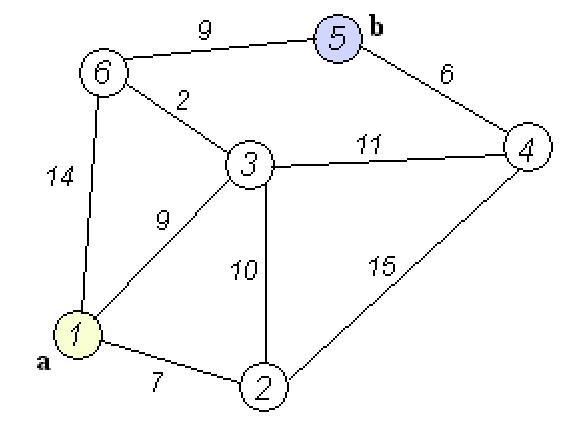
# **每天一个算法——最短路径之Dijkstra算法**

极客码农 2017-04-27 16:51:01

今天为大家分享的算法是为解决最短路径算法的Dijkstra算法（简称D算法），这是一个解决从点到点之间最短路径的问题，看下面这张图：



这里，我们想要得出****节点a****（节点1）到****节点b****（节点5）的最短路径，就是怎么走可以使得权重值的和最小，每一条边都有一个权重。

今天我们介绍的D算法就是解决这类问题的，这是一种贪心算法，每次只取权重和最小的点，通过不断加入节点，来更新源节点a到各个节点的最短路径，直到所有节点遍历完。

算法步骤：

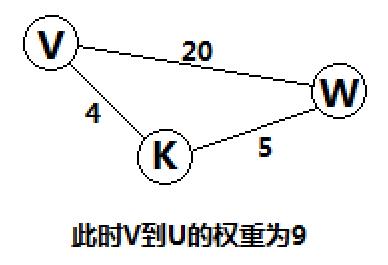
1、定义，遍历过的节点集合为S，集合U为其余节点（即未遍历）。初始时，S只包含源点v，即S＝{v}，v的距离为0。U包含除v外的其他顶点，即:U={其余顶点}。若v与U中顶点u有边，则<u,v>正常有权值，若u不是v的出边邻接点，则<u,v>权值为∞。

注：这里集合S、U，是为了判断哪些节点已经遍历过，如果U为空了，就不继续执行。

2、从集合U中选取一个距离v最小的顶点k，把k加入到S中。

3、以k为新考虑的中间点，修改v到U中各顶点的距离；若从源点v到顶点w的距离（经过顶点k）比原来距离（不经过顶点k）短，则修改v到w的距离值。

例子:

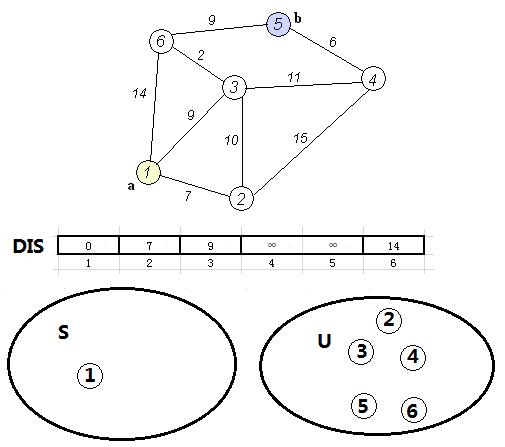


4、重复步骤2、3直到所有顶点都包含在S中。

上面就是D算法的处理步骤，可能大家第一次看和我一样很迷茫，不要紧，我们结合上面这个图，使用D算法来详细介绍每个步骤：

****1、初始化步骤****

用一个一维数组DIS来表示****节点1****到各个节点的最短路径（即权重），没有连线的用∞表示。除此之外，为了防止节点重复计算，我们把节点分成两组，一组已经遍历的节点集合S，另一组还没遍历集合U。初始化的时候，****节点1****在集合S中。



