Assessment of approximation method for TSP path length on road networks: a simulation study

Bachelor Thesis

Koen Stevens, S5302137

2025-04-19

Abstract

1 Introduction

The Traveling Salesman Problem (TSP) is an important problem in operations research. It is particularly relevant for last-mile carriers and other logistics companies where efficient routing directly impacts cost, time and service quality. Since the number of parcels worldwide has increased between 2013 and 2022 and is expected to keep increasing [Statista (2025)], the need for fast, scalable route planning methods becomes ever more pressing.

The TSP is an NP-hard problem, it is computationally intensive to find the exact solution for large instances. In many real-world scenarios, the exact optimal routes may not be needed, but instead a rough, reliable estimate of the optimal route length. For instance, consider a postal delivery company. This firm may need to assign a certain amount of deliveries or a certain area to each postman. Reliable estimates for the route length can provide valuable information for making such decisions.

Efficient approximation methods provide a solution for such practical applications where exact solutions are too computationally intensive to conduct or not feasible due to insufficient data. These methods aim at approximating the expected optimal total travel time or distance, while using minimal data and computational effort.

There is extensive research on such approximation methods and how they perform in the euclidean plane. Consider n uniformly drawn locations from some area in $\mathbb{R}^2$ with area A. Beardwood, Halton, and Hammersley (1959) prove the relation:

$$L \to \beta\sqrt{nA}, \quad \text{as } n \to \infty \tag{1}$$

as an estimation for the length of the shortest TSP path measured by Euclidean distance through these random locations, where β is some proportionality constant. This formula is a very elegant result, and it requires very little data. However, its assumptions, uniform random locations and euclidean space differ from real-world applications, which are defined by complex geographic features, such as road networks.

This research investigates how well this approximation method performs when considering real road networks. Using OpenStreetMap data, TSP instances are simulated in a wide variety of different urban areas in the Netherlands, then solve these for the actual shortest paths using the Lin–Kernighan (Lin and Kernighan 1973) heuristic. Then the β from equation 1 is estimated and the performance of this formula is analyzed. Additionally, the results for β and the performance across the selected areas is compared.

In section 2 a deep dive in the context and previous research in this field is provided. In section 3 the experimental design is documented.

2 Literature Review

In this section the existing literature on the Beardwood formula and some applications, and on the Lin-Kernighan heuristic and its implementations is reviewed.

2.1 Applications of the Beardwood formula

This research concerns the performance of formula 1 for reasonable amounts of locations a delivery person can visit in a workday, say $10 \leq n \leq 90$. Lei et al. (2015) estimates the values of β for a selection of values for n. In their research, the points were generated uniformly and the L_2 distance metric was used. Table 1 lists the results.

Table 1: Empirical estimates of β as a function of n, $20 \leq n \leq 90$ (Lei et al. (2015))

n	$\beta(n)$
20	0.8584265
30	0.8269698
40	0.8129900
50	0.7994125
60	0.7908632
70	0.7817751
80	0.7775367
90	0.7773827

Figliozzi (2008) is the first research to apply approximation formulas to real-world instances of TSPs (and VRPs (Vehicle Routing Problems)). An extension of formula 1 that works for VRPs is assessed in a real-world setting. It is found that this model has an R^2 of 0.99 and MAPE (Mean Absolute Prediction Error) of 4.2%. This prediction error is slightly higher than when it is applied to a setting where euclidean distances are considered (3.0%), but the formula still performs well (Figliozzi (2008)).

Merchán and Winkenbach (2019) use circuity factors to measure the relative detour incurred for traveling in a road network, compared to the euclidean distance. This circuity factor is defined as, where p and q are locations:

$$c = \frac{d_c(p,q)}{d_{L_2}(p,q)} \quad (2)$$

By construction, c is greater or equal to 1, a value closer to 1 indicates a more efficient network. Then, β_c is estimated by $\beta_c = c\beta$. This value c, is estimated for three different areas in São Paulo, for which the results are listed in table 2. These values indicate real travel distances are on average 2.76 times longer in area 1 compared to the L_2 metric. These values were obtained by uniformly generating n locations (for n ranging from 3 to 250), computing near-optimal tour lengths under the Euclidean metric, and solving for β, then scaling by the empirical circuity factor.

