A Heuristic Solution to the Metric Travelling Salesperson Problem

Rosie Bartlett
lvff38@durham.ac.uk

October 30, 2021

Abstract

In this paper I will present a heuristic solution to a variant of the TSP(travelling salesperson problem) knows as the metric TSP or ΔTSP , where all points can be represented on a two dimensional Euclidian plane. This new approach uses an approach inspired by physics simulations, where we stretch a path around the set of points s.t. all points are either on this path or bounded inside the polygon of this path. Through bisectors of edges, margins, and normals with edges through points, the remaining points are placed along edges, and thus finding a heuristic solution to the given problem

1 A description of the ΔTSP

This section pertains to a formal definition of the ΔTSP and the format of the input to the function that solves the problem.

The ΔTSP problem can be formally defined as the matrix P of shape $2 \times n$ where n is the number of points. We use a matrix here to allow us to perform matrix rotation which is an essential part of the solution. For each element in P $(p_{i,j})$, $p_{i,j} \in [-1, 1]$.The points represented by this matrix must conform to the triangle rule s.t. for the points p_a, p_b, and p_c for unique values of a, b, and c where $a, b, c \in [1, n]$:

$$|p_b - p_a| + |p_c - p_b| \geq |p_c - p_a| \tag{1}$$

2 The proposed solution

I will now present my proposed heuristic solution to a general ΔTSP problem. The solution is in two sections, the bounding section and compression section, and will be detailed as such. This section will also use an example matrix to demonstrate the two sections of this solution. This is shown in fig. 1

2.1 Bounding process

The bounding process is an iterative process that creates a bounding polygon around all of the points. This begins as follows once the lowest point has been found, which we will call p_l:

1. For each point with an x co-ordinate greater than the x co-ordinate of p_l, find the gradient of the line through each point and p_l

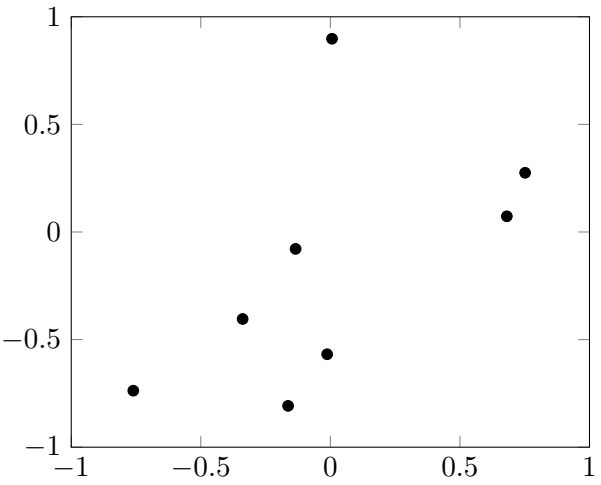

Figure 1: Example matrix P with 8 points

Notation	Description
P	Matrix of points
p_i	The point $(p_{i,1}, p_{i,2})$
$p_{i,j}$	The value in the matrix P with index i, j
$m(p_i, p_j)$	The gradient of the line from p_i to p_j

Figure 2: Notation used throughout this paper

2. The new lowest point p_{ln} is the point that gives the lowest gradient from p_l
3. Rotate P by θ radians where $\theta = \arctan m(p_l, p_{ln})$
4. Repeat the process with the new lowest point, assuming that the ew lowest point is not the same as the original lowest point

The result of this can be seen in fig. 3

2.2 Compression process

This is so-named after the inspiration for this method. This method was inspired by considering the ΔTSP as a set of pegs in between two panes of glass, with a ring around the outside. The centre of the ring is then depressurised and the resulting path through all the points would be a heuristic solution to the solution. In reality, this would be very difficult to code so alterations and improvements were made to this method to achieve a more accurate heuristic solution. This inspired the name for this section, since in practice, it moves some of the edges inwards to contain more points, and thus the edge is compressed towards the centre.

This section can be split into multiple sub-sections, the first of which is the triangle bisector bounding section.

2.2.1 Bisector Triangle Bounding

This section allocates each edge e_n a set of points s_n where each element of s_n is a point that does not lie in the bounding polygon. To achieve this, we rotate P to align e_n with the

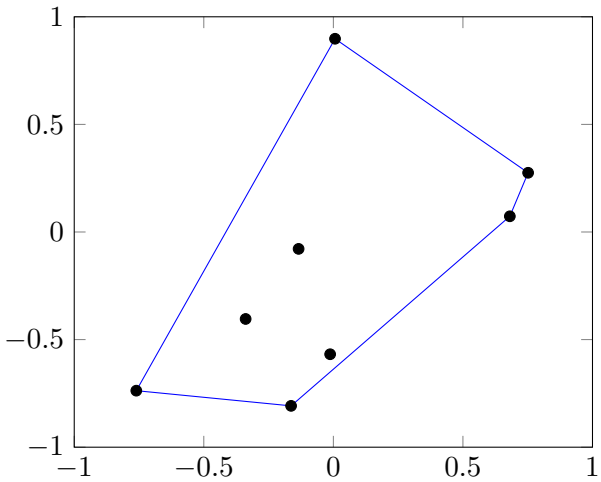

Figure 3: Result of the bounding process on an example matrix P with 8 points

horizontal. We will now consider everything in polar coordinates with the left point of e_n being the pole. We assign θ to be the angle between the horizontal and angle bisector of e_n and the leading edge e_{n-1}. From this we can see that $\theta \in (0, \frac{\pi}{2})$ since the angle along which $en - 1$ lies is in the range $(0, \pi)$, since no points lie below e_n. We can then define the angle α as the angle between the angle bisector of e_n and the following edge e_{n+1}, and the horizontal i.e. $\theta = 0$. We can then derive an equation for the angle bisector of e_n and e_{n+1} in polar form as the following:

$$r = |e_d| \frac{cos(\alpha)}{cos(\theta - \alpha)} \tag{2}$$

We can then calculate the angle and distance from the pole to each point not in the bounding polygon, and then compare each points distance from the pole to the value of r from eq. (2) using θ as the angle from the pole for the current point. If this value greater than or equal to the distance from the pole and the angle is less than or equal to θ, the point is considered to be inside the bounding area of the triangle.