## PART I

1. **Write a web crawler/spider (click here for the Wikipedia definition) which crawls the cse.uta.edu (CSE) and crystal.uta.edu (CRYSTAL) domains. E.g.: While starting from cse.uta.edu, if cse.uta.edu/faculty is encountered, it is followed. If www.uta.edu is found on cse.uta.edu, it is ignored since it is outside the CSE or CRYSTAL domain. A crawler is a tool which starts at a page and follows some algorithm to traverse to pages that can be reached by following hyperlinks. In the real world, crawlers are used primarily for traversing the structure of the domain and to create indices on the content of the underlying pages. For the purpose of this project the focus should be to only find the underlying structure of the document graph where each document represents a single vertex. Directed edges exist between vertices whenever there are hyperlinks connecting one document to the other. Thus, after the crawling procedure is complete, a directed graph representing the underlying structure should be the output of the crawl. The first task consists of writing a Breadth First Search crawler to crawl cse.uta.edu and crystal.uta.edu as mentioned above. This task requires proper and adequate data structures to represent the underlying graph. Parsing the right information from every page needs to be done carefully (Look at free crawler code bases available online). After the crawl is complete, the next task is to find the following information:
    a. Diameter of the underlying graph
    b. Shortest path between http://www.cse.uta.edu and the class webpage on crystal.uta.edu

2. Consider a communication network that can be modeled as a weighted undirected connected graph G = (V, E). Each site in the network is represented as a vertex and each line of communication is bidirectional and has a cost associated with it. The cost may correspond to the expected delay on the line, or to the tariff for using each line. Each site has online *local* information; i.e., it knows online the edges (and vertices) adjacent to it. An MCST of the network can be used to broadcast messages to all sites. If we broadcast the messages by using only the edges of the MCST, then the total cost is minimized. Assume that such an MCST is computed by some method and that each site knows which of the edges adjacent to it belongs to the MCST. Assume now that the sites in a certain subset U subset of V share between them the information that is known to all of them. In other words, every site in U knows not only about the edges and vertices adjacent to itself, but it also knows all the edges and vertices adjacent to all vertices of U. Furthermore, assure that the partial MCST restricted to U is connected (i.e. it is a tree). Consider an edge e € U, which belongs to the MCST, whose cost has just changed.
    a. Find the conditions under the change in e's cost is guaranteed not to affect the MCST. Consider only conditions that can be checked with the information known to the sires in U. In other words, how can sites in U determine that no action needs to be taken top modify the MCST after the change?

b. Find the conditions under which the modified MCST is different from the original one only in edges that belong to U (hence, the change can be handled locally). Consider only conditions that can be checked with the information known to the sites in U.
c. Design and implement an algorithm to check for the conditions in parts a and b (again only within U), and to modify the MCST accordingly. The algorithm does not need to handle the case where the change to the MCST may be outside U. Generate a random graph with uniform distribution of edges to be the underlying structure.

3. Write a program that constructs Huffman code for a given English text and encode it. Write a program for decoding an English text that has been encoded with Huffman code. Experiment with your encoding and decoding programs for at least 5, 10, 25, 50 and 100 different texts, each text having a minimum of 50 words and at least one page in length. What range of compression ratios do you get? Can you improve the compression ratios by using some estimates of frequencies instead of actual frequencies? Feel free to look into published literature for ideas to improve the compression ratio.

** Literature on web crawling:

- Arvind Arasu, Junghoo Cho, Hector Garcia-Molina, Andreas Paepcke, Sriram Raghavan: Searching the Web. ACM Trans. Internet Techn. 1(1): 2-43 (2001)
- M. Najork and A. Heydon. High-performance web crawling. SRC Research Report 173, Compaq Systems Research Center, Palo Alto, CA, Sep. 2001.