A Survey on Traveling Salesman Problem

Ruichen Luo

June 30, 2021

Abstract

This survey is on the traveling salesman problem (TSP). We extensively study its NP-completeness, several heuristics and approximation algorithms, and also a couple of TSP variant problems. Some important results are implemented, and the source code can be found under './src' directory.

1 Introduction

The traveling salesman problem (TSP) is a corner stone and difficult problem in graph theory and algorithms. The typical TSP problem can be formulated simply as follows:

Problem 1.1 (TSP Problem)**.** *Let $G(V, E)$ be a **fully-connected** graph with positive weights $w : E \to \mathbb{R}^+$ on its edges. Unless otherwise mentioned, we assume that the weights satisfy the **metric** properties (or triangle inequalities), i.e. $w(s,t) + w(t,u) \geq w(s,u)$. Try to find a **Hamiltonian cycle** $\mathcal{H}$, with a minimal sum of weights on its edges:*

$$\min \sum_{e \in \mathcal{H}} w(e).$$

The importance of TSP not only lies in its simple abstraction of real-world problem, but also because it is a representative of a large class of difficult problems in algorithms [6]. It lies in the NP-complete class, which has not yet been found, or may even not exist a polynomial-time algorithm.

Numerous searching heuristics and approximation algorithms are proposed. One practically useful heuristic is multi-fragment (MF) algorithm, a greedy-based method. Many tricks are applied further, some of which claims to be today's best method and reache $2\% - 3\%$ away from the optimal solution in practical cases with high probability [10]. Compared to these heuristics, an exact approximation ratio is hard to determine. In 1976, Christofides proposes a simple and elegant algorithm [1], which yields a $3/2$-approximation ratio. For several decades, this bound had not yet been proven tight, and also had not yet been advanced. Until recently, [5] proposes a notable method, which is modified

from Christofides's algorithm and is based on randomly chosen trees. It claims to reach $1.5 - 10^{-36}$ approximation ratio, but has not yet been validified by the research field up to date.

This article is orgainized in following way:

In section 2.1, we show that this problem is NP-complete. Then we show that the metric properties are necessary to ensure that constant-ratio polynomial-time approximation algorithm can exist.

In section 2.2, we surveys on several heuristics. Among the bunches of methods, we study the well-known MF algorithm in detail.

In section 2.3, we survey on approximation algorithms. One well-known algorithm, Christofides algorithm, is studied in detail. Notably, a latest result claims to slightly increase this approximation ratio. We also study this work abstractly.

In section 3, we implement some of the important methods, and compare their results and performances.

Finally, we give a conclusion in section 4.

2 Key Results

2.1 TSP's Hardness

The TSP's NP-completeness follows from the NP-completeness of the Hamiltonian cycle decision problem, which is fomulated below. The NP-completeness of the problem is first proven by Garey et al. in [3], we hereby use it directly as theorem.

Problem 2.1 (Hamiltonian Cycle Decision Problem)**.** *Let $G(V, E)$ be a graph. Decide whether there exists a Hamiltonian cycle $\mathcal{H}$ in G.*

Theorem 2.1. *The problem 2.1 is NP-complete. [3]*

Then it yields the NP-compleness of TSP problem by reduction.

Corollary 2.1. *The problem 1.1 is NP-complete.*

The above corollary can be viewed as a special case of $r = 1$ in corollary 2.2. We also show that the metric properties in TSP problem is necessary, since without this constraint, unless $P = NP$, the TSP problem may not have a polynomial-time algorithm with constant approximation ratio:

Corollary 2.2. *In TSP problem without metric properties, unless $P = NP$, there cannot be a polynomial-time r-approximation algorithm for some constant $r \geq 1$.*

Proof. For the Hamiltonian cycle decision problem on graph $G(V, E)$, we construct a fully-connected graph $G'(V, E')$ and define weights on its edges:

$$w(e) = \begin{cases} 1, & e \in E \\ r|V| + 1, & e \in E' - E \end{cases}$$

Then if graph G has a Hamiltonian cycle $\mathcal{H}$, the length of the TSP tour tracing $\mathcal{H}$ in G' equals $|V|$. Otherwise, the TSP tour in G' will be at least $r|V|+1$, since it must contain at least one edge in $E'-E$. Hence, whether G has a Hamiltonian cycle is equivalent to whether the minimum length of the TSP tour in G' can reach $|V|$, and also equivalent to finding an r-approximation solution to the problem.

