# CS217 - Algorithm Design and Analyis
# Homework Assignment 3

**Group Name:** Static
**Group Members:** Zhenjia Xu, JiaCheng Yang
Zhuolin Yang, Wenda Qiu

1st Nov, 2016

## 3  Minimum Spanning Trees

**Exercise 3.1.**

**Lemma 1** (**Cut Lemma**). *Let $X \subseteq E$ be a good set. If $(V, X)$ is not a spanning tree, then $(V, X)$ consists of two or more connected components. Let $V = S \cup \overline{S}$ be a cut of $X$. That is, no edge of $X$ goes from $S$ to $\overline{S}$. Let $e \in E$ be an edge of minimum cost connecting two connecting components of $(V, X)$. Then $X \cup \{e\}$ is good, too.*

*Proof.* Suppose the $X \cup \{e\}$ is not a good set. Then all the spanning trees exclude the minimum edge which connects two connecting components of $(V, X)$. Denote these two connecting components as $C_1$ and $C_2$. Pick up one of the minimum spanning trees $T_m$ and substitute the edge $e_t \in E(T_m)$ connecting $C_1$ and $C_2$ to $e_m$.
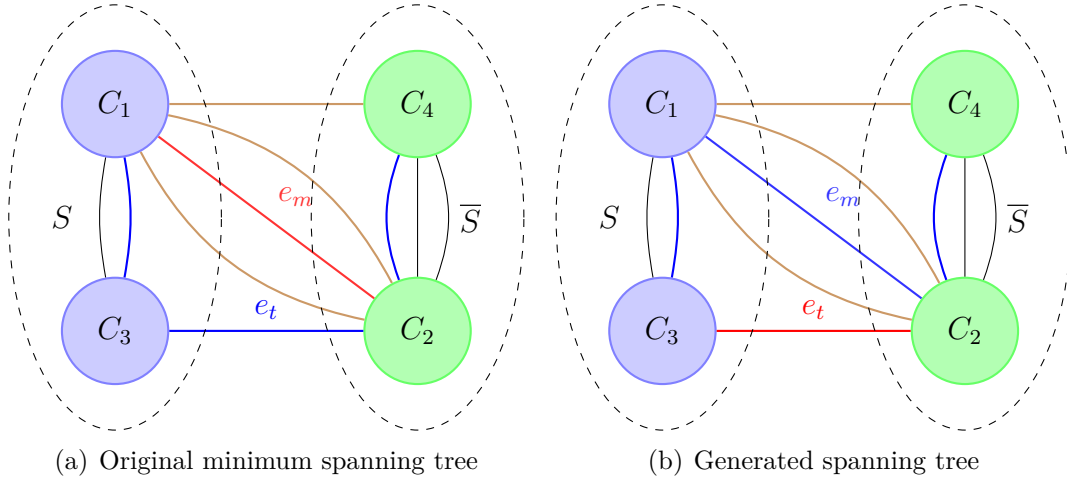


(a) Original minimum spanning tree      (b) Generated spanning tree

Figure 1: The illustration of the proof

Then the "costs" of the new spanning tree $T'_m$ is:

$$c(T'_m) = \sum_{e \in E(T')} c(e) = c(T_m) - c(e_t) + c(e_m)$$

Using the condition $e_m$ is the minimum edge connecting $C_1$ and $C_2$, we can easily know that:

$$c(e_m) \leq c(e_t)$$

Then we obtain a more optimal spanning tree(or we obtain another minimum spanning tree containing the edge $e_m$) which leads the contradictory according to the inequality below:

$$c(T'_m) = c(T_m) - c(e_t) + c(e_m) \leq c(T_m)$$

Hence, $X \cup \{e\}$ is good. □

**Lemma 2 (The Inverse of Cut Lemma).** *If $X$ is good, $e \notin X$, and $X \cup \{e\}$ is good, then there is a cut $S$, $V\backslash S$ such that the following two holds:*

1. *no edge from $X$ crosses this cut;*

2. *$e$ is minimum weight edge of $G$ crossing the cut.*

*Proof.* Because $X \cup \{e\}$ is good, there exists a minimum spanning tree $T$ of $G$ such that $X \cup \{e\} \subseteq E(T)$. In this spanning tree $T$, $\{e\}$ connects two components, denoted by $S, V \setminus S$.
Consider the cut of G: $S, V \setminus S$.

