

**Analysis and Optimization of the**

**Charging Station Network in Germany**

Scientific work for the attainment of the degree

Master of Science

at the TUM School of Management

of the Technical University of Munich

Examiner: Univ.-Prof. Dr. Dr. h. c. mult. Horst Wildemann

Research Institute

Corporate management, logistics and production

Supervisor: Sebastian Junker M.Sc.

Course of study: Master TUM-BWL

Submitted by: Giovanni Filomeno

Arcistraße 21

80333 Munich

Matricula 03669006

Submitted on:

Table of contents

[Table of contents I](#_Toc138415438)

[List of figures II](#_Toc138415439)

[List of tables III](#_Toc138415440)

[1. Introduction 4](#_Toc138415441)

[1.1 Anxiety of Range 4](#_Toc138415442)

[1.2 The infrastructure-EV Adoption Nexus 4](#_Toc138415443)

[1.3 The Role of Government Policies and Regulations 4](#_Toc138415444)

[1.4 EV Prices, Battery Capacity and Charging Infrastructure 4](#_Toc138415445)

[2. Model and Analysis 6](#_Toc138415446)

[2.1 Network Theory 6](#_Toc138415447)

[2.1.1 Diameter 6](#_Toc138415448)

[2.1.2 Average Distance 6](#_Toc138415449)

[2.1.3 Average Clustering 7](#_Toc138415450)

[2.2 Genetic Algorithm 8](#_Toc138415451)

[2.2.1 Process 9](#_Toc138415452)

[2.2.2 Variants 10](#_Toc138415453)

[2.2.3 Strengths and Weakness of Genetic Algorithms 10](#_Toc138415454)

[3. Results 11](#_Toc138415455)

[3.1 Pre-Processing 11](#_Toc138415456)

[3.2 Data analysis 12](#_Toc138415457)

[3.3 Network construction 14](#_Toc138415458)

[3.4 Optimization 18](#_Toc138415459)

[4. Conclusion and Further Implementations 18](#_Toc138415460)

[Bibliography IV](#_Toc138415461)

List of figures

[Figure 1: Cumulative histogram per year 13](#_Toc138415544)

[Figure 2: Cumulative histogram per year divided into Normal and Fast charging 13](#_Toc138415545)

[Figure 3: Power clustering histogram per year 13](#_Toc138415546)

[Figure 4: Charging Type clustering per year 14](#_Toc138415547)

[Figure 5: Evolution of the charging station network in Germany from 2009 to 2022 17](#_Toc138415548)

# List of tables

[Table 1: Example table 11](#_Toc138415467)

# Introduction

The transformation of the global automobile industry toward electric mobility is underway. A critical factor influencing this trend is the availability and accessibility of Electric Vehicle (EV) charging infrastructure. It has been widely studied and proven that the prevalence of charging stations strongly influences the acceptance and adoption of EVs [1] [2] [3] [4].

## Anxiety of Range

The phenomenon termed as "range anxiety" is a common concern among potential EV owners. Range anxiety refers to the worry that an EV has insufficient reach to get to its destination and would leave the driver stranded. It is one of the key barriers to EV adoption [5]. This fear can be mitigated by the availability of a comprehensive, convenient, and reliable charging network, thus directly linking EV market penetration to charging infrastructure.

## The infrastructure-EV Adoption Nexus

Studies have demonstrated a strong correlation between the availability of public charging stations and the adoption of EVs. [4] conducted a study in the U.S, showing that the existence of public charging infrastructure positively correlates with EV uptake. When charging infrastructure was more available and accessible, the likelihood of consumers purchasing an EV increased. Moreover, [6] examined the influence of various factors on international electric vehicle policies and EV sales in 30 countries. They found that charging infrastructure was one of the most substantial determinants of national electric vehicle market share.

## The Role of Government Policies and Regulations

Government policies also play a crucial role in the proliferation of charging stations and consequently, the promotion of EV adoption. Incentives like grants, subsidies, and tax reductions have been applied in various countries to accelerate the deployment of EV charging stations [7]. Such policy measures have had a substantial impact on both the expansion of charging infrastructure and the adoption of electric vehicles.

