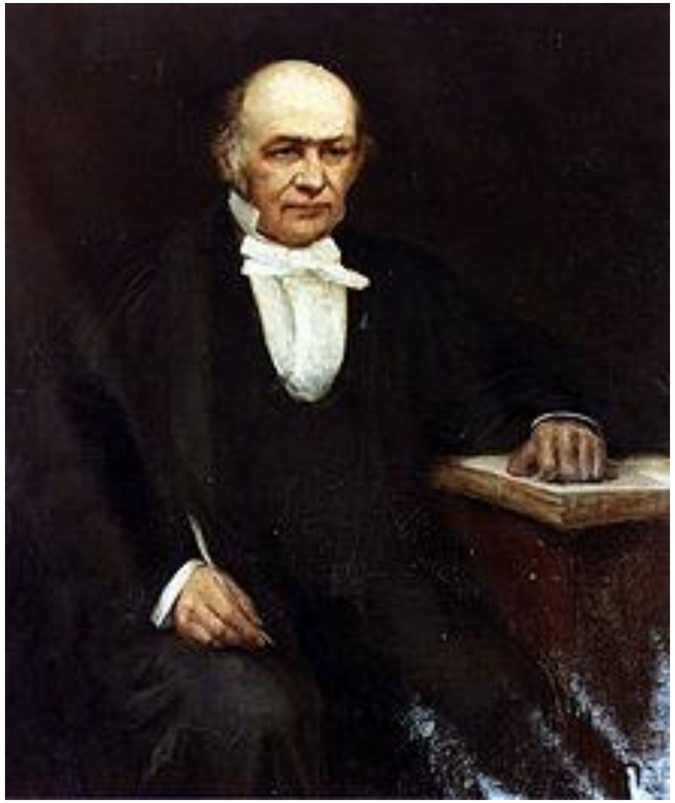
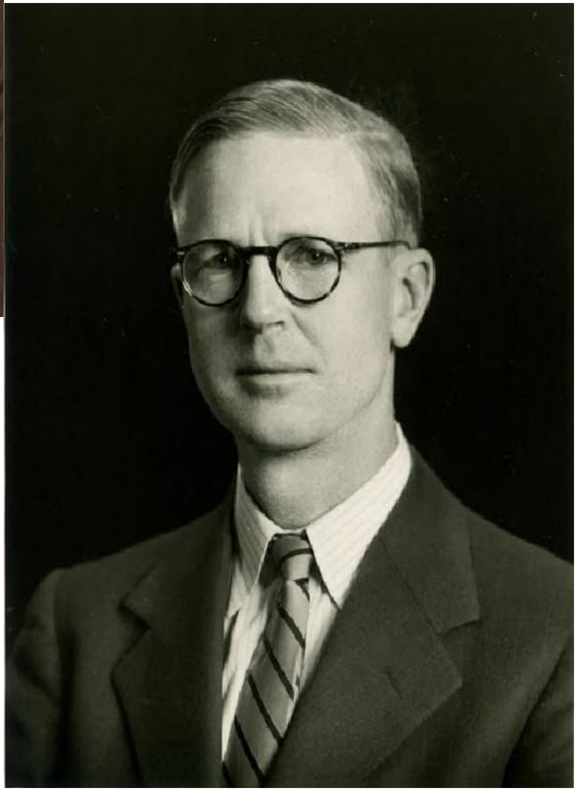
THE TRAVELLING SALESMAN PROBLEM

(TSP)

The origins of the travelling salesman problem are unclear

- 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





It was first considered mathematically in the 1930s by Merrill Flood who was looking to solve a school bus routing problem. Hassler Whitney at Princeton University introduced the name travelling salesman problem soon after.

