|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **НУЛП, САПР, СПК** | | **Тема** | **Оцінка:** | **Підпис:** |
| КНСП-11 | 5 | Запрограмувати ГА для задачі комівояжера |  |  |
| Янчук Н. Ю. | |
| Варіант 10 | |
| Методи нечіткої логіки та  еволюційні алгоритми | | **Викладач:**  Кривий Р. З. | |

**Мета:** запрограмувати ГА для задачі комівояжера використовуючи заданий метод селекції.

**Теоретичні відомості**

Селекція - це вибір тих хромосом, які будуть брати участь в створенні нащадків для наступної популяції, тобто для чергового покоління. Такий вибір проводиться відповідно до принципу природного відбору, за яким найбільші шанси на участь в створенні нових особин мають хромосоми з найбільшими значеннями функції пристосованості. Існують різні методи селекції.

Селекція методом усічення - використовується при великій кількості осіб, на першому етапі обирають осіб з найкращими фітнес функціями. На другому - серед вибраних осіб проводять випадковий відбір, в кожної особи рівні шанси.

**Завдання**

(Варіант 10)

У вас є безліч міст (представлені у вигляді точок на площині з X і Y координати). Мета полягає в тому, щоб знайти найкоротший маршрут, який відвідує кожне місто рівно один раз, повертаючись в кінці до своєї відправної точки.

Дано від 10 до 50 точок. Метод селекції — усічення селекція.

**Хід роботи**

Координати точок(міст) читаються з файлу \*.csv, де першим значенням є назва міста, другим — координата Х, а третім — координата Y.

Для виконання завдання була використана функція ga пакету MatLab. Окремо були реалізовані функції для генерації початкової вибірки, мутації та схрещування. Функція для відбору методом усіченя є стандартною в пакеті MatLab.

|  |
| --- |
| Функція для оцінки шляху |
| function [output\_args] = FitnessFcn( input\_args )  %% Цільова функція. Довжина пройденого шляху  % input\_args = [x1, x2, x3, ... ]  % x1, x2 ... - інденкси міст, в порядку їх проходження  global WAY\_MATRIX;  sum = 0;  way\_length = length(input\_args);  for i = 1:1:way\_length-1  sum = sum + WAY\_MATRIX(input\_args(i), input\_args(i+1));  end  % Додаємо шлях до повернення в початкове місто  sum = sum + WAY\_MATRIX(input\_args(way\_length), input\_args(1));  output\_args = sum;  end |

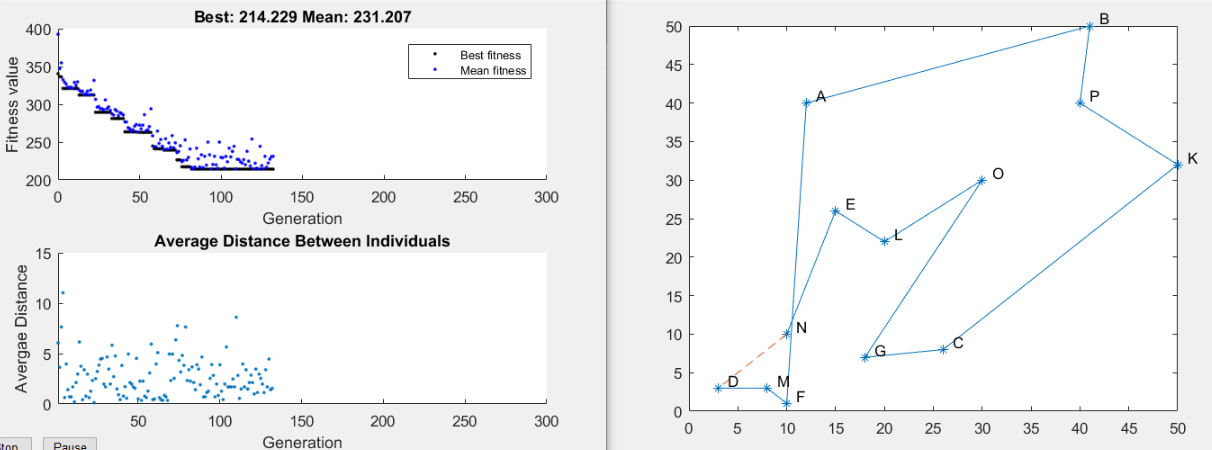
|  |
| --- |
| Функція для генерації початкової вибірки |
| function Population = CreationFcn( GenomeLength, FitnessFcn, options )  %% Функція для створення початкової популяції негомологічих особин  ret = zeros(options.PopulationSize, GenomeLength);    for i = 1:1:options.PopulationSize  vars = 1:1:GenomeLength;  for j = 1:1:GenomeLength  t = randi(length(vars));  ret(i,j) = vars(t);  vars(t) = [];  end;  end;    Population = ret;    end |

