UNIVERSIDAD DE INVESTIGACIÓN DE TECNOLOGÍA EXPERIMENTAL YACHAY

Escuela de Ciencias Matemáticas y Computacionales

Optimal location of the electric vehicle charging stands using simulation and genetic algorithms.

Trabajo de integración curricular presentado como requisito para la obtención del título de Ingeniero en Tecnologías de la Información

Autor:
Utreras Tobar Héctor Ronny

Tutor:
PhD. Armas Andrade Tito Rolando

Urcuquí, julio de 2023

AUTORÍA

Yo, **UTRERAS TOBAR HÉCTOR RONNY**, con cédula de identidad 0401471776, declaro que las ideas, juicios, valoraciones, interpretaciones, consultas bibliográficas, definiciones y conceptualizaciones expuestas en el presente trabajo; así cómo, los procedimientos y herramientas utilizadas en la investigación, son de absoluta responsabilidad del autor del trabajo de integración curricular. Así mismo, me acojo a los reglamentos internos de la Universidad de Investigación de Tecnología Experimental Yachay.

Urcuquí, julio de 2023.

_____

Héctor Ronny Utreras Tobar
CI: 0401471776

AUTORIZACIÓN DE PUBLICACIÓN

Yo, **UTRERAS TOBAR HÉCTOR RONNY,** con cédula de identidad 0401471776, cedo a la Universidad de Investigación de Tecnología Experimental Yachay, los derechos de publicación de la presente obra, sin que deba haber un reconocimiento económico por este concepto. Declaro además que el texto del presente trabajo de titulación no podrá ser cedido a ninguna empresa editorial para su publicación u otros fines, sin contar previamente con la autorización escrita de la Universidad.

Asimismo, autorizo a la Universidad que realice la digitalización y publicación de este trabajo de integración curricular en el repositorio virtual, de conformidad a lo dispuesto en el Art. 144 de la Ley Orgánica de Educación Superior

Urcuquí, julio de 2023.

Firmado electrónicamente por:
HECTOR RONNY
UTRERAS TOBAR

_____

Héctor Ronny Utreras Tobar
CI: 0401471776

Dedication

To my beloved mother Betty, my dear grandmother Clara, my brother Joel, and all my family, especially my father Héctor who is no longer with us. For their love, support, and unwavering belief in me throughout my academic journey. Their sacrifices, wisdom, and encouragement have been a driving force in my pursuit of knowledge and personal growth. I am forever grateful for their guidance and hope to make them proud with my accomplishments.

Héctor Ronny Utreras Tobar

Acknowledgment

I would like to take this opportunity to express my sincere gratitude and appreciation to my tutor, Rolando, for his patience, dedication, and guidance throughout my thesis research. His expertise and knowledge in the field of research have been invaluable, and without his support, this thesis would not have been possible.

I want to thank the Salesian community of Machala for their support, companionship, affection, and for providing me with the opportunity to have the wonderful experience as a Salesian volunteer while also having the space to develop my thesis.

Finally, I would like to acknowledge my beloved university, Yachay Tech, for providing me with an environment that fosters learning and personal growth. The resources and facilities provided by the university have been instrumental in the completion of this thesis, and I am grateful for the opportunities that have been presented to me.

Héctor Ronny Utreras Tobar

Resumen

La presente tesis está enfocada en optimizar la localización de estaciones de carga para vehículos eléctricos (EVs) dentro de un escenario determinado, en este caso la ciudad de Cuenca. Se ha abordado la problemática de la contaminación ambiental como factor fundamental que lleva a los ciudadanos a optar por opciones más amigables. Sin embargo, la falta de infraestructura constituye uno de los mayores impedimentos para su implementación.

Este estudio utiliza algoritmos evolutivos multiobjetivo (MOEAs) para optimizar el tiempo de viaje, la cantidad de estaciones de carga y la calidad de servicio. Se diseñó una interfaz que nos permita la interacción entre un simulador de transporte (MATSim) y un framework evolutivo (DEAP). A través de MATSim, hemos podido configurar el escenario, los planes de movilidad, y el movimiento de los agentes en la red de vial. Mediante DEAP configuramos el algoritmo genético, los individuos, la población, los parámetros y operadores. En este estudio, se identificaron 20 posibles ubicaciones para las estaciones de carga, que se codificaron como variables de decisión.

Después de ejecutar el simulador, se obtuvo un conjunto de soluciones óptimas a través del proceso evolutivo NSGA-II. Se graficó el frente de Pareto para elegir las mejores soluciones, enfocándose principalmente en el objetivo del número de estaciones. Las mejores configuraciones se mapearon sobre la red vial de Cuenca y se realizaron análisis de hipervolumen y correlación entre los objetivos. Concluimos que la interfaz permite obtener un conjunto de soluciones óptimas mediante la interacción entre MATSim y DEAP. Los métodos de análisis, como el hipervolumen y la correlación de objetivos, ayudaron a evaluar la calidad de las soluciones y la correlación cuantitativa de los objetivos.

Palabras Clave: Estaciones de carga para vehículos eléctricos, algoritmos evolutivos multiobjetivo, MATSim, DEAP, NSGA-II Frente de Pareto.

Abstract

This thesis aims to optimize the location of electric vehicle (EV) charging stations within Cuenca city. The study addresses environmental pollution as a fundamental factor that leads citizens to opt for more environmentally friendly options. However, the lack of infrastructure constitutes one of the major obstacles to its implementation.

This study uses multi-objective evolutionary algorithms (MOEAs) to optimize the travel time, number of charging stations, and quality of service. We created an interface that allows the interaction between a transportation simulator (MATSim) and the evolutionary framework (DEAP). Using MATSim, we configured the transportation scenario in Cuenca, including the loading of mobility plans and the movement of agents through the road station placement. With the DEAP framework, we could configure the genetic algorithm with individuals, population, parameters, and operators. The study identified 20 potential locations for charging stations and coded them as decision variables to be optimized by the algorithm.

After running the simulator, we obtained a set of optimal solutions through the NSGA-II evolutionary process. We graphed the Pareto front to select the best solutions, focusing primarily on the objective of the number of stations. Finally, we mapped the best configurations onto Cuenca's road station placement and performed analyses of hypervolume and correlation between objectives. The study conclude that the interface allows obtaining a set of optimal solutions through the interaction between MATSim and DEAP. We evaluated the quality of the solutions and analyzed the quantitative relationship between objectives using analytical methods such as hypervolume and objective correlation.

Key Words: Electric vehicle charging stations, multi-objective evolutionary algorithms, MATSim, DEAP, NSGA-II, Pareto Front.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background

The increasing demand for sustainable transportation has led to the development of electric vehicles (EVs) as a more environmentally friendly alternative to traditional gasoline-powered cars. Public and private institutions are currently seeking solutions to reduce the use of fossil fuels and thus mitigate CO2 emissions. The automobile industry is introducing more proposals to the market that incorporate the use of clean energy. Cities of all countries are beginning to focus on the construction of infrastructure to enable the change from fuels-based mobility to more ecological proposals. The change brings several considerations for its implementation, such as location, infrastructure, costs, demand, etc.

This research aims to determine the optimal charging stations infrastructure in the city of Cuenca using a multi-objective optimization approach. By integrating the traffic simulator MATSim [14], the evolutionary framework DEAP [15], and data exploration methods, we will identify the best solutions for conflicting objectives such as travel time (tt), number of charging stations (N_{st}), and quality of service (QoS). Specifically, we will use the Non-dominated Sorting Genetic Algorithm-II (NSGA-II) as the Multi-Objective Evolutionary Algorithm (MOEA) to obtain the best individuals through the evolutionary process.

The scenario selected for this study is the city of Cuenca, the third-largest city in Ecuador, which has taken initiatives in implementing ecological technologies , including the proposal to use electric transportation [16]. With a current population of 417632,

Cuenca has a well-designed road network and ample spaces for experimenting with the traffic simulator. This research aims to obtain solutions for a given scenario rather than conducting a comprehensive study of mobility in Cuenca. The transportation simulator implements a combination of artificial and real-world data. Artificial data includes parameters such as the number type of vehicles within the city and charging capacity. In contrast, real-world data includes aspects such as the road network, peak traffic hours, suitable spaces, and areas of highest human activity. By integrating these two data types, Cuenca's electric vehicle charging infrastructure simulation is more comprehensive and accurate.

The interdisciplinary study combines transportation engineering, mathematical optimization, data, and computer sciences. First, transportation engineering is essential in understanding the city's road network, traffic flow, and mobility patterns. Second, mathematical optimization is crucial in developing and applying algorithms to find the best solutions based on multiple conflicting objectives, such as travel time, number of charging stations, and quality of service. Third, data science is necessary to collect and analyze the data and apply exploratory methods to interpret the optimization process results. Finally, computer science is crucial in developing and implementing software tools to integrate and process the data, run simulations, and perform the necessary calculations for the optimization algorithms. Combining these fields allows for a comprehensive approach to solving the problem of limited charging infrastructure for electric vehicles in a given scenario city.

1.2 Problem statement

Implementing new technology always poses significant challenges for authorities and citizens, particularly when changing a traditional mobility model. As the population seeks solutions to climate change and the greenhouse effect caused by CO2 emissions, electric car technology has emerged as a popular solution that offers many advantages. However, a significant challenge to the widespread adoption of Electric Vehicles (EVs) is the lack of sufficient charging infrastructure, limiting their range and usability and reducing their overall appeal to potential consumers.In many countries, including Ecuador, the lack of charging stations is a major barrier to adopting electric vehicles.

The problem this research address is finding the optimal network of charging stations for electric vehicles, which is a multi-objective optimization problem. Achieving the balance between minimizing travel time, optimizing the number and capacity of stations, and maximizing the quality of service for drivers is challenging due to the trade-off between these objectives. Finding the optimal solution requires a thorough understanding of the transportation system, the integration of multiple data sources and algorithms, and expertise in transportation engineering, electrical engineering, computer sciences, and data analytics

The solution to this mobility problem is important because it can significantly impact the adoption and usage of electric vehicles, which is a key factor in reducing greenhouse gas emissions and mitigating the impacts of climate change. Therefore, ensuring that charging stations are located in convenient and easily accessible locations with sufficient capacity is essential for encouraging the widespread use of electric vehicles. Additionally, optimizing the number of charging stations and their capacity can help to reduce the costs associated with building and maintaining the charging infrastructure, which is another important factor in promoting the widespread use of electric vehicles.

Finding the optimal solution for this electric mobility problem is a complex and multi-faceted issue that has yet to be fully addressed by previous studies. There are several reasons why previous authors may not have fully addressed the problem of finding the optimal network of charging stations for electric vehicles. Firstly, electric vehicle technology is a relatively new phenomenon, and the demand for charging infrastructure is still emerging. As such, there may not have been enough data to understand the problem's complexity fully. Secondly, the problem involves multiple objectives and trade-offs, and finding a single optimal solution is challenging. Finally, the problem is affected by uncertainties such as fluctuations in demand for charging services and the evolving adoption rate of electric vehicles, making it challenging to predict and accommodate the charging needs of electric vehicle users. Therefore, solving the problem of finding the optimal network of charging stations requires a comprehensive and interdisciplinary approach.

An integrated approach using a traffic simulator, evolutionary framework, and data exploration methods can identify the optimal location and capacity of charging stations to meet the demands of electric vehicle drivers while minimizing the costs of building and

maintaining the charging infrastructure, resulting in a more efficient and effective electric vehicle charging infrastructure in the city of Cuenca.

1.3 Objectives

1.3.1 General Objective

This research aims to develop a comprehensive solution for determining the optimal location and capacity of electric vehicle charging stations in the Cuenca scenario, considering the interdisciplinary nature of the problem and the trade-off between different objectives. By integrating the DEAP evolutionary framework with the MATSim traffic simulator, the study seeks to provide valuable insights for decision-makers and stakeholders in the development of electric vehicle charging infrastructure.

1.3.2 Specific Objectives

1. Develop an interface that integrates the MATSim transportation simulator with the DEAP evolutionary framework to optimize the placement of electric vehicle charging stations.

2. Utilize genetic algorithms within the DEAP framework to perform multi-objective optimization, balancing conflicting objectives such as travel time, the number of charging stations, and service quality.

3. Simulate the electric vehicle charging infrastructure using MATSim, considering factors such as the number and types of vehicles in Cuenca, the road network, mobility plans, peak traffic hours, and areas of high human activity.

4. Analyze the simulation results to evaluate the performance of the electric vehicle charging infrastructure and provide recommendations for future improvements and implementations.

Chapter 2

Theoretical Framework

2.1 Electromobility and Sustainable Transportation

The transportation sector significantly contributes to greenhouse gas emissions, negatively impacting public health and the environment. The global shift towards urbanization has increased the demand for transportation and its associated energy consumption. In recent years, there has been growing interest in developing and adopting electric vehicles (EVs) to reduce emissions and improve transportation sustainability.

Acording to Baker, Chon, and Keoleian, the development of electric vehicles has been driven by environmental concerns, advances in battery technology, and government policies to encourage their adoption. The ecological impacts of the transportation sector have been well documented, particularly regarding greenhouse gas emissions. EVs have the potential to significantly reduce CO2 emissions, particularly if powered by renewable energy sources [17].

According to International Energy Agency, the adoption of electric vehicles is expected to increase in the coming years due to technological advancements, government regulations, and consumer demand. This growth is expected to significantly reduce greenhouse gas emissions from the transportation sector [5]. However, despite their potential, electric vehicles face several challenges to widespread adoption.

According to the International Energy Agency's (IEA) Global EV Outlook 2021, electric vehicles in use in 2020 saved 54 million tons of CO2 emissions, compared to CO2 emissions by combustion engines [18]. In 2021, the sales of electric vehicles have reached an all-time

high globally, as stated in a report by BloombergNEF, with an estimated 5.1 million electric cars sold worldwide, representing 7.5% of total passenger car sales. This means a notable surge from the previous year, where electric cars made up only 4.2% of the total sales of passenger vehicles. The report from BloombergNEF also forecasts a further increase in electric car sales in the upcoming years, with a projection of 31% of total passenger car sales being electric by 2040 [19]. As the Figure 2.1 shows, the number of electric vehicle sales has increased significantly over the past decade, with a notable acceleration in growth in recent years. As a result, the percentage of total vehicle sales represented by electric vehicles has also increased steadily, reaching over 5% in 2021. Governments and organizations worldwide are increasingly adopting policies and incentives to encourage the adoption of electric vehicles, which is expected to continue in the coming years.

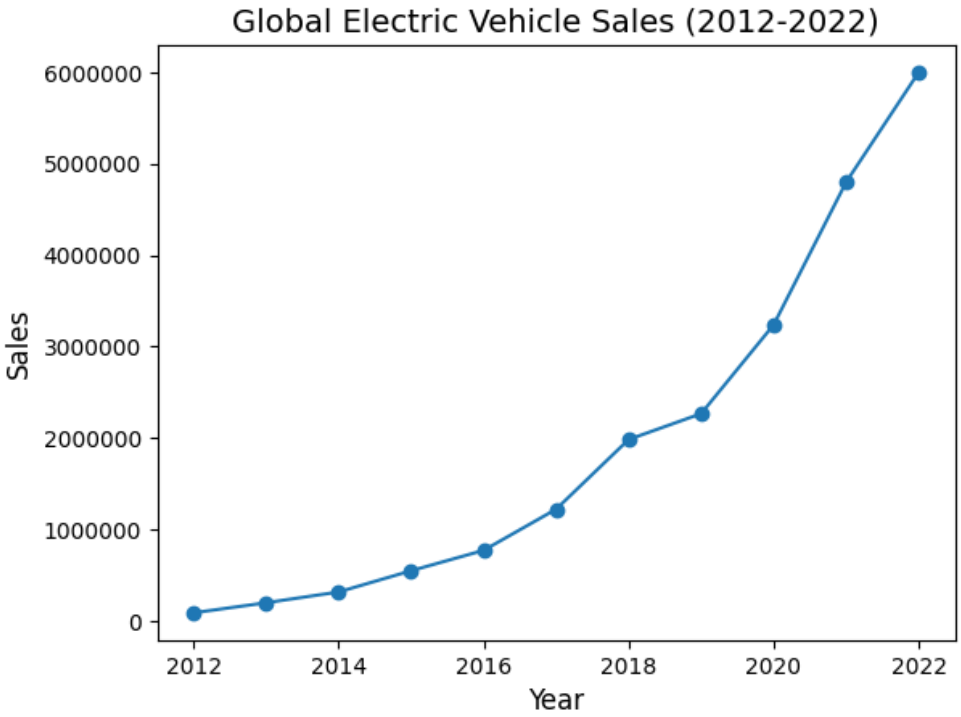


Figure 2.1: EV sales growth over the past decade [5]

2.1.1 Overview of Electromobility

Electric vehicles (EVs) use electric motors for propulsion, rather than internal combustion engines (ICEs) that burn gasoline or diesel fuel. According to Kurani and Turrentine, EVs can be powered by grid electricity or on-board batteries, which are charged by regenerative

braking or external charging stations.

There are three main types of electric vehicles: battery electric vehicles (BEVs), plug-in hybrid electric vehicles (PHEVs), and hybrid electric vehicles (HEVs) [20]. Battery electric vehicles (BEVs) are fully electric vehicles powered exclusively by electricity stored in on-board batteries. Lu et al., describes that BEVs have the highest driving range of all-electric vehicles and are the most energy-efficient type of electric vehicle. On the other hand, plug-in hybrid electric vehicles (PHEVs) combine a gasoline or diesel engine with an electric motor and battery[21]. PHEVs can run on electricity alone, gasoline or diesel alone, or a combination of both, and have an electric driving range shorter than BEVs. Hybrid electric vehicles (HEVs) use both a gasoline or diesel engine and an electric motor, but the electric motor is not powerful enough to propel the vehicle alone [20]. Figure 2.2 shows different types of EVs and the ICEs.

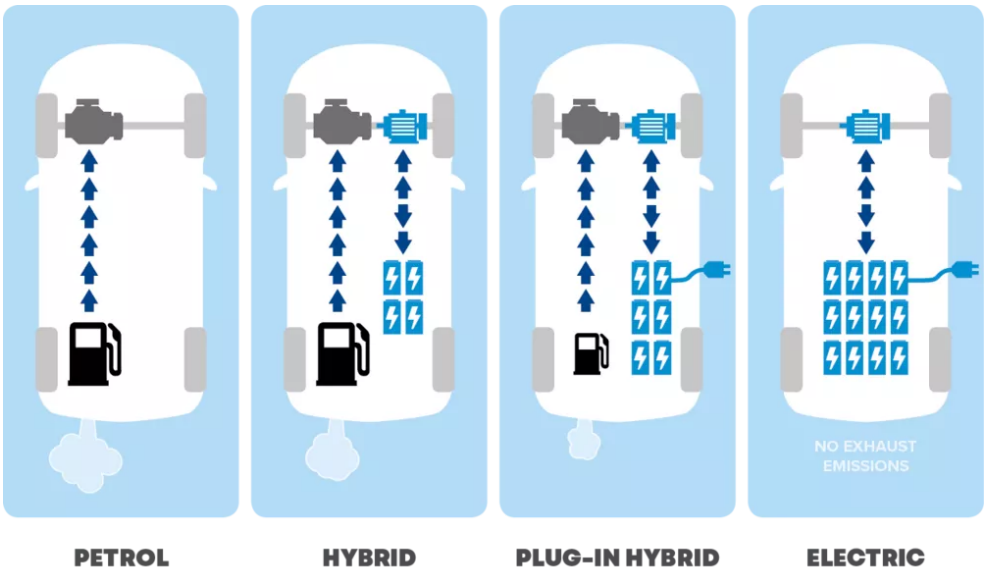


Figure 2.2: Different types of engines and mechanisms. Obtained from [6]

In Zhang et al research, they describe that EVs have several advantages over conventional vehicles, including lower operating costs, reduced greenhouse gas emissions, and improved energy efficiency [22]. For example, EVs have a higher efficiency rate than internal combustion engines, which waste a large amount of energy as heat [21]. In addition, EVs produce no tailpipe emissions, reducing local air pollution and improving public health [17]. Furthermore, using EVs powered by renewable energy sources can significantly reduce greenhouse gas emissions and contribute to the decarbonization of the transportation

sector [5].

Sierzchula et al., conclude that EVs face several challenges to widespread adoption. One of the biggest challenges is the high upfront cost of EVs compared to conventional vehicles. In addition, the limited driving range of EVs and the lack of charging infrastructure are barriers to adoption. Public charging stations are not yet widespread, making it difficult for EV owners to find a place to charge their vehicles while away from home [23].

2.1.2 Advantages of Electromobility

Electromobility has several advantages over conventional vehicles, both for individual users and for society as a whole. One of the primary advantages of electromobility is that it can reduce greenhouse gas emissions and improve air quality and noise reduction. EVs produce zero tailpipe emissions, which can help to reduce local air pollution and improve public health [17]. In addition, using renewable energy to power EVs can significantly reduce greenhouse gas emissions, which is critical for addressing the global climate crisis [5].