History

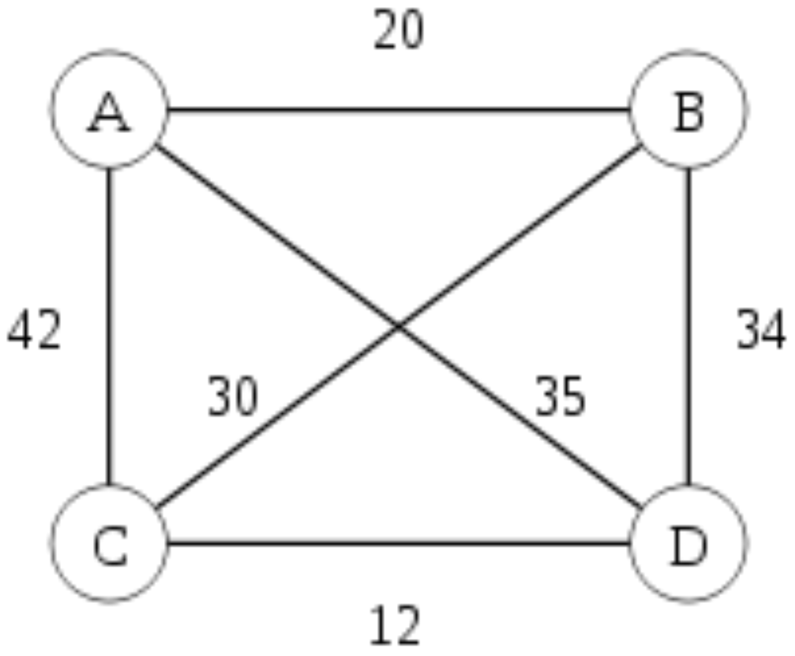
- In the 1950s and 1960s, the problem became increasingly popular in scientific circles in Europe and the USA after the RAND Corporation in Santa Monica, offered prizes for steps in solving the problem.
- Notable contributions were made by George Dantzig, Delbert Ray Fulkerson and Selmer M. Johnson from 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.

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 length. 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.

Asymmetric and symmetric

- In the symmetric TSP, the distance between two cities is the same in each opposite direction, forming an undirected graph. This symmetry halves the number of possible solutions. In the asymmetric TSP, paths may not exist in both directions or the distances might be different, forming a directed graph.
- Traffic collisions, one-way streets, and airfares for cities with different departure and arrival fees are examples of how this symmetry could break down.



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 Hamiltonian path problem.
- Another related problem is the bottleneck travelling salesman problem (bottleneck TSP): Find a Hamiltonian cycle in a weighted graph with the minimal weight of the weightiest edge. The problem is of considerable practical importance, apart from evident transportation and logistics areas.
- 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".
- The sequential ordering problem deals with the problem of visiting a set of cities where precedence relations between the cities exist.
- 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.

Integer linear programming formulation

- TSP can be formulated as an integer linear program. Label the cities with the numbers 0, …, 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 = 0, …, n, let u_i be an artificial variable, and finally take c_{ij} to be the distance from city i to city j. Then TSP can be written as the following integer linear programming problem:

$$\begin{array}{ll}
\min \sum_{i=0}^{n} \sum_{j \neq i, j=0}^{n} c_{ij} x_{ij} & \\
0 \leq x_{ij} \leq 1 & i, j = 0, \cdots, n \\
u_i \in \mathbf{Z} & i = 0, \cdots, n \\
\sum_{i=0, i \neq j}^{n} x_{ij} = 1 & j = 0, \cdots, n \\
\sum_{j=0, j \neq i}^{n} x_{ij} = 1 & i = 0, \cdots, n \\
u_i - u_j + n x_{ij} \leq n - 1 & 1 \leq i \neq j \leq n
\end{array}$$

Computing a solution

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

- Devising algorithms for finding exact solutions (they will work reasonably fast only for small problem sizes).
- Devising "suboptimal" or heuristic algorithms, i.e., algorithms that deliver either seemingly or probably good solutions, but which could not be proved to be optimal.
- Finding special cases for the problem ("subproblems") for which either better or exact heuristics are possible.

