Кузнецова Маргарита

Б01-908б

https://github.com/Margarita-Kuznetsova/Graph\_algorithm

**Реализация алгоритма Фидуччиа–Мэтьюза**

Цель:

Одним из способов декомпозиции задачи большой размерности на подзадачи является представление ее в виде графа или гиперграфа и последующее разбиение на приблизительно равные части, причем число связей между подграфами должно быть минимальным. Для каждой вершины присвоим переменную Gain, обозначающую, со сколькими другими вершинами связана исходная. Чтобы минимизировать число разрезанных ребер, «сильно» связанные вершины поместим по одну сторону. Но при этом введем критерий, ограничивающий перемещение всех вершин в одну сторону - балансировка. Каждый новый разрез рождает новую итерацию.

Реализация:

Граф представлен в виде вектора. Новый разрез описывается набором флагов, положением вершин и gain’ом. Gain -контейнер хранит в себе наборы вершин с отсортированным gain’ом каждой.

Модификация:

Введен массив флагов, отслеживающих перемещенные вершины. Для каждой вершины просчитывается вклад в стоимость всего разделения. Быстрее происходит обработка вершин как последовательность итеративная. Также вершины после итерации (с изменённым gain’ом) помещаются в конец списка (push\_back).

Результаты:

В таблицах ниже результаты для основного алгоритма и для модифицированного. Графики показывают, что модифицированный алгоритм даёт лучший разрез, причём за меньшее время. Количество итераций не изменилось. Но общее время работы сократилось.

Выводы:

В работе реализован алгоритм Фидуччиа–Мэтьюза. С помощью модификаций показано, что производительность повысилась за счет введения дополнительных структур контроля.

