CS6100 Course Project
Report

Chriostofides' Short-cutting heuristics for Euclidean metric Travelling salesman problem

Submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology
in
Computer Science and Engineering

Submitted by

Roll No	Names of Students
CS17B005	A. Subhash
CS17B011	D. Varun Teja
CS17B012	D.M.S Krishna
CS17B021	P. Jaitesh

Under the guidance of
Prof. B. V. Raghavendra Rao

Department of Computer Science and Engineering
IIT Madras

Abstract

Several $O(n), O(n^2)$ short-cutting heuristics are described which are used in Christofides' algorithm for solving n-city travelling salesman problems whose cost matrix satisfies the triangularity condition. The Christofides' algorithm invovles computation of a shortest spanning tree of the graph G defining the TSP, and finding the minimum cost perfect matching of a certaing induced subgraph of G.

Contents

List of Figures

Chapter 1

Goal

The goal is to find more efficient(in terms of solution cost) short-cutting heuristics which are used in the last step of Christofides' algorithm for finding the TSP tour.

Chapter 2

Introduction

2.1 Metric space

A metric d on a set X, also called a distance function, is a function that defines a distance between each pair of elements of the set. A set with a metric is called a **metric space**.

Formally, $d : X \times X \to R$ is a metric if it is a function satisfying the following properties $\forall x, y, z \epsilon X$:

1. Non-negativity : $d(x, y) \geq 0$
2. Indiscernability : $d(x, y) = 0$ iff $x = y$
3. Symmetry : $d(x, y) = d(y, x)$
4. Subadditivity : $d(x, y) + d(y, z) \geq d(x, z)$

2.2 The Christofides' algorithm

The Christofides' algorithm is an approximation algorithm for Metric TSP with an approximation ratio of 1.5. Let G be a graph with n points in the euclidean metric space. Below is a description of the steps involved in the Christofides' algorithm.

2.2.1 Pseudo code

Algorithm

1. Find an MST of G, say T.
2. Compute a minimum cost perfect matching, M, on the set of odd-degree vertices of T.
3. Add M to T and obtain an Eulerian multi-graph H.
4. Find an Euler tour, E of this graph.
5. Output the tour that visits vertices of G in order of their first appearance in E.

Figure 2.1: MST T of G

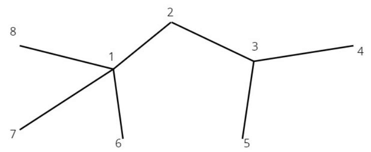


Figure 2.2: Multi-graph H of G

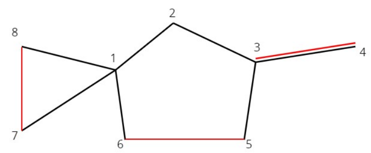
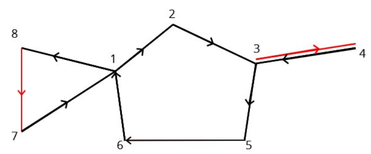


Figure 2.3: Euler tour E of H



2.2.2 Time complexity

1. Creating MST T of G : $O(nlogn)$
2. Finding the minimum cost perfect matching M : $O(n^3)$
3. Creating multi-graph H : $O(n)$
4. Finding Euler tour E in H : $O(n)$

Chapter 3

Short-cutting heuristics

3.1 Simple Heuristic

1. The order of appearance of points in the hamiltonian cycle is the order of their appearance in the Euler tour. (For each of the points, we retain the first occurence of the point and discard the remaining).

2. Time complexity is $O(n)$.

3.2 Tri-Opt Heuristic

1. In this heuristic we take a Euler tour and greedily remove the repeated vertices which have higher heuristic value.

2. The heuristic value used is sum of distances from the vertex to adjacent vertices minus distance between the adjacent vertices.

3. We can see that this is better than previous one but we are performing it on a Euler tour, hence there is still room for imprevement.

4. Time complexity is $O(n)$.

3.3 Tri-Comp Heuristic

1. This heuristic is applied on Multi graph (H) instead of one Euler tour.

2. Here we start with vertices of order greater than two and greedily remove its edges until its order is two.

3. The idea is that in the final hamiltonian cycle which we need to arrive by short cutting has degree two for all the vertices.

4. Two things we need to do are - pair up the free vertices formed greedily and make sure the process does not result in two disjoint components

5. The heuristic value here is sum of distances between paired up vertices and distace of two edges that remained with our vertex.

6. Since our problem is in 2d space, each vertex in MST will have a degree of maximum 5, so our multi graph will have a maximum degree of 6. This property highly affect the theoritical complexity of our heuristic.

7. Time complexity for checking the graph connectivity is $O(n)$ and therefore, time complexity for the tri-compt heuristic is $O(n^2)$.

