Aurora Tiffany-Davis
attiffan
CSC 505
HW 4
Problem 3
11/14/17

a Description

The algorithm is modified only very slightly from the Dijkstra's SSSP algorithm given in the text. The modifications are in the initialization and relax functions. Each vertex has a boolean attribute which is true if there is a unique shortest path from the source to the vertex.

During initialization, this attribute is set to true for all vertices. That is, until we learn otherwise, we assume that there is a unique shortest path from the source to each vertex.

During relaxation, this attribute is set to false if we learn that there is another way to get to the vertex, that has the exact same distance as the current shortest path estimate. The attribute is set to true if we learn of a way to get to the vertex, that has a shorter distance than the current shortest path estimate.

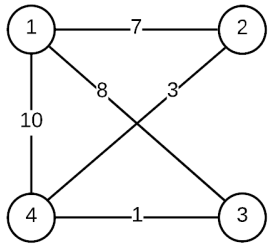
Pseudocode

Modifications from the pseudocode given in the textbook are in bold text.

```
1   Dijkstra-USP (G, w, s) {
2       Initialize-Single-Source (G, s)
3       S = Ø
4       Q = G.V
5       while Q != Ø
6           u = Extract-Min(Q)
7           S = S U {u}
8           for each vertex v ∈ G.Adj[u]
9               Relax(u, v, w)
10  }
11  Initialize-Single-Source(G,s) {
12      for each vertex v ∈ G.V
13          v.d = ∞
14          v.π = NIL
15          v.USP = true
16      s.d = 0
17  }
18  Relax(u, v, w) {
19      if v.d == u.d + w(u, v)
20          v.USP = false
21      else if v.d > u.d + w(u, v)
22          v.USP = true
23          v.d = u.d + w(u, v)
24          v.π = u
25  }
```

a Worked Example

The worked example is a small graph, but one with an interesting property. The algorithm will think that there is a non-unique shortest path from v1 to v4, and then will have to change its mind later and learn that there is a USP after all.



After Initialize-Single-Source:

v	v.d	v.pi	v.USP
1	0	NIL	TRUE
2	INFINITY	NIL	TRUE
3	INFINITY	NIL	TRUE
4	INFINITY	NIL	TRUE

After first iteration of while loop ("u" is vertex 1, because it has the minimum v.d):

v	v.d	v.pi	v.USP
1	0	NIL	TRUE
2	7	1	TRUE
3	8	1	TRUE
4	10	1	TRUE

After second iteration of while loop ("u" is vertex 2, because it has the minimum v.d):

v	v.d	v.pi	v.USP
1	0	NIL	TRUE
2	7	1	TRUE
3	8	1	TRUE
4	10	1	FALSE

After third iteration of while loop ("u" is vertex 3, because it has the minimum v.d):

v	v.d	v.pi	v.USP
1	0	NIL	TRUE
2	7	1	TRUE
3	8	1	TRUE
4	9	3	TRUE

After fourth iteration of while loop ("u" is vertex 4, because it has the minimum v.d):

v	v.d	v.pi	v.USP
1	0	NIL	TRUE
2	7	1	TRUE
3	8	1	TRUE
4	9	3	TRUE

a These results indicate (correctly) that

- There is a unique shortest path from v1 to v1, with length zero
- There is a unique shortest path from v1 to v2, with length 7
- There is a unique shortest path from v1 to v3, with length 8
- There is a unique shortest path from v1 to v4, with length 9

Indication of correctness

Dijkstra's SSSP algorithm is well-established as a correct algorithm for single-source shortest paths. Therefore, this indication of correctness will focus on the modifications made to determine, for each vertex in the graph, whether the shortest path from the source is unique.

Loop invariant:

At the start of each iteration of the while loop of line 5, for each vertex $v \in G.V$, v.USP is correct when considering paths from source to v that only include those vertices in S. That is to say, for each vertex $v \in G.V$, v.USP is true if there is a unique shortest path from source to v, where other vertices along a path from source to v can only be drawn from those vertices in S. For each vertex $v \in G.V$, v.USP is false if there is more than one path from source to v all of which have the minimum path length, where other vertices along a path from source to v can only be drawn from those vertices in S.

Initialization:

Before the first iteration of the while loop of line 5, for each vertex $v \in G.V$, v.USP, v.USP is true, indicating that there is a unique shortest path from source to v. This is trivially correct because for each vertex $v \in G.V$, the path from source to v is NIL. Exception: the source vertex itself also has v.USP true. This is also correct because the problem statement specifies that edge weights in G are positive - therefore there cannot be another path from the source vertex to itself with the minimum path length (zero).

Before the first iteration of the while loop of line 5, S is empty.

Therefore, before the first iteration of the while loop of line 5, for each vertex $v \in G.V$, v.USP is correct when considering paths from source to v that only include those vertices in S.

a Maintenance:

Assume that before a specific iteration of the while loop of line 5, the loop invariant holds.

For each vertex $v \in G.V$, v.USP is correct when considering paths from source to v that only include those vertices in S.

The body of the while loop of line 5 extracts a vertex from Q which has the minimum estimated distance from source and adds this vertex to S. It then relaxes all edges incident to this vertex. During each relaxation, the adjacent vertex may have its v.USP unaltered, made true, or made false. v.USP is set to false if we learn that there is another way to get to the vertex, that has the exact same distance as the current shortest path estimate. v.USP is set to true if we learn of a way to get to the vertex, that has a shorter distance than the current shortest path estimate. If neither condition holds, then v.USP is unaltered.

Only the vertices in S are considered when updating v.USP, and v.USP correctly goes to false in cases where a non-unique shortest path comprised of vertices in S is found, and correctly goes to true in cases where a new unique shortest path comprised of vertices in S is found.

By these steps, the loop invariant is maintained for the next iteration.

For each vertex $v \in G.V$, v.USP is correct when considering paths from source to v that only include those vertices in S.

Termination:

The condition causing the while loop of line 5 to stop is that Q has been emptied of all $v \in G.V$. That is, all $v \in G.V$ have been added to S.

Thus at termination, for each vertex $v \in G.V$, v.USP is correct when considering paths from source to v using any $v \in G.V$.

This result satisfies the problem statement: to set v.USP true if and only if there is a unique shortest path from s to v.

Analysis of time complexity

The goal is to describe a modification of Dijkstra's algorithm that runs (asymptotically) as fast as the original algorithm. The modifications to Dijkstra's algorithm are in bold text in the "Description" section above. Each modification adds only a constant-time operation to an existing loop. Therefore the modified algorithm certainly runs (asymptotically) as fast as the original algorithm.

b Please see Dijkstra.java