Another advantage of electromobility is increased energy efficiency. EVs are more efficient than conventional vehicles, as they convert a higher percentage of energy from the grid into vehicle propulsion [21]. Traditional vehicles waste a large amount of energy as heat, which is not the case for EVs. In addition, greater efficiency means that EVs can travel farther on a single charge, making them more convenient for long-distance travel. Moreover, electric vehicles are quieter and provide a smoother driving experience than conventional vehicles. Electric motors have fewer moving parts than ICEs, resulting in less vibration and noise [17]. The smoother driving experience can also improve passenger comfort and reduce motion sickness, making EVs more appealing to passengers.

Furthermore, electromobility can help to reduce dependence on fossil fuels and increase energy security. The transportation sector is heavily dependent on petroleum-based fuels, which are subject to price fluctuations and supply chain disruptions [5]. Using renewable energy sources to power EVs can reduce dependence on fossil fuels and increase energy security. Finally, electric vehicles can provide cost savings over the vehicle's lifetime. Although the upfront cost of EVs is higher than conventional vehicles, EVs have lower operating prices, particularly regarding fuel and maintenance [23]. In addition, electric motors require less maintenance than ICEs, as they have fewer moving parts and do not require oil

changes, making EVs more cost-effective in the long run.

2.1.3 Challenges of Electromobility

Despite the various advantages of electromobility, it is necessary to address several challenges to its widespread adoption. According to Hidrue et al., one of the primary challenges of electromobility is the range limitation of electric vehicles. Although battery technology has improved significantly in recent years, electric vehicles still have a limited driving range compared to conventional vehicles [24]. As a result, range anxiety, or the fear of running out of battery charge before reaching the destination, is a significant barrier to EV adoption [25].

According to Acheampong et al., another challenge of electromobility is the lack of charging infrastructure. Without a sufficient number of charging stations, EV drivers may experience difficulties in finding a place to charge their vehicles, particularly during long-distance trips [26]. Potential EV buyers may be discouraged from switching to electric due to the inadequate charging infrastructure.

Furthermore, the upfront cost of electric vehicles is still higher than that of conventional vehicles, although the cost decreases as battery technology improves and economies of scale are achieved [21]. However, the high initial cost remains a significant barrier to EV adoption for many consumers, particularly in low- and middle-income countries.

In addition, the production and disposal of EV batteries can have negative environmental impacts. Battery production involves extracting raw materials, which can result in environmental degradation and human rights abuses[27]. Furthermore, at the end of their life, EV batteries must be disposed of or recycled, which can also have environmental and social consequences [22].

Finally, stakeholders must address several regulatory and policy challenges to facilitate the widespread adoption of electromobility. These include issues related to charging infrastructure, battery disposal, and financial incentives for EV adoption [25]. Addressing these challenges will be critical for the widespread adoption of electromobility and the realization of its potential benefits.

2.1.4 Role of Electromobility in Sustainable Transportation

The role of electromobility in sustainable transportation has continued to gain attention and interest from researchers and policymakers in recent years. As a result, electromobility has been identified as a potential solution to mitigate transportation's environmental impact and enhance urban population mobility.

In a study by Liu et al., the authors evaluated the environmental impact of electric vehicles (EVs) and internal combustion engine vehicles (ICEVs) in China. The results showed that EVs could significantly reduce carbon dioxide (CO2) emissions and improve air quality, particularly in urban areas [28].

Electromobility can also enhance the quality of life in cities by reducing traffic congestion and improving the accessibility of transportation. According to a study by Gao et al., the adoption of EVs in Beijing could significantly reduce traffic congestion and travel time, thereby enhancing the city's livability [29]. Moreover, the role of electromobility in sustainable transportation extends beyond the environment and quality of life. Electromobility can also contribute to energy security and the growth of the renewable energy sector. In a study by Li et al., the authors evaluated the impact of EV adoption on renewable energy integration in China. The results showed that EVs could serve as a significant source of flexibility in the power system, enhancing renewable energy integration and energy security [30].

However, the widespread adoption of electromobility faces several challenges, including high costs, range anxiety, and limited charging infrastructure. According to a study by Wang et al., the lack of charging infrastructure is one of China's most significant barriers to EV adoption. The authors recommended the development of public charging infrastructure to enhance the adoption of EVs in the country[31].

In summary, the role of electromobility in sustainable transportation is critical in mitigating the environmental impact of transportation, enhancing energy security, and improving the quality of life in urban areas. However, addressing the challenges facing the adoption of electromobility is critical for its widespread adoption. Researchers and policymakers must collaborate to develop and deploy innovative solutions that address electromobility's challenges.

2.2 Evolutionary Algorithms

Evolutionary Algorithms (EAs) encompasses a variety of optimization techniques inspired by natural evolution called Evolutionary Algorithms. Evolutionary algorithms are a class of optimization algorithms that use the principles of natural evolution to solve complex problems. They iteratively generate and evaluate a population of candidate solutions and use selection, crossover, and mutation operators to create new candidate solutions. EAs can be used to solve optimization problems with multiple objectives, constraints, and variables. [4]. The following are some common types of evolutionary algorithms:

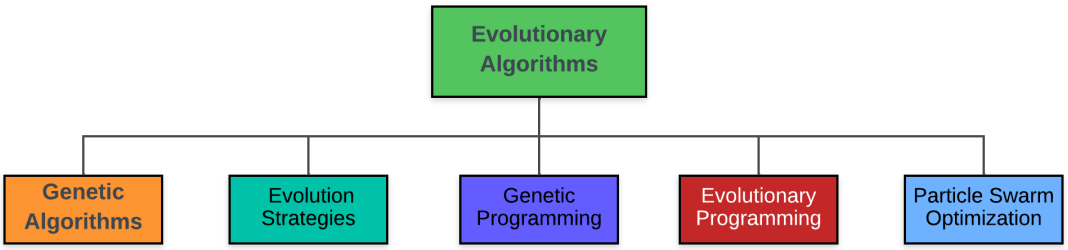


Figure 2.3: Classification of Evolutionary Algorithms

- Genetic Algorithms (GA): Mechanisms inspired by natural selection, crossover, and mutation are used to evolve a population of candidate solutions. They are often used to solve optimization problems involving binary or real-valued variables. GAs solve single objective problems and also Multiobjective problems (MOEA).
- Evolution Strategies (ES): A variant of genetic algorithms that works with continuous variables. Mutation and selection are used to evolve a population of candidate solutions toward the optimum. ES algorithms typically use a self-adaptive approach to adjust the mutation rate based on the population's fitness.
- Genetic Programming (GP): A population of computer programs is evolved to solve a problem using the principles of genetic algorithms. A tree-like structure is used to represent the solution space, and genetic operators are used to evolve the tree toward an optimal solution.
- Evolutionary Programming (EP): A variant of genetic algorithms that uses mutation

and selection to optimize a population of candidate solutions. However, the mutation operator is used to exploit the solution space rather than explore it.

- Particle Swarm Optimization (PSO): A metaheuristic optimization algorithm inspired by the movement of swarms in nature. Candidate solutions are represented by particles that move through the solution space. The movement of particles is guided by their own best-known position and the best-known position of the swarm.

In the upcoming sections, we will focus on Genetic Algorithms (GAs) as they are one of the main aims of this research. GAs are one of the most widely used techniques in the field of evolutionary computation, and have proven to be effective in solving complex optimization problems. They are inspired by the natural process of evolution and use a combination of selection, crossover, and mutation to generate a population of candidate solutions, which are then iteratively improved until an optimal or near-optimal solution is found. Understanding GAs is crucial for the success of this thesis, as they will be used to solve multi-objective optimization problem.

2.3 Genetic Algorithms

Charles Darwin developed his theory of natural evolution in his famous book "Origin of Species". According to Sivanandam and Deepa [32], Darwin describes that specific laws of survival govern every living being. Species have disappeared throughout history because they did not know how to adapt, referring to "the survival of the fittest." The efficiency and perfection in swimming of dolphins and sharks, the specialization of finch beaks, the color of the polar bear's fur, etc., are examples of a random evolution that preserves the best characteristics of each generation to have the "fittest individual."

Since the 1950s, researchers have used Darwin's evolutionary theory applied to search and optimization algorithms. Holland, in his book "Adaptation in Natural and Artificial Systems," describes for the first time how to use natural evolution to optimization problems making the first Genetic Algorithm [32]. Nowadays, Holland's theory has been developed, and GAs startup as a powerful and efficient tool for solving search and optimization problems.

2.3.1 Genetic Algorithms Definition

As defines Thanura et al., Genetic Algorithms (GAs) are heuristic search algorithms that equal other algorithms (neural networks, ant colony algorithm) that try to replicate natural evolution to find near-optimum solutions to complex problems. Genetic Algorithms take the properties of "natural evolution" and "survival of the fittest." [33] Genetic Algorithms (GAs) are a popular type of evolutionary algorithms that have been used in a wide range of optimization problems. According to Deb et al. (2019), GAs are a subset of the broader evolutionary computation field that uses genetic operators, including selection, crossover, and mutation, to evolve a population of candidate solutions. These operators are inspired by natural selection principles and genetics and are used to simulate the process of evolution. In GA, the population evolves over a number of generations, with fitter individuals being selected for reproduction and recombination and less fit individuals being replaced by new offspring [12, 33].

2.3.2 Stages of Genetic Algorithms

The classical genetic algorithm is based primarily on starting with a set of random candidate solutions not optimized that represent a solution to the optimization problem that we want to solve. When these solutions pass through the genetic algorithm, candidate solutions with better characteristics begin to be generated [1]. A solution is a potential candidate when it achieves the best results when evaluated with the fitness function. Solution or set of solutions can be maximum or minimum. The representation of the solutions plays an essential role since it determines the type of genetic operators used. In general, as described by [32], the solutions' representation can be scalar or a bit strings. The coding of the solution, which is subject to the evolutionary process, is called the genotype or chromosome. Algorithm 1 indicates the basic genetic algorithm's pseudocode, which can be a model for problems of different approaches [1].

In Natural Selection some specific properties or characteristics survive over time. Genetic algorithms take advantage of this fact and make an analogy with the natural evolution. As defines [34], GA takes the inspiration of the natural selection mechanism, where stronger individuals are likely to be the winners in a competing environment.

Algorithm 1 Basic Genetic Algorithm

```
Starts with initial population P(0).
For each solution i in the previous step P(t) do
    Evaluate i fitness
End for
While the stop criteria not reached do
    Selection of the parents solutions or chromosomes for crossover
    In the selected chromosomes of population P(t) apply crossover operator
    the mutation operator is applied to new population P(t)
        For each individual i in the new population P(t) do
            Test the individual fitness
        End for
End while
```

Table 2.1: Genetic Algorithm Pseudocode. Based on [1], and [2]

GA aims to obtain a potential solution as an individual representing a vector. Genetic evolution begins from a population of chromosomes; some fitter chromosomes tend to produce good quality offspring; with this process, the better solutions are obtained.

A genetic algorithm (GA) is a type of optimization algorithm inspired by the process of natural selection. According to Jiang et al., GAs typically consist of several phases:

1. **Initialization:** In this phase, a population of candidate solutions is randomly generated as the initial set of potential solutions. This phase is critical because the quality of the initial population can significantly impact the optimization process [35].

2. **Selection:** In this phase, a subset of the population is selected for further processing based on their fitness, which is evaluated by a fitness function. Various selection methods have been proposed in recent years, such as tournament selection and roulette wheel selection [36].

3. **Crossover:** In this phase, pairs of selected individuals are combined to produce new offspring. The crossover operation involves exchanging parts of the genetic information between parents to create new candidate solutions. There are many crossover techniques, such as single-point crossover and uniform crossover [37].

4. **Mutation:** In this phase, some individuals are randomly modified to introduce new genetic information into the population. The mutation operation can help to prevent the population from converging prematurely to suboptimal solutions [38].

5. **Replacement:** In this phase, some individuals in the current population are replaced by the newly generated offspring based on a replacement strategy. For example, the worst-performing individuals in the current population can be replaced by the best-performing offspring [39].

6. **Termination:** The algorithm terminates when a stopping criterion is met. The criterion can be a maximum number of generations or a minimum fitness level achieved by the population [40].

2.3.3 Applications of Genetic Algorithms

Genetic algorithms (GAs) have been successfully applied in various fields, including engineering design, machine learning, scheduling, transportation, and optimization. In engineering design, GAs have been used to optimize the design of structures and components, such as trusses, beams, and frames [40]. In machine learning, GAs have been used to train neural networks and to select features for data classification, and clustering [41]. In scheduling, GAs have been used to optimize the scheduling of tasks in manufacturing systems and transportation networks [42]. GAs have been used in transportation to optimize vehicle routing, fleet management, design transportation networks, and traffic signal timings [43]. Finally, in optimization, GAs have been used to solve complex optimization problems, such as the traveling salesman problem and the knapsack problem [44]. By leveraging the strengths of GAs in exploring large solution spaces and optimizing complex objectives, these applications have demonstrated the potential of GAs as a powerful optimization tool in many fields.

2.3.4 Biological and GAs terminology

GAs operate on potential solutions populations, where each solution is represented by a chromosome that encodes the values of the problem variables. The concepts of chromosome, gene, allele, locus, genotype, phenotype, and epistasis are central to the genetic algorithm approach. Table 2.2 compares the genetic concepts of chromosome, gene, allele, locus, genotype, phenotype, and epistasis and their computational counterparts in genetic algorithms.

Concept	Definition in Nature	Computational Equivalent in GA.
Chromosome	A thread-like structure made up of DNA that carries genetic information	A solution or candidate solution to a problem represented by a string of values, such as binary digits or real numbers.
Gene	A segment of DNA that encodes a specific trait or function	A subset of the chromosome that represents a specific characteristic or variable of the solution.
Allele	One of the possible forms of a gene that can occur at a specific location	One of the possible values that a gene can take, representing a variation of the trait encoded by the gene.
Locus	The specific location on a chromosome where a gene is located	The index or position of a gene within the chromosome.
Genotype	The genetic makeup of an individual, including all of its alleles	The complete set of genes and their values in a particular solution.
Phenotype	The observable physical or behavioral characteristics of an individual resulting from its genotype and environment	The result of applying the values of the genes in the genotype to the problem, representing the actual solution.
Epistasis	The interaction between different genes that affects the expression of a particular trait	The influence of one gene on the effect of another gene in the solution, affecting its contribution to the fitness or quality of the solution.

Table 2.2: Comparison of genetic concepts in nature and in genetic algorithms. Based on [3]

2.4 Genotype and Phenotype definitions

The genotype and phenotype are critical components of genetic algorithms (GAs) that can significantly impact their performance. The genotype refers to the genetic information of an individual in the population, typically represented as a bit string or vector. On the other hand, the phenotype refers to the physical expression of the genotype in the problem space. In this section, we will discuss the importance of genotype and phenotype representation in genetic algorithms

2.4.1 Genotype

As the authors in [1] explain, in nature, biologically, one or more chromosomes mix to form a complete genetic structure containing both the physical and operational characteristics of an individual. This grouping of chromosomes is known as the genotype. According to Chen et al., the genotype is "the complete set of genes or variables that define the structure of a solution" [45].

The genotype is a critical component of the genetic algorithm, as it determines the range of possible solutions that can be evaluated and improved through the selection, crossover, and mutation operators. By manipulating the genes or parameters within the genotype, the genetic algorithm can generate new potential solutions and explore the solution space in search of an optimal or near-optimal solution to the problem.

It is worth noting that the specific definition of genotype may vary depending on the context and application of the genetic algorithm, and the terminology used in the literature may differ slightly. However, in general, the genotype is an essential component of the genetic algorithm that represents the potential solutions to the optimization problem being solved.

Chromosome Structure

According to Jain et al., the chromosome is the data structure that represents a potential solution to the optimization problem. It typically consists of a string of genes, where each gene represents a variable or parameter of the solution [46]. The chromosome structure is critical to the performance of the genetic algorithm, as it determines the range of possible solutions that can be generated and evaluated. A well-designed chromosome can ensure that the search space is appropriately defined and that the genetic algorithm can efficiently explore the solution space. Figure 2.4 represents a Choromose with its components previously described in the Table 2.2.

Importance of Correct Chromosome Structure

The chromosome structure is a critical factor in the performance of genetic algorithms. Several authors have emphasized the importance of a correct chromosome structure in

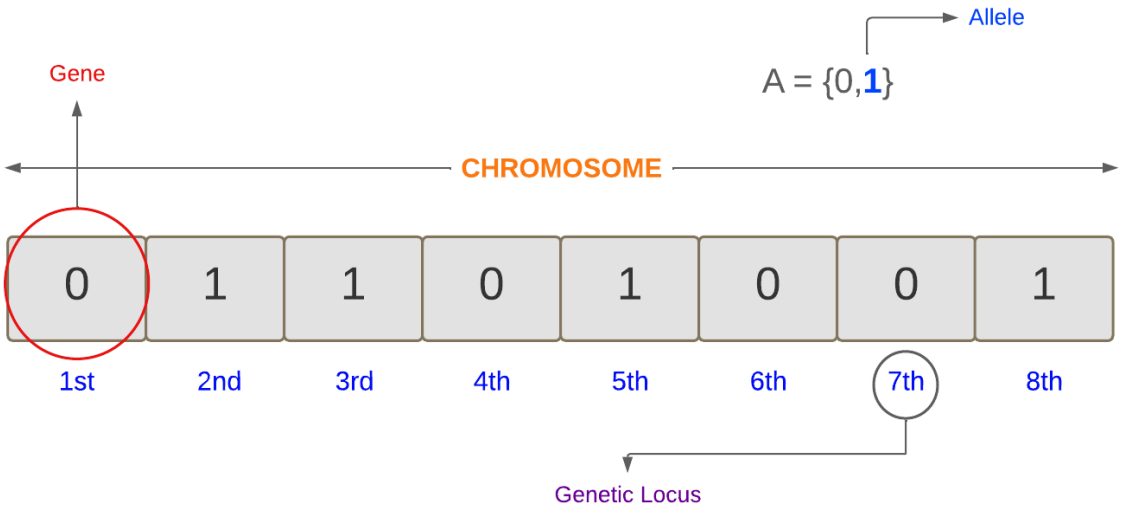


Figure 2.4: Basic Structure of a Chromosome[7]

finding good solutions. For example, Jain et al. state that "the chromosome representation is the most crucial factor in the design of a genetic algorithm," and Singh et al. conclude that "the representation of the chromosome plays a crucial role in determining the performance of the genetic algorithm" [46, 47]. By understanding the importance of the chromosome structure and its interaction with other components of the GA, researchers and practitioners can design effective and efficient genetic algorithms for a wide range of optimization problems

2.4.2 Phenotype

The organism formed by the interaction of the entire known genetic component and the environment in which it inserts and interacts is as a phenotype. According to Forbes, in an artificial intelligence system, the phenotype refers to the alternative solutions produced by the genetic algorithm within the search space. [32].

Phenotype importance

Several authors have emphasized the importance of phenotype in the success of genetic algorithms. For example, Deb notes that "the phenotype represents the true value of the solution to the given problem, and the genotype encodes the information that is necessary

to derive the phenotype" [12].

The phenotype representation determines the range of solutions that can be generated and evaluated by the genetic algorithm. By selecting an appropriate phenotype representation, researchers and practitioners can ensure that the genetic algorithm can efficiently explore the solution space and converge to good solutions [12].

Examples of Genotype and Phenotype representation in real computational problems

1. **Vehicle Routing Problem:** The genotype is typically represented as a sequence of integers that represents the order in which customers are visited, while the phenotype is the actual routing plan that the delivery vehicles will follow [48].

2. **Image Processing:** In the context of image processing, the genotype might be a binary string that represents the parameters for a specific image processing operation, while the phenotype is the transformed image resulting from the application of these parameters [49].

3. **Scheduling Problem:** For the job shop scheduling problem, the genotype could be a permutation encoding that represents the sequence of jobs and their operations, while the phenotype is the actual schedule for each job on each machine [50].

4. **Protein Folding Problem:** In the protein folding problem, the genotype is often represented as a string of amino acids, while the phenotype is the actual 3D structure of the protein [51].

5. **Text Classification:** In the context of text classification, the genotype could be represented as a binary vector that encodes the presence or absence of specific features in a document, while the phenotype is the actual classification label assigned to the document [52].

2.5 Encoding Schemes

The encoding scheme is a critical aspect of genetic algorithms, as it determines the representation of individuals and the operators used to manipulate them [53]. As Figure 2.5

shows, there are different encoding schemes that can be used, including binary, real-valued, permutation, and tree-based encoding. Each encoding scheme has its strengths and weaknesses, and the choice of encoding depends on the problem at hand. The selection of an appropriate encoding scheme can significantly impact the performance of the genetic algorithm[54]. According to Kumar here we have a description of some encoding schemes [8]

1. Binary Encoding: A method of representing variables as a sequence of 1s and 0s, where each bit represents a specific value. This encoding is commonly used for optimization problems that involve decision-making or classification tasks.

2. Real-Valued Encoding: A method of representing variables as real numbers within a specified range. This encoding is useful for problems that involve continuous variables, such as optimization problems involving physical quantities or engineering design parameters.

3. Permutation Encoding: It is a type of genetic algorithm encoding where each chromosome is represented as a string of values, which can be integers, real numbers, characters, or other objects.

4. Tree Encoding: A method of representing variables as a hierarchical structure of nodes and branches. This encoding is useful for problems that involve complex relationships between variables, such as problems involving mathematical expressions or decision trees.

