COMP1201 Assignment 3

Q1:

Prove that in any Facebook community, there exists two people who have the same number of friends.

Assumptions:

1) There are *n* people in the community.

2) A person *cannot* be friends with themselves.

3) Assuming that there *cannot* be a person with *0* friends. That means that the maximum number of friends a person can have is *n-1.* In this case, because friendship is symmetric, everybody else must at least know this person, and the minimum number of people a person can know is *1.* This gives us the set *{1, 2, ..., n-1}* which represents the possible number of people each person can know.

4) Assuming that there *can* be a person with *0* friends. This means that the maximum number of friends a person can have is *n-2*. Following the logic from *assumption 3*, this gives us the set *{0, 1, …., n-2}*.

Both of the sets, described in *assumption 3* & *assumption 4,* have *n-1* elements. These elements describe all the *possibilities* for the number of people each person can have as a friend.

Every single person must be assigned one of these *n-1* possible numbers. But since there are *n* people (*assumption 1*), one of these numbers must be used twice due to the pigeonhole principle.

This proves that there are at least two people that have the same number of friends.

Example:

There are 20 people (*n = 20*). This means that there are 19 (*n-1*) possibilities of how many friends each person can have.

$$\lceil 20 / 19 \rceil = 2$$

We can see that there are at least two people with the same amount of friends.

Prove that any simple graph *G* with at least two vertices must contain two vertices of the same degree.

Assumptions:

1) A *simple* graph is unweighted and undirected. It cannot contain graph loops or multiple edges.

2) Since we have a *simple* graph, there can be *no edges*.

3) There are *n* vertices in a graph.

4) Since we can have a vertex with a degree of *0* (*assumption 2*), this gives us the set *{0, 1, …., n-2}* which represents the possible degrees a vertex can have.

Because there are *n-1* elements in the set (*assumption 4*) of possible degrees, and every vertex (of which there are *n*) must have a degree, using the pigeonhole principle, we can see that there are at least two vertices must have the same degree.

Example:

There are 4 vertices (*n = 4*). This means that there are 3 (*n-1*) possibilities of what degree each vertex can be.

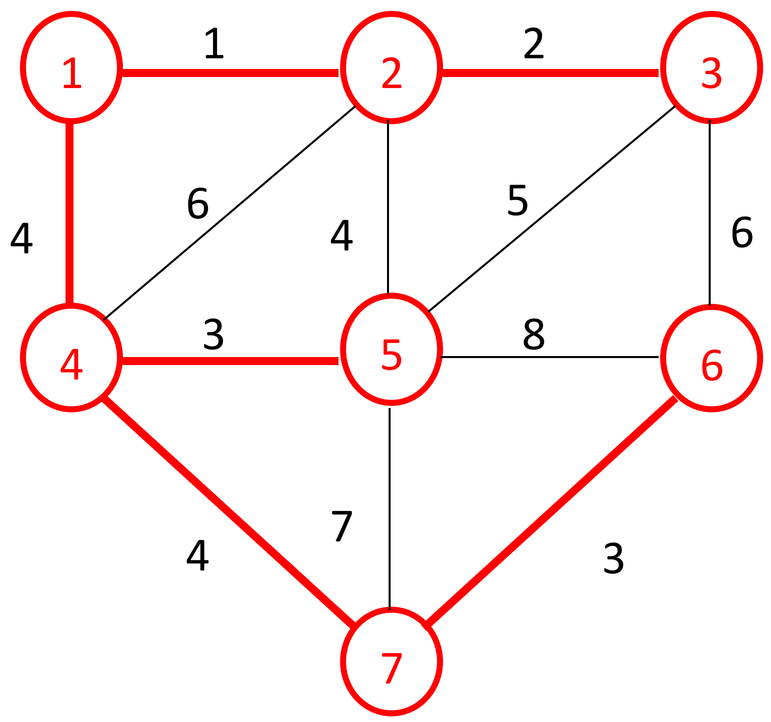
$$\lceil 4 / 3 \rceil = 2$$

We can see that there are at least two vertices with the same degree.

Q2:

(A) Consider a minimum spanning tree (MST) of a connected, weighted graph. If we remove an edge (u, v) of the MST, then we get two separate trees.

Example MST:



Are these two trees the MSTs on their respective sets of nodes?

