Кратчайшие расстояния в графе. В большинстве задач мы считаем, что циклов отрицательного веса нет, оптимальный путь простой, а на кратчайшие расстояния накладывается неравенство треугольника.

Алгоритм Форда-Беллмана. Задача — найти кратчайшие расстояния из s до всех вершин графа. Мы хотим на каждом шаге перебирать все ребра, и поддерживать инвариант — для шага i мы считаем, что алгоритм нашел кратчайшие расстояния среди всех путей длины i. Тогда на следующем шаге мы рассмотрим все ребрра, и продолжим старые пути новыми ребрами, сделаем релаксации.

Циклы отрицательного веса. Заметим, что наш алгоритм не будет делать релаксации после шага n, потому что больше будет нечего релаксировать. Но в случае с циклами отрицательного веса это не так — мы знаем, что на каждом шаге после n-го мы будем делать релаксацию, проходя по улучшающему ответ циклу. Тогда, если мы для каждой релаксации запомнили, какая вершина улучшала наш текущий ответ, мы можем восстановить цикл (или оптимальный путь), просто проходясь по предкам.

Алгоритм Флойда. Задача — найти матрицу кратчайших расстояний. Пусть $f_k(u, v)$ — это кратчайшее расстояние между u и v, если в качестве промежуточных вершин разрешено использовать вершины $1, \ldots, k$. Тогда при переходе $f_k(u, v) \to f_{k+1}(u, v)$ мы хотим сделать релаксацию пути как $u \to k + k \to v$.

Алгоритм Дейкстры. В реальных задачах редко присутствуют ребра отрицательного веса, и в таких ситуациях часто используется алгоритм Дейкстры. Инвариант алгоритма — известны кратчайшие расстояния от s до вершин множества A. Мы знаем, что $\max\ p(v) \le \min\ p(u),\ v \in A,\ u \in V \setminus A$. Давайте для всех вершин поддерживать какое-то текущее $p(v)$. Понятно, что для вершин из A это будет итоговым ответом. Утверждается, что в A можно добавить вершину с минимальным текущим $p(v)$. Доказательство от противного — если текущее расстояние до v было не кратчайшим, то в v был путь, проходящий через вершину $u \in V \setminus A$, но тогда $p(u) < p(v)$, противоречие.

Тогда мы хотим добавлять вершины в A по очереди, по ходу обновляя значения $p(v)$ для всех соседей добавляемой вершиной. Тогда нам нужна структура данных, которая сделает n извлечений минимума и m уменьшений ключа. Оптимальная структура (асимптотически!) — фибоначчиева куча, с ней алгоритм работает за $O(n \log n + m)$.

Потенциалы. Создадим новый граф, где каждой вершине зададим потенциал c_v. Теперь за вес ребра примем $w_c(u, v) = w(u, v) + c_v - c_u$. То есть, вес ребра плюс разность потенциалов. Тогда длина пути $s \to t$ в новом графе определяется как длина в старом графе с дополнительным слагаемым $c_t - c_s = \text{const}$.

Задача о размещении потенциалов эквивалентна поиску отрицательного цикла. То есть, если цикл отрицательного веса есть, то не существует потенциалов, для которых выполняется неравенство треугольника (например, вес такого цикла в новом графе будет меньше нуля). Если же циклов нет, то сделаем $c_v = p(s, v)$. Тогда $w_p(u, v) = w(u, v) - p(v) + p(u)$, но $p(v) \le p(u) + w(u, v)$, а тогда вес новых ребер положителен, а неравенство треугольника выполнено.

A-star. Пусть мы хотим решить задачу на плоскости, при этом хочется не посещать заведомо бесполезные состояния. Введем какую-то метрику $d(u, v)$, в качестве которой введем евклидово расстояние. Теперь мы вытаскиваем новые вершины по минимуму $p(u) + w(u, v) + d(v, t)$ (то есть минимальная сумма расстояния и оценки на метрику). Здесь t - это вершина, кратчайший путь в которую мы пытаемся найти. Поскольку на метрику введено неравенство треугольника, то наша новая Дейкстра все еще работает.

Двусторонняя Дейкстра. Аналогично обычной Дейкстре, запускаем алгоритм в две стороны, и обновляем ответ, если видим ребро $u \leftrightarrow v,\ v \in B,\ u \in A$.