REPORT

The travelling salesman problem（TSP)

TAMEEM HEZAM
GRADUATE STUDENT IN NJUST

CONTENTS

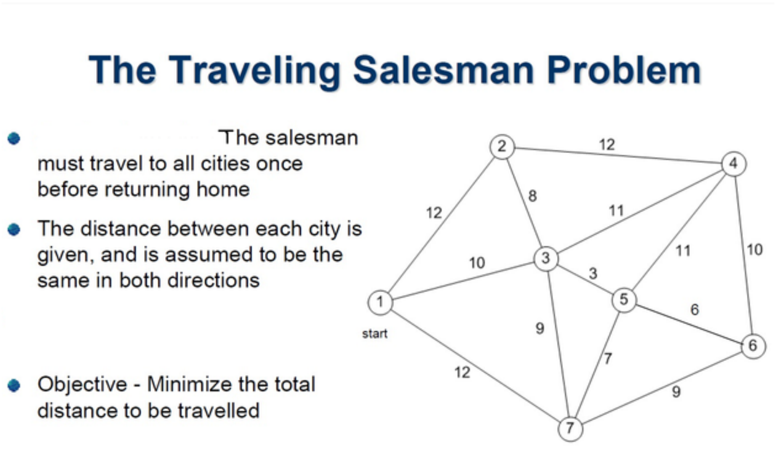
TAMEEM HEZAM

• Abstract:

Since Pokemon Go sent millions on the quest of collecting virtual monsters, an important question has been on the minds of many people: Is going after the closest item first a time-and-cost-effective way to play? Here, we show that this is in fact a good strategy which performs on average only 7% worse than the best possible solution in terms of the total distance traveled to gather all the items. Even when accounting for errors due to the inability of people to accurately measure distances by eye, the performance only goes down to 16% of the optimal solution.

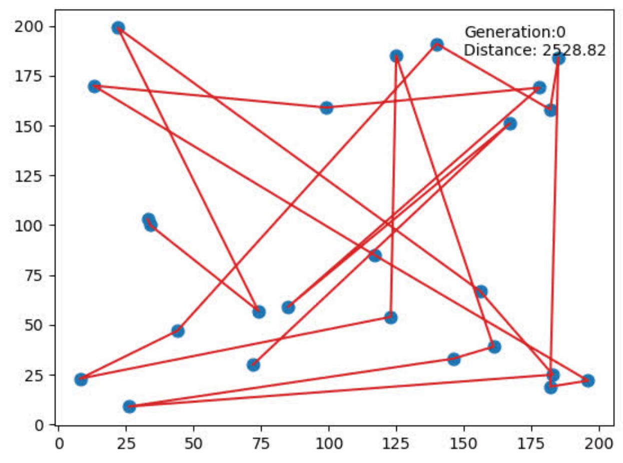
• an introdaction

The travelling salesman problem (TSP) asks the following question: "Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the origin city?" It is an NP-hard problem in combinatorial optimization, important in theoretical computer science and operations research. The travelling purchaser problem and the vehicle routing problem are both generalizations of TSP. In the theory of computational complexity, the decision version of the TSP (where given a length L, the task is to decide whether the graph has a tour of at most L) belongs to the class of NP-complete problems. Thus, it is possible that the worst-case running time for any algorithm for the TSP increases superpolynomially (but no more than exponentially) with the number of cities. The problem was first formulated in 1930 and is one of the most intensively studied problems in optimization. It is used as a benchmark for many optimization methods. Even though the problem is computationally difficult, many heuristics and exact algorithms are known, so that some instances with tens of thousands of cities can be solved completely and even problems with millions of cities can be approximated within a small fraction of 1%. The TSP has several applications even in its purest formulation, such as planning, logistics, and the manufacture of microchips. Slightly modified, it appears as a sub-problem in many areas, such as DNA sequencing. In these applications, the concept city represents, for example, customers, soldering points, or DNA fragments, and the concept distance represents travelling times or cost, or a similarity measure between DNA fragments. The TSP also appears in astronomy, as astronomers observing many sources will want to minimize the time spent moving the telescope between the sources. In many applications, additional constraints such as limited resources or time windows may be imposed.

