Free and Open Source Software for Geospatial (FOSS4G) Conference Proceedings

Volume 17 *Boston, USA* Article 5

2017

A New Spatial Approach for Efficient Transformation of Equality - Generalized TSP to TSP

Mohammed Zia
National Innovation & Research Center for Geographical Information Technologies, Turkey

Ziyadin Cakir
Istanbul Technical University

Dursun Zafer Seker
Istanbul Technical University

A New Spatial Approach for Efficient Transformation of Equality - Generalized TSP to TSP

Mohammed Zia[a,c,*], Ziyadin Cakir[b,c], Dursun Zafer Seker[a,c]

[a] *Geomatics Engineering Department, Faculty of Civil Engineering*

[b] *Department of Geology, Faculty of Mines*

[c] *Istanbul Technical University, Turkey*

Abstract: The Equality - Generalized Travelling Salesman Problem (E-GTSP) asks to find a *Hamiltonian* cycle visiting each group exactly once, where each group represents a type of visiting node. This can represent a range of combinatorial optimization problem of NP-hard type like planning, logistics, etc. Its solution requires transformation of E-GTSP to TSP before solving it using a given TSP solver. This paper presents 5 different search-algorithms for optimal transformation which considers spatial spread of nodes of each group. Algorithms are tested over 15 cities with different street-network's fractal-dimension for 5 instances of group-counts each. It's observed that the R-Search algorithm, which selects nodes from each group depending upon their radial separation with respect to the start-end point, is the optimal search criterion among all other algorithms with a mean length error of 8.8%. This study will help developers and researchers to answer complex routing problems from a spatial perspective.

*Corresponding author
Email address: zia@itu.edu.tr,mohammed.zia33@gmail.com (Mohammed Zia)
URL: https://linkedin.com/in/zia33 (Mohammed Zia)

1. Introduction

The Travelling Salesman Problem (TSP) is one of the well-known and extensively studies combinatorial optimization problem which demands to estimate the shortest path in terms of length, time, etc. that visits each station/node/vertex in a given group/set exactly once before returning to the starting point. Mathematically, it could be stated as follows: Given a directed/undirected graph $G = (V, E)$ with set of vertices V and set of weighted edges E, find the shortest path between start vertex s and end vertex e (e could be same as s for closed path) that visits each vertex for a given set $V' \subseteq V - \{s, e\}$ exactly once. It is nothing but calculating the minimum-cost *Hamiltonian* cycle in G Hamilton 1856. It has always remained a great source of attraction due to its ranging areas of application like routing, communication networking, sequencing and scheduling, etc. Lenstra and Kan 1975. A detailed classification of types of TSP and possible solutions can be found at Psaraftis et al. 2016.

A variety of heuristic and tabu search algorithms which are constructive are developed by researchers in the past to solve this NP-hard problem, like Nearest Neighbour Gutin and Yeo 2007, Clarke-Wright heuristic Clarke and Wright 1964, Minimum Spanning Tree heuristic (eg. Kruskal's algorithm) Rosenkrantz et al. 2009, Christofides heuristic T. and Roberto 2015. Other algorithms include • K-OPT Croes 1958, • Tabu search Glover 1990, • Simulated Annealing Kirkpatrick et al. 1983, • Held-Karp lower bound Valenzuela and Jones 1997, • Lin-Kernighan algorithm Helsgaun 2000, • Lin-Kernighan-Helsgaun Helsgaun 2009, and • Cutting Plane and Branch-Bound techniques. A good description is also given in TSP Wiki page and Curtin et al. 2014.