## EV Prices, Battery Capacity and Charging Infrastructure

While the charging infrastructure is a significant factor, other elements like EV prices, government subsidies, and battery capabilities are also key contributors to EV market penetration [7]. Reduction in battery prices and improvements in battery technology have increased the affordability and driving range of EVs, thereby supporting the growth of the EV market. Nevertheless, without an extensive and reliable charging network, widespread EV adoption remains a challenge

# Model and Analysis

## Network Theory

### Diameter

The diameter of a graph is a fundamental concept in the field of graph theory, offering significant insights into the overall structural properties of the graph. In a formal sense, the diameter of a graph is defined as the greatest shortest path length between any pair of vertices within the graph. It signifies the longest of all the shortest paths which can be navigated from one vertex to any other vertex within the graph [8]. To express this mathematically, let's denote a graph as G = (V, E), where V is the set of vertices and E is the set of edges present in the graph. For any pair of vertices, u and v in V, let d(u, v) denote the shortest path distance between these vertices. Thus, the diameter D of G is represented by the formula:

D = max {d(u, v) | u, v ∈ V}

Here, the diameter D is the maximum over all shortest path distances d(u, v) for every pair of vertices u and v in the graph [9]. In the case of a disconnected graph, where there exist vertices that do not have a path between them, the diameter is often defined as infinity. For a graph containing only a single vertex, the diameter is defined as 0. The concept of the diameter of a graph is applied in numerous applications, such as in network design and analysis, devising algorithms for effective information routing in distributed systems, and in discerning the structure of social networks among others [10]. It's important to note that the process of calculating the diameter of a graph can be computationally expensive, particularly for large graphs. A number of algorithms have been developed to compute the diameter of a graph, which include but are not limited to, the Floyd-Warshall algorithm and Johnson’s algorithm [9].

### Average Distance

The average distance in a graph, also known as the average path length or the characteristic path length, is another crucial concept in graph theory. This metric gives an indication of the overall navigability and connectivity of the graph. Formally, the average distance L is the mean shortest path length between all pairs of vertices in the graph. That is, it represents the expected distance between two vertices chosen uniformly at random [10]. Let's denote a graph as G = (V, E), where V is the set of vertices and E is the set of edges in the graph. For any pair of vertices, u and v in V, let d(u, v) denote the shortest path distance between these vertices. The average distance L of G can be given by the formula:

L = Σ d(u, v) / n(n - 1)

Here, the sum is over all pairs of distinct vertices, and n is the number of vertices in the graph. Essentially, we are summing up the shortest path distances between all pairs of vertices and then dividing by the total number of such pairs [9]. The average distance is a critical metric in understanding the properties of real-world networks, which often exhibit the small-world property. The small-world property describes networks where the average path length is relatively small, meaning that one can get from any given node to any other node in the network through a small number of steps [10]. It's important to note that the average distance only makes sense for connected graphs, where a path exists between every pair of vertices. If a graph is not connected, this metric can't be defined without modification. The computation of average distance can be a computationally intensive process, especially for larger graphs, due to the necessity to calculate the shortest path between each pair of vertices. Efficient algorithms have been developed to aid in this calculation, such as the Floyd-Warshall algorithm [9].

### Average Clustering

The average clustering coefficient is an essential measure in the field of graph theory, frequently used to evaluate the tendency of nodes in a network to cluster together. It provides insights into the overall clustering of the graph and the interconnectedness of its nodes. The clustering coefficient of a single node in a graph quantifies how close its neighbors are to being a complete graph. A complete graph is one where every pair of vertices is connected by a unique edge. The average clustering coefficient is simply the mean of the clustering coefficients of all the nodes in the graph [10]. To be more precise, for a given node v in a graph G = (V, E), where V is the set of vertices and E is the set of edges, let Kv denote the number of vertices that are neighbors of v, and let Ev denote the number of edges between the neighbors of v. The clustering coefficient Cv for the node v is given by:

Cv = 2Ev / Kv(Kv - 1)

The average clustering coefficient C of the graph is then calculated as:

C = Σ Cv / n

Here, the sum is over all vertices in the graph, and n is the total number of vertices [9]. The average clustering coefficient measures the degree to which nodes in a graph tend to cluster together. It has found widespread application in the study of various types of networks, such as social networks, biological networks, and the World Wide Web, among others [10]. In calculating the average clustering coefficient, it is important to note that it is typically only defined for nodes with at least two neighbors, since the denominator of the formula for Cv would be zero for nodes with fewer than two neighbors. Computing the average clustering coefficient can be a computationally intensive task, especially for larger networks. Various algorithms have been developed to more efficiently calculate it, but the computational complexity is generally high due to the need to examine the local neighborhood of each node [9].

## Genetic Algorithm

Genetic Algorithms (GAs) are a family of search and optimization techniques inspired by the concept of natural selection [11]. Throughout time, GAs have been effectively used to resolve a vast variety of engineering challenges, covering function optimization, machine learning, and scheduling, as cited in reference [12]. This section provides a comprehensive overview of Genetic Algorithms, addressing their theoretical basis, elements, and variants. It also points out the positive and negative aspects of GAs.

The theoretical inspiration of GAs is based on the fundamental principles of biological evolution. According to Darwin's theory of evolution, the most physically fit people are chosen for reproduction, passing on their qualities to the next generation. Over a period of time, this phenomenon results in the acclimatization of species to their surroundings, thereby enhancing their likelihood of survival.

In GAs, a population consists of a set of potential solutions, each referred to as an individual. Genetic operators such as selection, crossover, and mutation are applied to this population. There operators simulate an evolutionary process that occurs over multiple generations [11]. The iterative nature of GAs enables efficient search space exploration and convergence to optimal or near-optimal solutions [13].

### Process

**Representation** is an essential component of Genetic Algorithms, as it defines how potential solutions are encoded for the manipulation within the algorithm. The format that is used the most often is the binary string, where each possible answer is represented as a series of binary digits (0s and 1s) [11]. According to the nature of the issue, different representations, such as real-valued vectors, integers, and permutations, may also be utilized [12].

An **initial population** of potential solutions is created during the startup phase (cf. **Error! Reference source not found.**). Both random and domain-specific information may be used to create this population [14]. The choice of population size is based on the complexity of the issue at hand and the preferred ratio of exploitation to exploration [15].

XXXXXXXXXXXXXXXXXXXXXXXXX

The **fitness function** is a way to measure the “quality” of each individual in the population. It acts as a gauge of how effectively the solution fulfills both the objectives and the constraints of the challenge [11]. The fitness function must be created to represent the intended result and to direct the search for the best solutions [16].

**Selection** is the process of choosing individuals from the present population to engage in reproduction in order to have the population aggregate around an ideal solution (cf. **Error! Reference source not found.**). This mechanism has a preference for those with greater levels of fitness, which is consistent with the concept of "survival of the fittest" [11]. Numerous selection methods have been put forward in the literature, including proportional selection [12], tournament selection [17], and ranking-based selection [18], have been proposed in the literature.

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

The act of integrating the genetic material from two parent solutions to produce one or more child solutions is known as a **crossover**, often referred to as recombination. Crossover aims to explore the search area by developing new solutions that may have higher fitness values [11]. Common crossover operators include single-point, multi-point, and uniform crossover [12].

**Elitism** is an optional element of genetic algorithms that guarantees the preservation of the best person(s), called incumbent, in each generation [13]. Elitism works to stop the loss of the best solutions brought on by genetic operators (e.g., crossover) and to direct search efforts in the direction of the ideal solution by keeping a copy of the fittest individual(s) [12].

