

Shortest Route Algorithm: On-the-graph

The Problem

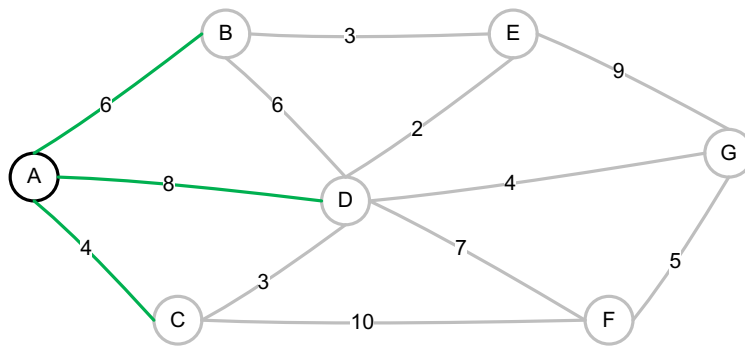
Find the path of least weight from the origin node to the destination node.

Algorithm

The shortest route algorithm fans out from the origin growing multiple paths. On each iteration it finds the shortest path to a “new” (so far unreached) node in the graph. It continues until it reaches the destination node.

Example

We begin highlighting the first node to show we have reached it, and look for the shortest path we can build from it. The three options (shown in green) have lengths 6, 8 and 4.



Paths under consideration:

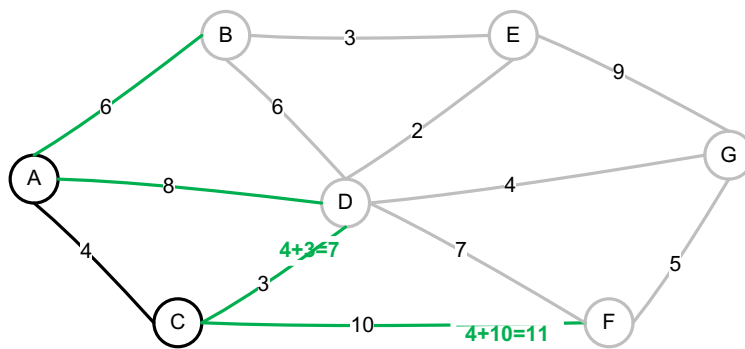
A-B @ 6

A-D @ 8

A-C @ 4

Since 4 is the shortest we choose it and connect A to C. (We have found that the shortest path in the graph that starts at A is A-C with weight 4.) This connection creates two more possible paths for us to consider A-C-D and A-C-F. We calculate the total weights of these paths and record them on the graph.

Now we have four paths to consider: A-B @ 6, A-D @ 8, A-C-D @ 7, and A-C-F @ 11.



Paths found:

A-C @ 4

Paths under consideration:

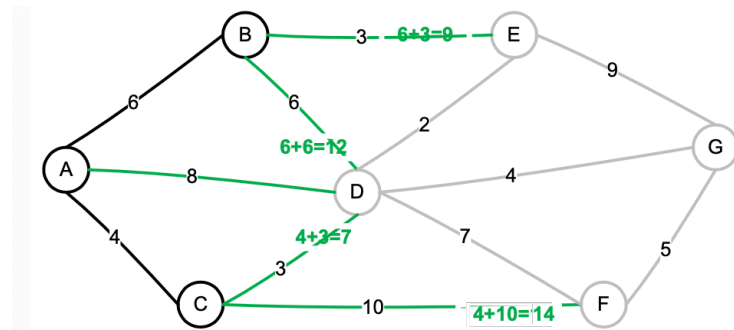
A-B @ 6

A-D @ 8

A-C-D @ 7

A-C-F @ 11

The lightest is A-B so we connect A to B and have found that the second shortest path in the graph from A is A-B with weight 6. As before the new path creates some new possibilities to consider: A-B-E @ 9 and A-B-D @ 12.



