COMPSCI 330: Design and Analysis of Algorithms 3/01/2016

Minimum Spanning Tree

Lecturer: Debmalya Panigrahi *Scribe: Tianqi Song*

1 Overview

This lecture introduces basic concepts and two algorithms for minimum spanning tree: Kruskal's algorithm and Prim's algorithm.[1]

2 Minimum Spanning Tree

Definition 1. *Given an undirected weighted connected graph $G = (V,E)$, a spanning tree is a subgraph $G' = (V,E')$ of G, where $E' \subseteq E$, such that G' is connected and acyclic.*

Definition 2. *A minimum spanning tree (MST) is a spanning tree with minimum total weight.*

2.1 Generic Property of Minimum Spanning Tree

Let $F = (V,A)$ be a subgraph of $G = (V,E)$. We say that a cut $(S,V \setminus S)$ respects the components of F if the vertices of each connected component of F is a subset of S or $V \setminus S$. In other words, both of the endpoints of each of the edges in A are in S or $V \setminus S$ and therefore, no edge in A crosses the cut.

Lemma 1. *Suppose $F \subset T$ for some MST T. Then, if $(S,V \setminus S)$ is a cut that respects the components of F, and e is the minimum weight edge crossing the cut $(S,V \setminus S)$, then $F \cup e \subseteq T'$ for some MST T'. We call e a safe edge for F.*

Proof. Let $e = (u,v)$, and u be the endpoint in S. If e is an edge of T, then $F \cup e \subseteq T$ and we are done. Otherwise, there is a path between u and v in T and $T \cup e$ contains one cycle. There is at least one edge crossing the cut $(S,V \setminus S)$ in any path between u and v. Let e' be such an edge in the path between u and v in T. We claim that $T' = T \cup e - e'$ is an MST. The reason is that $T \cup e$ contains one cycle and deleting e' from the edges make it acyclic again and T' is a spanning tree. In addition, e is the minimum weight edge crossing the cut and therefore, replacing e' by e does not increase the weight of T and T' is an MST.

We know that the cut respects the components of F and no crossing edge (especially e end e') is in F. e' is the only edge which is in T and is not in T'. We can conclude that $F \subset T'$. Moreover, e is in T' and therefore, $F \cup e \subseteq T'$. $\square$

The generic idea for finding MST is the following. Start with $F = (V,\Phi)$ which is a proper subgraph of any MST. We call it F_0. In the ith iteration we use the generic property and add a safe edge to F_{i-1} to find F_i. Finally, after $|V|-1$ iteration we have an MST. Note that when $i < |V|-1$, the lemma says that F_i is a subgraph of some MST and we know that each MST has exactly $|V|-1$ edges and therefore, it is a proper subgraph and we can use the lemma one more time. In the last iteration we know that $F_{|V|-1}$ is a subgraph of an MST and has $|V|-1$ edges. Therefore it is the MST.

[1]Some of the material is from a previous note by Yilun Zhou for this course in Fall 2014.

We can design different algorithms with this generic idea. The important point in these algorithms is the way we choose the cut. We will show how Prim's and Kruskal's choose the cut in each iteration in the following sections.

2.2 Prim's Algorithm

The pseudocode is:

Algorithm 1 Prim's Algorithm

```
function PR($G = (V,E)$)
    $c[s] = 0$
    $\forall v \neq s \in V$, $c[v] = +\infty, prev[v] = NIL$
    $E' = \emptyset$
    $H = V$
    while $H \neq \emptyset$ do
        $u = deletemin(H)$
        $E' = E' \cup (prev[u], u)$, if $u \neq s$
        for all $(u,v) \in E$, where $v \in H$ do
            if $c[v] > l(u,v)$ then
                $c[v] = l(u,v)$
                $prev[v] = u$
```

2.2.1 Running Time

Prim's algorithm has the same running time as Dijkstra's algorithm, $O(|E|\log|V|)$, by binary heap. It can be improved to $O(|E| + |V|log|V|)$ by Fibonacci heap.

2.2.2 Correctness Proof

Prim's algorithm is a way to implement the generic idea for finding MST. In the Prim's algorithm $F = (V,E')$ contains a tree and some single vertices in each step. Let S shows the vertices of the tree at each step. we add the minimum edge crossing the cut $(S, V \setminus S)$ to E'. In the first step $S = s$. This algorithm is correct according to the generic property. It is obvious that the cut $(S, V \setminus S)$ respects F and we add the minimum edge crossing this cut in each step which is a safe edge.

2.3 Kruskal's Algorithm

The pseudocode is:

Algorithm 2 Kruskal's Algorithm

```
1: function KR($G = (V,E)$)
2:     $E' = \emptyset$
3:     $\forall v_i \in V$, make $V_i = \{v_i\}$
4:     Sort edges in nondecreasing order
5:     for each edge $(u,v) \in E$, taken in nondecreasing order do
6:         if $u$ and $v$ are not in the same set then
7:             $E' = E' \cup (u,v)$
8:             Union(U,V), where $u \in U$ and $v \in V$
9:     Return $E'$
```

Lemma 2. *$G' = (V,E')$ is a spanning tree.*

Proof. If G' is not connected, there exist edges that should be selected by the algorithm but not in E', contradiction. Line 6 guarantees that G' is acyclic. □

Lemma 3. *After each selection of an edge by Kruskal's algorithm, there exists a minimum spanning tree $T = (V,E_t)$ such that $E' \subseteq E_t$.*

Proof. We prove it by induction. For the base case when $E' = \emptyset$, it is true. Assume that there exists a minimum spanning tree $T_n = (V,E_n)$ such that $E' \subseteq E_n$ when E' has n edges. For the $(n+1)th$ selection e_{n+1}, if T_n already contains e_{n+1}, T_n is the tree that we want. otherwise, we add e_{n+1} to T_n and make a cycle. There exists an edge e in the cycle such that $e \notin E'$, since T_n is acyclic. The weight of e must be not smaller than the weight of e_{n+1}, otherwise, e should have been selected by the algorithm. Therefore, the tree constructed by adding e_{n+1} to T_n and deleting e from T_n is also a minimum spanning tree. □

Theorem 4. *$G' = (V,E')$ is a minimum spanning tree.*

Proof. Directly from lemma 2 and lemma 3. □

We can also prove the correctness of the Kruskal's by using generic property. In Kruskal's $F = (V,E')$ is a forest in each step because we add an edge only if it doesn't create a cycle. Before adding an edge $e = (u,v)$ in one step we claim that there is a cut which respects the components of F and e is the minimum edge crossing that cut. Let S be the vertices of the tree containing u in the forest. $(S, V \setminus S)$ is such a cut.

2.3.1 Running Time

The running time of Kruskal's algorithm depends on the implementation of $''Union''$ and $''Find''$. We will discuss it in next lecture.