Отчёт о результатах измерения производительности программы

Цель работы

Провести замеры производительности решения задачи подсчёта количества путей в данной графовой базе данных, отвечающих данному регулярному выражению.

Основные сведения

Исходные данные: набор баз данных refinedDataForRPQ, базы данных LUBM300, LUBM500, LUBM1M, LUBM1.5M, LUBM1.9M, каждая с набором из 280 запросов в виде регулярного выражения.

Оборудование:

* процессор: Intel(R) Core(TM) i3-8130U CPU @ 2.20GHz 2.21HGz
* оперативная память: 4793 МБ
* операционная система: Ubuntu 18.04.3 x64

Детали тестирования:

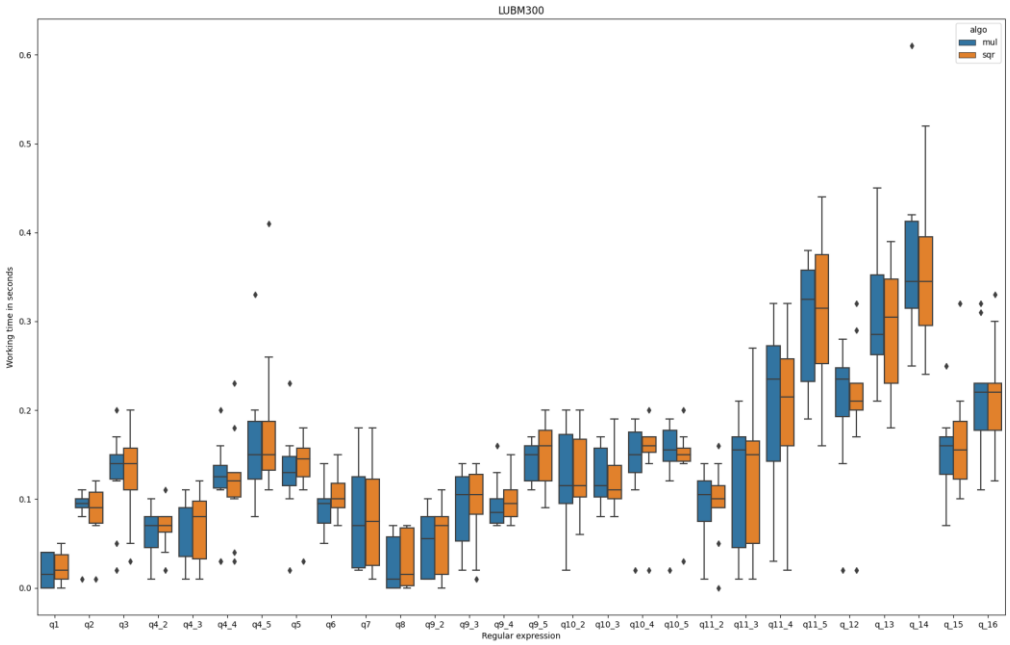
* тестирование проходило через утилиту `pytest`
* тестирование проходило для алгоритма подсчёта количества путей без их вывода
* тестировались 2 разные реализации данного алгоритма: через транзитивное замыкание посредством возведения в квадрат и посредством умножения на матрицу достижимости по одному ребру
* каждая база данных тестировалась по одной чтобы сэкономить оперативную память

Последние 2 пункта явно выделены комментариями в коде в файлах src/request.py и main_test.py соответственно.