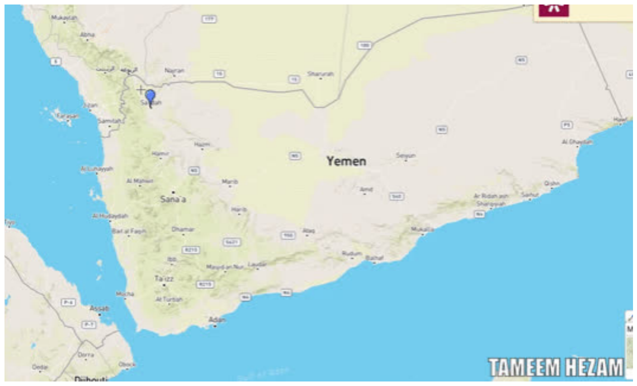

The Traveling Salesman Problem

- The salesman must travel to all cities once before returning home
- The distance between each city is given, and is assumed to be the same in both directions
- Objective - Minimize the total distance to be travelled

• image (1) Illustration of the traveling salesman problem (TSP)



• image (2) Solution of a travelling salesman problem: the black line shows the shortest possible loop that connects every red dot.



• image (3) Yemen map- Solution of a travelling salesman problem: the red line shows the shortest possible loop that connects every red dot.

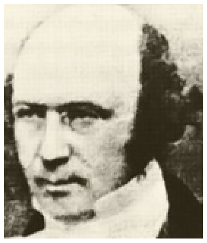
• History

The origins of the travelling salesman problem are unclear. A handbook for travelling salesmen from 1832 mentions the problem and includes example tours through Germany and Switzerland, but contains no mathematical treatment. The travelling salesman problem was mathematically formulated in the 1800s by the Irish mathematician W.R. Hamilton and by the British mathematician Thomas Kirkman. Hamilton's icosian game was a recreational puzzle based on finding a Hamiltonian cycle. The general form of the TSP appears to have been first studied by mathematicians during the 1930s in Vienna and at Harvard, notably by Karl Menger, who defines the problem, considers the obvious brute-force algorithm, and observes the non-optimality of the nearest neighbour heuristic: We denote by messenger problem (since in practice this question should be solved by each postman, anyway also by many travellers) the task to find, for finitely many points whose pairwise distances are known, the shortest route connecting the points. Of course, this problem is solvable by finitely many trials. Rules which would push the number of trials below the number of permutations of the given points are not known. The rule that one first should go from the starting point to the closest point, then to the point closest to this, etc., in general, does not yield the shortest route.It was first considered mathematically in the 1930s by Merrill M. Flood who was looking to solve a school bus routing problem. Hassler Whitney at Princeton University generated interest in the problem, which he called the "48 states problem". The earliest publication using the phrase "travelling salesman problem" was the 1949 RAND Corporation report by Julia Robinson, "On the Hamiltonian game (a travelling salesman problem)In the 1950s and 1960s, the problem became increasingly popular in scientific circles in Europe and the US after the RAND Corporation in Santa Monica offered prizes for steps in solving the problem. Notable contributions were made by George Dantzig, Delbert Ray Fulkerson and Selmer M. Johnson from the RAND Corporation, who expressed the problem as an integer linear program and developed the cutting plane method for its solution. They wrote what is considered the seminal paper on the subject in which with these new methods they solved an instance with 49 cities to optimality by constructing a tour and proving that no other tour could be shorter. Dantzig, Fulkerson and Johnson, however, speculated that given a near-optimal solution we may be able to find optimality or prove optimality by adding a small number of extra inequalities (cuts). They used this idea to solve their initial 49 city problem using a string model. They found they only needed 26 cuts to come to a solution for their 49 city problem.Gerhard Reinelt published the TSPLIB in 1991, a collection of benchmark instances of varying difficulty, which has been used by many research groups for comparing results. In 2006,

While this paper did not give an algorithmic approach to TSP problems, the ideas that lay within it were indispensable to later creating exact solution methods for the TSP, though it would take 15 years to find an algorithmic approach in creating these cuts. As well as cutting plane methods, Dantzig, Fulkerson and Johnson used branch and bound algorithms perhaps for the first time. In 1959, Jillian Beardwood, J.H. Halton and John Hammersley published an article entitled "The Shortest Path Through Many Points" in the journal of the Cambridge Philosophical Society. The Beardwood–Halton–Hammersley theorem provides a practical solution to the travelling salesman problem.