Thus finding r-approximation solution to problem 1.1 without metric properties is at least at hard as problem 2.1. Then according to theorem 2.1, finding r-approximation solution to problem 1.1 without metric properties is also NP-complete. □

2.2 Heuristics

The multi-fragment (MF) method [4] is a simple greedy-based method. Besides this, we also investigate two groups of well-known TSP heuristics [10]. They belongs to Lin–Kernighan (LK) methods [8] and stem-and-cycle (S&C) ejection chain methods [9] respectively.

2.2.1 MF Method

The MF method is a greedy heuristic. It starts from an empty graph and iteratively collects the cheapest edges. During each iteration, the edge with minimum cost is collected, which either connectes two existing tour fragments between their endpoints, or completes the TSP round-trip as the last edge.

A psuedocode is given in algorithm 1.

Algorithm 1 MF algorithm

Require: An input graph $G(V, E)$ with weights w on its edges.
Ensure: A TSP tour T in G (may or may not be the minimum).
  Initialize tour fragments disjoint set: $(v_i),\ i = 1, \ldots, |V|$.
  for i from 1 to $|V|$ **do**
    if $i < |V|$ **then**
      Find the two tour fragments $(s_1, \ldots, s_i)$, $(t_1, \ldots, t_j)$ with closest endpoints $d(s_i, t_1)$.
      Merge the two tour framents into $(s_1, \ldots, s_i, t_1, \ldots, t_j)$.
    else
      Close the tour fragment $T = (v_1, \ldots, v_{|V|}, v_1)$.
      return T
    end if
  end for

2.2.2 LK Methods

A more sophisticated method is Lin–Kernighan heuristic and has long been considered as the state-of-the-art searching heuristics in TSP problem. Since

first proposed, there are numerous attempts to modify and optimize it, but almost none of these optimization methods are formally proven to achieve a better result. Hereby, we only investigate the classic method proposed by [8].

In brief, the LK method generates a random initial solution T. And then adapt this solution by swapping its paths. It adaptively decides the number of paths to be swapped in each step.

The psuedocode is given in algorithm 2.

Algorithm 2 LK heuristics

Require: An input graph $G(V, E)$ with weights w on its edges.
Ensure: A TSP tour T in G (may or may not be the minimum).
  Generate a random initial solution T.
  while true **do**
    for i in $1, 2, \cdots$ **do**
      Select $u_i \in T - \{u_1, \ldots, u_{i-1}\}$ and $v_i \in T - \{v_1, \ldots, v_{i-1}\}$ that maximizes their distance d.
      if d satisfies the stopping criterion **then**
        break
      end if
      $i = i + 1$
    end for
    if there is an improvement **then**
      Suppose the best improvement is found for $i = k$, update T by exchanging $u_1, \ldots, u_k$ with $v_1, \ldots, v_k$.
    else
      break
    end if
  end while
  return T

2.2.3 S&C Ejection Chain Methods

Similar to LK methods, S&C ejection chain methods are also based on alternating paths. However, it considers a broader class of alternating paths in each step. Therefore, there is a performance-complexity tradeoff between these two methods. And also, there are theoretical analysis on the differences and similarities of the two methods, and try to find some midpoints between these two methods [2].

The psuedocode is given in algorithm 3.

2.3 MST-Based Approximation Algorithm

Though the local searching heuristics surveyed in section 2.2 are fast to implement and can generate decent results with high probability in many practical

Algorithm 3 S&C ejection chain heuristics

Require: An input graph $G(V, E)$ with weights w on its edges.
Ensure: A TSP tour T in G (may or may not be the minimum).
Generate a random initial solution T.
$T^* = T$
Let s be a node in T.
$k^* = s$
repeat
$d = 0$
while there are valid moves **do**
$i = k^*$
$d = d + 1$
Find the best valid move that maintains the reference structure, namely the structure given by (i, j^*), (j^*, k^*) for which the gain $w(j, k) - w(i, j)$.
Perform this move and get a new T, say $T(d)$ at search depth d. Denote the optimal associated solution as $T^*(d)$.
Record the (i, j^*) so that it cannot be further placed off, and the (j^*, k^*) so that it cannot be further placed on.
end while
Denote d^* as the search depth at which the best solution $T^*(d^*)$ is found.
if $d^* > 0$ **then**
$T^* = T^*(d^*), T = T^*$
end if
until $d^* = 0$
return T^*

cases, the theoretical bounds are still not clear. In this section, we survey several approximation algorithms. They are all constructed from minimal spanning tree (MST), and their approximation ratios are clear and well established.

