

Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Методи планування експерименту
Лабораторна робота №3а
«Дослідження генетичного алгоритму»

Виконав:
студент II курсу ФІОТ
групи ІВ-92
Скворцов П. С.
номер у списку групи – 21
Перевірив:
ас. Регіда П. Г.

Мета:

ознайомлення з принципами реалізації генетичного алгоритму, вивчення та дослідження особливостей даного алгоритму з використанням засобів моделювання і сучасних програмних оболонок.

Завдання:

Налаштувати генетичний алгоритм для знаходження цілих коренів діофантового рівняння $ax_1 + bx_2 + cx_3 + dx_4 = y$. Розробити відповідний мобільний додаток і вивести отримані значення. Провести аналіз витрат часу на розрахунки.

Лістинг програми:

```
public async void Genetic(object sender, EventArgs e)
{
    Double.TryParse(aEntry.Text, out double a);
    Double.TryParse(bEntry.Text, out double b);
    Double.TryParse(cEntry.Text, out double c);
    Double.TryParse(dEntry.Text, out double d);
    Double.TryParse(yEntry.Text, out double y);
    Random random = new Random();
    bool Successful = false;
    double[][] population = new double[4][];
    population[0] = new double[4];
    population[1] = new double[4];
    population[2] = new double[4];
    population[3] = new double[4];
    var startTime = DateTime.Now;
    for (int i = 0; i < population.Length; i++)
    {
        for (int j = 0; j < population[i].Length; j++)
        {
            population[i][j] = (int)(1 + random.NextDouble() * (y / 2));
        }
    }
    while (!Successful)
    {
        double[] delta = new double[4];
        double roulette_parameter = 0;
        double[] chance_of_parenthood = new double[4];
        for (int i = 0; i < 4; i++)
        {
            delta[i] = Math.Abs(y - (a * population[i][0] + b * population[i][1] + c * population[i][2] + d * population[i][3]));
            roulette_parameter += 1 / delta[i];
            if (delta[i] == 0)
            {
                Successful = true;
                await DisplayAlert("Result", $"Знайдені корені рівняння:\nX1 = {population[i][0]}\nX2 = {population[i][1]}\n" +
                    $"X3 = {population[i][2]}\nX4 = {population[i][3]}\n" +
                    $"Execution time = {(DateTime.Now - startTime).TotalMilliseconds} ms\n", "Got it!");
                break;
            }
        }
        chance_of_parenthood[0] = 1 / delta[0] / roulette_parameter;
        for (int i = 1; i < 4; i++)
        {
            chance_of_parenthood[i] = chance_of_parenthood[i - 1] + 1 / delta[i] / roulette_parameter;
        }
        double chance1 = random.NextDouble();
        double chance2 = random.NextDouble();
        double chance3 = random.NextDouble();
        double chance4 = random.NextDouble();
        double[] father1 = new double[4];
        double[] father2 = new double[4];
        double[] father3 = new double[4];
        double[] father4 = new double[4];
        //знаходження 1 та 2 батьків
        for (int i = 0; i < 4; i++)
        {
            if (chance1 < chance_of_parenthood[i])
            {
                father1 = population[i];
            }
        }
    }
}
```

```

        for (int j = 0; j < 4; j++)
        {
            if (chance2 < chance_of_parenthood[j])
            {
                father2 = population[j];
                break;
            }
        }
        break;
    }
}
//знаходження 3 та 4 батьків
for (int i = 0; i < 4; i++)
{
    if (chance3 < chance_of_parenthood[i])
    {
        father3 = population[i];
        for (int j = 0; j < 4; j++)
        {
            if (chance4 < chance_of_parenthood[j])
            {
                father4 = population[j];
                break;
            }
        }
        break;
    }
}
//кросовер 1 та 2 батьків
int index1 = random.Next(3);
for (int i = 0; i <= index1; i++)
{
    double k = father1[i];
    father1[i] = father2[i];
    father2[i] = k;
}
//кросовер 3 і 4 батьків
int index2 = random.Next(3);
for (int i = 0; i <= index2; i++)
{
    double k = father3[i];
    father3[i] = father4[i];
    father4[i] = k;
}
//нова популяція
population = new double[][] { father1, father2, father3, father4 };

double chance_of_mutation = random.NextDouble();
//шанс мутації 20%
if (chance_of_mutation < 0.2)
{
    int mutation = random.Next(2);
    if (mutation == 0)
        population[random.Next(4)][random.Next(4)] += 1;
    else population[random.Next(4)][random.Next(4)] -= 1;
}
}
}

```

Виконання роботи програми:

4G 55% 13:54

MAIN TASK

Genetic algorithm

4

2

3

5

20

START THE PROCESS!

4G 55% 13:54

MAIN TASK

Genetic algorithm

4

Result

Знайдені корені рівняння:

X1 = 7

X2 = -5

X3 = 4

X3 = -2

Execution time = 358,863 ms

GOT IT!

20

START THE PROCESS!

Висновок:

У ході виконання лабораторної роботи ознайомлено з принципами реалізації генетичного алгоритму, а також вивчено та досліджено особливості даного алгоритму з використанням засобів моделювання і сучасних програмних оболонок. Розроблено відповідну програму з використанням мови програмування C#. Результати роботи, наведені у протоколі, підтверджують правильність виконання – кінцеву мету роботи було досягнуто