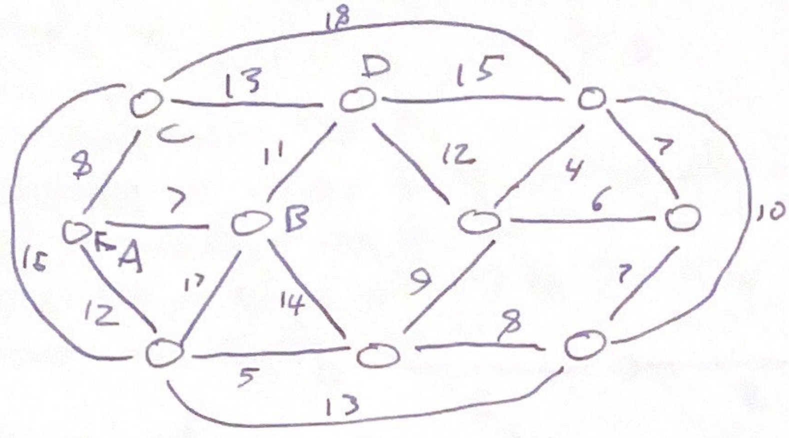
Homework 3

Nathan Bick

1



Using the above network answer the following questions.

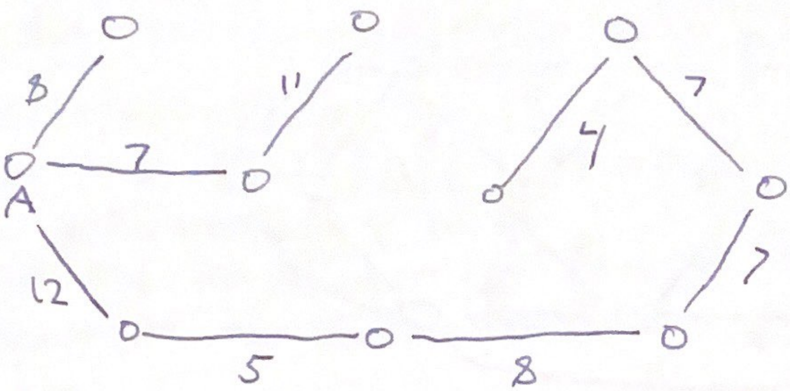
a) Solve for the minimum spanning tree using Prim's algorithm frome node A.

b) solve for the minimum spanning tree using Kruskal's Algorithm.

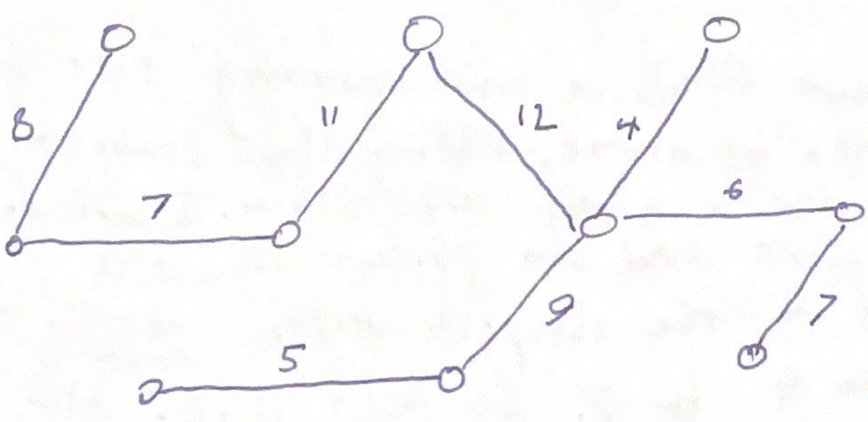
c) Apply Christophide's Algorithm.

a) Prim's algoritum finds a min spanning tree by building this one vertex at a time, adding the lowest cost edge vertex. It is a greedy algorithm. I will describe the first iterations and then present the tree.
If we start at A, the cheapest vertex not in the tree is connected to A by 7. So add that! call it B Now, the cheapest vertex is connected to the tree thru A by 8, so add that. Call it C. Now, there are many options, But the cheapest connection to a new node is from B via 11 to D. so add that.
The min span tree is shown →

Below is a min spanning tree. Note that we have a choice of edges with length 12 and can arbitrarily pick between them.

