# ГОУ ВПО «ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

### Кафедра АСУ

### 

Отчёт

о лабораторной работе №2

по курсу: «Оптимизация многомерных функций с помощью ГА»

# Вариант «2»

Выполнил:

ст. гр. ИСм-22

Мариничев И.И

Проверила:

Васяева Т.А

# 

# Донецк 2022

**Ход работы:**

1. Изучить теоретический материал.
2. Разработать генетический алгоритм нахождения оптимума функции согласно таблице вариантов в приложении Б.
3. Выполнить программную реализацию разработанного алгоритма на языке высокого уровня. Рекомендованный язык Python с применением DEAP. Предусмотреть возможность просмотра процесса поиска решения.
4. Для n=2 вывести на экран график данной функции с указанием найденного экстремума. Предусмотреть возможность просмотра процесса поиска решения.
5. Исследовать зависимость числа поколений (генераций), точности нахождения решения от основных параметров генетического алгоритма: − число особей в популяции; − вероятность кроссинговера, мутации. Критерием остановки вычислений выбирать повторение лучшего результата заданное количество раз и / или достижение популяцией определенного возраста (например, 100 эпох).

*Задание:*

Для выполнения лабораторной были выбраны следующие методы и параметры:

Для отбора хромосом при репродукции был использован турнирный метод (на каждом шаге итерации – выбор из 3х случайных хромосом).

Партнёр для кроссинговера выбирается по средству аутбридинга.

Максимальное количество итераций установим равным 500. Минимумом функции является F(0;0) = 0;

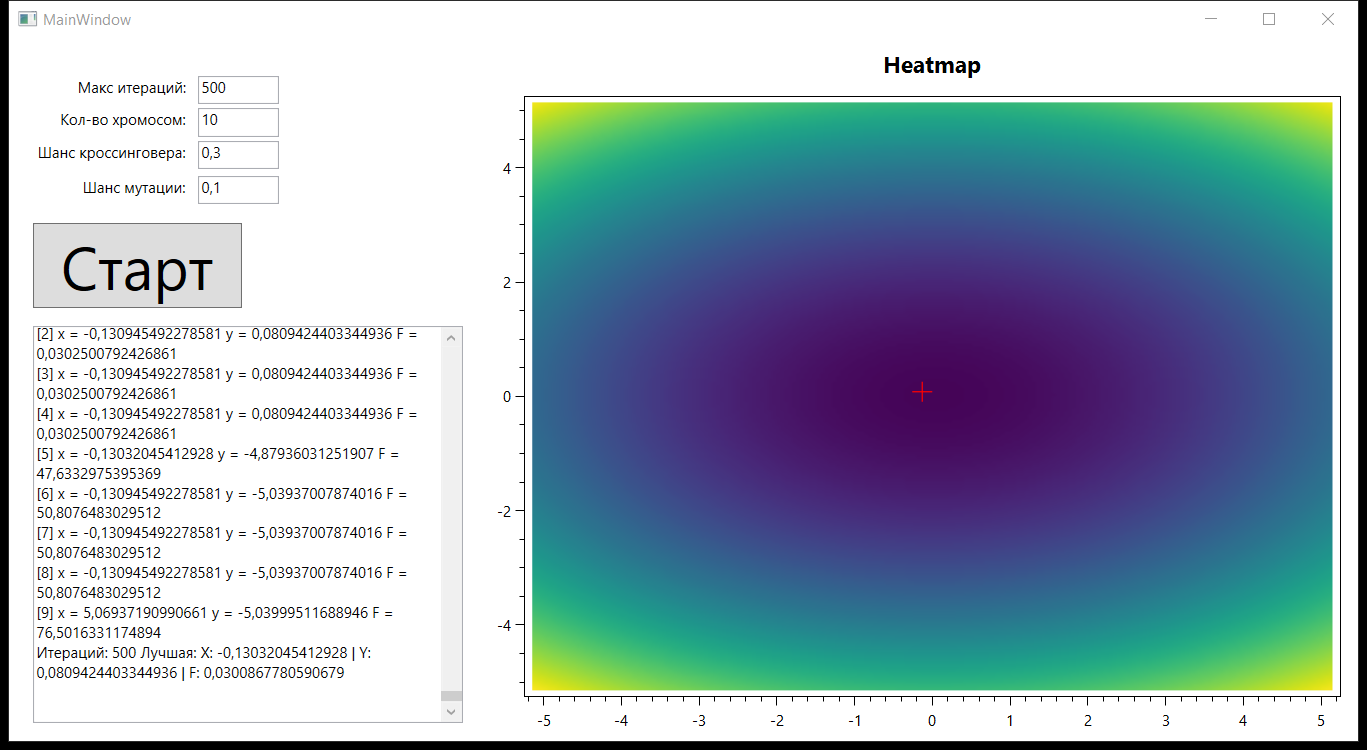
**Этап 1**

Для проверки работы алгоритма и сбора начальных данных, оставим лишь ограничение по количеству циклов.

За исходные данные примем:

1. Ограничение циклов = 500;
2. Кол-во хромосом = 10;
3. Шанс кроссинговера = 0,3;
4. Шанс мутации = 0,1.