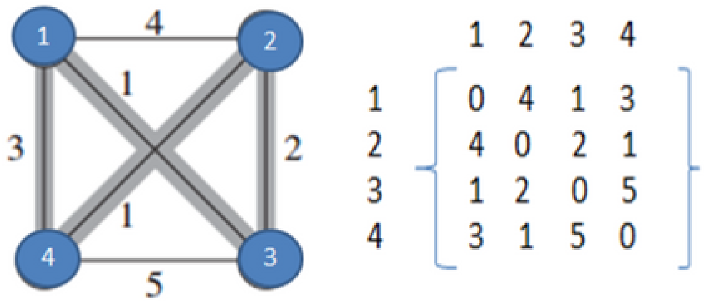
The authors derived an asymptotic formula to determine the length of the shortest route for a salesman who starts at a home or office and visits a fixed number of locations before returning to the start. In the following decades, the problem was studied by many researchers from mathematics, computer science, chemistry, physics, and other sciences. In the 1960s however a new approach was created, that instead of seeking optimal solutions, one would produce a solution whose length is provably bounded by a multiple of the optimal length, and in doing so create lower bounds for the problem; these may then be used with branch and bound approaches. One method of doing this was to create a minimum spanning tree of the graph and then double all its edges, which produces the bound that the length of an optimal tour is at most twice the weight of a minimum spanning tree In 1976, Christofides and Serdyukov independently of each other made a big advance in this direction: the Christofides-Serdyukov algorithm yields a solution that, in the worst case, is at most 1.5 times longer than the optimal solution. As the algorithm was so simple and quick, many hoped it would give way to a near-optimal solution method. This remains the method with the best worst-case scenario. However, for a fairly general special case of the problem, it was beaten by a tiny margin in 2011. Richard M. Karp showed in 1972 that the Hamiltonian cycle problem was NP-complete, which implies the NP-hardness of TSP. This supplied a mathematical explanation for the apparent computational difficulty of finding optimal tours. Great progress was made in the late 1970s and 1980, when Grötschel, Padberg, Rinaldi and others managed to exactly solve instances with up to 2,392 cities, using cutting planes and branch and bound. In the 1990s, Applegate, Bixby, Chvátal, and Cook developed the program Concorde that has been used in many recent record solutions.

Cook and others computed an optimal tour through an 85,900-city instance given by a microchip layout problem, currently the largest solved TSPLIB instance. For many other instances with millions of cities, solutions can be found that are guaranteed to be within 2–3% of an optimal tourIn 2020, a slightly improved approximation algorithm was developed.


William Rowan Hamilton

• Description

1. *As a graph problem:* TSP can be modelled as an undirected weighted graph, such that cities are the graph's vertices, paths are the graph's edges, and a path's distance is the edge's weight. It is a minimization problem starting and finishing at a specified vertex after having visited each other vertex exactly once. Often, the model is a complete graph (i.e., each pair of vertices is connected by an edge). If no path exists between two cities, adding an arbitrarily long edge will complete the graph without affecting the optimal tour



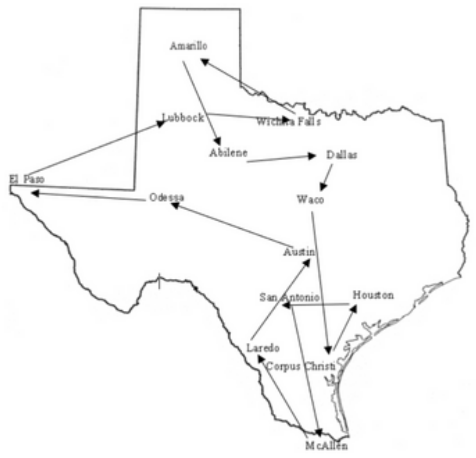
$$\begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 4 & 1 & 3 \\ 2 & 4 & 0 & 2 & 1 \\ 3 & 1 & 2 & 0 & 5 \\ 4 & 3 & 1 & 5 & 0 \end{array}$$

• **GRAPH (3) Symmetric TSP with four cities**

2. *Asymmetric and symmetric:* TSP can be modelled as an undirected weighted graph, such that cities are the graph's vertices, paths are the graph's edges, and a path's distance is the edge's weight. It is a minimization problem starting and finishing at a specified vertex after having visited each other vertex exactly once. Often, the model is a complete graph (i.e., each pair of vertices is connected by an edge). If no path exists between two cities, adding an arbitrarily long edge will complete the graph without affecting the optimal tour

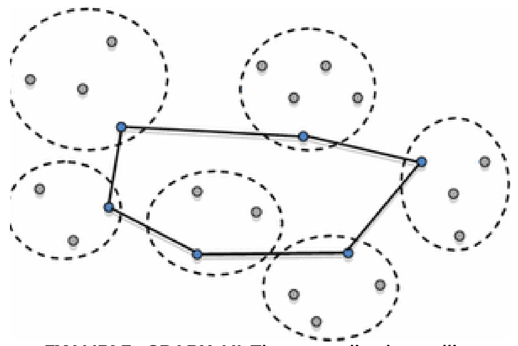
3. *Related problems:*

- An equivalent formulation in terms of graph theory is: Given a complete weighted graph (where the vertices would represent the cities, the edges would represent the roads, and the weights would be the cost or distance of that road), find a Hamiltonian cycle with the least weight.
- The requirement of returning to the starting city does not change the computational complexity of the problem, see the Hamiltonian path problem.
- Another related problem is the Bottleneck traveling salesman problem (bottleneck TSP): Find a Hamiltonian cycle in a weighted graph with the minimal weight of the weightiest edge. For example, avoiding narrow streets with big buses.The problem is of considerable practical importance, apart from evident transportation and logistics areas. A classic example is in printed circuit manufacturing: scheduling of a route of the drill machine to drill holes in a PCB. In robotic machining or drilling applications, the "cities" are parts to machine or holes (of different sizes) to drill, and the "cost of travel" includes time for retooling the robot (single machine job sequencing problem).



