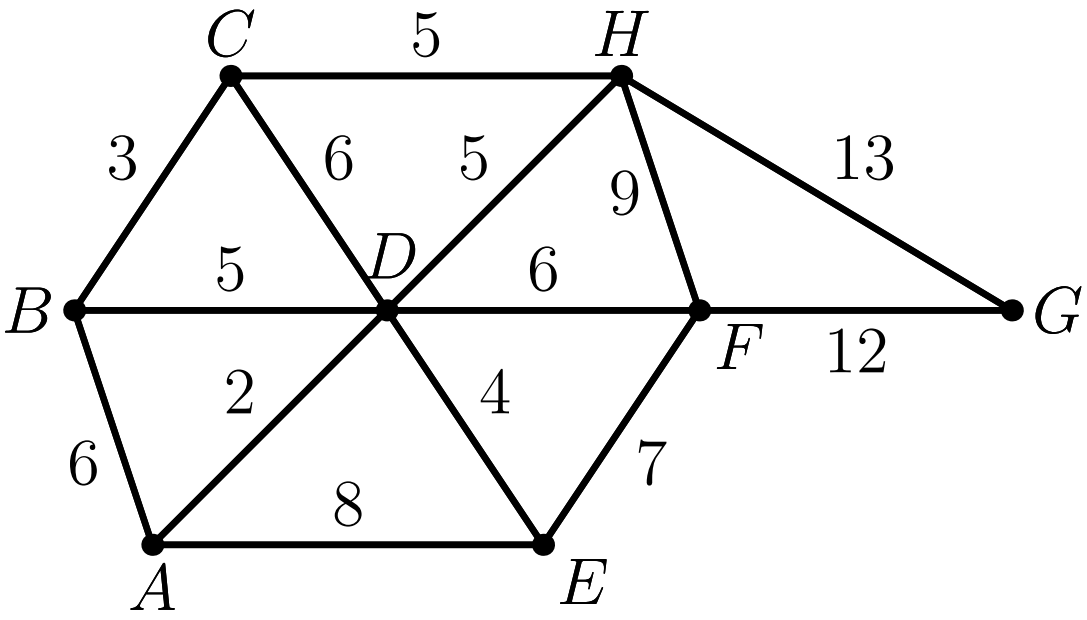
MA2C03 Mathematics
School of Mathematics, Trinity College
Hilary Term 2016
Lecture 36 (January 27, 2016)

David R. Wilkins

Example

We apply Kruskal's Algorithm to find a minimal spanning tree for the following graph:—



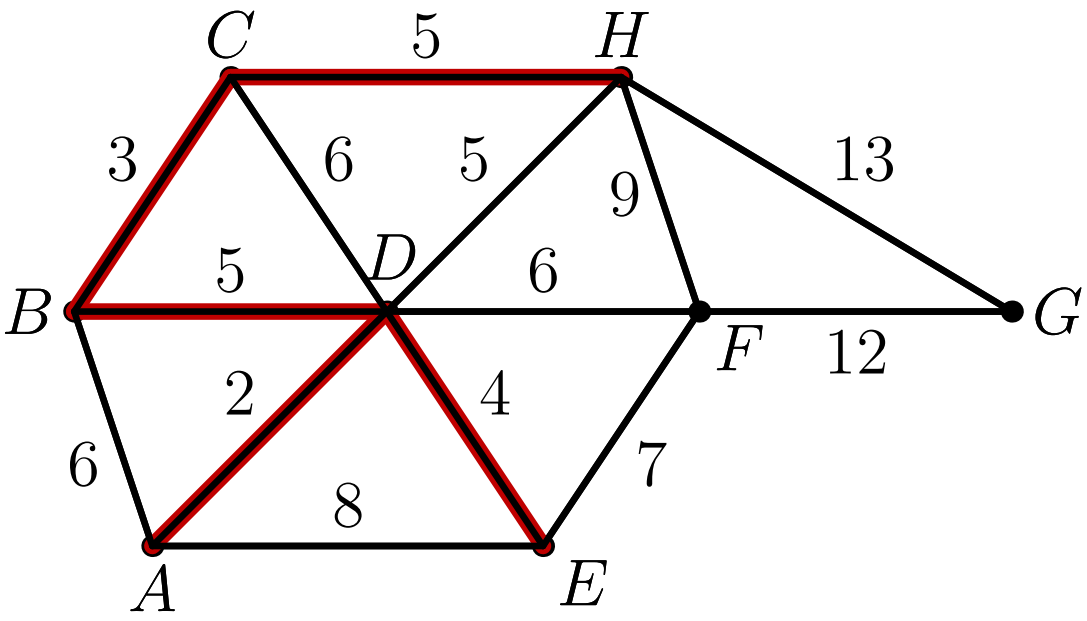
(Costs are specified next to the relevant edge.)