Таблица 1. Данные для оригинальной реализации

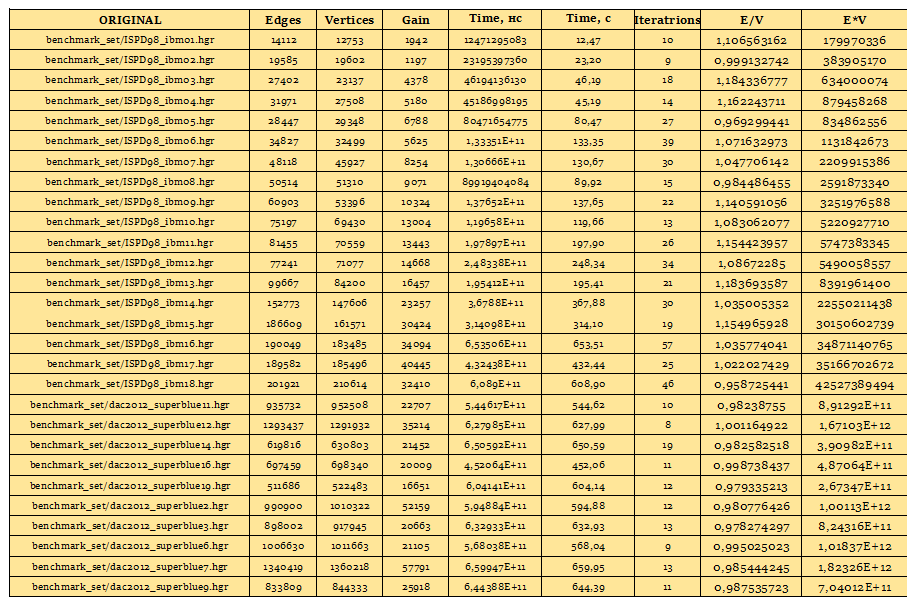


Таблица 2. Данные для модифицированной реализации

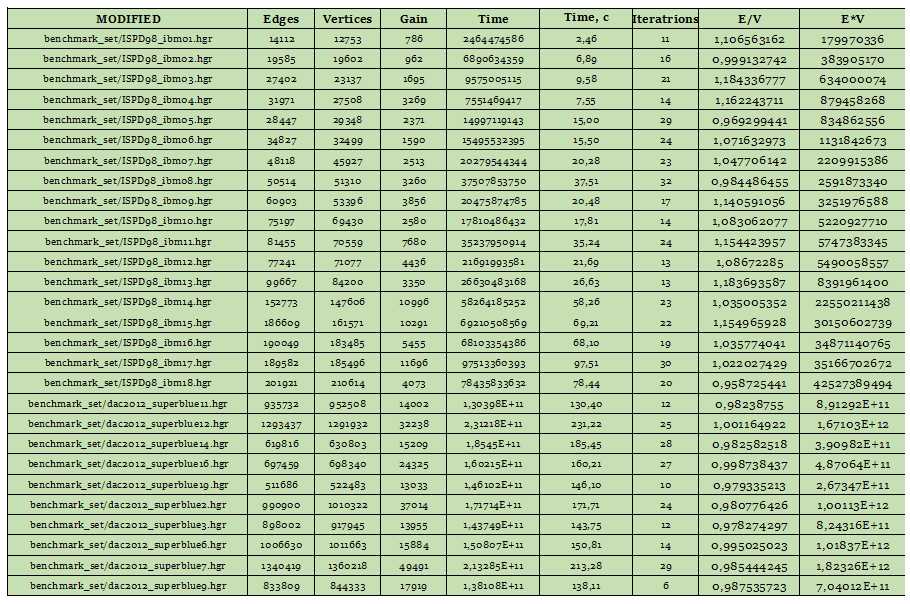


График 1. Зависимость времени от числа вершин

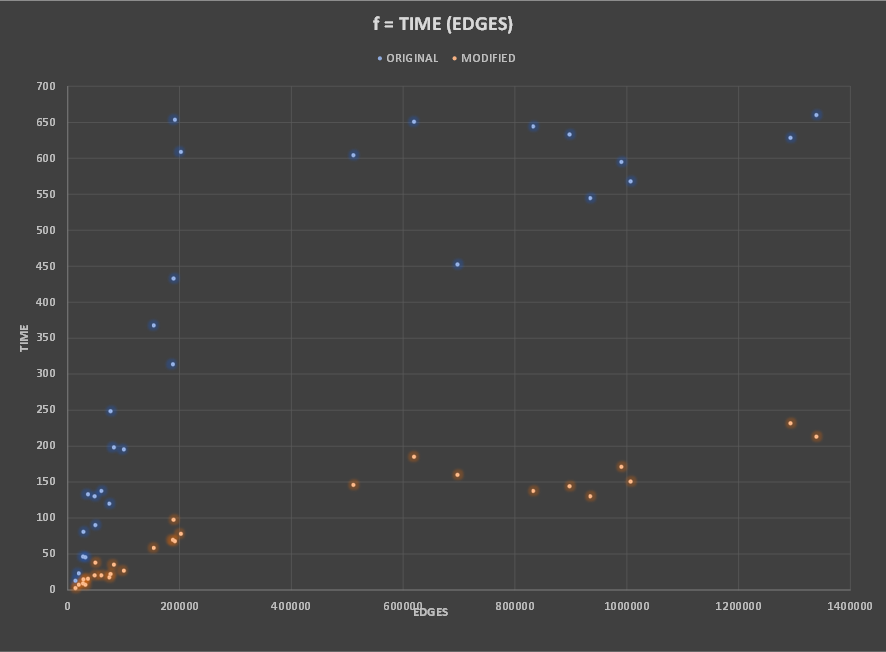