Let $G = (V, E)$ be a graph where $V = \{v_1, v_2, ..., v_n\}$ is the set of vertex, $E = \{(v_i, v_j) \mid i \neq j; v_i, v_j \in V\}$ is the edge set, and $W = \{w_{ij}\}$ is the non-negative cost or weightage defined on E. If E is undirected then directions are irrelevant and $(v_i, v_j) = (v_j, v_i)$. Now, in the GTSP, V is partitioned into x mutually exclusive and exhaustive groups such that $V^g = \{V_1, V_2, ..., V_x\}$ and $V = V_1 \cup V_2 \cup ... \cup V_x$ with $V_\alpha \cap V_\beta = \varnothing$ for all $\alpha, \beta = 1, 2, ..., x$ and $\alpha \neq \beta$. The Generalized-TSP (GTSP) asks to determine the shortest *Hamiltonian* circuit passing through each group *at least once* (introduced independently by Henry-Labor 1969, Saksena 1970, and Srivastava et al. 1969) or *exactly once* (by Noon and Bean 1991, and Laporte and Nobert 1983). The *exactly once* case of GTSP is known as the Equality-GTSP or E-GTSP Helsgaun 2015, where the route contains exactly one station from each V^g group. It has prime relevance in location-based problems, urban planning, postal routing, logistics, microchips manufacturing, telecommunication problems, and railway track optimization. Its existing solutions, however, are difficult to replicate by general users for TSP models representing GIS problems, primarily vehicle-routing.

In order to transform the E-GTSP to TSP, authors have tried to think from a spatial perspective as the underlying model primarily represents an urban setup. Once done, it could be solved using the state-of-the-art Lin-Kernighan-Helsgaun TSP solver Helsgaun 2015. Five different search algorithms are presented and analysed on OpenStreetMap (OSM) street dataset for best possible transformation. Different group-counts (i.e. $|V^g| = 1, 2, 3, 4, 5$) are modelled to test proposed algorithms at different complexity levels and results are discussed in the Results and Discussion Section 5. In the following sections, the proposed search algorithms are explained and tested, and finally a commentary is provided in the Conclusion Section 6.

2. Transformation of E-GTSP to TSP

The GTSP was introduced through the record balancing problems aroused in computer design by Henry-Labor 1969, Srivastava et al. 1969, and Saksena 1970. The shortcoming of E-GTSP to TSP transformation is that it increases problem's dimension drastically, and therefore it becomes practically unfeasible to solve. Dynamic programmatically, an E-GTSP's solution is based upon deciding the following two **decisions** in written order:

1 Selection of a vertex subset V^s, also termed as station, such that $V^s \subseteq V$ and $V^s \cap V_\alpha = 1$ for all $\alpha = 1, 2, ..., x$. *Note* that, $|V^s| = |V^g|$.

2 Calculation of the minimum-cost *Hamiltonian* cycle in subgraph $G^s = (V^s, E^s)$ of G produced by V^s.

The presented search algorithms do not increase problem's size by increasing vertex or edge count. It is much easier to filter-out distant and sub-optimal vertices from each group considering their spatial spread with respect to the end vertices. In this article, *points* represent all possible vertices of V including start/end, *group* represents each set of vertex from V exhibiting one particular type of vertex, and *station* represents the selected point from each group.

3. Methodology

Five different search criteria to select one point from each group are presented here (Fig. 1). Although a number of such algorithms are possible, authors believe presented ones to be mutually exclusive and covering a range of search possibilities.

3.1. E-Search and D-Search

Euclidean-Search (E-Search) is based upon the euclidean line between start-end points in a given instance. The basic approach behind this is to connect both the start/end points by a straight line and select the closest point from each group with respect to this line (Fig. 1). It involves two stages before reducing E-GTSP to TSP. It is $O(n)$, where $n = |V|$.

Dijkstra-Search (D-Search), on the other hand, involves calculating Dijkstra route between given start/end points before selecting the closest point with respect to that route (Fig. 1). Computationally it is more expensive than E-Search, with $O(n^2)$. Algorithm 1 is the pseudo-code of the above two search algorithms.

3.2. R-Search

Radial-Search (R-Search) is based upon the radial distance of stations from start/end points for a given E-GTSP instance. Closest point from each group is selected twice, making $|V^s| = 2 \times x$ (Section 2), leading 2^x different TSPs, making **decision 2** computationally expensive (Fig. 1). It involves an O-complexity of $O(n)$. Furthermore, it evaluates points lying outside the proximity of start-end region more efficiently, unlike E-Search and D-Search.

