за сваког мрава која се такође може представити у облику матрице једначина (19) за n мрава у d димензија.

$$M_{FCmrv} = \begin{bmatrix} f([A_{1,1}, A_{1,2}, \dots, A_{1,d}]) \\ f([A_{2,1}, A_{2,2}, \dots, A_{2,d}]) \\ \vdots \\ f([A_{n,1}, A_{n,2}, \dots, A_{n,d}]) \end{bmatrix} \quad (19)$$

Аналогно мраву, испитује се квалитет функције циља тренутне позиције за n мраволовца у d димензија која се чува у матрици, једначина (20).

$$M_{FCmrvolovca} = \begin{bmatrix} f([AL_{1,1}, AL_{1,2}, \dots, AL_{1,d}]) \\ f([AL_{2,1}, AL_{2,2}, \dots, AL_{2,d}]) \\ \vdots \\ f([AL_{n,1}, AL_{n,2}, \dots, AL_{n,d}]) \end{bmatrix} \quad (20)$$

Да би оптимизација била успешна потребно је применити неке услове који се могу дефинисати у следећих пар тачака:

- Мрави се крећу по простору могућих решења на основу случајног лутања (енгл. *randomwalk*) у свим димензијама;
- Мраволовци са бОљом функцијом циља имају већу вероватноћу да ухвате мрава;
- Сваки мрав може бити ухваћен од стране мраволовца у свакој итерацији;

- Распон случајног хода се смањује повећањем броја итерација;
- Ако функција циља мрава постане боља него од мраволовца, онда се сматра да је мрав ухваћен у клопку;
- Мраволовац прави нову замку на позицији са бољом функцијом циља.

С обзиром да се мрав креће случајно и да је простор могућих решења ограничен, постоји могућност да мрав изађе из простора могућих решења. Овај проблем се решава ограничавањем случајног кретања мрава на основу нормирања по једначини (21).

$$X_i^t = \frac{(X_i^t - a_i) \times (d_i - c_i^t)}{(b_i - a_i)} + c_i \quad (21)$$

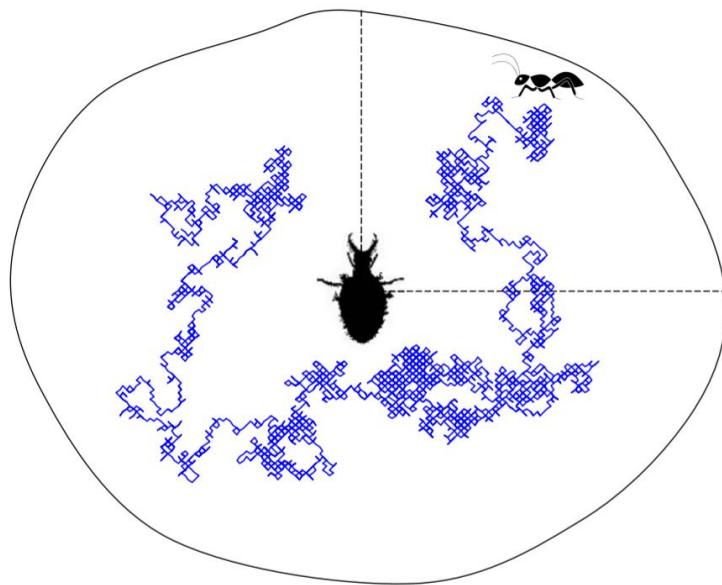
где a_i представља минималну вредност случајног хода у i -тој итерацији, b_i представља максималну вредност случајног хода у i -тој итерацији, c_i^t представља минималну вредност границе домена мрава у i -тој итерацији на основу додељеног мраволовца, d_i^t представља максималну вредност домена мрава у i -тој итерацији на основу додељеног мраволовца.

Како би се израчунале границе домена кретања мрава на основу додељеног мраволовца користе се једначине (22) и (23).

$$c_i^t = AL_j^t + c^t \quad (22)$$

$$d_i^t = AL_j^t + d^t \quad (23)$$

где су c^t и d^t редом минимална и максимална вредност свих променљивих у итерацији t , а AL_j^t представља позицију изабраног мраволовца у итерацији t . Постављена ограничења на основу дводимензионалног простора дефинисаног једначинама 23 и 24, се могу илустровати сликом 20.



Слика 20. Ограничавање кретања мрава око изабраног мраволовца [3]

Када мраволовац ухвати мрава радијус домена могућег стохастичког кретања мрава се смањује адаптивно према једначини (24) и једначини (25).

$$c^t = \frac{c}{I} \quad (24)$$

$$d^t = \frac{d}{I} \quad (25)$$

где је параметар I изражен помоћу једначине (26):

$$I = 10^w \cdot \frac{t}{T} \quad (26)$$

где t представља тренутну итерацију, T представља максималан број итерација, а w константа која се дефинише на основу једначине (27).

$$w = \begin{cases} 2 & \text{if } t > 0,1T \\ 3 & \text{if } t > 0,5T \\ 4 & \text{if } t > 0,75T \\ 5 & \text{if } t > 0,9T \\ 6 & \text{if } t > 0,95T \end{cases} \quad (27)$$

Након ажурирања ограничења за следећу итерацију потребно је изабрати позиције мраволоваца у следећој итерацији. За нови положај мраволоваца усвајају се позиције са најбољом функцијом циља из те итерације, рачунајући и мраве и мраволовце. Па тако нови мраволовци могу заузети положај мрава из претходне итерације што се може дефинисати на основу једначине (28).

$$NoviAL_j^t = A_i^t \quad \text{if } f(A_i^t) > f(AL_j^t) \quad (28)$$

Да би се искористило својство мраволовца са најбољом функцијом циља (елитни мраволовац), при рачунању нове позиције мрава узима се у обзир и позиција елитног мраволовца. Након што се израчунати нови положај мрава на основу додељеног мраволовца, исти поступак се ради и за рачунање новог положаја на основу елитног мраволовца. Утицај случајног лутања око елитног мраволовца на нови положај је представљен помоћу једначине (29).

$$A_i^t = \frac{R_A^t + R_E^t}{2} \quad (29)$$

где R_A^t представља утицај случајног лутања око додељеног мраволовца, док R_E^t представља утицај случајног лутања око елитног мраволовца.

За потребе пројекта, у Табели 10 је приказан псеудокод који описује алгоритам инспирисан интелигенцијом мраволовца који је коришћен.

Табела 10. Псеудо код ALO алгоритма

Псеудо код ALO алгоритма	
Корак 1:	Иницијализација почетне популације мрава и мраволоваца и максималног броја итерација;
Корак 2:	Одредити функцију циља сваког мрава и мраволовца;
Корак 3:	Најбољег мраволовца прогласити за елитног;
Корак 4:	Поновити итеративни поступак до максималног броја итерација
Корак 5:	Почетак петље
Корак 6:	За сваког мрава селектовати мраволовца рулет селекцијом
Корак 7:	Ажурирати c^t и d^t
Корак 8:	Креирати путању случајног лутања и извршити нормализацију
Корак 9:	Ажурирати позицију мрава
Корак 10:	Крај петље
Корак 11:	Одредити функцију циља за све мраве
Корак 12:	Испитати да ли је неки мрав „бољи“ од мраволовца, а затим их заменити
Корак 13:	Ажурирати елитног мраволовца
Корак 14:	Увећати број итерације за 1
Корак 15:	Све док не достигне максималан број итерација
Корак 16:	Крајњи реултат оптимизације (елитни мраволовац у задњој итерацији)