Результаты:



Фрагмент текстового отслеживания работы программы:

Исходная популяция

[0] x = 1,07350302142465 y = -3,85867301471037 F = 30,9311236059159

[1] x = -1,75916987120796 y = 3,02174693279619 F = 21,3565876874922

[2] x = 1,36852102789477 y = 4,2693230788012 F = 38,3270889061592

[3] x = -0,0615662577061586 y = -1,92167979002625 F = 7,38949683487858

[4] x = -1,441650491363 y = -0,121569920039065 F = 2,10791463016378

[5] x = -4,99624244643838 y = -2,16231947750717 F = 34,3136896292064

[6] x = 4,85123359580053 y = -1,64291277543795 F = 28,9327921764182

[7] x = -1,31914301409998 y = -1,58665934200085 F = 6,77511402676595

[8] x = -3,31926509186352 y = 2,28607703106879 F = 21,4698171340243

[9] x = -4,35432826710615 y = -4,22619544649942 F = 54,6816305617445

Итерация 1

Репродукция:

[0] x = -3,31926509186352 y = 2,28607703106879 F = 21,4698171340243

[1] x = -1,31914301409998 y = -1,58665934200085 F = 6,77511402676595

[2] x = -3,31926509186352 y = 2,28607703106879 F = 21,4698171340243

[3] x = 1,07350302142465 y = -3,85867301471037 F = 30,9311236059159

[4] x = -3,31926509186352 y = 2,28607703106879 F = 21,4698171340243

[5] x = 1,07350302142465 y = -3,85867301471037 F = 30,9311236059159

[6] x = -3,31926509186352 y = 2,28607703106879 F = 21,4698171340243

[7] x = -4,35432826710615 y = -4,22619544649942 F = 54,6816305617445

[8] x = -1,31914301409998 y = -1,58665934200085 F = 6,77511402676595

[9] x = 1,07350302142465 y = -3,85867301471037 F = 30,9311236059159

0 Кроссинговер с шансом 0,3

Parent 1 x= 00101101000001 -3,31926509186352y= 10111001001001 2,28607703106879

Parent 2 x= 01011111000001 -1,31914301409998y= 01011000010101 -1,58665934200085

K = 2 ,размер = 14

Child 1 x: -3,87929927363731 y: 0,973496917536472

Child 2 x: -0,759108832326191 y: -0,274079228468534

5 Кроссинговер с шансом 0,3

Parent 1 x= 10011010110101 1,07350302142465y= 00011111100010 -3,85867301471037

Parent 2 x= 01101101000001 -0,759108832326191y= 01111001001001 -0,274079228468534

K = 1 ,размер = 14

Child 1 x: 4,36120368674846 y: -0,274079228468534

Child 2 x: -4,04680949765 y: -3,85867301471037

7 Кроссинговер с шансом 0,3

Parent 1 x= 00010011001001 -4,35432826710615y= 00010110010110 -4,22619544649942

Parent 2 x= 01011111000001 -1,31914301409998y= 01011000010101 -1,58665934200085

K = 9 ,размер = 14

Child 1 x: -4,35932857230056 y: -4,22682048464872

Child 2 x: -1,31414270890557 y: -1,58603430385155

сокращённая популяция

[0] x = -1,31414270890557 y = -1,58603430385155 F = 6,75798068535744

[1] x = -3,87929927363731 y = 0,973496917536472 F = 16,944355351349

[2] x = 4,36120368674846 y = -0,274079228468534 F = 19,1703364442642

[3] x = -3,31926509186352 y = 2,28607703106879 F = 21,4698171340243

[4] x = -3,31926509186352 y = 2,28607703106879 F = 21,4698171340243

[5] x = -3,31926509186352 y = 2,28607703106879 F = 21,4698171340243

[6] x = 1,07350302142465 y = -3,85867301471037 F = 30,9311236059159

[7] x = 1,07350302142465 y = -3,85867301471037 F = 30,9311236059159

[8] x = -4,04680949765 y = -3,85867301471037 F = 46,1553819791783

[9] x = -4,35932857230056 y = -4,22682048464872 F = 54,7357684201681

Среднее значение: 27,0035521494222

Минимальное значение: 6,75798068535744

…

Финальный геном

[0] x = -0,13032045412928 y = 0,0809424403344936 F = 0,0300867780590679

[1] x = -0,130945492278581 y = 0,0809424403344936 F = 0,0302500792426861

[2] x = -0,130945492278581 y = 0,0809424403344936 F = 0,0302500792426861

[3] x = -0,130945492278581 y = 0,0809424403344936 F = 0,0302500792426861

[4] x = -0,130945492278581 y = 0,0809424403344936 F = 0,0302500792426861

[5] x = -0,13032045412928 y = -4,87936031251907 F = 47,6332975395369

[6] x = -0,130945492278581 y = -5,03937007874016 F = 50,8076483029512

[7] x = -0,130945492278581 y = -5,03937007874016 F = 50,8076483029512

[8] x = -0,130945492278581 y = -5,03937007874016 F = 50,8076483029512

[9] x = 5,06937190990661 y = -5,03999511688946 F = 76,5016331174894

Итераций: 500 Лучшая: X: -0,13032045412928 | Y: 0,0809424403344936 | F: 0,0300867780590679

Как видно из результата, алгоритм справляется с задачей.

**Эксперименты:**

Для каждой серии экспериментов результаты будут приводиться в виде таблицы, для удобства чтения.

**Количество хромосом:**

Проверим результаты работы с количествами хромосом равными 5, 10 и 15 соответственно.

