Algorithms and Complexity
Homework Assignment 1

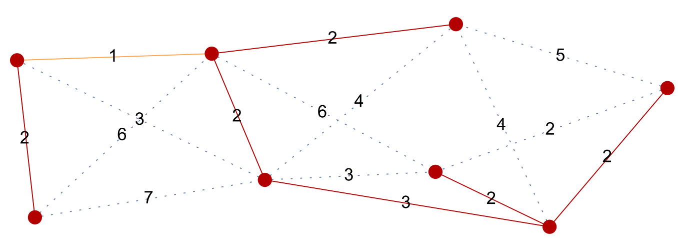
Dumb Ways to Die

2015-10-07

1 Minimum Spanning Trees

Exercise 1.1[Good sets and the Cut Lemma]

1. If X is good and there is a cut $S, V \setminus S$ such that (i) no edge from X crosses this cut and (ii) e is a minimum weight edge of G crossing this cut, then $X \cup e$ is good.

   $X =$ 

   In the picture above we can find that X is good. There is an edge e connecting the two connected components with a weight of 1. It is obvious that $X \cup e$ is also good.

   Proof. Since X is good, there is a minimum spanning tree T such that $T \supseteq X$. If we insert edge e into T, we will get a cycle. There must be another edge e' in this cycle and crossing the cut.

   From our assumption, e is the minimum weight edge of G crossing the cut, we can replace e' by e, then we get another minimum spanning tree. So that $X \cup e$ is good. □

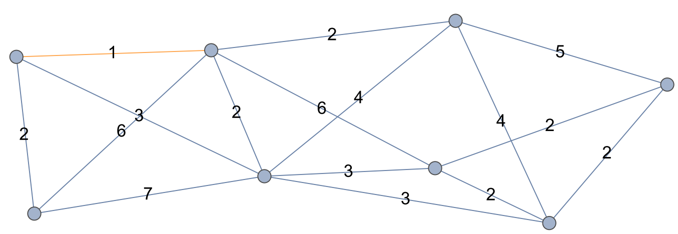
2. *Proof.* Since $X \cup e$ is good, there is a minimum spanning tree T such that $X \cup e \subseteq T$.

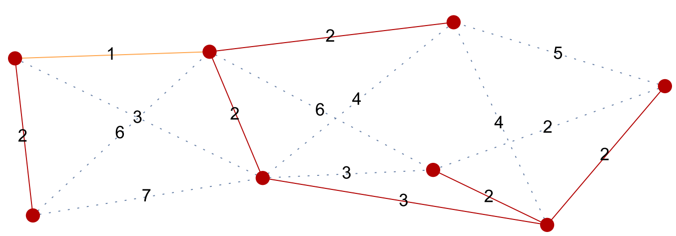
   Then every edge in T is a cut edge. If we remove e from T, we will get two connected components A, B. It is obvious that A, B is also a cut of X. So that e is an edge crossing the cut.

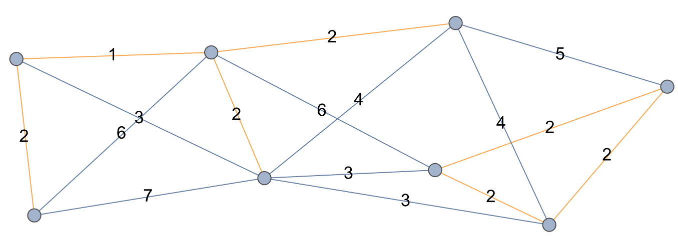
   To prove it is the minimum one, we assume it is not the minimum one. Then there is another edge e' whose weight is strictly smaller than e and crossing the cut A, B.

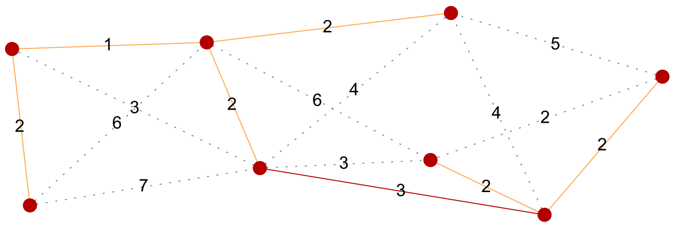
   Because A and B is a cut of the final minimum spanning tree, we can get that $T \setminus \{e\} \cup e'$ which is a spanning tree of even smaller weight. There comes a contradiction. $\square$