### Variants

In order to improve their efficiency and customize them for certain problem domains, many variations of genetic algorithms have been presented. Below is a discussion of a few of the more common variations:

* Real-valued Genetic Algorithms (RVGAs): employ real-valued representations rather than binary strings, enabling a more accurate and effective description of certain problem domains [19]. Arithmetic crossover [20] and non-uniform mutation [21] are two examples of the specific crossover and mutation operators needed for RVGAs.
* Constraint-Handling Techniques: GAs must take into account restrictions in order to effectively solve optimization issues in the real world. Numerous methods for addressing constraints have been put forward, such as penalty functions [22], repair algorithms [23], and multi-objective approaches [16].
* Hybrid Genetic Algorithms (HGAs): combine GAs with additional optimization methods to enhance convergence and search effectiveness [24]. Hill climbing or simulated annealing are two examples of local search techniques that are often used into HGAs to improve the solutions produced by GAs [25].
* Parallel Genetic Algorithms (PGAs): make use of GAs' built-in parallelism to improve performance [26]. PGAs may be implemented using a variety of parallelization techniques, including island models, fine-grained parallelization, and global parallelization.

### Strengths and Weakness of Genetic Algorithms

Strengths:

* Global search capability: due to their lower propensity to get stuck in local optima than gradient-based approaches, GAs are well-suited for identifying global optima in complicated search domains [12].
* Applicability: GAs can be applied to a wide range of optimization problems, as they only require the definition of a fitness function and an appropriate representation [11].
* Robustness: GAs are resilient and simple to apply in real-world problems since they are comparatively insensitive to parameter selection [13].
* Parallelism: GAs may be significantly speed up by leveraging parallel computer resources since they are inherently parallelizable [26].

Weakness:

* Slow convergence: GAs may take a while to converge, particularly in cases where the search spaces are big and complex, or the population is huge [12].
* Premature convergence: GAs can sometimes converge prematurely to suboptimal solutions, particularly if the selection pressure is too high or the mutation rate is too low [13].
* Problem-dependent performance: depending on the problem domain and the selected representation, crossover, and mutation operators, the performance of GAs might vary dramatically [12].

Parameter tuning: although GAs are generally reliable, determining the ideal set of parameters for a given issue may be difficult and time-consuming [15].

# Results

Test

|  |  |
| --- | --- |
| **A** | **B** |
| Lorem | 42,0% |
| Ipsum | 37,5% |
| Dolor | 37,5% |

Table 1: Example table

## Pre-Processing

The raw dataset, procured from the Federal Network Agency, underwent meticulous preprocessing to render it suitable for subsequent analysis. This preprocessing involved several steps aimed at enhancing the quality and readability of the data, thereby facilitating more accurate and effective analysis. One key aspect of the preprocessing was the elimination of duplicate entries from the dataset. It is not uncommon for datasets, particularly those aggregated from multiple sources or entered manually, to contain duplicate entries. Duplicate entries, if not removed, can lead to redundancy in the data and distort the subsequent analysis. Thus, duplicate entries in the dataset were identified and carefully pruned to create a streamlined and precise representation of the unique electric vehicle charging stations distributed across the regions. Following this, the data underwent a process of transformation aimed at standardizing its structure and making it more conducive to analysis. A prime example of this is the transformation of the '*type\_of\_charger*' variable. The original values, '*Schnellladeeinrichtung*' and '*Normalladeeinrichtung*', were remapped to 'fast' and 'normal' respectively. This alteration simplified the task of classifying charging stations based on their charging speed, thereby paving the way for more straightforward subsequent analysis. In the original dataset, geographical coordinates were presented in a format unsuitable for computational processing, with commas employed as decimal points. This posed significant challenges for any computational or spatial analyses that relied on these coordinates. To address this, the commas were replaced with decimal points, and these columns were converted to the float data type. Consequently, calculations involving distances became significantly more straightforward, and the feasibility of creating detailed geospatial visualizations was substantially improved. Another vital transformation step involved the '*commissioning\_date*' column, which stored the commissioning date of each charging station as a string in the day-month-year format. For a more detailed and useful chronological analysis, it was necessary to convert this data into a datetime format. This modification facilitated an overview of the temporal progression of charging station installations, offering valuable insights into their spread over time. Additionally, leading and trailing spaces found in several columns of the dataset were removed to maintain uniformity in the data. Ensuring this consistency was especially important for object columns, such as city names, where minor differences in formatting could lead to inaccurate grouping or comparison of values. The city names present in the dataset were also standardized for improved accuracy and simplicity. Distinct neighborhoods or districts within a city, initially represented separately, were consolidated under the name of the main city. This step significantly reduced geographical complexity during the analysis, fostering a more coherent representation of data. In a further move to streamline the dataset, certain columns containing public keys were dropped. Although these keys are commonly used for cryptographic operations, they were deemed irrelevant to the intended study and were therefore excluded from the dataset, leading to a more simplified data structure. In the final stage of preprocessing, the refined dataset was saved into a new CSV file termed '*ChargingStationCleaned.csv*'. This file stands as a meticulously cleaned and preprocessed version of the original dataset, primed for a detailed examination of the distribution, characteristics, and evolution of electric vehicle charging stations across Germany.