2.6 Selection Methods

According to [3], selection is a crucial component in GAs that helps to determine the most promising individuals in a population and generate offspring for the next generation. This section will delve into the various selection methods utilized in GAs, including their classifications, examples, and mathematical definitions. Two primary selection methods in GAs are fitness-proportional selection and tournament selection [55].

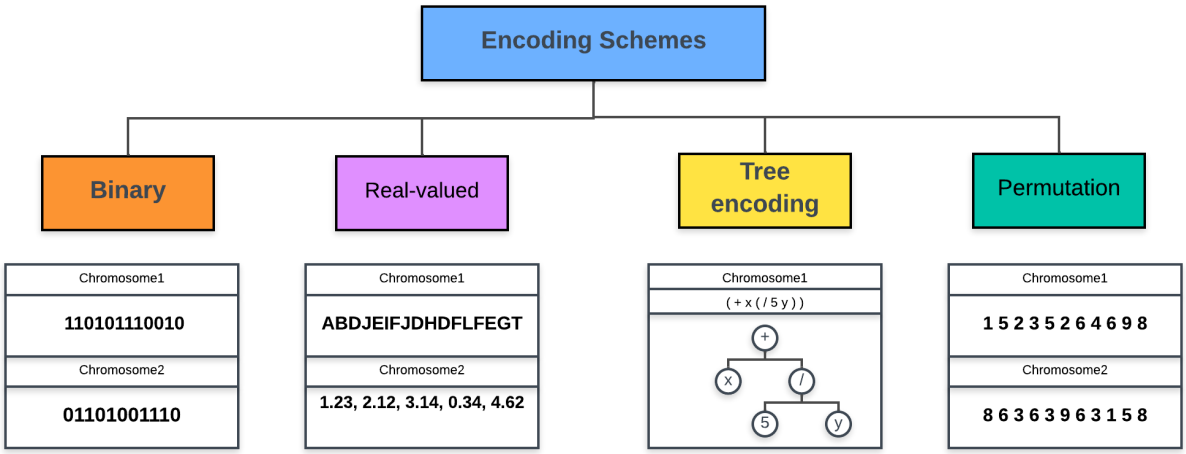


Figure 2.5: Some examples of encoding schemes. Based on [8]

2.6.1 Fitness-Proportional Selection

Fitness-proportional selection methods, also known as roulette-wheel selection or stochastic universal sampling, select individuals for reproduction with a probability proportional to their fitness value. Some of the commonly used fitness-proportional selection methods include [3]:

- Roulette-Wheel Selection: In roulette-wheel selection, each individual is assigned a slice of a roulette wheel that is proportional to its fitness value. A random spin of the wheel then determines which individuals are selected for reproduction.
- Stochastic Universal Sampling: Stochastic universal sampling is similar to roulette-wheel selection but uses multiple points on the wheel to select multiple individuals at once. This approach can be more efficient than roulette-wheel selection but requires more computation.

The fitness-proportional selection method can be mathematically defined as follows:

$$p_i = \frac{f_i}{\sum_{j=1}^{N} f_j} \tag{2.1}$$

where p_i is the probability of selecting individual i, f_i is the fitness value of individual i, and N is the size of the population.

2.6.2 Tournament Selection

Tournament selection methods randomly select a group of individuals from the population and choose the best individual from that group to reproduce. The size of the tournament and the probability of selection can be adjusted to control the selection pressure. Some of the commonly used tournament selection methods include [56]:

- Binary Tournament Selection: In binary tournament selection, two individuals are randomly selected from the population and compared based on their fitness value. The fitter individual is then chosen for reproduction. This process is repeated until the desired number of offspring is produced.
- Multi-Way Tournament Selection: Multi-way tournament selection is similar to binary tournament selection but selects more than two individuals for each tournament. The best individual from the tournament is then chosen for reproduction. This approach can be more effective at preserving genetic diversity but can also increase the computational cost.

The tournament selection method can be mathematically defined as follows:
Let T be a tournament of size k, and let p_i be the probability of selecting individual i.

$$p_i = \frac{1}{k} \sum_{j=1}^{k} [i = argmax_{j \in T} f_j] \tag{2.2}$$

where $argmax_{j \in T} f_j$ is the index of the individual in tournament T with the highest fitness value.

2.7 Genetic operators

Genetic algorithms (GAs) use mutation and crossover as primary genetic operators to create new offspring and explore the search space. Mutation and crossover are important in finding optimal solutions in genetic algorithms because they allow for the exploration and exploitation of the solution space.

2.7.1 Crossover

Crossover is a genetic operator that combines information from two or more individuals to create new offspring that inherit some features from their parents. In Figure 2.7 we can observe a single point crossover in a binary encoding scheme. Crossover is used to explore new areas of the search space and generate diverse solutions. Crossover is based on exchanging information between individuals by selecting parts of their chromosomes and recombining them to form new offspring [9].

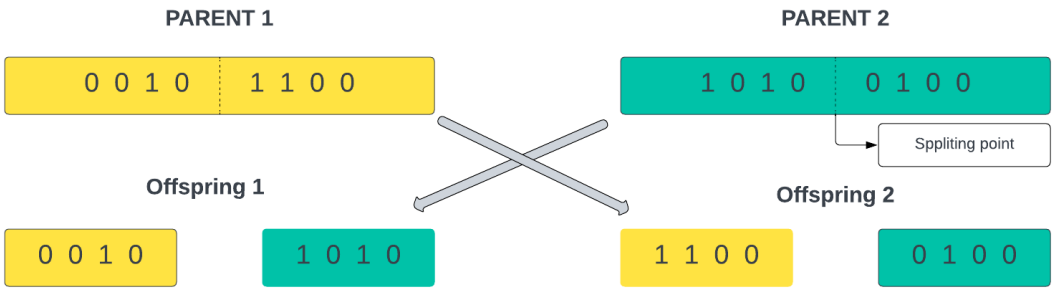


Figure 2.6: Single point Crossover in a Binary scheme

There are different types of crossover operators that can be used, depending on the problem and the encoding scheme used. Some of the commonly used crossover operators include [9]:

- One-point crossover: In single-point crossover, a random point is selected in the chromosomes of the two parents, and the chromosomes are exchanged at that point. The resulting offspring inherit one part of their chromosomes from one parent and the other part from the other parent.
- Multi-point crossover: Multi-point crossover is similar to single-point crossover but involves selecting multiple points in the chromosomes of the parents to exchange genetic material.
- Uniform crossover: In uniform crossover, the offspring are created by selecting genes from each parent with a fixed probability. This approach can introduce more diversity in the offspring and is often used in problems where the location of genes is not critical.

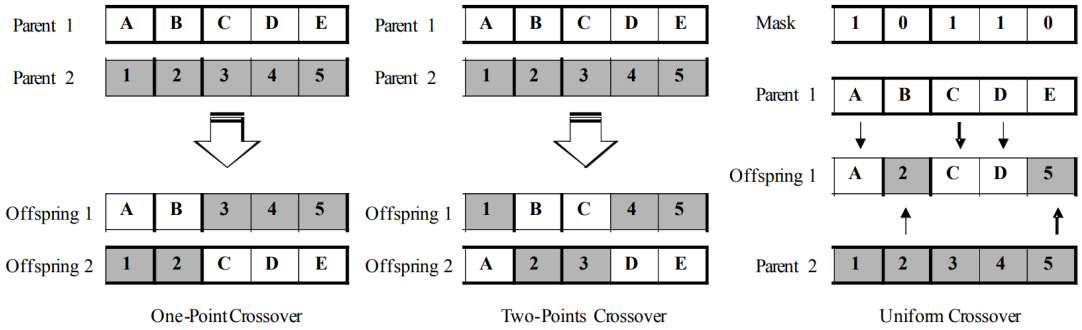


Figure 2.7: Crossover operators. Obtained from [9]

2.7.2 Mutation

According to [55], mutation is a fundamental genetic operator in genetic algorithms that randomly changes an individual's genes in the population. The primary objective of mutation is to add diversity to the population and avoid premature convergence by exploring new regions in the search space. It can be applied to different encoding schemes such as binary, real-valued, and permutation encoding. Various mutation operators are used in GAs to introduce randomness to the population. These include bit-flip mutation, swap mutation, inversion mutation, and Gaussian mutation. Bit-flip mutation is the simplest operator and involves flipping a single bit in the genome. Swap mutation, on the other hand, swaps two genes in the genome, while inversion mutation inverts a segment of the genome. Gaussian mutation adds a random value sampled from a Gaussian distribution to each gene [33].

2.8 Fitness

According to Hussain et al, fitness measures the quality of a candidate solution in a genetic algorithm. It is defined as the objective function or evaluation function that calculates the fitness value of each candidate solution. The fitness value represents how well the candidate solution satisfies the optimization criteria and is used to guide the search toward better solutions [57].

2.8.1 Fitness Function

The fitness function plays an important role in the success of genetic algorithms in solving complex optimization problems. The fitness function guides the search for better solutions by evaluating the quality of candidate solutions and selecting the best ones for reproduction and survival in the next generation [57]. A well-designed fitness function can significantly improve the efficiency and effectiveness of a genetic algorithm by reducing the number of evaluations needed to find the optimal solution. Furthermore, the fitness function can be used to incorporate domain-specific knowledge and problem-specific constraints, which can improve the quality of the solutions obtained [58].

2.8.2 Types of Fitness Functions:

Different types of fitness functions can be used in genetic algorithms depending on the optimization problem being solved. Some common types of fitness functions include:

- Objective-based Fitness Functions: These fitness functions are based on the objective(s) of the optimization problem. They calculate the fitness value of a candidate solution based on how well it satisfies the objective(s) of the problem. For example, in a vehicle routing problem, the fitness value of a candidate solution can be calculated based on the total distance traveled by the vehicles. [59]

- Constraint-based Fitness Functions: These fitness functions are used when the optimization problem has constraints that must be satisfied. The fitness value of a candidate solution is calculated based on how well it satisfies the constraints of the problem. For example, in a scheduling problem, the fitness value of a candidate solution can be calculated based on how well it satisfies the constraints of resource availability and task deadlines. [60]

- Hybrid Fitness Functions: These fitness functions combine objective- and constraint-based fitness functions. They are used when the optimization problem has both objectives and constraints. The fitness value of a candidate solution is calculated based on how well it satisfies both the objectives and the constraints of the problem. [45]

2.9 Termination Criteria

Termination criteria are essential in genetic algorithms (GAs) because they dictate when the optimization process should stop. If the algorithm runs indefinitely, it may lead to overfitting or poor performance. Therefore, choosing appropriate termination criteria that ensure the GA's convergence to an optimal solution within a reasonable time is crucial [58].

One common termination criterion is a fixed number of generations. This approach terminates the algorithm after a predetermined number of generations. However, this criterion may lead to suboptimal solutions if the specified number of generations is insufficient for the algorithm to converge. Another approach is to use a convergence criterion, which stops the algorithm when the population has converged. The convergence can be measured by the fitness function's standard deviation or the change in the best fitness value over generations [59].

Another popular termination criterion is the computing time limit. The algorithm stops after a specified amount of time, regardless of whether the algorithm has converged or not. This criterion is often used in real-time applications where the optimization process must be completed within a certain time frame. However, as suggested by [60], the computing time limit must be set carefully to avoid terminating the algorithm prematurely, leading to suboptimal solutions.

2.10 Single an Multi-objective Optimization

Optimization problems can be classified into two main categories: single-objective and multi-objective optimization. In single-objective optimization, the goal is to find the optimal solution for one objective function, and the solution is optimized based on a single criterion

In contrast, multi-objective optimization deals with optimizing multiple conflicting objectives simultaneously. This approach is typically used when several objectives are considered, and finding the best solution is not as straightforward as in single objective optimization.

In this section, we will discuss the basic concepts and approaches of both single-objective

and multi-objective optimization and highlight the differences.

2.10.1 Single objective optimization problem

Before learning what multi-objective optimization is all about, it is very convenient to define single-objective optimization. The optimization of a simple objective can be defined mathematically according to [4] as maximizing or minimizing $f(\mathbf{x})$ (the only one objective function) subject to:

$$\begin{aligned} g_i(\mathbf{x}) \leq 0, i = \{1, \ldots, m\}, \\ h_j(\mathbf{x}) = 0, j = \{1, \ldots, p\} \mathbf{x} \in \Omega \end{aligned} \tag{2.3}$$

The found solution can minimize or maximize the scalar $f(\mathbf{x})$ where,
$\mathbf{x}$ is a n-dimensional decision variable vector $\mathbf{x} = (x_1, \ldots, x_n)$ from some universe Ω.

Notice that Equations 2.3 are the constraints that must be fulfilled in the optimization process of $f(\mathbf{x})$.Ω contains all possible $\mathbf{x}$ that can be used to satisfy an evaluation of $f(\mathbf{x})$ and its constraints.

2.10.2 Multi-objective optimization problem (MOP)

Multiobjective optimization, or multicriteria optimization, involves finding a set of decision variables that satisfies specific constraints and optimizes a vector function. The vector function comprises objective functions that often represent performance criteria that conflict with each other. The goal is to find a solution that provides acceptable values for all objective functions, as the decision maker judges [55].

As stated by Coello et al. [4], the problem of multiobjective optimization aims to optimize two or more objectives that conflict with each other simultaneously. This means that improving one objective may negatively impact another objective [55].

Decision Variables

The decision variables refer to the numerical values that give solution to the optimization problem. Mathematically we can define x as the vector of n decision variables [4]:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \tag{2.4}$$

That can be convenient written as:

$$\mathbf{x} = [x_1, x_2, \ldots, x_n]^T \tag{2.5}$$

Constraints

Constraints represent restrictions imposed by the environment or available resources [58]. They are expressed as mathematical inequalities or equalities [4]. Inequalities are typically represented as $g_i(x) \leq 0$, where $i = 1, ..., m$, and equalities are represented as $h_j(x) = 0$, where $j = 1, ..., p$. It is important to note that the number of equality constraints, represented by p, must be less than the number of decision variables, represented by n, to ensure that the problem is not overconstrained [58]. Additionally, constraints can be explicit or implicit, and algorithms must be able to compute $g_i(x)$ for any given vector x if constraints are implicit [4].

Objective Functions

According to [9], an objective function is a function that maps a candidate solution in the search space to a single real number that measures how good that candidate solution is concerning the problem being solved. In other words, an objective function is a mathematical expression that takes a candidate solution as input and produces a scalar value as output, indicating the solution's quality [9].

According to [4], each objective function can be defined as $f_1(x), f_2(x), \ldots, f_k(x)$. The number of objective functions in a Multiobjective Optimization Problem (MOP) is denoted by k. Hence, the objective functions are represented as a vector function **F(x)** in the MOP being solved.

$$\mathbf{F(x)} = \begin{bmatrix} f_1(x) \\ f_2(x) \\ \vdots \\ f_k(x) \end{bmatrix} \tag{2.6}$$

In Multiobjective Optimization Problems, two Euclidean spaces are typically considered. The first is the n-dimensional space of decision variables, where each axis corresponds to a component of the vector x. The second is the k-dimensional space of objective functions, where each axis corresponds to a component vector $f_k(x)$. Each point in the decision variables space represents a solution and maps to a corresponding point in the objective functions space, which provides information about the quality of the solution in terms of its objective function values.

Mathematical Definition of MOP

The Multiobjective Optimization Problem (MOP) is a problem in which the aim is to optimize k objective functions simultaneously. This requires finding a set of decision variables that minimize or maximize a vector function $F(x)$ subject to $m + p$ constraints on the objective functions and n decision variables. In general, there is not one unique solution but a set of solutions, which can be found using the Pareto Optimality Theory [4].

Eiben and Smith [9] provide a formal symbolic definition of a general MOP as follows:

"A general MOP is defined as minimizing (or maximizing) $F(x) = (f_1(x), \ldots, f_k(x))$ subject to $g_i(x) \leq 0$, $i = 1, \ldots, m$, and $h_j(x) = 0$, $j = 1, \ldots, p$, $x \in \Omega$."

Here, x represents a vector of n decision variables, $f_1(x), \ldots, f_k(x)$ are the objective functions, $g_i(x) \leq 0$ and $h_j(x) = 0$ represent constraints that must be fulfilled while optimizing $F(x)$, and Ω contains all possible values of x that can be used to satisfy the evaluation of $F(x)$.

The decision variables are represented in the n-dimensional space, while the objective functions are represented in the k-dimensional space. The evaluation function, $F : \Omega \to \Lambda$, maps the vector of decision variables $x = (x_1, \ldots, x_n)$ to output vectors $y = (a_1, \ldots, a_k)$. The k objective functions can be linear or nonlinear and continuous or discrete in nature.

The Ideal Vector

The ideal vector in multi-objective optimization is defined as the vector obtained by evaluating the optimal values of each objective function at a point in the feasible region. Mathematically we have in Equation2.7 a vector of variables that optimizes the i_{th} function, either minimizes or maximizes:

$$\mathbf{x}^{0(i)} = \left[x_1^{0(i)}, x_2^{0(i)}, \ldots, x_n^{0(i)}\right]^T \tag{2.7}$$

We can also define it as vector $\mathbf{x}^{0(i)} \in \Omega$:

$$f_i\left(\mathbf{x}^{0(i)}\right) = \text{opt}_{x\in\Omega}\, f_i(\mathbf{x}) \tag{2.8}$$

In Equation 2.9, we have a vector of ideal solutions for each objective. $\mathbf{f_i}^0$ represents the optimum of the corresponding i_{th}objective function. The corresponding point in $\mathbb{R}^n$ determines this vector is the ideal or utopical.

$$\mathbf{f}^0 = \left[f_1^0, f_2^0, \ldots, f_k^0\right]^T \tag{2.9}$$

In conclusion, the ideal vector consists of the optimal solutions for each individual objective function that are attained at the identical point in $\mathbb{R}^n$.

2.10.3 Pareto Optimality

In multiobjective optimization problems, the concept of "optimum" differs from that in global optimization problems (single objective) because there are multiple objective functions to consider. Rather than a single solution, the goal is to find a set of solutions that represent good trade-offs or compromises among the different objectives. This set of solutions is known as the Pareto Optimal Set, which consists of solutions that cannot be improved in one objective without worsening at least one other objective. In other words, the Pareto Optimal Set represents a set of non-dominated solutions where no other feasible solution is better in all the objectives.

Coello et al. defines a solution x $\in \Omega$ is said to be Pareto Optimal with respect to Ω if and only if (iff) there is no $\mathbf{x}' \in \Omega$ for which $\mathbf{v} = F(\mathbf{x}') = (f_1(\mathbf{x}'), \ldots, f_k(\mathbf{x}'))$ dominates

$\mathbf{u} = F(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_k(\mathbf{x}))$. The phrase Pareto Optimal means concerning the entire decision variable space unless otherwise specified [4].

Pareto Dominance

Pareto dominance is a comparison criterion used in multi-objective optimization to compare two solutions in terms of their objective function values [12] . For example, solution A is said to dominate solution B if A is no worse than B in any objective and is strictly better than B in at least one objective. In other words, solution A dominates B if it provides a better trade-off in the objectives than B. [4] expresses it mathematically as:

A vector $\mathbf{u} = (u_1, \ldots, u_k)$ is said to dominate another vector $\mathbf{v} = (v_1, \ldots, v_k)$ (denoted by $\mathbf{u} \preceq \mathbf{v}$) if and only if $\mathbf{u}$ is partially less than $\mathbf{v}$, i.e., $\forall i \in \{1, \ldots, k\}, u_i \leq v_i \wedge \exists i \in \{1, \ldots, k\} : u_i < v_i$

Pareto Optimal Set

Coello et al. [4] defined Pareto optimal solutions as non-inferior, admissible, or efficient solutions. The set of all solutions whose associated vectors are non-dominated is called the Pareto optimal set. Pareto optimal solutions refer to solutions in the search space of genotypes (decision space) whose corresponding objective vector components in the phenotype space cannot be improved at the same time.

Coello et al. provide a mathematical definition of the Pareto Optimal Set as $\mathcal{P}^*$, where:

$$\mathcal{P}^* := \left\{ \mathbf{x} \in \Omega \mid \neg \exists \mathbf{x}' \in \Omega \quad F\left(\mathbf{x}'\right) \preceq F(\mathbf{x}) \right\} \tag{2.10}$$

This equation defines $\mathcal{P}^*$ as the set of all solutions $\mathbf{x}$ in the decision space Ω for which there does not exist another solution $\mathbf{x}'$ in Ω whose corresponding objective function values $F(\mathbf{x}')$ are all dominated by the objective function values $F(\mathbf{x})$ of $\mathbf{x}$, represented by the Pareto dominance relation $\preceq$. In other words, the solutions in $\mathcal{P}^*$ are non-dominated and no other solution exists that is better in all objectives.

Pareto Front

In the previous section, we learned that the Pareto optimal set is a set of solutions in the decision space, whereas the Pareto front, as defined by [9], is a set of objective function

values in the objective space. See figure 2.8. The Pareto front represents the trade-offs between the different objectives and provides a visualization of the optimal trade-offs between the objectives. The Pareto optimal set gives us the set of solutions that achieve these optimal trade-offs, while the Pareto front gives us the corresponding objective function values of these solutions.