За исходные данные примем:

Ограничение циклов = 100;

Шанс кроссинговера = 0,3;

Шанс мутации = 0,1.

|  |  |  |
| --- | --- | --- |
| **№** | **Начальная популяция** | **Скриншот** |
| **Конечная** |
| **Результат** |
| 1 | [0] x = -3,74179088079106 y = 1,24351339803455 F = 17,093650137754  [1] x = 0,539720441921505 y = -3,8736739302936 F = 30,3019975919004  [2] x = -3,40552035646707 y = 4,88498565586279 F = 59,3237386142819  [3] x = -3,57553073307697 y = 1,6147860587194 F = 17,999488054047  [4] x = -1,90292864554721 y = -2,42921076725874 F = 15,4232673335758 |  |
| [0] x = -1,01912470243545 y = -1,70416651406946 F = 6,84698217446547  [1] x = -1,25663919916987 y = 2,61172190685467 F = 15,2213247143795  [2] x = -3,29426356589147 y = -2,26357565769395 F = 21,0997219577692  [3] x = -1,09412928035158 y = 3,41677104315449 F = 24,5457676048007  [4] x = -1,09412928035158 y = 3,41677104315449 F = 24,5457676048007 |
| X: -1,01912470243545 | Y: -1,70416651406946 | F: 6,84698217446547 |
| 2 | [0] x = 1,82229872428737 y = 1,17788439235793 F = 6,09559592406021  [1] x = -3,04987364951474 y = 2,75923091008973 F = 24,5284397083935  [2] x = -2,64609900506623 y = -0,98474760422389 F = 8,94129563266186  [3] x = 4,6080937557224 y = 4,47308551547336 F = 61,2515161190028  [4] x = 3,16113044009034 y = 5,03249465909785 F = 60,6447506469624 |  |
| [0] x = -2,47921381920283 y = 0,907242873710554 F = 7,79268042512346  [1] x = 2,64109869987182 y = 0,902242568516145 F = 8,60348564734984  [2] x = 2,64109869987182 y = 0,902242568516145 F = 8,60348564734984  [3] x = 2,64109869987182 y = -4,21806995055851 F = 42,5596305580739  [4] x = 2,64109869987182 y = 4,74247695782213 F = 51,9575777334124 |
| X: -2,47921381920283 | Y: 0,907242873710554 | F: 7,79268042512346 |
| 3 | [0] x = -0,97287187938717 y = -3,8742989684429 F = 30,9668646874577  [1] x = 1,83417444912409 y = -1,56978331196972 F = 8,29263520289694  [2] x = -1,39477263016542 y = 0,28282976255875 F = 2,10537603903663  [3] x = -4,441208569859 y = -1,67291460660441 F = 25,3216201229698  [4] x = -3,98430568271989 y = 3,55552951229934 F = 41,1582719990171 |  |
| [0] x = -0,0453152658243301 y = 0,161572361594336 F = 0,0542647293790716  [1] x = -0,0453152658243301 y = 0,161572361594336 F = 0,0542647293790716  [2] x = -0,0653164866019651 y = 0,162197399743637 F = 0,0568822363892192  [3] x = -2,60547152536166 y = 0,161572361594336 F = 6,84069312553273  [4] x = -0,0653164866019651 y = 2,72172862113166 F = 14,8198796175965 |
| X: -0,0453152658243301 | Y: 0,161572361594336 | F: 0,0542647293790716 |
| 4 | [0] x = -3,17988158456937 y = -4,06931087102484 F = 43,2302288219653  [1] x = 3,23238478911066 y = 0,180323506073369 F = 10,5133445585592  [2] x = -0,653477385094305 y = -4,87123481657816 F = 47,8848899693162  [3] x = 3,98555575901849 y = -1,92355490447415 F = 23,2847816492986  [4] x = 2,99362021607764 y = 1,87917719587377 F = 16,0243758650928 |  |
| [0] x = -4,79185497161692 y = -3,51365195629616 F = 47,6533742089775  [1] x = -4,79185497161692 y = -3,51365195629616 F = 47,6533742089775  [2] x = -4,79185497161692 y = -3,51365195629616 F = 47,6533742089775  [3] x = -4,79185497161692 y = -3,51365195629616 F = 47,6533742089775  [4] x = -4,79185497161692 y = -4,79373008606482 F = 68,9215703450959 |
| X: -4,79185497161692 | Y: -3,51365195629616 | F: 47,6533742089775 |
| 5 | [0] x = 2,39045840200208 y = -3,83742171763413 F = 35,1659022496425  [1] x = 1,15100775193798 y = -2,13169260819142 F = 10,4130455966572  [2] x = 0,46409082585607 y = 4,09743758774339 F = 33,7933698655486  [3] x = -5,1181248855521 y = 0,169072819385949 F = 26,2523735806179  [4] x = -1,53603125190747 y = -0,88224134773851 F = 3,91609159815534 |  |
| [0] x = 0,405962277971068 y = 1,34039431117622 F = 3,7581191900026  [1] x = 0,405962277971068 y = 1,34039431117622 F = 3,7581191900026  [2] x = 0,405962277971068 y = 1,34039431117622 F = 3,7581191900026  [3] x = 0,408462430568273 y = 1,34039431117622 F = 3,76015537605289  [4] x = -4,71435024110358 y = -3,77991820789843 F = 50,8006615125976 |