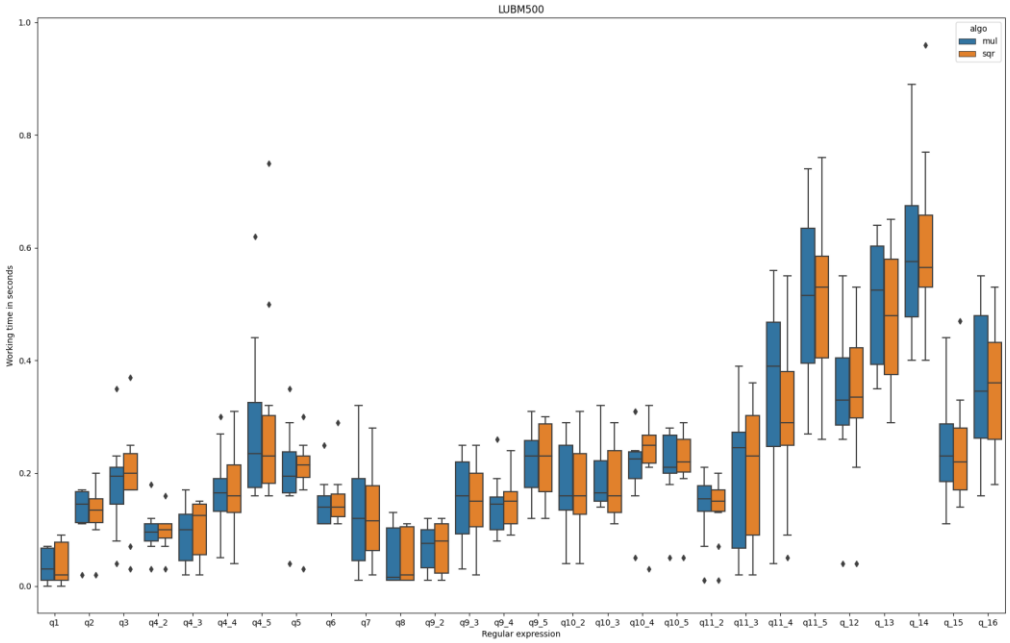
Результаты тестирования

Ниже приведены результаты измерения работы 2 реализаций алгоритма на 5 различных графовых базах данных. Для каждой базы данных были протестированы 28 различных наборов регулярных выражений, по 10 однотипных регулярных выражений в каждом. На графике представлены результаты замеров для реализаций через умножение на матрицу (mul) и возведение матрицы в квадрат (sqr).