Coello et al. define the Pareto Front $\mathcal{PF}^*$ for a given MOP, $F(x)$, and Pareto optimal set $\mathcal{P}^*$ as:

$$\mathcal{PF}^* := \{\mathbf{u} = F(\mathbf{x}) \mid \mathbf{x} \in \mathcal{P}^*\} \tag{2.11}$$

Here, $\mathbf{x}$ represents a solution in the decision space that belongs to the Pareto optimal set, and $\mathbf{u}$ represents the corresponding objective vector in the objective space that belongs to the Pareto front. The Pareto front provides a visualization of the optimal trade-offs between different objectives, while the Pareto optimal set gives the set of solutions that achieve these optimal trade-offs [55].

The usual approach for obtaining the Pareto front is to calculate a large number of points in the decision space Ω, along with their corresponding objective function values $f(\Omega)$. In contrast to single-objective optimization problems, where there is typically only one optimal solution, multi-objective optimization problems often have a vast number of solutions that lie on a Pareto front.

As illustrated in Figure 2.8, the Pareto front represents a trade-off between minimizing $f1$ and $f2$. The points lying on the Pareto front, such as X_1 and X_2, are superior solutions compared to X_3 in some of the objectives. [10]

The function $f : \Omega \to \Lambda$ in an MOP maps the decision variables $x = x_1, ..., x_n$ to vectors $y = a_1, ..., a_k$ as illustrated in Figure 2.9. For the case where $n = 2$, and $k = 3$, this function can either cover or not cover a region of the objective function space, depending on the functions and constraints that define the specific MOP.

2.11 Performance Indicators in MOP

According to [61], Evolutionary Multi-objective (EMO) procedures aim to achieve two goals: (i) to converge towards the Pareto-optimal front and (ii) to obtain a diverse set of

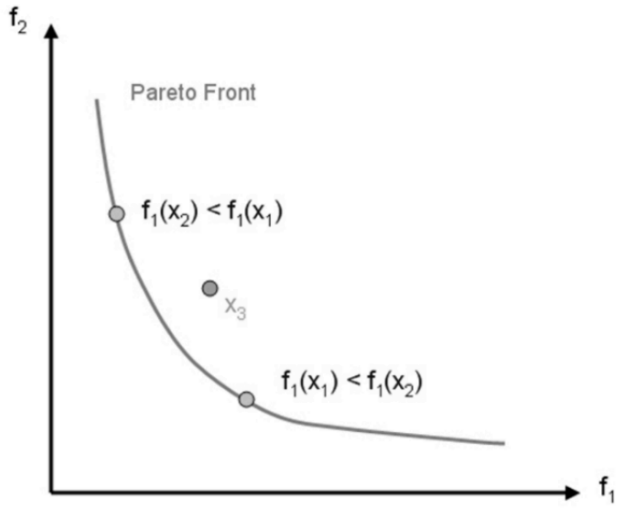


Figure 2.8: Pareto Front representation of two conflicting objectives. Obtained from [10]

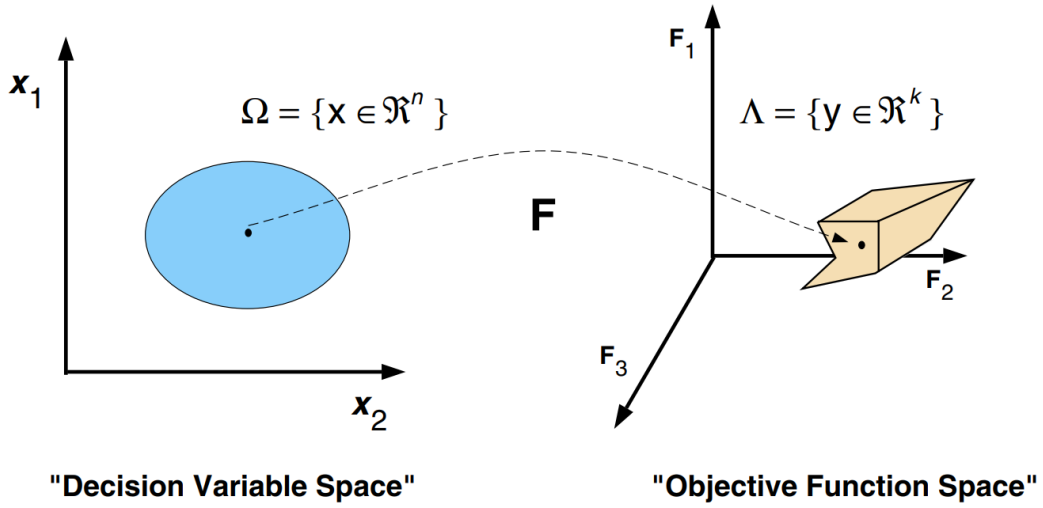


Figure 2.9: Evaluation Mapping from Decision Variable Space to Objective Function Space [4]

solutions. There are three sets of performance indicators used in EMO research:

- Convergence metrics evaluating the convergence to the known Pareto-optimal front.
- Spread metrics evaluating the spread of solutions on the Pareto-optimal front.
- Metrics evaluating the convergence and spread of solutions, such as hypervolume, coverage, and R-metrics.

The study of Knowle et al. suggests that binary performance metrics, such as the

epsilon indicator, hypervolume indicator, and utility indicators R1 to R3, are more suitable measures for multi-objective optimization when comparing two sets of solutions A and B [62].

2.11.1 Hypervolume as a Performance Indicator

Hypervolume is a performance indicator used in multi-objective optimization (MOO) to evaluate the quality of a set of solutions concerning the Pareto-optimal front. It is based on the volume of the objective space dominated by a set of solutions, relative to a reference point in the objective space [11]. The larger the hypervolume, the better the set of solutions is considered regarding coverage of the Pareto front. Figure 2.10 illustrates the concept of hypervolume and hypervolume contributions in both 2-D and 3-D spaces. The top left panel shows an example of a 2-D hypervolume calculation, where the hypervolume is the area enclosed by the dotted line, and the reference point is marked with an "r." The top right panel shows the hypervolume contributions of each solution in the set. The bottom figures show the hypervolume and hypervolume contributions but in 3-D.

2.11.2 Calculation of Hypervolume

To calculate the hypervolume, a reference point in the objective space is needed. This point is typically set to be a point dominated by all solutions, or a user-defined point [12]. According to Custodio et al. ,the hypervolume indicator, also known as the S-metric or size of space covered, is a measure used in multi-objective optimization to evaluate a set A of approximated solutions in $\mathbb{R}^m$ relative to a reference point r in $\mathbb{R}^m$ that is dominated by all points in A. It is calculated as the volume of the space dominated by the solutions in A and bounded by the reference point r [63]. This can be written mathematically as:

$$HI(A) = \text{Vol}\left\{b \in \mathbb{R}^m \mid b \leq r \wedge \exists a \in A : a \leq b\right\} = \text{Vol}\left(\cup_{a \in A}[a, r]\right) \tag{2.12}$$

where $Vol(.)$ represents the Lebesgue measure of a m-dimensional set of points and $[a, r]$ is the interval box with lower corner a and upper corner r. In two dimensions, this corresponds to the covered area, while in three dimensions, it corresponds to the covered volume.

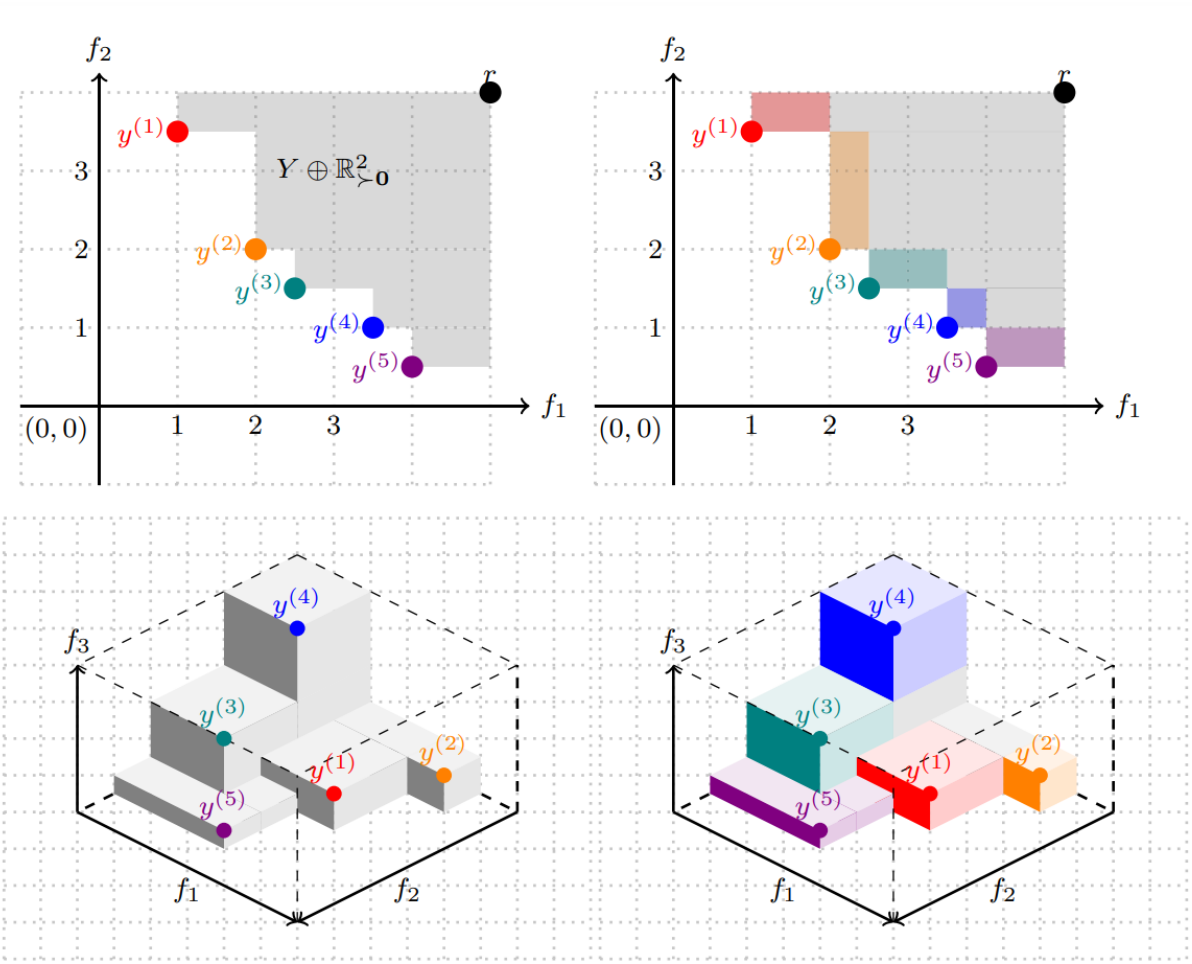


Figure 2.10: Hypervolume graphical representation in 2-D and 3-D. [11]

2.12 Non-dominated Sorting Algorithm II

Non-dominated Sorting Algorithm II (NSGA-II) is a non-explicit black-box multi-objective evolutionary algorithm designed for solving multi-objective problems (MOPs). It is an extension of Srinivas and Deb's original nondominated sorting algorithm (NSGA). NSGA-II is widely used and is characterized by being of the Elitist type [4]. According to Yusoff et al., it incorporates a mechanism for preserving the dominant solutions through several iterations in the execution of the genetic algorithm. The algorithm uses modified crossing, selection, and mutation mechanisms defined by the classic genetic algorithm [64].

2.12.1 NSGA-II definition

The NSGA-II algorithm is a Multi-Objective Evolutionary Algorithm (MOEA) developed by Deb as an extension of the NSGA algorithm [4]. NSGA-II is characterized by three essential features: a fast crowded distance estimation procedure, a simple crowded comparison operator, and a fast non-dominated sorting approach [64].

2.12.2 NSGA-II Algorithm Overview and Pseudocode

As shown in figure 2.11, the algorithm creates a population of competing individuals, ranks and sorts each individual according to their level of non-dominance, and applies evolutionary operations (EVOP) to obtain a new pool of offspring. This offspring is combined with the parents, and the pool is partitioned into fronts. The NSGA-II algorithm then does niching by calculating the crowding distance of each individual. Finally, the selection operator uses the crowding distance to maintain a diverse front and ensures that a crowding distance separates each individual. This process provides a diverse population and helps the algorithm explore the fitness landscape.

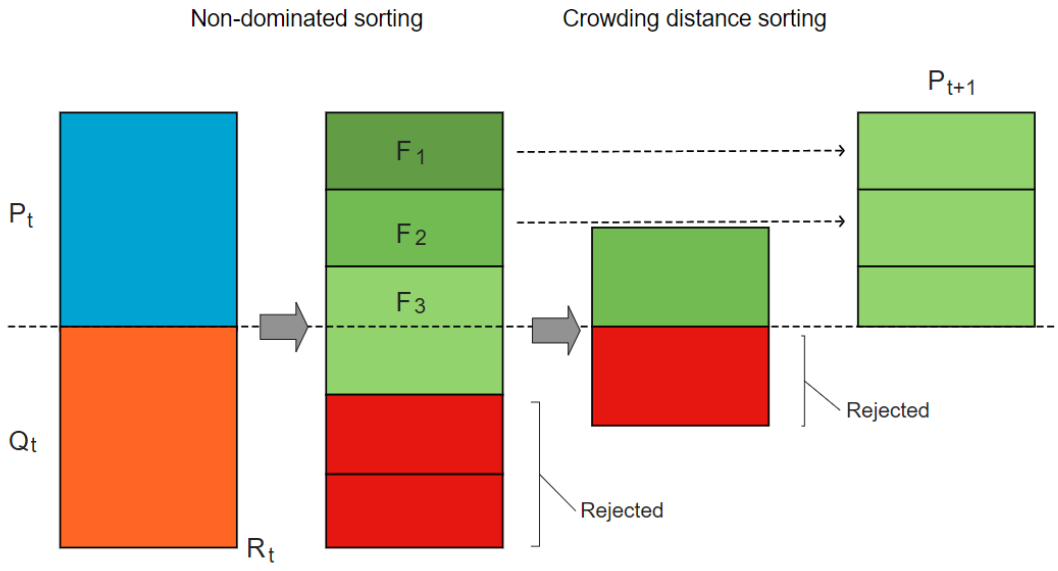


Figure 2.11: The classic representation of the NSGA-II algorithm behavior. P_t (parent's population), Q_t (offspring population at generation t). F_1 are the best solutions from the combined populations (parents and offspring). F_2 are the second best solutions, and so on. Based on[4]

Algorithm 1 shows the NSGA-II pseudocode proposed by Deb as a better version of the NSGA [4].

Algorithm 1 NSGA-II algorithm

```
1 : procedure NSGA-II (𝒩′, g, f_k (x_k))   ▷ 𝒩′ members evolved g generations to solve f_k(x)
2 :     Initialize Population ℙ′
3 :     Generate random population - size 𝒩′
4 :     Evaluate Objective Values
5 :     Assign Rank (level) Based on Pareto dominance - sort
6 :     Generate Child Population
7 :         Binary Tournament Selection
8 :         Recombination and Mutation
9 :     for i = 1 to g do
10 :        for each Parent and Child in Population do
11 :            Assign Rank (level) based on Pareto - sort
12 :            Generate sets of nondominated vectors along PF known
13 :            Loop (inside) by adding solutions to next generation starting from the first front until 𝒩′ individuals found determine crowding distance between points on each front
14 :        end for
15 :        Select points (elitist) on the lower front (with lower rank) and are outside a crowding distance
16 :        Create next generation
17 :            Binary Tournament Selection
18 :            Recombination and Mutation
19 :    end for
20 : end procedure
```

Table 2.3: NSGA-II Pseudocode. Obatained from [4]

The NSGA-II algorithm starts by randomly generating a population of individuals. Each individual is evaluated using the objective functions. The algorithm assigns a rank (level) to each individual based on Pareto dominance and sorts the population accordingly.

The next step is to generate a child population using binary tournament selection, recombination, and mutation. The same process of rank assignment and sorting is applied to the parent and child population.

The algorithm then generates sets of non-dominated vectors along the Pareto front. It determines the crowding distance between points on each front and selects points on the lower front, which are outside a crowding distance to create the next generation.

This process continues for a specified number of generations until the optimal solution(s)

are found. The algorithm's output is a set of Pareto optimal solutions that represents the trade-off between the objective functions.

2.12.3 Crowding Distance

The crowding distance is a measure of the diversity of solutions in the objective space in multi-objective optimization problems (MOPs). According to Coello et al., [4], the crowding distance of a point i is a measure of the objective space around i that is not occupied by any other solution in the population. This measure is used to maintain a diverse front and ensures that each individual in the front has enough space around it. The crowding distance concept was introduced in the original NSGA algorithm [65] and is widely used in many multi-objective optimization algorithms. See Figure 2.12.

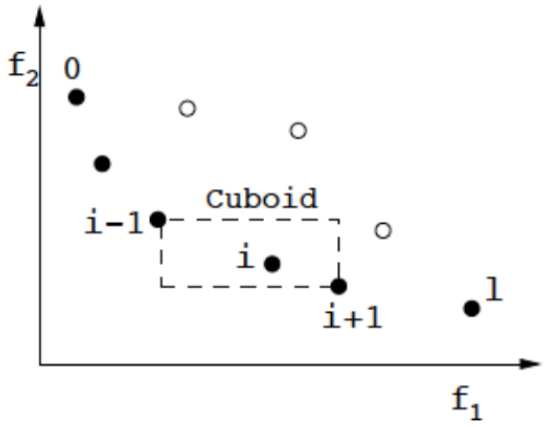


Figure 2.12: Calculation of Crowding distance. Obtained from[12]

Chapter 3

State of the Art

This section reviews some of the algorithms and proposals to solve the problem. All the proposed methods have, as their primary objective or as one of their objectives, found the best location for electric vehicle charging stations.

3.1 Optimization trough Evolutionary algorithms

Evolutionary algorithms are powerful tools for solving nonlinear problems that handle many variables. These problems can be found in industry, in the intelligent design of cities, in mobility planning, etc.

Various optimization techniques, including genetic and swarm algorithms, have been proposed to address traffic problems by adapting the schedules of traffic signals. A review of this research can be found in [66]. In a related work, [67] focuses on optimizing the location and dimensions of charging stations along a highway network while taking budget constraints into account. The proposed model is applied to a road network that is represented by nodes and arcs.

The algorithm searches for potential nodes or cities where electric charging stations can be located. The methodology of this project is based on the use of two algorithms: A genetic algorithm used for the optimization of combinational problems and a Heuristic Algorithm explicitly designed for this problem. In the search carried out by [68], a new fuzzy bi-objective mathematical model is presented for the supply chain's production and distribution problem. This research uses multi-objective fuzzy programming, possibility programming, and a self-adaptive evolution algorithm. In the article developed by [69],

researchers address vehicle routing problems with backhauls. The Backhaul concept is a new business idea. It is based on preventing the vehicle from traveling empty on the return trip. The route is planned based on the providers who wish to take advantage of the free space. This problem is classified as NP; therefore, it produces a high computational cost; the researchers propose a linear algorithm for deliveries with few stays, but for significant problems, genetic algorithms are used. Zhou et al. [70] create a model that considers the total social cost and the cost of electric charging station operations under various distribution conditions. Among its main objectives is the location of charging stations using genetic algorithms. Finally, this work seeks to determine the factors that directly influence the cost of the stations, and for this, a Sensitivity study is carried out. The social cost model considers two fundamental aspects, the economic and environmental costs. The model proposed by the researchers suggests three objective functions to be optimized: Building cost, charging costs and environmental costs.

As we can see, urban problems can be widely resolved through different evolutionary algorithms since they deal with restrictions and variables that make a model approach reality. Mohammad et al.[71] study analyzes the situation of suspended sediments that can cause damage to drainage systems or contamination in rivers. The researchers propose a novel hybrid approach of several algorithms for estimating SSL (River Suspended Sediment Load) in which the multi-layer perceptron together with particle swarm optimization (PSO) and then integrated with differential evolution (DE) algorithm called MLP-PSODE. This study is carried out in the Mahabad river in northwest Iran. The model works satisfactorily and shows high confidence [72]. Evolutionary Algorithms addressed the bus synchronization problem in Montevideo, Uruguay. This study indicates that the use of a $(\mu + \lambda)$ model shows an improvement of 13% compared to other intuitive algorithms and the current organization system. Another example of evolutionary algorithms application is the one proposed by Jahandideh et al.. Researchers seek to guarantee the optimal use of water reservoir systems. A large number of evolutionary algorithms are discussed, artificial bee colony (ABC), ant colony optimization (ACO), bat algorithm (BA), particle swarm optimization (PSO), etc. All algorithms outperform traditional reservoir optimization methods, such as non-linear programming (NLP) and dynamic programming (DP) [73]. In Adenaw and Lienkamp's research, authors have designed a project that seeks to

optimize the charging routes of electric vehicles through a co-evolutionary algorithm. For agents simulated through the MATSIM framework [14], decision making is directly related to model assumptions or an integrated process that considers both charging behavior and location choice. The co-evolutionary algorithm, regarded as an evolutionary method, consists of assigning a score to mobility charging plans based on the usefulness of the performance behavior. Depending on the score, mobility plans are optimized, replanning heuristically over several generations until having the desired optimized charging plan. It is essential to point out that the authors have developed a co-evolutionary model due to the dependency between the agents' behavior and the score obtained by the charging plans [74].