График 2. Зависимость времени от числа ребер

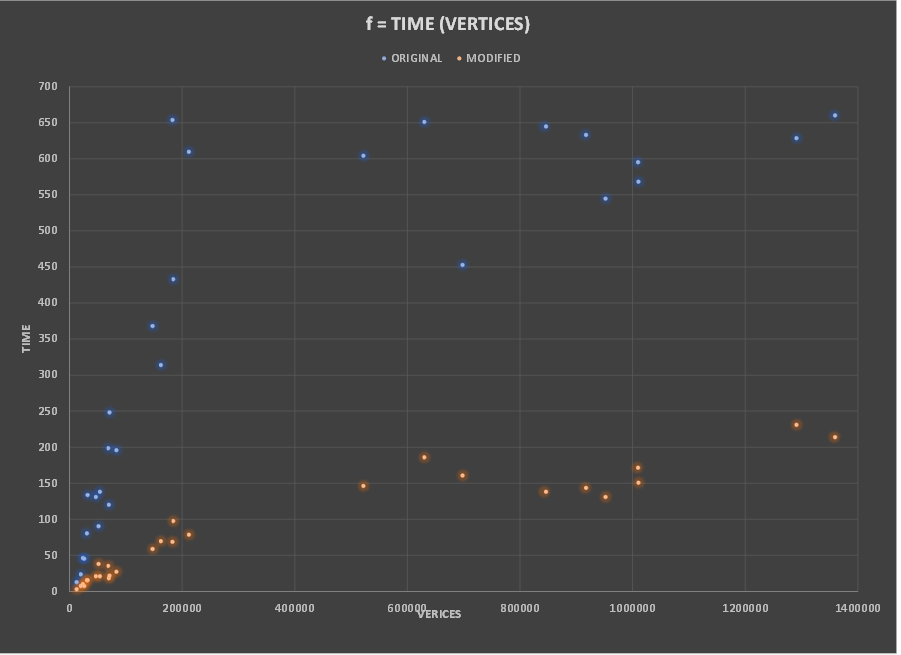


График 3. Зависимость gain’а от тестов

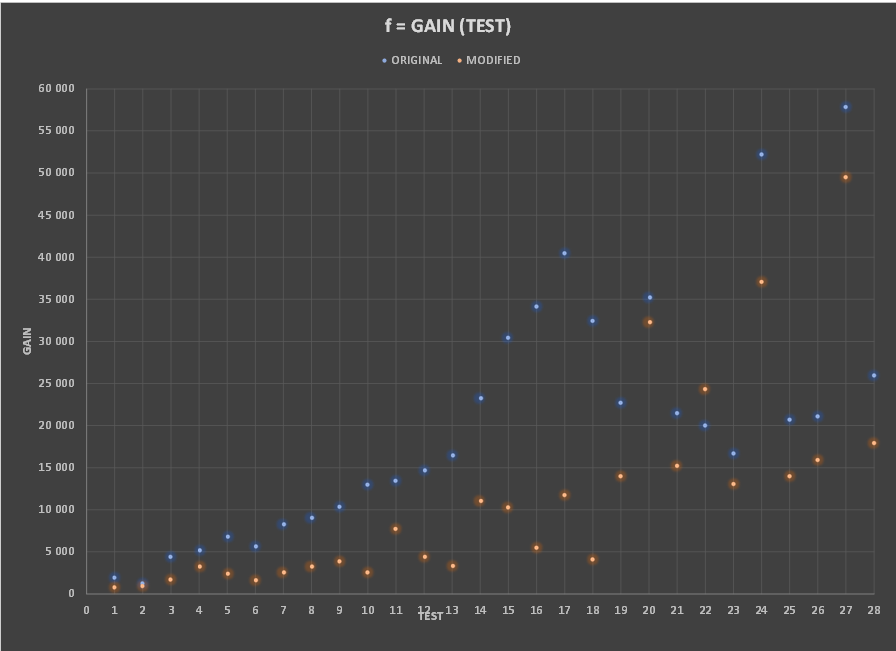


График 4. Зависимость времени от тестов

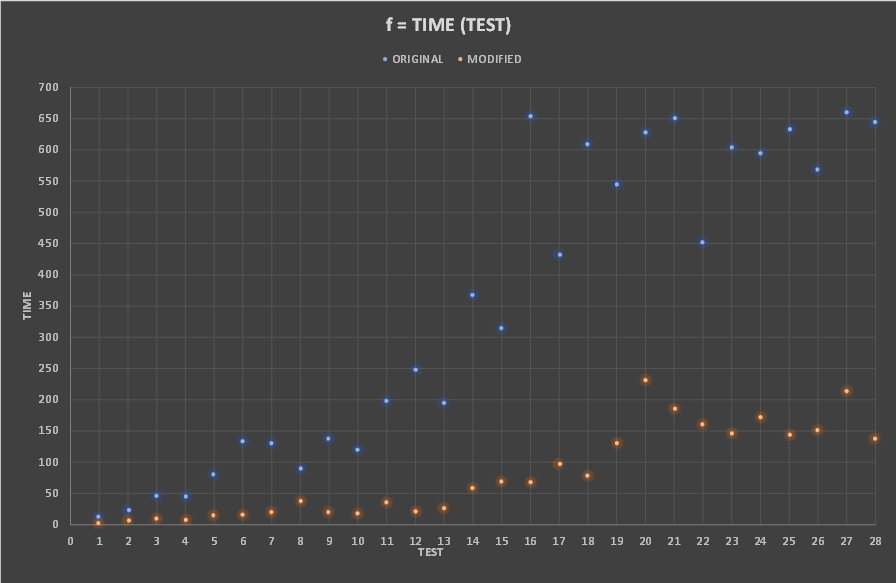


График 5. Зависимость количества итераций от тестов