3.4 DIH-Tri-Comp Heuristic

```
Root the MST at a node of degree 1;

Let r' denote unique child of the root;

Insert all children of r' into queue Q;

While(Q is not empty) {
        extract node v from Q;
    insert all children of v into Q;
    if(deg(parent(v))+deg(v) <= D)
        redefine all children of v
          to be children of parent(v);
}
```

Figure 3.1: Pseudo code for DIH heuristic

1. This heuristic is DIH(Degree Increasing Heuristic) optimization applied on the MST T to get a new tree T', then creating a multigraph H' from T', followed by applying Tri-compt heuristic on the multigraph H'.

2. We want to increase the order of a vertex in MST by adding the vertices of its children to itself

3. We can see that the tree T' is no longer MST but this process preserves the Euler tours i.e. the set of Euler tours of this tree T' is super set of that of the MST T.

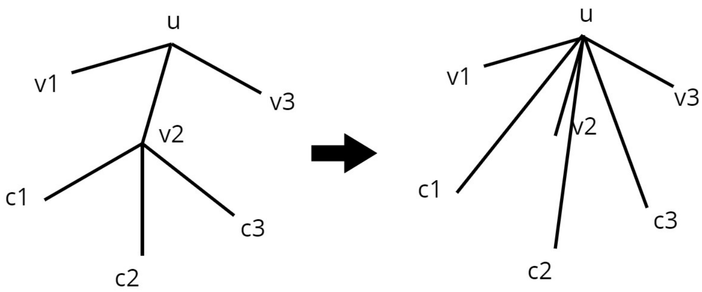


Figure 3.2: Applying DIH on T(left) to get T'(right)

4. This way we are applying heuristic on a bigger space than that of the former one and might have a chance of getting better results.

5. DIH can be implemented with $O(n)$, so comp heuristic is the bottleneck here. Overall time complexity is $O(n^2)$

3.5 Performance analysis of heuristics

We measured the performance of the four heuristics over 100 TSP problems from TSPLIB which is a library of sample instances for the TSP (and related problems) from various sources and of various types.

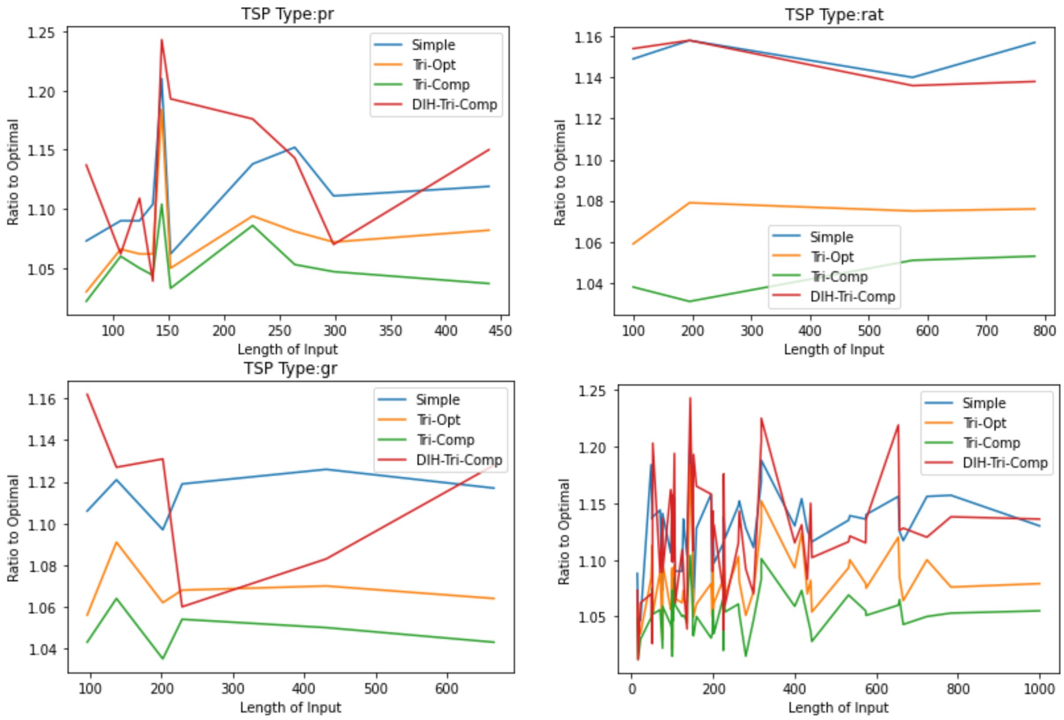


Figure 3.3: Performance of heuristics on different families of datasets

Chapter 4

Perturbation of inputs

4.1 Methodology for perturbation

1. Zero mean IID (Independent identical distribution) noise of certain standard deviation is added to each of the n points of the graph.