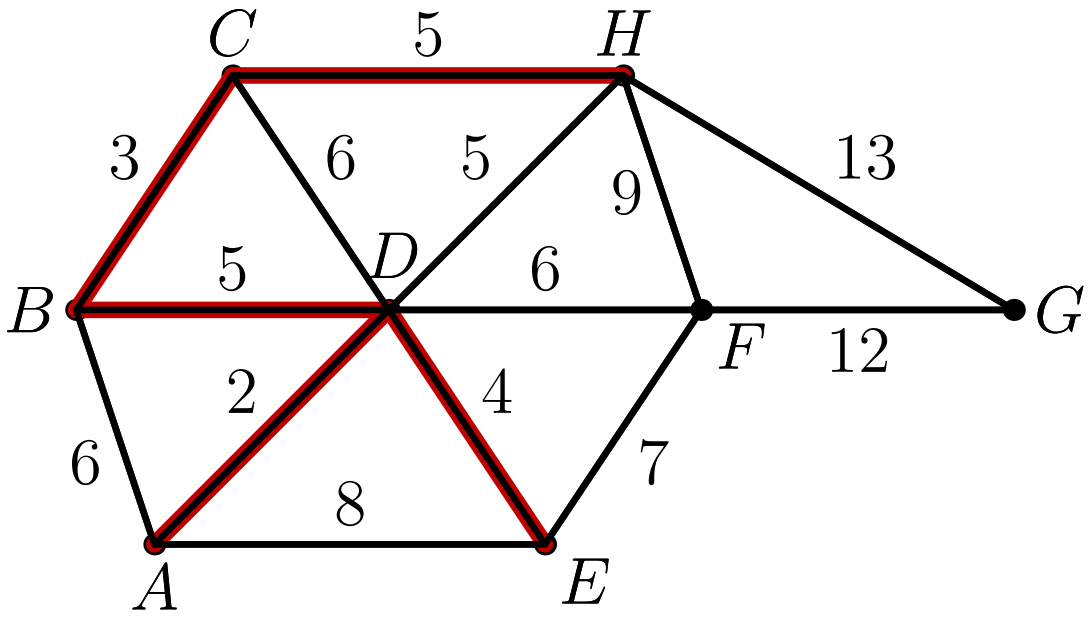
We order the edges so that the associated costs are non-decreasing. The edges are listed in order with their associated costs as follows:—

AD	BC	DE	BD	CH	DH	AB
2	3	4	5	5	5	6
CD	DF	EF	AE	FH	FG	GH
6	6	7	8	9	12	13

We build up an acyclic subgraph of the given connected graph as follows. We start with an acyclic subgraph consisting of the vertices of the original graph. We then treat the above list of edges as a queue, taking edges in turn from the head of the queue, and add them to the subgraph if and only if doing so does not create any circuits in the subgraph.

