


# Generating Instances of TSP with Known Optima and Near-Optima

Andrew Jackson   
(Dated: 2018-ish)

There are many instances where it is useful to have access to NP-hard problems with known solutions and near misses. The performance of algorithms to find slightly better solutions is of great importance in an age where many instances of the problem may be solved and the solution will be used many times, such as when calculating delivery routes. The naive way to obtain such problems, for testing purposes or otherwise, is to randomly choose the parameters of the problem and then solving the generated problem. This is inefficient. In this paper I develop an algorithm for selecting an optimal tour, a nearly-optimal tour, and the difference in their tour lengths, then building an instance of TSP with the specified properties.

## I. THE TRAVELING SALESMAN PROBLEM

### A. History, Recent Progress, and Our Contribution

The travelling salesman problem (TSP) began as a problem (circa 1832) of genuine consideration for actual travelling salesmen. However, in the decades since it was formulated, the travelling salesman problem has become primarily studied as a textbook example of an NP-hard [1] combinatorial optimisation problem [2, 3]. Despite its proven difficulty, research into this problem has continued, with approximations and greedy algorithms taking centre stage.

This work has featured negative results:

- Limits on the effectiveness of approximations [4–6]
- Greedy Algorithms can fail badly [7–9]

As well as positive results:

- Effective Heuristics and approximations [10–13]
- General advances in combinatorial optimisation solving [14]
- Estimations of bounds [15]

The ability of algorithms to find slightly better solutions than others is of great importance in an age where many instances of the problem may be solved and the solution will be used many times, such as when calculating delivery routes. This could save large amounts of time, energy, and money. In this paper, I extend the methods of Ref. [16], to be present a method of constructing problems similar to Ref. [16] but with the additional ability to specify the near optima and the spectral gap between them and the optima. It is hoped this will help in the development and testing of algorithms for solving large combinatorial optimisation problems, so that problems with the consequences described above may be better solved.

### B. Informal Formulation

The most natural form of the traveling salesman problem is its expression in plain English and is presented below:

*A salesman has a list of cities he must visit. He wants to visit each city exactly once and return to the city he set off from (presumably his home). Such a journey is known as a tour. For a given list of cities and distances between them, what is the length of the tour with the shortest total distance?*

At this point, it is useful to fix conventions and nota-

tion. I consider an instance of the travelling salesman problem in two ways. First as a weighted graph, with the  $N \in \mathbb{N}$  cities of the problem being the nodes/vertices of the graph and being indexed with integer values: the distances between any two cities are captured by the weights on the edge between those same two cities. The second way is as a  $N \times N$  matrix, with the distance between the city with index  $i \in \mathbb{N}^{\leq N}$  and the city with index  $j \in \mathbb{N}^{\leq N}$  recorded as the  $(i, j)$  element of the matrix. Then, for easier discussion, let:

- $E_{i,j}$  denote the edge connecting nodes  $i$  and  $j$
- $d_{i,j}$  be the weight of  $E_{i,j}$ , referred to as the edge's length or distance.

### C. Linear Programming Formulation

Alternatively, the traveling salesman problem can be formulated more mathematically, as a linear program.

A linear program is any problem that may be expressed as minimising or maximising a linear combination of non-negative variables, which are subject to inequality conditions that are also formed of a linear combinations of those variables. The traveling salesman problem can be written as a linear program, as:

$$\min_{x_{i,j}} \left( \sum_{i=1}^N \left( \sum_{j=1}^N (d_{i,j} x_{i,j}) \right) \right), \quad (1)$$

subject to the conditions:

- A.  $\sum_{i=1}^N (x_{i,j}) = \sum_{j=1}^N (x_{i,j}) = 1,$
- B.  $\{x_{i,j}\}_{i,j=1}^N$  defines a tour,
- C.  $\forall i, j \in \mathbb{N}^{\leq N}, x_{i,j} \in \{0, 1\}.$

## II. GENERATING INSTANCES OF THE TRAVELING SALESMAN PROBLEM

### A. Duality

The core concept behind the following method is duality. Here duality is only discussed for the linear case, as this is all that is required.

It can be shown that optimising over the dual variables and the primal variables (the ones from the original problem), maximising one and minimising the other produces a new problem, known as the dual problem, which has the same solution as the original problem (known as the primal problem). Most importantly, it provides conditions for a solution being optimal, these are known as complementary slackness conditions.