2. The span of points is measured for each graph and is used for scaling the Standard deviation accordingly.

3. Standard deviation is scaled for each graph instead of scaling the given points in view of getting better precision in calculations.

4. The algorithm is run 20 times for each experiment and the average of tour lenght was recorded.

4.2 Results

4.2.1 Perturbation with different standard deviations

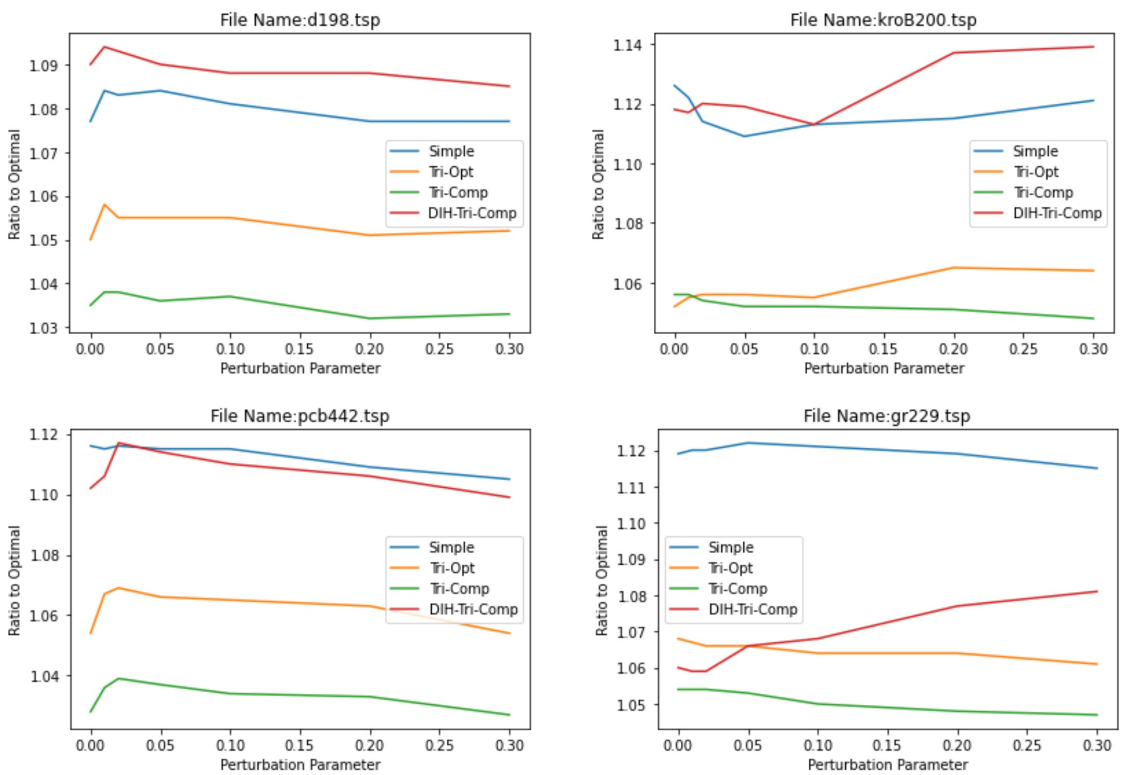


Figure 4.1: Performance of heuristics on few selected datasets

4.2.2 Worst case performance of heuristics on perturbation

