Centre for Information Technology and Electronics

DTU Diplom
Lautrupvang 15, 2750 Ballerup

Assignment 5

Minimum Spanning Trees

Supervisor(s): **Roger Munck-Fairwood**

Student name	**Student number**	**Signature**
James Testmann	S071954	
Aimo Suikkanen	S082577	
Lukas Janocko	S135272	
Shicheng Dai	S133342	
Sudhir Kumar Chaurasiya	S137239	

- Describe an algorithm (pseudocode) that will give the cheapest routing for connection of the houses in Figure 1 (there must be no cycles).

```
prims( vertex v )
{       mark v as visited and
                include in spanning tree;
        while( there are unvisited vertices )
        {       find the least-cost edge(v, u)
                                from some visited vertex v
                                to some unvisited vertex u;
                mark( u );
                add u and edge(v, u) to spanning tree;
        } // end while
}
```

From starting point (vertex) we look around for the cheapest edge. We move to the new vertex and we add edge together with vertex to the spanning tree. Then we continue from current vertex, we search for cheapest edge, but we also consider if there is cheaper edges from already visited vertices. We continue until there is no unvisited vertex.

Centre for Information Technology and Electronics

DTU Diplom
Lautrupvang 15, 2750 Ballerup

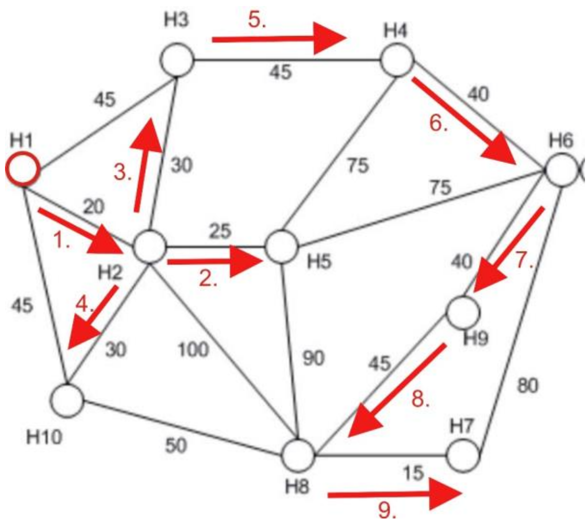
- At least two different houses should be used as starting point (use H1 and H5 to make it easier to compare solutions)

  This solution is shown in the next point.

- Run the algorithm by hand, describing the routing point by point together with the total cost (use the following tie-breaker: Select the node having the lexographically lowest label).
  - Starting point at H1.

Start
Visited/Shortest path

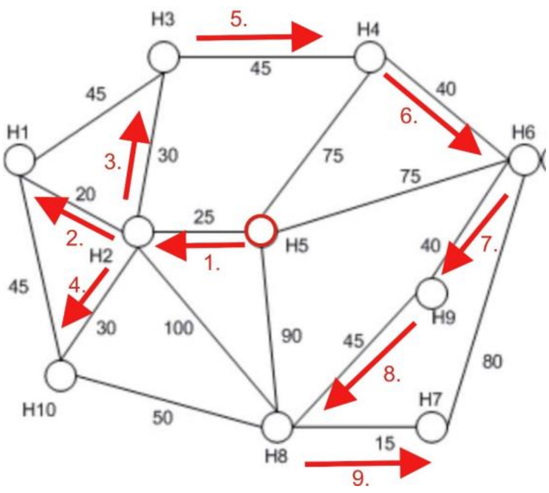
V	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10
Distance	0	**20**	45							45
From V		1	1							1
V	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10
Distance	0	0	30		**25**			100		30
From V	1	1	2		2			2		2
V	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10
Distance	0	0	**30**	75	0	75		90		30
From V	1	1	2	5		5		5		2
V	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10
Distance	0	0	0	45	0	75		90		**30**
From V	1	1	2	3	2	5		5		2
V	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10
Distance	0	0	0	**45**	0	75		50		0
From V	1	1	2	3	2	5		5		2
V	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10
Distance	0	0	0	0	0	**40**		50		0
From V	1	1	2	3	2	4		5		2
V	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10
Distance	0	0	0	0	0	0	80	50	**40**	0
From V	1	1	2	3	2	4	6	5	6	2
V	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10
Distance	0	0	0	0	0	0	80	**45**	0	0
From V	1	1	2	3	2	4	6	9	6	2
V	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10
Distance	0	0	0	0	0	0	**15**	0	0	0
From V	1	1	2	3	2	4	8	9	6	2



- Starting point in H5.

Start
Visited/Shortest path

V	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10
Distance		**25**		75	0	75		90		
From V		5		5	5	5		5		
V	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10
Distance	**20**	0	30	75	0	75		90		30
From V	2	5	2	5	5	5		5		2
V	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10
Distance	0	0	**30**	75	0	75		90		30
From V	2	5	2	5	5	5		5		2
V	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10
Distance	0	0	0	45	0	75		90		**30**
From V	2	5	2	3	5	5		5		2
V	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10
Distance	0	0	0	**45**	0	75		50		0
From V	2	5	2	3	5	5		10		2
V	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10
Distance	0	0	0	0	0	**40**		50		0
From V	2	5	2	3	5	4		10		2
V	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10
Distance	0	0	0	0	0	0	80	50	**40**	0
From V	2	5	2	3	5	4	6	10	6	2
V	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10
Distance	0	0	0	0	0	0	80	**45**	0	0
From V	2	5	2	3	5	4	6	9	6	2
V	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10
Distance	0	0	0	0	0	0	**15**	0	0	0
From V	2	5	2	3	5	4	8	9	6	2



Centre for Information Technology and Electronics

DTU Diplom
Lautrupvang 15, 2750 Ballerup

- More than one solution may exist. In this case how many solutions can you find?

  There is only one solution. No matter from where we start, the solution will still be the same. Which is the idea of a minimum spanning tree.

- Compare the minimum spanning tree algorithm with the Dijkstra shortest path algorithm and reflect on the principal similarities and differences.

  In both Dijkstra and Minimum Spanning Tree, the primary subject is to notice the shortest edge to the connected vertices.

  Dijkstra shortest path algorithm is preferable when we want to know the cheapest route from starting node to specific one.
  Minimum spanning tree algorithm is convenient when we want to know the cheapest route through all nodes.

- A graph may be implemented as an adjacency matrix, or an adjacency list. What is the complexity (in big O notation) of the minimum spanning tree algorithm with the two different graph models?

  All edges connected to the notes is considered, therefore more edges in a system become more calculations. Therefore $O(N^2)$