ln1 + ln2 + ln3 + … + lnN =

Proof of Prims’s Algorithm:

Assuming T is the tree that is created by Prim’s Algo and T\* is an MST. If T is different from T\*, then there exists an edge e in T but e not in T\*. e is an edge that connecting two parts S and V-S, since T\* is an MST, there must exist an edge e\* != e in T\* that connects S and V-S. Since e is selected in Prim’s algo that is the smallest cost edge connecting S and V-S, hence w(e) <= w(e\*), then we replace e\* by e in T\*, it creates another tree (T\*+e will create a cycle in T\* while removing e\* eliminates the cycle) T# with w(T#) <= w(T\*), which means T# is another MST. If the weights of edges are all distinct, then we arrive at a contradiction. Otherwise, we can repeat this process k times until we get {T#}\_k = T.

Proof of Kruskal’s algo:

Assuming T is the tree that is created by Kruskal’s algo and T\* is an MST. Assuming edge e is the first edge in T that is not in T\*, since T\*+e will create a cycle C, then there exist an edge e\* in T\* which does not belong to T, otherwise all edges in C belongs to T which is not possible. Since e is the first edge in T - T\*, which means w(e) < w(e\*), we replace e\* with e in T\*, then it creates another tree T# with w(T#) <= w(T\*), which means T# is another MST. If the weights of edges are all distinct, then we arrive at a contradiction. Otherwise, we can repeat this process until we get T### = T.

The aforementioned proof is called Exchange Argument, a method to prove a greedy algorithm is optimal.

1. Assume the greedy algorithm does not produce the optimal solution, so the greedy and optimal solutions are different.
2. Show how to exchange some part of the optimal solution with some part of the greedy solution in a way that improves the optimal solution.
3. Reach a contradiction and conclude the greedy and optimal solutions must be the same.