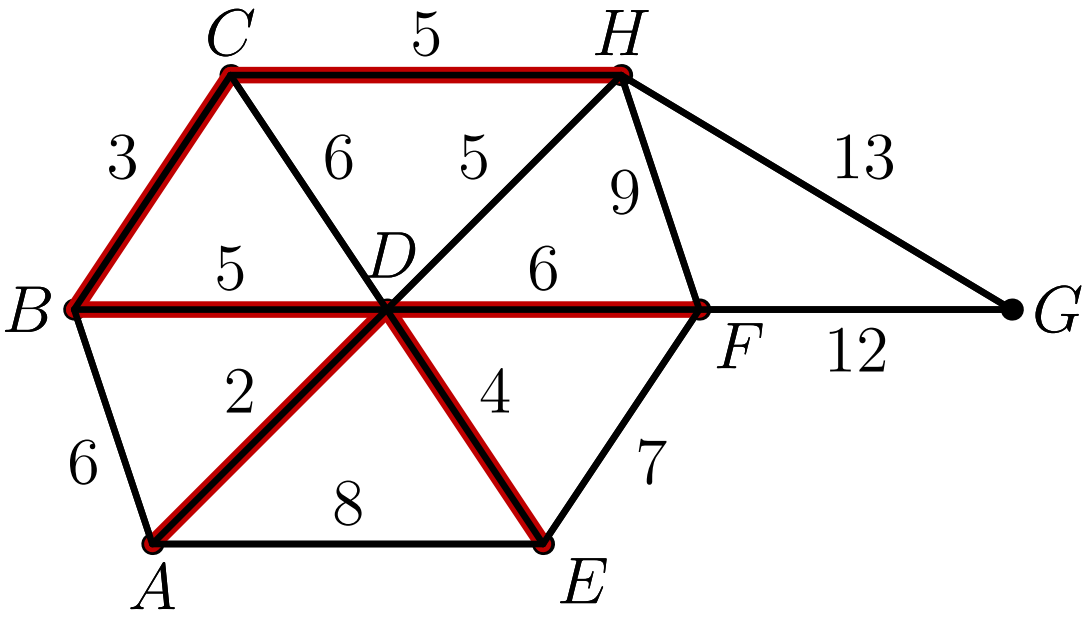
In the first five iterations we add the edges AD, BC, DE, BD and CH. No circuit is created at any stage, and the resultant acyclic subgraph is represented by the vertices and thick edges in the following diagram:—



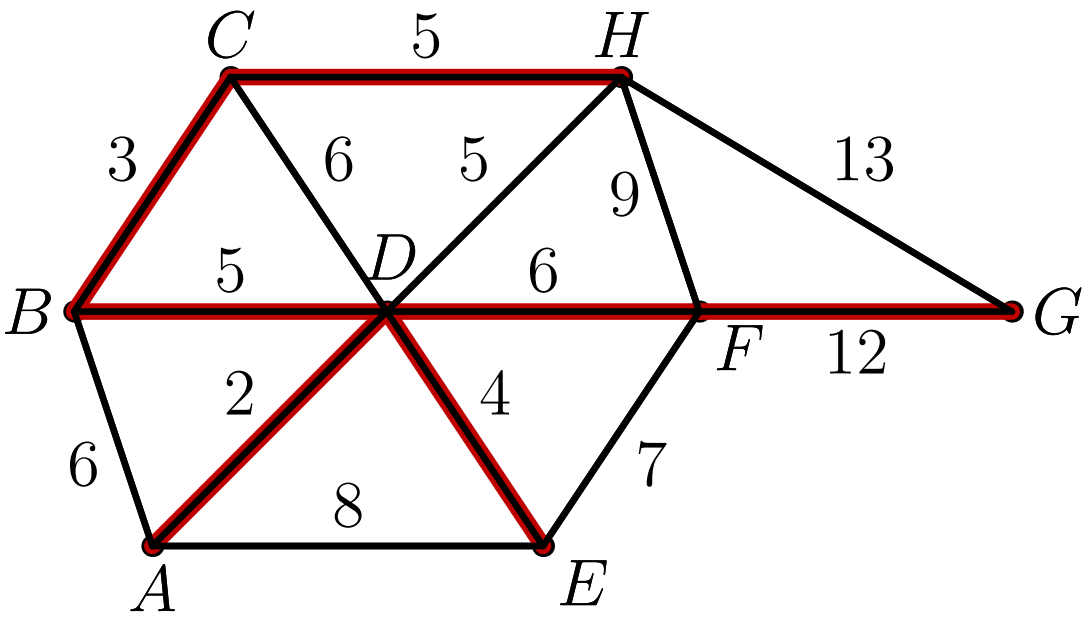


Adding the edge DH would create a circuit $DBCHD$; therefore the edge DH is discarded. Adding the edge AB would create a circuit $ABDA$; therefore the edge AB is discarded. Adding the edge CD would create a circuit $DBCD$; therefore the edge CD is discarded.