### B. Nearly Degenerate Problem

The problem of finding both the shortest tour and the second shortest tour can be written as a linear program:

$$\min_{\vec{x}, \vec{y}} \left( \sum_{i=1}^N \sum_{j=1}^N \left[ d_{i,j} (x_{i,j} + y_{i,j}) \right] \right), \quad (2)$$

subject to the conditions:

- A.  $\sum_{i=1}^N (x_{i,j}) = \sum_{i=1}^N (y_{i,j}) = 1, \quad \forall j \in \mathbb{N}^{\leq N};$
- B.  $\sum_{j=1}^N (x_{i,j}) = \sum_{j=1}^N (y_{i,j}) = 1, \quad \forall i \in \mathbb{N}^{\leq N};$
- C.  $x_{i,j}, y_{i,j} \in \{0, 1\} \quad \forall i, j \in \mathbb{N}^{\leq N};$
- D.  $\{x_{i,j}\}_{i,j=1}^N, \{y_{i,j}\}_{i,j=1}^N$  each define a tour;
- E.  $\sum_{i=1}^N \sum_{j=1}^N \left[ d_{i,j} (x_{i,j} - y_{i,j}) \right] > 0.$

As shown previously, the traveling salesman problem is NP-hard. Solving this problem gives the solution to the corresponding traveling salesman problem. Therefore, this problem is also NP-hard. A useful way forward, in the spirit of Ref. [16], is to use duality to find complimentary slackness conditions for nearly optimal tours. By relaxing the condition that the optimal solution define a tour and changing the condition,  $\forall i, j \in \mathbb{N}^{\leq N}$  to:

$$x_{i,j}, y_{i,j} \in \{0, 1\} \quad \text{to} \quad x_{i,j}, y_{i,j} \in \mathbb{R}^+ \quad (3)$$

This gives the auxiliary problem:

$$\min_{\vec{x}, \vec{y}} \left( \sum_{i=1}^N \sum_{j=1}^N \left[ d_{i,j} (x_{i,j} + y_{i,j}) \right] \right), \quad (4)$$

subject to the conditions:

- A.  $\sum_{i=1}^N (x_{i,j}) = \sum_{i=1}^N (y_{i,j}) = 1, \quad \forall j \in \mathbb{N}^{\leq N};$
- B.  $\sum_{j=1}^N (x_{i,j}) = \sum_{j=1}^N (y_{i,j}) = 1, \quad \forall i \in \mathbb{N}^{\leq N};$
- C.  $x_{i,j}, y_{i,j} \in \mathbb{R}^+ \quad \forall i, j \in \mathbb{N}^{\leq N};$
- D.  $\sum_{i=1}^N \sum_{j=1}^N \left[ d_{i,j} (x_{i,j} - y_{i,j}) \right] > 0.$

### C. The Dual of the Auxiliary Problem

The first step in finding the dual problem of any primal problem is to find the Lagrangian,  $\mathcal{L}$ :

$$\mathcal{L} = \sum_{i=1}^N \sum_{j=1}^N \left( d_{i,j} (x_{i,j} + y_{i,j}) - \alpha_j^x x_{i,j} - \alpha_j^y y_{i,j} - \beta_i^x x_{i,j} - \beta_i^y y_{i,j} - \Gamma d_{i,j} (x_{i,j} + y_{i,j}) \right) + \sum_{j=1}^N (\alpha_j^x + \alpha_j^y) + \sum_{i=1}^N (\beta_i^x + \beta_i^y) + \Gamma z \quad (5)$$

$$= \sum_{i=1}^N \sum_{j=1}^N \left( \left[ (1 - \Gamma) d_{i,j} - \alpha_j^x - \beta_i^x \right] x_{i,j} + \left[ (1 + \Gamma) d_{i,j} - \alpha_j^y - \beta_i^y \right] y_{i,j} \right) + \sum_{j=1}^N (\alpha_j^x + \alpha_j^y + \beta_j^x + \beta_j^y) + \Gamma z. \quad (6)$$

For the Lagrangian to have a finite minimum, it is required that  $\forall i, j = 1, \dots, N$ :

- A.  $\left[ (1 - \Gamma) d_{i,j} - \alpha_j^x - \beta_i^x \right] \geq 0,$
- B.  $\left[ (1 + \Gamma) d_{i,j} - \alpha_j^y - \beta_i^y \right] \geq 0.$