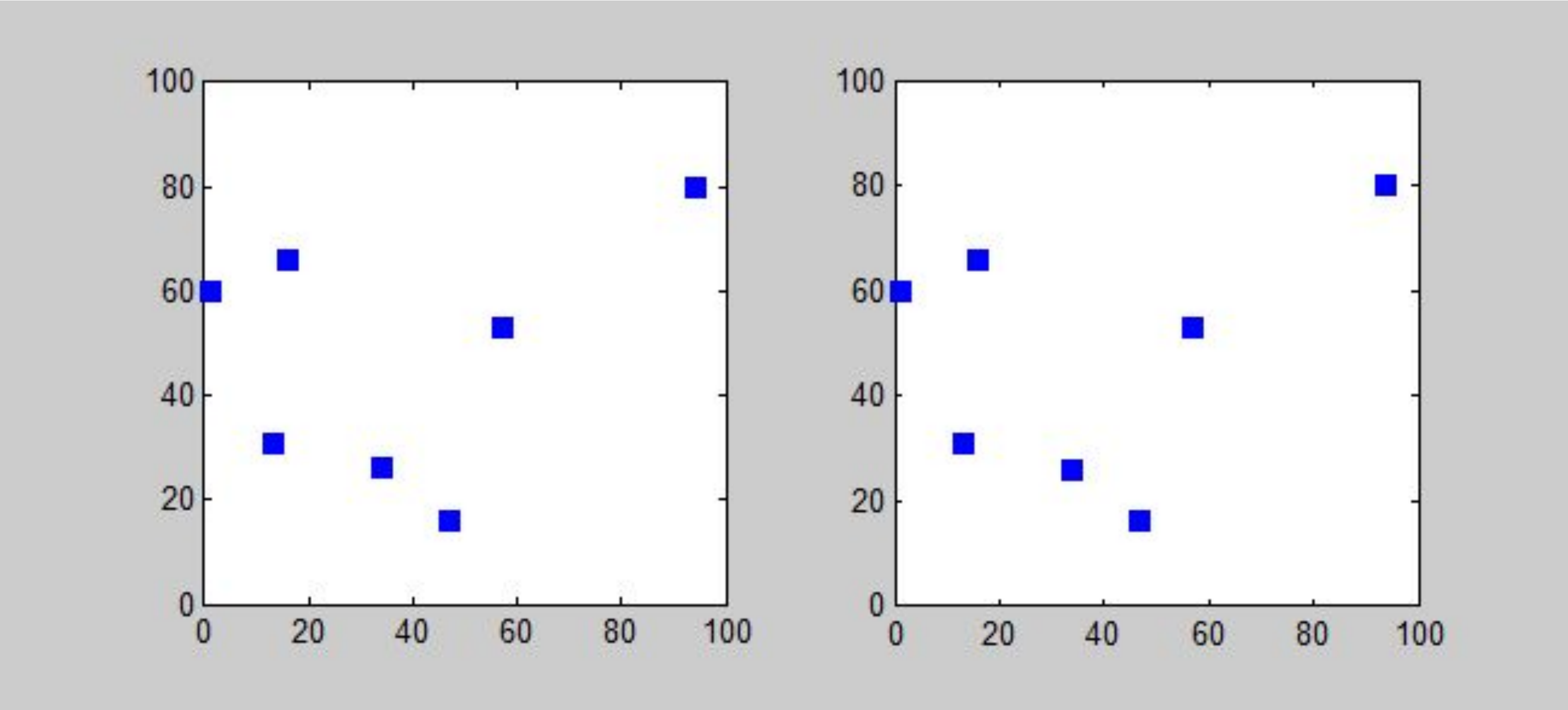
Exact algorithms

- The most direct solution would be to try all permutations (ordered combinations) and see which one is cheapest (using brute force search).
- For example, it has not been determined whether an exact algorithm for TSP that runs in time $O(1.9999^n)$ exists.

Other approaches include:

- Various branch-and-bound algorithms, which can be used to process TSPs containing 40–60 cities.
- Progressive improvement algorithms which use techniques reminiscent of linear programming. Works well for up to 200 cities.
- Implementations of branch-and-bound and problem-specific cut generation (branch-and-cut[15]); this is the method of choice for solving large instances. This approach holds the current record, solving an instance with 85,900 cities, see Applegate et al.