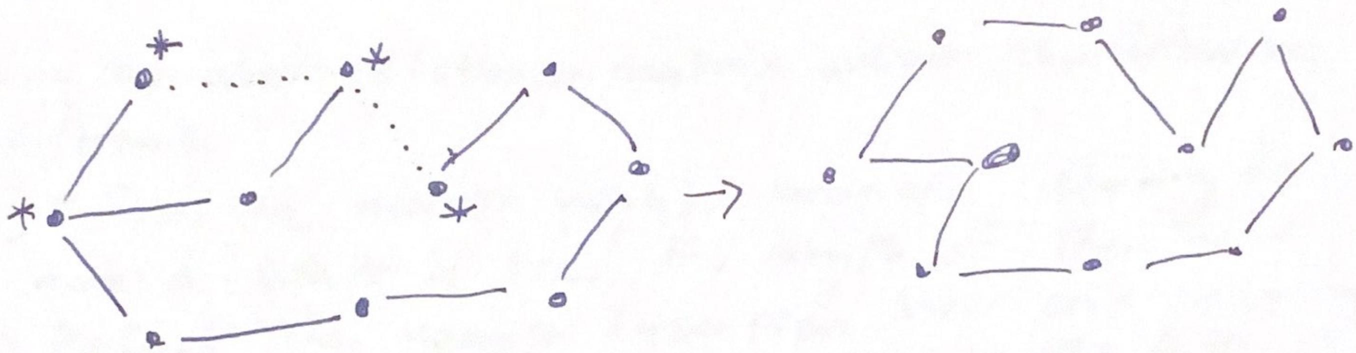


b) Now we use Kruskal's Algorithm. It is a similar greedy algorithm to Prim's, but it relaxes the requirement to search for an edge/vertex that can be connected to the tree as it has been built up to that point. This temporarily allows for disjoint trees until the last step.



c) To use Christofede's Algorithm, we start with a min spanning tree. We can use the one we got in part a. The algorithm in general is given by

1. Find a min spanning tree T.
2. Let O be set of nodes with odd degree in T. Find a minimum cost perfect matching M on O.
3. Add the set of edges of M to T. Find an Euler tour.
4. Shortcut the Euler tour



From/to	A	B	C	D	E	F
A	X	13	9	7	5	12
B	13	X	4	5	8	15
C	9	4	X	1	3	5
D	7	5	1	X	2	6
E	5	8	3	2	X	9
F	12	15	5	6	9	X

Using the above distance matrix answer the following questions.

a) Perform the nearest neighbor heuristic starting from node A. What is the full length of the tour.

b) Perform the nearest insertion heuristic starting with the cycle $A \to E \to A$. What is the full length of the final tour?

c) Would the Christofides Algorithm guarantee of $<$ $1.5 \cdot$ optimal hold for this problem? (You don't need to solve to answer)? Why or why not?

d) Using the distance matrix, can you describe a simple way to bound the shortest and longest tours you could make?

a) The nearest neighbor algorithm is a greedy one; at each step, we go to the nearest unvisited node.

starting at A, the nearest is E with distance 5.

A → E. From E, the nearest is D. ~~A~~ → E → D. From D, the nearest is C. A → E → D → C. From C, go to B. ~~three~~ From B, we can only go to F, and then we have:

$$\underset{5}{A \to E} \underset{2}{\to D} \underset{1}{\to C} \underset{4}{\to B} \underset{15}{\to F} \underset{12}{\to A}$$

b) To implement nearest insertion, we start with a tour, and add the unvisited node that is closest to any one of the nodes in the current tour, repeatedly,

starting with A → E → A, the nearest to A or E is now D, distance 2. We have A → E → D → A. The nearest is now D to C. We have A → E → D → C → A. Next we add C to B: A → E → D → C → B → A. Now add C to F. we have A → E → D → C → F → B → A.

c) The christofedes Algorithm ~~is applicable~~ gives bound of 1.5 • optimal for metric space (symmetric and triangle equality holds). We check these two for the table. we see the table is indeed symmetric.

The triangle inequality states that the sum of two sides of a triangle must be greater than or equal to the length of the other side.

d) A way to bound the longest tour is the sum of all paths. This would be a tour with redundant or repeated visits, but would be the longest and a high upper bound.

A lower bound could be estimated using the shortest collection of edges that are not necessarily valid cycles. The shortest valid cycle would be at least as long as that.