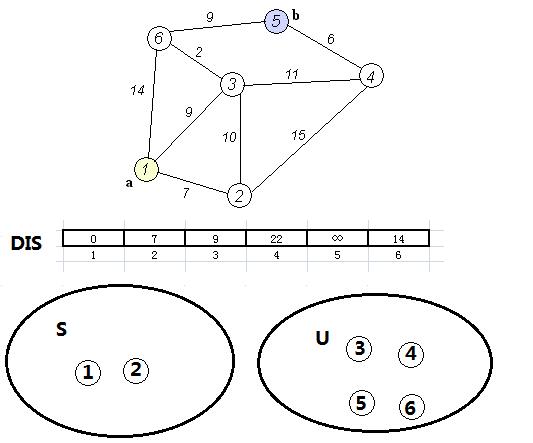
注：节点1到自己的权重为0。

****2、从集合U中获取离节点1最近的点（参考U在数组中的最小值，是节点2），加入到集合S，并重新计算DIS数组。****

（1）节点2有到节点3和4的边，所以数组DIS的3和4位置的值可能会变动。

（2）2到3的权重为10，所以1到3的权重为17（7+10），17大于9，所以不用变动。

（3）3到4的权重为15，所以1到4的权重为22（7+15），22小于无穷，所以DIS[4]=22。（这里数组角标大家注意下，DIS[4]表示第四个位置，起始位置为1）

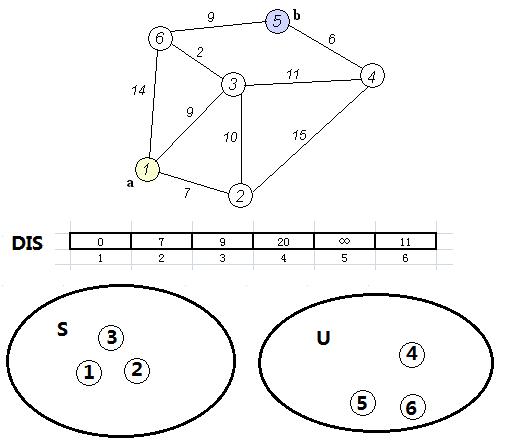


****3、重复步骤2，获取U里面距离最近的点（这里为节点3），重新计算DIS数组。****

（1）节点3这里和4、6有边，所以DIS的位置4、6可能需要修改值。（节点2在S中，只考虑U中没遍历过的节点）。

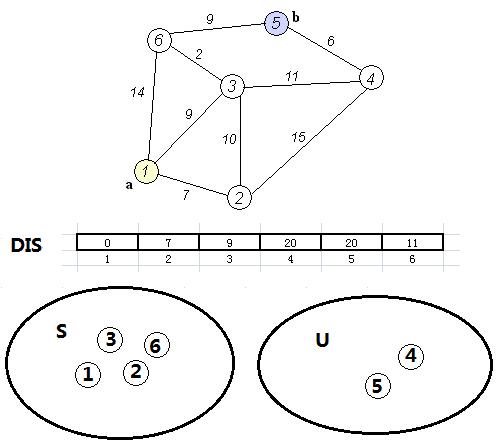
（2）3到4的权重为11，所以1到4的权重为20（9+11），小于22（****DIS[4]****），所以DIS[4]=20。

（3）3到6的权重为2，所以1到6的权重为11（9+2），小于14（****DIS[6]****），所以DIS[6]=11。



****4、重复步骤2，取节点6加入S，重新计算DIS。****

节点6只有到5的边了，所以只修改DIS[6]的值。这里节点1到节点5的权重为20（11+9），小于无穷（DIS[5]），所以DIS[5]=20。



****5、继续重复。****

这次获取到节点4，从DIS数组可以知道1到4的权重（20）已经大于等于1到5的权重（20），所以无论如何也无法从节点4取到权重更小的路径了，所以可以舍弃（****D算法是无法解决负权重问题，所以图的权重必须为正****）。

由于节点1到节点5没有边连接，所以权重为无穷，大于20。所以，算法的最终结果就是：

IMG_262

节点1到节点5的最短路径是20，

顺序是1->3->6->5。

参考https://www.toutiao.com/i6413501458238407169/，增加了回溯路径的代码