Paths found:

A-C @ 4

A-B @ 6

Paths under consideration:

A-B-E @ 9

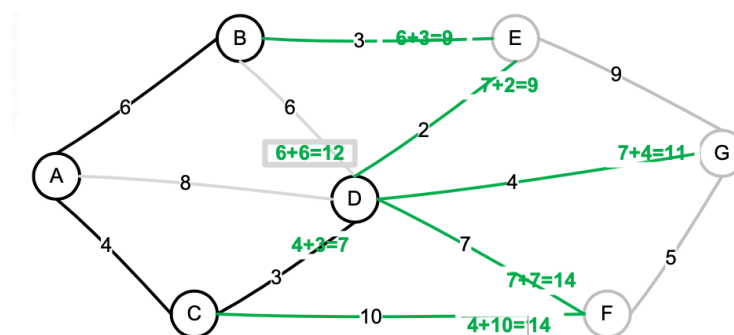
A-B-D @ 12

A-D @ 8

A-C-D @ 7

A-C-F @ 11

Of the five paths to consider A-C-D @ 7 is the lightest so we connect D. Connecting D opens three new possible paths as shown on the diagram, but it also eliminates some paths we were considering, A-D and A-B-D, because we found a shorter way than them to reach D.



Paths found:

A-C @ 4

A-B @ 6

A-C-D @ 7

Paths under consideration:

A-B-E @ 9

A-C-D-E @ 9

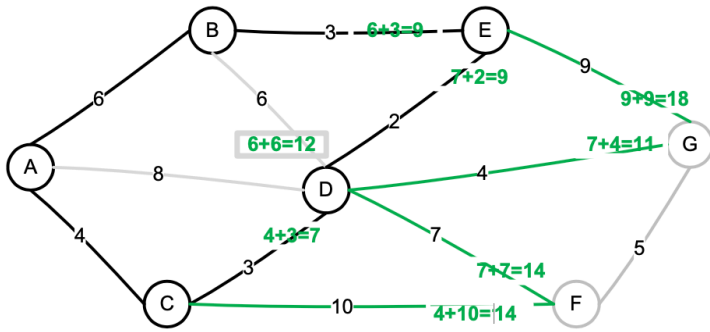
A-C-D-G @ 11

A-C-D-F @ 14

A-C-F @ 11

Of the paths under consideration an old one and a new one tie for lightest, A-B-E and A-C-D-E both @ 9, so we connect E. (It is not uncommon to have a tie for the shortest path to a node. This is not a problem. It just means that there are two equally optimal paths instead of a single unique solution.)

Making this connection opens one new path: A-C-D-E-G @ 18.



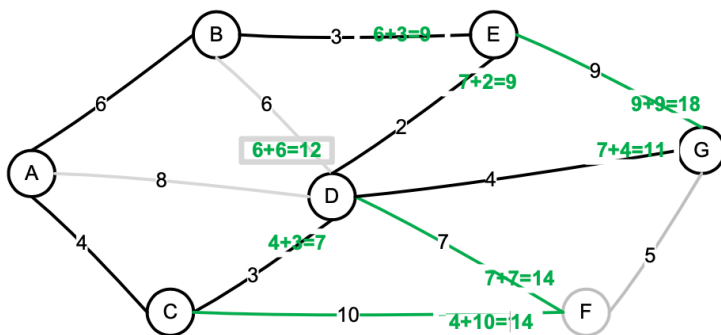
Paths found:

A-C @ 4
 A-B @ 6
 A-C-D @ 7
 A-B-E & A-C-D-E @ 9

Paths under consideration:

A-B-E-G & A-C-D-E-G @ 18
 A-C-D-G @ 11
 A-C-D-F @ 14
 A-C-F @ 11

There are two equally light paths to consider, A-C-D-G and A-C-F @ 11. This tie is a bit different from the previous one. The last tie was between two paths to the *same* node. This time it is a tie between two paths to *different* nodes. It doesn't matter which one we add first, but since G is our destination node we can connect it and be done.



Paths found:

A-C @ 4
 A-B @ 6
 A-C-D @ 7
 A-B-E & A-C-D-E @ 9
 A-C-D-G @ 11

We have discovered that the shortest path from A to G is A-C-D-G with a total weight of 11.

How do we know it is the shortest path? Look at the list of "Paths found". You can see that it lists all shortest paths in the graph in order of increasing weight. If there was a shorter route we would have already found it. Since we didn't this must be the shortest.