1 Janaury 9 Lecture

We discussed solving the TSP via complete enumeration and using dynamic programming to cut down on the search space.

We noted that there were $\frac{(n-1)!}{2}$ tours through n cities since we can assume the tour starts at city 1, and traversing a tour in the opposite direction yields a tour with the same cost.

We discussed generating these tours systematically via recursion and by generating the tours in lexicographical order. This is one of the problems on Assignment 1 and code for both procedures will be available after the assignments are handed in and marked.

We also discussed generating for each subset S of cities $\{C_2, C_3, ..., C_n\}$ and each city C_j in S, only the shortest possible tour through S from C_1 to C_j. To do so, we needed to build a table with $O(n2^n)$ entries containing the cost of each such tour and a predecessor pointer which allows us to reconstruct the tour by traversing the table. We then saw that we could compute the table entry for S, C_j in $O(n)$ time by looking at the table entry for $S - C_j, C_i$ for each C_i in $S - C_j$. This yielded an $o(n^2 2^n)$ algorithm for solving the problem. Again code will be made avialable after Assignment 1 is marked.

We spent some time discussing other ways of generating permutations. One was to extend the notion of Josephus Permutation on page 318 of Cormen to include the case when m was bigger than n. The following optional exercises show that the set of Josephus permutations is not the set of all permutations:

1) show that 4,1,2,3 is not an extended Josephus permutation.

2) show there are only 12 extended Josephus permutations of length 4.

3) show that the number of extended Josephus permutations is at most the Lowest common multiple of 1,2,...,n and hence o(n!).

We also discussed generating permutations in some order which required only one transposition to move between permutations. Details can be found in Chapter 7 of Combinatorial Algorithms by Nijenhuis and Wilf (Second Edition, 1978, Academic Press)