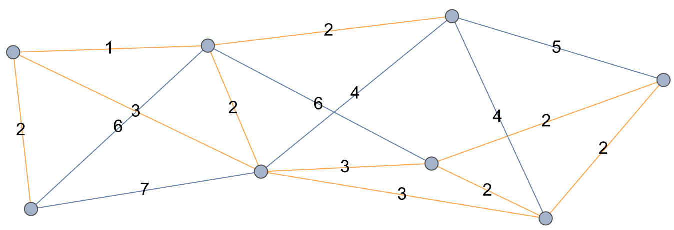
Exercise 1.2

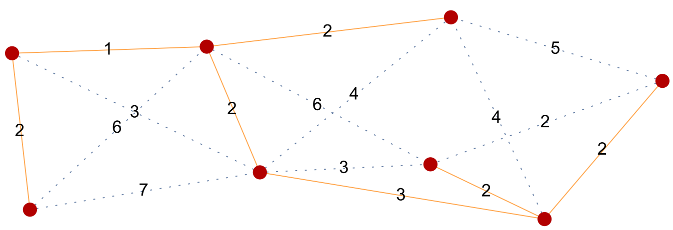
1. $G_1 =$ 

   $T_1 =$ 

   $G_2 =$ 

$T_2 =$ 

$G_3 =$ 

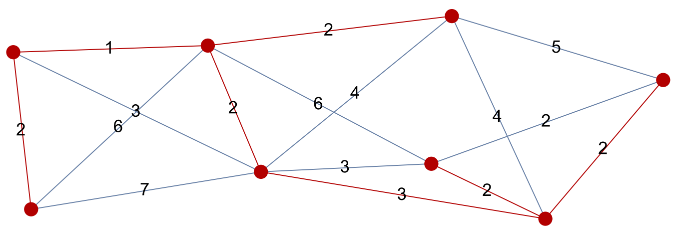
$T_3 =$ 

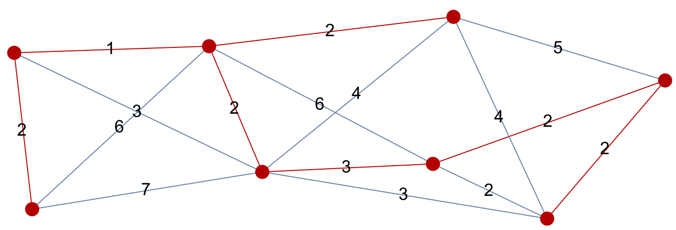
Obviously they have exactly the same connected components.

2. *Proof.* It is obvious that if two vertices $u, v \in V$ are connected in T_c, they will also be connected in G_c. So we have to prove that if two vertices $u, v \in V$ are connected in G_c, they will also be connected in T_c.

   Suppose there is an edge e in G_c connecting u and v, but u and v are not connected in T_c. Therefore, e should be add to T_c in order to construct a minimum spanning tree. There comes a contradiction. So e must be in T_c and u, v are connected in T_c. $\square$

Exercise 1.3

1. $T =$ 

$T' =$ 

$m_1(T) = m_1(T') = 1$
$m_2(T) = m_2(T') = 6$
$m_3(T) = m_3(T') = 7$

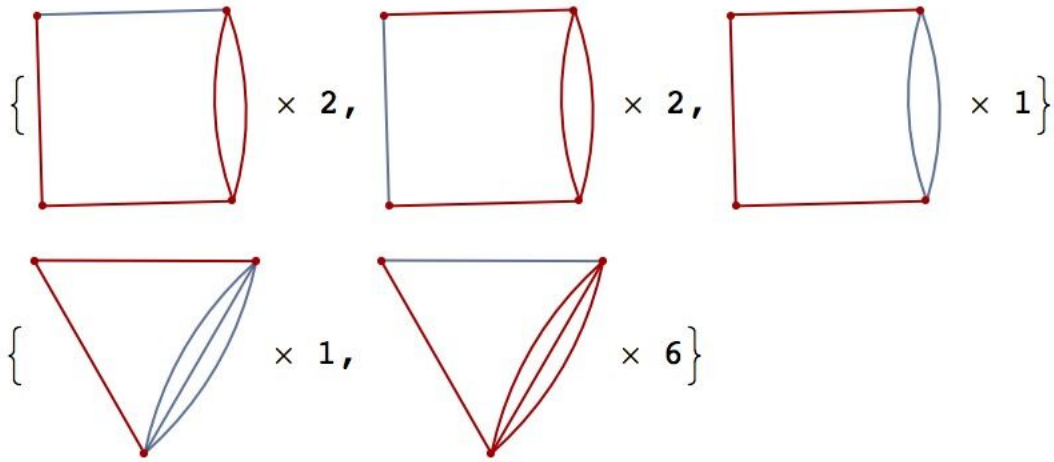
2. *Proof.* Let T and T' be two minimum spanning trees of G.
According to lemma 2, T_c, T'_c and G_c have exactly the same connected components.
Thus the number of connected components in T_c, T'_c and G_c should be the same.
$m_c(T) = m_c(T') = |G_c|$ - the number of connected components in G_c. □

Exercise 1.4

Proof. Assume that there are two different minimum spanning trees T and T' in G.
According to lemma 4,
$m_c(T) = m_c(T')$ holds for any c
$\Rightarrow m_c(T)$ - $m_{c-1}(T) = m_c(T')$ - $m_{c-1}(T')$ holds for any c
$\Rightarrow$ the number of edges of weight c in T and T' must be exactly the same.
If T and T' are different, there must be two different edges with the same weight.
But no two edges of G have the same weight. There comes a contradiction.
Thus $T = T'$, G has exactly one minimum spanning tree. □

Exercise 1.5

Obviously, the two connect components are independent. So we can get the answer by multiply the answers of the two connect components.



Thus, there are $(2 + 2 + 1) * (1 + 6) = 35$ different spanning forests.

Exercise 1.6

The algorithm works as follows.

1. We have to use a disjoint set S. Let f be the mapping from V to V in S, which tells us that which connected component every vertex belongs to.
   Initially all vertices are in V, all edges are in E and $ans = 1$.

2. Let E' and V' be two empty sets.
   Assume that the weight of the edges with the smallest weight in E is c.
   For every edge (u, v) in E with a weight of c, add $(f(u), f(v))$ to E', add $f(u), f(v)$ to V', and then removes (u, v) from E.

3. Use the subroutine to compute the number of spanning forests "*count*" of (V', E') with disjoint set S.

4. $ans \leftarrow ans \times count$
   If E is not empty, go to Step 2.

Obviously the algorithm above is correct because according to lemma 2, which edge we choose does not affect the connected components in graph. So the choices of edges with different weights are independent, and we can multiply them directly to get the number of minimum spanning trees.