In Rizopoulos et al. investigation, the aim is to solve the Daily Activity Chains Optimization problem, which is a problem that, like [74] , is a route search problem similar to TSP. This project seeks to use genetic algorithms to generate practically usable activity chains. The proposed model considers factors such as the consumption of Electric Cars, the location of the charging stations, and the types of plugs. This model suggests a pre-optimization; the authors affirm it improves the final results. One of the main objectives is travel time minimization, and the results show an improvement of 31.42% compared to other travel schedules [75].

3.2 Charging Station Location Models

One of this research's main objectives is to search for the optimal location of charging stations for electric cars and thus form a solid distribution network. In the study carried out by [76], a comparative summary of the most popular models for solving this type of problem is made. They show the importance of the leap from mobility based on fossil fuels to electromobility. They also suggest that a good infrastructure significantly reduces implementation costs. The authors of this article mention the popularity of two types of algorithms such as genetic algorithms and particle algorithms, always showing more significant popularity of genetic algorithms. However, they also talk about other algorithms with optimal results, such as gray wolf optimization, spider monkey optimization, game theory, grasshopper optimization, etc. [74]

In Gampa's et al. study, it is proposed to use a fuzzy multi-objective approach based on the Grasshopper optimization algorithm (GOA). This large-scale project seeks to optimally size and locates components for an electrical charge distribution system. The projector consists of Distributed Generations (DG), Shunt Capacitors (SC), and Electric Vehicles (EVs). The fuzzy Grasshopper optimization algorithm is used to identify the optimal locations for EV charging stations and the number of vehicles at the charging stations. The simulation shows the rapid convergence in the results against the techniques of Particle Swarm Optimization (PSO) and Genetic Algorithms (GA) [77]. The study developed by Pan et al. proposes a model based on the driver's activities. The main goal is to maximize the number of EV user activities. Firstly, the researchers carry out a deterministic process that seeks to simulate the charging behavior of EV drivers. This step involves drivers' existing activities and public and domestic charging availability. A coverage location model for public electric vehicle charging stations is proposed as a second step. The study is applied in Beijing, China. Results show the model covers 90% of the driver's needs without modifying the daily trips [78]. Zhang et al. proposes a model for ride-hailing or public transport. This model talks about using autonomous electric vehicles (AEV) to reduce greenhouse gas emissions and costs. Before launching a proposal, the authors discuss the design of a reliable and extensive network of charging stations for electric vehicles and propose the following methodology: First, they use the MATSIM BEAM module to simulate driving, parking, and driving behaviors of AEV in San Francisco city. The second step is to record the load needs, when, where, and how the load needs occur within the system. Finally, the charging stations are located according to the needs and consider the service quality as a restriction [79].

In Ji's et al. research, a charging demand model is proposed to analyze the utility of each point where a charging station is placed. They take factors such as the number of vehicles that park in one place, the battery charge state, the parking time, and where are the most frequently associated locations. The authors solve the problem using a Spatio-temporal analysis model. The proposed model is based on driver behavior and uses MATSIM to simulate electric vehicles in Berlin. Through a series of iterations, the algorithm searches for the most optimal result based on an initial budget and the above criteria. The new stations are placed according to the results of the optimization. The objective of this study

is that drivers do not change their mobility plans and do not have to deviate from their route to complete the charge. As a result, this model allows for Spatio-temporal planning of the deployment of charging infrastructure. The results indicate that the method proposed by the authors helps reduce range anxiety without adding restrictions due to using electric vehicles instead of conventional ones [80]. In the research carried out by Raith et al. , the authors develop a simulator known as ETSIM (Electronic Taxi Simulation) with a focus on implementing an electric taxi service, remarkably unlike other projects. The methodology used by this project is based on a simulation of discrete events. According to the authors, this means there are a certain number of available taxis in any place in the scenario. In the common taxi situation, a passenger hails the taxi, and it takes the passenger to their destination and then becomes inactive or returns to the taxi stop. In the case of e-taxi, before accepting a passenger's request, the state of charge (SoC) is analyzed. If there is enough charge to make the trip and go to a charging station, then the e-taxi accepts the request; otherwise, the request is rejected. The developed framework allows the measurement of certain variables, such as the SoC of a certain e-taxi in a day. Together with other factors, for instance, the rejection of trips, the charging time, and the popularity of particular places in the scenario, allow determining the best position for charging stations [81].

3.3 Electromobility Frameworks

In the world of computational research, there is a vast diversity of frameworks, and electromobility is a field that takes advantage of these resources. Research such [79, 74, 80] use the MATSIM framework and some kinds of extensions to simulate agents and scenarios. For instance, Zhang et al. [79] uses BEAM extension (Behaviour, Energy, Autonomy, and Mobility). BEAM is an MATSIM extension that shifts some behavioral emphasis from planning throughout the day to planning within the day. This is important because BEAM can simulate modern mobility services in a way that reflects the emerging transportation system. According to authors, agents dynamically respond to the system state during mobility simulation.The research of [74] employs a new coevolutionary learning model for adaptive charging behavior. The simulation framework used to implement the system is

based on the MATSiM framework using the EVContrib. The authors use MATSIM's scoring and rescheduling to optimize agent charging plans through an iterative algorithm that optimizes the resulting plans. In [82] study, an online vehicle charger assignment model is proposed to minimize the total vehicle inactivity time for recharging. This is a relevant problem in electromobility, especially in public transport, since fleets must waste as little time queuing at charging stations as possible. The model applied by the researchers is a MILP (Mixed-Integer Linear Programming) model based on the rolling time-window framework. The case study is developed in the bus service of Luxembourg city. Authors execute a simulation scenario. The results show the proposed method considerably reduces up to 27% vehicle charging operation time and queue delays. In [81], the authors develop their own simulation tool. The authors use the Julia language and the Open Street Maps (OSM) data to identify specific characteristics that make implementing an electric taxi service possible. The simulator models the e-taxi behavior, charging needs, and the state of charge-based services. With the results obtained, it is possible to understand the space-time demand and thus also look for the best location for the charging stations. The system, E-Taxi Simulation or ETSIM, obtains results of the number of trips that the e-taxis could not execute due to the state of charge (SOC) and thus decides the best location for the charging stations.

3.4 Evolutionary Frameworks approach

Evolutionary frameworks have been extensively used for solving optimization problems in various domains, including engineering, finance, and biology. With the availability of multiple optimization frameworks, some researchers compare their performance in terms of convergence and diversity to select the most appropriate framework for a particular application [83, 84, 85, 86, 87]. Here we present a review of some authors, the problem to solve, the used frameworks, and their respective conclusions.

In a study by [83], several optimization frameworks, including DEAP, PyGMO, and Platypus, were compared for optimizing the design of offshore wind turbines. The study showed that DEAP outperformed the other frameworks regarding convergence and diversity, while Platypus provided a better balance between convergence and diversity. Oyama

et al. research talks about the performance of several multi-objective optimization frameworks, including Platypus, NSGA-II, and MOEA/D, compared for optimizing water distribution network design. The study showed that Platypus outperformed the other frameworks in terms of convergence, while NSGA-II and MOEA/D provided better diversity. Mariscal et al. research compares DEAP, PyGMO, and Platypus frameworks in optimizing aircraft wings desing. The study showed that Platypus provided the best balance between convergence and diversity, while DEAP and PyGMO provided better convergence. In a survey by Rodríguez-Fernández et al., the performance of several optimization frameworks, including PyGMO, DEAP, and Optunity, was compared for optimizing the design of photovoltaic systems. The study showed that PyGMO and DEAP provided better convergence and diversity than Optunity. In a study by Chen et al., optimization frameworks, including DEAP and PyGMO, were compared for optimizing the design of power systems. The study showed that DEAP provided better convergence and diversity than PyGMO.

3.5 Station location through GA

As the number of electric vehicles (EVs) on the road increases, so does the need for electric vehicle charging stations. Therefore, the placement of these charging stations plays a critical role in the widespread adoption of EVs, as the location and number of charging stations directly affect the convenience and accessibility of EVs. To address this challenge, researchers have developed various genetic algorithm (GA) based approaches to optimize the placement of charging stations, taking into account a wide range of factors such as user convenience, charging station utilization, network connectivity, and environmental impact.

In the study by Liu et al., researchers optimized the placement of electric vehicle charging stations in urban areas. The study used a multi-objective optimization framework to find the optimal balance between user convenience, charging station utilization, and network connectivity [88]. Gao et al. proposed a hybrid GA-based approach for optimizing the placement of electric vehicle charging stations in large-scale transportation networks. The study used a combination of GA and simulation techniques to find the optimal location and capacity of the charging stations, taking into account factors such as traffic demand, energy consumption, and network connectivity [89]. Niu et al. studied the placement of

electric charging stations in rural areas. The study used a multi-objective optimization framework to find the optimal balance between user convenience, charging station utilization, and environmental impact [90]. Li et al. studied the placement in urban areas with limited space. The study used a novel clustering technique to group charging stations based on their proximity and usage patterns and then used GA to find the optimal placement of the clusters [91]. Zhou et al. used a multi-objective GA-based approach for optimizing the placement of fast charging stations for electric vehicles. The study used a novel encoding scheme and genetic operators to improve the algorithm's performance [92]. Zhang et al. cared about large-scale transportation networks with uncertain demand. Therefore, the study used a robust optimization framework to find the optimal location and capacity of the charging stations under different demand scenarios [93]. Huang et al. optimized the placement of electric vehicle charging stations in urban areas with limited parking spaces. The study used a novel parking-based optimization framework to find the charging stations' optimal placement, considering the parking spaces' availability and the charging needs of electric vehicles [94].

GA-based approaches have proven to be effective tools for optimizing the placement of electric vehicle charging stations in a variety of settings, including urban areas, large-scale transportation networks, rural areas, and areas with limited space. By using multi-objective optimization frameworks, clustering techniques, and simulation-based approaches, researchers have found optimal solutions that balance the competing objectives of convenience, efficiency, and sustainability. These studies provide valuable insights into the potential of GA-based approaches to address the challenges of building a robust and accessible EV charging infrastructure and suggest promising avenues for future research.

Chapter 4

Methodology

4.1 Phases of Problem Solving

This section outlines the general phases of solving the charging station allocation problem, including data gathering, tools and techniques selection, assumptions and limitations of the problem-solving process. In upcoming sections, we will discuss in detail all the methodological phases that led to the problem's solution based on the theory studied and the research in this field. Figure 4.1 shows the phases of problem solving.

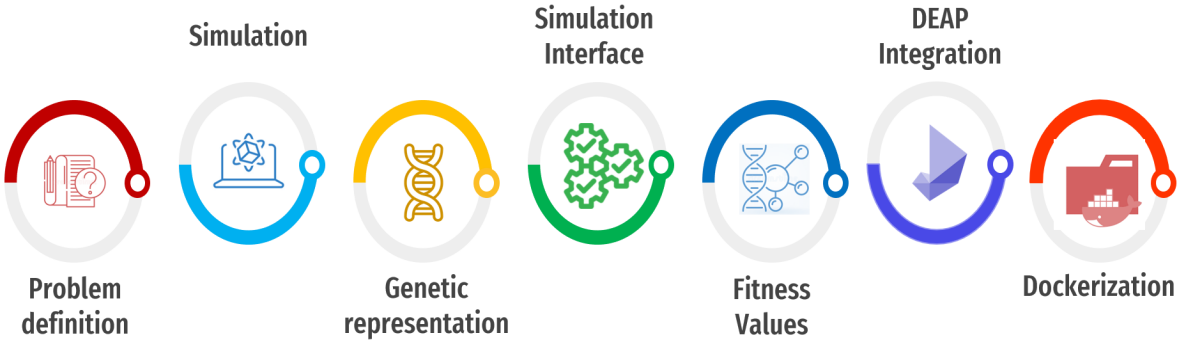


Figure 4.1: Problem solving phases

4.1.1 Problem Definition

The first step in solving the charging station allocation problem was to identify the possible tools and techniques necessary to find the ideal location and capacity of each electric charging station. Next, the objective optimization functions were defined, identifying

three essential characteristics: travel time, the number of stations, and quality of service. The problem was identified as multi-objective, and the resolution was carried out through MOEA, specifically the NSGA-II.

4.1.2 Simulation

The next step was to choose a transportation simulator; in this case, we decided on the MATSim simulation framework, which allowed us to simulate the movement of different agents within a road network.

The road network data for the city of Cuenca was obtained through OpenStreetMap (OSM) to conduct our experiments. We set different charging values, loaded mobility plans, and located departure and arrival points for the agents. We also identified areas with high space availability and high activity to help determine the candidate locations for charging stations. Through this study, we identified 20 possible locations for charging stations.

4.1.3 Genetic Representation

After setting up the simulator, we translated the problem into a genetic configuration. We defined a chromosome of 20 genes, where each gene represented a charging station with a value between 0 and 4, indicating the capacity or number of chargers for that station.

4.1.4 Simulation Interface

We developed a Python interface to facilitate the communication between the simulator and the genetic algorithm. Specifically, the interface can read the population of solutions and call the simulator to generate reports on various metrics, such as travel time, vehicle charging statistics, and charging times. To achieve this, we defined a set of Python classes and created a test interface to read a pre-defined station configuration or a population of solutions. This interface interacts with the simulator and obtains the necessary information for fitness value calculation through configuration files.

4.1.5 Fitness Value Calculation

Using the generated reports from the simulation interface, we calculated fitness values for each solution in the population. Specifically, we aimed to minimize travel time and the number of stations while maximizing quality of service simultaneously. These values were obtained using the travel time report, vehicle charging statistics report, and charging times report generated by the MATSim simulator.

4.1.6 DEAP Integration

We select DEAP framework for our experimentation as it allowed us to genetically define our individual and generate a random population with those characteristics. Furthermore, DEAP enables us to carry out the evolutionary process with the NSGA-II algorithm, which allows us to obtain the Pareto optimal set according to the fitness values and generate the Pareto front with these solutions. Thus, we created an interface between the genetic algorithm and the transportation simulator.

4.1.7 Dockerization

Finally, we created a Docker image that can run on any platform after obtaining the best individuals with their fitness values. Due to the high computational cost, we used an HPC to run our experiment in parallel and repeated it 12 times using a different seed.

4.2 Description of the Problem

The limited availability of fossil fuels and the environmental problems caused by greenhouse gas emissions have prompted cities to transition from conventional transportation to electromobility. Electric cars are becoming an increasingly popular option, offering accessibility and environmental benefits. However, the growing number of electric vehicles creates a demand for charging stations, which must be strategically located to meet the needs of electric vehicle users while minimizing building costs. Developing a robust charging station network is critical to promoting electromobility and incentivizing people to purchase electric vehicles. In urban areas with high charging demand and limited space,

authorities and urban planners require planning tools to allocate financial and organizational resources efficiently. Simulations are a useful tool for successful decision-making in many city planning tasks

4.3 Analysis of the Problem

Given an specific geographical area and a mobility demand, we need a framework that can simulate the plans of mobility, agents, and charging configurations. This simulator must be integrated through a set of classes that can interface between the simulator and an external application. The external application will be a framework for generating sets of candidate solutions using evolutionary algorithms. These candidate solutions should have a genetic configuration that will enable us to express the variables of the problem in the form of genes and chromosomes.

The decision variables within the genetic representation are each gene that encodes a value for the station's capacity. The goal is to find the optimal combination that minimizes the number of charging stations and travel time while maximizing the quality of service.

The trade-off between the different objective functions in our problem is an area of interest. For example, it is possible that reducing the number of charging stations could lead to an increase in travel time, while increasing the number of charging stations might improve service quality but could also increase construction and maintenance costs. We aim to analyze the relationship between these objectives to identify potential trade-offs and find an optimal balance.

4.4 Variables of the problem

The problem variables that we consider are geographic area, mobility patterns, the location of the stations, and the characteristics of electric vehicles. Below we will detail each of the variables.

4.4.1 Geographic area and Scenario Setup in MATSim

This section discusses the geographic area where we conducted our research and how we set up our scenario using MATSim. We carried out our study in the city of Cuenca, Ecuador, and utilized MATSim, an agent-based simulation framework, to model the movement of agents within the city.

To set up our scenario in MATSim, we needed to provide the network infrastructure, the mobility plans, and the respective incorporation of the charging stands to simulate the agents' movements. To achieve this, we studied the city of Cuenca and identified 20 potential locations for charging stations based on their proximity to areas with high activity, such as schools, airports, stadiums, and malls, or based on space availability.

We used Open Street Maps to extract the road network information and obtain the network infrastructure. We then converted this information into the format required by MATSim, which is XML. The resulting network contained 26361 links and represented an area of 56 km^2. In Figure 4.2, we appreciate Cuenca's street network.

Figure 4.2: Cuenca's network infrastructure. Obtained from Open Street Maps

Artifitial scenario

According to the scope of this work, we artificially adjusted the scenario as we did not use completely real data about Cuenca's transportation. We set the number of circulating vehicles to a pre-defined value and used only one type of vehicle. However, we also used real data such as peak-hour traffic, mobility plans based on places with high activity, and

availability of space. The scenario was modified artificially to verify that the simulator works optimally, rather than to conduct a mobility study in Cuenca.

4.4.2 Mobility patterns

The mobility patterns define the distribution of trips that agents execute. These patterns represent agents' behavior; for example, a mobility pattern could describe that a vehicle has made an average of two daily trips and has spent 30 minutes per trip. Mobility patterns are closely related to mobility plans since these specify the details of an individual agent's trips within a determined mobility pattern. Mobility plans contain information such as the exact starting and ending locations of each trip, the duration time of the trip, and the vehicle type.

Mobility plans

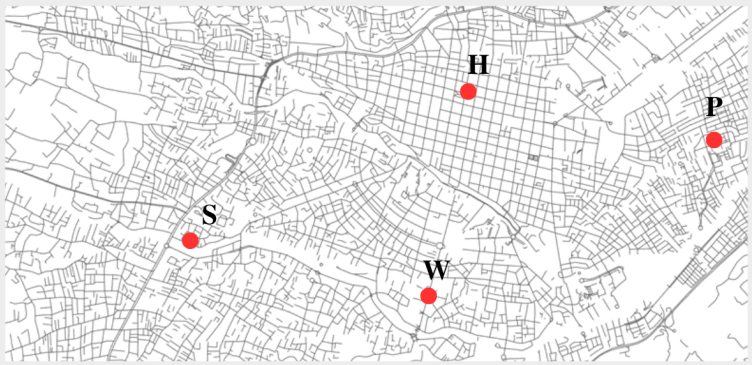
Mobility plans are defined by an XML file containing the main parameters for agents' trips in the road network, including origin location, activity, departure time, end time, activity duration for each trip, and location of selected activity. The agents execute two main trips: home-activity and activity-home. The mobility plan file is configured within the transport simulation framework. Figure 4.3 shows an example of the most important configurations in a mobility plan for each agent.

Origins and Destinations

The agent's travel origin point is the home location, assigned according to the proportion of the current population. The agents travel through the network of links and nodes until reaching their destination defined by shopping, work, personal activities and locations. Each location in the city where these activities can be carried out has an assigned weight, which was determined probabilistically. The geographical distribution considered places with a high concentration of these activities.

Times for Departures and Activity Durations

We defined times for departures and activity durations in the mobility plans. Then, we randomly assigned starting times and durations for each activity, sampling from the des-



Agent ID	Activity Type	Start time	End time	Origin (Home position)	Destination (Activity position)
6	work	.....	.....	x="721872.31" y="9679603.3"	x="721787.62" y="9678868.97"
7	shopping	6:25:56	"14:25:56"	x="721583.7" y="9679491.76"	x="722504.22" y="9679749.97"
8	personal	"08:14:46"	"17:14:46"	x="722326.64" y="9679989.77"	x="720585.99" y="9678725.58"
9	shopping	"06:08:34"	"14:08:34"	x="722333.34" y="9679949.57"	x="719829.87" y="9677874.77"
10	work	"08:11:44"	"16:11:44"	x="722140.69" y="9680188.88"	x="720509.17" y="9676438.38"

Figure 4.3: Exemplary mobility plan of agents trips.

ignated ranges for each type of activity. We set the times for departures to obey the peak traffic, with a higher concentration of trips during the morning, midday, and afternoon periods.

4.4.3 Location of Charging Stations Candidates

To determine the most suitable location for the charging stations, we relied on locations with high activity, such as schools, parking lots, stadiums, markets, shopping centers, transportation stations, airports, or places that provide the necessary space for the implementation of electric charging stations. As a result, we were able to identify a total of 20 locations that meet the requirements. In the Figure 4.4, we can see the street network of the city of Cuenca, the selected geographical area, and the location of the 20 electric charging stations. MATSim requires the network and the charging station locations as input files. Each of the 20 stations has a geographical area regarding latitude and longitude.

