Quotient algorithm: a simple heuristic for the travelling salesperson problem inspired by human solutions

Conference or Workshop Item

Accepted Version

Kyritsis, M., Ud Din, S. and Gulliver, S. (2019) Quotient algorithm: a simple heuristic for the travelling salesperson problem inspired by human solutions. In: 2018 Fifth HCT Information Technology Trends (ITT), 28-29 Nov 2018, Dubai, United Arab Emirates, pp. 47-49. Available at http://centaur.reading.ac.uk/82530/

Quotient Algorithm: A Simple Heuristic for the Travelling Salesperson Problem Inspired by Human Solutions

Markos Kyritsis, Shahab Ud Din
Computer Information Science Department
Higher Colleges of Technology
Sharjah, UAE
mkyritsis@hct.ac.ae, suddin@hct.ac.ae

Stephen Gulliver
Business Informatics Department
University of Reading, Henley Business School
Reading, UK
s.r.gulliver@henley.ac.uk

***Abstract*—** The Travelling Salesperson Problem (TSP) is a classic example of a non-polynomial (NP) hard problem, which cannot be practically solved using exhaustive algorithmic approaches. This study explores the human approach, and presents a Quotient Algorithm (**Quot**) - a modification to the nearest neighbor algorithm - inspired by human path crossing avoidance behavior when solving ETSP graphs. We compared the developed **Quot** results against standard heuristic algorithms and found that this simple modification outperforms the NN, as well as other existing heuristic approaches.

Keywords*— *Travelling Salesman; Optimisation Problems; Heuristic Algorithms

I. Introduction

A travelling salesperson starts in their home city and attempts to find the shortest tour that allows them to visit every city once before returning back home. In computational terms, the travelling salesperson problem (TSP) is considered to be a non-polynomial (NP) hard problem, since exhaustive testing would require us to evaluate the cost of every single possible tour [2]. Assuming that costs don't change with direction (i.e., the graph is symmetrical) and we can move in any direction, there are (n-1)!/2 possible solutions where n is equal to the number of nodes. The exponentially large amount of possible solutions means that it would soon become impractical to solve such problems using an exhaustive-search algorithm [17].

As an alternative to exhaustive testing, near optimum "good enough" TSP solutions can be derived using either stochastic or non-stochastic approaches. Stochastic approaches - e.g. Hopfield (or derivatives of Hopfield), artificial neural networks [7], evolutionary algorithms [10], simulated annealing [8] - are random, and may not be precisely predicted. Non-stochastic approaches, often referred to as the heuristic algorithms - e.g. the nearest neighbor (NN), nearest insertion, farthest insertion, and cheapest insertion [15] - model a predefined relationship between factors. The aim of TSP heuristic approaches is to provide simple, flexible, and easy to implement near optimal solutions. A trade-off therefore occurs between stochastic and non-stochastic approaches, i.e. tour cost vs processing speed, with stochastic approaches obtaining more near optimal solutions, yet non-stochastic approaches requiring less computational expense.

II. Human Performance in the TSP

Humans are able to provide near optimal TSP solutions in an efficient near linear time period, given that the problem is presented in 2D Euclidean space (i.e., using geometric distances as a measure of the costs of routes between cities) [2,3]. Average human solutions to the TSP, are approximately 1% more costly than optimal solutions for problems where the number of nodes is between 10 and 20 [11], and approximately 11% more costly for problems where the number of nodes is 120 [2]; although the results can vary greatly amongst participants. For example, Kyritsis et al. [9], found that the number of trials containing crossings, and hence sub-optimal solutions, ranged from 0% to 93% for specific participants.

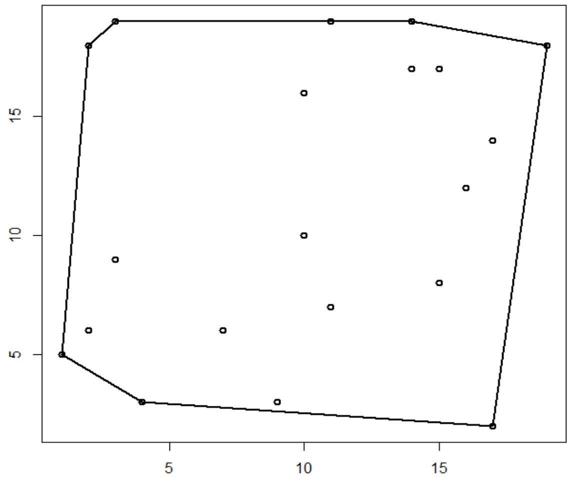

Figure 1 - Convex hull of a Euclidean TSP problem graph

MacGregor et al. [12] suggest that when solving the ETSP, people apply a global-to-local heuristic. That is, they initially form an imaginary perimeter around the boundary nodes of the graph, termed the convex hull (see Fig. 1). Empirical TSP research has shown that the properties of the convex hull, such as the number of boundary nodes, correlate with individual performance on the TSP [19]. Accordingly, MacGregor & Ormerod [11] theorised that subsequent to forming the global convex hull, individuals solve the TSP by sequentially inserting local internal nodes in either a clockwise or a counterclockwise direction; a.k.a. a global-to-local perceptual organizing heuristic. Whilst this heuristic can provide an efficient means of solving the problem through perceptual processing of geometric properties of the graph, van Rooji, et al. [18] argue that there is insufficient evidence in literature to support the convex hull hypothesis, and instead propose an alternative heuristic that may also be implemented by humans, named the 'crossing-avoidance hypothesis'. Theorists suggest such crossing-avoidance occurs because humans are 'trained' to regard routes with crossed lines as non-optimal.

