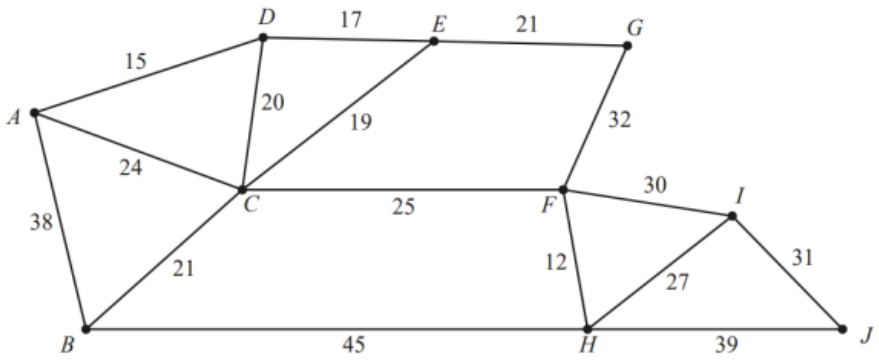
3500 hw 5

Casey Provitera

March 2024

1 (40 points) Minimum spanning tree

1. In the diagram, we have 10 cities, and the distances between them are measured in kilometers. Our objective is to establish an electricity network that delivers power to these cities while minimizing the cost of the wires required for the connection.
   To achieve this, we need to find a minimum spanning tree for the city shown in the diagram. The process should clearly illustrate the order in which you selected the edges for your tree. You should accomplish this using:



   Find a minimum spanning tree for the network in the diagram above, showing clearly the order in which you selected the edges for your tree, using:

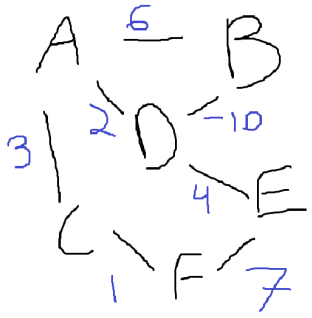
   (a) Kruskal's algorithm
   This algorithm simply picks the lowest weighted edges that does not create a cycle and continues until all nodes are in a single tree.
   {(H,F), (A,D), (D,E), (E,C), (C,B), (E,G), (C,F), (H,I), (I,J)}

   (b) Prim's algorithm, starting from A
   This algorithm starts from a 'root' and explores the lowest weighted edge that is visible and explores a new node from that 'root', when

an edges is explore new edges are able to be seen. this continues until all nodes are in a single tree.
{(A,D), (D,E), (E,C), (C,B), (E,G), (C,F), (F,H), (H,I), (I,J)}

2. Can Prim's and Kruskal's algorithm yield different minimum spanning trees? Explain why or why not.
   Both Prim's and Kruskal's algorithms are meant to yield minimum spanning trees but they do so in slightly different fashions. For Prim's you are given a starting node and explore the smallest weighted edges that are connected to that node while you explore you repeat this process until all nodes are connected. But for Kruskal's we simply select the smallest weighted edges from the entire graph until all nodes are connected. These can lead to different solutions to spanning trees if there are multiple edges with the same weight in a single graph.

3. Can Prim's algorithm produce the correct Minimum Spanning Tree (MST) when negative edge weights are present? Explain the reasoning behind your answer.
   Prim's algorithm can produce a correct minimum spanning tree with negative edges but it is not always guaranteed to. Lets take for an example the following graph:



   For the above graph Prim's algorithm starting at node 'A' finds the following solution:
   {(A,D), (A,C), (C,F), (A,B), (F,E)}
   This solution has the total weight of the spanning tree as:
   2+3+1+6+7 = 19, but the real minimal spanning tree is the following:
   {(A,C), (C,F), (A,B), (B,D), (D,E)}
   This solutions has the total weight of the spanning tree as:
   3+1+6+(-10)+4=4, this is the true solution for the minimum spanning.
   Which was not found by Prim's algorithm showing it will not always find the right solution with negative edges.

