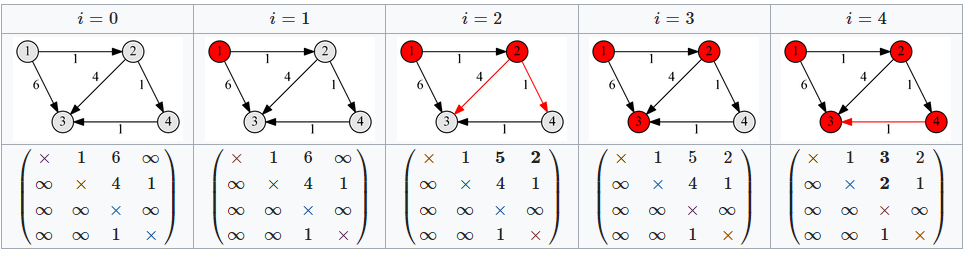
**Алгоритм Флойда (I can’t breathe)**

**Алгоритм Флойда (алгоритм Флойда–Уоршелла)** — алгоритм нахождения длин кратчайших путей между всеми парами вершин во взвешенном ориентированном графе. Работает корректно, если в графе нет циклов отрицательной величины, а в случае, когда такой цикл есть, позволяет найти хотя бы один такой цикл. Алгоритм работает за Θ(n3)Θ(n3) времени и использует Θ(n2)Θ(n2) памяти. Разработан в 1962 году.

Если надо определить наименьшую кратчайший путь (стоимость) между всеми вершинами, то для решения этой задачи можно использовать алгоритм Флойда.

основе алгоритма лежат два свойства кратчайшего пути графа.   
Пример работы:  
  
Первое:  
Имеется кратчайший путь p1k=(v1,v2,… ,vk) от вершины v1 до вершины vk, а также его подпуть p'(vi,vi+1,… ,vj), при этом действует 1 <= i <= j <= k.

Если p — кратчайший путь от v1 до vk, то p' также является кратчайшим путем от вершины vi до vj  
Это можно легко доказать, так как стоимость пути p складывается из стоимости пути p' и стоимости остальных его частей. Так вот представив что есть более короткий путь p', мы уменьшим эту сумму, что приведет к противоречию с утверждением, что эта сумма и так уже была минимальной.  
Второе свойство является основой алгоритма. Мы рассматриваем граф G с пронумерованными от 1 до n вершинами {v1,v2,… ,vn} и путь pij от vi до vj, проходящий через определенное множество разрешенных вершин, ограниченное индексом k.  
То есть если k=0, то мы рассматриваем прямые соединения вершин друг с другом, так как множество разрешенных промежуточных вершин рано нулю. Если k=1 — мы рассматриваем пути, проходящие через вершину v1, при k=2 — через вершины {v1, v2}, при k=3 — {v1, v3, v3} и так далее.

Анализ времени работы и использования памяти

Алгоритму требуется O(n3) памяти, для сохранения матриц. Однако количество матриц можно легко сократить до двух, каждый раз переписывая ненужную матрицу или вообще перейти к двухмерной матрице, убрав индекс k у dkij. Лучший вариант, который чаще всего используется — писать сразу в матрицу смежности, тогда нам совсем не нужна дополнительная память, правда если сразу переписывать изначальную матрицу, то нужно дополнительно показать корректность алгоритма, так как классическое академическоле доказательство верно только для случая, когда матрица предыдущей итерации не изменяется.  
  
Что касается времени работы — три вложенных цикла от 1 до n — Θ(n3).

Случай отрицательных циклов  
Если в графе есть циклы отрицательного веса, то формально алгоритм Флойда-Уоршелла к такому графу неприменим. Но на самом деле алгоритм корректно сработает для всех пар, пути мужду которыми никогда не проходят через цикл негативной стоимости, а для остальных мы получим какие-нибудь числа, возможно сильно отрицательные. Алгоритм можно научить выводить для таких пар некое значение, соответствующее -∞  
  
Кстати после отработки такого графа на диагонале матрицы кратчайших путей возникнут отрицательные числа — кратчайшее расстояние от вершины в этом цикле до неё самой будет меньше нуля, что соответствует проходу по этому циклу, так что алгоритм можно использовать для определения наличия отрицательных циклов в графе.

Как и любой базовый алгоритм, алгоритм Флойда — Уоршелла используется очень широко и много где, начиная от поиска транзитивного замыкания графа, заканчивая генетикой и управлением проектами. Но первое что приходит в голову конечно же транспортные и всякие другие сети.