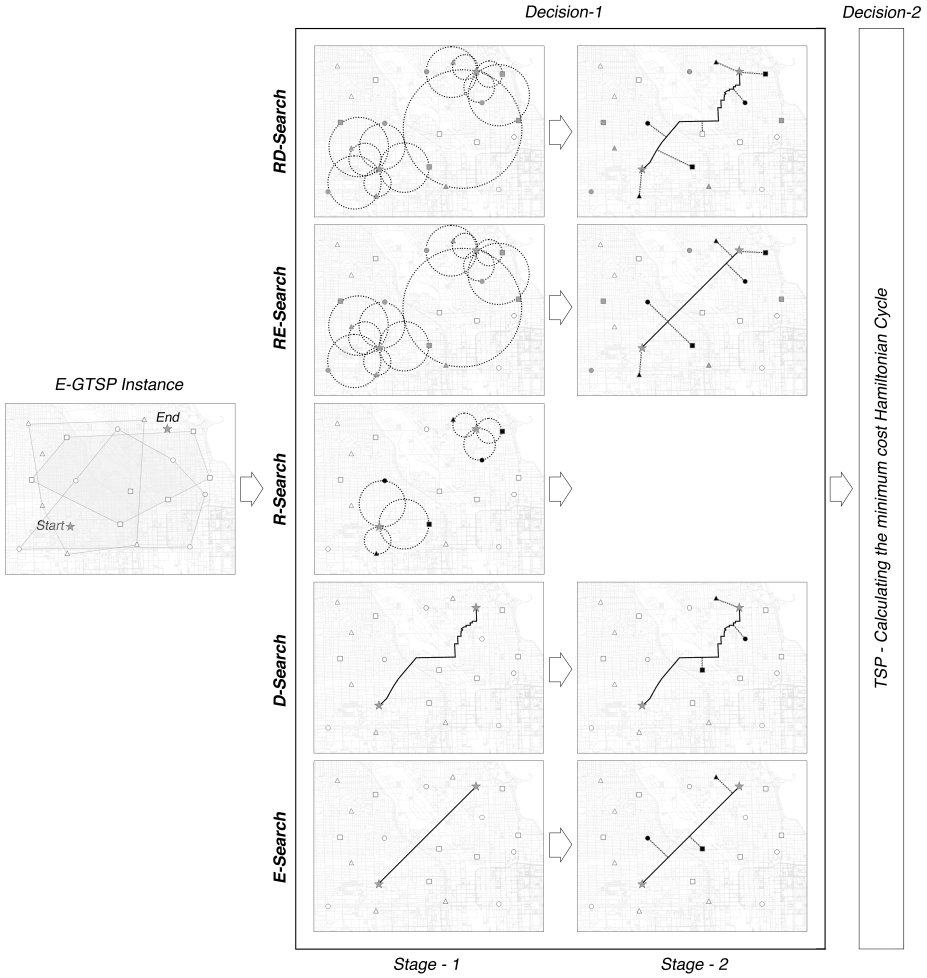

Figure 1: Diagrammatic representation of 5 different proposed search algorithms. Three groups of different kind of points, marked by oval, triangular, and rectangular markers, are used to develop an E-GTSP instance. It should be noted that for *R-Search*, *RE-Search*, and *RD-Search* there are two stations from each group after **decision 1** (Section 2), thus, generating 8 ($2 \times 2 \times 2$, for drawn instance) possible TSPs for **decision 2**.

3.3. RE-Search and RD-Search

Finally, the last two search algorithms are a hybrid of R-Search and E-Search (RE-Search), and R-Search and D-Search (RD-Search). They first find the radially closest two points from each group with respect to both the start/end points independently, making $|V^s| = 2 \times 2 \times x$. Later they select the closest stations from both the ends with respect to the euclidean line (for RE-Search) or Dijskstra route (for RD-Search), Fig. 1. They have the complexities of $O(n)$ and $O(n^2)$, respectively. Similar to R-Search, the second step leads to a total of 2^x TSPs. Once estimated, the optimal V^s from each group it is solved by Simulated Annealing Kirkpatrick et al. 1983 TSP Solver.