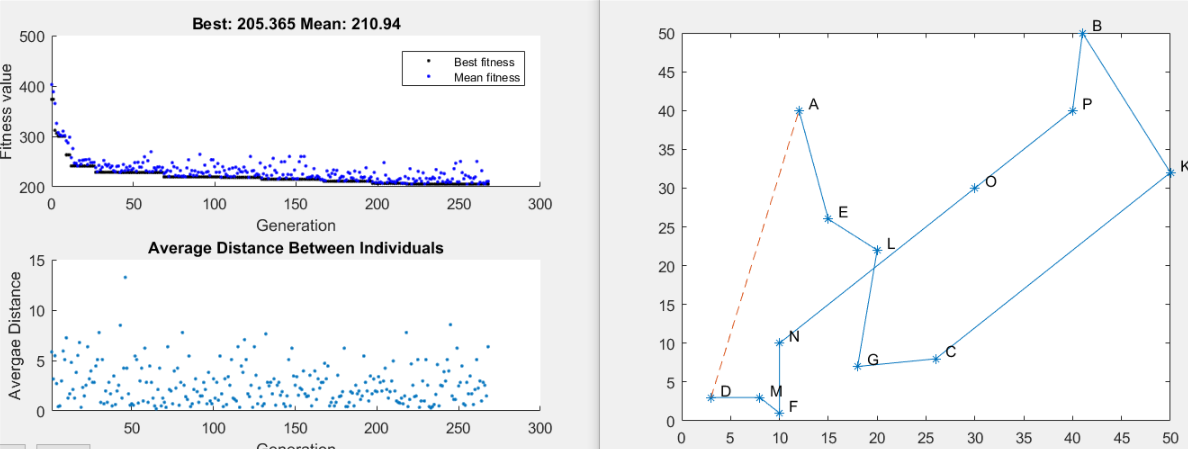
|  |
| --- |
| Точка входу в програму |
| function main()  [names, x, y] = textread('cities.csv', '%s %d %d', 'delimiter', ',');  global WAY\_MATRIX;  WAY\_MATRIX = getWayMatrix(x, y, 1000);  population\_size = ceil(length(x)^(1/2)); % до більшого цілого  nvars = length(x);    global RET;  RET = struct('generation', 0, 'population', struct, 'fvals', struct);    options = gaoptimset(...  'CreationFcn', @CreationFcn, ...  'PopulationSize', population\_size, ...  'MutationFcn', @MutationFcn, ...  'CrossoverFcn', @CrossoverFcn, ...  'OutputFcns', {@OutputFcn}, ...  'SelectionFcn', @selectiontournament, ...  'PlotFcns', {@gaplotbestf, @gaplotdistance} ...  );    [xval,fval,exitflag,output,population,scores] = ga(@FitnessFcn, nvars, options);  % графічне представлення шляху  XX = x; YY = y; NNames = names;  for i = 1:1:nvars  XX(i) = x(xval(i));  YY(i) = y(xval(i));  NNames(i) = names(xval(i));  end  XR = [XX(nvars), XX(1)];  YR = [YY(nvars), YY(1)];    figure  plot(XX, YY, '-\*', XR, YR, '--')  text(XX+1,YY+1,NNames)    % вивід результатів  disp('Початкова популяція:');  PrintIter(RET.population.s0, RET.fvals.s0, nvars, population\_size);  disp('Покоління 1:');  PrintIter(RET.population.s1, RET.fvals.s1, nvars, population\_size);  disp('Покоління 2:');  PrintIter(RET.population.s2, RET.fvals.s2, nvars, population\_size);  disp('Результат:');  PrintIter(population, scores, nvars, population\_size);  fprintf('Best:\n');  PrintOne(xval, fval, nvars);  PrintOneStr(NNames, fval, nvars);  end  function way\_matrix = getWayMatrix(X, Y, M)  %% Формує матрицю з відстаннями між кожним містом  ret = zeros(length(Y), length(X));  for i = 1:1:length(Y)  for j = 1:1:length(X)  if (i == j)  ret(i,j) = M;  else  v = ((X(i) - X(j)).^2 + (Y(i)-Y(j)).^2).^(1/2);  ret(i,j) = v;  end  end  end  way\_matrix = ret;  end |

