Design and Analysis of Algorithms — Lab

R S Milton
Department of CSE, SSN College of Engineering

Session 7: Greedy Algorithms
Kruskal's Algorithm
20 February 2020

Problem: Given a graph $G = (V, E)$, a spanning tree $M = (V, T)$ is a tree formed with all the vertices of V and a subset of edges of E, such that the vertices are connected. Cost of a spanning tree is the sum of the costs (weights) of the edges forming the tree. Given a weighted graph $G = (V, E)$, find a minimum-cost spanning tree (MST) of G.

1. A partition of a set V is a set of subsets called *blocks* such that every element of S is in exactly one block. An ADT for partition (known as union-find ADT) should implement the two operations `Find(e)` and `Union(b1, b2)` efficiently. `Find(e)` returns the block of element `e`, and `Union(b1, b2)` combines two blocks `b1` and `b2` into a single block.

   In union-find ADT, represent each block by a tree and the elements of the block by the nodes of the tree. Implement all the trees of the set S of n elements using an array `parent[1:n]` where `parent[e]` is the position of the parent of element `e`. `parent[e] = -1` indicates that `e` is a root. The root of a tree is the representative of a block.

   Provide implementations of union and find operations. Your union should run in constant time and find in $O(\log |S|)$ time.

2. Implement Kruskal's algorithm with time complexity not exceeding $O(|E| \log |V|)$. You should use your union-find implementation to achieve this time complexity.