| X: 0,405962277971068 | Y: 1,34039431117622 | F: 3,7581191900026 |
| 6 | [0] x = -1,90480375999512 y = -1,84542513581151 F = 10,4394652278614  [1] x = -0,857864859915766 y = -5,11562473295489 F = 53,0751649347179  [2] x = 0,0634413721540623 y = 4,86873466398096 F = 47,4131792642003  [3] x = 2,967368613807 y = -0,794110968687053 F = 10,066500951385  [4] x = 1,05600195324422 y = 2,64797411951413 F = 15,1386740004889 |  |
| [0] x = 4,17994262345114 y = -0,661602881035219 F = 18,3473570797318  [1] x = 4,25994750656168 y = -0,661602881035219 F = 19,0225895030493  [2] x = 4,25994750656168 y = -0,661602881035219 F = 19,0225895030493  [3] x = 4,17994262345114 y = -0,981622413477385 F = 19,3990854606259  [4] x = 4,25994750656168 y = -0,981622413477385 F = 20,0743178839434 |
| X: 4,17994262345114 | Y: -0,661602881035219 | F: 18,3473570797318 |
| 7 | [0] x = 1,76354513825307 y = -0,812862113166086 F = 4,4315810846977  [1] x = 1,81979857169017 y = 1,91042910333883 F = 10,6111455592936  [2] x = 0,540345480070806 y = -4,75435268265885 F = 45,4997121000438  [3] x = 2,56609412195569 y = -2,42421046206433 F = 18,3384317714999  [4] x = -2,66860037844107 y = -2,83173533540866 F = 23,1588779994197 |  |
| [0] x = -2,66672526399316 y = -2,83736067875237 F = 23,2126548762796  [1] x = -2,66922541659037 y = -2,83736067875237 F = 23,2259955672322  [2] x = -2,66922541659037 y = -2,83736067875237 F = 23,2259955672322  [3] x = -2,66922541659037 y = -2,83736067875237 F = 23,2259955672322  [4] x = -3,94930354635903 y = -4,11743880852103 F = 49,5036031851141 |
| X: -2,66672526399316 | Y: -2,83736067875237 | F: 23,2126548762796 |
| 8 | [0] x = 0,347833730086065 y = 1,52415552707074 F = 4,76708844518619  [1] x = 3,54490386376122 y = 1,29914179332235 F = 15,9418822016228  [2] x = -0,293455411096868 y = -4,65059634987487 F = 43,342208897241  [3] x = -3,77679301715193 y = -2,89111395959226 F = 30,981245349106  [4] x = -1,60603552462919 y = 3,55677958859794 F = 27,8807121901048 |  |
| [0] x = 0,320332051516816 y = 0,000312519074650908 F = 0,102612818565316  [1] x = 0,320332051516816 y = 0,000312519074650908 F = 0,102612818565316  [2] x = 0,320332051516816 y = 0,000312519074650908 F = 0,102612818565316  [3] x = 0,320332051516816 y = 0,000312519074650908 F = 0,102612818565316  [4] x = -4,79998046755783 y = 2,56046877861198 F = 36,1518132214301 |
| X: 0,320332051516816 | Y: 0,000312519074650908 | F: 0,102612818565316 |
| 9 | [0] x = 2,09169016663615 y = 3,4236464627968 F = 27,8178779576445  [1] x = 1,85542574620033 y = 0,215325642434231 F = 3,53533496424248  [2] x = -2,48358908624794 y = 3,85054751876946 F = 35,8216471379331  [3] x = -3,15300494414942 y = 1,30726728926326 F = 13,3593357089861  [4] x = 4,45620948544223 y = 4,12118903741684 F = 53,8262011423947 |  |
| [0] x = 4,78935481901972 y = -0,32283220411402 F = 23,1463608464936  [1] x = 5,03436977354575 y = 0,637226393212477 F = 26,1569939692042  [2] x = 5,03436977354575 y = 0,637226393212477 F = 26,1569939692042  [3] x = 4,95999023377892 y = 2,55734358786547 F = 37,6815155719755  [4] x = 5,1193749618507 y = 5,1174998474028 F = 78,5856093763591 |
| X: 4,78935481901972 | Y: -0,32283220411402 | F: 23,1463608464936 |
| 10 | [0] x = -0,206575108344015 y = -2,81173411463102 F = 15,8543707381471  [1] x = -0,185323811267777 y = -0,640976622108282 F = 0,856046975201502  [2] x = -3,7605420252701 y = -2,96299334676189 F = 31,7003354697329  [3] x = 1,6697894158579 y = -3,46927424769578 F = 26,8599243047613  [4] x = -2,77610694012086 y = -1,37727156198498 F = 11,5005236538923 |  |
| [0] x = -0,967871574192761 y = -0,000312519074650019 F = 0,936775579466717  [1] x = -0,977872184581578 y = -0,000312519074650019 F = 0,956234204714692  [2] x = -0,967871574192761 y = 5,12 F = 53,3655753841304  [3] x = 4,15244094488189 y = 4,95999023377892 F = 66,4457720390961  [4] x = 4,15244094488189 y = 5,12 F = 69,6715658007316 |
| X: -0,967871574192761 | Y: -0,000312519074650019 | F: 0,936775579466717 |