## Data analysis

Test

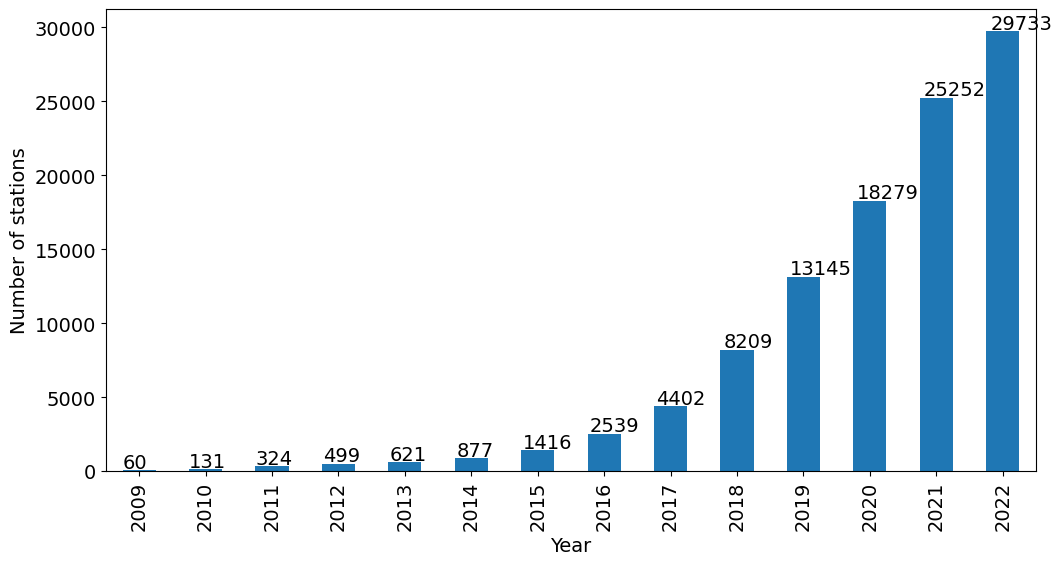


Figure 1: Cumulative histogram per year

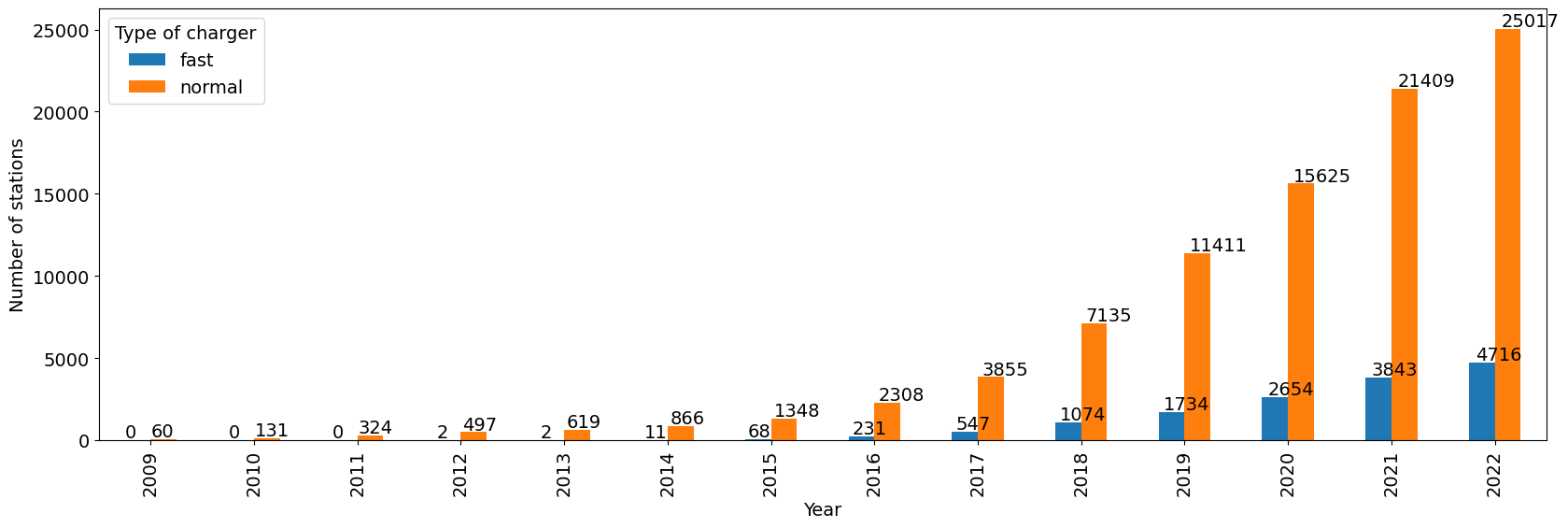


Figure 2: Cumulative histogram per year divided into Normal and Fast charging

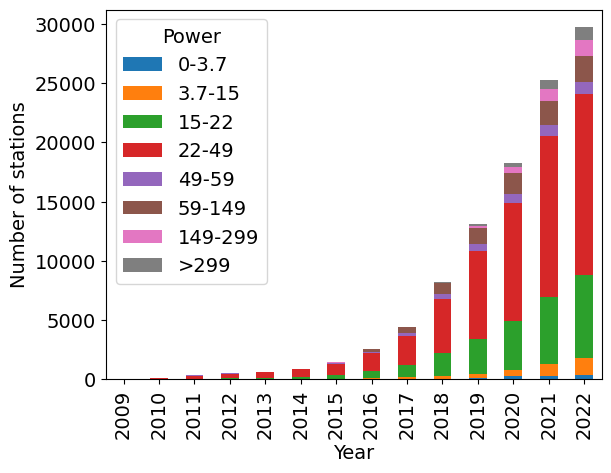


Figure 3: Power clustering histogram per year

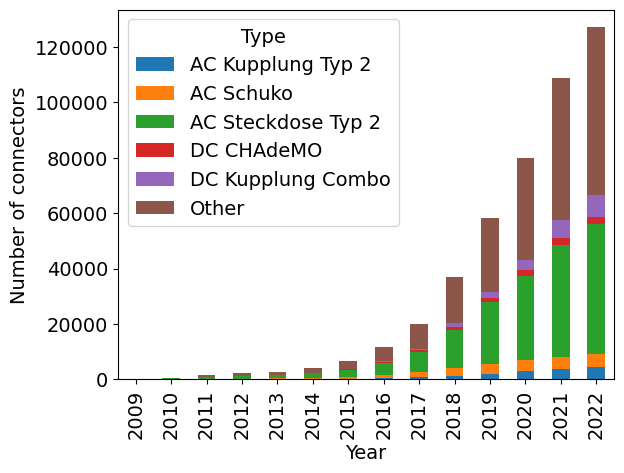


Figure 4: Charging Type clustering per year

## Network construction

To further scrutinize the distribution and accessibility of electric vehicle charging stations across various geographical regions, a meticulous process for network creation was deployed. The outcome was a detailed and comprehensive graphical representation, visualizing connections between the charging stations for specified years and within user-determined distance parameters. Several programming libraries, each with a unique role, were brought into play in this complex process. The sqlite3 library provided the means for interaction with a SQLite database. The requests library was employed to handle HTTP requests required for distance calculation, whereas the math library was used to compute intricate mathematical operations, including trigonometric functions necessary for distance calculations. The urllib3 library, an HTTP client for Python, managed and streamlined HTTP protocol details. The Transformer module from the pyproj library was crucial for geographic coordinate conversions. An important part