МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ

по учебной практике

Тема: Генетические алгоритмы

Студент гр. 3343 Жучков О.Д.

Преподаватель Жангиров Т.Р.

Санкт-Петербург
2025

Цель работы

Изучение принципов генетических алгоритмов и их применения для решения оптимизационных задач, программная реализация генетического алгоритма для решения задачи и пользовательского интерфейса для него.

Задание

Вариант 12. Задача прямоугольного раскроя.

Дана полубесконечная полоса шириной N, необходимо разместить M прямоугольников, размеры i-го прямоугольника задаются пользователем как ширина wi и длина li. Разместить прямоугольники необходимо так, чтобы потребовался участок полосы минимальной длины.

Требования к решению:

Программа должна иметь GUI

Должна быть возможность задать данные из разных источников по выбору пользователя: из файла, ввод через GUI, случайная генерация

При реализации алгоритмов нельзя использовать библиотеки решающие задачу напрямую. Генетический алгоритм должен быть реализован вручную. Использовать библиотеки для графического интерфейса/загрузки данных – можно.

Пользователь должен иметь возможность задавать параметры генетического алгоритма через GUI.

В приложении должна быть реализована пошаговая демонстрация работы генетического алгоритма. Т.е. должна быть кнопка «следующий шаг» и кнопка «выполнить до конца» (чтобы перейти к конечным результатам). На каждом шаге алгоритма должна быть показано графическое представление наилучшего решения, стоимость этого решения, средняя стоимость поколения.

В GUI должно быть поле с графиком изменения приспособленности лучший и средней в зависимости от поколения. График должен строиться по ходу выполнения генетического алгоритма.

После завершения выполнения генетического алгоритма, программа не должна закрывать, а должна быть возможность выполнить его на этих же данных, но с другими параметрами алгоритма, либо загрузить новые данные.

Выполнение работы

Программа для реализации алгоритма написана на языке Python. Для графического пользовательского интерфейса и вывода графиков используются библиотеки tkinter и matplotlib.

Структура программы:

В файле interface.py описан класс, создающий окно tkinter и обеспечивающий пользовательское взаимодействие с программой.

В файле model.py описан класс, производящий все вычисления для решения задачи генетическим алгоритмом.

Основной файл main.py отвечает за создание объектов пользовательского интерфейса и исполнителя генетического алгоритма и осуществляет взаимодействие между ними.

Принцип работы алгоритма:

В основе генетического алгоритма лежит дарвинская теория эволюции. Каждое возможное решение задачи рассматривается как особь в популяции и представляется в виде описания её генотипа, набором генов (хромосомой).

Для данной задачи выбрано следующее генетическое представление решения – перестановки чисел от 0 до N, которые обозначают то, в каком порядке берутся прямоугольники для расстановки на полосе. Для декодирования данного представления написан алгоритм, работающий следующим образом: берется i-ый прямоугольник из перестановки и размещается в ближайшее к началу доступное место (не занятое предыдущими i-1 прямоугольниками), позиции рассматриваются слева направо, снизу вверх. Таким образом высчитывается длина участка, требуемая для размещения. Данный подход к кодированию решения исключает возможность некорректного решения или заведомо неоптимального. Число приспособленности равно длине участка, в данном случае генетический алгоритм решает задачу нахождения минимума функции приспособленности.

Первым шагом алгоритма является создание случайного набора хромосом, каждый последующий шаг – создание новой популяции на основе предыдущей.

1) Используется принцип “естественного отбора” из эволюционной теории, где выживают и дают потомство только особи с лучшими свойствами, а часть погибает. В данной работе отбор реализован стохастической универсальной выборкой с ранжированием: особи располагаются на рулетке с секторами, размер которых пропорционален их рангу (индексу из отсортированного по приспособленности массива). Рулетка крутится один раз, и по равноудаленным друг от друга точкам отбора определяются отбираемые особи. Данный подход старается сохранить разнообразность популяции.

2) После получения выборки особей они используются для образования новой популяции путем скрещивания. Используется принцип наследственности: получившаяся путем скрещивания особь совмещает в себе определенные признаки своих родителей. Для данной задачи выбран метод упорядоченного скрещивания: сначала происходит двухточечное скрещивание (родители обмениваются частями, случайно определенными путем указания двух точек на хромосоме), затем оставшиеся гены записываются на свободные места с сохранением их порядка в родительской хромосоме. Пары родителей выбираются случайным образом и скрещиваются с некоторой вероятностью (иначе записываются в следующую популяцию в исходном виде).

3) Когда завершено скрещивание особей, каждый потомок с некоторой вероятностью подвергается мутации: меняются местами два случайных числа в перестановке. Принцип изменчивости помогает алгоритму не зацикливаться на одном локально оптимальном решении.

4) Формирование популяции закончено, алгоритм повторяется множество раз до того момента, как выполнится условие окончания. Реализованы два возможных условия окончания: алгоритм выполняется конкретно заданное число раз или пока среднее значение приспособленности в популяции не

начнет сходиться с минимальным значением приспособленности (что означает, что решения в популяции становятся однородными) с определенным порогом схождения.

Графический пользовательский интерфейс:

В левой верхней части экрана находятся поля для ввода числовых значений параметров генетического алгоритма: размер популяции, размер выборки (сколько родителей отбирается из популяции для дальнейшей работы), вероятности скрещивания и мутации, количество поколений и нижний порог разности среднего и минимума приспособленности.