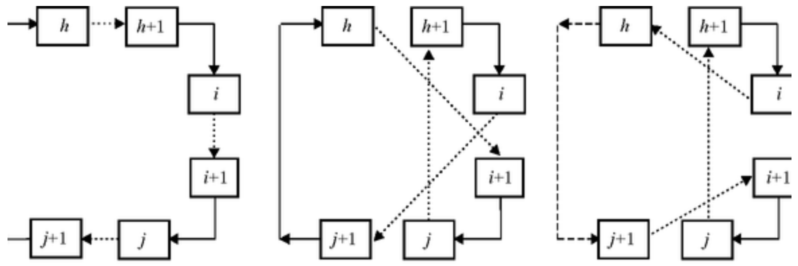
• **EXAMPLE -GRAPH (3) Solution Map of Bottleneck Traveling Salesman Problem**

- The generalized travelling salesman problem, also known as the "travelling politician problem", deals with "states" that have (one or more) "cities" and the salesman has to visit exactly one "city" from each "state". One application is encountered in ordering a solution to the cutting stock problem in order to minimize knife changes. Another is concerned with drilling in semiconductor manufacturing, see e.g., U.S. Patent 7,054,798. Noon and Bean demonstrated that the generalized travelling salesman problem can be transformed into a standard travelling salesman problem with the same number of cities, but a modified distance matrix.



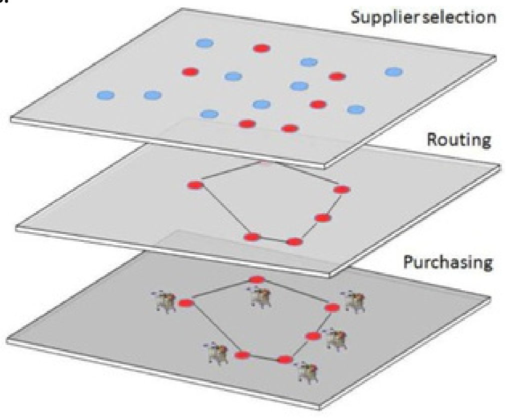
• **EXAMPLE -GRAPH (4)** The generalized travelling salesman problem

- The sequential ordering problem deals with the problem of visiting a set of cities where precedence relations between the cities exist.



- **EXAMPLE -GRAPH (5) SEQUENTIAL ORDERING PROBLEM**

- The travelling purchaser problem deals with a purchaser who is charged with purchasing a set of products. He can purchase these products in several cities, but at different prices and not all cities offer the same products. The objective is to find a route between a subset of the cities, which minimizes total cost (travel cost + purchasing cost) and which enables the purchase of all required products.



- **EXAMPLE -GRAPH (6) The Traveling Purchaser Problem and its variants**

• Integer linear programming formulations:

The TSP can be formulated as an integer linear program. Several formulations are known. Two notable formulations are the Miller–Tucker–Zemlin (MTZ) formulation and the Dantzig–Fulkerson–Johnson (DFJ) formulation. The DFJ formulation is stronger, though the MTZ formulation is still useful in certain settings.

1. *Miller–Tucker–Zemlin formulation:*

Label the cities with the numbers 1, ..., n and define:

$$x_{ij} = \begin{cases} 1 & \text{the path goes from city } i \text{ to city } j \\ 0 & \text{otherwise} \end{cases}$$

For i = 1, ..., n, let {\displaystyle u_{i}} be a dummy variable, and finally take {\displaystyle c_{ij}>0} to be the distance from city i to city j. Then TSP can be written as the following integer linear programming problem:

$$\min \sum_{i=1}^{n} \sum_{j\neq i, j=1}^{n} c_{ij} x_{ij} \colon$$

$$\begin{aligned} & x_{ij} \in \{0,1\} && i,j = 1,\ldots,n; \\ & u_i \in \mathbf{Z} && i = 2,\ldots,n; \\ & \sum_{i=1, i\neq j}^{n} x_{ij} = 1 && j = 1,\ldots,n; \\ & \sum_{j=1, j\neq i}^{n} x_{ij} = 1 && i = 1,\ldots,n; \\ & u_i - u_j + n x_{ij} \le n - 1 && 2 \le i \neq j \le n; \\ & 1 \le u_i \le n \quad 1 && 2 \le i \le n. \end{aligned}$$

