Section 5.6

The Traveling Salesman Problem

THE TRAVELING SALESMAN PROBLEM

Consider the dilemma of a traveling salesman. He must visit each city in his region and return to his home office on a regular basis. He seeks a route that allows him to visit each city at least once (exactly once would be even better) and return home, with the added property that this route covers the least distance. Clearly, a weighted graph can be used to model the possible routes. Since we can insert edges with infinite distances, we can also restrict our attention to complete graphs.

NOTES ON TSP

- The traveling salesman problem is NP-complete.
- Approximate algorithms are used with heuristics to try to make some gains.

THE NEAREST NEIGHBOR APPROACH

- The **nearest neighbor** approach begins with a single vertex, adds the edge of minimum distance, and continues to build from either end of the path by repeatedly taking the nearest neighbor.
- Unfortunately, in the nearest neighbor approach, to close the path to a cycle is often expensive. Many short edges can be ignored this way as well.

SHORTEST INSERTION HEURISTIC

- The **shortest insertion heuristic** begins with some short cycle and expands this cycle by inserting the vertex that causes the length of the cycle to increase the least.
- Difficulties can arise.
- Arbitrarily weighted graphs need not satisfy the "reasonable rules" of distance. That is, the triangle inequality need not apply.
- If the triangle inequality does hold some progress can be made.
- In the following algorithm, we assume a single vertex and a K_2 are (degenerate) cycles.

SHORTEST INSERTION ALGORITHM

Algorithm 5.6.1 Shortest Insertion Algorithm.

Input: A weighted graph $G = (V, E)$ satisfying the triangle inequality.

Output: A hamiltonian cycle C that approximated the salesman cycle.

SHORTEST INSERTION ALGORITHM (CONCLUDED)

1. Select any vertex and consider it a 1-cycle C_1 of G. Set $i \leftarrow 1$.
2. If $i = p$, then halt since $C = C_p$ is the desired cycle;
   else if C_i has been selected $(1 \leq i \leq p)$, then find a vertex v_i not on C_i that is closest to a consecutive pair of vertices w_i and w_{i+1} of C_i.
3. Let C_{i+1} be formed by inserting v_i between w_i and w_{i+1} on C_i and go to step 2.