I decided to give the edges unique identifiers. This removes the requirement for an O(n2) search for edges. The downside of doing this is that it makes it more complicated to expand the size of the structure. The edges are essentially perfectly hashed. This means that I can represent edges by either two integers (the endpoints) or by a single integer. Conversion from one format to the other is simple:

edgeNumber=startPoint\*n+endPoint. I did make versions of all of the edge methods using both signatures.

The test expands on the "MainSimpleGraph" class that we were given. The goal is to test all the versions of all the methods. Note that we are starting with five cities. Later on I add another. I didn't adjust what is printed when the matrices are outputted so they show lots of zeros for points that don't exist.

hub 0's Name is Philadelphia

its routes are:

0,1

0,3

hub 1's Name is New York

its routes are:

1,2

1,3

hub 2's Name is Boston

its routes are:

2,1

hub 3's Name is Los Angeles

its routes are:

3,4

hub 4's Name is Houston

its routes are:

4,0

4,3

The Adjacency matrix

[0, 1, 0, 1, 0, 0, 0, 0, 0, 0]

[0, 0, 1, 1, 0, 0, 0, 0, 0, 0]

[0, 1, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 1, 0, 0, 0, 0, 0]

[1, 0, 0, 1, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

The Transitive Closure Matrix

[0, 1, 1, 1, 1, 0, 0, 0, 0, 0]

[1, 0, 1, 1, 1, 0, 0, 0, 0, 0]

[1, 1, 0, 1, 1, 0, 0, 0, 0, 0]

[1, 1, 1, 0, 1, 0, 0, 0, 0, 0]

[1, 1, 1, 1, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

Insert a node for Cleveland

Add new routes using edge numbers, Cleveland->Boston(52) and Houston->Cleveland(45)

The Adjacency matrix

[0, 1, 0, 1, 0, 0, 0, 0, 0, 0]

[0, 0, 1, 1, 0, 0, 0, 0, 0, 0]

[0, 1, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 1, 0, 0, 0, 0, 0]

[1, 0, 0, 1, 0, 1, 0, 0, 0, 0] // The green entries are the new ones.

[0, 0, 1, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

Name is Cleveland

true // These two test for existence of an edge that is there.

true

false // These test for existence of an edge that is not there.

false

Delete vertex 3

The Adjacency matrix

[0, 1, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 1, 0, 0, 0, 0, 0, 0, 0]

[0, 1, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[1, 0, 0, 0, 0, 1, 0, 0, 0, 0]

[0, 0, 1, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

Delete edges 21 and 40 by different methods

The Adjacency matrix

[0, 1, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 1, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 1, 0, 0, 0, 0]

[0, 0, 1, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

Update Boston to Hartford

Name is Hartford

The Transitive Closure Matrix

[0, 1, 1, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 1, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 1, 0, 0, 1, 0, 0, 0, 0]

[0, 0, 1, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

One thing this doesn't show is that all the edges which end on node 2 have been deleted but node 2 has not. If we check the vertexExists[2] it will be true.