of this procedure was the calculation of the Haversine distance, a method utilized to determine the 'great-circle' distance between two points on the Earth's surface given their longitudes and latitudes. Here, the Earth was assumed to be a perfect sphere, and its radius was taken as 6371 kilometers. The function get\_osrm\_distance was developed for more accurate distance measurements. This function initially made a quick estimation based on the Haversine formula. If this approximated distance was found to be less than or equal to 100 kilometers, the Open-Source Routing Machine (OSRM) service was employed. OSRM is an open-source service that uses data from the OpenStreetMap project to find the shortest routes between coordinates. Efficiency and speed were prioritized by creating a SQLite database named 'distances.db', intended to save previously calculated distances. This efficient system considerably reduced the need for repetitive API calls to the OSRM service. If a previously calculated distance value was found in the database, it was retrieved and returned. If not, the function executed an API call to the OSRM service, acquired the precise distance, and subsequently stored this new distance value in the database for future use. The subsequent phase of the process was encapsulated in the create\_network function. This function utilized the previously described distance calculations to construct an undirected graph, which served as a representation of the network of charging stations for a particular year and within a user-specified distance threshold. Each node in the graph corresponded to an electric vehicle charging station, assigned with specific attributes like latitude, longitude, and federal state. Edges between these nodes were added based on the distance between them, calculated by the get\_osrm\_distance function. After the graph's construction, it was visualized on a map. For geographical reference, the boundaries of Germany were used as a backdrop. These boundaries were acquired from a GeoJSON file. The resulting visual output was a graph overlaid on Germany's geographic boundaries, portraying the spread of charging stations and the connectivity between them. The final phase of this process was storing the constructed graph for future uses. The graph was saved in the GraphML format, a comprehensive and easy-to-use file format for graphs, enabling potential future network analysis. Moreover, a visual representation of the graph was saved as a PNG image. This image served as an easily accessible, ready-to-view snapshot of the network for the designated year and distance threshold, aiding in immediate analysis and comparison.

|  |  |
| --- | --- |
|  |  |
| (2009) | (2010) |
|  |  |
| (2011) | (2012) |
|  |  |
| (2013) | (2014) |
|  |  |
| (2015) | (2016) |
|  |  |
| (2017) | (2018) |
|  |  |
| (2019) | (2020) |
|  |  |
| (2021) | (2022) |