4. Study Area and Data-Set Used

OSM dataset of 15 cities belonging to five different fractal dimension (frac-D) bins were used to test proposed algorithms, out of 210 cities worldwide. Frac-D value is directly proportional to road density of a region with 1-D representing area with just one road section and 2-D representing area completely filled up with roads. Bin size is kept 0.1 for highly resolved classes

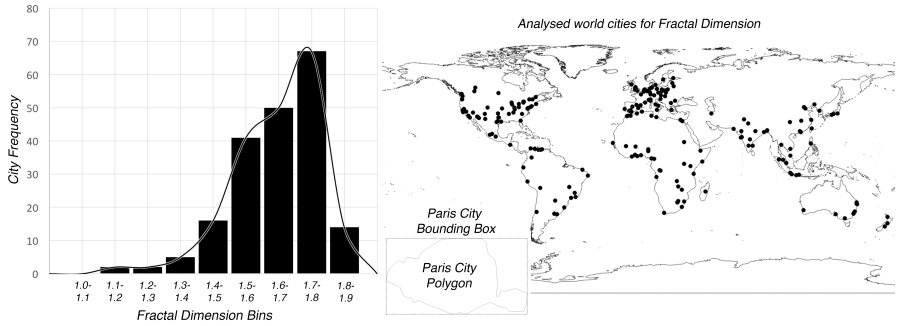

Figure 2: Histogram showing the frequency of cities (marked on the world map *right*) for each Fractal-Dimension bin. Increase in dimension represents increase in road-density. 3 cities are selected from each top 5 bins to test the proposed search algorithms. Each city-polygon's bounding box is used to download corresponding OSM XML data from its Overpass API.

(Fig. 2). **F**ractal **D**imension **C**alculator is used to calculate the dimension of each city Bourke 2003. All 105 start-end points to test algorithms are randomly generated and 5 points from each group are likewise selected from OSM point dataset to better represent physical stops in the city. Observations regarding best search algorithm with increased complexity are discussed in Section 5.

5. Results and Discussion

5 different instances, i.e. $|V^s|$, are modelled for each city to test each algorithm. Fig. 3a is a 3D-plot between *different search algorithms* used, *different number of stations to be visited*, i.e. $|V^g|$, and *different type of street-networks*, where each circle represents the average fractional error in E-GTSP route-length coming out of all 105 start-end pairs estimated with respect to the optimal route (by brute-force). There are 375 ($15\texttt{cities} \times 5\texttt{group-counts} \times 5\texttt{algorithms}$) colored circles, with black ones representing error more than 20%. It is clear from the horizontal color tone variation that the percentage error in route length will increase with higher $|V^s|$ instances (marked by white arrow Fig. 3a) irrespective of the algorithm used. The whole plot is divided into 5 different cross-sections representing each group-count. Fig. 3b represents 5 graphs between *different search algorithms* and *different type of street-networks* for different values of $|V^s|$. The black-boxed circle marks the lowest fractional error for that row. It is observed that for lower $|V^s|$, the D-Search algorithm gives lowest percentage error for most of the cities irrespective of their frac-Ds. However, this out-performance gets shifted towards R-Search for higher group-counts, with performance of similar intensity at $|V^s| = 5$ as that for D-Search at $|V^s| = 1$. It is, therefore, concluded that for complex E-GTSP scenarios it is better to select radially close points from each group with respect to the start-end points than to find points closest to their corresponding Dijkstra-route.

6. Conclusion

The E-GTSP, which is an extension of TSP, is proven by many researchers to better model real-life combinatorial optimization problems, where the task is to estimate open/close *Hamiltonian* cycle visiting each group exactly once in a given graph. A general approach is to reduce it to corresponding TSP before solving it to optimality. In this article, 5 different spatially driven search algorithms are presented and tested to transform E-GTSP to TSP. They have been tested out for 15 cities selected based upon the frac-D for 5 different $|V^s|$ instances each.