First we give a baseline 2-approximation algorithm. Then we survey on the well-known Christofides algorithm proposed by Nicos Christofides et al. in 1976, which achieves 3/2-approximation ratio and remains the state-of-the-art method till now. Finally, we briefly investigate a slightly improved version which claims to reach $3/2 - 10^{-36}$ approximation ratio but has not yet been validified.

2.3.1 Baseline Alorithm and Christofides Algorithm

We first give a simple, baseline, 2-approximation algorithm in algorithm 4. The time complexity is $O(E + V \log V)$, which is due to the MST step.

Algorithm 4 Baseline 2-approxiamtion algorithm

Require: An input graph $G(V, E)$ with weights w on its edges.
Ensure: A TSP tour T in G ($w(T) \leq 2w(T^*)$).
Find a MST M of the graph G.
Trace the Depth-First-Search (DFS) path in M while skipping the visited nodes. Record the track T.
return T

Then we prove its approximation ratio:

Theorem 2.2. *The algorithm 4 is a 2-approximation algorithm.*

Proof. Due to the metric properties, the length of the DFS-track T is at most twice of the weights of the MST M. Moreover, any TSP round-trip must include a spanning tree, say M'. Then we have:

$$w(T) \leq 2w(M) \leq 2w(M') < 2w(T^*).$$

□

Now we introduce the Christofides algorithm [1] (algorithm 5):

Algorithm 5 Christofides 3/2-approxiamtion algorithm

Require: An input graph $G(V, E)$ with weights w on its edges.
Ensure: A TSP tour T in G ($w(T) \leq \frac{3}{2}w(T^*)$).
Find a MST M of the graph G.
Identify all odd-degree nodes O in M.
Find a minimum weight perfect matching P in the subgraph induced by O.
Find a Eulerian curcuit of in $M + P$ while skipping the visited nodes. Record the track T.
return T

The time complexity of algorithm 5 is $O(E^3)$, which is due to the computation of minimum perfect matching.

Then we prove its approximation ratio:

Theorem 2.3. *The algorithm 5 is a* $3/2$*-approximation algorithm.*

Proof. Due to the metric properties, we have $w(T) \leq w(M) + w(P)$. Note that the minimum TSP tour, say T^*, can be decomposed into two perfect matchings interchangeably. Then $w(T^*) \geq 2w(P)$. Therefore,

$$w(T) \leq w(M) + w(P) \leq w(T^*) + \frac{1}{2}w(T^*) = \frac{3}{2}w(T^*).$$

□

2.3.2 A Slightly Improved Algorithm

In this section, we briefly introduce Karlin et al.'s works [5]. In 2020, they have proposed a novel approximation algorithm which claims to achieve approximation ratio slightly better than the state-of-the-art $3/2$ in algorithm 5.

Their methods are based on linear-programming relaxation:

$$\begin{aligned} \min & \sum_{s,t} x_{(s,t)} w(s,t) \\ \text{s.t.,} & \sum_{s} x_{(s,t)} = 2, \quad \forall t \in V, \\ & \sum_{s \in S, t \notin S} x_{(s,t)} \geq 2, \quad \forall S \subset V, \\ & x_{(s,t)} \geq 0 \quad \forall s, t \in V. \end{aligned} \tag{1}$$

Their algorithm is presented in algorithm 6.

Algorithm 6 Karlin's slightly better approximation algorithm

Require: An input graph $G(V, E)$ with weights w on its edges.
Ensure: A TSP tour T in G ($w(T) \leq \frac{3}{2}w(T^*)$).
Find an optimum solution x^0 of equation 1, and let $e_0 = (s_0, t_0)$ be an edge with $x^0_{e_0} = 1$, $w(e_0) = 0$.
Let $E_0 = E \cup \{e_0\}$ be the support of x^0 and x be x^0 restricted to E and $G = (V, E)$.
Find a vector $\lambda : E \to \mathbb{R}^{\geq 0}$ such that for any $e \in E$, $\mathbf{P}_{\mu_\lambda}[e] = x_e(1 \pm 2^{-n})$.
Sample a tree $T \sim \mu_\lambda$.
Let M be the minimum cost perfect matching on odd degree vertices of T.
return $T \cup M$.

