This problem will explore graph traversal algorithms. As you know, this topic is central to many areas of VLSI CAD.

1 Minimum Spanning Trees

A minimum spanning tree is a spanning tree of a connected, undirected graph. It connects all the vertices together with the minimal total weighting for its edges.

A single graph can have many different spanning trees. We can also assign a weight to each edge, which is a number representing how unfavorable it is, and use this to assign a weight to a spanning tree by computing the sum of the weights of the edges in that spanning tree. A minimum spanning tree (MST) or minimum weight spanning tree is then a spanning tree with weight less than or equal to the weight of every other spanning tree. More generally, any undirected graph (not necessarily connected) has a minimum spanning forest, which is a union of minimum spanning trees for its connected components.

One example would be a telecommunications company laying cable to a new neighborhood. If the company is constrained to bury the cable only along certain paths (e.g. along roads), then there would be a graph representing which points are connected by those paths. Some of those paths might be more expensive, because they are longer, or require the cable to be buried deeper; these paths would be represented by edges with larger weights. Currency is an acceptable unit for edge weight – there is no requirement for edge lengths to obey normal rules of geometry such as the triangle inequality. A spanning tree for that graph would be a subset of those paths that has no cycles but still connects to every house; there might be several spanning trees possible. A minimum spanning tree would be one with the lowest total cost, thus would represent the least expensive path for laying the cable.

1. Describe, in words or pseudo-code, a simple algorithm to find a minimum spanning tree for a graph.

   Solution

   (a) Initialize a tree with a single vertex, chosen arbitrarily from the graph.

   (b) Grow the tree by one edge: of the edges that connect the tree to vertices not yet in the tree, find the minimum-weight edge, and transfer it to the tree.

   (c) Repeat step 2 (until all vertices are in the tree).

2. Find the minimum spanning tree for the following graph:

```
    B --1-- D ---2-- F ---3-- H ---4-- J ---5-- L ---6-- N ---7-- Q ---8-- S
1 /   \   /    \   /    \   /    \   /    \   /    \   /    \   /    \   /   \ 6
 /     \ / 2    \ / 11   \ / 8    \ / 7    \ / 4    \ / 1    \ / 1    \ / 8   \
A       /        /        /        /        /        /        /        /       U
 \     / \ 9    / \ 5    / \ 1    / \ 25   / \ 8    / \ 1    / \ 15   / \ 9   /
2 \   /   \    /   \    /   \    /   \    /   \    /   \    /   \    /   \   / 7
    C --1-- E ---2-- G ---3-- I ---4-- K ---5-- M ---6-- O ---7-- R ---8-- T
```

Solution

```
    B --1-- D ---2-- J ---3-- H ---4-- J        L -      N        Q ---8-- S
1 /                     \                     /    \   /        /        /   \ 6
 /                       \                   / 4    \ / 1      / 1      / 8   \
A                         \                 /        /        /        /       U
 \                         \ 1             /        / \ 1    /        /
2 \                         \             /        /   \    /        /
    C --1-- E ---2-- G        I ---4-- K ---5-- M -      O ---7-- R ---8-- T
```

2 Optimal Substructure

In computer science, a problem is said to have optimal substructure if an optimal solution can be constructed efficiently from optimal solutions of its subproblems. This property is used to determine the usefulness of dynamic programming and greedy algorithms for a problem.

Typically, a greedy algorithm is used to solve a problem with optimal substructure if it can be proved by induction that this is optimal at each step. Otherwise, provided the problem exhibits overlapping subproblems as well, dynamic programming is used.

Consider finding the least costly way of getting from point A to point B in a directed graph.

1. Describe, in words or pseudo-code, a simple algorithm to find the lowest cost path. Why might a greedy approach fail?

   Solution

   Starting from point A, explore paths to other nodes. For each node, explore all outgoing edges, but only keep the incoming edge provides the least costly way of reaching that node. Continue until all nodes have been explored, including point B.

   The invariant is that, for each node that has been fully explored, one knows the least costly way of getting to it. This can be used to find the least costly way to reach subsequent nodes.

   A greedy approach might fail because one might choose a route that is initially cheap, but then gets expensive. Choosing a route that was initially more expensive might have been better.

2. Find the lowest cost route through the following graph. Here there is a cost associated with both nodes and edges. (Edges with no listed value have cost zero.) Assume that all edges point left to right.

```
   1 ----- 25 ----- 3 ------ 6 ------ 12 ----- 12 ----- 5 ------ 25 ----- 1
  /   \   /    \   /    \   /    \   /    \   /    \   /    \   /    \   /   \
 /     \ / 2    \ / 11   \ / 8    \ / 7    \ / 4    \ / 1    \ / 1    \ / 8   \
A       /        /        /        /        /        /        /        /       B
 \     / \ 9    / \ 5    / \ 1    / \ 25   / \ 8    / \ 1    / \ 15   / \ 9   /
  \   /   \    /   \    /   \    /   \    /   \    /   \    /   \    /   \   /
   7 ----- 10 ----- 17 ----- 9 ------ 1 ------ 11 ----- 7 ------ 1 ------ 3
```

   Solution

```
First find:

   1 ----- 25 ----- 3 ------ 6 ------ 12       12       5 ------ 25       1
  /                  \                        /        /                   \
 /                    \                      / 4      / 1                   \
A                      \                    /        /                       B
 \                      \ 1                /        /                       /
  \                      \                /        /                       /
   7 ----- 10 ----- 17       9 ------ 1 ------ 11 ----- 7 ------ 1 ------ 3
```

```
Conclude that best path is:

    1 ----- 25 ----- 3 -
  /                     \
 /                       \
A                          \                                                        B
                             \ 1                                                   /
                              \                                                   /
                                9 ------ 1 ------ 11 ----- 7 ------ 1 ------ 3
```