Figure 5: Evolution of the charging station network in Germany from 2009 to 2022

## Optimization

Test

# Conclusion and Further Implementations

# Bibliography

|  |  |
| --- | --- |
| [1] | S. Jordan, D. Newport, S. Sandland and P. Vandergert, "Impact of Public Charging Infrastructure on the Adoption of Electric Vehicles," University of East London, 2020. |
| [2] | E. Kontou, C. Liu, F. Xie, X. Wu and Z. Lin, "Understanding the Linkage between Electric Vehicle Charging Network Coverage and Charging Opportunity Using GPS Travel Data," 2019. |
| [3] | B. Haidar and M. T. A. Rojas, "The relationship between public charging infrastructure deployment and other socio-economic factors and electric vehicle adoption in France," *Research in Transportation Economics, DOI: 10.1016/j.retrec.2022.101208.,* vol. 95, 2022. |
| [4] | S. A. Funke, F. Sprei, T. Gnann and P. Plötz, "How much charging infrastructure do electric vehicles need? A review of the evidence and international comparison," *Transportation Research Part D: Transport and Environment,* vol. 77, pp. 224-242, 2019. |
| [5] | T. Franke and J. F. Krems, "Understanding charging behaviour of electric vehicle users," *Transportation Research Part F: Traffic Psychology and Behaviour,* vol. 21, pp. 75-89, 2013. |
| [6] | W. Sierzchula, S. Bakker, K. Maat and B. van Wee, "The influence of financial incentives and other socio-economic factors on electric vehicle adoption," *Energy Policy,* vol. 68, pp. 183-194, 2014. |
| [7] | Y. Zhang, Y. Yu and B. Zou, "Analyzing public awareness and acceptance of alternative fuel vehicles in China: The case of EV," *Energy Policy,* vol. 39, no. 11, pp. 7015-7024, 2011. |
| [8] | R. Diestel, Graph Theory, Springer, 2010. |
| [9] | T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, Introduction to algorithms, MIT Press, 2009. |
| [10] | M. Newman, Networks: An Introduction, Oxford University Press, 2010. |
| [11] | H. Holland, Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence, MIT Press Direct, 1992. |
| [12] | D. E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Longman Publishing Co., Inc., 1989. |
| [13] | K. De Jong, "An Analysis of the Behavior of a Class of Genetic Adaptive Systems," Department of Computer and Communication Sciences, University of Michigan, 1975. |
| [14] | M. Mitchell, An Introduction to Genetic Algorithms, MIT Press, 1998. |
| [15] | A. E. Eiben and J. E. Smith, Introduction to Evolutionary Computing, Springer, 2003. |
| [16] | K. Deb, Multiobjective Optimization Using Evolutionary Algorithms, Jhon Wiley and Sons Ltd, 2001. |
| [17] | B. L. Miller and D. E. Goldberg, "Genetic Algorithms, Tournament Selection and the Effects of Noise," *Complex Symstems,* vol. 9, pp. 193-212, 1995. |
| [18] | J. E. Baker, "Adaptive Selection Methods for Genetic Algorithms," in *Proceedings of the 1st International Conference on Genetic Algorithms*, 1985. |
| [19] | A. H. Wright, "Genetic Algorithms for Real Parameter Optimization," *Foundations of Genetic Algorithms,* vol. 1, pp. 205-2018, 1991. |
| [20] | K. Deb and R. B. Agrawal, "Simulated Binary Crossover for Continuous Search Space," *Complex Systems,* vol. 9, no. 2, pp. 115-148, 1995. |
| [21] | Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, Springer, 1996. |
| [22] | A. E. Smith and D. W. Coit, Constraint-Handling Techniques - Penalty Functions, Handbook of Evolutionary Computation, Institute of Physics Publishing and Oxford, 1997. |
| [23] | J. C. Bean, "Genetic Algorithms and Random Keys for Sequencing and Optimization," *ORSA Journal on Computing,* vol. 6, no. 2, 1994. |
| [24] | X. Yao, Y. Liu and G. Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation, DOI: 10.1109/4235.771163,* vol. 3, no. 2, pp. 82-102, 1999. |
| [25] | D. E. Goldberg and S. Voessner, "Optimizing global-local search hybrids," in *GSECCO'99*, 1999. |
| [26] | E. Cantú-Paz, Efficient and Accurate Parallel Genetic Algorithms, Springer, 2001. |

**Insurance**

I certify that I have written the above thesis independently and have not used any outside help. All passages taken verbatim or in spirit from published or unpublished literature have been marked as such.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name

**Blocking notice**

Inspection of the thesis "*Title*" is not permitted. Exceptions to this are the supervising lecturers and the authorised members of the examination board.

Publication and reproduction of the work - even in excerpts - is not permitted up to and including xy.