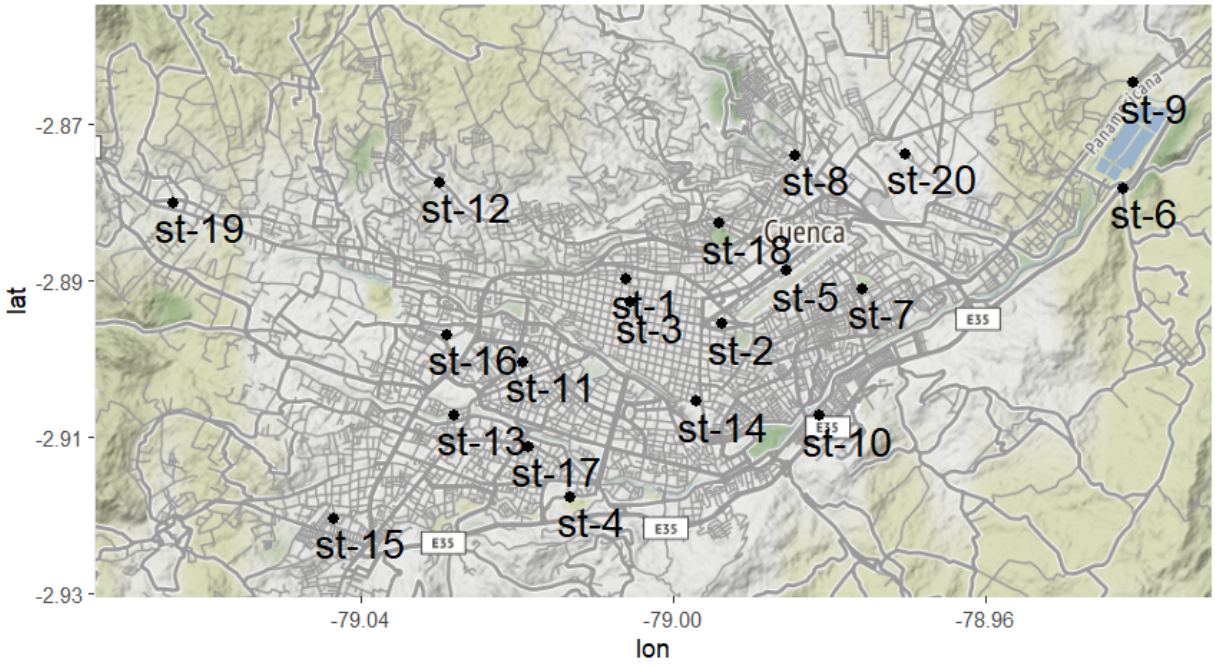


Figure 4.4: Geographical location of the 20 charging stations

4.5 Search space

The search space refers to the entire set of possible solutions for a given optimization problem. In our research, we aim to find the optimal configuration for 20 charging stations for electric vehicles, where each station has a capacity ranging from 0 to 4. Where 0 means there's no charging station, 1 represents a station with five chargers, 2 represents 10 chargers, 3 represents 15 chargers, and 4 represents 20 chargers. Determining the number of possible solutions is crucial because it helps us choose an appropriate method. We have defined the search space in two ways: the entire space of solutions and the search space that considers unique solutions.

4.5.1 All possible solutions

To find all possible solutions, we have defined the problem as a permutation problem that allows repetitions, where order matters since each variable represent a different station with its geographic location. Mathematically, we can define the search space as follows:

$$\text{Search Space} = n^k \text{ possibilities}, \tag{4.1}$$

n is the number of options for each digit (in this case, 5)

k is the number of genes (in this case, 20).

Substituting these values into the formula, we get:

$$\text{Search Space} = 5^{20} = 9.537 \times 10^{16} \text{ possible solutions} \tag{4.2}$$

4.5.2 Unique solutions

We determined the search space that considers unique solutions where order still matters as a combination problem. Mathematically, we can express the search space as:

$$\binom{20}{5} = \frac{20!}{5!(20-5)!} = 15504300 \text{ possible solutions} \tag{4.3}$$

So we have 15,504,300 possible combinations without repetition for an individual with 20 digits, where each digit can take on values of 0, 1, 2, 3, or 4.

We can see that this problem has a high-dimensional search space that is difficult to solve using traditional optimization methods. Common methods require significant computational resources and time to explore the entire search space. Therefore, we have chosen to use evolutionary algorithms, which are well-suited to solving high-dimensional optimization problems. Additionally, to speed up the optimization process, we can take advantage of parallelization techniques to distribute the computations across multiple processors or nodes.

4.6 Electric cars

In our research, one of the fundamental aspects is electric cars. MATSim enables us to simulate these agents through the EV module. Electric vehicles (EVs) move within the proposed scenario and have specific configuration files where their charging capacity at the beginning of the simulation, vehicle type, and mobility plans that indicate their routes are

specified. Once the simulation completes, it generates reports, including the status of the vehicle's battery consumption, the record of charging times, travel time, waiting time, and charging time.

4.6.1 Battery capacity

We determined the maximum charging capacity of each vehicle and station through experimentation. We simulated a scenario with 500 vehicles in motion to measure the required charge to complete the trips and fulfill the mobility plans. As a result, we set the maximum capacity of each vehicle to 40 kWh and the stations' capacity to 50 kWh. It is important to note that the vehicles start the simulation with a full charge by default, but we randomized the initial charge, which could also be incomplete.

4.6.2 Mid-size vehicles

MATSim allows us to run simulations using different types of vehicles. In our research, we have decided to configure family or personal car types specifically chosen by the "size" class. Mid-size vehicles in MATSim refer to vehicles with moderate size and capacity, such as sedans, SUVs, and station wagons. These vehicles are typically used for personal transportation.

4.6.3 EVs charging simulation

The Simulation runs with a specified number of vehicles, each EV is assigned a state of charge (SoC). Some of the electric vehicles are set with a low initial SoC, which forces them to find the best route to the nearest charging station and minimize travel time to reach their destination without running out of energy. When the vehicle arrives at a station and there are no available ports, it has to wait until a port becomes available.

4.6.4 Charge and Discharge Model

According to the study by Bischoff et al. [13], the integration of electric vehicles (EVs) into the MATSim simulation cycle was extended in several ways to better account for the behavior and needs of EVs.

First, the authors addressed the issue of vehicle routing for long-distance trips. To do this, they pre-estimated EV charging breaks at the beginning of each MATSim iteration. The overall trip route was calculated, considering the vehicle's initial state of charge (SoC) and the location of available charging infrastructure. This information determined suitable charging locations along the route, and the route was adjusted accordingly. Charging breaks were modeled as a MATSim activity, but the agent was not given a positive score for performing this action. The duration of a charging activity was determined based on the estimated required charging duration.

Second, the authors tackled the issue of energy consumption by EVs. The queue model used in MATSim allows tracking energy consumption by analyzing vehicles as they enter and leave links in the network. The authors used this information to calculate the average speed driven on a link and the vehicle's energy consumption on that link. The energy consumption model used in the study was based on previous work by the authors and calculated energy consumption as a function of average speed and road slope, using the World harmonized Light Vehicle Test Procedure (WLTP) drive cycle [13]. Fig. 4.5 shows the energy consumption of a medium-sized, compact class car, with relatively high consumption per distance at low speeds due to the significant contribution of auxiliary systems (modeled as a constant power load) and the more frequent decelerations and accelerations in stop-and-go traffic.

Finally, the authors addressed the charging logic for long-distance travel. To minimize delay and disruption to the initial travel schedule, fast charging or dynamic charging (i.e., charging while driving) were considered the most suitable technical solutions. In the study, all vehicles were assumed to start their long-distance trip with a fully charged battery, and all charging during the trip was handled by fast charging infrastructure. Upon arrival at a charging station, charging would commence immediately if a free spot was available, or the vehicle would be queued. The charging process was modeled to mimic real-world fast charging behavior, with charging happening at full speed up to 50% SoC, before decreasing linearly.

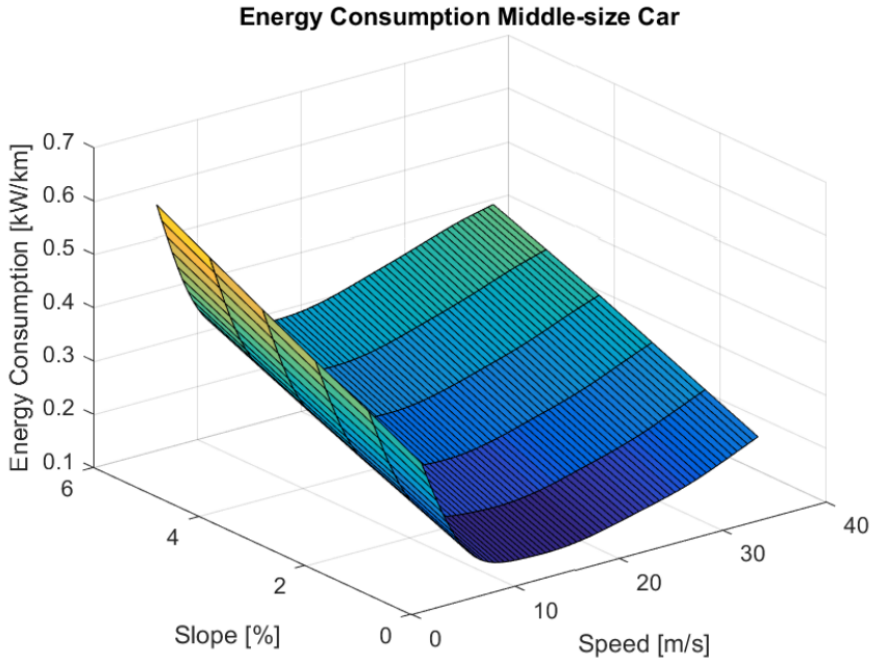


Figure 4.5: Energy consumption on a middle size car in MATSim modele based on [13]

4.7 Algorithm Design

The algorithm aimed to find an optimal balance between travel time, the number of stations, and quality of service for electric vehicles. This section describes the three objectives and their mathematical representation. It also describes the genetic configuration of individuals, which represent different combinations of charging stations, and how the NSGA-II algorithm was used to generate a diverse set of Pareto-optimal solutions. Additionally, the section discusses the role of DEAP and MATSim, two powerful tools used in the research, and how they were integrated to optimize the location and distribution of electric vehicle charging stations. Finally, the section explains the genetic operators used in the experiment and how the algorithm was tested and deployed.

4.7.1 Objective functions

In this research, we aimed to find an optimal balance between travel time, number of stations, and quality of service for electric vehicles. The trade-off between these objectives was crucial for balancing efficiency, building costs, and accessibility in our proposed scenario.

Travel Time

$(f1 = tt)$: The travel time objective refers to the time vehicles must travel to reach the destination. It computes the average time needed to complete the agents' trips requiring charging. and it is expressed by Equation 4.4 .

$$f_1 = tt = \frac{1}{n_{EV}} \sum_{i=1}^{n_{EV}} \sum_{i=1}^{L} t_{il} \tag{4.4}$$

The equation calculates the travel time for electric vehicles (EVs) that finish their journey with a charge. It considers the travel time on each link of the route, represented by t_{il}, where l is the link, and i is the vehicle. L means the total links in the route. The calculation only considers the commuting time between activities and does not include the charging or waiting time. We include the time to go to the station in the total travel time for vehicles that require recharging during their journey, but not the charging or waiting time.

Number of Stations

$(f2 = n_{St})$: The number of stations objective aimed to determine the optimal amount of stations required to ensure efficient mobility and accessibility. This objective counts the active stations, expressed as $\left(C_j \neq 0\right)$ in the representation and can be calculated using Equation 4.5.

$$f_2 = n_{St} = \sum_{j=1}^{P} D_j, \text{ where } D_j = \begin{cases} 1, & \text{if } C_j \neq 0 \\ 0, & \text{otherwise} \end{cases} \tag{4.5}$$

Quality of Service

$(f3 = QoS)$: The quality of service objective was based on the ratio between the number of vehicles charging (Ev_{charg}) and the number of vehicles in the queue ($Ev_{waiting}$). It is expressed by Equation 4.6.

$$QoS = \frac{\sum_t^T Ev_{charg}(t)}{\sum_t^T Ev_{\text{waiting}}(t)} \tag{4.6}$$

The variable t represents a time interval of 5 minutes, while T refers to the entire duration of the simulation.

4.7.2 Genetic configuration of individuals

We have genetically defined each individual or solution as a chromosome of 20 genes, where each gene represents an electric charging station. The values of the genes are integers ranging from 0 to 4, where 0 represents the absence of a charging station, and 1, 2, 3, and 4 represent charging stations with 5, 10, 15, and 20 chargers, respectively. Figure 5.5, represents an individual's population sample and configuration.

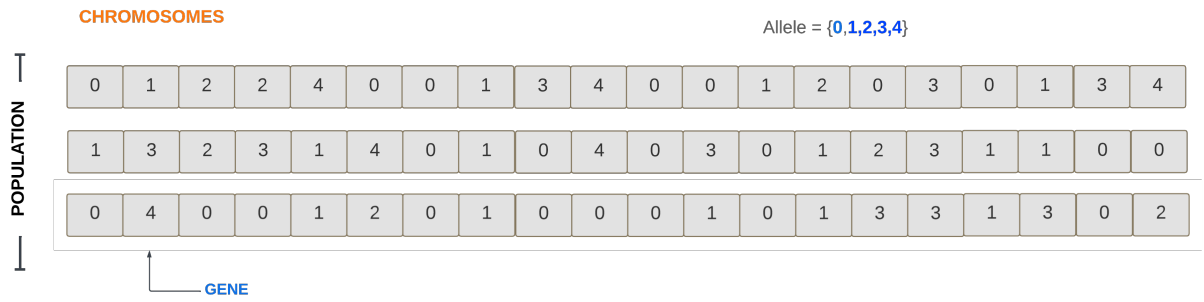


Figure 4.6: Genetic representation of a population of possible solutions

The optimization process seeks to find the best combination of charging stations that can serve a given population of electric vehicles, balancing the objectives of minimizing travel time, reducing the number of charging stations, and maximizing the quality of service.

The NSGA-II algorithm is used to generate a population of diverse individuals, where each individual represents a different combination of charging stations. The fitness of each individual is evaluated based on the multi-objective optimization function, which considers the three objectives.

The goal of the optimization process is to find a set of individuals that represents the Pareto optimal set, where no individual can be improved in one objective without sacrificing performance in another. The Pareto optimal set represents the best trade-offs between the three objectives and provides decision-makers with a range of solutions to choose from.

By using the genetic configuration of each individual, we can explore different combinations of charging stations and evaluate their performance based on the three objectives.

This allows us to identify the best set of charging stations that can serve the population of electric vehicles with minimum travel time, reduced number of charging stations, and high quality of service.

4.7.3 Fitness Values Calculation

In any optimization problem, fitness values are the key to evaluating the quality of the solutions generated by the algorithm. In the electromobility problem, the fitness values were defined to optimize the travel time, the number of charging stations, and the service quality. We defined a set of Python classes to obtain the fitness values that integrated the MATSim simulator with the genetic algorithm. MATSim simulator generated reports on travel time, vehicle charging statistics, and charging times, which were then used to calculate fitness values. In our experimentation, the fitness of each solution is given by:

$$fitness = (tt, n_{st}, QoS) \tag{4.7}$$

Where tt, the average travel time to be minimized; n_{st}, the number of charging stations to be minimized; and QoS, the quality of services to be maximized. The fitness value is represented as a tuple consisting of these three variables.

The fitness value calculation process can be broken down into the following steps:

1. Read the charging station configuration for the given solution or individual.
2. Configure the MATSim simulator with the charging station configuration.
3. Simulate the travel of the agents in the transportation network and record the necessary statistics.
4. Use the recorded statistics to calculate the fitness values for the solution or individual.

Once the fitness values were obtained for the population of solutions or individuals, the evolutionary algorithm evaluated and compared them, selecting the best individuals for the next generation.

4.7.4 Simulator Integration

Integrating the simulator with the genetic algorithm led us to design a set of Python classes that serve as a interface. These classes play a crucial role in the simulation and obtaining results. In the annexes, we can find a description of these classes. In summary, their role is:

- Receive the parameters for the simulation.
- Assign the parameters to the different variables.
- Set the transportation network and mobility plans.
- Run the scenario and set the fitness variables.
- Evaluate the individuals according to the obtained fitness.
- Define the fitness functions used by the evolutionary algorithm.
- Run the simulation through parallelization.

4.7.5 NSGA-II optimizing the Location of Charging Stations

NSGA-II (Non-dominated Sorting Genetic Algorithm II) is a popular multi-objective optimization algorithm widely used in various research fields due to its numerous advantages.

One of the main advantages of NSGA-II is its ability to generate a diverse set of Pareto-optimal solutions. NSGA-II uses a non-dominated sorting technique to rank solutions based on their dominance, ensuring that the Pareto front is always preserved throughout the optimization process.

In this research, the objectives are to minimize travel time, minimize the number of stations, and maximize the quality of service. These objectives are conflicting; finding the optimal solution requires balancing them in a trade-off. Using NSGA-II, we generated a set of Pareto-optimal solutions, representing the optimal trade-off between the objectives.

To solve the multi-objective problem, we created the Python class deap_nsgaii_ev, which calls the NSGA-II tool from DEAP. Within this class, we have defined several functions that contain the following aspects:

- The simulator object, which contains all the necessary parameters to run the scenario, and the algorithm parameters that are passed on the command line.
- The definition of individuals, population, and operators through DEAP toolboxes.
- The definition and registration of the operator we designed for mutation and the mutation parameter.
- The calculation of fitness values per individual, the parameters of crossover and mutation, and the call to the NSGA-II selection algorithm.
- The main function "eaMuPlusLambda" which carries out the generational process and returns the evolved population.
- The saving and loading function of checkpoints to resume a population's evolution from a specific point.

4.7.6 DEAP and MATSim role

This thesis explores the integration of two powerful tools, the MATSim transport simulation platform and the DEAP evolutionary computation framework, to optimize the location and distribution of electric vehicle charging stations. MATSim enables the simulation of large-scale travel behavior scenarios in urban areas, while DEAP can solve complex optimization problems using MOEAs.

DEAP

DEAP (Distributed Evolutionary Algorithms in Python) is a Python-based evolutionary computation framework that allows users to develop and compare different algorithms for solving problems using genetic algorithms. It is an open-source software package that provides a flexible set of tools for implementing genetic algorithms, genetic programming, and other evolutionary algorithms.

DEAP allowed us to define the individuals, population, and evolutionary parameters such as selection, crossover, and mutation. Furthermore, we were able to integrate DEAP with MATSim to create a simulation framework for optimizing the charging station locations.

MATsim

We used the MATSim simulator to model and simulate the mobility patterns of electric vehicles (EVs) in an artificial scenario set in Cuenca, Ecuador. MATSim is a powerful open-source agent-based transport simulation platform that enables the simulation of large-scale travel behavior scenarios of urban areas, including various modes of transport such as car, public transport, bike, and walking.

In our research, we specifically used the MATSim EV module, which is a contribution that provides additional functionality for modeling and simulating electric vehicles. This module allows us to simulate the behavior of EVs, including their charging patterns and range limitations, and to evaluate the impact of EVs on the transportation network.

By using the MATSim EV module, we were able to investigate the feasibility of deploying EV charging infrastructure in the scenario, and to evaluate the trade-offs between travel time, the number of charging stations, and the quality of service. The module enabled us to model the charging behavior of EVs, and to simulate the impact of different charging station configurations on the overall system performance.

4.7.7 DEAP Genetic Operators

In Section 2.7, we described the different types of genetic operators. DEAP offers several options for this operators, as well as the possibility of creating new ones. For our experimentation, we used a selection operator **selNSGA2()**, crossover operator **cxOnePoint()**, and additionally we have designed a mutation operator **stepMut()**. Here is a brief description of their functioning:

stepMut() function:

We designed this operator according to our needs. Its function is to increment or decrement the value of each decision variable (gene) of the chromosome. Once the gene to mutate has been established, there is a 50% probability that its value will increase or decrease. Later, we will also discuss the probability of mutation. In our experiment, each variable can have discrete values ranging from 0 to 4, so we had to adjust this operator to ensure that it does not exceed or fall below this range.

MUTATION FUNCTION

`deap.tools.stepMut(ind):`

Components	
Parameters:	**ind** – The individual to be mutated with a mutation probability
Retunrs:	The mutated individual

Table 4.1: Self designed mutation function: stepMut()

cxOnePoint() function:

Executes a one point crossover on the input sequence individuals. The two individuals are modified in place. The resulting individuals will respectively have the length of the other [15].

DEAP FUNCTION

`deap.tools.cxOnePoint(ind1, ind2)`

DEAP Components	
Parameters:	**ind1** – The first individual participating in the crossover. **ind2** – The second individual participating in the crossover.
Retunrs:	A tuple of two individuals.

Table 4.2: DEAP evolutionary tools: cxOnePoint()

selNSGA2() function

The NSGA-II selection operator is applied to a group of individuals. It is typical for the size of the input individuals to be larger than the resulting output list (k), as each individual can only appear once in the returned list. If the size of the input individuals is equal to k, the only effect will be the sorting of the population by front rank. The output list is comprised of references to the input individuals. Further information on the NSGA-II operator can be found in [95].

DEAP FUNCTION

```
deap.tools.selNSGA2(individuals, k, nd='standard')
```

DEAP Components	
Parameters:	**individuals** – A list of individuals to select from. **k** – The number of individuals to select. **nd** – Specify the non-dominated algorithm to use: 'standard' or 'log'.
Retunrs:	A list of selected individuals.

Table 4.3: DEAP evolutionary tools: selNSGA2()

4.7.8 Testing and deployment

The integration project of the simulator with the genetic algorithm requires a directory structure that contains both the Genetic Algorithm, Python classes, and the simulator. The class structure is designed as shown in the Figure 4.7. These directories allow us to execute the simulator through the console and obtain the corresponding results. Within our experiment, we have used this mechanism for testing data acquisition, integrating new modules, and components.

