**АЛГОРИТМ А\***

Для понимания этого алгоритма необходимо знание об алгоритме поиска в ширину и алгоритме Дейкстры. Неразумно изучать его, не зная эти два алгоритма.

Итак.

**Поиск в ширину.**

Предположим, у вас есть карта городов России и дорог между ними. Ваша задача определить кратчайшее расстояние между Нижним Новгородом и Новосибирском. В этой задаче вам и сможет помочь поиск в ширину. Он уже реализован на Яндекс-картах, и вы можете наблюдать, как рендерится ваш маршрут в Новосибирск.

Работает этот алгоритм так. Ищет ближайший круг городов от Нижнего Новгорода. Каждому найденному городу задает вопрос: «ты Новосибирск?». Если да, то алгоритм прекращают работу. Если нет, алгоритм ищет ближайший круг городов уже того города, которому задал этот вопрос. И так, пока не дойдет до Новосибирска.

Принцип прост, однако есть в этой простоте подводные камни. Допустим, этот алгоритм обратился ко одному из ближайших городов Нижнего Новгорода. Получил отрицательный ответ. И пошел проверять уже ближайшие города того города, к которому только что обращался. Получается алгоритм игнорирует другие ближайшие Нижнему Новгороду города, а ведь вполне вероятно, что именно среди них и есть нужный нам город. Чтобы этого не происходило, будем сначала проходить ближайшие города для Новгорода. Потом ближайшие города этих городов. Потом ближайшие ближайших и т.д.

Еще один нюанс. Чтобы по стечению обстоятельств несколько раз не спрашивать один и тот же город, нужно помечать, спрашивали ли мы его или нет.

Вот и все.

**Алгоритм Дейкстра.**

У поиска в ширину в нашем примере есть один очень существенный изъян. Он не учитывает ни качество, ни длину дороги. Соответственно, если мы воспользуемся только поиском в ширину, мы можем вообще не доехать до Новосибирска (не хватит бензина, в болоте утонем). Поэтому определим для каждой дороги в России качество и длину одним каким-нибудь числом. Если говорить языком математики, то у нас получится взвешенный граф (в предыдущем случае был невзвешенный) от слова вес.

Итак, как работает алгоритм Дейкстра.

1. Так как мы находимся в Новгороде и еще даже не сдвинулись с места, пометим этот город меткой 0. Остальные города пометим метками «бесконечность», так как мы в данный момент мы в абсолютном неведении, сколько нам до них ехать.
2. Узнаем круг ближайших до Новгорода городов и присваиваем им метки, которые мы до этого понаставляли на дорогах.
3. Теперь двигаемся в город с минимальной меткой (минимальным числом, в данном случае число обозначает, что на этой дороге у нас минимальная вероятность затонуть в болоте).
4. Узнаем круг ближайших городов для этого города и узнаем метки, уже существующие на них. Если при сложении метки города, в котором мы находимся, и метки дороги, которая ведет к следующему городу получается меньшее число, чем существующая метка, мы присваиваем это число в качестве метки следующему городу.
5. И т. д., пока не останется городов с метками «бесконечность».

Так мы пока что лишь расставляем метки и ничего больше. Но нам надо ким-то образом найти кратчайший маршрут.

Делается это так.

Если при переходе из одного города в другой, в городе, куда мы переходим изменилась метка, мы говорим, что город, из которого мы выходим, является родителем города, в который мы идем. Так, проделав это с несколькими городами мы заносим в какую-нибудь таблицу кто кому родитель. Так проходим всю карту.

Теперь берем Новосибирск. Спрашиваем у него, кто у него родитель. Потом спрашиваем у родителя, кто у него родитель и т. д., пока не дойдем до Нижнего Новгорода. Все. Маршрут построен.

Стоит еще оговорится, что алгоритм Дейкстры работает только для положительный меток или весов. Как говорит интернет, это можно обойти с помощью алгоритма Беллмана-Форда.

**А\***

Как вы могли уже догадаться, у алгоритма Дейкстры тоже есть минусы. Этот алгоритм ищет во всех направлениях, что не будет любой нормальный человек. Зачем искать город там, где его нет? Именно для этого и есть алгоритм А\*.

А\* похож на жадный алгоритм, однако в отличие от жадного в состоянии найти кратчайший путь.

Чтобы получить А\* нужно немного преобразовать алгоритм Дейкстра. Кроме меток мы будем еще высчитывать примерное расстояние от конкретного города до конечной цели. То есть просто к метке города добавим примерное расстояние от этого города до Новосибирска. Высчитывать расстояние будем по самой обыкновенной теореме Пифагора. Также добавим к алгоритму Дейкстры условие, что, если метка больше, допустим 3500, мы туда уже не идем. Так мы значительно ограничим границы поиска.