7. ЕКСПЕРИМЕНТАЛНО ПОРЕЂЕЊЕ РЕЗУЛТАТА ALO, PSO И WOA

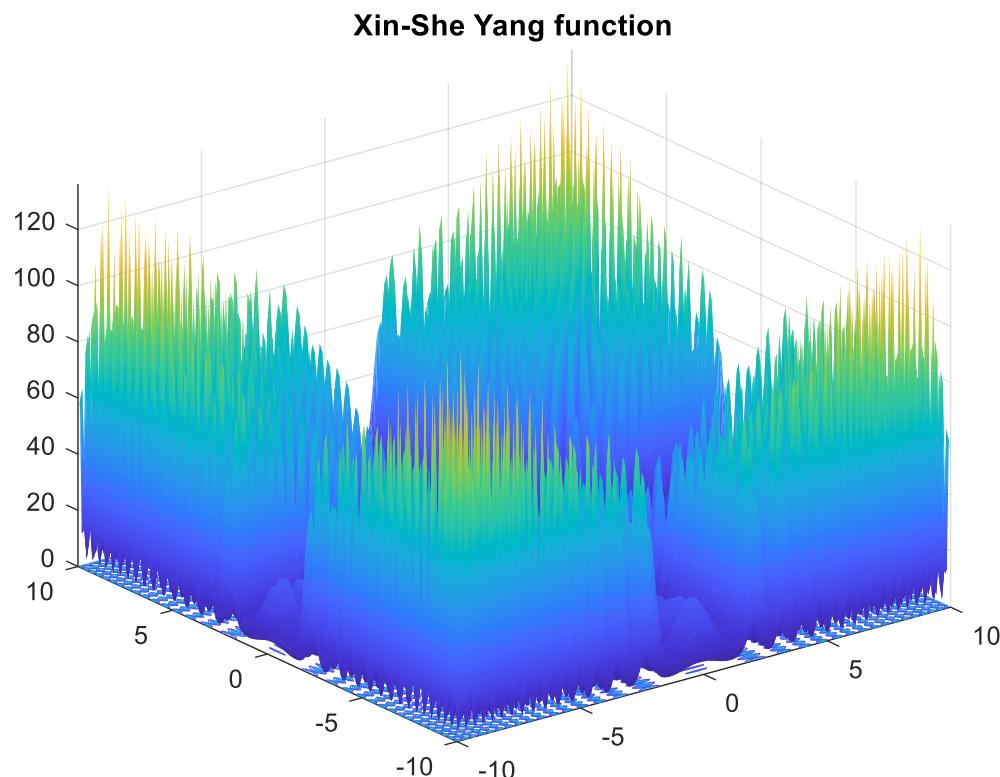
У експерименту је извршено поређење три алгоритма (ALO, PSO WOA) за проналажење минимума функција „*Xin-She Yang function*“ и „*Rastrigin function*“ уз варирање броја итерација, величине популације и специфичких параметара за сваки од алгоритама.

7.1 *Xin-She Yang* функција

Функција „*Xin-She Yang*“ има глобални минимум у $x_i = 0$ за свако $i \leq n$ где је n димензија домена. Није конвексна ни диференцијабилна и због тога се често користи за тестирање перформанси оптимизационих алгоритама. Представљена је једначином (30), а њен график је приказан на слици 21.

$$f(x) = \left(\sum_{i=1}^n |x_i| \right) e^{\left(-\sum_{i=1}^n \sin(x_i^2) \right)} \quad (30)$$

$$n = 20, -10 \leq x_i \leq 10$$



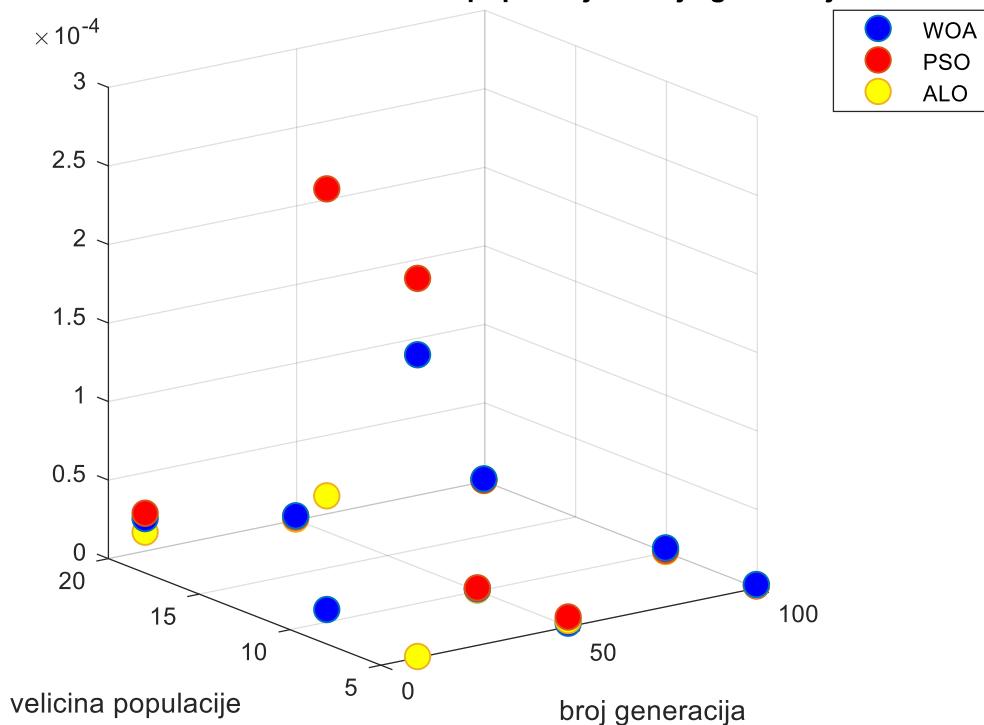
Слика 21. График функције „*Xin-She Yang*“

У циљу поређења резултата сва три алгоритма WOA, PSO, ALO за оптимизацију „Xin-She Yang“ функције варирани су величина популације, број генерација и специфични параметри за сваки од алгоритама.

- Величина популације `vektor_velicina_populacije = [5, 10];`
- Број итерација `vektor_broj_iteracija =[10, 50, 100];`
- Параметри WOA: **a** и **b** ($a = [1, 2, 3]; b = [1, 2];$)
- Параметри PSO: **c** и **w** ($c = [1.9, 2.0, 2.1]; w = [0.9, 1.0, 1.1];$)
- Параметри ALO: алгоритам селекције – рулет или униформна (`selekcije = ['R', 'U'];`)

Добијени резултати су представљени у табели 11 и на слици 22.

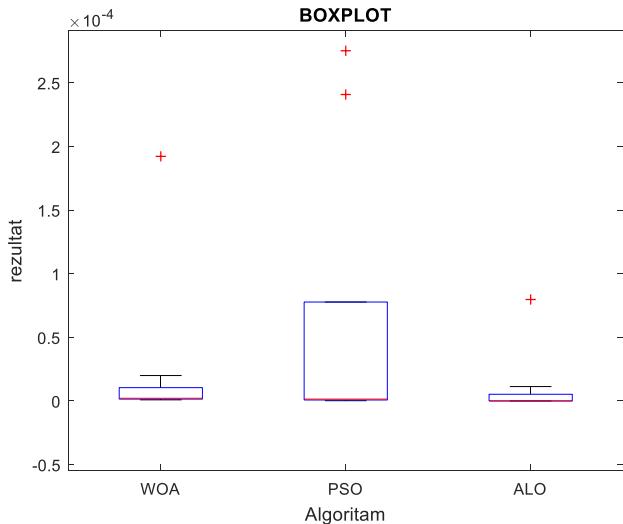
Резултати у зависности од величине популације и броја генерација



