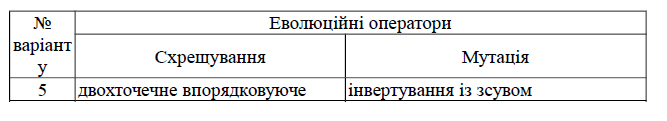
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **НУЛП, ІКНІ, САПР** | | Тема | оцінка | підпис |
| КНс-13 | 3 | КОМБІНАТОРНАОПТИМІЗАЦІЯ ЗА ДОПОМОГОЮ ЕВОЛЮЦІЙНИХМЕТОДІВ |  |  |
| Сологуб С.І. | |
| № залікової: | |
| **Методи нечіткої логіки та еволюційні алгоритми при автоматизованому проектуванні** | | Викладач: | |
| к.т.н. Кривий Р.З. | |

**Мета:**

Ознайомитися з основними теоретичними відомостями, вивчити еволюційні оператори схрещування та мутації, що використовуються при розв’язуванні задач комбінаторної оптимізації.

**Завдання:** Розробити за допомогою пакету Matlab програмне забезпечення для вирішення задачі комівояжера.

**Варіант 5.**



**Теоретичні відомості:**

При використанні методів еволюційного пошуку для розв’язку задач комбінаторної оптимізації, як правило, застосовуються негомологічні числові хромосоми, тобто такі хромосоми, гени яких можуть приймати значення в заданому інтервалі. При цьому інтервал однаковий для всіх генів, але в хромосомі не може бути двох генів з однаковим значенням. Комбінаторні задачі оперують із дискретними структурами або розміщенням об'єктів, незначні зміни яких часто викликають стрибкоподібну зміну показників якості (фітнесс-функції). Традиційні оператори еволюційні оператори, що генерують нових нащадків, не можуть бути застосовані при використанні негомологічних хромосом, оскільки внаслідок виконання таких операторів генеруються нащадки, що містять однакові гени і тому не можуть бути інтерпретовані при розв’язку комбінаторної задачі. Тому для розв’язку задач комбінаторної оптимізації були розроблені спеціальні генетичні оператори, що не створюють неприпустимих рішень.

При двохточечному впорядковуючому схрещуванні змінюється частина хромосоми, що перебуває між точками схрещування. Також застосовується позиційно впорядковуюче схрещування, що виконується в наступній послідовності.

Крок 1. Випадковим чином вибрати деяку кількість позицій (генів) у першій батьківській хромосомі.

Крок 2. Скопіювати в хромосому першого нащадка обрані на попередньому кроці гени першої батьківської хромосоми. При цьому копіювання генів відбувається в ті ж позиції, на яких вони розташовувалися в першій батьківській хромосомі.

Крок 3. Інші гени в нащадку копіюються із другого батька в упорядкованому вигляді зліва направо, крім елементів, що вже ввійшли в нащадка.

Для створення другого нащадка застосовується аналогічна послідовність дій за винятком того, що перший батько тепер уважається другим, а другий - першим.

В цьому випадку кожна хромосома представляє собою впорядковану послідовність чисел і застосування стандартних операторів схрещування неминуче привело б до утворення нелегальних рішень.

Інвертування із зсувом: випадковим чином вибирається елемент, який переміщується на випадково обирану позицію, зсуваючи інші елементи вправо по циклу. Іншою формою такого виду інвертування є випадковий вибір підстроки й переніс її на випадково обирану позицію із зсувом елементів, що залишилися.

**Виконання лабораторного завдання:**

Схрещення: двохточечне впорядковуюче.

Дане схрещення полягає у випадковому виборі двох генів у батьківській хромосомі та створення нащадка, таким чином: копіюємо гени першого батька, що знаходяться зліва від 1го гена та справа від 2го гена із збереженням індексів, тоді копіюємо гени без повторення від другого батька у місце між першим та другим обраним геном.