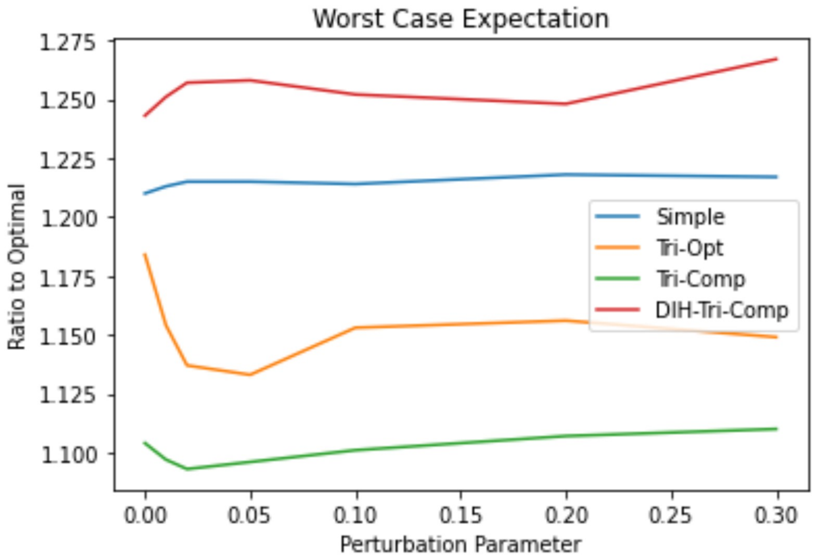


Figure 4.2: Worst case performance among all files for different heuristics

4.2.3 Performance of heuristics on different size graphs for each standard deviation

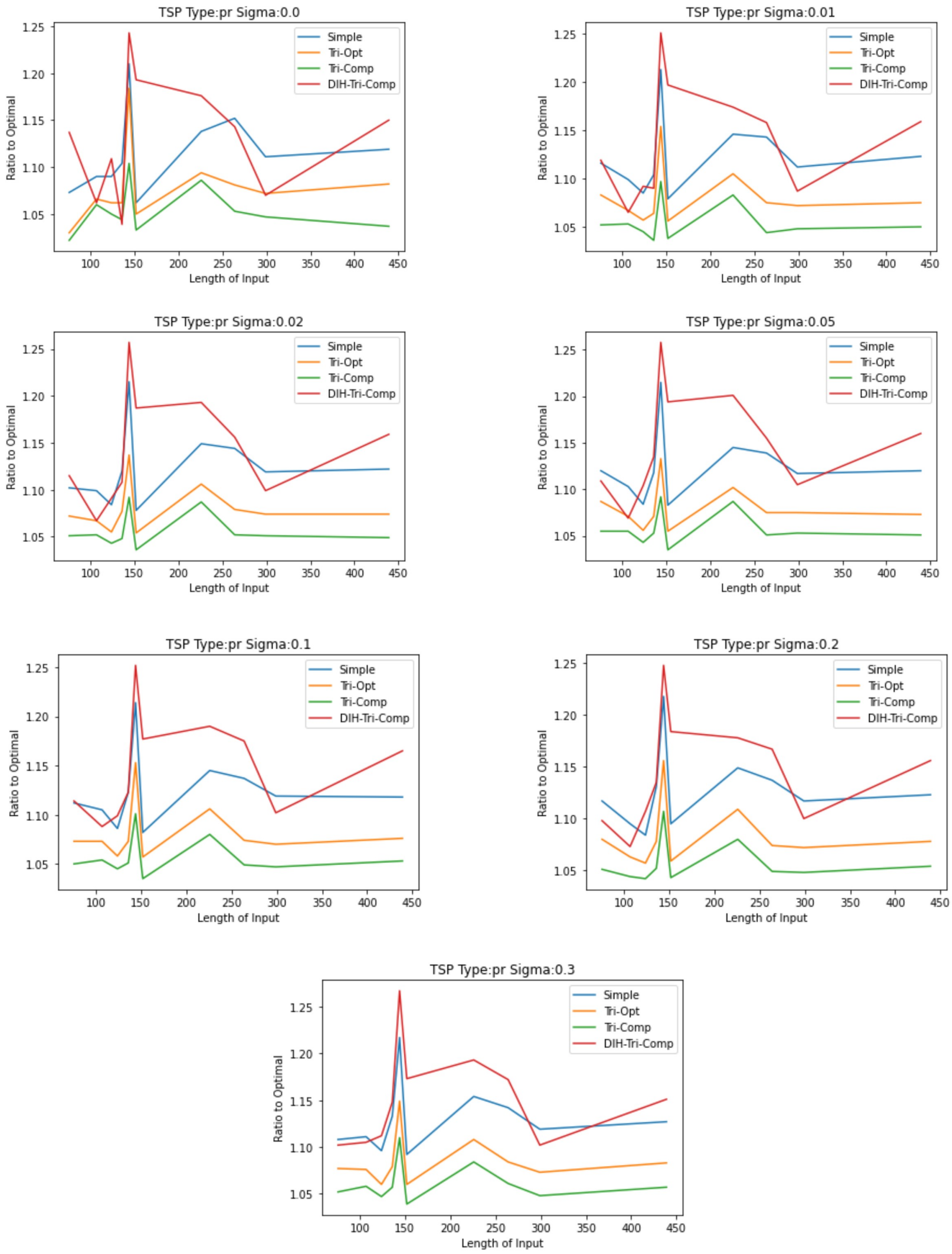


Figure 4.3: Performance of heuristics on few selected datasets

References

[1] Minimum-weight double-tree shortcutting for Metric TSP: Bounding the approximation ratio
https://arxiv.org/abs/0711.2399

[2] Fast minimum-weight double-tree shortcutting for metric TSP: Is the best one good enough?
https://dl.acm.org/doi/10.1145/1498698.1594232