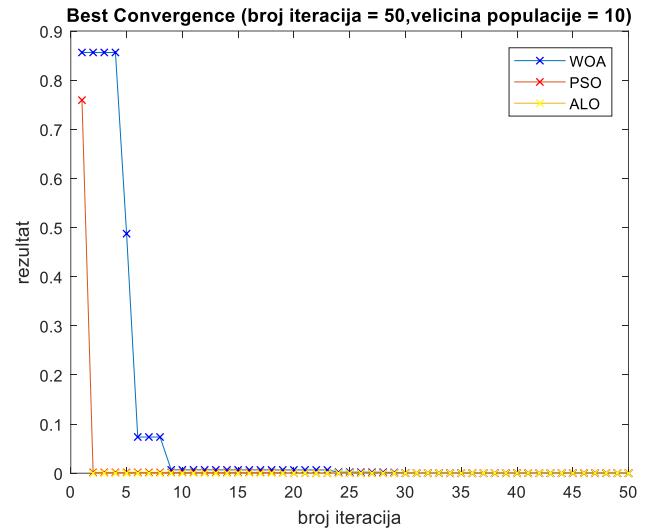
Слика 22. График резултата експеримента за функцију „Xin-She Yang“

Табела 11. Резултати експеримента за функцију „Xin-She Yang“

Величина популације	Број генерација	WOA_min	a	b	PSO_min	c	w	ALO_min	ALO_selection_param
5	10	0.000192137	1	2	0.000240742	2.1	1	0	R
10	10	7.27697E-06	2	2	0.000275108	1.9	1.1	7.96876E-05	U
20	10	1.99555E-05	3	2	2.33709E-05	2	0.9	1.13025E-05	U
5	50	1.57638E-06	2	2	5.51386E-06	1.9	1	3.22501E-06	R
10	50	8.69328E-07	1	2	1.45733E-06	1.9	0.9	0	U
20	50	2.09757E-06	1	2	1.21888E-06	1.9	1	0	R
5	100	1.62117E-06	3	2	7.21254E-07	1.9	0.9	1.32328E-06	U
10	100	2.32655E-06	2		7.58119E-07	1.9	0.9	0	R
20	100	1.07986E-06	3	1	1.88907E-07	1.9	0.9	0	U



Слика 23. Boxplot резултати експеримента за функцију „Xin-She Yang“



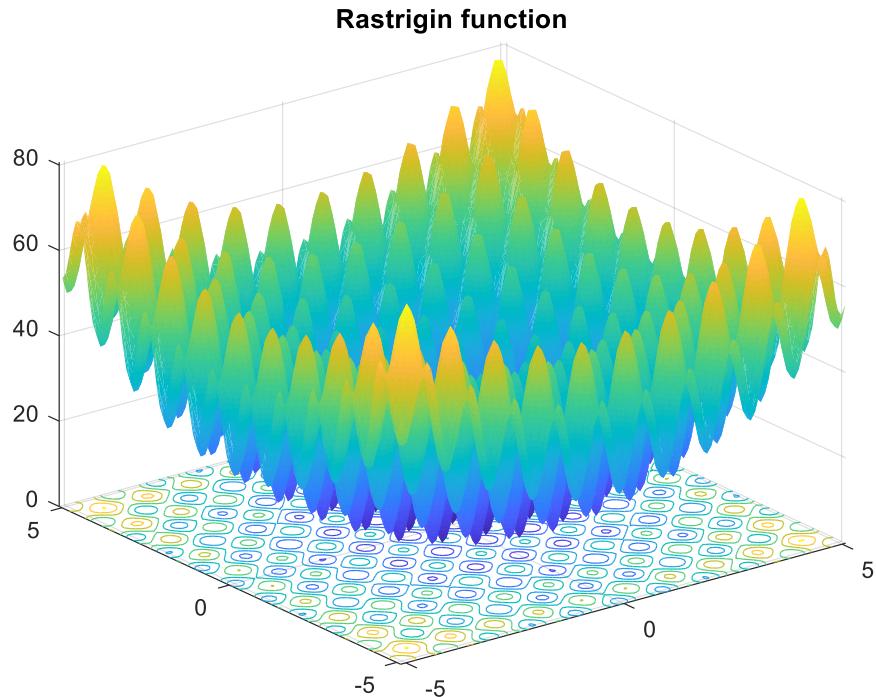
Слика 24. Крива конвергенције јединке са најбољим резултатом за функцију „Xin-She Yang“

7.2 Rastrigin функција

Функција „Rastrigin“ има више локалних минимума који су равномерно распоређени у оквиру домена и због тога се често користи за тестирање перформанси оптимизационих алгоритама. Представљена је једначином (31), а њен график је приказан на слици 25.

$$f(x) = 10n + \sum_{i=1}^n [x_i^2 - 10\cos(2\pi x_i)] \quad (31)$$

$$n = 30, -5,12 \leq x_i \leq 5,12$$



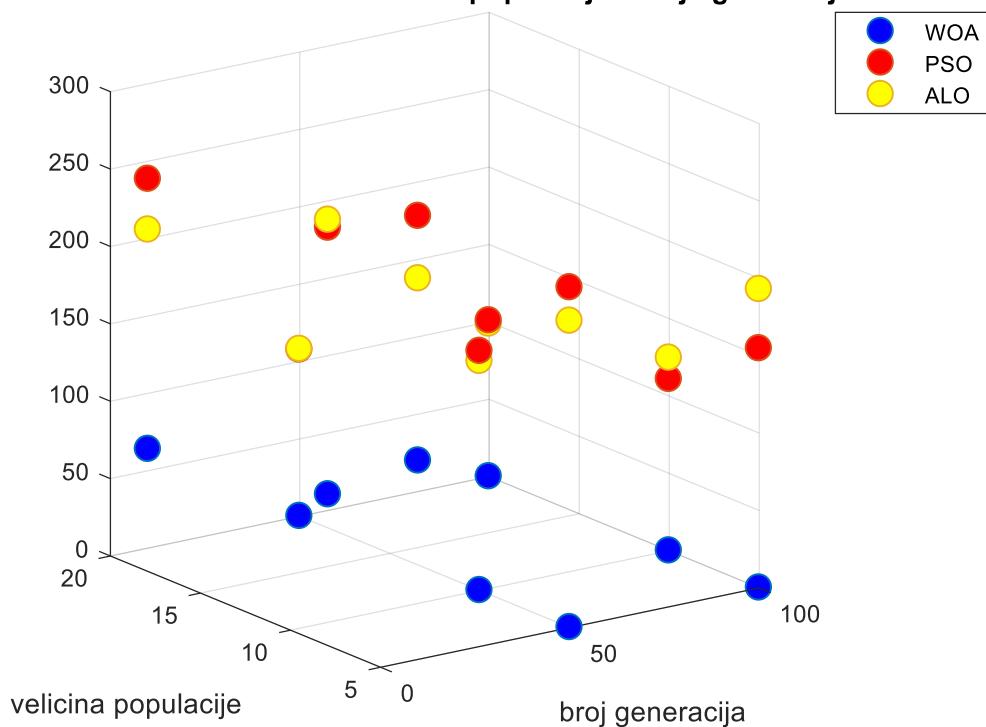
Слика 25. График функције „Rastrigin“

У циљу поређења резултата сва три алгоритма WOA, PSO, ALO за оптимизацију „Rastrigin“ функције варирани су величина популације, број генерација и специфични параметри за сваки од алгоритама.

- Величина популације `vektor_velicina_populacije = [5, 10];`
- Број итерација `vektor_broj_iteracija =[10, 50, 100];`
- Параметри WOA: **a** и **b** ($a = [1, 2, 3]$; $b = [1, 2]$);
- Параметри PSO: **c** и **w** ($c = [1.9, 2.0, 2.1]$; $w = [0.9, 1.0, 1.1]$);
- Параметри ALO: алгоритам селекције – рулет или униформна (`selekcije = ['R', 'U']`);

Добијени резултати су представљени у табели 12 и на слици 26.

