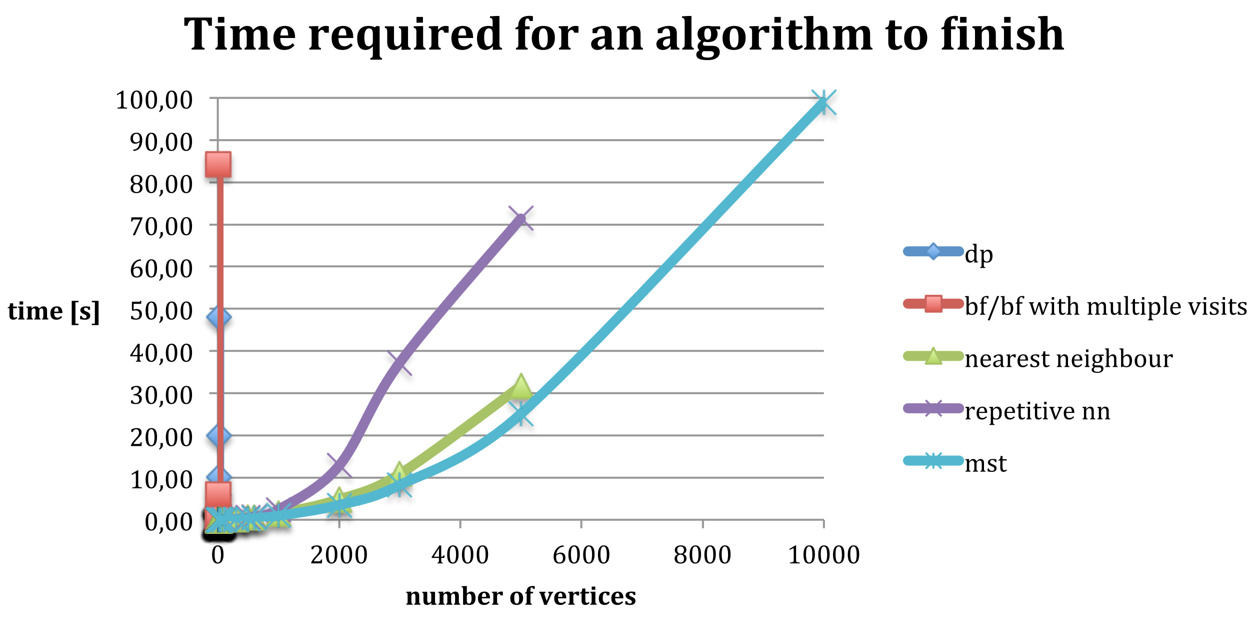

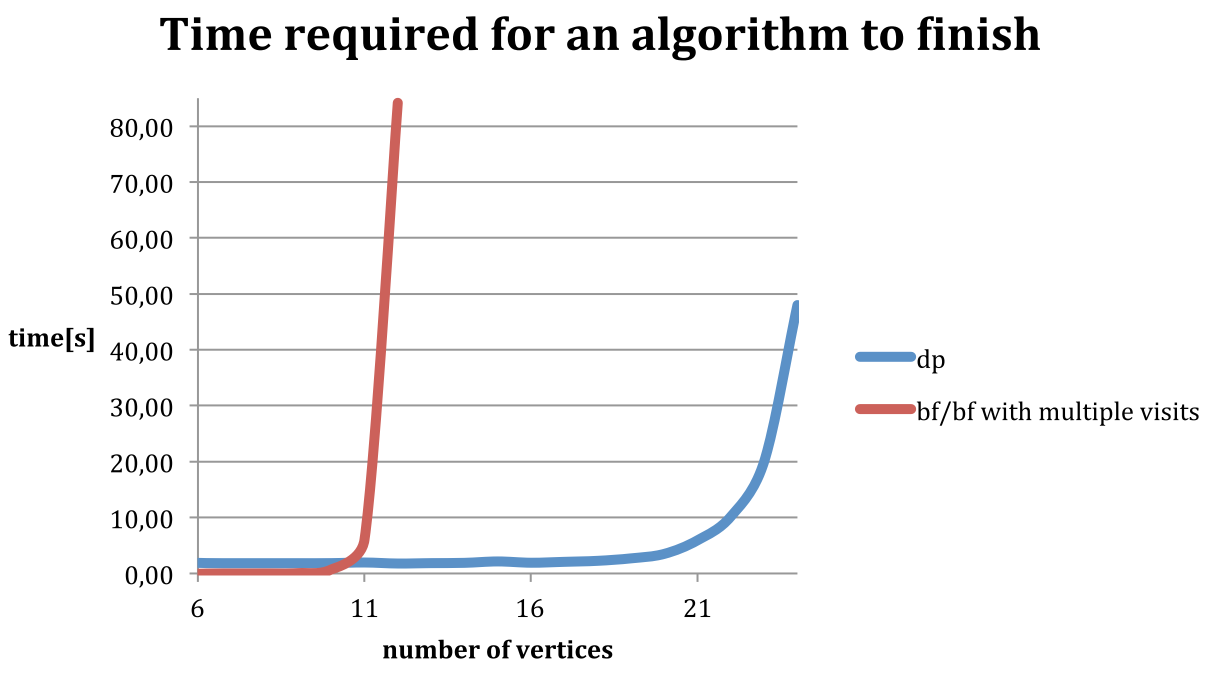

n or V or |V|- num of vertices

E - num of edges, E=n^2 for this problem

Brute force (brute_force_all_permutations) - checks all permutations - complexity O(n! * n). This solution is very easy to modify to use as many threads as you wish.

DP (dynamic_programming_tsp_solution) - O(N^2 * 2^N) time and O(N * 2^N) space. In this case memory turns out to be the bottleneck. For 24 vertices, peak memory used is 3GB and 25 vertices would require around 7GB.

Both work for symmetric and asymmetric (directed and undirected) graphs. Both brute force and DP have been checked (up to 10 vertices instances) by submitting the solutions to the uva online judge, problem:
http://uva.onlinejudge.org/index.php?option=onlinejudge&page=show_problem&problem=1437
Source code is available in 10496 – Collecting Beepers and 10496 – Collecting Beepers DP.

Given condition that you can visit each city multiple times, you can simply compute distances between all vertices using Floyd-Warshall algorithm in O(n^3). This has been done in brute_force_multiple_visits. Overall complexity doesn't change. Same could be easily done for dynamic programming solution. We didn’t bother to print the in-between vertices on the path.

Nearest neighbour – we wrote an O(n^2logn) version, although you could make an O(n^2) one easily. From our experience logn factor hardly ever matters (log(40k)=10.6), so we didn’t bother.

Repetitive nearest neighbour is O(Nearest neigbour) * O(n) = O(n^3logn) in this case.

Both work for directed and undirected graphs. For n==5000 peak memory used both in nn and rnn is 1.2GB.

