***Градиентный спуск*** - Чаще всего алгоритм выдавал неоптимальное решение. Неплохо работает на графах с небольшой плотностью, но при более сложных графах в качестве результата выводит некоторый путь, не являющийся наибольшим.

***Метрополис*** - Алгоритм работает точнее алгоритма Градиентного спуска, но выдает неправильный ответ чаще алгоритма с отжигом. Экспериментальным путем получилось, что алгоритм работает лучше при температурном значении 25-30. При ее увеличении, участились случаи, когда алгоритм «выскакивает» из глобального минимума. Выдает лучший результат на графах, с бОльшей плотностью.

***Метрополис/С отжигом*** - Алгоритм работает с наибольшей частотой получения оптимального результата. В качестве температурного диапазона было протестировано 2 варианта : диапазон от 100000 до 1 и от 50000 до 1. Проверив результат работы на некоторых различных графах, получилось, что алгоритм работает лучше, когда установлен первый температурный диапазоном. В этом случае бОльшее количество итераций позволяет лучше «проходить» точки локального минимума, увеличивая частоту попаданий в глобальный минимум по сравнению с запусками при втором температурном диапазоне. Также как и предыдущий, алгоритм работает точнее при более плотных графах.

Алгоритм был запущен на нескольких звездчатых графах с разным количеством вершин (5-15, 30-50) и на обычных графах с разной плотностью (0.1, 0.5, 0.9).

Пример запуска Алгоритмов на примере графа на 10 вершинах с 17 ребрами:  
10

17

0 2

2 6

6 3

6 8

0 3

0 4

1 3

1 5

3 5

4 3

4 5

5 8

8 9

5 7

4 9

4 7

1 9

Результаты работы:

Отжиг - 4 9 8 6 2 0 3 1 5 7

Метрополис - 4 9 8 6 2 0 3 1 5 7

Градиентный спуск - 9 1 3 5 7 4 0 2 6 8

Постепенно увеличивая количество ребер, получаем более наглядный результат точности работы алгоритмов. Так на 30 ребрах Отжиг дает длину 15, Метрополис - 13, Градиентный спуск - 10.