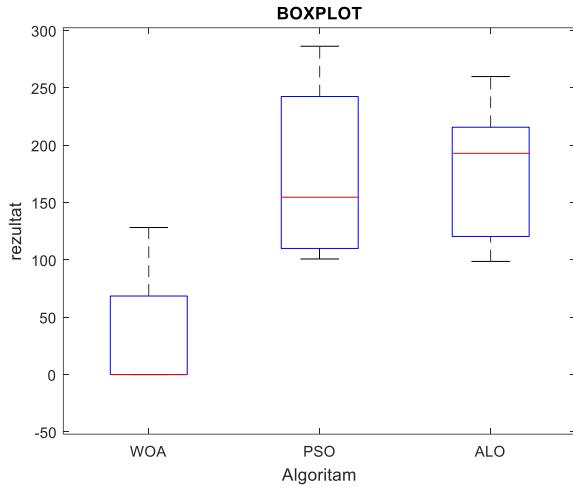
Rezultati u zavisnosti od velicine populacije i broja generacija



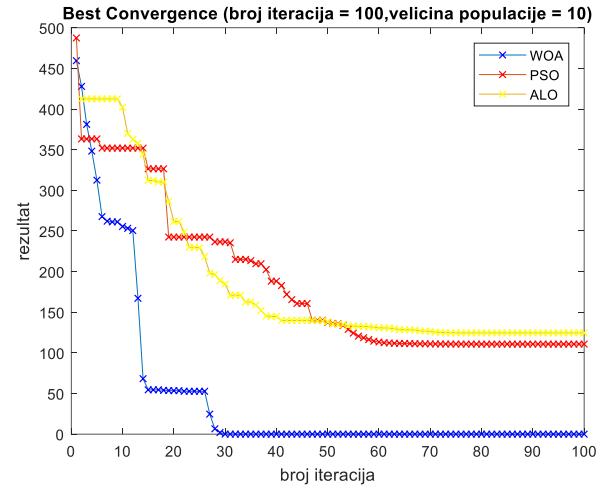
Слика 26. График резултата експеримента за функцију „Rastrigin“

Табела 12: Резултати експеримента за функцију „Rastrigin“

Величина популације	Број генерација	WOA_min	a	b	PSO_min	c	w	ALO_min	ALO_selection_param
5	10	128.2204413	3	1	286.2713534	1.9	1	245.9424699	Rulet
10	10	82.39833748	1	2	254.6829749	2	0.9	259.8390771	Rulet
20	10	63.84410355	2	1	238.350848	1.9	1	205.5679972	Uniformna
5	50	0.061150713	3	1	219.6505233	1.9	1.1	198.0583636	Uniformna
10	50	0.000429056	2	1	154.5711948	1.9	1	148.10888	Uniformna
20	50	5.91137E-09	2	2	107.5389444	1.9	1.1	107.9523826	Rulet
5	100	2.27374E-13	3	2	154.6003186	1.9	1	192.8837414	Uniformna
10	100	0	1	2	110.7809453	1.9	0.9	124.5714899	Rulet
20	100	0	1	2	100.7745161	1.9	1	98.65317246	Uniformna



Слика 27. Boxplot резултати експеримента за функцију „Rastrigin“



Слика 28. Крива конвергенције јединке са најбољим резултатом за функцију „Rastrigin“

8. ТЕРМИНИРАЊЕ ТЕХНОЛОШКИХ ПРОЦЕСА

8.1 Флексибилни технолошки процеси

Флексибилни технолошки систем је високо аутоматизована технолошка ћелија, која се састоји од једног или више обрадних система, међусобно повезаних аутоматским системом за руковање и складиштење материјала и којим управља рачунарски систем [4]. Важна особина ових система јесте њихова флексибилност која се одликује у томе што се различити делови могу истовремено обраћивати на различитим машинама и тиме омогућити да се производња прилагоди новонасталим условима на тржишту. Неки од типова флексибилности технолошких процеса су:

- флексибилност машина алатки;
- флексибилност алата;
- флексибилност оријентације алата;
- флексибилност процеса и
- флексибилност редоследа операција.

Пројектовање технолошких процеса представља одређивање детаљних метода којима се делови могу произвести од иницијалне фазе па све до финалног производа. Излаз из пројектовања технолошког процеса је документ који се назива технолошки поступак. Технолошки поступак служи за управљање производњом и садржи све информације потребне за техничку припрему и за припрему производне опреме. Компјутерски подржано пројектовање технолошких процеса или CAPP (енгл. *Computer Added Process Planning*) представља интеграцију компјутерски подржаног пројектовања - CAD (енгл. *Computer – Aided Design*) и компјутерски подржане производње – CAM (енгл. *Computer – Aided Manufacturing*). Основна функција овог система јесте да омогући пројектовање детаљних метода којима се делови или склопови могу произвести.

Улази за пројектовање технолошког процеса могу бити:

- пројектни подаци;
- подаци о сировом материјалу;
- подаци о обрадним системима;
- подаци о захтеваном квалитету;
- подаци о типу производње и слично.

Како излаз добија се технолошки процес обраде, којим се описује редослед одвијања свих операција обраде дела.

Приликом оптимизације технолошког процеса могу се у обзир узети различити критеријуми као што су време производње, трошкови, енергија и слично. Најчешће се за оптимизацију користи критеријум минималног производног времена (енгл. *Production time-PT*). Овај критеријум представља збир свих времена обраде дела и времена транспорта дела од једне до друге машине алатке. Са повећањем броја машина увећава се и време транспорта делова између различитих машина алатки. Оптимални флексибилни технолошки процес је онај са најмањим производним временом. Математичка формулатија за модел производног времена је представљена једначином (32).

$$TP(i) = \sum_{j=1}^{Pil} TW(i, j, k, l) + \sum_{j=1}^{Pil-1} TT(i, l, (j, k_1), (j+1, k_2)) \quad (32)$$

$$i \in [1, n], j \in [1, P_l]$$

где су :

- $TP(i)$ - производно време за i -ти део;
- $TW(i, j, k, l)$ - време трајања операције O_{ijl} на k -тој алтернативној машини;
- $TT(i, l, (j, k_1), (j + 1, k_2))$ – време транспорта дела i између алтернативних машина k_1 и k_2 ;
- n – укупан број делова;
- O_{ij} – j -та операција l -тог технолошког процеса i -тог дела;
- P_{il} – број операција l -тог технолошког процеса i -тог дела;
- k – алтернативне машине за операцију O_{ij} .

Претпоставке које се користе приликом примене овог математичког модела су следеће:

- све машине су доступне у почетном тренутку ($t_0 = 0$);
- на једној машини се може вршити само једна операција једног дела у једном временском тренутку;
- операције једног дела се не могу обављати истовремено на више машина алатки (док се не заврши операција O_{ijl} , не може почети операција O_{ij+1l} i -тог дела).

У оквиру пројектног задатка потребно је да се за задатих 18 делова изврши оптимизација технолошког процеса где је за критеријум оптимизације изабрано производно време. Приликом покретања алгоритма потребно је сачувати три најбоља (минимално време производње) различита алтернативна технолошка процеса који ће се касније користити за задатак терминирања производње. Функција циља или функција одлучивања за овај проблем се може представити једначином (33):

$$\max f(i) = \frac{1}{TP(i)} \quad (33)$$

Одакле се може приметити да функција циља има највећу вредност код технолошког процеса које има најмање производно време.

Мреже алтернативних технолошких процеса, табела алтернативних машина алатки, време транспорта између машина алатки и време обраде дела на машини алатки за све делове се налазе у прилогу овог рада (Прилог 1). Резултати са три најбоља алтернативна технолошка процеса су приказани у Табели 13.

