Алгоритм Дейкстры

Алгоритм Дейкстры является предпочтительным методом поиска кратчайших путей в графах со взвешенными ребрами(вершинами). Алгоритм находит кратчайшие пути от заданной вершины S ко всем другим вершинам графа, включая требуемую конечную вершину t.

Допустим, что кратчайший путь от вершины s к вершине t графа проходит через определенную промежуточную вершину x. Очевидно, что путь должен содержать кратчайший путь от вершины sдо вершины x, в качестве префикса, ибо в противном случае можно было бы сократить путь s-t, используя более короткий префиксный путь s-x. Таким образом, прежде чем найти кратчайший путь от начальной вершины s к конечной вершине t, нам нужно найти кротчайший путь от начальной вершины s к промежуточной вершине x.

Алгоритм Дейкстры работает поэтапно, находя на каждом этапе кратчайший путь от вершины s к некоторой новой вершине. Говоря конкретно, вершина X такова, что сумма dist(s,vi)+w(v1,x) минимальна для всех необработанных 1<=i<=n, где w(I,j)-длина ребра между вершинами I и j, а dist(I, j) длина кратчайшего пути между ними.

Здесь напрашивается стратегия, аналогичная ДП. Кратчайший путь от вершины s к самой себе является тривиальным, при условии отсутствия ребе с отрицательным весом, поэтому dist(s,s)=0. Если (s,y) является самым легким ребром, входящим в вершину s, то это означает, что dist(s,y)=w(s,y). Определив кратчайший путь к вершине x, мы проверяем все исходящие из нее реба, чтобы узнать, не существует ли лучшего пути из начальной вершины s к какой-нибудь неизвестной вершине через вершину x.   
рассмотрим псевдокод данного алгоритма.

1. ShortestPath-Dijkstra(G,s,t)
2. known = {s}
3. for i = 1 to n, dist[i] = infinity
4. for each adge(s,v), dist[v] = w(s,v)
5. last = s
6. while (last != t)
7. delect Vnext, неизвестная вершина, минимизирующая dist[v]
8. for each adge (Vnext, x), dist[x] = min[dist[x],dist[Vnext]+w(Vnext,s)]
9. last =  Vnext
10. known = known Объединение {Vnext}

в каждом цикле мы добавляем одну вершину к дереву вершин, для которых мы знаем кратчайший путь из вершины s. Мы сохраняем информацию о наилучшем пути на данное время для всех вершин вне дерев и вставляем их в дерево в порядке возрастания веса.