Как видно из результатов, приведённых в таблице, малое количество хромосом плохо сказывается на финальном результате (среднее значение = 13,185). А сам результат, по большей части, зависит от того, насколько удачными были хромосомы в исходном геноме.

**10 хромосом:**

|  |  |  |
| --- | --- | --- |
| 1 | [0] x = -0,799736312030763 y = -2,52921687114692 F = 13,433454131369  [1] x = 4,42558261612647 y = -1,98355856680706 F = 27,4547906680682  [2] x = -1,12975645486175 y = 0,475341512543491 F = 1,72824875439605  [3] x = -3,54427882561191 y = -4,6649722273088 F = 56,0858441568057  [4] x = -2,83923579320027 y = 0,427213575047306 F = 8,42628276679896  [5] x = 4,75122749191235 y = 3,12737838002808 F = 42,1351537436378  [6] x = -3,91180125740096 y = 1,08475370811207 F = 17,6555702919295  [7] x = 0,434714032838919 y = -2,19669657571873 F = 9,83992798189589  [8] x = -0,544720747115912 y = 3,083625709577 F = 19,314215725867  [9] x = -4,79873039125923 y = -1,52478056522004 F = 27,6777249121405 |  |
| [0] x = -0,00156259537325276 y = 0,000937557223951835 F = 4,19973139687948E-06  [1] x = -0,00156259537325276 y = 0,000937557223951835 F = 4,19973139687948E-06  [2] x = -0,00156259537325276 y = 0,000937557223951835 F = 4,19973139687948E-06  [3] x = -0,00218763352255369 y = 0,000312519074650908 F = 4,98107677304198E-06  [4] x = -0,00218763352255369 y = 0,000312519074650908 F = 4,98107677304198E-06  [5] x = -0,00156259537325276 y = 0,00218763352255458 F = 1,201318515852E-05  [6] x = -0,161572361594335 y = 0,000937557223951835 F = 0,026107386058267  [7] x = -2,56234389305988 y = 0,000312519074650908 F = 6,5656064216376  [8] x = -0,00218763352255369 y = 1,92042971372765 F = 7,37610535647652  [9] x = -0,00156259537325276 y = 1,92105475187695 F = 7,38090516112229 |
| X: -0,00156259537325276 | Y: 0,000937557223951835 | F: 4,19973139687948E-06 |
| 2 | Лучшая: X: 0,340333272294452 | Y: -0,00531282426905921 | F: 0,115883188434077 |  |
| 3 | Лучшая: X: 2,73672953671489 | Y: 0,752858450833181 | F: 8,62328025110956 |  |
| 4 | Лучшая: X: 2,07981444179943 | Y: 0,415962888359886 | F: 4,67167836130286 |  |
| 5 | Лучшая: X: 0,105318928157237 | Y: 0,0884428981261065 | F: 0,0267363690860789 |  |
| 6 | X: -3,31238967222121 | Y: -2,93361655374474 | F: 28,184137509448 |  |
| 7 | Лучшая: X: -1,68479033144113 | Y: -3,35114203747787 | F: 25,2988243716202 |  |
| 8 | Лучшая: X: 0,325332356711225 | Y: -0,0384398461820172 | F: 0,108796385872274 |  |
| 9 | Лучшая: X: 0,868490508453886 | Y: -1,78979674052371 | F: 7,16102050805311 |  |
| 10 | Лучшая: X: 0,025939083195996 | Y: 0,0103131294634684 | F: 0,000885557315709324 |  |

Исходя из данных приведённых в таблице видно, что общий результат улучшился (среднее значение = 7,429)

Вывод: увеличение кол-ва хромосом положительно влияет на результат.

**Увеличение шанса кроссинговера:**

Для эксперимента увеличим шанс кроссинговера для 5 и 10 хромосом

5 хромосом

|  |  |  |
| --- | --- | --- |
| 1 | X: 3,83992187023134 | Y: -3,19988280534701 | F: 35,2234999053918 |  |
| 2 | X: -1,11038027223341 | Y: -1,44665079655741 | F: 5,41854140332551 |  |
| 3 | X: -0,00468778611975829 | Y: 0,0103131294634684 | F: 0,000234696617365119 |  |
| 4 | X: 1,8391747543185 | Y: -2,28045168772508 | F: 13,7834835770189 |  |
| 5 | X: 4,06993590917414 | Y: 0,648477079899896 | F: 17,4054233510962 |  |
| 6 | X: 1,28289080144052 | Y: -0,170322895684551 | F: 1,70382858600944 |  |
| 7 | X: -5,08999816883355 | Y: -2,39733382164439 | F: 37,402500263529 |  |
| 8 | X: 1,62041140206312 | Y: 0,320957089666118 | F: 2,83176001875004 |  |
| 9 | X: -2,45233717878288 | Y: 2,54734297747665 | F: 18,99187012824 |  |
| 10 | X: 0,360959531221388 | Y: -1,76854544344748 | F: 6,38579775425722 |  |

Среднее значение: 11,112. Вывод, при малом количестве хромосом, увеличение шанса кроссинговера практически не влияет на результат.

10 кромосом:

|  |  |  |
| --- | --- | --- |
| 1 | X: -1,28789110663493 | Y: -0,045940303973631 | F: 1,66288452560771 |  |
| 2 | X: -0,250952816944393 | Y: -0,00218763352255369 | F: 0,062986887813184 |  |
| 3 | X: 0,609724714643228 | Y: 1,03975096136239 | F: 2,53392835095479 |  |
| 4 | X: -2,35545626564121 | Y: 2,47421351400842 | F: 17,7916392451523 |  |
| 5 | X: 2,03731184764695 | Y: -0,445339681377037 | F: 4,54729442818064 |  |
| 6 | X: -0,0846926692302992 | Y: 0,0209387780015877 | F: 0,00804971306975241 |  |
| 7 | X: 0,0403149606299218 | Y: -0,000312519074650019 | F: 0,00162549138693618 |  |
| 8 | X: 0,168447781236648 | Y: 0,638476469511079 | F: 0,843679059242213 |  |
| 9 | X: 0,000312519074650908 | Y: 0,0203137398522859 | F: 0,000825393721744718 |  |
| 10 | X: 2,82173472501984 | Y: 0,294705487395471 | F: 8,13588950698479 |  |

Среднее значение: 7,11.

Вывод: увеличение шанса кроссинговера при малом и среднем количестве ботов – ненамного улучшает общий результат.

Подводя итоги можно сказать, что наиболее положительно влияющим на результат параметром является количество особей в геноме.