Table 2: Estimates of the circuity factor c and its corresponding β_c (Merchán and Winkenbach (2019))

	Area 1	Area 2	Area 3
c	2.76	2.34	1.82
β_c	2.48	2.10	1.64

It is important to note, however, that the assumptions in this study may limit the generality of the findings. In particular, the use of uniformly distributed locations does not accurately reflect the spatial distribution of delivery points in real urban environments, where locations tend to cluster in residential, commercial, or industrial zones. Additionally, within small urban areas, high-rise buildings and single-family homes may coexist in the same neighborhoods, further challenging the assumption of uniformly distributed delivery points. Furthermore, the circuity factor c can vary significantly within a single city, depending on local street patterns, infrastructure, and topography. These variations suggest that a fixed circuity factor may oversimplify the complexity of real-world delivery contexts, especially when applied to smaller subregions or neighborhoods.

2.2 Lin-Kernighan Heuristic

To be able to efficiently solve many TSPs, to find a good estimate for β, a fast and reliable solution algorithm is needed. The Lin-Kernighan (Lin and Kernighan 1973) heuristic provides outcome, it is generally considered to be one of the most effective methods of generating (near) optimal solutions for the TSP. In this research a modified implementation of the heuristic is used (Helsgaun 2000). The run times of both heuristics increase by approximately $n^{2.2}$, but the modified heuristic is much more effective. It is able to find optimal solutions to large instances in reasonable times (Helsgaun 2000).

PARAGRAPH ABOUT HOW THE HEURISTIC WORKS

3 Experimental design

3.1 Data

3.2 Generation and solving of TSPs

4 Results

5 Discussion

6 Conclusion

7 References

Beardwood, Jillian, John H Halton, and John Michael Hammersley. 1959. "The Shortest Path Through Many Points." In *Mathematical Proceedings of the Cambridge Philosophical Society*, 55:299–327. 4. Cambridge University Press.

Figliozzi, Miguel Andres. 2008. "Planning Approximations to the Average Length of Vehicle Routing Problems with Varying Customer Demands and Routing Constraints." *Transportation Research Record* 2089 (1): 1–8.

Helsgaun, Keld. 2000. "An Effective Implementation of the Lin–Kernighan Traveling Salesman Heuristic." *European Journal of Operational Research* 126 (1): 106–30.

Lei, Hongtao, Gilbert Laporte, Yajie Liu, and Tao Zhang. 2015. "Dynamic Design of Sales Territories." *Computers & Operations Research* 56: 84–92.

Lin, Shen, and Brian W Kernighan. 1973. "An Effective Heuristic Algorithm for the Traveling-Salesman Problem." *Operations Research* 21 (2): 498–516.

Merchán, Daniel, and Matthias Winkenbach. 2019. "An Empirical Validation and Data-Driven Extension of Continuum Approximation Approaches for Urban Route Distances." *Networks* 73 (4): 418–33.

Statista. 2025. "Global Parcel Shipping Volume Between 2013 and 2027 (in Billion Parcels)*." https://www.statista.com/statistics/1139910/parcel-shipping-volume-worldwide/.