Ultimately, in both global-to-local and local-to-global models, people intentionally keep the number of crossings to a minimum. Because of the crossing-avoidance behaviour, we can deduce that a good node candidate (i.e. the best next possible node) will:

1. Most likely be positioned away from the centre of the current problem, in order to increase the chance that it belongs to the convex hull. For humans, judgment of object distance, and identification of the centre of mass is an automatic process of the visual system for computing positions within, and between, groups through object centroids [1, 13, 16]. In light of this, we hypothesise that identification of the centroid position of the shape (i.e., taking the mean distance for x and y) is used by humans to judge relative distance between the remaining nodes

2. Be close enough to the current node to minimise the number of possible intersections, a geometric property that has been shown to inversely impact human performance [3, 19]

III. THE QUOTIENT ALGORITHM

To incorporate the 'human' approach, we developed a naive (i.e. simple and non-optimised) quotient algorithm (**Quot**), using R code [14], which emulates the aforementioned human problem-solving heuristic. This algorithm was modified from a simple nearest-neighbour algorithm, with a complexity of $O(n^2)$, where cost-to-current/cost-to-centroid defines the 'distance' from the current node.

```
Quotient Algorithm: ListVisted: Compute List of nodes visited in order
Input: ListNodes: List of nodes, initial node
Output: ListVisted: List of nodes visited in order
ListAvailable ← ListNodes – initialNode
currentNode ← initialNode
ListVisted ← initialNode
while ListAvailable ≠ Φ do
        centroid ← centroid (ListAvailable + initialNode)
        for each node i ε ListAvailable do
                Cij ← Cost (i, currentNode)
                Cik ← Cost (i, centroid)
                ListQuotient ← ListQuotient + Cij / Cik
        end for
        nextNode ← minmum(ListQuotient)
        ListAvailable ← ListAvailable – nextNode
        ListVisted ← ListVisted + nextNode
        currentNode ← nextNode
        ListQuotient ← Φ
end do
```

IV. METHOD

To benchmark the quotient algorithm we chose to compare **Quot** performance results against four existing heuristic algorithms, i.e. the nearest neighbor (NN) algorithm and three common variations of the insertion algorithms (Nearest, Farthest, Cheapest). In order to test and compare heuristic algorithms we generated thirty sets of TSP graphs. Each set contained randomly generated graphs, with the number of nodes ranging incrementally from 50 nodes to 399 nodes, for a total of 8750 graphs. The starting node for each graph was fixed to keep comparison of results consistent. NN, farthest insertion, and cheapest insertion were implemented with the help of the TSP library [6, 7]. The costs for all algorithms (including the quotient algorithm) were stored in a .csv file for further analysis and uploaded to https://github.com/markoskyritsis/ETSPPartData, for the sake of replication and transparency.

V. RESULTS

The distribution of our data was fairly normal, however, results failed to satisfy the homogeneity of error variances required for use of ANOVA (Levene's test was < 0.01). As such, we used the Kruskal Wallis test as a nonparametric alternative, with tour costs as the dependent variable, and the heuristic as the independent categorical variable. Results from the Kruskal Wallis test showed that there were significant differences in average tour costs between algorithms [$H(5) = 450.29$; $p < 0.001$], a finding illustrated in Fig. 2.

The Dunn test was used to perform pairwise comparisons, which revealed that the **Quot** performed significantly better than the NN [$z = 6$, $p < 0.001$], the nearest insertion [$z = 4.26$, $p < 0.001$], and the cheapest insertion [$z = 0.14$, $p < 0.001$], although the difference between cheapest insertion and **Quot** scores were not very meaningful (Cohen's $d < 0.01$). Finally, the **Quot** underperformed compared to farthest insertion [$z = -10.53$, $p < 0.001$].