(i) Obviously, for each edge $e(u,v)$ in $X$, either $u, v \in S$ holds or $u, v \in V \setminus S$ holds; that is to say it can't crosss the cut.

(ii) If $e$ is't a minimum weight edge of $G$ crossing this cut; that is to say, there exist a smaller edge crossing $G$, denoted by $e'$.
Then $T \setminus \{e\} \cup \{e'\}$ forms a smaller spanning tree, which contradicts the condition that $T$ is the minimum spanning tree.
So $e$ is a minimum weight edge of $G$ crossing this cut.

□

**Exercise 3.4.** The example is illustrated below, in which we can see that no matter which minimum spanning trees we choose, the connected components is indentical.
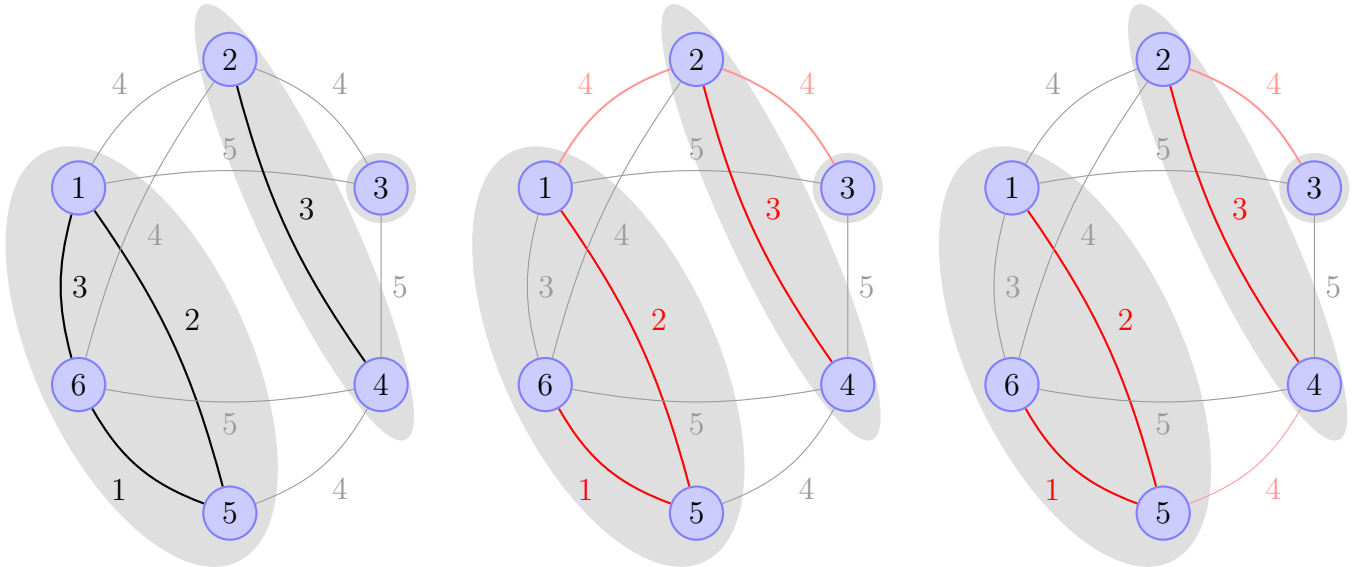


(a) The original graph and its connect components only considering the edges whose weight is less than 4.

(b) A possible minimum spanning tree and its connect components only considering the edges whose weight is less than 4.

(c) Another possible minimum spanning tree and its connect components only considering the edges whose weight is less than 4.

