CS 4000

Homework # 2

due Friday Feb. 15th, 2019

(30 pts.)

1 Introduction

The Hamiltonian Path Problem is a classic computer science problem: Given a graph and two vertices i and j, determine whether there is a path from i to j in the graph that visits each vertex in the graph exactly once. Now, this problem is well-known to be NP-complete.

I have written a solution to this problem (`Hamiltonian_Path.cc`) that uses the `next_permutation` function in C++ to generate all of the permutations (tours) of the vertices that start at vertex i and end in vertex j. This program will find a Hamiltonian Path, if it exists. Otheriwse, it will say that no such path exists. However, this program is painfully slow. On a small graph (`small_graph.dat`) with 5 vertices, it finds the the tour `2 0 1 3 4` from 2 to 4 in much less than a second. But, on a bigger graph (`big.dat`) with 13 vertices, it takes over one minute to find a solution. On bigger graphs (`bigger.dat` and `biggest.dat`), it takes much longer to solve.

For example, on input

```
5 2 4
0 : 1 2 4
1 : 0 2 3 4
2 : 0 1 3
3 : 1 2 4
4 : 0 1 3
```

The output of the program should be "Tour = 2 0 1 3 4".

Your task is to make this code faster using parallel computing.

Modify `Hamiltonian_Path.cc` using OpenMP so that

1. Your modified program still produces the correct results, and

2. It is at least 75% efficient on `bigger.dat` on a machine with 4 cores/processors.