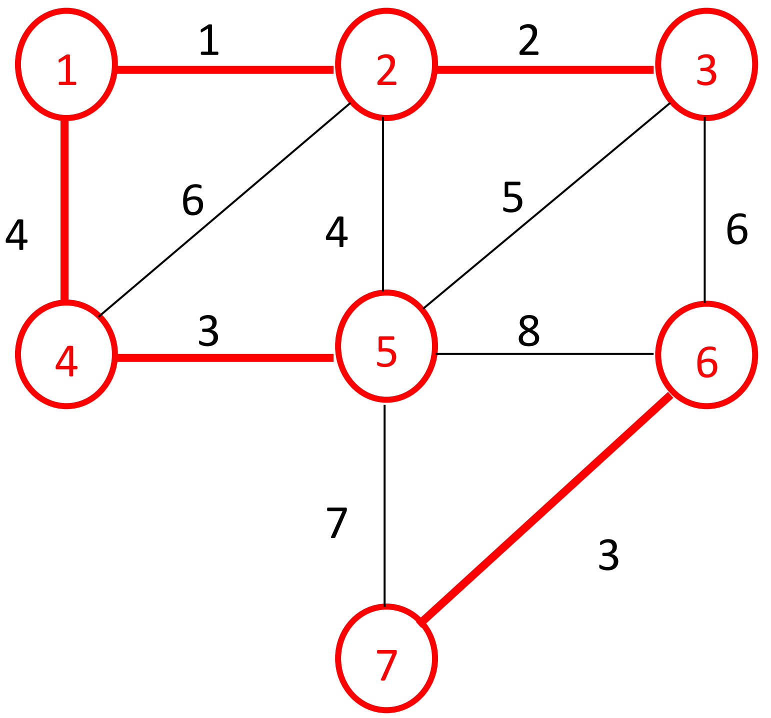
Is the edge (u, v) a least-weight edge crossing between those two sets of nodes?

1) We cannot remove edges (6, 7), (4, 5) or (2, 3) because if we do remove them, we will not be left with trees (as for a set of vertices to be considered a tree, there must be at least 2 vertices).

Note: If we do remove the edges mentioned above, we can still get the MST of the graph by rerunning the MST finding algorithm.

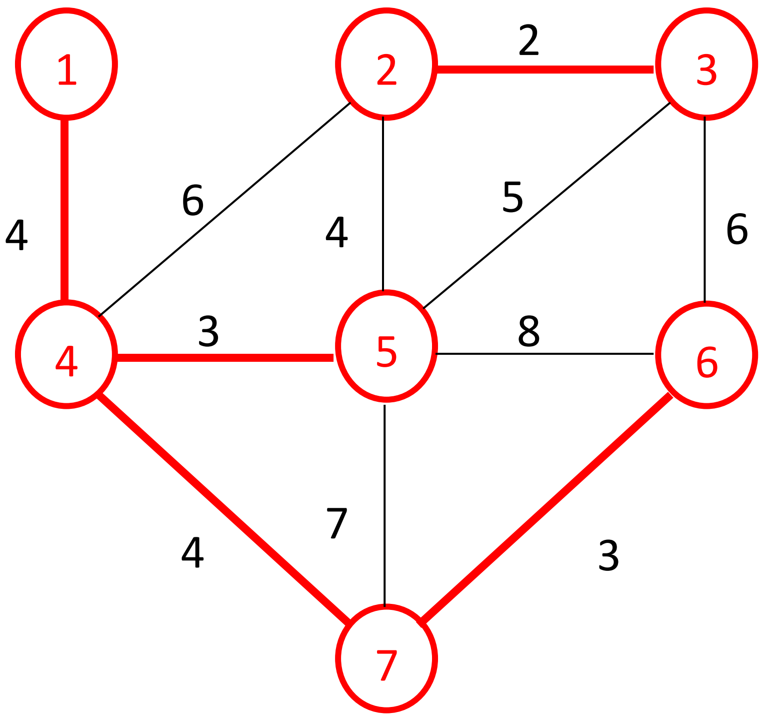
2) If we remove an edge, which is not listed in 1) and is not the least-weight edge crossing between the two sets of nodes (edge (1, 2)), we get two trees, which *are* the MSTs on their respective sets of nodes.

Example: After removing edge (4, 7)



We have two trees, which are the MSTs on their respective sets of nodes.

3) If we delete the least-weight edge crossing between the two sets of nodes (edge (1, 2)) we still get two trees, which are the MSTs on their respective sets of nodes.



