Министерство науки и высшего образования Российской Федерации

Национальный исследовательский университет ИТМО

Генетические алгоритмы

Осень

2025

Лабораторная работа №2

ВВЕДЕНИЕ В ЭВОЛЮЦИОННЫЕ ВЫЧИСЛЕНИЯ

Цель работы

Целью данной лабораторной работы является получение студентом представления об возможностях применения эволюционных алгоритмов для решения различных классов задач и программных средств для их разработки.

Оборудование и программное обеспечение

Для выполнения лабораторной работы потребуется:

- браузер с доступом к сети Интернет;
- Java JDK версии 1.8 и выше;
- Watchmaker framework версии 0.7.1.

Краткие теоретические сведения

Эволюционные Вычисления (ЭВ) являются областью информатики в основе методов и алгоритмов которой лежат идеи и концепции, взятые из естественных природных процессов. Наиболее популярным примером является Генетический Алгоритм (ГА), вдохновением на создание которого был взят процесс эволюции. Эволюция считается одним из наиболее мощных природных процессов, который проявляется в создании разновидностей, стремящихся к выживанию в своей среде обитания за счёт адаптации к ней. Таким образом, ГА представляет собой популяцию индивидов, где индивид является одним из возможных решений некоторой поставленной задачи. Индивид или решение характеризуется двумя составляющими: генотип и фенотип. Генотип является внутренней структурой решения, тем, как решение представляется в программе для реализации алгоритма в виде набора генов. Фенотип представляет собой непосредственный предметно-ориентированный вид решения в терминах поставленной задачи. Качество каждого решения может быть оценено функцией полезности – фитнес-функцией. Начальный набор индивидов подвергается трём операциям: кроссовером, мутацией и селекцией. Кроссовер является ничем иным, как скрещиванием двух или более индивидов с целью воспроизведения потомства. Дочерние решения наследуют части генов от родителей. Мутация представляет собой небольшое случайное изменение в генотипе решения. Операторы мутации и кроссовера необходимы для обхода пространства поиска с целью нахождения оптимального решения. Селекция является регуляризатором сходимости алгоритма и осуществляет отбор индивидов среди популяции и полученных в процессе кроссовера потомков на основе значений их фитнес-функции. Перечисленные эволюционные операторы повторяются итерацией за итерацией над текущем поколением (популяцией решений) до тех пор, пока не будет удовлетворено условие терминации алгоритма. Схема генетического алгоритма представлена на рисунке 1.1.

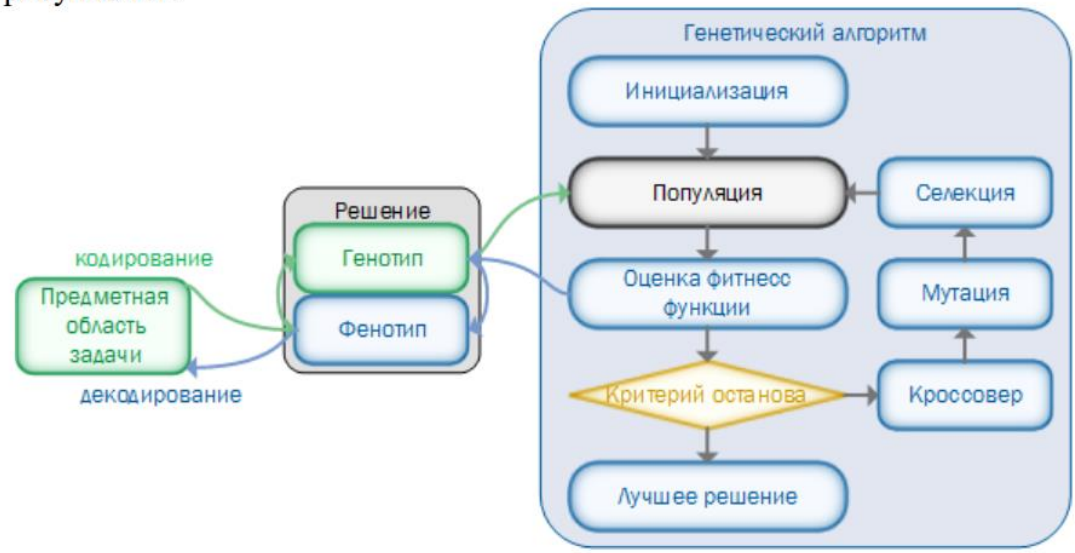

Рисунок 1.1 – Схема генетического алгоритма

Эволюционные алгоритмы получили широкое распространение и нашли своё применение для решение огромного числа задач, включающих промышленные, научные, образовательные и даже задачи развлекательного характера. В качестве примеров можно привести классические оптимизационные задачи:

- задача коммивояжёра;
- задача расстановки ферзей;
- транспортная задач;
- обучение параметров искусственной нейронной сети;
- задачи планирования и составления расписаний;
- разработка игровых стратегий.