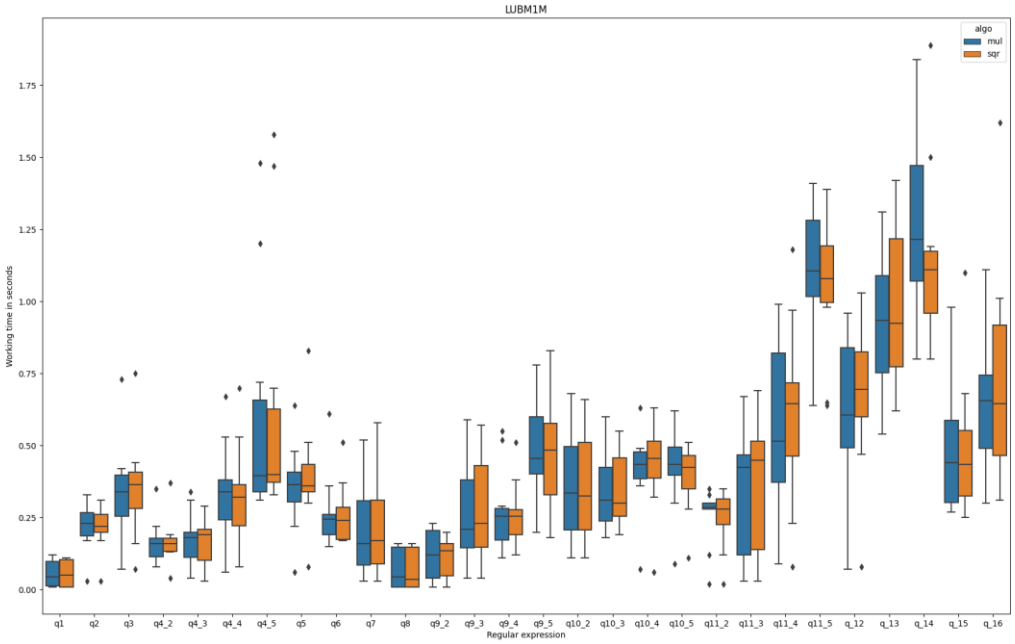
LUBM300



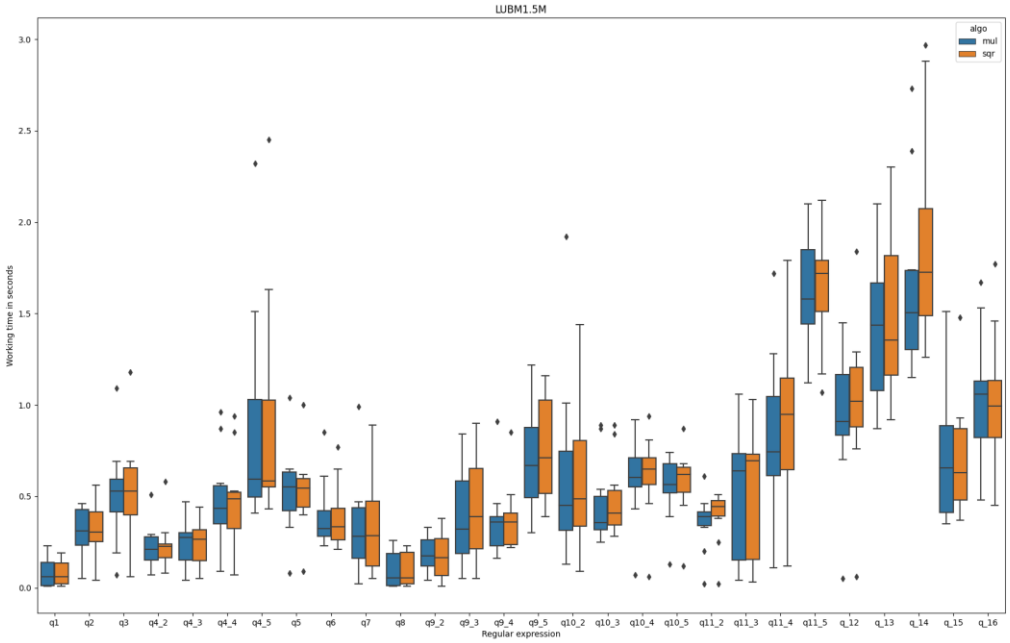
LUBM500



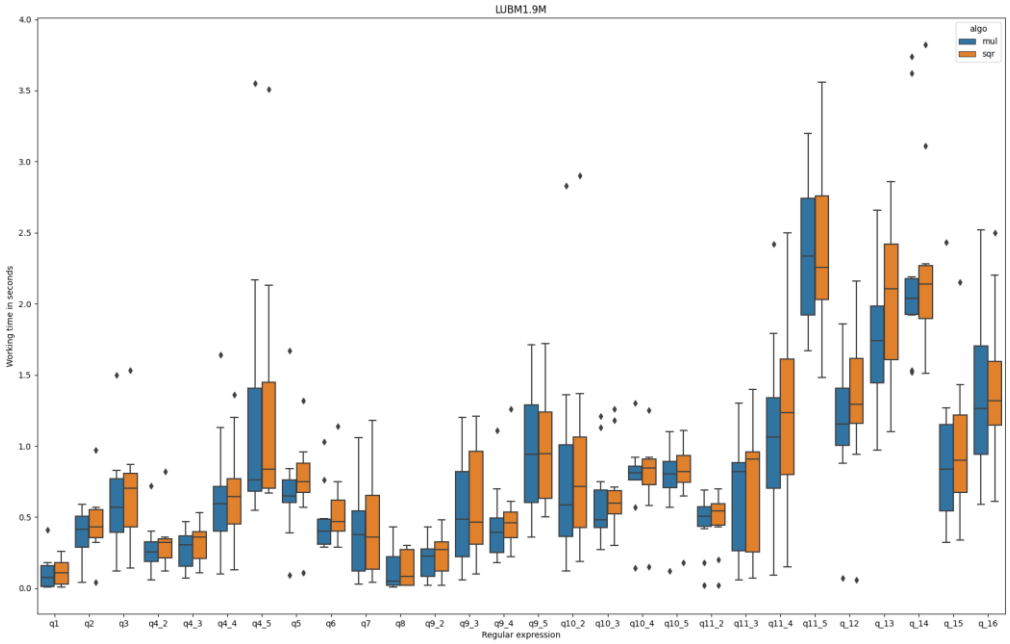
LUBM1M



LUBM1.5M



LUBM1.9M



Вывод

Я не выявил какой-либо существенной разницы между 2 реализациями данного алгоритма.