Meihu Qin

CSI 4999

Oct 14, 2020

Dijkstra Algorithm Research Paper

1. **Define the overview**

Dijkstra algorithm is a typical single-source shortest path algorithm used to calculate the shortest path from one node to all other nodes. The main feature is the starting point as the center to extend layer by layer, until it reaches the end. Dijkstra algorithm is a representative shortest path algorithm, which is introduced in detail as the basic content in many professional courses, such as data structure, graph theory, operations research and so on. Note that the algorithm requires no negative edge weights in the graph.

Problem description: in the undirected graph G=(V,E), assuming that the length of each edge E[I] is W [I], find the shortest path from vertex V0 to the rest points. (Single source shortest path)

2. **Algorithm description**

Ideas: 1) algorithm for G = (V, E) is a weighted directed graph, the graph vertex set V is divided into two groups, the first group is the vertices of the shortest path (with S, says there is only one source point, in the initial S after each, finding a shortest path will be added to the set S, until all the vertices are added to the S, the algorithm is over), the second for the rest of the undetermined vertices of the shortest path (U), according to the length of the shortest path in increasing order of the second set of vertices joined S. In addition, the shortest path length of each vertex from source v to S is always maintained to be no greater than the shortest path length of any vertex from source V to U. In addition, each vertex corresponds to a distance, the distance of a vertex in S is the shortest path length from V to this vertex, and the distance of a vertex in U is the current shortest path length from V to this vertex including only the vertex in S that is the middle vertex.

2) **Algorithm Steps**:

A. Initially, S contains only the source point, that is, S = {v}, and the distance of V is 0. U contains other vertices except v, that is, U={other vertices}, if v has an edge with U in U, then <U,v> has normal weights, if U is not the edge adjacency point of v, then <U,v> has weights of infinity.

B. Select a vertex K from U that is the smallest distance from V and add k to S (the selected distance is the shortest path length from V to k).

C. Take K as the new considered intermediate point, and modify the distance of each vertex in U; If the distance from the source point V to vertex U (through vertex K) is shorter than the original distance (through vertex K), then the distance value of vertex U is modified, and the distance of vertex K of the modified distance value is added to the weight of the edge.

D. Repeat steps B and C until all vertices are contained in S.

The animation process is shown below:

Diagram

Description automatically generated

Reference:

https://www.cnblogs.com/biyeymyhjob/archive/2012/07/31/2615833.html