Табела 13. Алтернативни технолошки процеси за све делове

Део	Алтернативни технолошки процеси						Функција циља
1	1-14	2-11	5-13	6-4	7-5	8-8	0.0048
	1-9	2-15	5-13	6-12	7-1	8-8	0.0048
	1-9	2-15	5-13	6-4	7-11	8-8	0.0047
2	1-8	2-8	3-4	6-1			0.0090
	1-14	2-8	3-4	6-5			0.0089
	1-8	4-6	5-11	6-1			0.0085
3	1-7	2-8	3-14	4-2			0.0108
	1-7	5-3	6-1	4-6			0.0103
	1-7	5-4	6-1	4-6			0.0098
4	13-2	14-14	16-9				0.0135

	13-6	14-14	16-9				0.0128
	13-1	15-12	16-7				0.0127
5	14-10	15-5	16-8	17-3	18-3		0.0076
	14-10	15-5	16-4	17-3	18-3		0.0074
	14-10	15-5	16-4	17-7	18-3		0.0074
6	1-3	2-3	3-5	4-5			0.0111
	15-14	16-4	19-5	20-9			0.0100
	15-13	16-4	19-12	20-11			0.0097
7	1-7	19-1	20-6	21-7			0.0094
	1-7	19-2	20-6	21-7			0.0088
	1-7	19-2	20-6	21-1			0.0088
8	17-10	19-11	20-12				0.0097
	17-2	18-7	20-12				0.0094
	17-10	18-7	20-12				0.0092
9	16-4	17-3	20-8				0.0103
	16-4	18-7	19-2	20-8			0.0095
	16-13	17-1	20-6				0.0093
10	1-1	2-6					0.0169
	1-3	2-15					0.0161
	1-3	2-6					0.0159
11	8-9	9-9					0.0213
	8-11	9-13					0.0204
	8-14	9-5					0.0204
12	13-2	14-1	15-9	18-9			0.0095
	13-2	16-7	17-12	18-9			0.0093
	8-12	9-14	10-4	11-7	12-6		0.0089
13	17-8	18-15					0.0161
	17-11	18-15					0.0149
	17-9	18-15					0.0149
14	9-12	12-10	13-10				0.0189
	9-5	12-14	13-10				0.0167
	9-12	12-9	13-10				0.0156
15	12-7	13-1	15-1				0.0094
	12-7	13-1	15-12				0.0093
	7-9	8-8	9-2	11-13			0.0090
16	18-4	20-14	21-14				0.0172
	18-10	20-8	21-14				0.0169
	18-10	20-14	21-14				0.0161
17	18-2	19-6	20-11	22-11			0.0179
	18-10	21-12	22-11				0.0164
	18-10	21-12	22-14				0.0156
18	1-3	4-5	5-9	6-5			0.0096
	1-8	4-10	5-9	6-8			0.0095
	1-3	4-10	5-9	6-5			0.0093

У табели 13 су поред три најбоља технолошка процеса за сваки део приказани и редослед операције, као и машина на којима се наведене операције изводе, респективно. Такође у табели је приказана и функција циља која представља реципрочну вредност производног времена, где се може увидети који је од технолошких процеса за дати део најбољи.

8.2 Терминирање

Терминирање (енгл. *scheduling*) представља примену оптимизационих метода и метода одлучивања у циљу планирања и оптималног временског распоређивања активности производно-технолошких ентитета. У овом пројектном задатку под појмом терминирање се подразумева терминирање флексибилних технолошких система (машина алатки, алати, помоћних прибора, транспортних средстава, мобилних робота и сл.) у складу са усвојеним критеријумима. У процесу терминирања флексибилних технолошких система свака операција се додељује одговарајућој машини алатки, пазећи на оптимално временско распоређивање. Као резултат се добија редослед операција делова на одговарајућим машинама алаткама.

Проблеми терминирања за овај задатак се могу поделити у две групе на основу функција циља које се рачунају:

- производно време обраде свих делова (енгл. *makespan*) и
- правнотежено искоришћење машина алатки (енгл. *Balanced level of machine utilization*)

Након што се израчунати најбољи технолошки процес за сваки део, што је описано у претходном поглављу, потребно је изабрати неке од тест проблема за терминирање који су приказани у табели 14 и извршити процес оптимизације по задатом критеријуму.

Табела 14: Тест проблеми за терминирање

Проблем	Број деловеа	Број операција	Делови за терминирање
1	6	79	1-2-3-10-11-12
2	6	100	4-5-6-13-14-15
3	6	121	7-8-9-16-17-18
4	6	95	1-4-7-10-13-16
5	6	96	2-5-8-11-14-17
6	6	109	3-6-9-12-15-18
7	6	99	1-4-8-12-15-17
8	6	96	2-6-7-10-14-18
9	6	105	3-5-9-11-13-16
10	9	132	1-2-3-5-6-10-11-12-15
11	9	168	4-7-8-9-13-14-16-17-18
12	9	146	1-4-5-7-8-10-13-14-16
13	9	154	2-3-6-9-11-12-15-17-18
14	9	151	1-2-4-7-8-12-15-17-18
15	9	149	3-5-6-9-10-11-13-14-16
16	12	179	1-2-3-4-5-6-10-11-12-13-14-15
17	12	221	4-5-6-7-8-9-13-14-15-16-17-18
18	12	191	1-2-4-5-7-9-10-11-13-14-16-17
19	12	205	2-3-5-6-8-9-11-12-14-15-17-18
20	12	195	1-2-4-6-7-8-10-12-14-15-17-18
21	12	201	2-3-5-6-7-9-10-11-13-14-16-18
22	15	256	2-3-4-5-6-8-9-10-11-12-13-14-16-17-18
23	15	256	1-4-5-6-7-8-9-11-12-13-14-15-16-17-18
24	18	300	1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18

8.3 Примена генетичких алгоритама у оптимизацији плана терминирања

У циљу отимизације технолошког процеса и плана терминирања често се користе генетички алгоритми. Да би се извршила оптимизација плана терминирања потребно је правилно извршити сваки од следећих 5 корака:

- генерисање јединки у иницијалној популацији;
- евалуација функције циља и иницијализација параметара генетичких алгоритама;
- селекција;
- укрштење и
- мутација.

8.3.1 Генерисање јединки у иницијалну популацију

Приликом генерисања јединки у иницијалну популацију код оптимизације плана терминирања важно је напоменути да се посматра укупан процес за све делове који учествују у плану терминирања.

Иницијална популација за генетички алгоритам се састоји од два подстринга тј. хромозома:

- План терминирања (главни подстринг)
- Технолошки процес (помоћни подстринг)

Дужина хромозома за план терминирања одређена је бројем делова (параметар n) и максималним бројем операција за све алтернативне технолошке процесе (параметар q). За објашњење може се узети проблем 6 где се из Табеле 12 види да је број делова за наведени проблем 6, док из Табеле 11 да је максималан број операција алтернативних технолошких процеса 5 (део 12 – 5 операција). Даље следи да подстринг за план терминирања има $n \cdot q = 6 \cdot 5 = 30$ елемената, односно 30 гена који су приказани на слици 29.

3	0	1	2	4	1	0	5	0	6	4	1	0	4	3	0	2	5	6	0	1	0	6	0	3	4	2	5	0	2
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Слика 29. Подстринг за план терминирања

Помоћни подстринг, односно хромозом за технолошки процес има онолико гена колико делова учествује у укупном технолошком процесу, а то је у овом случају 9 где сваки ген представља одобрани технолошки процес за одређени део. Пример изгледа помоћног подстринга је приказан на слици 30.