The first set of equalities requires that each city is arrived at from exactly one other city, and the second set of equalities requires that from each city there is a departure to exactly one other city. The last constraints enforce that there is only a single tour covering all cities, and not two or more disjointed tours that only collectively cover all cities. To prove this, it is shown below (1) that every feasible solution contains only one closed sequence of cities, and (2) that for every single tour covering all cities, there are values for the dummy variables {\displaystyle u_{i}} that satisfy the constraints. (The dummy variables indicate tour ordering, such that {\displaystyle u_{i}<u_{j}} implies city {\displaystyle i} is visited before city {\displaystyle j}. This may be accomplished by incrementing {\displaystyle u_{i}} each time it is visited.)To prove that every feasible solution contains only one closed sequence of cities, it suffices to show that every subtour in a feasible solution passes through city 1 (noting that the equalities ensure there can only be one such tour). For if we sum all the inequalities corresponding to {\displaystyle x_{ij}=1} for any subtour of k steps not passing through city 1, we obtain:

$$nk \le (n-1)k,$$

which is a contradiction.It now must be shown that for every single tour covering all cities, there are values for the dummy variables {\displaystyle u_{i}} that satisfy the constraints.Without loss of generality, define the tour as originating (and ending) at city 1. Choose {\displaystyle u_{i}=t} if city i is visited in step t (i, t = 1, 2, ..., n). Then

$$u_i - u_j \le n - 1,$$

since {\displaystyle u_{i}} can be no greater than n and {\displaystyle u_{j}} can be no less than 1; hence the constraints are satisfied whenever {\displaystyle x_{ij}=0.} For {\displaystyle x_{ij}=1}, we have: $u_j + n x_{ij} = (t) - (t+1) + n = n - 1,$ satisfying the constraint.

02.Dantzig–Fulkerson–Johnson formulation:

Label the cities with the numbers 1, ..., n and define:

$$x_{ij} = \begin{cases} 1 & \text{the path goes from city } i \text{ to city } j \\ 0 & \text{otherwise} \end{cases}$$

Take {\displaystyle c_{ij}>0} to be the distance from city i to city j. Then TSP can be written as the following integer linear programming problem:

$$\min \sum_{i=1}^{n} \sum_{j\neq i, j=1}^{n} c_{ij} x_{ij}:$$

$$\sum_{i=1, i\neq j}^{n} x_{ij} = 1 \qquad j = 1, \ldots, n;$$

$$\sum_{j=1, j\neq i}^{n} x_{ij} = 1 \qquad i = 1, \ldots, n;$$

$$\sum_{i\in Q} \sum_{j\neq i, j\in Q} x_{ij} \leq |Q| - 1 \qquad \forall Q \subsetneq \{1, \ldots, n\}, |Q| \geq 2$$

The last constraint of the DFJ formulation ensures that there are no sub-tours among the non-starting vertices, so the solution returned is a single tour and not the union of smaller tours. Because this leads to an exponential number of possible constraints, in practice it is solved with delayed column generation.

• Computing a solution:

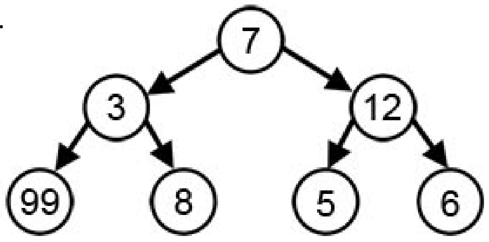
The traditional lines of attack for the NP-hard problems are the following:

1. Devising exact algorithms, which work reasonably fast only for small problem sizes.
2. Devising "suboptimal" or heuristic algorithms, i.e., algorithms that deliver approximated solutions in a reasonable time.
3. Finding special cases for the problem ("subproblems") for which either better or exact heuristics are possible.

in this problem, we will use Greedy algorithm.

• *Greedy algorithms.*

A greedy algorithm is a simple, intuitive algorithm that is used in optimization problems. The algorithm makes the optimal choice at each step as it attempts to find the overall optimal way to solve the entire problem. Greedy algorithms are quite successful in some problems, such as Huffman encoding which is used to compress data, or Dijkstra's algorithm, which is used to find the shortest path through a graph.However, in many problems, a greedy strategy does not produce an optimal solution. For example, in the animation below, the greedy algorithm seeks to find the path with the largest sum. It does this by selecting the largest available number at each step. The greedy algorithm fails to find the largest sum, however, because it makes decisions based only on the information it has at any one step, without regard to the overall problem.



graph (7)

in this graph (7) With a goal of reaching the largest sum, at each step, the greedy algorithm will choose what appears to be the optimal immediate choice, so it will choose 12 instead of 3 at the second step and will not reach the best solution, which contains 99.

