

Project2 最短路径

1.概览

Dijkstra(迪杰斯特拉)算法是典型的单源最短路径算法，用于计算一个节点到其他所有节点的最短路径。主要特点是以起始点为中心向外层层扩展，直到扩展到终点为止。

2.算法描述

1)算法思想：设 $G=(V,E)$ 是一个带权有向图，把图中顶点集合 V 分成两组，第一组为已求出最短路径的顶点集合（用 S 表示，初始时 S 中只有一个源点，以后每求得一条最短路径，就将加入到集合 S 中，直到全部顶点都加入到 S 中，算法就结束了），第二组为其余未确定最短路径的顶点集合（用 U 表示），按最短路径长度的递增次序依次把第二组的顶点加入 S 中。在加入的过程中，总保持从源点 v 到 S 中各顶点的最短路径长度不大于从源点 v 到 U 中任何顶点的最短路径长度。此外，每个顶点对应一个距离， S 中的顶点的距离就是从 v 到此顶点的最短路径长度， U 中的顶点的距离，是从 v 到此顶点只包括 S 中的顶点为中间顶点的当前最短路径长度。

(1) 初始时， S 只包含起点 s ； U 包含除 s 外的其他顶点，且 U 中顶点的距离为"起点 s 到该顶点的距离" [例如， U 中顶点 v 的距离为 (s,v) 的长度，然后 s 和 v 不相邻，则 v 的距离为 ∞]。

(2) 从 U 中选出"距离最短的顶点 k "，并将顶点 k 加入到 S 中；同时，从 U 中移除顶点 k 。

(3) 更新 U 中各个顶点到起点 s 的距离。之所以更新 U 中顶点的距离，是由于上一步中确定了 k 是求出最短路径的顶点，从而可以利用 k 来更新其它顶点的距离；例如， (s,v) 的距离可能大于 $(s,k)+(k,v)$ 的距离。

(4) 重复步骤(2)和(3)，直到遍历完所有顶点。

3.问题描述

给定图 G ，在input.txt中给出图的描述，第一行有两个数字，第一个数字表示节点数目 n ，节点标号从0到 $n-1$ ，第二个表示边的数目 e ，接下来有 e 行，每行有三个数字 u,v,w ，表示边 $u \rightarrow v$ 的权值为 w 。

(1) 给出第1个点到第 n 个点的最短路径，给出最短路径的大小和路径，输出到output.txt中。
output.txt最开始有两行，第一行表示最短路径的长度，第二行表示路径的个数(可能不止一条最短路径)，接下来每一行都表示一条最短路径(0, n_0 ..., $n-1$)

(2) 对于加权有向图 G ，如果从顶点 s 到顶点 t 的一条路径上所有边的权重是严格单调递增或递减的，那么这条路径称为单调路径，路径中不能出现重复顶点。单调最短路径是单调路径中最短的那条路径。给出从顶点0到顶点 $n-1$ 的单调最短路径。

4.提交材料

(1)源代码

(2)输出文件output.txt

将源代码和设计文档使用 zip 格式打包压缩后，重命名为“学号-姓名.zip”提交。

[注]：提供的模板定义了所需实现的行为，通过makefile来测试（input.txt仅为其中一组测试使用，我们会使用不同的测试用例来对程序进行测试）