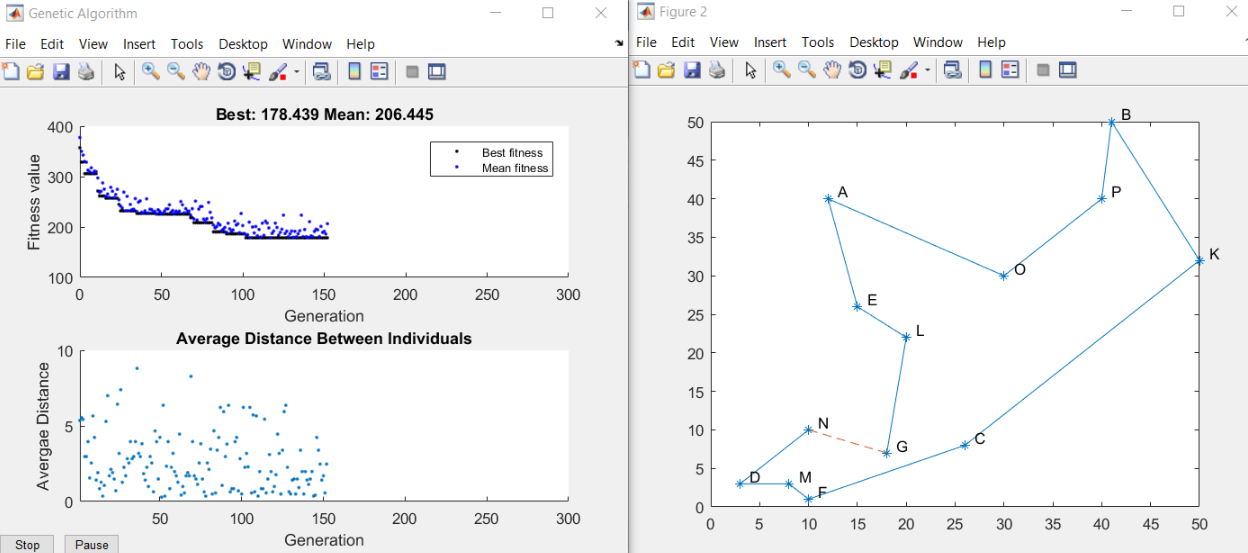
|  |  |
| --- | --- |
|  | https://lh6.googleusercontent.com/kmsLK-quAHaXLhxuu5B-ZgJJ2DLTE_LV1RkQ2sYZxUJj9hbsY0eB69D5ceeqGlUPZn5NvFoIh6yI6UsbqoTY5MCjqfW5rp2sTRxfL0Op3vR4ZbORXEkRIZXaGb5q7wQuIBcSLFPn |

Рис. 1. Початкові дані

**Результат виконання програми з елітою**







Нижче приведено параметри які були на вході функції у найкращого розв’язку

Початкова популяція:

[ 13,   5, 1, 10,   3, 2, 7, 4,   8, 11, 6, 12, 9, ] => 392.57

[  8,  1, 11,   9, 13, 2,   6, 10, 12, 4,   5, 3, 7, ] => 351.01

[  8,  3, 11,   1, 5, 6,   4, 7, 13, 12,  10, 2, 9, ] => 355.43

[ 11,  12, 4,   8, 7, 2,   1, 13, 5, 3,   6, 9, 10, ] => 389.97

Генерація 1:

[  8,  1, 11,   9, 13, 2,   6, 10, 12, 4,   5, 3, 7, ] => 351.01

[  8,  1, 11,   9, 13, 2,   6, 10, 12, 4,   5, 3, 7, ] => 351.01

[  8,  1, 11,   9, 13, 2,   6, 10, 12, 4,   5, 3, 7, ] => 351.01

[  8,  1, 11,   9, 13, 6,   2, 10, 12, 4,   5, 3, 7, ] => 444.76

Генерація 2:

[  8,  1, 11,   9, 13, 2,   6, 10, 12, 4,   5, 3, 7, ] => 351.01

[  8,  1, 11,   9, 13, 2,   6, 10, 12, 4,   5, 3, 7, ] => 351.01

[  8,  1, 11,   9, 13, 2,   6, 10, 12, 4,   5, 3, 7, ] => 351.01

[  8,  1, 11,   9, 13, 2,   5, 10, 12, 4,   6, 3, 7, ] => 327.33

Резульата:

[  3,  9, 5,  12, 8, 13,   2, 1, 11, 4,  10, 6, 7, ] => 176.65

[  3,  9, 5,  12, 8, 13,   2, 1, 11, 4,  10, 6, 7, ] => 176.65

[  3,  9, 5,  12, 8, 13,   2, 1, 11, 4,  10, 6, 7, ] => 176.65

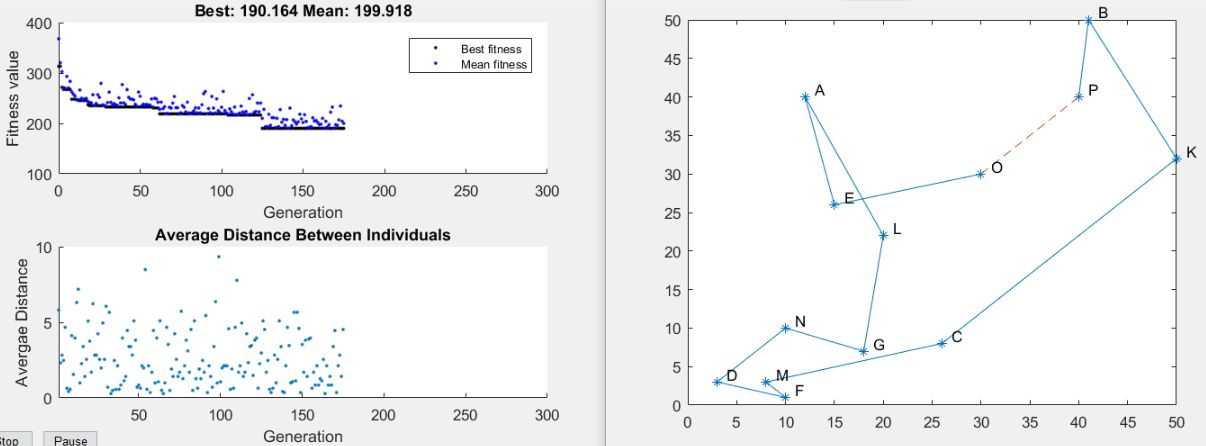
[  3,  9, 5,  12, 8, 13,   2, 10, 11, 4,   1, 6, 7, ] => 249.91

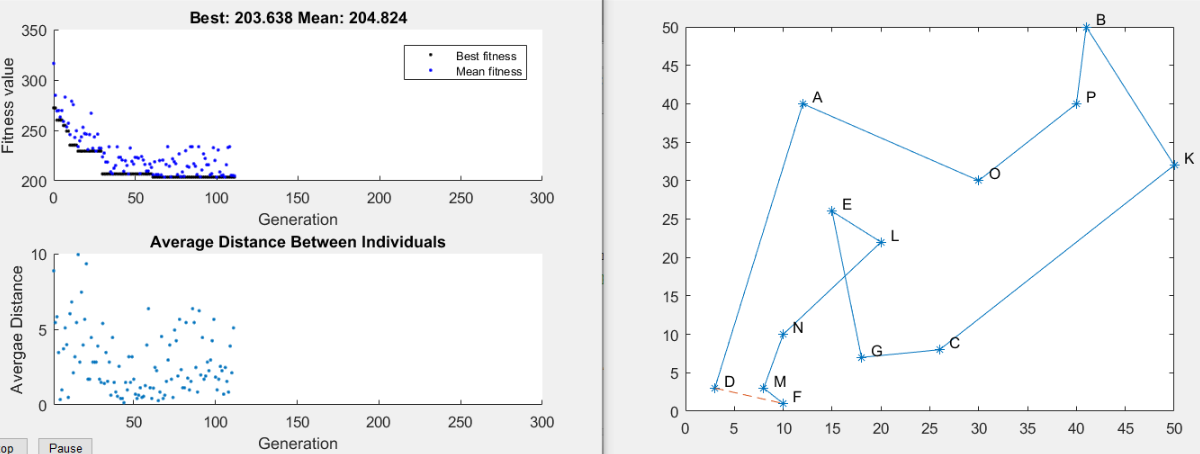
Оптимальний шлях:

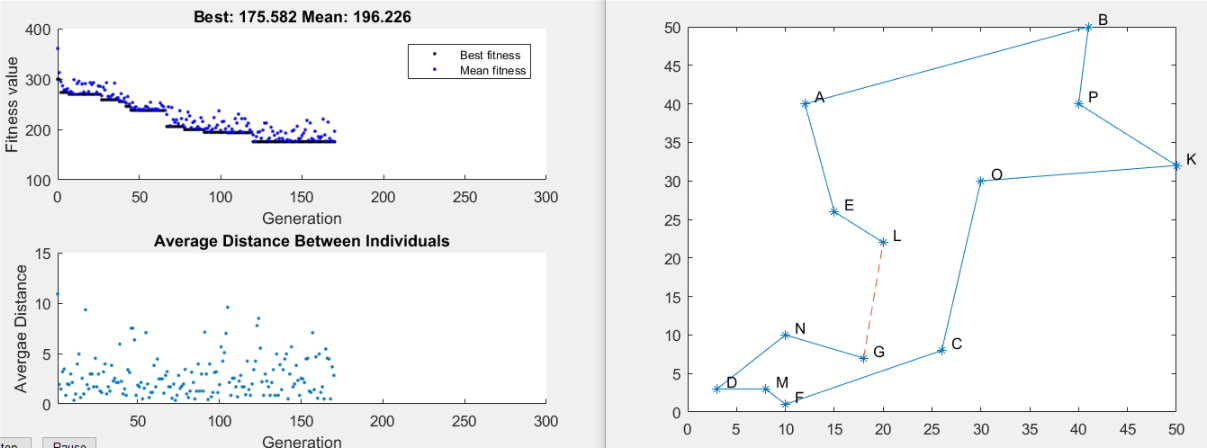
[  3,  9, 5,  12, 8, 13,   2, 1, 11, 4,  10, 6, 7, ] => 176.65

[  C,  L, E,   O, K, P,   B, A, N, D,   M, F, G, ] => 176.65

**Результат виконання програми без еліти**







Початкова популяція:

[  5,  3, 12,   7, 13, 1,   6, 2, 4, 10,  11, 8, 9, ] => 390.49

[ 11,   5, 1, 13,   4, 2, 3, 10,  12, 6, 7, 9,  8, ] => 407.59

[  2,  4, 10,  11, 3,  5, 12, 13,   9, 6, 8, 7,   1, ] => 345.20

[ 13,  12, 11,   3, 8, 2,   5, 7, 4, 9,   6, 10, 1, ] => 299.56

Генерація 1:

[ 13,  12, 11,   3, 8, 2,   5, 7, 4, 9,   6, 10, 1, ] => 299.56

[  2,  4, 10,  11, 3,  5, 12, 13,   9, 6, 8, 7,   1, ] => 345.20

[ 13,  12, 11,   3, 8, 2,   5, 7, 4, 9,   6, 10, 1, ] => 299.56

[ 13,  12, 11,   1, 8, 2,   5, 7, 4, 9,   6, 10, 3, ] => 306.79

Генерауція 2:

[ 13,  12, 11,   3, 8, 2,   5, 7, 4, 9,   6, 10, 1, ] => 299.56

[ 13,  12, 11,   1, 8, 2,   5, 7, 4, 9,   6, 10, 3, ] => 306.79

[ 13,  12, 11,   3, 8, 2,   5, 7, 4, 9,   6, 10, 1, ] => 299.56

[ 13,  12, 11,   3, 8, 2,   5, 9, 4, 7,   6, 10, 1, ] => 273.47

Резульата:

[  9,  5, 1,   2, 13, 8,  12, 3, 6, 10,   4, 11, 7, ] => 175.58

[  9,  5, 1,   2, 13, 8,  12, 3, 6, 10,   4, 11, 7, ] => 175.58

[  9,  5, 1,   2, 13, 8,  12, 3, 6, 10,   4, 11, 7, ] => 175.58

[  9, 10,   1, 2, 13,   8, 12, 3, 6,   5, 4, 11, 7, ] => 258.16

Оптимальний шлях:

[  9,  5, 1,   2, 13, 8,  12, 3, 6, 10,   4, 11, 7, ] => 175.58

[  L,  E, A,   B, P, K,   O, C, F, M,   D, N, G, ] => 175.58

**Висновок:** На даній лабораторній роботі, при використанні генетичного алгоритму, була використана селекція методом усічення. Розв’язано задачу комівояжера і досліджено результати виконання функції матлабу. Можемо зауважити, що результати не є однозначними, і можна аналізувати отримані результаті відштовхуючись від вибірки результатів.