The next edge in the queue is DF. We can add this edge to the acyclic subgraph. However the edges EF, AE and FH must be discarded, since adding any of those edges to the subgraph would create a circuit. The current acyclic subgraph is now as follows:—



We add the edge FG to the subgraph, as this is the next in the queue that may be added without creating a circuit in subgraph. We cannot then add GH. Thus the minimal spanning tree generated by Kruskal's Algorithm is as follows:—



The minimal spanning tree generated by Kruskal's Algorithm thus consists of the vertices A, B, C, D, E, F G and H of the given connected graph, together with the edges e_1, e_2, e_3, e_4, e_5, e_6 and e_7, where $e_1, \ldots, e_7$ denote the edges of the minimal spanning tree listed in the order in which they were added to the acyclic graph, so that

$$e_1 = AD, \quad e_2 = BC, \quad e_3 = DE, \quad e_4 = BD,$$

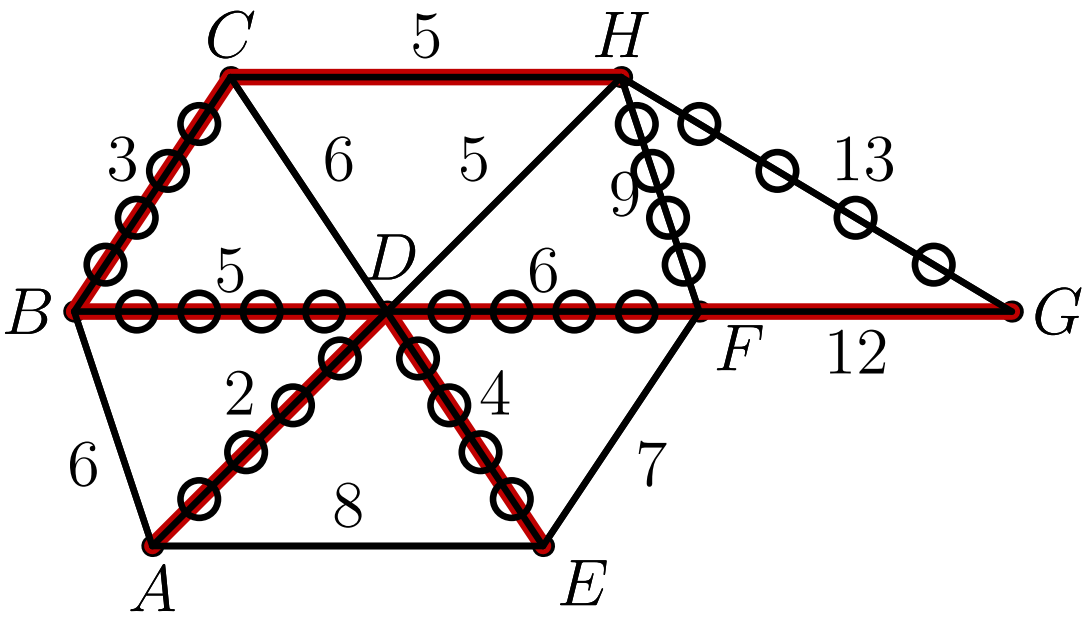
$$e_5 = CH, \quad e_6 = DF, \quad e_7 = FG.$$

We refer to the spanning tree generated by Kruskal's Algorithm as the *Kruskal tree*.

However we have not yet shown that the Kruskal tree has minimal cost.