8 Appendix

Province	Neighborhood	β	MAE
groningen	Hortusbuurt	2.2140	0.0813
groningen	Binnenstad	2.0544	0.0825
groningen	Oosterpoort	2.0563	0.0629
groningen	Rivierenbuurt	1.7688	0.0535
groningen	De Wijert	1.7599	0.0738
groningen	Oosterparkwijk	1.7643	0.0639
groningen	De Hoogte	1.7015	0.0639
groningen	Korrewegwijk	2.0562	0.0613
groningen	Schildersbuurt	2.2306	0.0585
groningen	Paddepoel	1.6707	0.0567
groningen	Oranjewijk	1.9327	0.0627
groningen	Tuinwijk	2.8923	0.0624
groningen	Selwerd	1.5130	0.0589
groningen	Vinkhuizen	1.4727	0.0483
groningen	Hoogkerk-zuid	1.5541	0.0728
groningen	Gravenburg	1.3103	0.0937
groningen	De Held	1.9304	0.0502
groningen	Reitdiep	1.6486	0.0571
groningen	Hoornse Meer	1.5956	0.0504
groningen	Corpus den Hoorn	1.6048	0.0566
groningen	Eemspoort	1.7358	0.0587
groningen	Euvelgunne	1.9045	0.1311
groningen	Driebond	1.8890	0.1046
groningen	Winschoterdiep	1.9587	0.0587
groningen	Eemskanaal	1.7677	0.0509
groningen	Helpman	2.0763	0.0678
groningen	Lewenborg	1.9188	0.0562
groningen	Beijum	1.7990	0.0442
groningen	Maarsveld	1.6826	0.0498
noord holland	Schrijverswijk	1.7471	0.0450
noord holland	Stad van de Zon	1.4905	0.1750
noord holland	Stadshart	1.4624	0.0732
noord holland	Jordaan	1.8342	0.0605
noord holland	Slotervaart	1.7528	0.0472
noord holland	IJburg	1.3401	0.0577
noord holland	Oostelijke Eilanden	1.7038	0.0584
noord holland	Oostelijk Havengebied	1.7370	0.0649
noord holland	Frederik Hendrikbuurt	2.2882	0.0622
noord holland	Van Lennepbuurt	1.7975	0.0680
noord holland	Da Costabuurt	2.4979	0.0668
noord holland	Kinkerbuurt	1.9964	0.0662
noord holland	Kersenboogerd	1.6460	0.0598

Province	Neighborhood	β	MAE
noord holland	Pax	2.2049	0.0606
noord holland	Graan voor Visch	2.2113	0.0596
noord holland	Vrijschot-Noord	2.3914	0.0789
noord holland	Toolenburg	1.2931	0.0490
noord holland	Floriande	1.9109	0.0542
noord holland	Overbos	1.7852	0.0490
noord holland	Bornholm	1.8148	0.0481
noord holland	Beukenhorst-Oost	1.7036	0.0959
noord holland	De Hoek	2.5377	0.0671
noord holland	West	2.0166	0.0587
noord holland	Zuid	1.6606	0.0635
noord holland	Oost	1.8681	0.0655
noord holland	Noord	1.6501	0.0559
noord holland	De President	1.5044	0.1347
noord holland	Graan voor Visch-Zuid	1.7181	0.0661
noord holland	Zuidwijk	1.3921	0.0476
noord holland	Buitenveldert-West	1.1965	0.0751
noord holland	Buitenveldert	1.1454	0.0792
noord holland	Apollobuurt	1.7341	0.0709
noord holland	Stadionbuurt	1.4915	0.0724
noord holland	Prinses Irenebuurt e.o.	1.8949	0.0473
noord holland	Hoofddorppleinbuurt	1.6722	0.0809
noord holland	Willemspark	1.9371	0.0640
noord holland	Schinkelbuurt	1.6451	0.0956
noord holland	Vondelparkbuurt	1.2848	0.0873
noord holland	Helmersbuurt	1.8131	0.0564
noord holland	Overtoomse Sluis	1.9524	0.0624
noord holland	Museumkwartier	1.7137	0.0719
noord holland	Rivierenbuurt	1.6937	0.0554
noord holland	IJselbuurt	1.5689	0.0710
noord holland	Scheldebuurt	1.3590	0.0595
noord holland	Rijnbuurt	1.6511	0.0557
noord holland	De Baarsjes	1.7948	0.0648
noord holland	Landlust	1.7654	0.0548
noord holland	Staatsliedenbuurt	1.8190	0.0609
noord holland	Spaarndammerbuurt	2.2411	0.0603
noord holland	De Pijp	2.2764	0.0572
noord holland	Grachtengordel	1.7978	0.0616
noord holland	Oud-Zuid	1.4458	0.0649
noord holland	De Pijp	2.2764	0.0572
noord holland	Grachtengordel	1.7978	0.0616

noord holland	Oud-Zuid	1.4458	0.0649