Double MST - creates minimum spanning tree (using Prim’s algorithm), then does n (each vertex as the starting point) times dfs (preorder traversal of the tree). Algorithm constructs for every dfs a tour, and finally it chooses the best tour. The output tour is guaranteed to be max not more than twice as long as the optimal solution (only when triangle-inequality holds). MST complexity is O(V^2), MST could be done in O(E * logV) but there is no point in doing that here, since we are considering only complete graphs. Algorithm works of course only for undirected graphs. DFS is O(V) and we use dfs |V| number of times, to find the best tour constructed by MST. Overall complexity is O(V^2). Algorithm should work easily for instances up to 10^4 number of vertices.

In our implementation for n==5000 peak memory used is 300MB and for n==10k it is 1.14GB.

Important fact: "Without the triangle inequality, a polynomial time approximate algorithm with constant approximation ratio not exist unless P=NP." We leave the proof as an exercise for the reader.

Most of our implementations work only for integers provided the cost of the Hamilton cycle is less than 10^9. You can easily modify them to work with doubles.

Example results:

12cities_symmetric.txt

Optimal tour found by both dp and all permutations: 1733

Brute force multiple visits 1675

MST 2848

Nearest neighbour 2110

Repetitive nearest neighbour 1733 (optimal) but other vertices order than dp or all permutations algorithm

11cities_symmetric.txt (no triangle-inequality proved by MST and optimal results)

brute force 1675

brute force multiple visits 1366

dp 1675

MST 3431

Nearest neighbour 2285

repetitive nearest neighbour 2139