**Листинг программного кода:**

Класс формы:

namespace gen\_2

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

private PlotModel model;

private int sizeMap = 200;

public double startPoint = -5.12, endPoint = 5.12;

private ScatterSeries point = new ScatterSeries();

GeneticAlgorithm geneticAlgorithm = new GeneticAlgorithm();

public MainWindow()

{

InitializeComponent();

initHeatMap();

point.ColorAxisKey = "ColorAxis";

point.MarkerSize = 8;

point.MarkerType = MarkerType.Plus;

}

private void initHeatMap()

{

model = new PlotModel { Title = "Heatmap" };

model.Axes.Add(new LinearColorAxis { Palette = OxyPalettes.Viridis(255) });

var data = new double[sizeMap, sizeMap];

double xValue = startPoint, yValue = startPoint, max = endPoint;

double det = (Math.Abs(xValue) + max) / sizeMap;

for (int y = 0; y < sizeMap; ++y)

{

xValue = startPoint;

for (int x = 0; x < sizeMap; ++x)

{

data[x, y] = getFunction(xValue, yValue);

xValue += det;

}

yValue += det;

}

var heatMapSeries = new HeatMapSeries

{

X0 = startPoint,

X1 = endPoint,

Y0 = startPoint,

Y1 = endPoint,

Interpolate = true,

RenderMethod = HeatMapRenderMethod.Bitmap,

Data = data

};

var axis1 = new LinearColorAxis();

axis1.Key = "ColorAxis";

axis1.IsAxisVisible = true;

model.Axes.Add(axis1);

axis1.Palette.Colors.Clear();

axis1.Palette.Colors.Add(OxyColor.FromArgb((byte)255, 255, 0, 0));

model.Series.Add(heatMapSeries);

testPlotView.Model = model;

}

private void Button\_Click\_2(object sender, RoutedEventArgs e)

{

int kolChromosome = Convert.ToInt32(textBlock2.Text);

int kolIteration = Convert.ToInt32(textBlock1.Text);

double VerMyta = Convert.ToDouble(textBlock4.Text);

double VerCrosengove = Convert.ToDouble(textBlock3.Text);

Chromosome components = geneticAlgorithm.Evolution(kolChromosome, kolIteration, VerCrosengove, VerMyta);

double x = components.xdecVal;

double y = components.ydecVal;//FitnessFunc.Func(x);//components.funcVal;

initHeatMap();

var random = new Random();

textBox1.Clear();

textBox1.AppendText(geneticAlgorithm.TextAll + Environment.NewLine);

//textBox1.AppendText(" | " + x + " | " + y + Environment.NewLine);

while (model.Series.Count > 1)

{

model.Series.Remove(model.Series.Last());

}

//point.Points.Clear();

//point.Points.Add(new ScatterPoint(x, y, double.NaN, 1));

model.Series.Add(components.Point);

//testPlotView.Model = model;

testPlotView.Model.InvalidatePlot(true);

}

}

Класс Геном:

using OxyPlot;

using OxyPlot.Series;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace gen\_2

{

class Chromosome

{

public string xbinVal;

public string ybinVal;

public double xdecVal;

public double ydecVal;

public double funcVal;

private ScatterSeries point;

public Chromosome(string xbin, string ybin)

{

xbinVal = xbin;

ybinVal = ybin;

}

public Chromosome(string xbin, double xdec, string ybin, double ydec, double fun)

{

xbinVal = xbin;

xdecVal = xdec;

ybinVal = ybin;

ydecVal = ydec;

funcVal = fun;

}

public ScatterSeries Point

{

get

{

this.point = new ScatterSeries();

this.point.ColorAxisKey = "ColorAxis";

this.point.MarkerSize = 8;

this.point.MarkerType = MarkerType.Plus;

point.Points.Add(new ScatterPoint(xdecVal, ydecVal, double.NaN, 1));

return point;

}

set => point = value;

}

public void ReCreate(string xbin, string ybin)

{

xbinVal = xbin;

ybinVal = ybin;

xdecVal = 0;

ydecVal = 0;

funcVal = 0;

}

}

class FitnessFunc

{

public static readonly double minInt = -5.12;

public static readonly double maxInt = 5.12;

public static double Func(double x, double y)

{

return (x \* x) + (2 \* y \* y);

}

}

class GeneticAlgorithm

{

private Random random = new Random();

private StringBuilder textAll = new StringBuilder();

public string TextAll { get { return textAll.ToString(); } }

public GeneticAlgorithm() { }

public Chromosome Evolution(int chromosomesCount, int iterationsCount, double crossingoverChance, double mutationChance)

{

textAll.Clear();

double minInterval = FitnessFunc.minInt;

double maxInterval = FitnessFunc.maxInt;

int intervalsCount = (int)(maxInterval - minInterval) \* 1000;

int chromosomesSize = getChromosomeSize(intervalsCount);

int currentCount = 0;

List<Chromosome> genome = CreateGenome(chromosomesSize, chromosomesSize, chromosomesCount);

foreach (Chromosome c in genome)

{

c.xdecVal = getChromosomeValue(c.xbinVal, minInterval, maxInterval);

c.ydecVal = getChromosomeValue(c.ybinVal, minInterval, maxInterval);

c.funcVal = FitnessFunc.Func(c.xdecVal, c.ydecVal);

}

textAll.Append("Исходная популяция" + Environment.NewLine);

printGenome(genome);

Chromosome best = genome.Find(g => g.funcVal == genome.Min(v => v.funcVal));

while (best.funcVal > 0 && currentCount < iterationsCount)

{

textAll.Append("Итерация " + (currentCount + 1) + Environment.NewLine);

Reproduction(genome, minInterval, maxInterval, chromosomesCount);

printGenome(genome);

foreach (Chromosome chromosome in genome)

{

if (random.NextDouble() <= crossingoverChance)

{

textAll.Append(genome.IndexOf(chromosome) + " Кроссинговер с шансом " + crossingoverChance + Environment.NewLine);

Crossingover(genome, chromosome, minInterval, maxInterval);