Exact algorithms

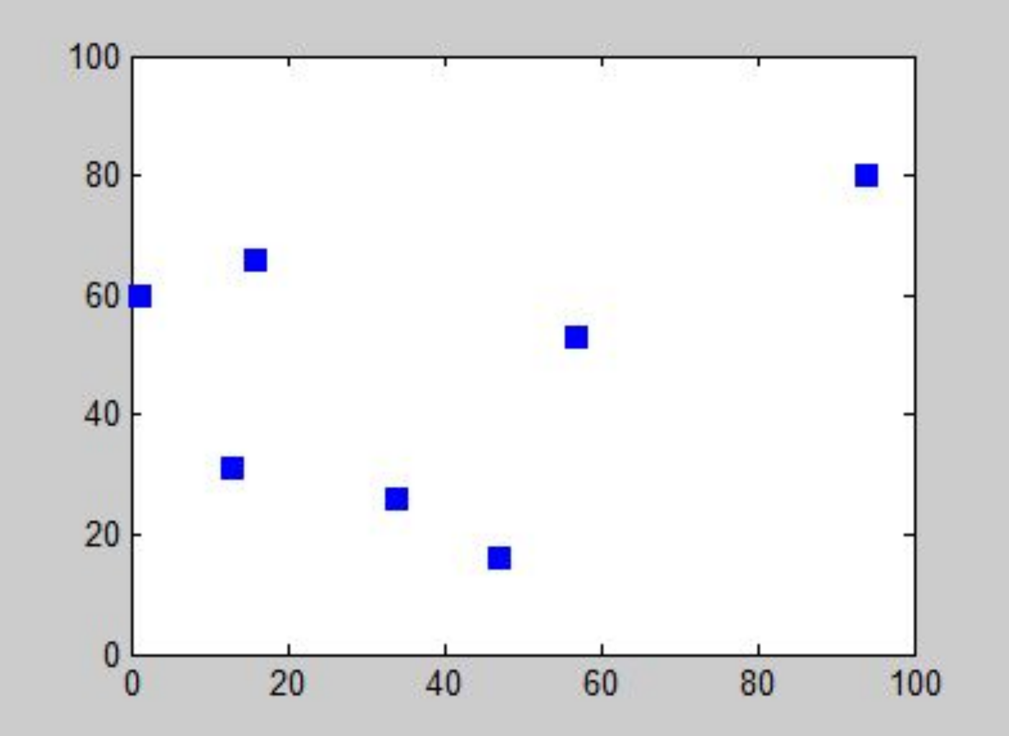


Heuristic and approximation algorithms

- Various heuristics and approximation algorithms, which quickly yield good solutions have been devised. Modern methods can find solutions for extremely large problems (millions of cities) within a reasonable time which are with a high probability just 2–3% away from the optimal solution.
- Several categories of heuristics are recognized.

Constructive heuristics

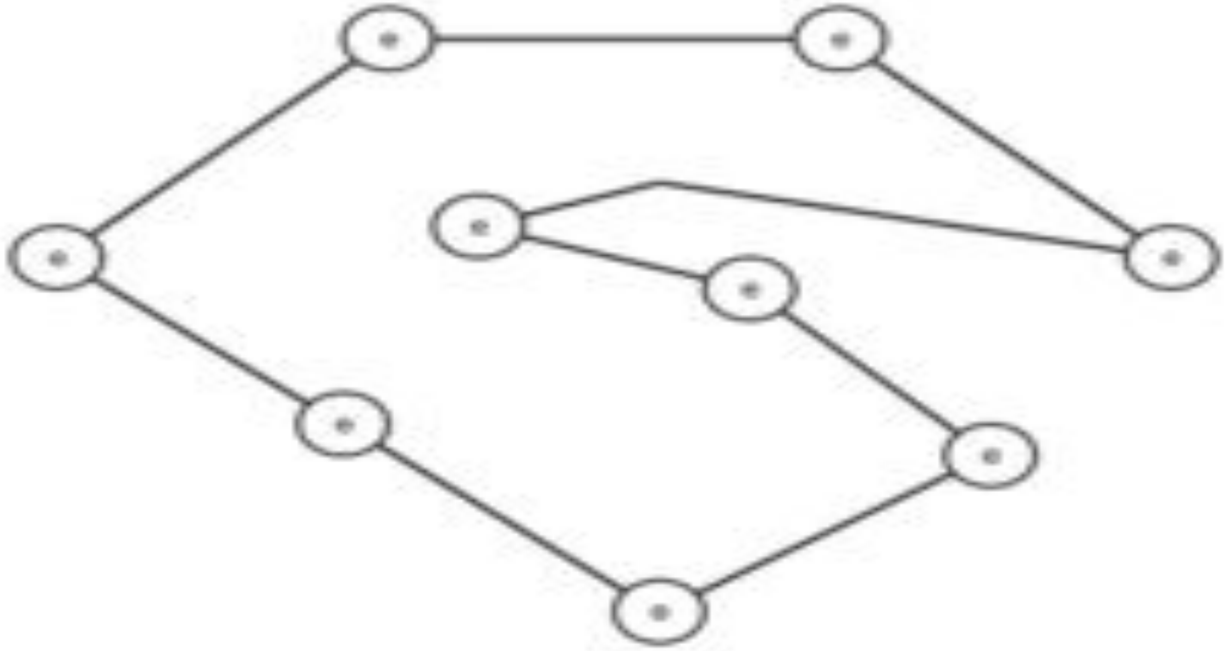
- The nearest neighbor (NN) algorithm (or so-called greedy algorithm) lets the salesman choose the nearest unvisited city as his next move. This algorithm quickly yields an effectively short route. For N cities randomly distributed on a plane, the algorithm on average yields a path 25% longer than the shortest possible path.[17] However, there exist many specially arranged city distributions which make the NN algorithm give the worst route (Gutin, Yeo, and Zverovich, 2002). This is true for both asymmetric and symmetric TSPs (Gutin and Yeo, 2007).
- The bitonic tour of a set of points is the minimum-perimeter monotone polygon that has the points as its vertices; it can be computed efficiently by dynamic programming.
- Another constructive heuristic, Match Twice and Stitch (MTS) (Kahng, Reda 2004 [19]), performs two sequential matchings, where the second matching is executed after deleting all the edges of the first matching, to yield a set of cycles. The cycles are then stitched to produce the final tour.