4. Can Kruskal's algorithm produce the correct Minimum Spanning Tree (MST) when negative edge weights are present? Explain the reasoning

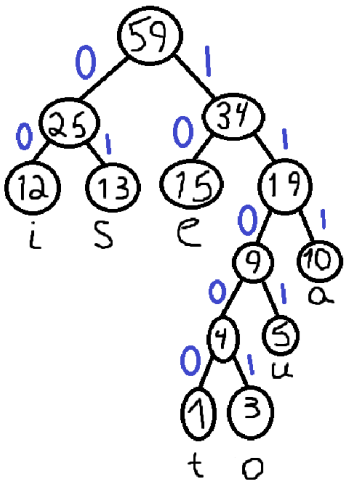
behind your answer.
Now Kruskal's will always find the correct solution of the minimum spanning tree on a graph, even when edges contain negative weights. This is because Kruskal's does not start at any node but simply picks the lowest weighted edges as it runs.

2 (40 points)Huffman Code

A file contains the following characters with the frequencies as shown.

Characters	Frequencies
a	10
e	15
i	12
o	3
u	5
s	13
t	1

- If you used fixed-length code to encode the message, how many bits do you need per character?
  3 bits
- If you used fixed-length code to encode the message, how many bits do you need to encode this message?
  3 bits * number of characters in the message
  3 * (10+15+12+3+5+13+1)
  3 * 59
  177 bits
- Create a Huffman encoding tree for the given file. Follow the algorithm taught in class precisely to obtain identical codes.



- What are the codes for each character after using Huffman coding? Fill the table.

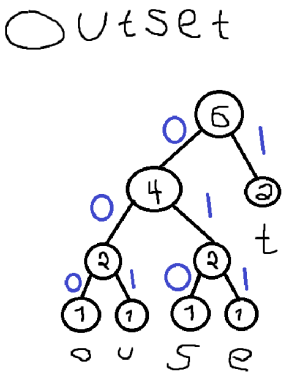
Character	Code	Number of Bits
a	111	3
e	10	2
i	00	2
o	11001	5
u	1101	4
s	01	2
t	11000	5

- What is the average number of bits per character after using Huffman coding?
  The average bits per character is each individual characters number of bits divide by the number of characters:
  total bits - 3+2+2+5+4+2+5=23
  total characters - 7
  average number of bits is 23/7 or 3.29.

- How many bits do you need to encode the file if Huffman codes are used?
  To figure out how many bits you need to encode the file you multiple each characters bits by its frequency.
  letter, bits, frequency a: 3*10 = 30
  e: 2*15 = 30
  i: 2*12 = 24
  o: 5*3 = 15
  u: 4*5 = 20

s: 2*13 = 26
t: 5*1 = 5
total bits = 150
Therefore we save 27 bits by using Huffman coding in this example.

- Encode the following word using Huffman codes
  Outset
  Huffman tree below:



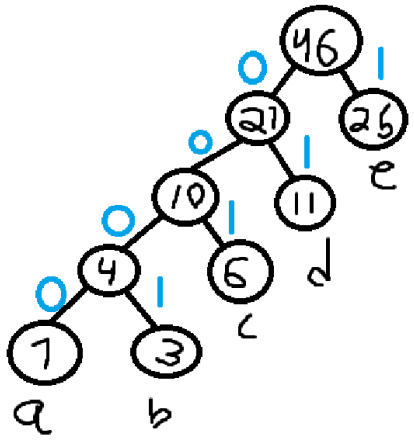
codes:
t - 1
e - 011
s - 010
u - 001
0 - 000

- Decode the following codes:
  11111011100011001
  auto
  01001100010
  site
  0110111
  sea
  101110111000
  east

3 (10 points)Huffman Code

Under a Huffman coding of n symbols with frequencies f1, f2, ..., fn, what is the longest a codeword could possibly be? Give an example set of frequencies that would produce this case.

The maximum length a code word can have after using Huffman coding of n symbols is n-1. This can happen when the frequency of the characters is extremely skewed. Meaning a single character shows up more than all other characters combined.
Take the following Huffman Coding tree as an example:



The reason the codes for a and b are so long is because in comparison to the other letters their frequency is so extremely small. This makes it ok for them to have large codes to save bits for the letters that show up more often.