Отчет по 3 лабораторной работе

ГЕНЕТИЧЕСКИЙ АЛГОРИТМ ДЛЯ ЗАДАЧИ ОПТИМИЗАЦИИ НЕПРЕРЫВНОЙ ФУНКЦИИ

В ходе выполнения данной лабораторной работы необходимо решить задачу оптимизации некой функции. В качестве задачи необходимо реализовать следующее:

1. Установить параметры ген. Алгоритма
2. Реализовать инициализацию индивида
3. Реализовать мутацию индивидов
4. Реализовать кроссовер

Лабораторная работы выполнялась на языке java в IDE IntelliJ

Реализация инициализации индивидов:

Согласно условию, значение индивида должно быть в рамках от -5 до 5 включительно. В качестве реализации функции рандома использовалось random.nextDouble() который возвращает случайное число от 0 до 1. Сама формула получения значения индивида имеет следующий вид:

solution[ind] = rangeMin + (rangeMax - rangeMin) * random.nextDouble();

Реализация мутации:

Реализация мутации состоит в том что мы изменяем значение (ген) на значение находящийся в некой окрестности. Формула состоит в следующем:

number = a * random.nextGaussian() + population.get(parent)[ind]; , где a это параметр который указывает на уровень исследования окрестностей точки.

Также мы можем взять абсолютно случайное значение в рамках заданной задачи по формуле представленной в инициализации.

Реализация кроссовера:

Кроссовер состоит в том что один ребенок получает часть значений 1 родителя и часть значений является комбинацией 1 и 2 родителя. Второй ребенок получает значения, наоборот.

Установка параметра мутации a: в ходе проведенного тестирования было выявлено что поиск в окрестностях не эффективен. Также есть лишь малая вероятность того, что мутация будет проведена над элементом.

Размер проблемы	Размер популяции	Количество итераций	Результат
2	75	750	9.999
10	99	9999	9.9
20	99	9999	9.611
50	10	9999	9.97
100	10	9999	8.99

Ответы на вопросы:

1. Нельзя однозначно сказать, что важнее так как без мутации невозможно выйти, как минимум из локального минимума т. к. популяция вырождается. Без кроссовера невозможно “оптимизировать” популяцию путем скрещивания значений. Однако решения

задачи только с помощью мутаций показывает более высокую оценку по сравнению с решением, основанным на одном кроссовере.

2. При увеличении размера популяции увеличивается количество вариантов решения, однако, падает скорость оптимизации решения. На практике следует подбирать оптимальное значение. В этой задаче это 10.
3. Зная область определения, мы можем более грамотно проводить инициализацию и мутацию что позволит оптимизировать процесс работы алгоритма.