Первой и наиважнейшей стадией при разработке эволюционного алгоритма является конструирование наиболее пригодного представления генотипа – структуры решений. Структура решений должна быть спроектирована таким образом, чтобы генотип решения мог быть раскодирован максимально однозначно в решение поставленной задачи с целью оценки его качества и валидности (при наличии ограничений). При этом, важно, чтобы генотип был удобен для реализации эволюционных операторов и позволял осуществлять полноценный поиск по пространству решений без усложнения размерности задачи и излишних вычислительных затрат. В общем можно выделить следующие типы представления решений: бинарные, целочисленные, вещественнозначные, комбинаторные и древовидные. Например, решения задачи обучения параметров искусственной нейронной сети очевидно могут быть закодированы в виде вектора вещественных чисел. В данном случае, вектор параметров является генотипом, а сама нейронная сеть, построенная на этом векторе значений, будет являться фенотипом.

К достоинствам эволюционных алгоритмов и мотивации для их применения относят:

- возможность представления решаемой задачи в формате оптимизации «чёрного ящика»;
- возможность изменять важность оптимизационных критериев или решать многокритериальную задачу без изменения в структуре самого алгоритма и кодировки решения;
- итеративность и возможность получения текущего результата в любой момент времени;
- возможность получения интуитивно неожидаемого решения;
- возможность динамической адаптации к изменяемой среде;
- интерес с точки зрения использования природных концепций и автоматического нахождения решений без использования специфики предметной области.

В настоящее время создано множество программных средств и фреймворков для упрощения разработки эволюционных алгоритмов и улучшения их производительности за счёт параллелизации вычислительных процессов. Также, существуют готовые примеры и решения задач, с некоторыми из которых и предлагается ознакомиться в рамках данной лабораторной работы.

https://watchmaker.uncommons.org/download.php.

Ход работы

Скачать проект, содержащий фрэймворк Watchmaker и реализованные на нём примеры по ссылке: https://watchmaker.uncommons.org/download.php.

Открыть среду разработки, поддерживающую Java (например, IntelijIDEA). Далее требуется создать проект из существующего источника, которым является путь к скачанному фреймворку. При создании проекта необходимо выбрать Maven в качестве средства сборки проекта. Структура проекта при правильной сборке должна принимать вид как показано на рисунке 1.2. Основные обязательные директории и файлы подчёркнуты красной линией.

Bits count. В данном примере решается задача подсчёта количество битов (‘1’) в строке заданной длины. Для работы с примером необходимо открыть файл BitsExample.java, который находится в директории:
“..\examples\src\java\main\org\uncommons\watchmaker\examples\bits\”.

Размерность проблемы определяется значением параметра ***BITS***, который изначально равен 20. Запустите пример, на экране должны выводиться результаты алгоритма на каждой итерации. Алгоритм останавливается в момент нахождения оптимального решения. Определите и выпишите, какой структурой данных и в каком виде закодированы решения в данной реализации примера.

Запустите программу несколько раз и заполните значения количеств итераций, которые были необходимы для получения конечного решения при

размерности проблемы 20 на основе таблицы 1.1. Повторите серию запусков для решения задачи с размерностями 50 и 100. Рассчитайте среднее количество итераций алгоритма для решения задачи в каждом случае.

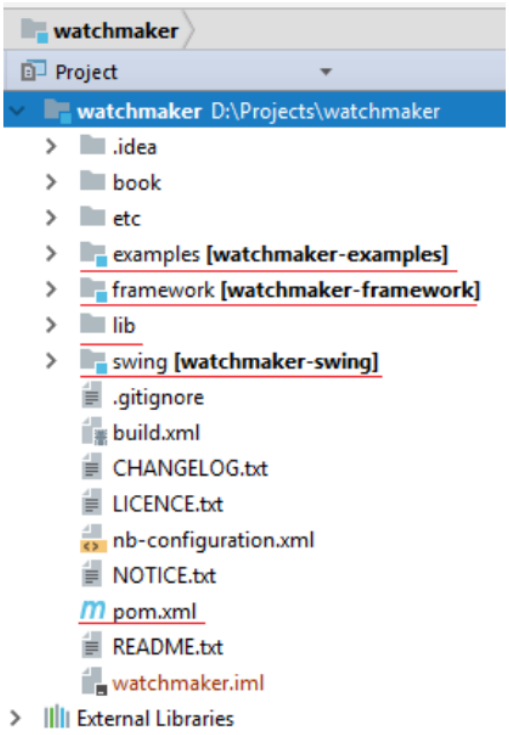

Рисунок 1.2 – Файловая структура проекта

Таблица 1.1 Результаты расчёта количества итераций алгоритма от размерности проблемы.

