Greedy Algorithm: Minimum Cost Spanning Tree

Mason U'Ren, James Musselman, Zander Nelson

October 24, 2016

Abstract

A minimum cost spanning tree (MCT) is a subset of the edges of a connected, edge-weighted undirected graph that connects all the vertices together without creating any cycles using the minimum weight possible. Assuming that we are working with an n-grid, we first look at the algorithm by which the edges are being selected then discuss the maximum number of edges that an n-grid may have while still being a MCT.

Algorithm 1 MinCost Spanning Tree Algorithm

```
 1: $A = \emptyset$                                        ▷ A: the MinCost Spanning Tree
 2: for all $v \in G.V$ do                                 ▷ G: the graph, V: vertex
 3:     $makeSet(v)$
 4: end for
 5: for all $(u, v)$ in G.E sorted increasing by $weight(u, v)$ do
 6:     if $findSet(u) \neq findSet(v)$ then
 7:         $A = A \cup (u, v)$
 8:         $union(u, v)$
 9:     end if
10: end forreturn A
```

1 What is the largest number of edges that an n-grid may have?

Assuming that our n-grid is connected by a minimum cost spanning tree, then the maximum number of edges in the n-grid is $n^2 - 1$. An n-grid is defined as a graph of n^2 nodes which is organized as a square array of $n \times n$ points, which are then connected in the cardinal directions without creating a loop.

Take for example a n-grid where $n = 4$. If we were to connect every point to another, restricted to the allowable directions and neglecting if loops were created, we would have a maximum of 24 edges. By simple deduction we can say that the minimum cost spanning tree then must have less than 24 edges. Thus, after eliminating all the loops we can see that only $n^2 - 1 = 4^2 - 1 = 15$ edges remain.

2 Correctness

Proof. This proof consists of two parts. The first, is that the algorithm produces a spanning and second, is to prove that the constructed spanning tree has minimal weight.

1. **Spanning Tree:** Let P be a connected, weighted graph and let Y be the subgraph of P produced by the algorithm. Obviously Y cannot form a circuit and Y cannot be disconnected, since the first edge that joins two components of Y would have been added to the algorithm. Thus, Y is a spanning tree of P.

2. **Minimality:** This proof is done by contradiction. Lets assume that Y is not a minimal spanning tree and among all minimum weight spanning trees, we pick Y_1 which has the smallest number of edges which are not in Y. Next we consider the edge e which was first to be added by the algorithm to Y of those which are not in Y_1. The union of Y_1 and 'e' ($Y_1 \cup e$) forms a cycle. Being a tree, Y cannot contain all edges of this cycle. Thus, this cycle contains an edge f which is not in Y. The graph $Y_2 = Y_1 \cup e$ is also a spanning tree and therefore its weight cannot be less than the weight of Y_1, since Y_1 is one the minimal spanning trees with the smallest number of edges and hence the weight of the edge e cannot not be less than the weight of f. However the edge e is selected at the first step of the algorithm which means that it must be the smallest weight out of all the edges. From this we can say that e and f therefore must be equal to each other and hence Y_2 is also a minimal spanning tree. But Y_2 has on e more edge in common with Y and Y_1, which contradicts to the choice of Y_1.

$\square$