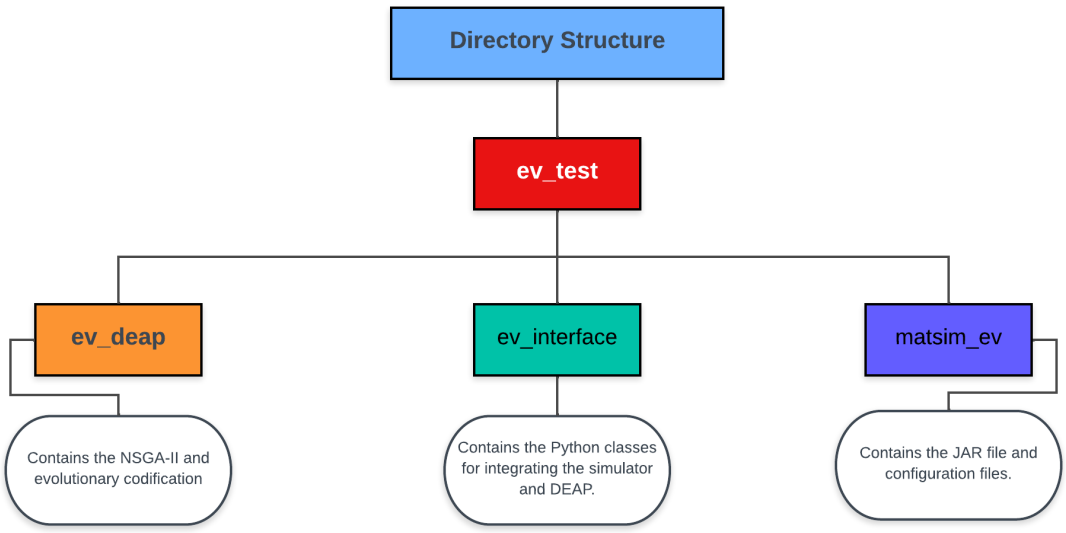


Figure 4.7: Directory structure and components for testing and deployment

4.8 Model Proposal

We have developed a model to identify the most suitable locations and capacities for charging stations. The model includes an evolutionary framework that assists in genetically defining and configuring individuals, operators, and parameters. For this study, we have chosen the DEAP framework to facilitate the configurations described above and the NSGA-II algorithm to create candidate solutions that form a subset of the already defined possible locations. The algorithm sets the activity or inactivity of a particular charging station and assigns its capacity randomly.

In the next phase, MATSim uses the population of candidate solutions to simulate the mobility of electric vehicles according to the established mobility plans. At the end of the simulation, the framework generates detailed reports, such as travel time, energy consumption, charging time, and the number of vehicles charged. The genetic algorithm calculates each candidate's fitness values in the cycle based on optimization criteria. Figure 4.8 illustrates this cycle and the model components.

The cycle continues according to the number of defined generations in the MOEA, obtaining solutions that cannot be further improved. Upon completion of the optimization process, we obtain optimal solutions with their respective active and inactive charging stations and the corresponding capacity for each station. As the final stage of this cycle, we get a set of optimal solutions that require further analysis according to specific requirements.

4.9 Analysis Method

Our analysis method section includes two key components: hypervolume calculation and optimal solution selection. The hypervolume is used to evaluate the quality of the Pareto front approximation. The optimal solutions represent a combination of adjusted decision variables searching for the best trade-offs between multiple objectives.

4.9.1 Hypervolume

Calculating the hypervolume is a useful method to assess the quality of the Pareto front approximation generated by a multi-objective optimization algorithm. As described in

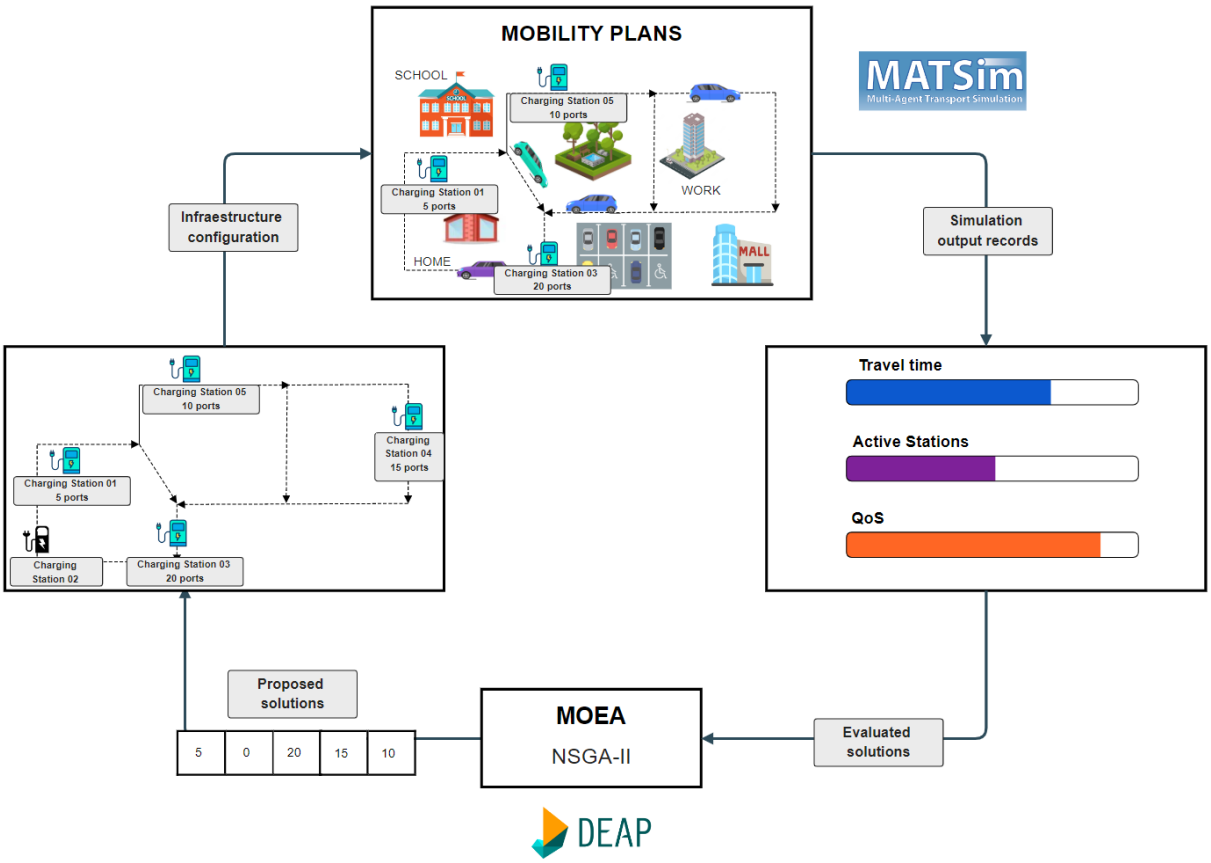


Figure 4.8: Integration of the model components

section 2, we used the hypervolume as a performance indicator in our experimentation. We utilized the PyGMO library to calculate the hypervolume, which requires a reference point and the fitness values, in our case (f1, f2, f3).

Firstly, we defined the reference point, which is outside the search space (RP = 900, 0, 5200). The first and third values represent the maximum thresholds for the travel time and quality of service objectives, respectively, which are subject to maximization. The second value represents the minimum threshold for the number of active charging stations, which is subject to minimization. Secondly, we modified the fitness values for the second objective by setting them to negative values, so that the PyGMO hypervolume function could accurately compute the values. Finally, we computed the hypervolume for each generation.

4.9.2 Optimal Solutions

In multi-objective optimization using MOEAs, the optimal solutions represent a combination of adjusted decision variables to search for the best trade-offs between multiple objectives. These decision variables represent a solution's phenotype or physical or functional characteristics. Following the criteria for obtaining the Pareto-optimal set described in Chapter 2, our research found several optimal solution sets. We executed 12 experiments, each with 50 generations, and grouped the optimal solution sets into a single set. With the help of DEAP, we obtained the Pareto-optimal set that contains only the best solutions and generates the final Pareto front.

From the entire set of optimal solutions, we will choose for further analysis and visualization on the map those that best fit the criterion of reducing building costs. That is, we will prioritize solutions with fewer active stations, a reasonable travel time, and good quality of service. The performance and advantages of each of these solutions will be analyzed, as well as how the number of chargers in each respective configuration influences the results.

4.10 Experimental Setup

This section describes the experimental setup for our study, which involved setting up evolutionary operators and parameters for the experiment and creating a reproducible research environment using Docker. We designed the mutation operator and selected the DEAP operators for selection and crossover. After conducting experiments and evaluating different parameter values, we identified optimal values for our MOEA, which are expected to balance exploration and exploitation of the search space. Finally, we set up the experimental environment using Docker, which bundles all the necessary dependencies, ensuring the same software runs consistently on different systems without compatibility issues.

4.10.1 Setting Evolutionary Operators and Parameters for an Experiment

As described in subsection 4.7.7, DEAP offers a variety of operators and the possibility of creating new ones. We designed the mutation operator `stepMut()` and selected the

DEAP operators `selNSGA2()` and `cxOnePoint()` for selection and crossover, respectively. Additionally, we set the algorithm parameters, such as the population size, the number of generations, the crossover probability, and the mutation probability.

- **Population size:** The population size typically refers to the number of individuals or solutions that exist in a population at any given generation in an evolutionary algorithm. This includes the initial population and any subsequent populations that are generated through the evolution process.
- **Mutation Probability:** It determines the probability of changing one or more genes of an individual in a population. In our experiment, we determined the optimal value to be 0.05, which is equivalent to a probability of mutating one out of the 20 genes (1/20). According to research studies, a low mutation probability can cause premature convergence, where the algorithm gets stuck in a local optimum, while a high mutation probability can result in chaotic behavior and poor convergence. The optimal mutation probabilities for evolutionary algorithms have been suggested to be between 1% and 5% [96, 97, 98].
- **Number of Generations:** In evolutionary algorithms, the number of generations refers to the number of iterations or cycles that the algorithm will run to find the optimal solution or solutions [96].
- **Number of Experiments:** According to Alaya et al., the number of experiments in optimization algorithms refers to the process of running the algorithm multiple times with different random seeds to obtain statistically significant results. This practice helps to evaluate the stability and reliability of the algorithm's performance [99].

After conducting experiments and evaluating different parameter values, we have identified optimal values for our MOEA. These values were determined through empirical studies and are expected to provide a balance between exploration and exploitation of the search space, leading to better convergence and more accurate results. Our experimental study is in line with the research efforts of several scholars who have also conducted empirical studies to find optimal parameter values for genetic algorithms [100, 101]. It is important

to note that there is no one-size-fits-all solution to parameter selection, as different problems may require different values to achieve optimal results. Table 4.4 shows the selected parameter and operator values after experimentation:

Parameter	Value
Population size	20
Crossover probability (CXPB)	0.95
Mutation probability (MUTPB)	$\frac{1}{20} = 0.05$
Number of generations (NGEN)	50
Number of experiments	12 (with different random seeds)
MATSim	12.0

Table 4.4: Selected evolutionary parameters after experimentation

4.10.2 Setting Up the Experimental Environment with Docker

Ensuring the reproducibility of experiments is a significant challenge in scientific research, especially in computational fields. One of the key solutions to this challenge is to create a software environment that is independent of the underlying system, and Docker has emerged as a popular tool for achieving this goal. Docker provides a containerization platform that enables the creation, packaging, and distribution of applications as portable containers that can run on any system supporting Docker [102].

Docker containers bundle all the necessary dependencies, libraries, and configuration files needed to run the application, ensuring that the same software runs consistently on different systems without compatibility issues [103]. Docker also allows the creation of reproducible research environments by providing a mechanism for version control of code and environment [104].

Required Components and Dependencies

This Dockerfile sets up an environment with the following components:

Components	Details
Operating System	Ubuntu 20.04
Timezone	America/Guayaquil
Utilities	wget, bzip2, git, unzip, and nano
Java Development Kit	OpenJDK 11
Python	Python 3 with pip and the following packages: lxml, pandas, matplotlib, deap
Maven	Apache Maven 3.6.3

Table 4.5: Required components and dependencies in the Dockerfile

Chapter 5

Results and Discussion

5.1 Pareto set of solutions

After running the simulation and the evolutionary process based on the NSGA-II algorithm, we obtained the Pareto Optimal Set of solutions. This set of optimal solutions resulted in 112 possible genetic configurations that optimize the three objectives (tt, N_{st}, QoS).

Out of the 112 solutions, we have decided to analyze the solutions whose configuration prioritizes the number of stations, as the lower the number of stations, the lower the construction costs. As a second decisive factor accompanying the number of stations, we have also focused on the quality of service. Finally, as a less decisive factor, we have defined travel time.

It is important to note that prioritizing the number of stations over the quality of service and travel time is based on reducing building costs. The construction and maintenance costs of stations can be high, especially if the number of stations is high. By prioritizing the number of stations, we aim to minimize these costs while maintaining an acceptable quality of service and travel time. Later on, we will analyze some solutions based on the criteria above.

From the obtained results, we can make the following considerations:

- There are few solutions with a low number of stations and high quality of service.
- There are solutions where the number of stations is low, and the quality of service is acceptable.

- We must be careful with the solutions where there is a high quality of service since we observed that this value is related to a higher number of available stations and chargers.

Figure 5.1 shows the 3D representation of Pareto Front based on the perspective of Number of Stations.

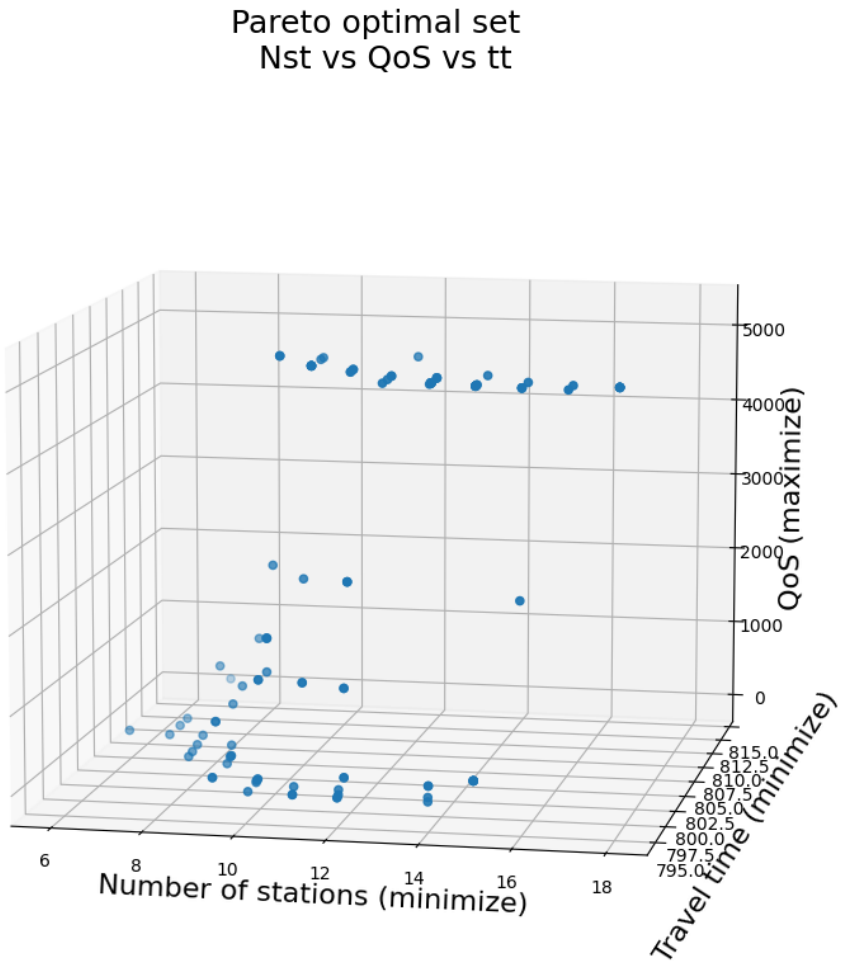


Figure 5.1: 3D plot of Pareto Optimal Set Nst vs QoS vs tt

Figure 5.2 shows a 2D plot of the Pareto Optimal set with the following configurations: travel time vs. the number of stations, and the solutions are colored according to the quality of service.

In contrast, Figure 5.3 shows a 2D plot of the Pareto Optimal set with the following configurations: quality of service vs. the number of stations, and the solutions are colored by travel time.

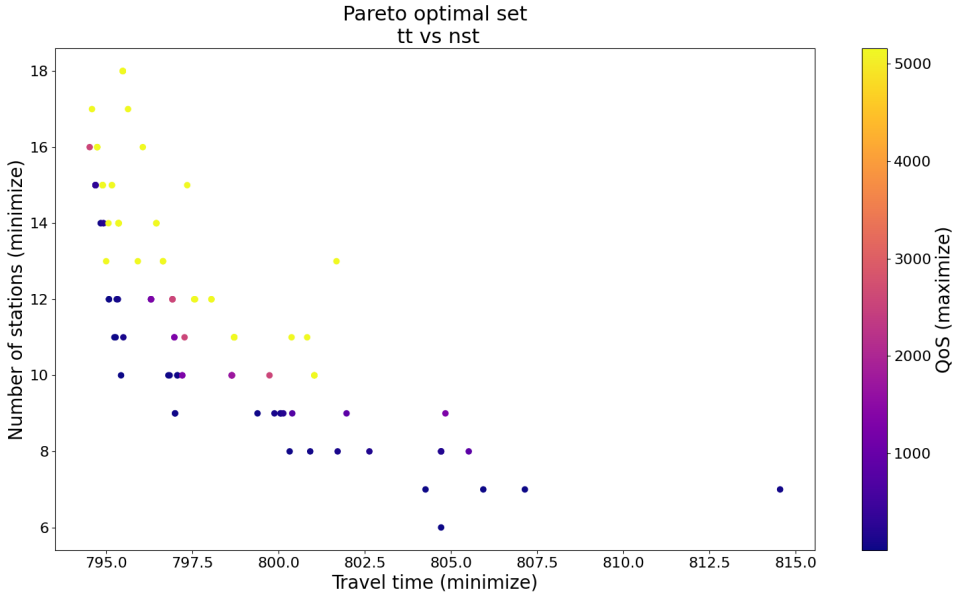


Figure 5.2: 2D plot of Pareto Optimal Set tt vs nst

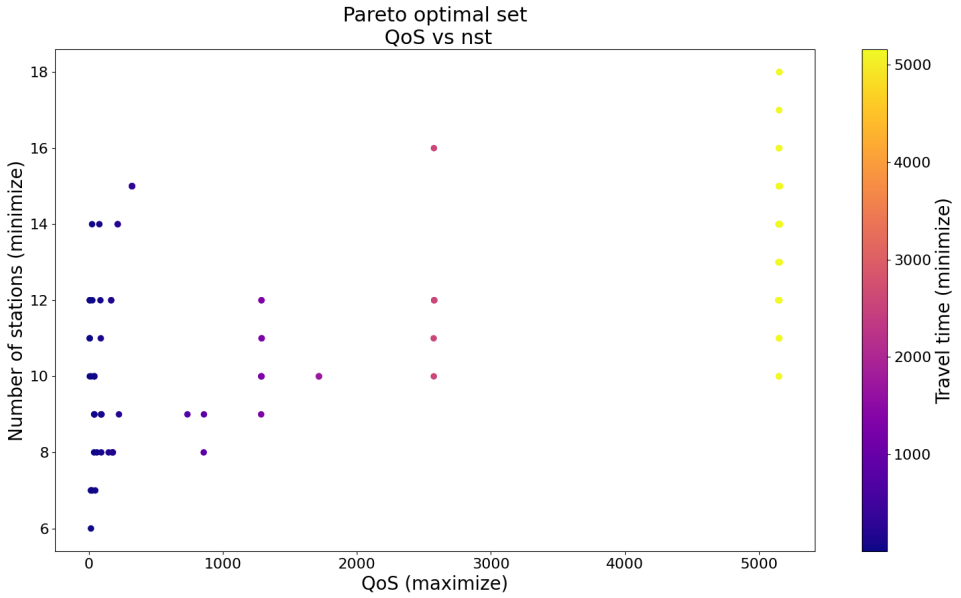


Figure 5.3: 2D plot of Pareto Optimal Set QoS vs nst

Overall, the Pareto front analysis helped us to identify the trade-offs between the different objectives and to find a set of optimal solutions that can be useful for decision-making in the planning of electric vehicle charging station placement.

5.2 Hypervolume results

As described in subsection 4.9.1, hypervolume is a quantitative measure of the quality of a set of solutions obtained from an optimization process. In this research, we executed 12 experiments, each running 50 generations, to obtain the Pareto Optimal Set of solutions. First, we calculated the hypervolume for each generation in each experiment. Then we combined the hypervolumes by generation and plotted them using boxplots. The results can be seen in Figure 5.4. The x-axis represents the generations, while the y-axis represents the hypervolumes through generations.