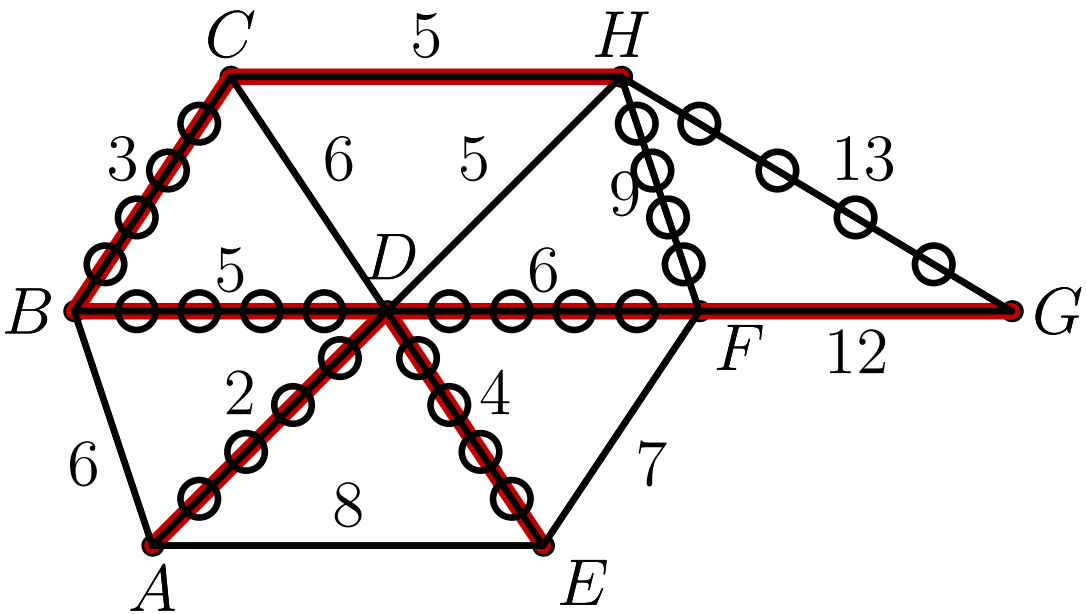
We claim that, given any integer k satisfying $0 < k \leq 7$, and given any spanning tree that includes edges e_i of the Kruskal tree whenever $i < k$ but does not include the edge e_k, this spanning tree can be modified to yield a spanning tree that includes edges e_i of the Kruskal tree for $i \leq k$, where the cost of the modified tree does not exceed that of the given spanning tree.

We consider the example below. The spanning tree T represented by the edges with circles includes edges e_1, e_2, e_3 and e_4 of the Kruskal tree (i.e., the edges AD, BC, DE and BD), but does not include the edge e_5 where $e_5 = CH$:–

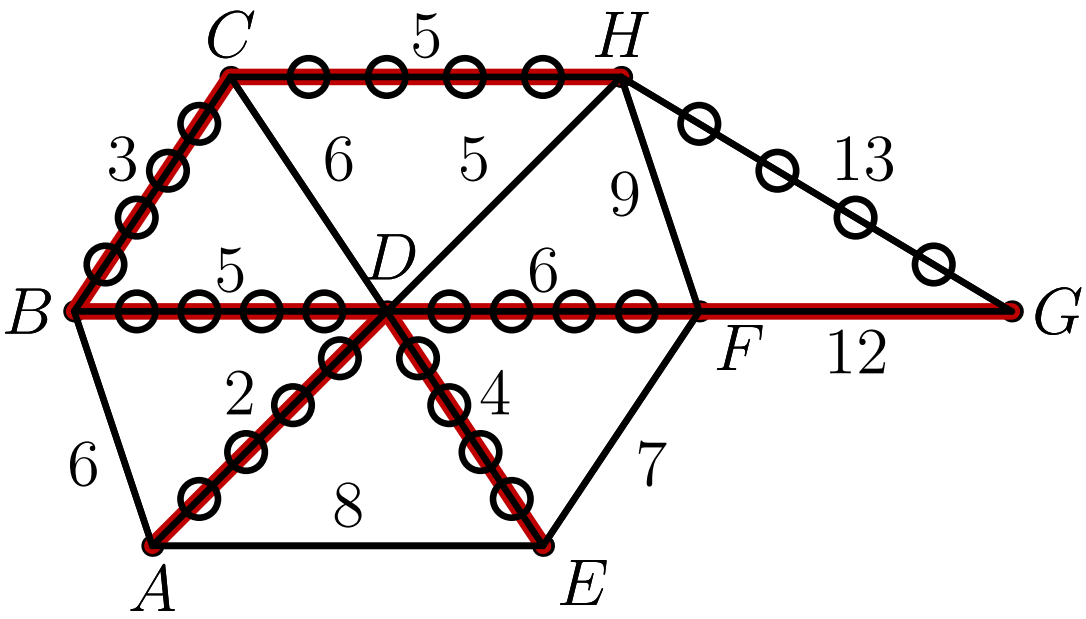


Adding any edge to a spanning tree creates a circuit. In particular, if we add the edge CH to the spanning tree T, then the resultant subgraph of the original connected graph must contain a circuit. In the present example the circuit created is $CHFDBC$. Now not all edges of that circuit can belong to the Kruskal tree, because trees cannot contain circuits. Therefore at least one edge of the circuit $CHFDBC$ does not belong to the Kruskal tree.

