CSC 6220: Parallel Computing I: Programming
ECE 5610: Introduction to Parallel and Distributed Systems
Homework 1
Fall 2016

Assigned on: Wednesday September 7, 2016
Due on: Monday September 19, 2016, 5:30pm

Implement a C/C++ program that solves the Traveling Salesman Problem (TSP) by exhaustive search over all possible cyclic permutations of cities. TSP is defined as follows. Given a set of cities $\{1, 2, \cdots, n\}$ and a matrix of non-negative integers $D = d_{ij}$, where d_{ij} is the distance between cities i and j, find a tour of cities (a permutation π) such that the length of the tour

$$d_{\pi(n)\pi(1)} + \sum_{i=1}^{n-1} d_{\pi(i)\pi(i+1)}$$

is minimized.

The following is a pseudo-code description of the exhaustive search algorithm that solves the TSP.

Algorithm 1 TSP-ES: Exhaustive Search Algorithm

```
1: Input: Distance matrix, $D$.
2: Output: Minimum cost tour.
3: $mincost \leftarrow \infty$
4: for all cyclic permutations $\pi$ of $\{1, 2, ..., n\}$ do
5:     $cost = d_{\pi(n)\pi(1)} + \sum_{i=1}^{n-1} d_{\pi(i)\pi(i+1)}$;
6:     if $cost < mincost$ then
7:         $mincost \leftarrow cost$;
8:         $tour \leftarrow \pi$;
9: output $tour$
```

To generate all cyclic permutations you must implement the following algorithm (called "Plain Changes") [the description of this algorithm is from D. Knuth, TAOCP, vol. 4, page 322]. The input is a sequence $a_1, a_2, \cdots, a_n$ of distinct elements. The algorithm uses two auxiliary arrays c and o of size n.

Algorithm 2 Generating All Permutations

```
[P1:] Initialize: $c_j \leftarrow 0$ and $o_j \leftarrow 1$, for $1 \leq j \leq n$.
[P2:] Visit permutation $(a_1, a_2, \cdots, a_n)$.
[P3:] $j \leftarrow n$ and $s \leftarrow 0$
[P4:] $q \leftarrow c_j + o_j$. If $q < 0$ go to P7; if $q = j$ go to P6.
[P5:] Swap $a_{j-c_j+s}$ with $a_{j-q+s}$. Then set $c_j \leftarrow q$ and return to P2.
[P6:] Terminate if $j = 1$; otherwise set $s \leftarrow s + 1$.
[P7:] Set $o_j \leftarrow -o_j$, $j \leftarrow j - 1$ and go back to P4.
```

You should not use the C/C++ goto statement when implementing the above algorithm.

The starting node of the tours should always be city 1, thus, you will need to use the above algorithm to generate only the permutations for the sequence of $n-1$ cities, that is, $\{2, 3, ..., n\}$. As an example, for $n = 4$, the algorithm should generate the following six cyclic permutations: (1, 2, 3, 4); (1, 2, 4, 3); (1, 4, 2, 3); (1, 3, 2, 4); (1, 3, 4, 2); and (1, 4, 3, 2).

All the code should be placed in a single .c or .cpp source file.

The input to your program should be read from a file in which you specify the number of cities and the distance matrix. You should test your implementation on the following input file:

```
4
0  3   4   7
3  0   5   1
4  5   0  10
7  1  10   0
```

You should submit the source file of your implementation via Blackboard dropbox.