2	1	3	2	2	3
---	---	---	---	---	---

Слика 30. Помочни подстринг за технолошки процес

8.3.2 Евалуација функције циља и иницијализација параметара генетичких алгоритама

У оквиру овог пројекта су коришћена два математичка модела за одређивање функције циља ($object1$ и $object2$) приказана једначинама (34) и (35). Функција $object1$ се односи на време неопходно за обраду свих делова чије се терминирање врши, а функција $object2$ на равномерно искоришћење машина алатки. Функција циља за сваку од индивидуа у иницијалној популацији рачуна се према $object1$ или $object2$ у зависности од одабраног критеријума за оптимизацију.

$$object1 = \max(c_{ij}) (c_{ij} \in T_d(s_{ij}, c_{ij})) \quad (34)$$

$$object2 = object1 + \sum_{a=1}^m \left| \sum p_{ij} - avgmt \right| (o_{ij} \in M_a) \quad (35)$$

где је:

c_{ij} - време завршетка операције o_{ij} ;

o_{ij} - j -та операција i -тог дела;

s_{ij} - време почетка операције o_{ij} ;

m - укупан број машина алатки;

$\sum p_{ij}$ - укупно време обраде на свакој појединачној машини алатки;

$avgmt$ - просечно време обраде на свим машинама алаткама;

T_d - скуп времена почетка s_{ij} и времена завршетка c_{ij} свих операција свих делова;

M_a - скуп алтернативних машина алатки за операцију o_{ij} .

Параметри генетичких алгоритама при иницијализацији су:

- Величина популације S,
- Укупан број генерација M,
- Вероватноћа мутације P_m,
- Вероватноћа укрштања P_c.

8.3.3 Селекција

Под појмом селекције подразумева се бирање два родитеља тј. хромозома из текуће популације од којих ће настати нови хромозоми. Бирање се врши на основу рулет селекције (енгл. roulette wheel selection), где је вероватноћа селекције пропорционална израчунатој функцији циља f(i) из текуће популације. Селекција се у алгоритму извршава онолико пута колико има елемената у популацији и одабране јединке селекцијом чине нову популацију. Случајно изабрана два хромозома рулет селекцијом, за проблем 6, представљају родитеље који су приказани на слици 31.

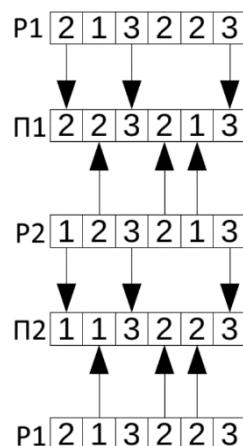
3 0 1 2 4 1 0 5 0 6 4 1 0 4 3 0 2 5 6 0 1 0 6 0 3 4 2 5 0 2	2 1 3 2 2 3
1 4 0 2 0 3 5 1 4 2 6 2 1 4 5 0 3 0 6 0 3 5 2 0 6 1 4 0 0 0	1 2 3 2 1 3

Слика 31. Две случајно изабрана хромозома

8.3.4 Укрштање

Укрштање представља оператор који се примењује како би се од родитеља добили потомци и примењује се и на главни и на помоћни подстринг.

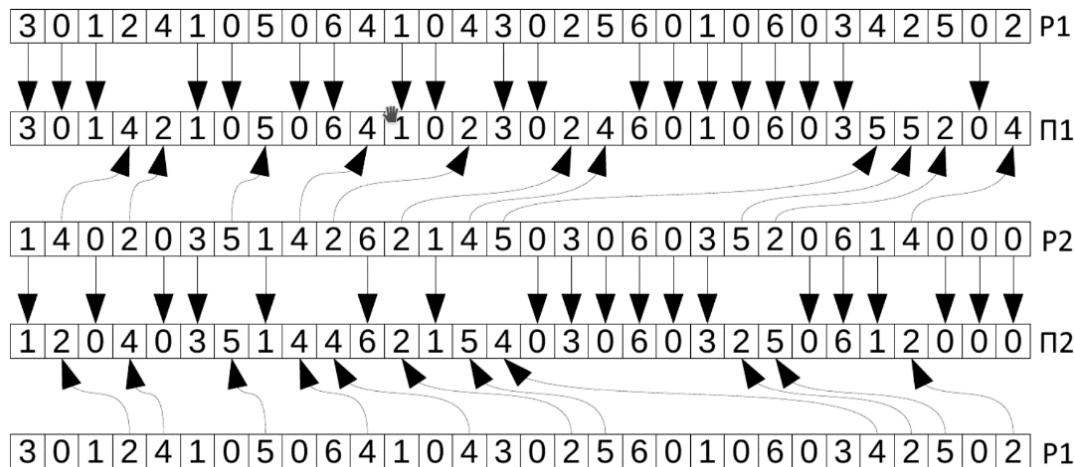
Прво се укршта помоћни подстринг тако што се на случајан начин врши одабир позиција у помоћном подстрингу првог родитеља и вредности одабраних позиција се додељују првом потомку на истим позицијама. Непопуњене места помоћног подстринга првог потомка се попуњавају од другог родитеља као на слици 32.



Слика 32. Укрштање помоћног подстринга

Укрштање главног подстринга се врши на основу укрштања помоћног подстринга. Први потомак на основу делова чије је алтернативне технолошке процесе преузео од првог родитеља,

преузима и распоред операција од првог родитеља. Непопуњена места у потомку се попуњавају редом од другог родитеља на начин као што је приказано на слици 33.

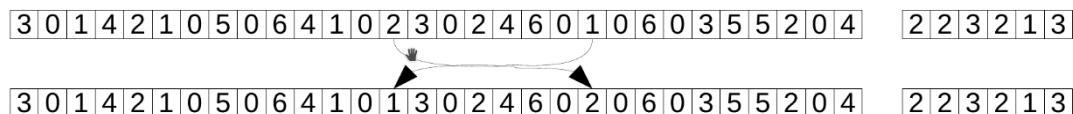


Слика 33. Укрштање главног подстринга

8.3.5 Мутација

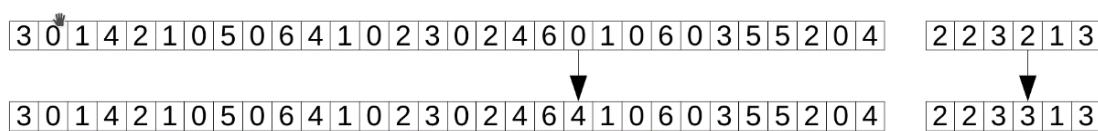
Оператор мутације се врши на основу дефинисане вероватноће мутације. У зависности да ли се мутација ради на главном или помоћном подстрингу постоје два оператора мутације.

Први оператор мутације који се односи на главни подстринг је двопозициона замена (енгл. *swapping*). Мутација се врши тако што се случајно одабере један родитељ, а затим се случајно одаберу два гена у том родитељу којима се замене места као што је приказано на слици 34. Потомак представља нови хромозом који је добијен заменом места случајно одабраних гена.



Слика 34. Мутација (двопозициона замена)

Други оператор мутације се користи за генерирање нових потомака променом једног алтернативног технолошког процеса једног дела у помоћном подстрингу који је случајно одабран као на слици 35. У случају да приликом мутације дође до промене алтернативног технолошког процеса у процес који има више или мање операција онда долази до промене у главном подстрингу.