Код CrossoverFcn.m

%% Двохточечне впорядковуюче схрещення (вар5)

function [xoverKids] = CrossoverFcn( parents, nvars, thisPopulation )

% parents - індекси батьків в поточній популяції, що беруть участь у схрещуванні.

% nvars - кількість змінних (генів)

% thisPopulation - поточна популяція (матриця)

result = zeros(length(parents)/2, nvars);

for i = 1:2:length(parents)-1

parent1 = thisPopulation(parents(i), :);

parent2 = thisPopulation(parents(i+1), :);

separator = randi([2, nvars-1],1,1); %випадковий індекс умовного поділу

gene1 = randi([1, separator-1],1,1); %випадковий перший ген

gene2 = randi([separator+1, nvars],1,1); %випадковий другий ген

child = zeros(1, nvars); %ініціалізація нащадка

child(1,1:gene1) = parent1(1:gene1); %копіювання генів зліва від 1го гена

child(1,gene2:nvars) = parent1(gene2:nvars);%копіювання генів справа від 2го гена

nextGene = gene1+1; %наступний ген після 1го

inarr = ismember(parent2(1,:), child); %перевірка наявності генів нащадка в другого батька

for j = 1:1:nvars

if (inarr(j) == 0)

child(1, nextGene) = parent2(j); %копіювання гену нащадку

nextGene = nextGene + 1;

end

end

result((i+1)/2,:) = child;

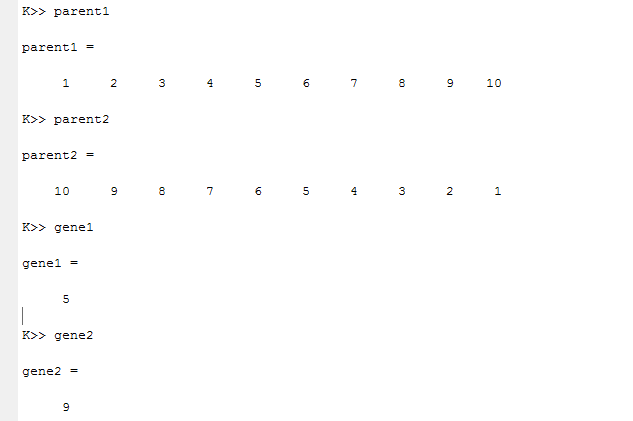
end;

xoverKids = result;

end

%% Кінець

Тестування на заданих значеннях:



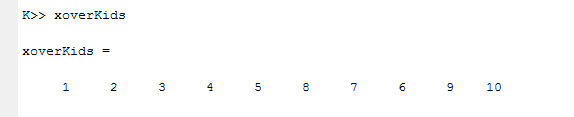


Рис.1. Результат тестування кросовера

Мутація: інвертування із зсувом.

Ця мутація полягає у зсуві елементів вправо на випадковий крок. Тобто обирається випадковий елемент та зсувається на випадкову позицію, зсуваючи таким чином і всі решту елементи вправо.

Код MutationFcn.m

%% Мутація інвертування із зсувом

function [ mutationChildren ] = MutationFcn( parents, nvars, thisPopulation)

% parents - номер особини в популяції, що мутує

% nvars - кількість змінних

% thisPopulation - поточна популяція

mutant = thisPopulation(parents, :); %особина, що мутує

offset = randi(nvars); %випадковий крок зміщення

mutationChildren = circshift(mutant',offset)'; %зміщення вправо на крок

end

%% Кінець

Тестування на заданих значеннях:

mutant = 1:1:10;

offset = 3;

mutationChildren = circshift(mutant',offset)';

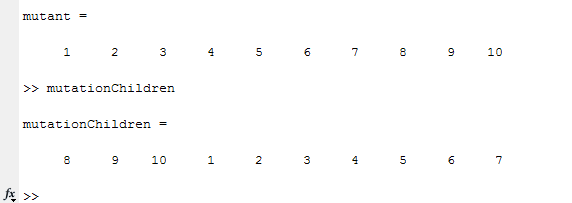


Рис.2. Результат тестування мутації

Оптимізаційна функція:

%% Оптимізуюча функція

function [ result ] = optim\_function( way, array)

sum = 0;

for i = 1:1:length(way)-1

sum = sum + array(way(i), way(i+1));

end

result = sum;

end

Основний код виклику ГА та задання параметрів:

%% Варіант 5

%% Стартова популяція

startPopulation = [

6, 10, 1, 2, 7, 3, 4, 9, 5, 8;

2, 5, 3, 9, 6, 7, 10, 4, 8, 1;

6, 7, 10, 8, 9, 3, 4, 5, 1, 2;

7, 8, 4, 9, 6, 2, 10, 1, 5, 3;

7, 10, 5, 8, 9, 2, 6, 1, 3, 4

];

%% Масив відстаней між містами

n = 10; % кількість міст

array = randi(1000, n, n); % діапазон відстаней 1-1000

for i=1:1:n

array(i,i)=100000;

end

disp('Array of distance')

disp(array)

%% Параметри ГА

options = gaoptimset(...

'EliteCount', 0, ...

'PopulationSize', 5, ...

'InitialPopulation', startPopulation, ...

'MutationFcn', @MutationFcn, ...

'CrossoverFcn', @CrossoverFcn, ...

'TimeLimit', 3 ...

);

%% Запуск методу

[x,fval,exitflag,output,population,scores] = ga(@(x)optim\_function(x, array), 10, options);

%% Відображення результатів

disp('Best child:');

disp(x);

fprintf('f(x) = %d\n', fval);

disp('Last population:');

for i=1:1:5

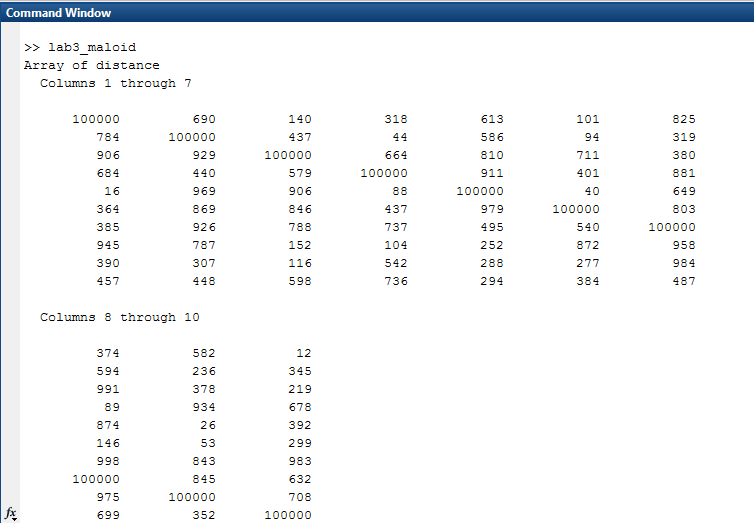
fprintf('\t%d', population(i,:));

fprintf('\t=>\t%d\n', scores(i));

end;

Результати виконання:

Прогін №1



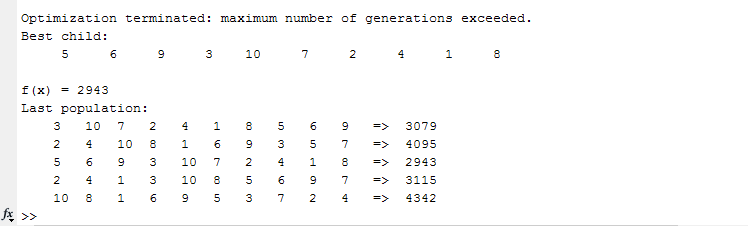
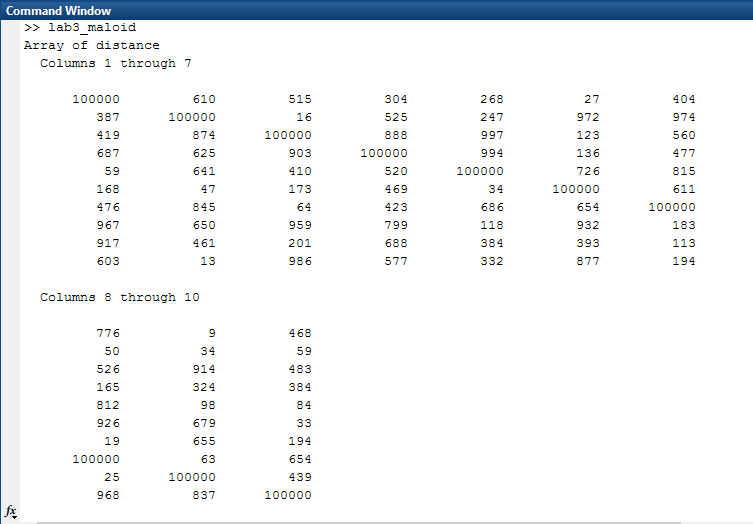


Рис.3. Результат прогону №1

Прогін №2



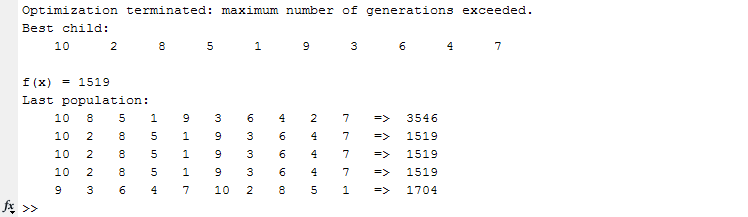
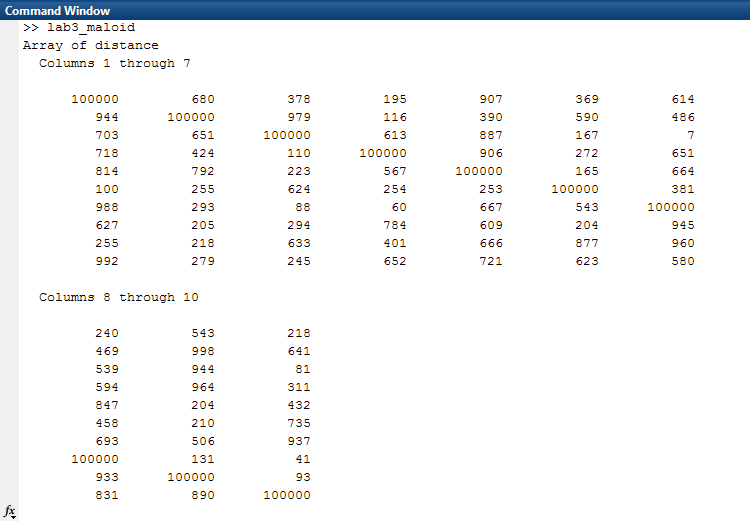


Рис.4. Результат прогону №2

Прогін №3



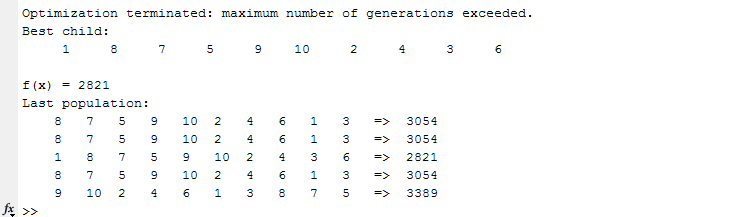


Рис.5. Результат прогону №3

**Висновки.**

Виконуючи лабораторну роботу я ознайомився з основними теоретичними відомостями та вивчив еволюційні оператори схрещування та мутації, що використовуються при розв’язуванні задач комбінаторної оптимізації.

Також я розробив за допомогою пакету Matlab програмне забезпечення для вирішення задачі комівояжера, використовуючи функції схрещування та мутації згідно індивідуального варіанту.