Figure 2: The illustration of the proof

Consider $T$ itself is a subgraph of $G$, so the only case is: there exists $u, v \in V$ such that $u, v$ are not connected in $T_c$, but are connected in $G_c$. Because the $u, v$ are not connected in $T_c$, so there must be a edge $e_1$ which $w(e_1) > c$ on the path from $u$ to $v$ in $T$. This edge $e_1$ will link two components of $T$ together, let's call them $T_1$ and $T_2$.

*Proof.* There must exists a edge $e_2 = (x, y)$ in $G_c$ which $x$ is belong to $T_1$ and $y$ is belong to $T_2$. Consider $u, v$ are connected in $G_c$, so there must exist a edge which can combine these two components together, otherwise $u, v$ will in two different components, and be isolated.

So, after adding this edge $e_2$ into $T$, there must form a circle in $T$. According the Cut Lemma, this edges can replace $e_1$ (for $w(e_2) \leq c < w(e_1)$), so $T$ is not the Minimum Spanning Tree of $G$, which is a contradictory. so the Lemma 3 is correct. $\square$

**Exercise 3.7.** The example is illustrated below, in which we can see that no matter which minimum spanning trees we choose, the $m_3$ of each minimum spanning trees is indentical. The edges colored red are the minimum spanning tree.



(a) A possible minimum spanning tree

(b) Another possible version of the minimum spanning tree

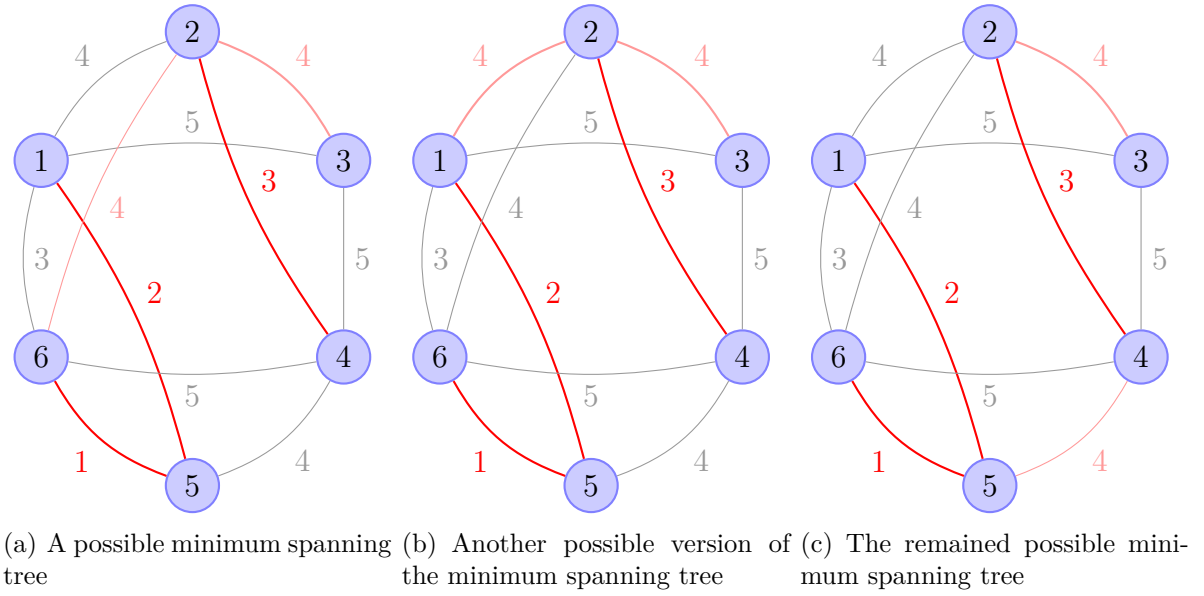(c) The remained possible minimum spanning tree

Figure 3: Three possible minimum spanning trees and its $m_3$

The proof is showed below:

*Proof.* Because $T$ and $T'$ are two minimum spanning trees of $G$, according to Lemma 3, $T_c$ and $T'_c$ and $G_c$ have exactly the same components. Consider the number of components equals to $V - m_c(T)$, which $V$ is the total number of vertexs in $G$. if $m_c(T') \neq m_c(T)$, the number of components is not equal too, which is a contradictory. so $m_c(T')$ must equal to $m_c(T)$. $\square$

