Соболева Наталья, 152-1

Домашняя работа 1
Часть 3

Реализация алгоритмов градиентного спуска и алгоритма метрополиса с отжигом и без, в применении к решению задачи нахождения в графе простого пути наибольшей длины.

Немного о том, как реализован класс, который хранит в себе простой пусть наименьшей длины:

Класс *LongestSimplePath* задает множество вершин — определяющих искомый простой путь в графе. При применении к данному классу алгоритмов градиентного спуска и метрополиса (с отжигом и без отжига) множество вершин сдвигается в соседнее решение.

Определение соседства

Соседним решением является множество, к которому:

а) добавили вершину; добавленная вершина должна быть соседом либо вершины — начала пути ($start$), либо вершины — конца пути ($finish$);

б) удалили вершину; удалить можно только стартовую или конечную вершину ($start$ or $finish$); стартовой или конечной вершиной тогда становятся их соседи.

Gradient Descent

Начинаем строить наибольший простой путь с одной случайной вершины (изначально она и стартовая и конечная вершина). С некой вероятностью добавляем новую вершину в начало или в конец уже имеющегося пути (набора вершин). Процесс остановится, если станет невозможным добавление новой вершины.

Metropolis

Так же начинаем с одной случайной вершины. По-прежнему с некоторой случайной вероятностью происходит добавление, но, если это добавление не произошло, то с вероятностью, которая вычисляется при помощи формулы Больцано, происходит удаление вершины из пути — это обеспечивает нам возможный выход из зоны локального максимума.

Отличие алгоритма с отжигом и без отжига в том, что в алгоритме с отжигом температура T уменьшается с увеличением номера итерации — тем самым позволяя избежать случая постоянного "выхода"из локальных

или глобального максимума.

Примечание

Чем больше количество итераций и попыток вызвать функции градиентного спуска и метрополиса, тем больше вероятность попадания в оптимум.

Проверка решения на работоспособность

Первое, проверим работоспособность алгоритма для нескольких, случайно сгенерированных, графоф, для наглядности — с небольшим количеством вершин.

а) *Случайный граф на 10 вершинах, с вероятностью ребра* 0.5

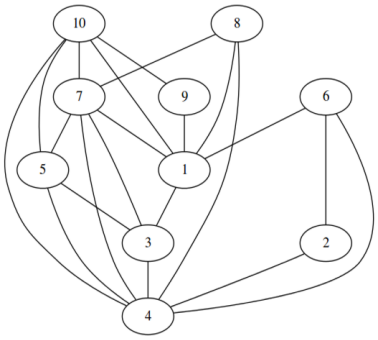

Рис. 1: Случайный граф со средним количеством ребер

Для данного примера все три алгоритма нашли один и тот же набор вершин (рис. 2, рис. 3), входящих в самый длинный простой путь. Все три алгоритма нашли оптимальное решение.

б) *Случайный граф на 10 вершинах, с вероятностью ребра* 0.1 *(рис. 4)*

Для данного примера все три алгоритма нашли простой путь одной и той же длины, но набор вершин отличается (рис.5, рис.6). То есть можно увидеть, что алгоритм метрополиса нашел соседнее решение, которое все еще является оптимальным.

в) *Star Graph, 7 вершин (рис.7)*

Помимо случайных графов маленьких размеров рассмотрим так же граф — звезду, где одна вершина соединена со всеми остальными.

Все три алгоритма нашли различные пути в графе (рис.8, рис.9, рис.10), но все пути являются оптимальными решениями для исходного графа.

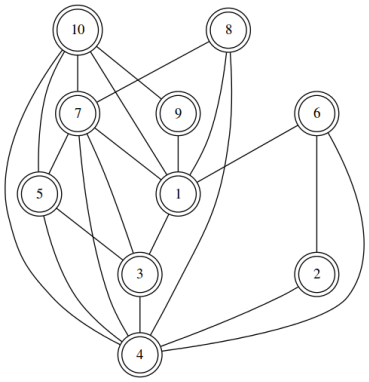

Рис. 2: Gradient Descent

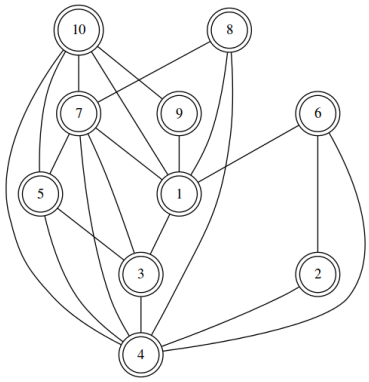

Рис. 3: Metropolis(annealing and not annealing)

