Отчет по 6 лабораторной работе

РАСПРЕДЕЛЕННЫЕ ЭВОЛЮЦИОННЫЕ АЛГОРИТМЫ

В ходе выполнения данной лабораторной работы необходимо решить задачу оптимизации некой функции с использованием распределенных моделей эволюционных алгоритмов. В качестве задачи необходимо реализовать следующее:

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

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 а 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 популяции на всех тестах был равен 100, количество итераций равно 1000, размер проблемы 100. В островной модели использовались 3 острова и 50 эпох.

Сравнивалось 3 модели:

* SingleThread – однопоточная модель
* Master-slave – распределенная модель
* Islands – остравная модель

|  |  |  |  |
| --- | --- | --- | --- |
| Архитектура | Сложность проблемы | Время выполнения (в милисек.) | Результат |
| SingleThread | 0 | 2113 | 0 |
| Master-slave | 0 | 487 | 0 |
| Islands | 0 | 48205 | 0 |
| SingleThread | 1 | 8628 | 7.37 |
| Master-slave | 1 | 760 | 7.69 |
| Islands | 1 | 59494 | 9.994 |
| SingleThread | 2 | 13986 | 7.38 |
| Master-slave | 2 | 909 | 7.73 |
| Islands | 2 | 68235 | 9.995 |
| SingleThread | 3 | 14899 | 7.41 |
| Master-slave | 3 | 940 | 7.79 |
| Islands | 3 | 71869 | 9.992 |
| SingleThread | 4 | 19138 | 7.36 |
| Master-slave | 4 | 1121 | 7.63 |
| Islands | 4 | 92568 | 9.992 |
| SingleThread | 5 | 23029 | 7.44 |
| Master-slave | 5 | 1541 | 7.73 |
| Islands | 5 | 112209 | 9.994 |

График зависимости времени выполнения SingleThread от сложности проблемы

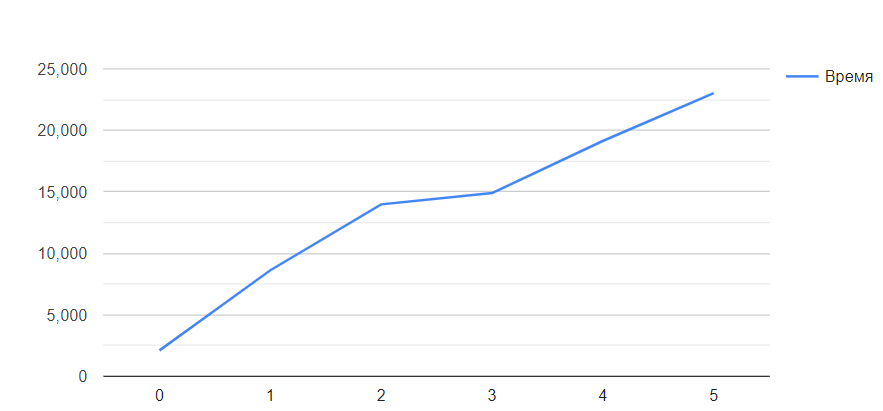


График зависимости времени выполнения Master-slave от сложности проблемы

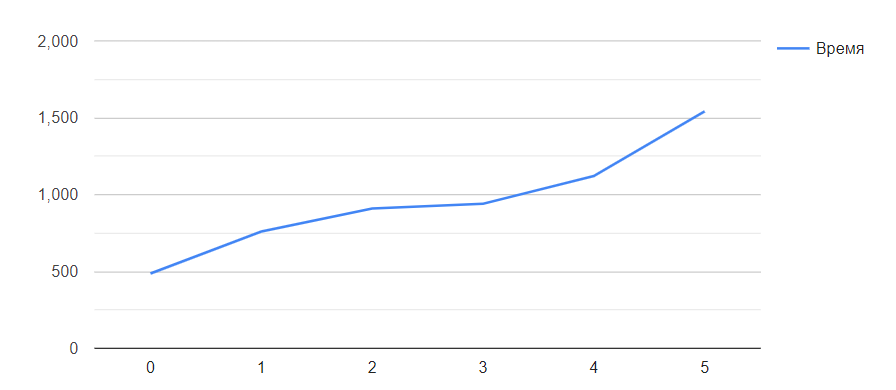
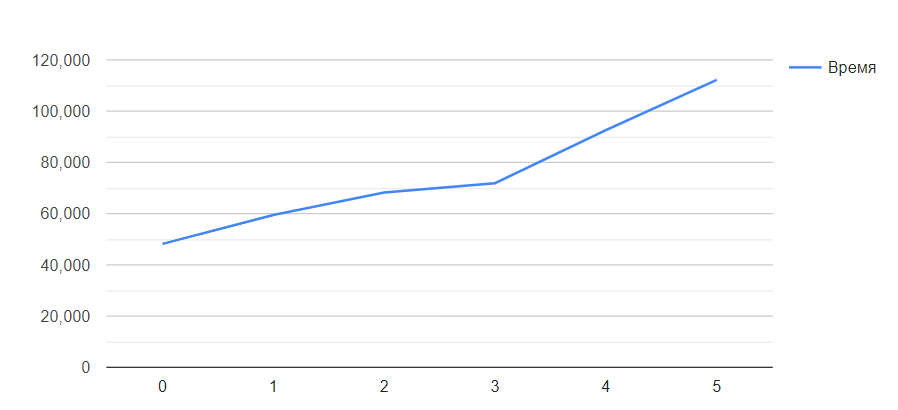


График зависимости времени выполнения Islands от сложности проблемы



Ответы на вопросы:

1. Распределенный алгоритм позволяет крайне сильно оптимизировать вычисление фитнесс функции, а островной улучшить качество решения. Нельзя сказать однозначно какой алгоритм лучших поскольку их следует использовать в различных ситуациях.
2. Увеличение размерности проблемы имеет слабое влияние на скорость распределенного алгоритма и более высокою на островную модель.
3. Увеличение размера популяции оказывает влияние на производительность как на распределенный алгоритм, так и на островную модель (в меньшей степени) за счет того, что происходит распределение популяции на несколько объектов.
4. Теоретически мы ограничены как ресурсами на выполнение селекционной стратегии между островами, так и размером популяции, которая так же уменьшается с ростом количества островов.