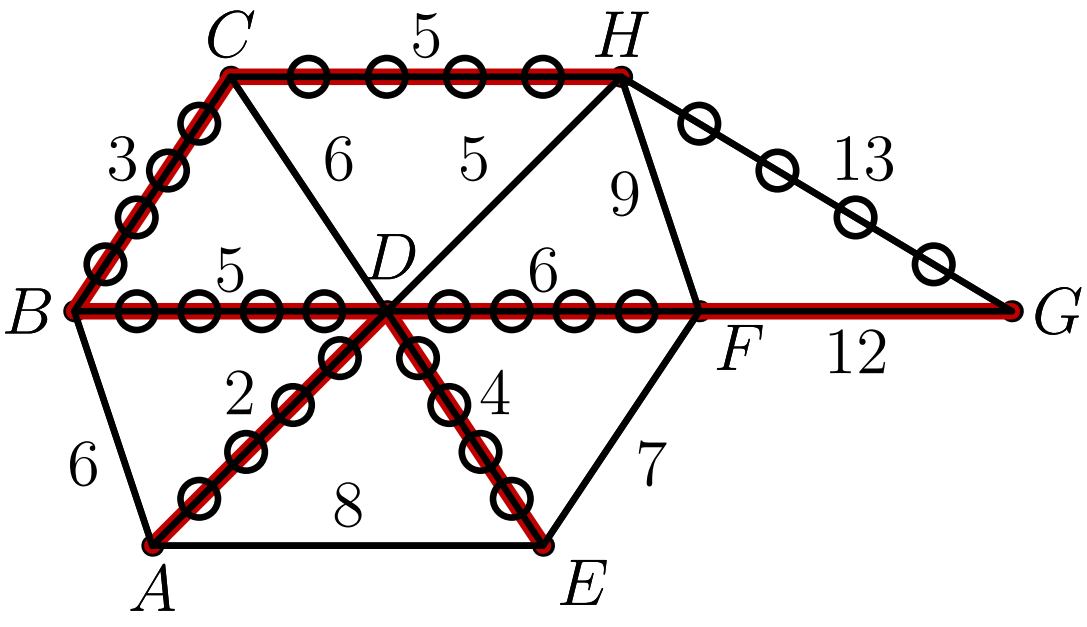
In the example under consideration, the edge FH does not belong to the Kruskal tree. Now if the cost of FH were less than that of CH then the Kruskal algorithm would have required the edge FH to be added to the Kruskal tree before CH. It follows that the cost of the edge FH cannot be less than CH. Indeed FH has cost 9, whereas CH has cost 5.



It follows that we can modify the spanning tree T to obtain a new spanning tree T', where T' includes the edges e_1, e_2, e_3, e_4 and e_5 of the Kruskal tree, and where the cost of T' does not exceed that of T. The resultant tree T' consists of the edges with circles in the following figure:—



The spanning tree T' includes edges e_i of the Kruskal spanning tree for $i < 7$ but does not include the edge e_7, where $e_7 = FG$. Now addition of the edge e_7 to the spanning tree creates a circuit $FGHCBDF$ in the resultant subgraph. This circuit cannot consist entirely of edges of the Kruskal tree. Therefore at least one edge of the circuit does not belong to the Kruskal tree. That edge is GH.



The cost of the edge GH cannot exceed that of the edge e_7, because otherwise GH would have been added to the Kruskal tree before FG was considered in the application of Kruskal's Algorithm. Therefore replacement of GH by FG results in a spanning tree T'' whose cost does not exceed that of T'. This spanning tree T'' then includes all the edges of the Kruskal tree, and must therefore be identical to the Kruskal tree. It follows that the cost of the Kruskal tree cannot exceed that of the tree T', and therefore cannot exceed that of the tree T.

We claim that the procedure just described can be applied to demonstrate that the cost of the Kruskal tree is less than or equal to the cost of any spanning tree for the given connected graph.