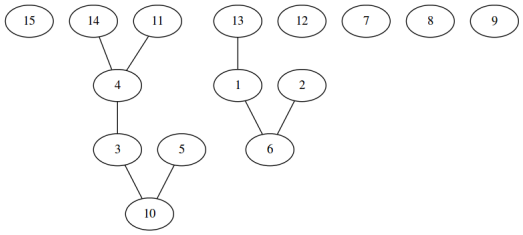

Рис. 4: Случайный разреженный граф

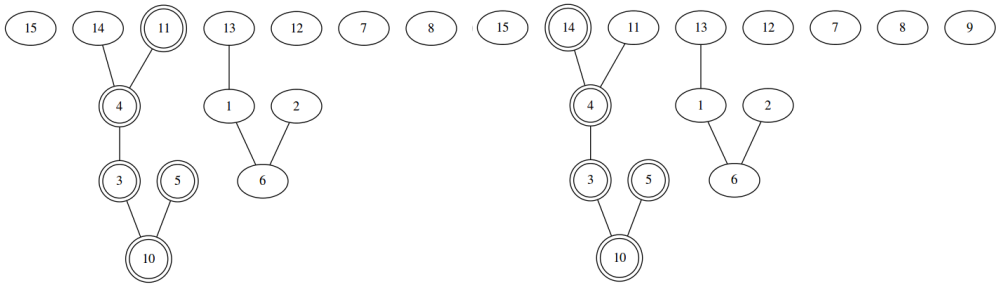

Рис. 5: Gradient Descent

Рис. 6: Metropolis(annealing and not annealing)

г) *Рассмотрим граф с большим количеством вершин (рис. 11)*

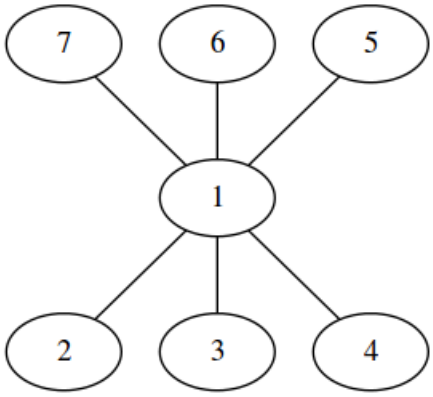

Рис. 7: Star Graph, 7 вершин

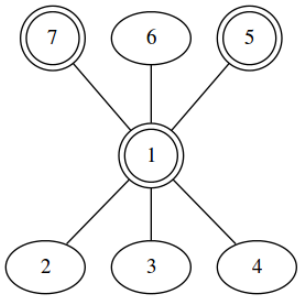

Рис. 8: Gradient Descent

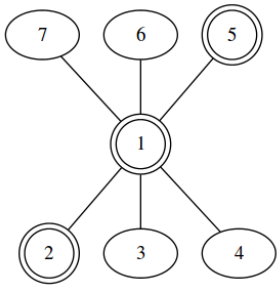

Рис. 9: Metropolis

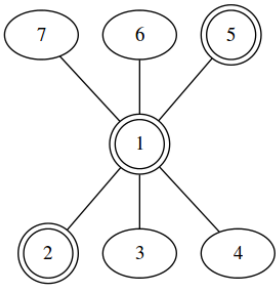

Рис. 10: Metropolis(annealing)

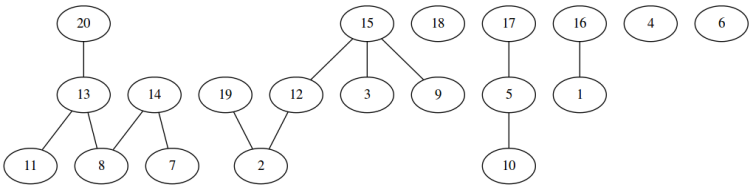

Рис. 11: Разреженный граф, 20 вершин

Все три алгоритма дли разные решения, но все эти решения являются оптимальными (рис. 12, рис. 13, рис. 14).

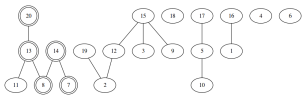

Рис. 12: Gradient Descent

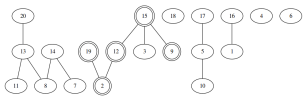

Рис. 13: Metropolis

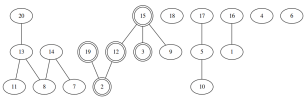

Рис. 14: Metropolis(annealing)

Сравнение алгоритмов *Gradient Descent, Metropolis with annealing and without*:

Так как на графах относительно небольшого размера при числе итераций = 100, алгоритмы практически всегда находит оптимальное решение, то сравним длины лучших путей алгоритмов на графах с большим количеством вершин. Создадим таблицу, где для нескольких графов приведем сравнение длин максимальных путей, полученных различными алгоритмамию.

$Number\&Freq$	Gradient Descent	Metropolis	Metropolis(annealing)
30&0.2	30	29	30
50&0.4	50	49	50
70&0.04	28	30	30
100&0.1	90	79	83
100&0.5	100	97	100

На различных графах некоторые алгоритмы оказываются точнее других, но все значения очень близки между собой.