Слика 35. Мутација једног алтернативног технолошког процеса

8.4 Експериментални резултати

Валидација развијеног генетичког алгоритма за терминирање почиње случајним одабиром једног од 24 тест проблема, приказаних у Табели 14. Тест проблеми, односно планови терминирања, формирани су од различитог броја делова за које је потребно извршити терминирање и различитог броја операција. Број делова по проблему креће се од минималних 6 до максималних 18 делова, док је број операција 79 за први и 300 за последњи проблем. Приказани експериментални резултати односе се на проблеме са редним бројевима 6, 13 и 23. За одабране технолошке процесе варирани су следећи параметри:

- Број генерација - `vektor_broj_generacija = [10, 20, 40];`
- Величина популације - `vektor_velicina_populacije = [5, 15, 25];`
- Број елитних хромозома - `vektor_broj_elitnih = [1, 2, 4];`
- Вероватноћа мутације - `vektor_verovatnoca_mutacije = [0.1, 0.2];`
- Вероватноћа укрштања - `vektor_verovatnoca_crossovera = [0.8, 0.9];`
- Функција циља - `vektor_object = [1, 2];`

Резултати одабраних проблема дати су у табели која се налази у прилогу (Прилог 2), чији исечак је приказан Табелом 15, као и у форми гантограма (слике 36, 39 и 42). Гантограм је тип стубастог дијаграма којим се илуструје распоред активности за сваки део. Појединачни делови су представљени правоугаоницима различитих боја, чија дужина означава време обраде датог дела на датој машини. Време завршетка последњег дела у гантограму означава укупно производно време свих делова (*makespan*).

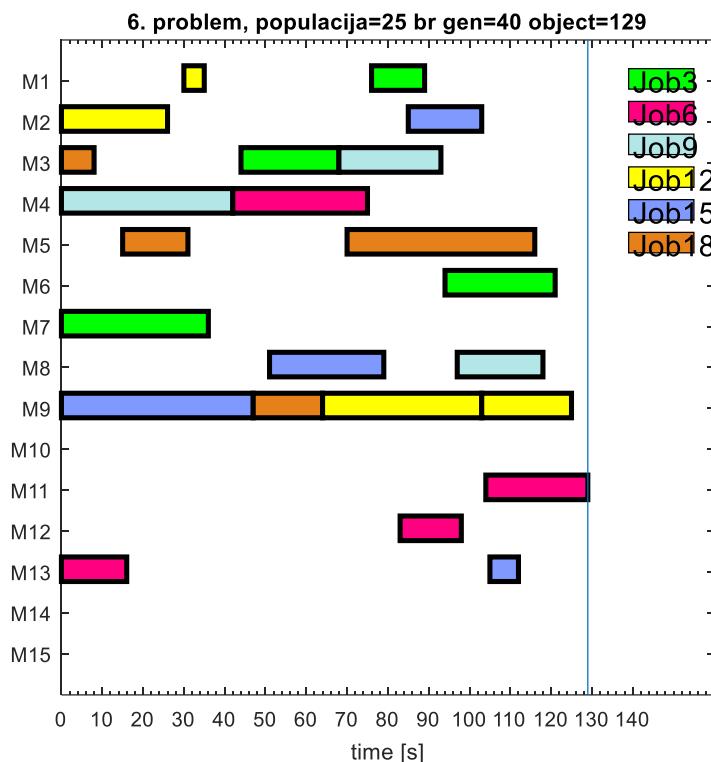
Табела 15. Резултати одабраних проблема (исечак)

Проблем број	Циљ	Величина популације	Број генерација	Број елитних	Вероватноћа мутације	Вероватноћа укрштања	Резултат
6	1	15	10	4	0,1	0,9	138
6	2	15	10	4	0,1	0,9	411
6	1	15	10	4	0,2	0,8	138
6	2	15	10	4	0,2	0,8	413
6	1	15	10	4	0,2	0,9	129
...
13	2	25	40	1	0,1	0,9	458,6667
13	1	25	40	1	0,2	0,8	137
13	2	25	40	1	0,2	0,8	387
13	1	25	40	1	0,2	0,9	132
13	2	25	40	1	0,2	0,9	414,0667
...
23	2	25	40	2	0,2	0,8	394,8667
23	1	25	40	2	0,2	0,9	152
23	2	25	40	2	0,2	0,9	291,8
23	1	25	40	4	0,1	0,8	149
23	2	25	40	4	0,1	0,8	410,9333

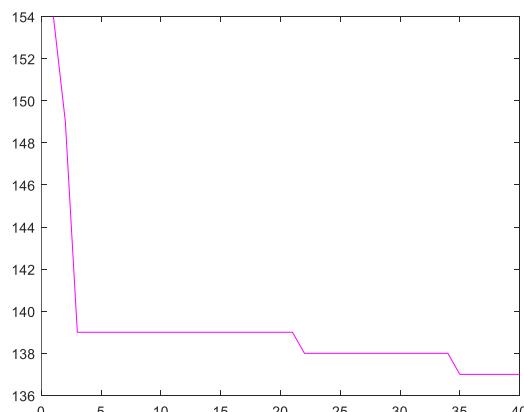
На основу резултата добијених извршавањем алгоритма за све варијације параметара може се закључити:

- Величина популације, као и број генерација утиче на резултат,
- Број елитних хромозома утиче до одређеног процента, наиме када постоји више од 30% елитних хромозома резултат је лошији,
- Са повећањем броја делова потребно је повећати величину популације и број генерација како би се добио оптимални резултат.

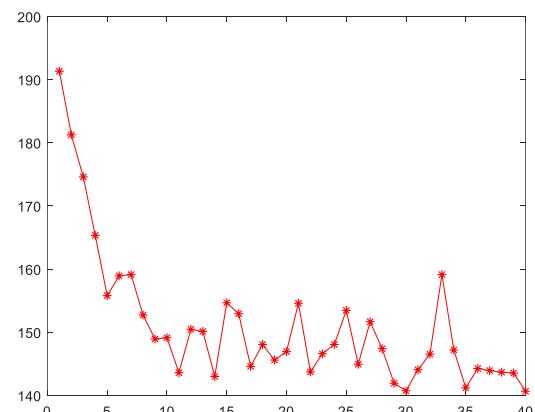
8.4.1 Експериментални резултати за проблем бр. 6 (делови 3-6-9-12-15-18)



