Abstract

The Traveling Salesman Problem (TSP) is a classic NP-hard combinatorial optimization problem. For metric instances (symmetric distances satisfying the triangle inequality), Christofides' algorithm provides a polynomial-time approximation that guarantees a solution within a factor of 3/2 of the optimum. This report presents a detailed description of the algorithm, a proof of its approximation ratio, an analysis of its time complexity, and a complete C++ implementation.

1 Introduction

The Traveling Salesman Problem (TSP) asks for a shortest closed tour that visits each vertex of a given complete graph exactly once. In the *metric* TSP, the edge weights satisfy the triangle inequality: $w(u,v) \leq w(u,x) + w(x,v)$ for all vertices u, v, x. This problem remains NP-hard, but approximation algorithms are possible. Christofides' algorithm, published in 1976, is a deterministic polynomial-time algorithm that produces a tour whose length is at most 3/2 times the optimal tour length. It is based on constructing a minimum spanning tree, adding a minimum-weight perfect matching on its odd-degree vertices, and then shortcutting an Eulerian circuit of the resulting multigraph.

2 Algorithm Description

Let $G = (V, E, w)$ be a complete metric graph with n vertices. Christofides' algorithm proceeds as follows:

Algorithm 1 Christofides' Algorithm for Metric TSP

1: Compute a minimum spanning tree T of G.
2: Let $O \subseteq V$ be the set of vertices of odd degree in T.
3: Find a minimum-weight perfect matching M in the subgraph induced by O.
4: Form the multigraph $H = T \cup M$.
5: Find an Eulerian circuit C of H.
6: Obtain a Hamiltonian cycle by shortcutting C (remove repeated vertices while preserving connectivity).
7: **return** the resulting tour.

Steps 1 and 3 are the algorithm's core. The minimum spanning tree can be found by Prim's or Kruskal's algorithm. The minimum-weight perfect matching in a general graph can be found in polynomial time by Edmonds' blossom algorithm. The Eulerian circuit exists because every vertex in H has even degree, and it can be constructed using Hierholzer's algorithm. Shortcutting does not increase the total weight because of the triangle inequality.

3 Approximation Ratio Proof

Let C^* be an optimal TSP tour, and let $w(C^*)$ denote its total weight. Let T be the minimum spanning tree, M the minimum-weight perfect matching on the odd-degree vertices of T, and

$H = T \cup M$. The algorithm's output tour is denoted by C_{out}.

1. **Spanning tree bound:** Removing any edge from C^* yields a spanning tree. Hence
$$w(T) \leq w(C^*).$$

2. **Matching bound:** Consider the odd-degree vertices O in cyclic order around C^*. Pair them along the tour in two different ways: the "odd" matching and the "even" matching. Both are perfect matchings on O, and their total weight is at most $w(C^*)$ because each edge of the tour appears in at most one of the matchings (thanks to the triangle inequality). Therefore the cheaper of the two matchings has weight at most $w(C^*)/2$. Since M is a minimum-weight perfect matching on O,
$$w(M) \leq \frac{w(C^*)}{2}.$$

3. **Combined bound:** The Eulerian circuit of H has weight
$$w(T) + w(M) \leq w(C^*) + \frac{w(C^*)}{2} = \frac{3}{2}w(C^*).$$

4. **Shortcutting:** Because the triangle inequality holds, replacing a path $u \to x \to v$ by the direct edge $u \to v$ does not increase the weight. Consequently, shortcutting the Eulerian circuit to a Hamiltonian cycle does not increase the total weight. Thus
$$w(C_{\text{out}}) \leq w(T) + w(M) \leq \frac{3}{2}w(C^*).$$

Hence Christofides' algorithm is a 3/2-approximation algorithm for the metric TSP.

4 Time Complexity Analysis

Let $n = |V|$.

- **Minimum spanning tree:** Using Prim's algorithm with an adjacency matrix requires $O(n^2)$ time. With a binary heap and adjacency list, the time can be reduced to $O((n + |E|)\log n) = O(n^2 \log n)$ for a complete graph.

- **Finding odd-degree vertices:** Scanning the tree degrees takes $O(n)$ time.

- **Minimum-weight perfect matching:** The blossom algorithm (Edmonds' algorithm) runs in $O(n^3)$ time. This step dominates the overall complexity.

- **Eulerian circuit:** Hierholzer's algorithm runs in $O(|E|)$ time; the multigraph H has $O(n)$ edges, so this step is $O(n)$.

- **Shortcutting:** Removing repeated vertices from the Eulerian circuit can be done in $O(n)$ time using a visited array.

Thus the overall worst-case time complexity of Christofides' algorithm is $O(n^3)$. If a greedy heuristic is used for the matching step instead of the exact blossom algorithm, the complexity reduces to $O(n^2)$, but the 3/2 approximation guarantee is lost.

5 Implementation Details

The provided C++ implementation follows the algorithmic steps closely. It uses:

- A distance matrix `vector<vector<double>> distance` to store metric weights.
- Prim's algorithm to compute the minimum spanning tree, stored as an adjacency list `mstAdj`.
- A greedy heuristic for perfect matching: repeatedly pairing the closest unmatched odd-degree vertices.
- Hierholzer's algorithm to construct an Eulerian circuit from the multigraph.
- Shortcutting by marking visited vertices while traversing the Eulerian circuit.

The code is self-contained and does not rely on external libraries. For a guaranteed 3/2 approximation, the matching step should be replaced by an exact minimum-weight perfect matching algorithm (e.g., Edmonds' blossom algorithm).

6 Conclusion

Christofides' algorithm is a landmark result in approximation algorithms for NP-hard problems. It provides a simple, deterministic 3/2-approximation for the metric TSP in polynomial time. The algorithm elegantly combines graph-theoretic concepts: minimum spanning trees, perfect matchings, Eulerian circuits, and the triangle inequality. While more recent algorithms have slightly improved the approximation ratio for general metric spaces, Christofides' method remains a classic and widely taught technique. The accompanying C++ implementation illustrates the practical steps of the algorithm, though for large instances a library implementation of the blossom algorithm is recommended for the matching step.