**Exercise 3.8.** Suppose that there are two different minimum spanning tree, called $T, T'$.

We can find the edge which belongs to only one of the minimum spanning tree and its weight is minimal. Because no two edges of $G$ have the same weight, the above edge is unique, denoted as $e$. (Without loss of generality, let $e$ belong to $T$)

By Lemma 3, we known that

$$m_{c_e}\{T\} = m_{c_e}\{T'\} \text{ and } m_{c_e-1}\{T\} = m_{c_e-1}\{T'\}$$

So

$$m_{c_e}\{T\} - m_{c_e-1}\{T\} = m_{c_e}\{T'\} - m_{c_e-1}\{T'\}$$

However, $e$ belongs to only $T$, in other word

$$m_{c_e}\{T\} - m_{c_e-1}\{T\} = 1 \text{ and } m_{c_e}\{T'\} - m_{c_e-1}\{T'\} = 0$$

which contradicts the equation above.

So, the hypothesis is wrong; that is to say $G$ has only one minimum spanning tree.

**Exercise 3.10.** The two parts of the graph are independent, so we only need to caltulate the minimal spanning tree of the two parts respectively.

- Left part:

  If we don't choose any of the parallel edges(containing three edges), we need to choose the other two edges.(1 choice).

  If we choose one of the parallel edges(3 choices), we only need to choose one edge from the other two edges(2 choices).

  In summary, we have $1 + 3 \times 2 = 7$ minimal spanning trees in the left part.

- Right part:

  Using the similar method, the right part have $1 + 2 \times 3 = 7$ minimal spanning trees in the right part.

- Combining two parts:

  Using Multiplication rule, we can get the total amount of the minimal spanning forests, namely $7 \times 7 = 49$

We also justify the answer by programming(enumerating every set of the edges, and check if it can form a minimal spanning forest):

```python
u = [1, 1, 2, 2, 2, 4, 4, 5, 6, 4] # Edge set
v = [2, 3, 3, 3, 3, 5, 5, 6, 7, 7]

def getfa(x) :
    if (f[x] == x) :
        return x
    else:
        f[x] = getfa(f[x])
        return f[x]

if __name__ == '__main__' :
    answer = 0
    for mask in range(1, 1 << 10) : # Enumerate every possible edge sets.
        flag = 1
        number = 0
        f = [i for i in range(8)]
        for i in range(0, 10) :
            if (((mask >> i) & 1) == 1) :
                number = number + 1
                if (getfa(u[i]) == getfa(v[i])) :
                    flag = 0
                else:
                    f[getfa(u[i])] = getfa(v[i])
        if (number == 5 and flag) :
            answer = answer + 1
    print answer
```

4

**Exercise 3.11.** Similar like Kruskal, we sort all the edges by their weight $w(e)$, then we process all the edges with the same weight in increasing order. We repeat the following works until all the edges have been processed.

- Define the smallest unprocessed edge weight as $c$. Label all the edges with $w(e) = c$ as processed.

- Use the given algorithm to caculate the number of spanning forests $N_c$ for a subgraph $g = (V, \{e \in E, w(e) = c\})$.

- According to Lemma.3, the connected components of a subgraph $G_c = (V, e \in E, w(e) \leq c)$ are the same in spanning tree $T_c$. We can arbitrarily pick a spanning forest of $g$ and combine all the nodes in the same connected component into one node.
  More specificly, we define a function:
  $$f(u) = u'$$
  where $u'$ is the node which has the smallest index in connected component containing $u$.
  Then we modify $G$ into $G'$:

  $$G' = (\{v \in V, f(v) = v\},$$
  $$\{e' = (u', v', w') | e = (u, v, w) \in E, w' = w, u' = f(u), v' = f(v), u' \neq v'\})$$

Finally, we get the number of spanning trees of a weighted tree:

$$N = \prod N_c$$