}

string xbinary = chromosome.xbinVal;

string ybinary = chromosome.ybinVal;

if (random.NextDouble() <= mutationChance)

{

textAll.Append(" Мутация с шансом " + mutationChance + Environment.NewLine);

textAll.Append("X: " + chromosome.xbinVal + " " + chromosome.xdecVal + Environment.NewLine);

textAll.Append("Y: " + chromosome.ybinVal + " " + chromosome.ydecVal + Environment.NewLine);

Chromosome chr = Mutation(chromosome);

chromosome.xbinVal = chr.xbinVal;

chromosome.ybinVal = chr.ybinVal;

chromosome.xdecVal = getChromosomeValue(chromosome.xbinVal, minInterval, maxInterval);

chromosome.ydecVal = getChromosomeValue(chromosome.ybinVal, minInterval, maxInterval);

chromosome.funcVal = FitnessFunc.Func(chromosome.xdecVal, chromosome.ydecVal);

textAll.Append("newX: " + chromosome.xbinVal + " " + chromosome.xdecVal + Environment.NewLine);

textAll.Append("newY: " + chromosome.ybinVal + " " + chromosome.ydecVal + Environment.NewLine);

}

}

Chromosome cr;

for (int i = 0; i < genome.Count; i++)

{

for (int j = 0; j < genome.Count; j++)

{

if (genome[i].funcVal < genome[j].funcVal)

{

cr = genome[i];

genome[i] = genome[j];

genome[j] = cr;

}

}

}

if (genome.Count > chromosomesCount)

{

for (int i = genome.Count - 1; i > chromosomesCount - 1; i--)

{

genome.RemoveAt(i);

}

}

textAll.Append("сокращённая популяция" + Environment.NewLine);

printGenome(genome);

textAll.Append("Среднее значение: " + genome.Average(g => g.funcVal) + Environment.NewLine + "Минимальное значение: " + genome.Min(g => g.funcVal) + Environment.NewLine);

currentCount++;

best = genome.Find(g => g.funcVal == genome.Min(v => v.funcVal));

int abc = 0;

}

textAll.Append("Финальный геном" + Environment.NewLine);

printGenome(genome);

//Chromosome best = genome.Find(g => g.funcVal == genome.Min(v => v.funcVal));

textAll.Append("Итераций: " + currentCount + " Лучшая: " + "X: " + best.xdecVal + " | " + "Y: " + best.ydecVal + " | F: " + best.funcVal + Environment.NewLine);

//textAll.Append("Лучшая: " + best.binVal + " | " + nom1 + ran + " | " + nom2+ran);

return best;

}

private Chromosome Mutation(Chromosome chromosome)

{

StringBuilder newChromosomex = new StringBuilder(chromosome.xbinVal);

StringBuilder newChromosomey = new StringBuilder(chromosome.ybinVal);

int i = random.Next(chromosome.xbinVal.Count());

newChromosomex[i] = (newChromosomex[i] == '0') ? '1' : '0';

i = random.Next(chromosome.ybinVal.Count());

newChromosomey[i] = (newChromosomey[i] == '0') ? '1' : '0';

chromosome.xbinVal = newChromosomex.ToString();

chromosome.ybinVal = newChromosomey.ToString();

return chromosome;

}

private void Reproduction(List<Chromosome> genome, double minInterval, double maxInterval, int chromosomesCount)

{

double fValuesSum = 0, a = 0, b = 0;

double sum;

List<Chromosome> newGenome = new List<Chromosome>();

textAll.Append("Репродукция:" + Environment.NewLine);

Chromosome cr;

//for (int i = 0; i < genome.Count; i++)

//{

// for (int j = 0; j < genome.Count; j++)

// {

// if (genome[i].funcVal < genome[j].funcVal)

// {

// cr = genome[i];

// genome[i] = genome[j];

// genome[j] = cr;

// }

// }

//}

Random rnd = new Random();

double[] mass22 = new double[genome.Count];

int[] mass11 = new int[3];

int[] rez = new int[genome.Count];

for (int i = 0; i < genome.Count; i++)

{

for (int j = 0; j < mass11.Length; j++)

{

mass11[j] = rnd.Next(0, genome.Count);

}

mass22[i] = genome[mass11[0]].funcVal;

rez[i] = mass11[0];

for (int j = 1; j < mass11.Length; j++)

{

if (mass22[i] > mass22[mass11[j]])

{

mass22[j] = mass22[mass11[j]];

rez[i] = mass11[j];

}

}

}

for (int i = 0; i < genome.Count; i++)

{

{

newGenome.Add(new Chromosome(genome[rez[i]].xbinVal, genome[rez[i]].xdecVal, genome[rez[i]].ybinVal, genome[rez[i]].ydecVal, genome[rez[i]].funcVal));

}

}

genome.Clear();

genome.AddRange(newGenome);

}

private void Crossingover(List<Chromosome> genome, Chromosome parent1, double minInterval, double maxInterval)

{

Chromosome parent2 = getParents(genome, parent1);

if (parent2 != null)

{

textAll.Append("Parent 1 " + "x= " + parent1.xbinVal + " " + parent1.xdecVal + "y= " + parent1.ybinVal + " " + parent1.ydecVal + Environment.NewLine);

textAll.Append("Parent 2 " + "x= " + parent2.xbinVal + " " + parent2.xdecVal + "y= " + parent2.ybinVal + " " + parent2.ydecVal + Environment.NewLine);

GetChildBySinglePointCrossingover(parent1, parent2, minInterval, maxInterval);

}

}

private string CreateBinaryModel(int size)