Then they prove the following bound in theorem 2.4. Their proofs are too long to follow and have not yet been validified.

Theorem 2.4 (Theorem 1.1 in [5], not yet been validified)**.** *For some* $\varepsilon > 10^{-36}$*, the algorithm achieves a randomized* $3/2 - \varepsilon$ *approximation ratio.*

Table 1: Performance Comparisons

Inputs	Optimum	MF	Baseline	Christofides
1 (size 4)	7	7	7	7
2 (size 17)	2085	2189	2352	2190
3 (size 26)	937	988	1136	964
4 (size 42)	699	1003	916	773
5 (size 48)	33523	40186	43980	38442

3 Experimental Results

In the experiment section, we implement several important algorithms listed in section 2.2 and 2.3.

The most difficult part is the implementation of Christofides algorithm. It's actually far more tricky than it seems to be. The most challenging part lies at solving the minimum perfect matching, which is based on linear programming in its dual form. Our implementation is based on the well-known Blossom-V interfaces [7]. The complexity of this part as well as the whole algorithm, thus, is $O(|V|^3)$.

We acknowledge using the real-world datasets in TSPLIB [11] to test the performance and correctness of the implemented algorithms.

The results are shown in table 1.

In our experiment results, the 3/2-approximation Christofides algorithm always achieves better results than the 2-approximation baseline algorithm. Due to the randomness and uncertainty of local searches, the performance of MF algorithm is somehow unstable. It can sometimes even out-perfrom the Christofides algorithm (as in input 2), but it can also falls far behind (as in input 4).

As shown in our experiments, Christofides algorithm is the most stable algorithm and achieves the best performance. While MF and baseline algorithms are easier to implement, but have relatively moderate performances.

4 Conclusions

In this survey, we broadly investigate the TSP problem and useful heuristics and algorithms. Many heuristics methods are proposed and compared in section 2.2. MF method is a basic method, while LK method searches at a deeper depth, and S&C ejection chain method searches at a broader class of neighbors. The best known approximation ratio is 3/2 given by Christofides algorithm (algorithm 5). A slightly better version is recently proposed but has not yet been validified.

We also implement some of these methods and analyze their results accordingly in experiment section 3.

References

[1] Nicos Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group, 1976.

[2] Birger Funke, Tore Grünert, and Stefan Irnich. A note on single alternating cycle neighborhoods for the tsp. *Journal of Heuristics*, 11(2):135–146, 2005.

[3] Michael R Garey, David S. Johnson, and R Endre Tarjan. The planar hamiltonian circuit problem is np-complete. *SIAM Journal on Computing*, 5(4):704–714, 1976.

[4] David S Johnson and Local Optimization. the traveling salesman problem,". In *Proc. 17th Colloquium on Automata, Languages and Programming, Springer-Verlag*, pages 446–461, 1990.

[5] Anna R Karlin, Nathan Klein, and Shayan Oveis Gharan. A (slightly) improved approximation algorithm for metric tsp. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 32–45, 2021.

[6] Richard M Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.

[7] Vladimir Kolmogorov. Blossom v: a new implementation of a minimum cost perfect matching algorithm. *Mathematical Programming Computation*, 1(1):43–67, 2009.

[8] Shen Lin and Brian W Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Operations research*, 21(2):498–516, 1973.

[9] Erwin Pesch and Fred Glover. Tsp ejection chains. *Discrete Applied Mathematics*, 76(1-3):165–181, 1997.

[10] César Rego, Dorabela Gamboa, Fred Glover, and Colin Osterman. Traveling salesman problem heuristics: Leading methods, implementations and latest advances. *European Journal of Operational Research*, 211(3):427–441, 2011.

[11] Gerhard Reinelt. Tsplib—a traveling salesman problem library. *ORSA journal on computing*, 3(4):376–384, 1991.