Ниже находятся три кнопки, с помощью которых производится ввод значений для задачи (набор прямоугольников и ширина полосы). Есть варианты ввода вручную через текстовое диалоговое окно, случайная генерация на основе параметров (ширина, число прямоугольников, диапазон значений, которые может принимать сторона прямоугольника), чтение из текстового файла.

Пользователю доступно изменение параметров алгоритма: размер популяции, количество популяций, нижний порог разности среднего и минимума, размер выборки, вероятности скрещивания и мутации.

В окне интерфейса пользователю предоставляются график средней и минимальной приспособленности каждого поколения, изображение наилучшего решения. Есть возможность приостановить и возобновить работу алгоритма, загрузить новые данные или параметры и запустить алгоритм заново.

Следующий набор кнопок контролирует процесс выполнения алгоритма. Нажатие любой из кнопок “Следующий шаг”, “Выполнить до ..” инициализирует алгоритм и начнет его выполнение либо пошагово, либо до условия окончания. При пошаговом выполнении можно также использовать кнопку “Шаг назад” для возврата к предыдущему состоянию (поколению) алгоритма. Кнопка “Приостановить работу” приостанавливает выполнение алгоритма в режиме “Выполнить до ..”, если пользователю нужно досрочно завершить вы-

полнение алгоритма или выполнить его пошагово. Кнопка “Перезапуск с параметрами” инициализирует алгоритм заново с теми же входными данными, но возможно другими введёнными параметрами, а “Сброс” полностью удаляет текущий объект алгоритма.

В окне также выводится в текстовом и графическом виде информация о выполнении алгоритма. График приспособленности показывает изменение среднего и минимального значений приспособленности в зависимости от популяции, ниже представлено лучшее решение из текущего поколения.

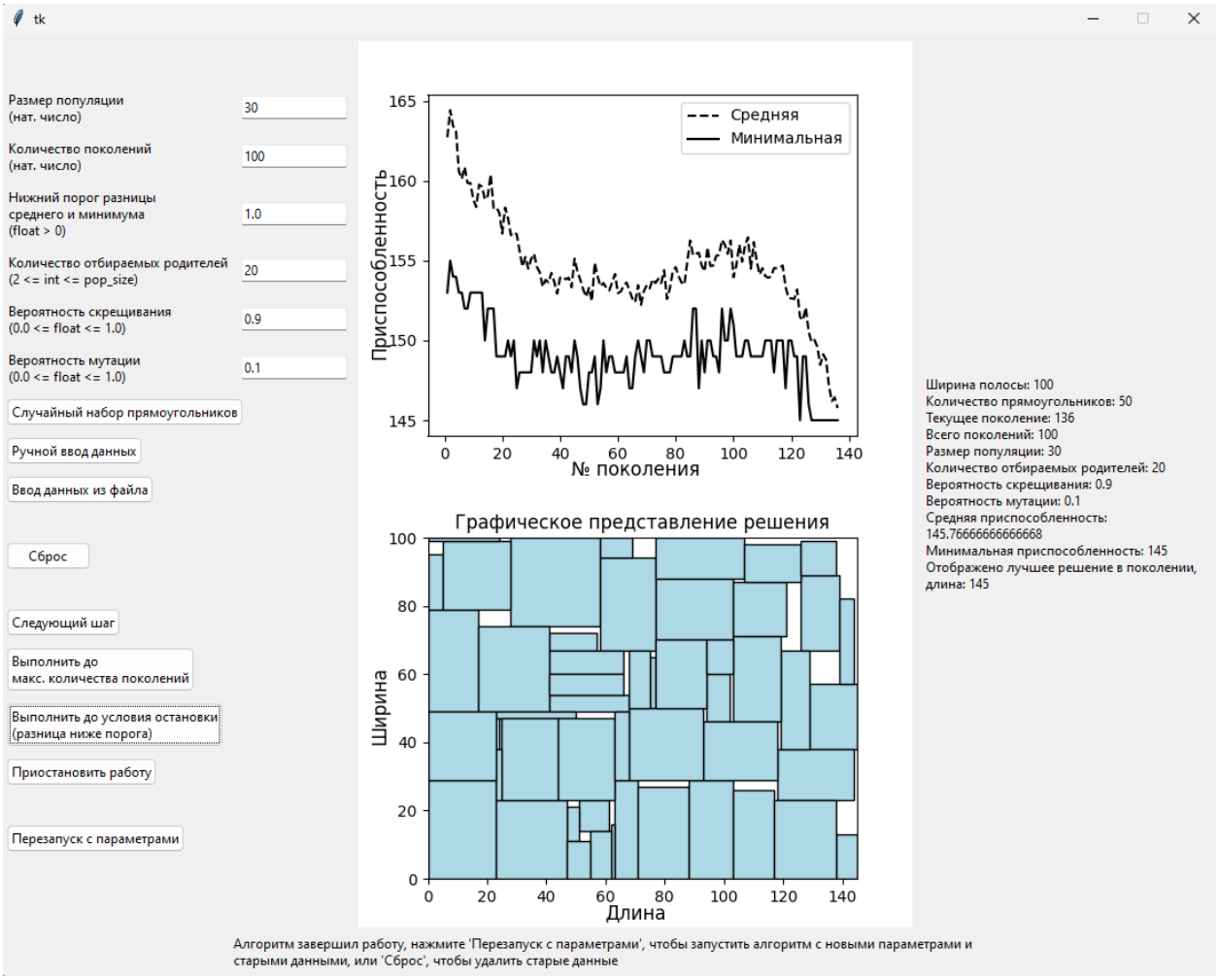

Рисунок 1 – Пример работы алгоритма

Тестирование показало, что алгоритм корректно находит приближение решения задачи.