CS310 - Advanced Data Structures and Algorithms

Spring 2014 – Class 25

May 8, 2014

Minimum Spanning Tree

- Given a connected graph, we often want the cheapest way to connect all the nodes together, and this is obtainable by a minimum cost spanning tree.
- A spanning tree is a tree that contains all the nodes in the graph.
- The total cost is just the sum of the edge costs.
- The minimum spanning tree is the spanning tree whose sum of edge cost is minimal among all spanning trees.

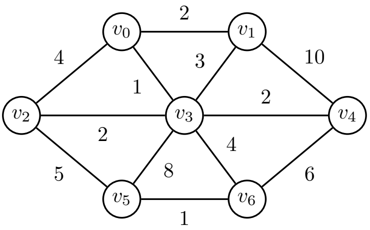
Minimum Spanning Tree

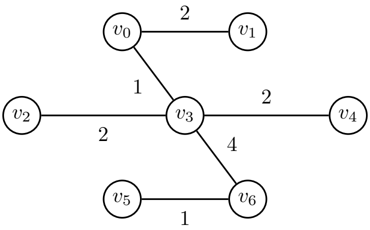
- The kind of tree involved here has no particular root node, and such trees are called free trees, because they are not tied down by a root.
- Weiss relates the min-cost spanning tree problem to the Steiner tree problem.
- If you want to see an example of a Steiner tree, follow this link: `http://www.cs.sunysb.edu/~algorith/files/steiner-tree.shtml`

Kruskal's algorithm

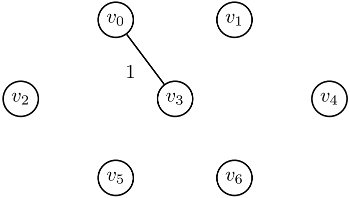
- This algorithm is a lot like Huffman's.
- Start with all the nodes separate and then stick them together incrementally, using a greedy approach that turns out to give you the optimal solution.
- Set up a partition, a set of sets of nodes, starting with one node in each set.
- Then find the minimal edge to join two sets, and use that as a tree edge, and join the sets.

Example

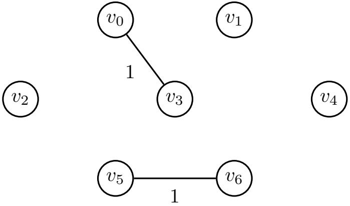




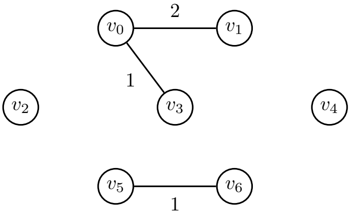
Example



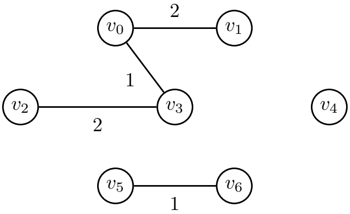
1



2

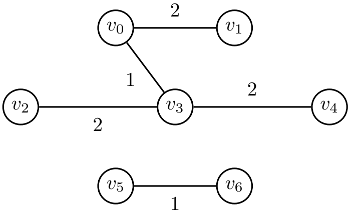


3

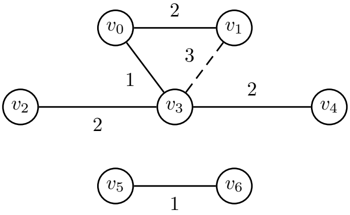


4

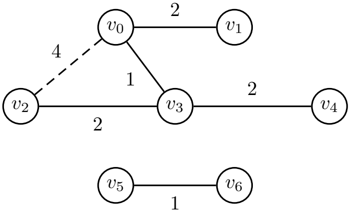
Example



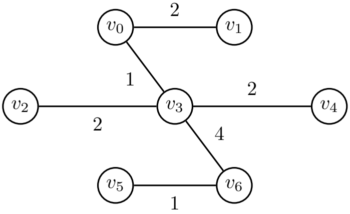
5



6



7



8

- Note that several edges are left unprocessed in the edge list: these are the high-cost edges we were hoping to avoid using.
- Implementation: There are fancy data structures for partitions.
- Come back to this chapter if you ever need to do this with good performance.
- In cases that dont need the very best performance, a Map from vertex to partition number will do the job of holding the partition.

Running Example

Here are the partition numbers for the first few steps:

Node	V_0	V_1	V_2	V_3	V_4	V_5	V_6	
Set	0	1	2	3	4	5	6	
V_0-V_3	0	1	2	0	4	5	6	(3s turned into 0s)
V_5-V_6	0	1	2	0	4	5	5	(6s turned into 5s)
V_0-V_1	0	0	2	0	4	5	5	
V_2-V_3	0	0	0	0	4	5	5	

- and so on. Edges are used in cost order.
- If an edge has both to and from vertices with the same partition number, it is skipped.
- The resulting tree is defined by the edges used.

Quite similar to Kruskal:

```
 Initialize tree V=v s.t.  v is an arbitrary node.
Initialize E={}.
Repeat until all nodes are in T:
    Choose an edge e=(v,w) with minimum
    weight such that v is in T and w is not.
    Add w to T, add e to E.
Output T=(V,E)
```