Christofides' algorithm for the TSP

- The Christofides algorithm follows a similar outline but combines the minimum spanning tree with a solution of another problem, minimum-weight perfect matching. This gives a TSP tour which is at most 1.5 times the optimal. The Christofides algorithm was one of the first approximation algorithms, and was in part responsible for drawing attention to approximation algorithms as a practical approach to intractable problems. As a matter of fact, the term "algorithm" was not commonly extended to approximation algorithms until later; the Christofides algorithm was initially referred to as the Christofides heuristic.
- This algorithm looks at things differently by using a result from graph theory which helps improve on the LB of the TSP which originated from doubling the cost of the minimum spanning tree. Given an Eulerian graph we can find an Eulerian tour in O(n) time.[5] So if we had an Eulerian graph with cities from a TSP as vertices then we can easily see that we could use such a method for finding an Eulerian tour to find a TSP solution. By triangular inequality we know that the TSP tour can be no longer than the Eulerian tour and as such we have a LB for the TSP. Such a method is described below.

1. Find a minimum spanning tree for the problem
2. Create duplicates for every edge to create an Eulerian graph
3. Find an Eulerian tour for this graph
4. Convert to TSP: if a city is visited twice, create a shortcut from the city before this in the tour to the one after this.



- Using a shortcut heuristic on the graph created by the matching below

Iterative improvement

- Pairwise exchange
- The pairwise exchange or 2-opt technique involves iteratively removing two edges and replacing these with two different edges that reconnect the fragments created by edge removal into a new and shorter tour. This is a special case of the k-opt method. Note that the label Lin–Kernighan is an often heard misnomer for 2-opt. Lin–Kernighan is actually the more general k-opt method.
- For Euclidean instances, 2-opt heuristics give on average solutions that are about 5% better than Christofides' algorithm.

k-opt heuristic, or Lin–Kernighan heuristics

- Take a given tour and delete k mutually disjoint edges. Reassemble the remaining fragments into a tour, leaving no disjoint subtours (that is, don't connect a fragment's endpoints together). This in effect simplifies the TSP under consideration into a much simpler problem.

***V*-opt heuristic**

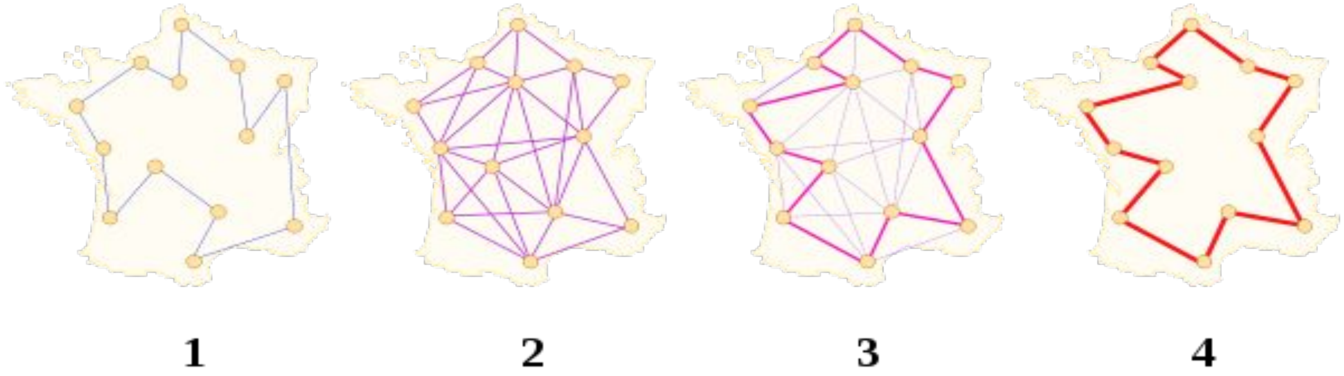
- The variable-opt method is related to, and a generalization of the k-opt method. Whereas the k-opt methods remove a fixed number (k) of edges from the original tour, the variable-opt methods do not fix the size of the edge set to remove. Instead they grow the set as the search process continues. The best known method in this family is the Lin–Kernighan method (mentioned above as a misnomer for 2-opt).