Algorithm 1 Greedy algorithm

```
1: procedure GREEDY($start\_pos, C$)
2:     $current\_pos \leftarrow start\_pos$
3:     $V \leftarrow \{\}$
4:     while $|V| < |C|$ do
5:         $next\_pos \leftarrow \min_c dist(current\_pos, c), \forall c \in C \setminus V$
6:         $V \leftarrow V \cup next\_pos$
7:         $current\_pos \leftarrow next\_pos$
8:     end while
9: end procedure
```

where C is a set of the positions of all the collectibles, and dist is a function which determines the Euclidean distance between two positions in a 2D plane.While a player is immersed in the game however, it might not be possible to measure the distance between two collectibles exactly, so the playes has to guestimate it, which may result in errors. To capture this, we propose the greedy with error algorithm which includes the following modification to the line 5 in the greedy algorithm.

$$next\_pos \leftarrow \min_c dist(current\_pos, c) + \zeta, \forall c \in C \setminus V$$

where zeta is an error term from a normal distribution with cutoffs at 0.7 and 1.3 to prevent unreasonable errors.

• *Results:*

For illustration, the following excerpt from Far Cry 3 (Fig. 1) will be used, where one-time collectibles are marked by pitchers.

Figure (1). An excerpt from the map from Far Cry 3. Pitchers denote the locations of collectibles

Not surprisingly, the performance of the greedy algorithm depends on the starting location as shown in Figure Figure 2. In the first case, the route chosen by the greedy algorithm is suboptimal and results in a total distance of 3601 (Fig. 2-upper left), opposed to the optimal distance of 2764 (Fig. 2-upper right), which is a difference of 30.2%. In the second case, a different starting location is chosen which results in the greedy algorithm finding a solution that is much close to the optimal one (2961-2883 or a difference of 2.4%), as indicated by the similar routes in the second row of Figure 2.

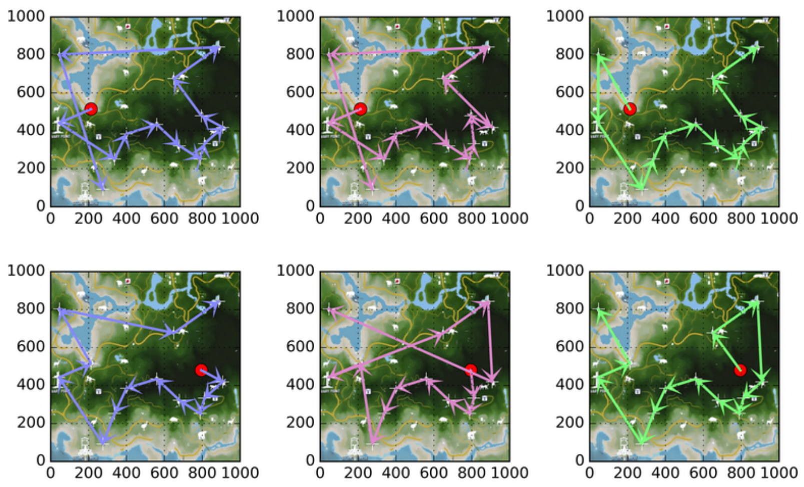


Figure (2)

Figure 2. Rows are different starting positions (red circle). Columns are different algorithms, from left to right: greedy, greedy with error (sigma=0.4) and optimal. Total distances are for the first row 3601-3740-2764; second row 2961-3923-2883. The performance of the greedy algorithm depends on the choice of starting position with the difference being much larger for the first row (30.2\% longer distance compared to the optimal) as compared to the second row (2.4\%). Furthermore the greedy algorithm with error clearly under-performs when compared to the greedy algorithm.Although, this example is intriguing a detailed analysis is needed to quantify the performance of the greedy algorithm. To this end the following setup was used: N collectibles were randomly positioned on a 2D plane (1000x1000 in size), assuming a uniform distribution. The center of the plane with coordinates (500,500) was set as the starting position. Afterwards the optimal distance to get all the collectibles was calculated as well as the total distance using the greedy algorithm. This was repeated 1000 times. Note that the actual choice of plane size does not influence our results as we are only interested in the relative performance of the greedy algorithm as compared to the optimal solution.We found that the performance of the greedy algorithm deteriorates as the number of collectibles N increases (Fig. 3). However, the overall performance was surprisingly good and for eleven collectibles N=11 it was on average only 7.3% worse than the optimal solution, with upper and lower quartiles at 2.6% and 13.9% respectively.