(B) Consider the following algorithm for finding a MST on a graph:

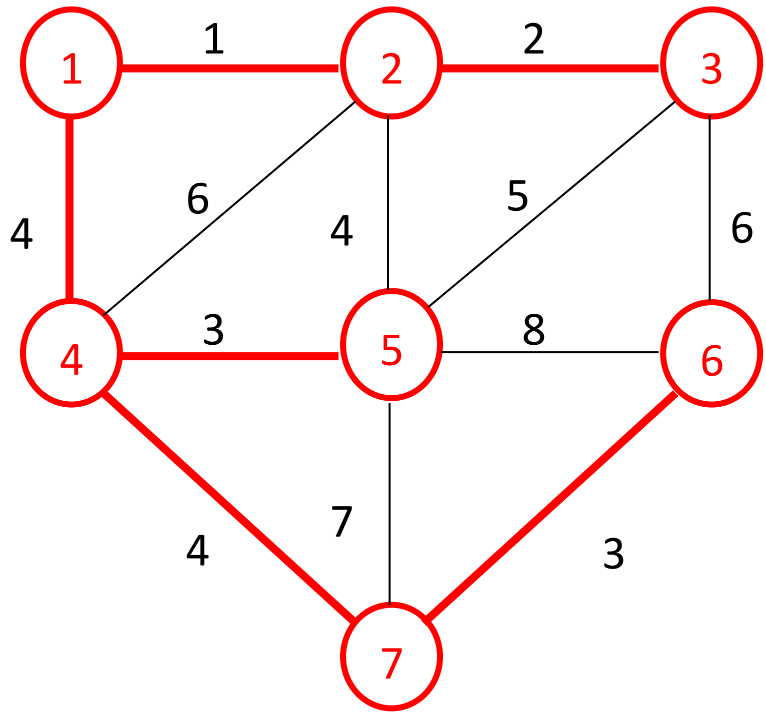
- Split the nodes of the graph arbitrarily into two nearly equal-sized sets.

- Find a MST on each of those sets.

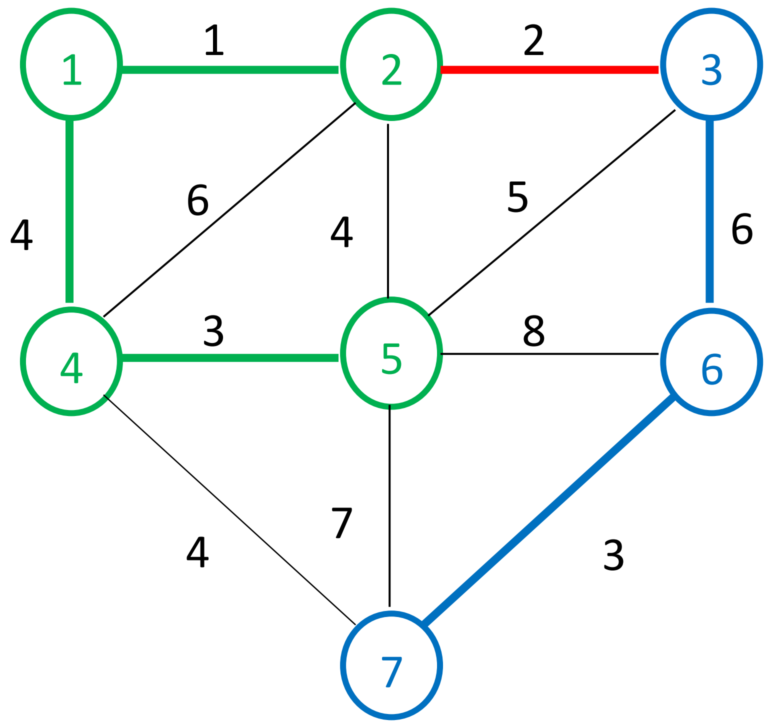
- Connect the two MSTs with the least-cost edge between them.

Would this algorithm always return a MST of the original graph?

- *Answer*: It will not always return a MST of the original graph.
- *Example*: A normal MST:



Now we get the graph, without the MST, and split the 7 vertices arbitrarily into two nearly equal-sized groups (marked by green and blue):



With our example tree we have a total weight of **17** of the MST.

Using the algorithm described in *Question 2 (B)* we have a total weight of **19** of the MST.

This proves that the algorithm will not always return a MST of the original path.