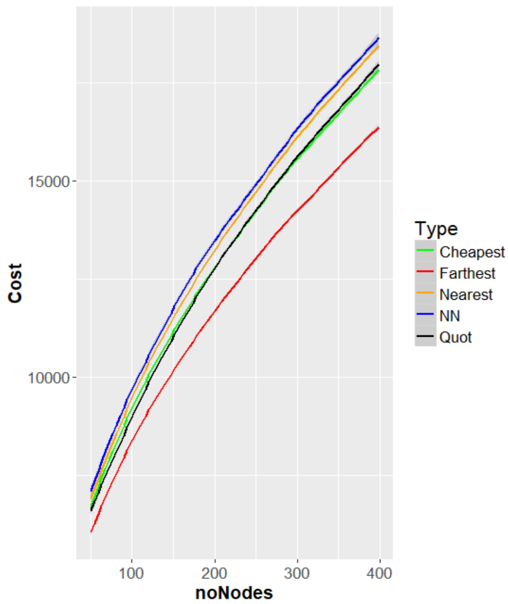

Figure 2 – Comparison of tour costs in an incremental volume of graphs. Cost measurements are in arbitrary units

VI. Discussion

In this study we introduce the quotient algorithm, which was inspired by the way humans solve the Euclidean variant of the TSP. We hypothesized that we could model human tours by taking the quotient between the cost of the distance between all non-visited nodes from their centroid, and the cost of the non-visited nodes from the current node in each step, and then selecting the node with the lowest result. We compared this algorithm on 8750 graphs ranging in node quantity from 50 to 399 nodes. Despite the **Quot** being a minor adaption of the Nearest Neighbour (NN) algorithm, where 'distance' is defined as cost-to-current/cost-to-centroid, the **Quot** performed significantly better than nearest neighbor, nearest insertion, and about as well as cheapest insertion. However, the **Quot** performed worse than farthest Insertion. Nevertheless, because of its simplicity, we suggest that the **Quot** algorithm be considered as a potential candidate for inclusion in benchmark tests of heuristic performance in the travelling salesperson problem.

References

[1] Badcock, D. R., Hess, R. F., & Dobbins, K. (1996). Localization of element clusters: Multiple cues. Vision Research, 36(10), 1467-1472.

[2] Dry, M., Lee, M.D., Vickers, D., & Hughes, P. (2006). Human Performance on Visually Presented Traveling Salesperson Problems with Varying Numbers of Nodes. The Journal of Problem Solving, 1(1), doi: 10.7771/1932-6246.1004

[3] Dry, M. J., & Fontaine, E. L. (2014). Fast and Efficient Discrimination of Traveling Salesperson Problem Stimulus Difficulty. The Journal of Problem Solving, 7(1), 9.

[4] Gandhi, R. (2001). Approximate solution for the Traveling Salesman's Problem Using Continuous Hopfield Network. ECE 559: Traveling Salesman's Problem's Solution using Hopfield NN, 1-9

[5] Graham, S. M, Joshi, A., & Pizlo Z. (2000). The Traveling Salesman Problem: A hierarchical model. Memory & Cognition, 28, 1191–1204.

[6] Hahsler M., and Hornik K. (2007), TSP - Infrastructure for the traveling salesperson problem. Journal of Statistical Software 23/2. URL: http://www.jstatsoft.org/v23/i02/.

[7] Hahsler M. and Hornik K. (2016). TSP: Traveling Salesperson Problem (TSP). R package version 1.1-4. https://CRAN.R-project.org/package=TSP

[8] Kirkpatrick, S. (1984). Optimization by simulated annealing: Quantitative studies. Journal of statistical physics, 34(5-6), 975-986.

[9] Kyritsis, M., Gulliver, S. R., & Feredoes, E. (2017). Acknowledging crossing-avoidance heuristic violations when solving the Euclidean travelling salesperson problem. Psychological Research, 1-13, doi:10.1007/s00426-017-0881-7.

[10] Larrañaga, P., Kuijpers, C.M.H., Murga, R.H., Inza, I., & Dizdarevic, S. (1999). Genetic Algorithms for the Travelling Salesman Problem: A Review of Representations and Operators. Artificial Intelligence Review 13(2), 129-170

[11] MacGregor, J. N. & Ormerod, T. (1996). Human performance on the traveling salesman problem. Perception & Psychophysics, 58(4), 527-539

[12] MacGregor, J. N., Chronicle, E. P., & Ormerod, T. C. (2004). Convex hull or crossing avoidance? Solution heuristics in the traveling salesperson problem. Memory & cognition, 32(2), 260-270.

[13] Morgan, M. J., & Glennerster, A. (1991). Efficiency of locating centres of dot-clusters by human observers. Vision research, 31(12), 2075-2083.

[14] R Core Team (2016). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

[15] Rosenkrantz, D. J., Stearns, R. E., & Lewis, II, P. M. (1977). An analysis of several heuristics for the traveling salesman problem. SIAM journal on computing, 6(3), 563-581.

[16] Seizova-Cajic, T., & Gillam, B. (2006). Biases in judgments of separation and orientation of elements belonging to different clusters. Vision research, 46(16), 2525-2534.

[17] Valenzuela, C.L., Jones, A.J. (1997). Estimating the Held-Karp lower bound for the geometric TSP. European Journal of Operational Research.102(1), 157–175

[18] van Rooij, I., Stege, U., & Schactman, A. (2003). Convex hull and tour crossings in the Euclidean traveling salesperson problem: Implications for human performance studies. Memory & cognition, 31(2), 215-220

[19] Vickers, D., Lee, M. D., Dry, M., & Hughes, P. (2003). The Roles of the Convex Hull and Number of Intersections Upon Performance on Visually Presented Traveling Salesperson Problems. Memory & Cognition, 31(7), 1094-1104