Лабораторная работа №6
Распределенные эволюционные алгоритмы.

Выполнил Домницкий Е.А. М4130

Цель работы

Освоение принципов построения распределенных и параллельных эволюционных алгоритмов для повышения их производительности и эффективности.

Оборудование и программное обеспечение

- Java JDK версии 1.8 и выше
- Watchmaker framework версии 0.7.1 (https://github.com/dwdyer/watchmaker)
- Шаблон проекта: https://gitlab.com/itmo_ec_labs/lab5

Ограничения в пределах работы

- Область определения всех переменных целевой функции: [-5, 5]
- Область определения целевой функции [0, 10]
- Заведомо известно, что глобальный оптимум = 10.

Инициализация

В классе **MyFactory** инициализация выполняется с помощью генерации случайных чисел в интервале [0,1] типа double методом random.nextDouble() с последующей репараметриазцией.

Кроссовер

В рамках настоящей работы был реализован арифметический кроссовер:

$$z_i = ax_i + (a-1)y_i$$

Метод **mate** класса **MyCrossover** на вход принимает пару родительских точек а так же числе точек, для которых производится операция (а так е случайную компоненту для выбора партисипантов). Значение **a** в настоящем отчете равно **0.5**;

Мутация

В рамках работы реализована равномерная мутация.

Результаты

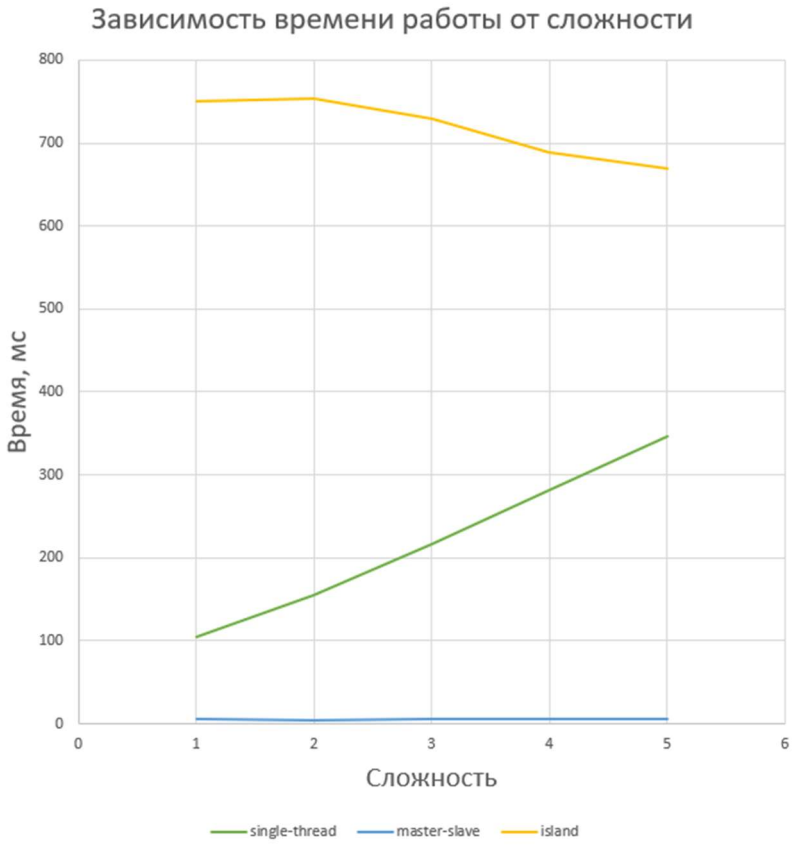
В таблице среднее по 10 запускам.

Во всех экспериментах были выставлены следующие параметры:

- число генераций: 100
- параметр арифметического кроссовера альфа: 0.5
- вероятность мутации: 0.05
- миграции между островами: 2

algorithm	complexity	time (ms)	result

single-thread	1	104	5,211
single-thread	2	155	5,163
single-thread	3	217	5,169
single-thread	4	282	5,155
single-thread	5	347	5,013
master-slave	1	44	5,138
master-slave	2	43	5,033
master-slave	3	51	5,175
master-slave	4	53	5,218
master-slave	5	57	5,099
island	1	751	5,522
island	2	753	5,675
island	3	729	5,608
island	4	689	5,704
island	5	670	5,596



Ответы на вопросы

1. Какая модель лучше при каких условиях?

Однопоточное выполнение - наиболее эффективное решение в случае, когда вычисление фитнес-функции не является самой сложной частью алгоритма. Однако, если вычисление

фитнес-функции требует большого объема вычислений, то параллельные островные алгоритмы и master-slave подход могут работать быстрее однопоточного. В большинстве случаев наилучшее значение фитнес-функции достигается с использованием островных алгоритмов, при этом master-slave обычно проявляет более высокую производительность.

2. Как повлияет увеличение размерности проблемы на алгоритмы?

Значение фитнесс-функции станет меньше, время выполнения вырастет. Ожидается, что наиболее сильно пострадает островной алгоритм.

3. Как повлияет увеличение размера популяции?

Увеличение размера популяции приведет к более высоким результатам (bestFit), но увеличит время, требуемое на выполнение.

4. Есть ли ограничения для количества островов?

Поскольку популяции на островах отделяются от общей популяции, увеличение количества островов быстро становится нецелесообразным из-за слишком маленьких популяций и накладных расходов.