Q3:

(A) *What we are given:*

An automotive company has **three factories**, each associated with different costs of producing vehicles – the cost of **raw materials**, the cost of **labour** and the **environmental footprint** per vehicle produced

	Materials	Labour	CO2 emissions
Factory 1	6	18	10
Factory 2	5	14	17
Factory 3	7	11	20

Each month the **company** can afford to pay for **4000 hours of labour** and **4000 units of raw materials**. Additionally, labour politics require **at least 100 cars to be produced at Factory 1** and environmental regulations allow the company to emit **at most 3000 units of CO_2**.

What we need to calculate:

What is the optimal allocation of manufacturing capacity between the factories (i.e. **how many vehicles** need to be produced by each factory to ensure that the company produces the **maximum possible number of vehicles each month**)?

Answer:

We will rename:

- Factory 1 = x_1
- Factory 2 = x_2
- Factory 3 = x_3

Equations:

The maximum amount of units of raw materials can be no more than 4000:

- (6*x_1) + (5*x_2) + (7*x_3) <= 4000

The maximum amount of labour hours can be no more than 4000:

- (18*x_1) + (14*x_2) + (11*x_3) <= 4000

The maximum amount of CO_2 can be no more than 3000:

- (10*x_1) + (17*x_2) + (20*x_3) <= 3000

The factories cannot produce less than 0 vehicles:

- x_1 => 0
- x_2 => 0
- x_3 => 0

Factory 1 (x_1) must produce at least 100 vehicles:

- x_1 >= 100

We want to maximise the amount of vehicles we can produce:

- maximise: x_1 + x_2 + x_3

Writing down the whole equation we get:

Maximise:

x_1 + x_2 + x_3

Subject to:

(6*x_1) + (5*x_2) + (7*x_3) <= 4000

(18*x_1) + (14*x_2) + (11*x_3) <= 4000

(10*x_1) + (17*x_2) + (20*x_3) <= 3000

x_1 >= 100

x_1 => 0 (this is obsolete because x_1 must be “=> 100” and 0 < 100)

x_2 => 0

x_3 => 0

Solving the equation we get:

- Factory 1 (x_1) must produce 188 vehicles.
- Factory 2 (x_2) must produce 0 vehicles.
- Factory 3 (x_3) must produce 56 vehicles.

The total amount of vehicles that can be produced, given the restrictions, is

- 188 + 0 + 56 = 244

(B)

What we are given:

Suppose that labour politics changes and the company is required to produce at least 75 vehicles at each factory (at least 225 vehicles in total), other requirements remaining in place as before.

What we need to calculate:

What would be the effect of such a development?

Answer:

We will rename:

- Factory 1 = x_1
- Factory 2 = x_2
- Factory 3 = x_3

Equations:

The maximum amount of units of raw materials can be no more than 4000:

- (6*x_1) + (5*x_2) + (7*x_3) <= 4000

The maximum amount of labour hours can be no more than 4000:

- (18*x_1) + (14*x_2) + (11*x_3) <= 4000

The maximum amount of CO_2 can be no more than 3000:

- (10*x_1) + (17*x_2) + (20*x_3) <= 3000

The factories cannot produce less than 0 vehicles:

- x_1 => 0
- x_2 => 0
- x_3 => 0

All factories (x_1, x_2, x_3) must produce at least 75 vehicles:

- x_1 >= 75
- x_2 >= 75
- x_3 >= 75

We want to maximise the amount of vehicles we can produce:

- maximise: x_1 + x_2 + x_3

Writing down the whole equation we get:

Maximise:

x_1 + x_2 + x_3

Subject to:

(6*x_1) + (5*x_2) + (7*x_3) <= 4000

(18*x_1) + (14*x_2) + (11*x_3) <= 4000

(10*x_1) + (17*x_2) + (20*x_3) <= 3000

x_1 >= 75

x_2 >= 75

x_3 >= 75

x_1 => 0 (this is obsolete because x_1 must be “=> 75” and 0 < 75)

x_2 => 0(this is obsolete because x_2 must be “=> 75” and 0 < 75)

x_3 => 0(this is obsolete because x_3 must be “=> 75” and 0 < 75)

Given the restrictions and requirements, each factory producing at least 75 vehicles is infeasible.

We can also prove that it is infeasible just by calculating the CO_2 emissions each factory would produce by making 75 vehicles each:

(10*x_1) + (17*x_2) + (20*x_3) <= 3000

(10*75) + (17*75) + (20*75) <= 3000

750 + 1275 + 1500 <= 3000

3525 <= 3000 which is never true. Thus proving that each factory producing at least 75 vehicles is infeasible.

Used software for calculating: Mathematica