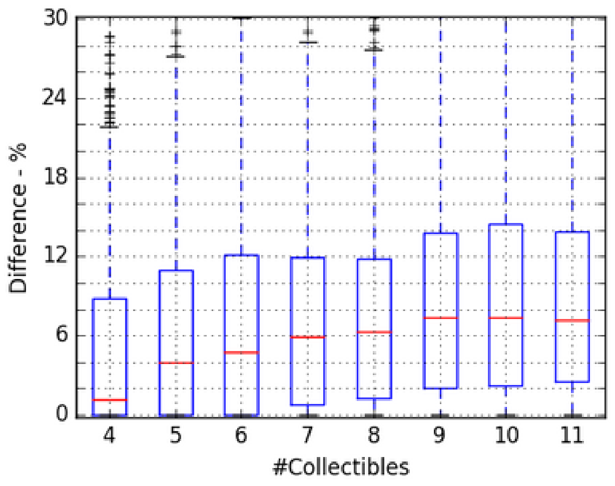


Figure (3). The average performance of the greedy algorithm compared to the optimal solution, for a different number of collectables.

Next, let us turn our attention to the greedy algorithm with error. This algorithm is not only worse than the optimal, but is also, perhaps unsurprisingly, worse than the greedy algorithm (Fig. 2). Further analysis was performed in order to estimate what effect the level of error in guestimating the distance has on the performance of the greedy algorithm with error. To this end, the number of collectibles was fixed to $N=10$ and the variance (σ) in the error term was varied. As expected, higher errors when estimating the distance between two points lead to increasingly poor performance up to $16.9%$ worse than the optimal (Fig. 4).

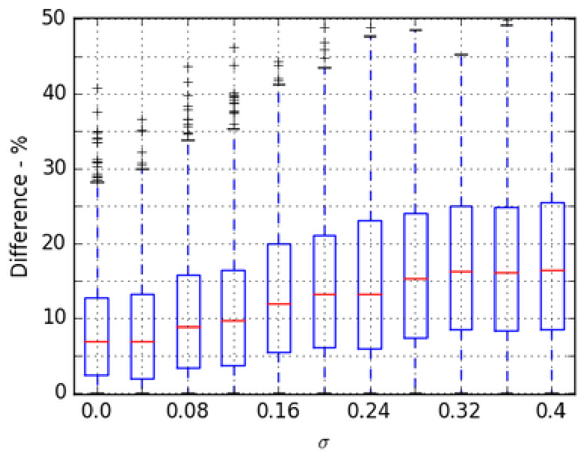


Figure (4). The average performance of the greedy algorithm with error compared to the optimal solution, for N=10 and different levels of error.

• Conclusion:

In this manuscript, we looked at how a greedy algorithm compares to an exhaustive search for a variant of the Traveling Salesman Problem, which is relevant in many open-world computer games. The main advantages of the greedy algorithm are that it can be performed by a player as he or she is playing, since no complex computations are required, as opposed to exhaustive search which can only be done by a computer. We found that the performance of the greedy algorithm is comparable to the optimal solution, with the difference depending on the number of collectables and choice of starting position. For N=11 the greedy algorithm resulted in routes 7.3% longer on average. Even when human errors in estimating the distances were included the greedy algorithm performed on average less than 16.9% worse than the optimal solution, depending on the level of error. Our analysis was unfortunately limited by the fact that finding the optimal solution is NP-hard with complexity N!, which prevented us from calculating the performance for N > 13. Nevertheless, the results suggest that players who use the greedy strategy of always going after the closest item first, are in fact playing close to optimally in terms of gathering collectables.

• References:

1. Bellman, Richard. *"Dynamic programming treatment of the travelling salesman problem."* Journal of the ACM (JACM) 9.1 (1962): 61-63.
2. Larrañaga, Pedro, et al. "Genetic algorithms for the travelling salesman problem: A review of representations and operators." *Artificial Intelligence Review* 13.2 (1999): 129-170.
3. Jones, *Jeff, and Andrew Adamatzky. "Computation of the travelling salesman problem by a shrinking blob."* Natural Computing 13.1 (2014): 1-16.
4. MacGregor, *James N., and Tom Ormerod. "Human performance on the traveling salesman problem."* Perception and Psychophysics 58.4 (1996): 527-539.
5. MacGregor, James N., Chu, Yun, *"Human performance on the traveling salesman and related problems: A review"*, Journal of Problem Solving, 3.2 (2011).
6. Applegate, D. L.; Bixby, R. M.; Chvátal, V.; Cook, W. J. (2006), The Traveling Salesman Problem, ISBN 978-0-691-12993-8.
7. Allender, Eric; Bürgisser, Peter; Kjeldgaard-Pedersen, Johan; Mitersen, Peter Bro (2007), "*On the Complexity of Numerical Analysis*" (PDF), SIAM J. Comput., 38 (5): 1987–2006, CiteSeerX 10.1.1.167.5495, doi:10.1137/070697926.
8. Arora, *Sanjeev (1998), "Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems" (PDF), Journal of the ACM, 45 (5): 753–782, doi:10.1145/290179.290180, MR* 1668147, S2CID 3023351.