Randomised improvement

- Optimized Markov chain algorithms which use local searching heuristic sub-algorithms can find a route extremely close to the optimal route for 700 to 800 cities.

- TSP is a touchstone for many general heuristics devised for combinatorial optimization such as genetic algorithms, simulated annealing, Tabu search, ant colony optimization, river formation dynamics (see swarm intelligence) and the cross entropy method.

Ant colony optimization



- Artificial intelligence researcher Marco Dorigo described in 1993 a method of heuristically generating "good solutions" to the TSP using a simulation of an ant colony called ACS (Ant Colony System).[21] It models behavior observed in real ants to find short paths between food sources and their nest, an emergent behaviour resulting from each ant's preference to follow trail pheromones deposited by other ants.

- ACS sends out a large number of virtual ant agents to explore many possible routes on the map. Each ant probabilistically chooses the next city to visit based on a heuristic combining the distance to the city and the amount of virtual pheromone deposited on the edge to the city. The ants explore, depositing pheromone on each edge that they cross, until they have all completed a tour. At this point the ant which completed the shortest tour deposits virtual pheromone along its complete tour route (global trail updating). The amount of pheromone deposited is inversely proportional to the tour length: the shorter the tour, the more it deposits.

Special cases of the TSP

Metric TSP

- In the metric TSP, also known as delta-TSP or Δ-TSP, the intercity distances satisfy the triangle inequality.
- A very natural restriction of the TSP is to require that the distances between cities form a metric to satisfy the triangle inequality; that is the direct connection from A to B is never farther than the route via intermediate C: $d_{AB} \leq d_{AC} + d_{CB}$

Euclidean TSP

Like the general TSP, the Euclidean TSP is NP-hard. With discretized metric (distances rounded up to an integer), the problem is NP-complete. For example, the minimum spanning trinstance ee of the graph associated with an of the Euclidean TSP is a Euclidean minimum spanning tree, and so can be computed in expected O (n log n) time for n points (considerably less than the number of edges).

Asymmetric TSP

- In most cases, the distance between two nodes in the TSP network is the same in both directions. The case where the distance from *A* to *B* is not equal to the distance from *B* to*A* is called asymmetric TSP. A practical application of an asymmetric TSP is route optimisation using street-level routing (which is made asymmetric by one-way streets, slip-roads, motorways, etc.).

Solving by conversion to symmetric TSP

- Solving an asymmetric TSP graph can be somewhat complex. The following is a 3×3 matrix containing all possible path weights between the nodes *A*, *B* and *C*. One option is to turn an asymmetric matrix of size *N* into a symmetric matrix of size 2*N*

Analyst's travelling salesman problem

- There is an analogous problem in geometric measure theory which asks the following: under what conditions may a subset E of Euclidean space be contained in a rectifiable curve (that is, when is there a curve with finite length that visits every point in E)? This problem is known as the analyst's travelling salesman problem:

I. **TSP path length for random sets of points in a square**

II. **Upper bound**

III. **Lower bound**

Computational complexity

- The problem has been shown to be NP-hard (more precisely, it is complete for the complexity class FPNP; see function problem), and the decision problem version ("given the costs and a number x, decide whether there is a round-trip route cheaper than x") is NP-complete. The bottleneck travelling salesman problem is also NP-hard.
- The problem remains NP-hard even for the case when the cities are in the plane with Euclidean distances, as well as in a number of other restrictive cases. Removing the condition of visiting each city "only once" does not remove the NP-hardness, since it is easily seen that in the planar case there is an optimal tour that visits each city only once (otherwise, by the triangle inequality, a shortcut that skips a repeated visit would not increase the tour length).

Complexity of approximation

- In the general case, finding a shortest travelling salesman tour is NPO-complete. If the distance measure is a metric and symmetric, the problem becomes APX-complete and Christofides's algorithm approximates it within 1.5.
- If the distances are restricted to 1 and 2 (but still are a metric) the approximation ratio becomes 8/7. In the asymmetric, metric case, only logarithmic performance guarantees are known, the best current algorithm achieves performance ratio 0.814 log(n);it is an open question if a constant factor approximation exists.

Thank you