{

StringBuilder value = new StringBuilder();

for (int i = 0; i < size; i++)

value.Append(random.NextDouble() <= 0.5 ? '1' : '0');

return value.ToString();

}

private List<Chromosome> CreateGenome(int sizex, int sizey, int count)

{

List<Chromosome> genome = new List<Chromosome>();

for (int i = 0; i < count; i++)

genome.Add(new Chromosome(CreateBinaryModel(sizex), CreateBinaryModel(sizey)));

return genome;

}

private Chromosome getParents(List<Chromosome> genome, Chromosome parent1)

{

//bool contains = false;

//foreach (Chromosome c in genome)

// if (c.Equals(parent1))

// {

// contains = true;

// break;

// }

//if (contains)

{

Chromosome parent2;

int[] mass = new int[genome.Count];

int count = 0;

double tmp1, tmp2;

bool b = false;

tmp1 = Math.Abs(parent1.funcVal - genome[0].funcVal);

for (int i = 1; i < genome.Count; i++)

{

parent2 = genome[i];

tmp2 = Math.Abs(parent1.funcVal - parent2.funcVal);

if (tmp2 > 0 && tmp1 < tmp2)

{

b = true;

tmp1 = parent1.funcVal - parent2.funcVal;//parent2.funcVal;

count = i;

}

}

if (b)

{

return genome[count];

}

else return genome[0];//null;

//Chromosome parent2;

//int[] mass = new int[genome.Count];

//int count = 100, count2 = -1, tmp;

//for (int i = 0; i < genome.Count; i++)

//{

// parent2 = genome[i];

// tmp = checkbin(parent1.binVal, parent2.binVal);

// if (count > tmp && tmp > 0)

// {

// count = tmp;

// count2 = i;

// }

// //mass[i] = checkbin(parent1.binVal, parent2.binVal);

//}

////do {

//// parent2 = genome[random.Next(genome.Count)];

////}

////while (!checkbin(parent1.binVal, parent2.binVal));

//if (count2 > 0 && count <= blrod)

// return genome[count2];

//else return null;

}

//else

// return null;

}

private int checkbin(string str1, string str2)

{

int count = 0;

for (int i = 0; i < str1.Length; i++)

{

if (str1[i] != str2[i])

count++;

}

return count;

}

private void GetChildBySinglePointCrossingover(Chromosome parent1, Chromosome parent2, double minInterval, double maxInterval)

{

int k = random.Next(parent1.xbinVal.Length);

textAll.Append("K = " + k + " ,размер = " + parent1.xbinVal.Count() + Environment.NewLine);

StringBuilder xchild1 = new StringBuilder();

StringBuilder xchild2 = new StringBuilder();

StringBuilder ychild1 = new StringBuilder();

StringBuilder ychild2 = new StringBuilder();

xchild1.Append(parent1.xbinVal.Substring(0, k));

xchild1.Append(parent2.xbinVal.Substring(k, parent2.xbinVal.Length - k));

xchild2.Append(parent2.xbinVal.Substring(0, k));

xchild2.Append(parent1.xbinVal.Substring(k, parent1.xbinVal.Length - k));

ychild1.Append(parent1.ybinVal.Substring(0, k));

ychild1.Append(parent2.ybinVal.Substring(k, parent2.ybinVal.Length - k));

ychild2.Append(parent2.ybinVal.Substring(0, k));

ychild2.Append(parent1.ybinVal.Substring(k, parent1.ybinVal.Length - k));

//textAll.Append("Child 1 " + parent1.binVal.Substring(0, k) + "\_" + parent2.binVal.Substring(k, parent2.binVal.Length - k) + Environment.NewLine);

//textAll.Append("Child 2 " + parent2.binVal.Substring(0, k) + "\_" + parent1.binVal.Substring(k, parent1.binVal.Length - k) + Environment.NewLine);

parent1.ReCreate(xchild1.ToString(), ychild1.ToString());

parent2.ReCreate(xchild2.ToString(), ychild2.ToString());

parent1.xdecVal = getChromosomeValue(xchild1.ToString(), minInterval, maxInterval);

parent1.ydecVal = getChromosomeValue(ychild1.ToString(), minInterval, maxInterval);

parent2.xdecVal = getChromosomeValue(xchild2.ToString(), minInterval, maxInterval);

parent2.ydecVal = getChromosomeValue(ychild2.ToString(), minInterval, maxInterval);

textAll.Append("Child 1 x: " + parent1.xdecVal + " y: " + parent1.ydecVal + Environment.NewLine);

textAll.Append("Child 2 x: " + parent2.xdecVal + " y: " + parent2.ydecVal + Environment.NewLine);

parent1.funcVal = FitnessFunc.Func(parent1.xdecVal, parent1.ydecVal);

parent2.funcVal = FitnessFunc.Func(parent2.xdecVal, parent2.ydecVal);

}

public void printGenome(List<Chromosome> genome)

{

int i = 0;

foreach (Chromosome c in genome)

{

textAll.Append("[" + i++ + "] " + "x = " + c.xdecVal + " y = " + c.ydecVal + " F = " + c.funcVal + Environment.NewLine);

}

}

public static int getChromosomeSize(int num)

{

int size = 0;

while (num > 0)

{

num >>= 1;

size++;

}

return size;

}

public static double getChromosomeValue(string s, double minInterval, double maxInterval)

{

int sum = 0;

double c, znach = 0;

for (int i = 0; i < s.Count(); i++)

{

c = char.GetNumericValue(s[i]);

znach = Math.Pow(2, s.Count() - i - 1);

sum += (int)(c \* znach);

}

//sum = sum;

return minInterval + sum \* ((maxInterval - minInterval) / (Math.Pow(2, s.Count()) - 1));

}

}

}