The dual problem is then:

$$\max_{\alpha^x, \alpha^y, \beta^x, \beta^y} \left( \min_{\vec{x}, \vec{y}} [\mathcal{L}] \right). \quad (7)$$

I then note that  $\mathcal{L}$  is minimised over  $\vec{x}, \vec{y}$  when  $\forall i, j \in \mathbb{N}^{\leq N}$ :

$$\left[ (1 - \Gamma) d_{i,j} - \alpha_j^x - \beta_i^x \right] x_{i,j} = \left[ (1 + \Gamma) d_{i,j} - \alpha_j^y - \beta_i^y \right] y_{i,j} = 0. \quad (8)$$

This is the complementary slackness condition. Therefore, the dual problem is:

$$\max_{\alpha^x, \alpha^y, \beta^x, \beta^y} \left( \sum_{j=1}^N \left[ \alpha_j^x + \alpha_j^y + \beta_j^x + \beta_j^y \right] + \Gamma z \right), \quad (9)$$

subject to the conditions:

- A.  $(1 + \Gamma) d_{i,j} \geq \alpha_j^x + \beta_i^x,$
- B.  $(1 - \Gamma) d_{i,j} \geq \alpha_j^y + \beta_i^y.$

### D. Enforcing the Symmetry

The aim is to generate symmetric Traveling salesman problems. Therefore,  $\forall i, j \in \mathbb{N}^{\leq N}$ :

$$d_{i,j} = d_{j,i}. \quad (10)$$

To satisfy this requirement, first consider the edges in the optimal and nearly optimal tours:

$$(1 + \Gamma)d_{i,j} = \alpha_j^x + \beta_i^x \text{ and } (1 + \Gamma)d_{j,i} = \alpha_i^x + \beta_j^x \quad (11)$$

$$\Rightarrow \alpha_j^x + \beta_i^x = \alpha_i^x + \beta_j^x. \quad (12)$$

This condition can easily be met by,  $\forall i \in \mathbb{N}^{\leq N}$ , setting  $\alpha_i^x = \beta_i^x$ . The dual problem can then be rewritten as:

$$\max_{\alpha^x, \alpha^y} \left( 2 \sum_{j=1}^N \left[ \alpha_j^x + \alpha_j^y \right] + z\Gamma \right), \quad (13)$$

subject to the conditions:

$$\mathbf{A.} \quad (1 + \Gamma)d_{i,j} \geq \alpha_j^x + \alpha_i^x,$$

$$\mathbf{B.} \quad (1 - \Gamma)d_{i,j} \geq \alpha_j^y + \alpha_i^y.$$

### E. An Algorithm to Generate Problems

From the above conditions it is then possible to develop many algorithms that generate nearly degenerate instances of the traveling salesman problem. However, for this project the simplest algorithm that meets the requirements is developed. Using the above conditions, if  $\Gamma$  is set to zero, then the conditions reduce to that for degenerate solutions. If then, for simplicity, set:  $\alpha_i^x = \alpha_i^y, \forall 1 \leq i \leq N$ , as single condition is obtained:

$$d_{i,j} \geq \alpha_i + \alpha_j, \quad (14)$$

with equality only when  $d_{i,j}$  is part of the optimum tour, and where the superscripts have been neglected, so  $\alpha_i^x = \alpha_i^y = \alpha_i$ . This implies that for all  $d_{i,j}$  in an instance of the problem:

$$d_{i,j} = \alpha_i + \alpha_j + Z_{i,j}, \quad (15)$$

with  $Z_{i,j} = 0$  if and only if  $d_{i,j}$  is in the optimum tour.

**Theorem 1.** The length of any tour,  $T = (i_1, i_2, \dots, i_N)$ , in an instance of the traveling salesman problem is given by:

$$D = 2 \sum_{k=1}^N (\alpha_k) + \sum_{E_{i,j} \in T} (Z_{i,j}). \quad (16)$$

*Proof.* From the earlier formulation of the traveling salesman problem:

$$D = \sum_{E_{i,j} \in T} (d_{i,j}) = 2 \sum_{k=1}^N (\alpha_k) + \sum_{E_{i,j} \in T} (Z_{i,j}). \quad (17)$$

**Corollary 1.** The length of the optimal tour is:

$$2 \sum_{k=1}^N (\alpha_k). \quad (18)$$

A method for generating instances of the traveling salesman problem with a known gap between the optimal and nearly optimal tours can then be defined:

---

**Algorithm 1:** Algorithm generating instances of the travelling salesman problem

---

**Input:**  $N \in \mathbb{N}$ : the desired number of cities

1. Assume all  $N$  cities are connected with all connections initially and temporarily having no length
2. Randomly assign values to all the  $\alpha_i$
3. Randomly select an optimal tour
4. Set the distance of all the edges,  $d_{i,j}$ , in the optimal tour to  $\alpha_i + \alpha_j$
5. Select two cities that appear next to each other in the optimal tour and switch their places in the tour to obtain the nearly optimal tour
6. Select a random value,  $\varrho$  to be the difference between the optimal and nearly optimal tour
7. Set the distances of the edges in the nearly optimal tour but not in the optimal tour (there will be exactly 2) to  $d_{i,j} = \alpha_i + \alpha_j + \frac{\varrho}{2}$
8. For all edges without lengths assigned yet, assign  $d_{i,j} > \alpha_i + \alpha_j + \frac{\varrho}{2}$

---

**Output:**  $d_{i,j}, \forall i, j \in \mathbb{N}^{\leq N}$

---

The result of this is that the optimal tour has length:

$$2 \sum_{k=1}^N (\alpha_k), \quad (19)$$

and the nearly optimal tour has length:

$$D_{Near} = 2 \sum_{k=1}^N (\alpha_k) + \sum_{E_{i,j} \in T} (Z_{i,j}) = 2 \sum_{k=1}^N (\alpha_k) + \varrho, \quad (20)$$

All other tours then have length greater than:

$$D_{Other} = 2 \sum_{k=1}^N (\alpha_k) + \sum_{E_{i,j} \in T} (Z_{i,j}) > 2 \sum_{k=1}^N (\alpha_k) + \varrho. \quad (21)$$

### III. DISCUSSION

Herein, I have presented a method of efficiently generating instances of the travelling salesman problem, with known op-

□

timal and nearly-optimal solutions. This has been achieved by considering the dual problem of the linear programming formulation of the travelling salesman problem. The performance of algorithms to find slightly better solutions is of great importance and there are many instances where problems such

as those this algorithm can generate are useful. A key example is when the found solution will be used many times and a longer solution will incur a - potentially small, but it will build up over many uses - cost, such as when calculating delivery routes (which are more expensive; in terms of time, fuel, and labour; the longer they are).

- 
- [1] R. Karp, in *50 Years of Integer Programming 1958-2008* (Springer, Berlin, Heidelberg, 2010) pp. 219–241.
  - [2] E. Lawler, *The Travelling Salesman Problem: A Guided Tour of Combinatorial Optimization*, Wiley-Interscience series in discrete mathematics and optimization (John Wiley & Sons, 1985).
  - [3] J. Beardwood, J. H. Halton, and J. M. Hammersley, *Mathematical Proceedings of the Cambridge Philosophical Society* **55**, 299–327 (1959).
  - [4] S. Arora, *J. ACM* **45**, 753–782 (1998).
  - [5] S. Sahni and T. Gonzalez, *J. ACM* **23**, 555–565 (1976).
  - [6] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy, *J. ACM* **45**, 501–555 (1998).
  - [7] G. Gutin and A. Yeo, *Algorithmic Operations Research* **2**, 33 (2007).
  - [8] J. Bang-Jensen, G. Gutin, and A. Yeo, *Discrete Optimization* **1**, 121 (2004).
  - [9] G. Gutin, A. Yeo, and A. Zverovich, *Discrete Applied Mathematics* **117**, 81 (2002).
  - [10] A. Wren and A. Holliday, *Journal of the Operational Research Society* **23**, 333 (1972).
  - [11] A. R. Karlin, N. Klein, and S. O. Gharan, in *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2021 (Association for Computing Machinery, New York, NY, USA, 2021) p. 32–45.
  - [12] R. van Bevern and V. A. Slugina, *Historia Mathematica* **53**, 118–127 (2020).
  - [13] C. Rego, D. Gamboa, F. Glover, and C. Osterman, *European Journal of Operational Research* **211**, 427 (2011).
  - [14] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, *Science* **220**, 671 (1983).
  - [15] C. L. Valenzuela and A. J. Jones, *European Journal of Operational Research* **102**, 157 (1997).
  - [16] J. L. Arthur and J. O. Frendewey, *The Journal of the Operational Research Society* **39**, 153 (1988).