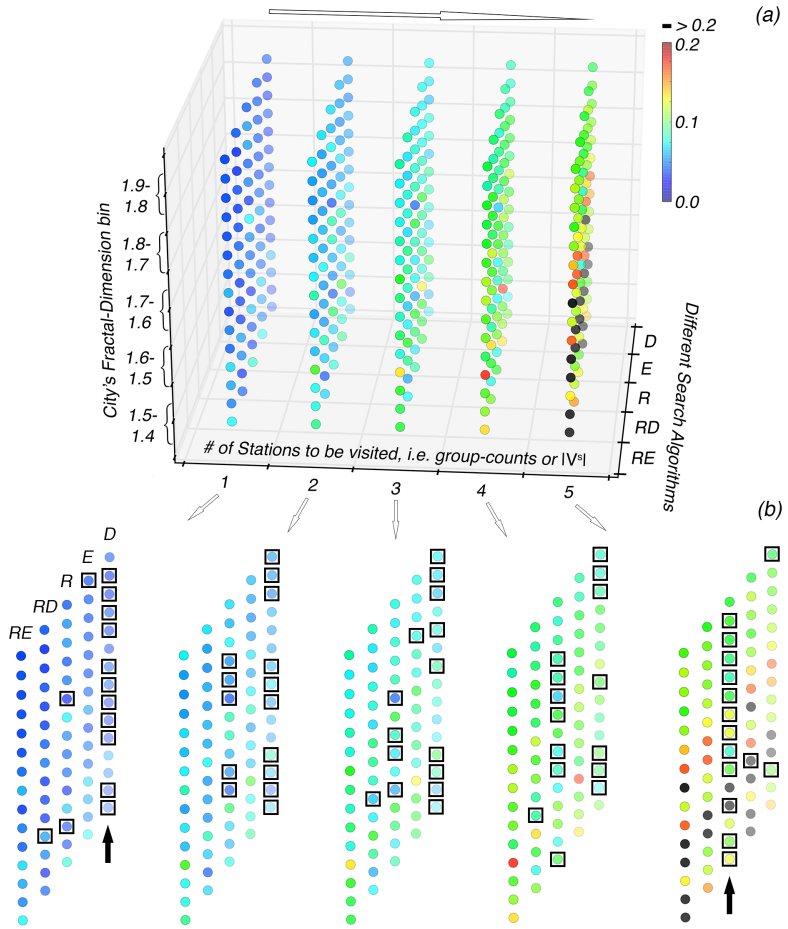

Figure 3: 3D-graph between *Different Search Algorithms*, *Number of Stations to be visited*, i.e. $|V^s|$, and *City's Fractal-Dimension bin*. Each bin represents 3 cities taken from Fig. 2. Color represents the average fractional error in route-length coming out from each algorithm with respect to the optimal one (estimated by Brute-Force), with black color for errors more than 20%. (a) It should be noted that with increase in $|V^s|$ for a given E-GTSP instance the error increases, irrespective of the choice of the algorithm (marked with big arrow). (b) Graph is sliced-down for each group-count, individually. For each city the algorithm with least error is marked with *black box*. It is clear that for small $|V^s|$ values the *D-Search* approach is better, and as one increases the number of stations the *R-Search* outperforms other, marked by solid-arrows.

It is observed that with increase in instance complexity all algorithms get erroneous, thereby giving longer routes which is independent of the choice of city street-pattern (Fig. 3a). For lower $|V^s|$ instances D-Search and for higher $|V^s|$ instances R-Search is better among all given algorithms (Fig. 3b). Overall, R-Search is the best approach, qualitative-wise, for tested cities by giving least route-length values. Its win could be attributed by its ability to consider stations belonging to regions outside the start-end proximity.

This study, thus, has brought forth a new search criterion of point/vertex selection from each group for an adequate E-GTSP to TSP transformation, tested on OSM dataset. This considers the radial spread of all points from each group with respect to the start/end point for optimal selection. It has widened up research possibilities in this pursuit; and researchers are encouraged to use tested dataset provided by Zia 2016 as benchmark for subsequent studies in order to escape long brute-force looping.

Future research will include testing R-Search criterion with other algorithms, like the state-of-the-art Lin-Kernighan-Helsgaun TSP solver, as discussed by researchers. Additional work

might involve developing heuristic-R-Search or ANN-R-Search algorithm. Authors are optimistic that this attempt has added up few novel concepts to understand the old famous GTSP problem.

References

Bourke, P., 2003. Fractal Dimension Calculator - FDC. Online, online; accessed 15-November-2016. URL http://paulbourke.net/fractals/fracdim/

Clarke, G., Wright, J. W., August 1964. Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. Operations Research 12(4), 568–581.

Croes, G. A., December 1958. A Method for Solving Traveling-Salesman Problems. Operations Research, 791–812.

Curtin, K. M., Voicu, G., Rice, M. T., Stefanidis, A., April 2014. A Comparative Analysis of Traveling Salesman Solutions from Geographic Information Systems. Transactions in GIS 18(2), 286–301.