Размерность	Run 1	Run 2	Run 3	Run 4	Run 5	Среднее
20						
50						
100						

Travelling salesman problem. В данном примере решается задача коммивояжёра, целью которой является найти кратчайший путь, проходящий через все заданные точки (города). Для запуска примера необходимо запустить файл:
"..\examples\src\java\main\org\uncommons\watchmaker\examples\travellingsalesman\TravellingSalesmanApplet.java".

В открывшемся окне (рисунок 1.3) выберите все города и попробуйте найти оптимальное решение задачи. Выпишите найденный маршрут по всем 15 городам и дистанцию найденного маршрута.

Попробуйте поварьировать параметры алгоритма для анализа их влияния на его производительность.

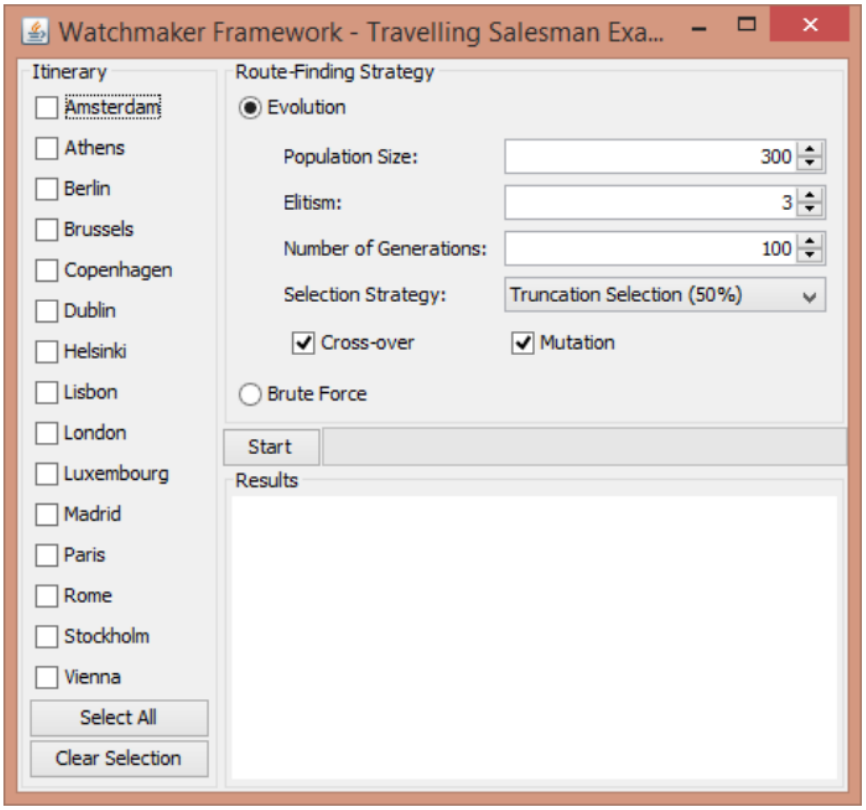

Рисунок 1.3 – Окно запуска алгоритма для решения задачи коммивояжёра

Mona Lisa. В последнем примере, который рассматривается в рамках лабораторной работы, представлена задача подбора множества разноцветных полигонов для наиболее точного воспроизведения картины. Для запуска примера необходимо запустить файл:
"..\examples\src\java\main\org\uncommons\watchmaker\examples\monalisa\MonaLisaApplet.java".

В открывшемся окне запустите алгоритм. Сделайте несколько снимков экрана, которые показывали бы плохое, среднее и хорошее решение задачи (субъективно). Вырежьте только части с получающимися на текущий момент картинами. Для выполнения можно остановить и запустить алгоритм несколько раз. В соответствии со структурой в таблице 1.2, заполните полученные результаты.

Таблица 1.2 Результаты оптимизации подбора полигонов для воспроизведения картины

Решение	Итерация	Фитнесс	Кол-во полигонов и углов	Рисунок
плохое				
среднее				
хорошее				

Вопросы:

1. К какому типу по структуре решений относится каждая из рассмотренных задач?
2. Как закодированы решения в задаче коммивояжёра?
3. Что является генотипом, а что фенотипом в задаче воспроизведения картины?

Литература

1. Рутковская Д., Пилиньский М., Рутковский Л. Нейронные сети, генетические алгоритмы и нечеткие системы. – 2013.
2. Eiben A. E., Smith J. Introduction to Evolutionary Computing. – 2015.
3. Satellite Truss Design, http://www.soton.ac.uk/~ajk/truss/welcome.html.
4. Genetic walkers, http://rednuht.org/genetic_walkers/.
5. Genetic cars, http://rednuht.org/genetic_cars_2/.