Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

ЛАБОРАТОРНАЯ РАБОТА №3
по СИСТЕМАМ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

Студент Пехова А. А.

Группа ПИ-19-1

Руководитель

Доцент Кургасов В. В.

Липецк 2022 г.

Задание кафедры

Разработать программу реализующую генетический алгоритм поиска максимального и минимального значений целевой функции f(x) = a + bx + cx2 + dx3 в интервале x є [-10, 53]. Язык реализации – на выбор студента.

Вариант индивидуального задания - № варианта по журналу группы (номер по списку группы уточнить у преподавателя).

№ варианта	a	b	c	d
16	7	-45	-63	1

Параметры алгоритма:

- размер популяции - 6 особей;

- число скрещиваний в популяции – 3;

- число мутаций – 2 потомка на поколение.

Цель работы

Освоение методов эволюционных вычислений на примере генетического алгоритма.

Ход работы

1. Расчет точек экстремума целевой функции), выполненный с помощью математического анализа.

y=7-45*x-63*x^2+x^3, - целевая функция.

Находим первую производную функции:

y' = 3·x^2-126·x-45

Приравниваем ее к нулю:

3·x2-126·x-45 = 0

x1 = -0.354

x2 = 42.354

Вычисляем значения функции на концах интервала

f(-0.354) = 14.991

f(42.354) = -38935

f(-10) = -6843

f(53) = -30468

Ответ:

fmin = -38935, fmax = 14.991

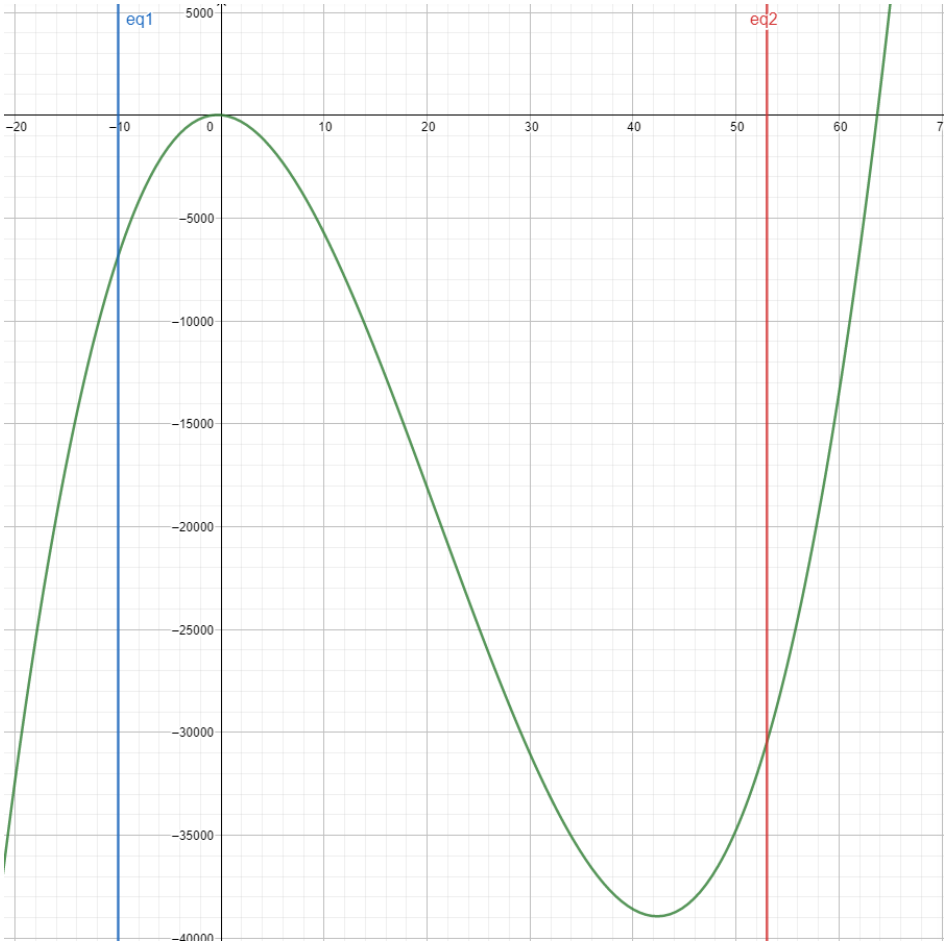

Рисунок 1 – График функции на отрезке x[-10 ; 53]

2. Расчет точек экстремума с помощью реализации генетического алгоритма

Вывод:

Max searching: Iter 0

[x= 37, y= -37252;x= -4, y= -885;x= 41, y= -38820;x= 3, y= -668;x= 0, y= 7;x= 31, y= -32140;]

Min searching: Iter 0

[x= 37, y= -37252;x= -4, y= -885;x= 41, y= -38820;x= 3, y= -668;x= 0, y= 7;x= 31, y= -32140;]

Max searching: Iter 1

[x= 3, y= -668;x= 0, y= 7;x= -8, y= -4177;x= 4, y= -1117;x= 0, y= 7;x= -5, y= -1468;]

Min searching: Iter 1

[x= 31, y= -32140;x= 29, y= -29892;x= 41, y= -38820;x= 25, y= -24868;x= 41, y= -38820;x= 47, y= -37452;]

result:

min population:

[x= 32, y= -33177;x= 32, y= -33177;x= 32, y= -33177;x= 32, y= -33177;x= 24, y= -23537;x= 48, y= -36713;]

особь с минимальным результатом: 42 y=-38927

max population:

[x= 6, y= -2315;x= -9, y= -5420;x= -10, y= -6843;x= -10, y= -6843;x= -10, y= -6843;x= -10, y= -6843;]

особь с максимальным результатом: x=0 y=7

Текст программы:

```
<?php
$population = array();
$x0 = -10;
$x1 = 53;
$population_count = 6;
$crossover_count = 3;
$mutations_count = 2;
function func($x)
{  //x[-10;53]
   return 7 - 45 * $x - 63 * $x ** 2 + $x ** 3;}
function mutation($crossed, $bit_amount = 6){
   $mutated = random_int(0, count($crossed) - 1);
   $genetic_num = random_int(0, $bit_amount - 1);
   $crossed[$mutated][$genetic_num] = $crossed[$mutated][$genetic_num] === '0' ? '1' : '0';
   return $crossed;}
function to_binary($number, $bit_amount = 6){
   $binary = decbin($number);
   if ($number >= 0) {
      if (strlen($binary) < $bit_amount) {
         $to_add = $bit_amount - strlen($binary);
         $binary = str_repeat("0", $to_add) . $binary;  }  } else {
      $binary = substr($binary, strlen($binary) - 6); }
   return $binary;}
function to_dec($binary){
   foreach ($binary as &$bin) {
      $bin = bindec($bin);  }
   return $binary;}
function crossover($parents, $bit_amount = 6){
   $tmp = array();
   for ($i = 0; $i < count($parents); $i++) {
      $next = $i + 1;
      if ($next >= count($parents)) {
         $next = 0;  }
      $crossing = random_int(1, $bit_amount - 1);
      $tmp[] = substr($parents[$i], 0, $crossing) . substr($parents[$next], $crossing); }
   $parents = array_merge($parents, $tmp);
   return $parents;}
function sort_yabs($population, $func_res){
   for ($i = 0; $i < count($func_res); $i++) {
      for ($j = 0; $j < count($func_res); $j++) {
         if ($i != $j) {
            if ($func_res[$j] > $func_res[$i]) {
               $tmp = $func_res[$i];
```

```
                $func_res[$i] = $func_res[$j];
                $func_res[$j] = $tmp;
                $tmp = $population[$i];
                $population[$i] = $population[$j];
                $population[$j] = $tmp;  }} } }
    return $population;};
function get_population($population, $x0 = -10){
    echo "[ ";
    foreach ($population as $dot) {
        echo "x= " . $dot + $x0 . ", ";
        echo "y= " . func($dot + $x0) . ";";   }
    echo "] ";}
function genetic($population, $population_count, $func_res1, $func_res2, $mutations_count){
    $population_best = $population;
    $population_worst = $population;
    for ($i = 0; $i < 100; $i++) {
        if ($i == 0 || $i == 1) {
            echo "Max searching: ";
            echo "Iter " . $i . "\n";
            get_population($population_best);
            echo "Min searching: ";
            echo "Iter " . $i . "\n";
            get_population($population_worst);      }
        array_multisort($func_res1, $population_best);
        array_multisort($func_res2, $population_worst);
        $population_worst = array_slice($population_worst, 0, $population_count / 2);
        $population_best = array_slice($population_best, $population_count / 2, $population_count);
        for ($j = 0; $j < $population_count / 2; $j++) {
            $population_best[$j] = to_binary($population_best[$j]);
            $population_worst[$j] = to_binary($population_worst[$j]);    }
        $population_best = crossover($population_best);
        $population_worst = crossover($population_worst);
        for ($j = 0; $j < $mutations_count; $j++) {
            $population_best = mutation($population_best);
            $population_worst = mutation($population_worst);    }
        $population_best = to_dec($population_best);
        $population_worst = to_dec($population_worst);
        for ($j = 0; $j < $population_count; $j++) {
            $func_res1[$j] = func($population_best[$j]);
            $func_res2[$j] = func($population_worst[$j]);   } }
    array_multisort($func_res1, $population_best);
    array_multisort($func_res2, $population_worst);
    return [
        'population_best' => $population_best,
```

```
      'population_worst' => $population_worst,   ]}
for ($i = 0; $i < $population_count; $i++) {
   $population[$i] = random_int($x0, $x1) - $x0;
   $func_res[$i] = func($population[$i]);}
$func_res1 = $func_res;
$func_res2 = $func_res;
$population = genetic($population, $population_count, $func_res1, $func_res2, $mutations_count);
echo "result: \n";
echo "min population: \n";
get_population($population['population_worst']);
echo "особь с минимальным результатом: " . $population['population_worst'][0] . "   y=" .
func($population['population_worst'][0]) . "\n";
echo "max population: \n";
get_population($population['population_best']);
echo "особь с максимальным результатом: x=" . $population['population_best'][$population_count - 1] . "   y=" .
func($population['population_best'][$population_count - 1]) . "\n";
```

Вывод

В ходе выполнения лабораторной работы получила навыки построения модели знаний.