Glover, F., August 1990. Tabu Search: A Tutorial. Interfaces, 74–94.

Gutin, G., Yeo, A., 2007. The Greedy Algorithm for the Symmetric TSP. Algorithmic Operations Research 2(1), 33–36.

Hamilton, W. R., December 1856. Memorandum respecting a new system of roots of unity. Philosophical Magazine 12, 446.

Helsgaun, K., 2000. An Effective Implementation of the Lin-Kernighan Traveling Salesman Heuristic. European Journal of Operational Research 126, 106–130.

Helsgaun, K., October 2009. General k-opt submoves for the LinKernighan TSP heuristic. Mathematical Programming Computation 1(2), 119–163.

Helsgaun, K., September 2015. Solving the equality generalized traveling salesman problem using the LinKernighanHelsgaun Algorithm. Operations Research Letters 7(3), 269–287.

Henry-Labor, A. L., 1969. The record balancing problem: A dynamic programming solution of a genera salesman problem. RAIRO - Operations Research 2, 43–49.

Kirkpatrick, S., Jr., C. D. G., Vecchi, M. P., May 1983. Optimization by Simulated Annealing. Science 220(4598), 671–680.

Laporte, G., Nobert, Y., 1983. Generalized Travelling Salesman Problem Through n Sets Of Nodes: An Integer Programming Approach. INFOR: Information Systems and Operational Research 21(1), 61–75.

Lenstra, J. K., Kan, A. H. G. R., November 1975. Some Simple Applications of the Travelling Salesman Problem. Operational Research Quarterly 26(4), 717–733.

Noon, C. E., Bean, J. C., August 1991. A Lagrangian Based Approach for the Asymmetric Generalized Traveling Salesman Problem. International Journal of Production Research 39(4), 623–632.

Psaraftis, H. N., Wen, M., Kontovas, C. A., January 2016. Dynamic vehicle routing problems: Three decades and counting. Networks 67(1), 3–31.

Rosenkrantz, D. J., Stearns, R. E., II, P. M. L., 2009. An analysis of several heuristics for the traveling salesman problem. Fundamental Problems in Computing 1, 45–69.

Saksena, J. P., November 1970. Mathematical model for scheduling clients through welfare agencies. Canadian Operational Research Society 8(3), 185.

Srivastava, S. S., Kumar, S., Garg, R. C., Sen, P., July 1969. Generalized travelling salesman problem through n sets of nodes. Canadian Operational Research Society 7(2), 97.

T., G. M., Roberto, T., 2015. The Christofides Approximation Algorithm. Algorithm Design and Applications 18(1), 513514.

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

Zia, M., 2016. E-GTSP-to-TSP-tested-instances. Online, online; accessed 15-November-2016. URL https://github.com/Zia-/E-GTSP-to-TSP-tested-instances

Algorithm 1 ***E-Search*** & ***D-Search*** pseudo-code

Require: Terminal points S and F, $G = (V, E)$ representing urban street-network, & $V^g = \{V_1, V_2, ..., V_x\}$, i.e. points' groups set, where $V_i \subseteq V$ & $V_i \cap V_j = \varnothing \ \forall \ i, j = 1, 2, ..., x$ & $i \neq j$.

procedure E-GTSP $\Rightarrow$ TSP
  if Algorithm 1 $\widehat{=}$ ***E-Search*** **then** Draw SF **Euclidean** Line
  end if ▷ $\widehat{=}$ means corresponds to
  if Algorithm 1 $\widehat{=}$ ***D-Search*** **then** Calculate SF **Dijkstra** Route
  end if
  Initialize an empty container $C1$
  for $V_i \in V^g \ \forall \ i = 1, 2, .., x$ **do**
    Initialize an empty container $C2$.
    for $v_m \in V_i \ \forall \ m = 1, 2, .., |V_i|$ **do**
      $C2 \leftarrow v_m - SF$ shortest Euclidean distance ▷ $\leftarrow$ means append
    end for
    return $C2$
    $C1 \leftarrow v \ \widehat{=} \ v_m - SF \in C2$, with the smallest Euclidean distance
  end for
  return $C1$
end procedure
procedure TSP
  Calculate the minimum-cost *Hamiltonian* cycle induced by $C1$. *Note*: $C1 = V^s$.
end procedure