9. Beardwood, J.; Halton, J.H.; Hammersley, J.M. (1959), "*The Shortest Path Through Many Points*", Proceedings of the Cambridge Philosophical Society, 55 (4): 299–327, Bibcode:1959PCPS...55..299B, doi:10.1017/s0305004100034095.

10. Bellman, R. (1960), "Combinatorial *Processes and Dynamic* Programming", in Bellman, R.; Hall, M. Jr. (eds.), Combinatorial Analysis, *Proceedings of Symposia in Applied* Mathematics 10, American Mathematical Society, *pp.* 217–249.

11. Bellman, R. (1962), "Dynamic *Programming Treatment of the Travelling Salesman Problem*", J. Assoc. Comput. Mach., 9: 61–63, doi:10.1145/321105.321111, S2CID 15649582.

12. Berman, Piotr; Karpinski, Marek (2006), "*8/7-approximation algorithm for* (1,2)-TSP", Proc. 17th ACM-SIAM Symposium on Discrete *Algorithms* (SODA '06), *pp.* 641–648, CiteSeerX 10.1.1.430.2224, doi:10.1145/1109557.1109627, ISBN 978-0898716054, S2CID 9054176, ECCC TR05-069.

13. Christofides, N. (1976), Worst-*case analysis of a new heuristic for the travelling salesman problem*, Technical Report 388, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh.

14. Hassin, R.; Rubinstein, S. (2000), "Better *approximations for max* TSP", *Information Processing* Letters, 75 (4): 181–186, CiteSeerX 10.1.1.35.7209, doi:10.1016/S0020-0190(00)00097-1.

15. Held, M.; Karp, R. M. (1962), "A Dynamic Programming Approach to *Sequencing Problems*", Journal of the Society for Industrial and Applied Mathematics, 10 (1): 196–210, doi:10.1137/0110015, hdl:10338.dmlcz/103900.

16. Kaplan, H.; Lewenstein, L.; Shafrir, N.; Sviridenko, M. (2004), "Approximation *Algorithms for Asymmetric* TSP *by Decomposing* Directed Regular Multigraphs", In Proc. 44th IEEE Symp. on Foundations of Comput. Sci, *pp.* 56–65.

17. Karpinski, M.; Lampis, M.; Schmied, R. (2015), "New *Inapproximability bounds for* TSP", Journal of Computer and System Sciences,81 (8): 1665–1677, arXiv:1303.6437, doi:10.1016/j.jcss.2015.06.003

18. Kosaraju, S. R.; Park, J. K.; Stein, C. (1994), "Long *tours and short superstrings*'", Proc. 35th Ann. IEEE Symp. on Foundations of Comput. Sci, IEEE Computer Society, *pp.* 166–177.

19. Orponen, P.; Mannila, H. (1987), "On *approximation preserving reductions: Complete problems and robust measures*'", Technical Report C-1987–28, Department of Computer Science, University of Helsinki.

20. Padberg, M.; Rinaldi, G. (1991), "A Branch-and-Cut Algorithm for the Resolution of Large-Scale Symmetric Traveling Salesman Problems", SIAM Review, 33: 60–100, doi:10.1137/1033004, S2CID 18516138.

21. Papadimitriou, Christos H. (1977), "The Euclidean traveling salesman problem is NP-complete", Theoretical Computer Science, 4 (3): 237–244, doi:10.1016/0304-3975(77)90012-3, MR 0455550.

22. Papadimitriou, C. H.; Yannakakis, M. (1993), "The traveling salesman problem with distances one and two", Math. Oper. Res., 18: 1–11, doi:10.1287/moor.18.1.1.

23.Schrijver, Alexander (2005). "On the history of combinatorial optimization (till 1960)". In K. Aardal; G.L. Nemhauser; R. Weismantel (eds.). Handbook of Discrete Optimization (PDF). Amsterdam: Elsevier. pp. 1–68.

24.Serdyukov, A. I. (1984), "An algorithm with an estimate for the traveling salesman problem of the maximum'", Upravlyaemye Sistemy, 25: 80–86.

25.Steinerberger, Stefan (2015), "New Bounds for the Traveling Salesman Constant", Advances in Applied Probability, 47 (1): 27–36, arXiv:1311.6338, Bibcode:2013arXiv1311.6338S, doi:10.1239/aap/1427814579, S2CID 119293287.