Слика 36. Гантов дијаграм за терминирање проблема бр. 6

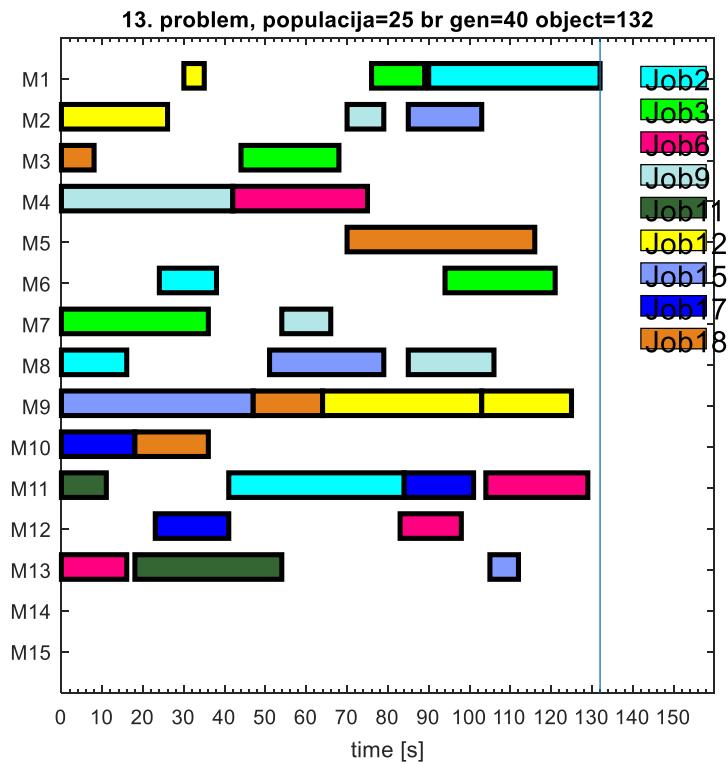


Слика 37. Минимална вредност ФЦ кроз итерације за проблем бр. 6

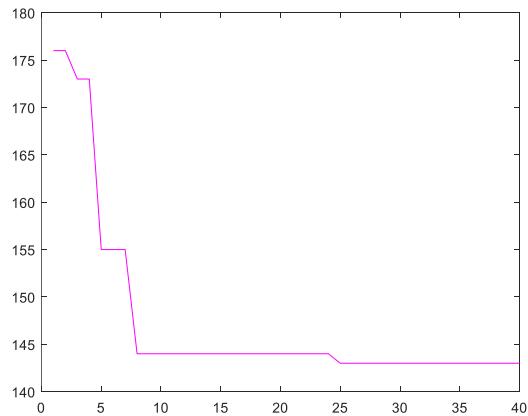


Слика 38. Средња вредност ФЦ кроз итерације за проблем бр. 6

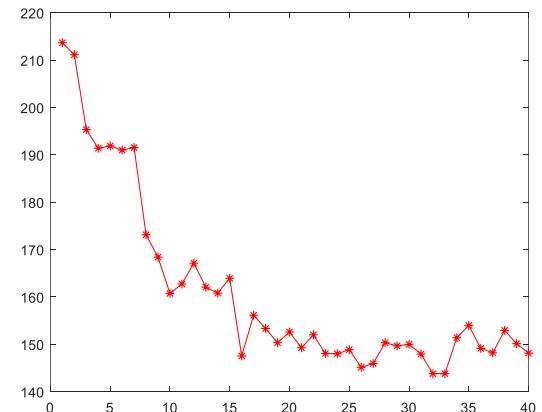
8.4.2 Експериментални резултати за проблем бр. 13 (делови 2-3-6-9-11-12-15-17-18)



Слика 39. Гантов дијаграм за терминирање проблема бр. 13

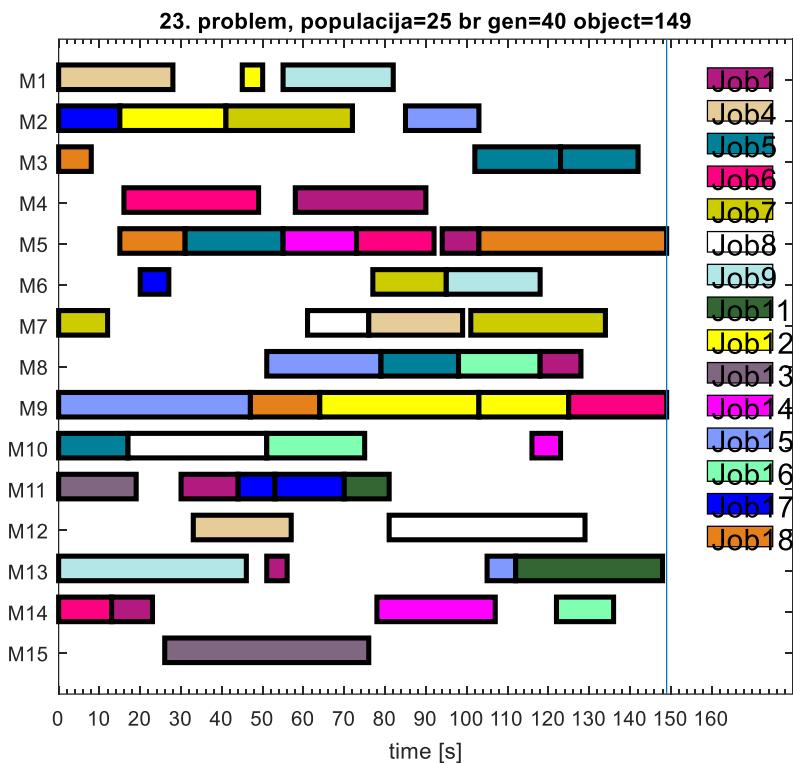


Слика 40. Минимална вредност ФЦ кроз итерације за проблем 13

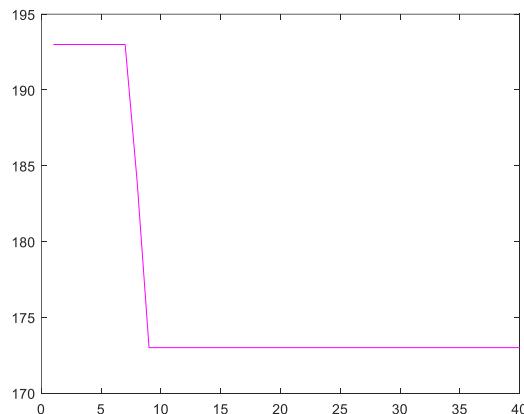


Слика 41. Минимална вредност ФЦ кроз итерације за проблем 13

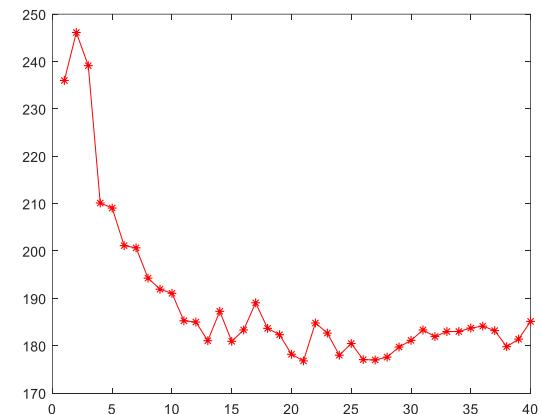
8.4.3 Експериментални резултати за проблем бр. 23 (делови 1-4-5-6-7-8-9-11-12-13-14-15-16-17-18)



Слика 42. Гантов дијаграм за терминирање проблема бр. 23



Слика 43. Минимална вредност ФЦ кроз итерације за проблем 23



Слика 44. Минимална вредност ФЦ кроз итерације за проблем 23

9. ЗАКЉУЧАК

У оквиру овог пројекта анализиран је проблем терминирања технолошких процеса и система. У ту сврху представљени су различити биолошки инспирисани алгоритми оптимизације и анализиране њихове перформансе. Применом генетичког алгоритма извршена је оптимизација технолошких процеса и плана терминирања.

Резултати истраживања у оквиру овог пројекта могу се груписати у два дела.

Први део односи се на упоредну статистичку анализу биолошки инспирисаних алгоритама оптимизације. За различите тест функције испитане су генералне перформансе, као и утицај промене одговарајућих параметара одабраних алгоритама.

Други део односи се на оптимизацију технолошких процеса и терминирање. Као резултат оптимизације терминирања технолошких система добијен је Гантов дијаграм, који приказује оптималан распоред свих активности потребних за израду претходно одабраних репрезентативних делова чије се терминирање врши. Као критеријум оптимизације у обзир је узето производно време.

За сва спроведена истраживања развијени су кодови у *MATLAB* софтверском окружењу.

У овом раду, разматрани проблем терминирања поједностављен је и сведен на флексибилност машина и редоследа операција. Сложенији проблеми оптимизације обухватали би и флексибилност процеса, алата, као и оријентације алата. Додатно, у циљу остваривања оптималног, поузданог и ефикасног система за унутрашњи транспорт, јавља се проблем терминирања мобилног робота, што представља један од примарних праваца савремених истраживања у домену интелигентних технолошких система.

10. ЛИТЕРАТУРА

- [1] проф. др Милица Петровић. Терминирање технолошких система и процеса – Изводи са предавања, Машински факултет Универзитет у Београду, 2021.
- [2] проф. др Миљковић, З. Доц. др Петровић, М. (2021) Интелигентни технолошки системи са изводима из роботике и вештачке интелигенције, Универзитет у Београду - Машински факултет,
- [3] Mirjalili, S.: The ant lion optimizer, Advances in Engineering Software
- [4] проф др Бабић, Б. (2017). Рачунарски интегрисани системи и технологије. Универзитет у Београду – Машински факултет
- [5] Kennedy, J. and Eberhart, R., 1995. Particle swarm optimization. In Proceedings of ICNN'95-International conference on neural networks