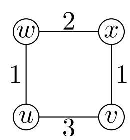
The following examples show how easy it can be to write an incorrect proof that Kruskal's algorithm produces a minimum spanning tree.

First Wrong Proof

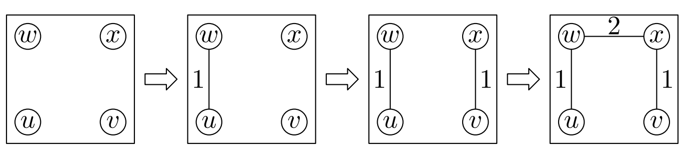
This proof came from earlier CS221 class notes:

1. We already know Kruskal's alg. finds a spanning tree T.
2. Assume another spanning tree, T_1, has *lower cost* than T.
3. Pick an edge $e_1 = (u, v)$ in T_1 that's *not* in T.
4. Kruskal's alg. connects u and v at some point during its execution using a different edge e.
5. But e must have at most the same cost as e_1 (or Kruskal's would have used e_1 to connect u and v)
6. So, replace e_1 in T_1 with e (at worst keeping the cost the same)
7. Repeat until T_1 is the same as T: **contradiction!**

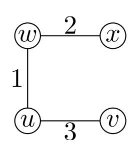
Consider the following graph:



Kruskal's algorithm produced a minimum spanning tree T for this graph as follows:



A different spanning tree T_1 is:



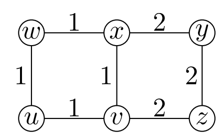
According to the proof, the edge $e_1 = (u, v)$ in T_1 can be replaced by $e = (w, x)$ but e is already in T_1. Oops.

Second Wrong Proof

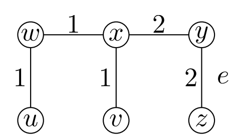
This proof comes from our textbook, Epp p.706 (4th Edition):

1. We already know Kruskal's finds a spanning tree T of G.
2. Let T_1 be a minimum spanning tree (MST) of G that has the most edges in common with T and assume $T_1 \neq T$.
3. There is an edge e in T that is not in T_1.
4. Adding edge e to T_1 produces a cycle. Let e_1 be an edge of this cycle that is not in T.
5. The weight of e is at most the weight of e_1 because at the time that Kruskal's added e, e_1 was also available to be added "*[since it was not already in T, and at that stage its addition could not produce a circuit since e was not in T]*"
6. Replace e_1 in T_1 with e to get a MST that is closer to T. **contradiction!**

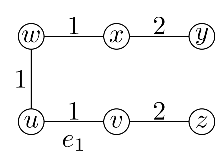
Consider the following graph G:



Kruskal's algorithm produced the following MST T for G:



A different spanning tree T_1 is:



According to the proof, the weight of edge $e = (y, z)$ in T is at most the weight of $e_1 = (u, v)$, but that is not true. Oops.

Correct Proof

From Wikipedia:

Let G be a connected, edge-weighted graph and K be the subgraph of G produced by Kruskal's algorithm. K contains no cycle (by design) and K is connected since the first (lowest weight) edge that joins two components of K would have been added by Kruskal's. Thus K is a spanning tree of G.

Loop invariant: At every iteration, the set, F, of edges chosen by Kruskal's so far is a subset of the edges of some minimum spanning tree of G.

This is true at the start of Kruskal's when $F = \emptyset$. Let's assume that it's true up to iteration $i-1$ and we'll show that it's true at iteration i. Let F be the set of edges at iteration $i-1$ and T be a minimum spanning tree that contains F. If the ith iteration adds no edge to F or adds an edge already in T to F then there's nothing to prove. So suppose $e \notin T$ is added to F. Since T is a spanning tree, $T + e$ contains a unique cycle C. Let f be an edge in C but not in F. (Since F contains no cycle, f must exist.) Since T is a minimum spanning tree and $T - f + e$ is a spanning tree, $w(e) \geq w(f)$. Since $F + f \subseteq T$, $F + f$ does not contain a cycle so Kruskal's must not have considered f yet, implying that $w(e) \leq w(f)$. Thus, $T - f + e$ is a minimum spanning tree containing $F + e$.

In particular, the invariant holds when F becomes a spanning tree, which eventually happens (see above).