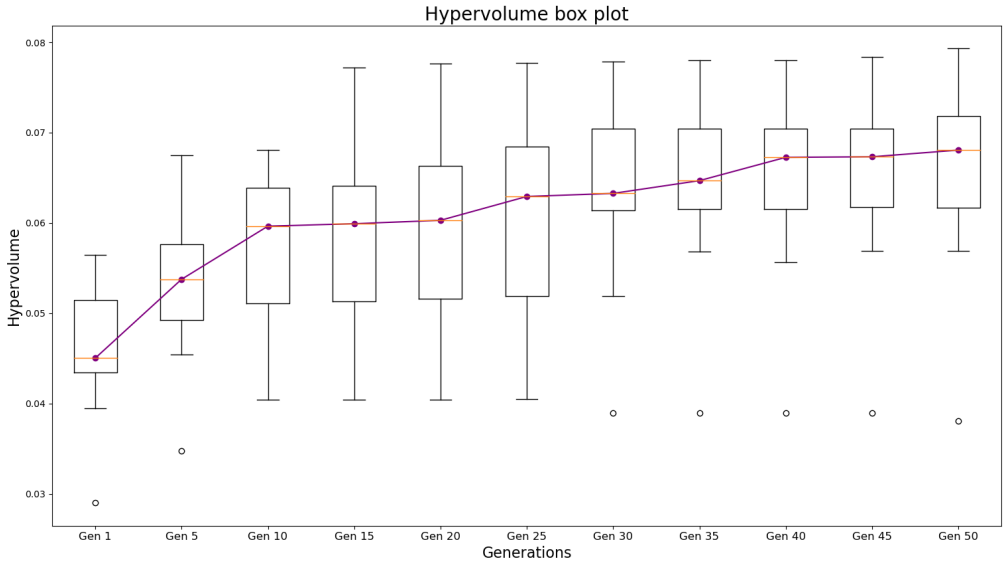


Figure 5.4: Hypervolume calculation by generations

Figure 5.4 shows variability in the hypervolumes for the initial generations, which gradually increases as the generations progress. The upper whisker in some boxplots is longer than the lower, indicating outliers with higher hypervolumes than with lower; this suggests that some solutions perform significantly better than others.

The most important observation is that the hypervolume increases steadily from the initial generations until generation 40, after which it starts to flatten out. The steady increase in hypervolume implies that the set of solutions is improving across all objectives. The gradual flattening out of the hypervolume curve from generation 40 suggests that the algorithm is approaching convergence and that the improvements in the solution set

become less significant beyond that point. Thus, it can be concluded that 50 generations are sufficient to obtain the most optimal solutions.

5.3 Analysis of the trade-off between objectives

When optimizing multiple objectives in a problem, it is common to face trade-offs between them. That is, improving one objective may result in the degradation of another. In this context, it is important to understand the relationships between the different objectives to make informed decisions. In this section, we will analyze the trade-off between our problem's three objectives using a parallel coordinate plot and a correlation analysis.

5.3.1 Analysis using parallel coordinate plot

A parallel coordinate plot is a visualization tool that allows us to plot multiple variables on the same plot. The vertical axes represent variables or attributes that are connected by lines. This graph helps us understand the relationship between multiple objectives. By plotting each objective on a separate axis, parallel coordinate plots can help visualize the trade-offs between objectives.

In our case, we plotted the three fitness values: travel time, number of stations, and quality of service. Then, we loaded the fitness values of the Pareto optimal set and normalized them. From Figure 5.5, we can observe a trade-off between travel time and the number of stations, where the travel time increases when the number of stations is lower. However, the correlation between the number of stations and service quality is unclear. Although we see some intersections, the correlation is not straightforward.

5.3.2 Correlation analysis

We performed a correlation analysis to obtain a more accurate understanding of the relationships between the objectives. As Figure 5.6 shows, there exists a negative correlation of -0.81 between travel time and the number of stations, indicating a trade-off between these objectives. We also found a positive correlation of 0.58 between the number of stations and the quality of service, but the correlation is not necessarily negative. Additionally, there is a negative correlation of -0.16 between travel time and the quality of service. Overall,

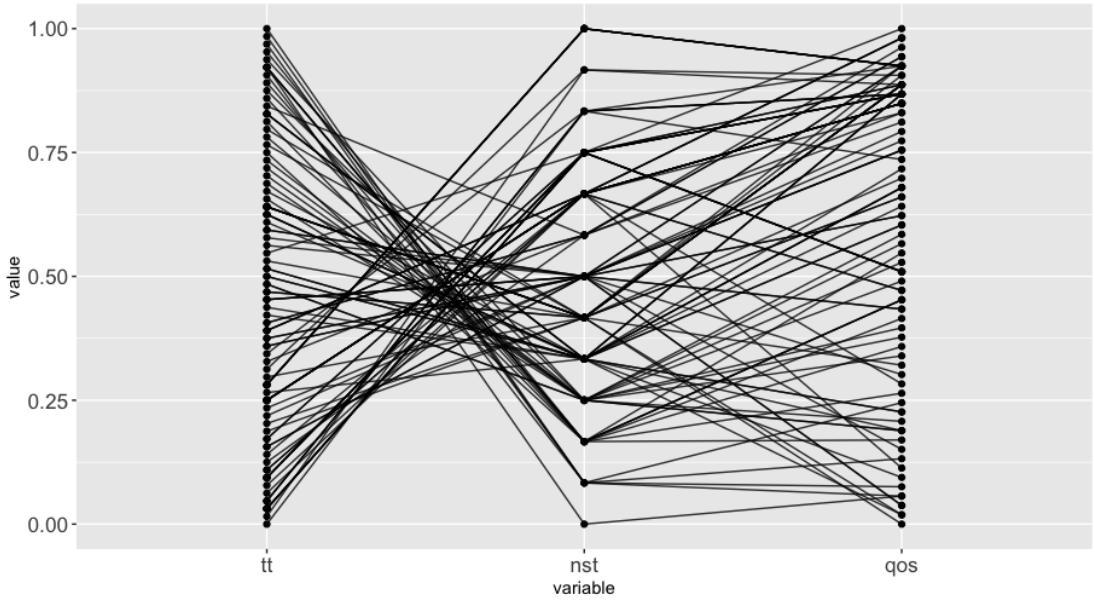


Figure 5.5: Coordinate plot graphical analysis

the correlation analysis provides a more quantitative way to understand the relationships between the objectives in our problem.

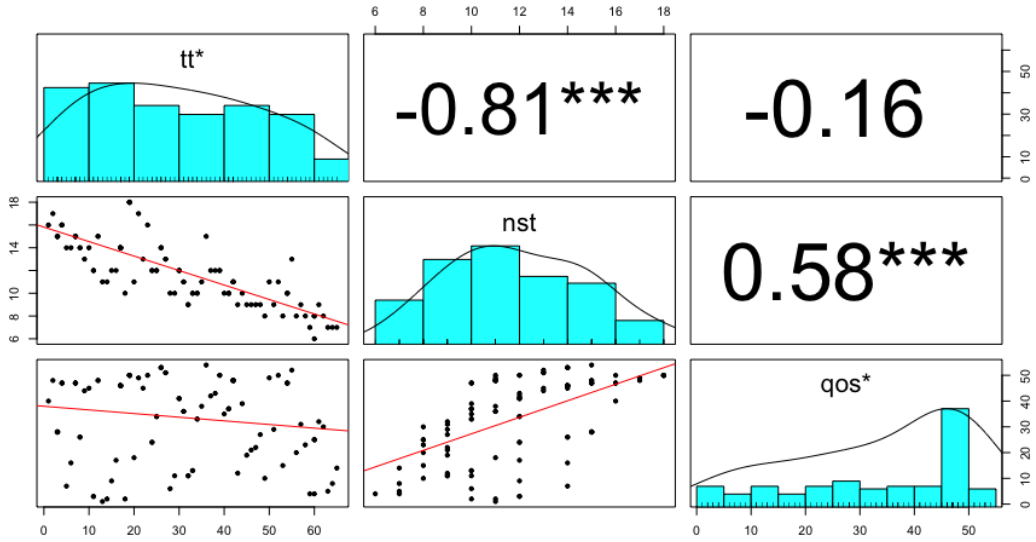


Figure 5.6: Quantitative analysis of objectives correlation

5.4 Analysis and plotting of solutions

Using R, we have configured a script that allows us to visualize on the map of Cuenca the solutions that we have in the Pareto optimal set. In this case, we have selected some examples prioritizing the number of stations. The active stations are colored blue, while the inactive stations are colored red. See figure 5.7.

5.4.1 Analysis of solution 107

The first solution we will analyze is solution 107 because it has the minimum number of stations in its configuration and is the only solution that fully minimizes this objective. Figure 5.7 shows the stations located on the map of Cuenca, where each active station is colored in blue with its respective station code and the number of chargers.

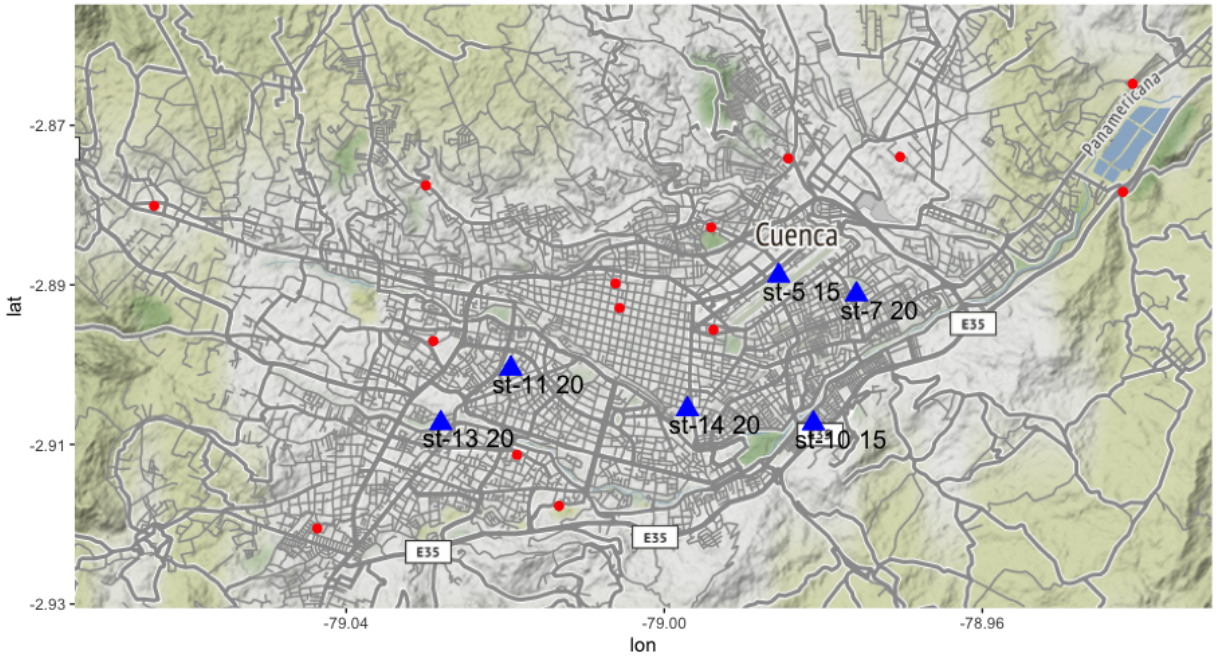


Figure 5.7: Spatial location of the stations on the Cuenca's map (solution 107)

Table 5.1 presents the genetic configuration and fitness values for solution 107. Although the travel time required for vehicles in this configuration is reasonable and falls within a range similar to that of other solutions with a higher number of stations, the quality of service offered by this arrangement is not optimal due to the reduced number of stations. However, this is an interesting solution for our study as it would optimize con-

struction costs. Additionally, this configuration requires a considerable number of chargers per station, either 15 chargers or the maximum amount of 20 per station.

Genetic configuration (solution 107)			
0 0 0 0 3 0 4 0 0 3 4 0 4 4 0 0 0 0 0 0			
tt	**nst**	**QoS**	**Chargers**
804.7112334	6	150.2686567	110

Table 5.1: Configuration of solution 107

Figure 5.8 shows the 3D spatial localization of solution 107 in the Pareto Optimal Set according to the fitness values.

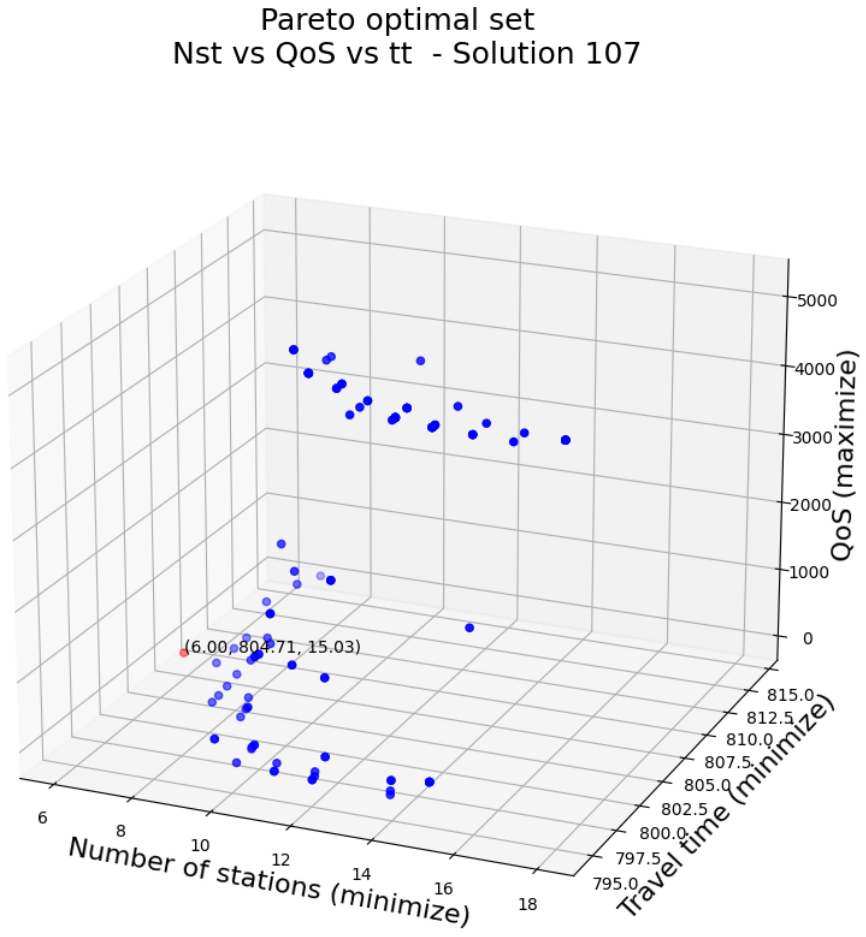


Figure 5.8: 3D Pareto optimal set - Spatial localization of Solution 107

5.4.2 Analysis of solution 112

The next solution to be analyzed also presents a minimum station configuration, in this case with 7 stations. Figure 5.9 shows how the stations are distributed on the map of Cuenca.

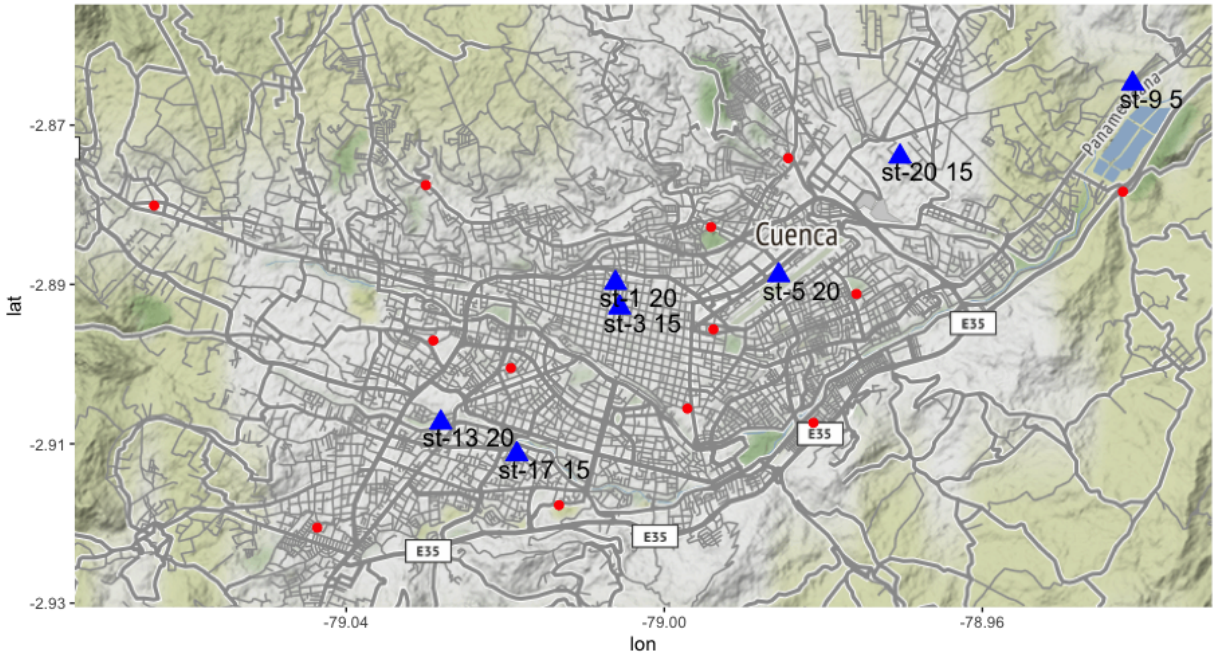


Figure 5.9: Spatial location of the stations on the Cuenca's map (solution 112)

Table 5.2 shows the genetic configuration of this solution, as well as the fitness values and the number of chargers. This configuration offers us a reasonable travel time wich is a higher than solution 107 and an acceptable quality of service. This is due to the fact that, equal as solution 106, the number of stations is prioritized over the other two objectives. The number of chargers in this case is mostly either 15 or 20 per station. However, in Figure **??**, we can see that station 9 has 5 chargers, which is because it is a peripheral station.

Genetic configuration (solution 112)			
4 0 3 0 4 0 0 0 1 0 0 0 4 0 0 0 3 0 0 3			
tt	**nst**	**QoS**	**Chargers**
814.5467789	7	46.96330275	110

Table 5.2: Configuration of solution 112

Figure 5.10 shows the 3D spatial localization of solution 112 in the Pareto Optimal Set according to the fitness values.

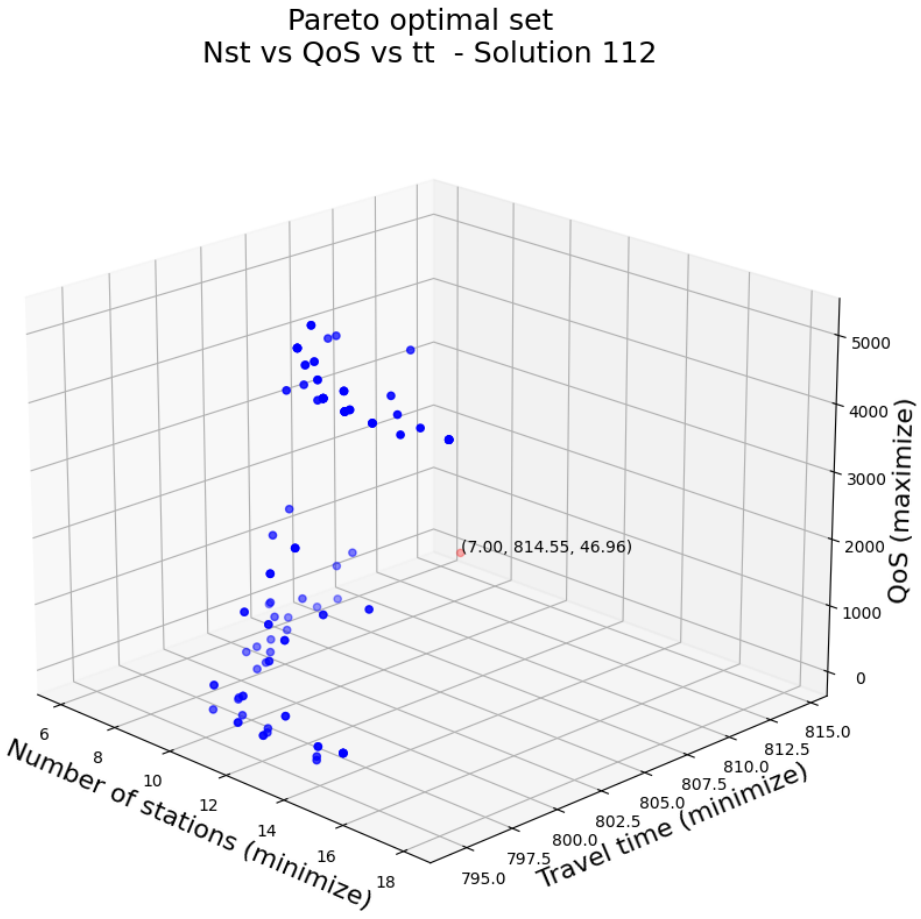


Figure 5.10: 3D Pareto optimal set - Spatial localization of Solution 112

5.4.3 Analysis of solution 106

Finally, solution 106 offers a configuration of 8 active charging stations that can be visualized in Figure 5.11. We continue to prioritize the criterion of a low number of charging stations.

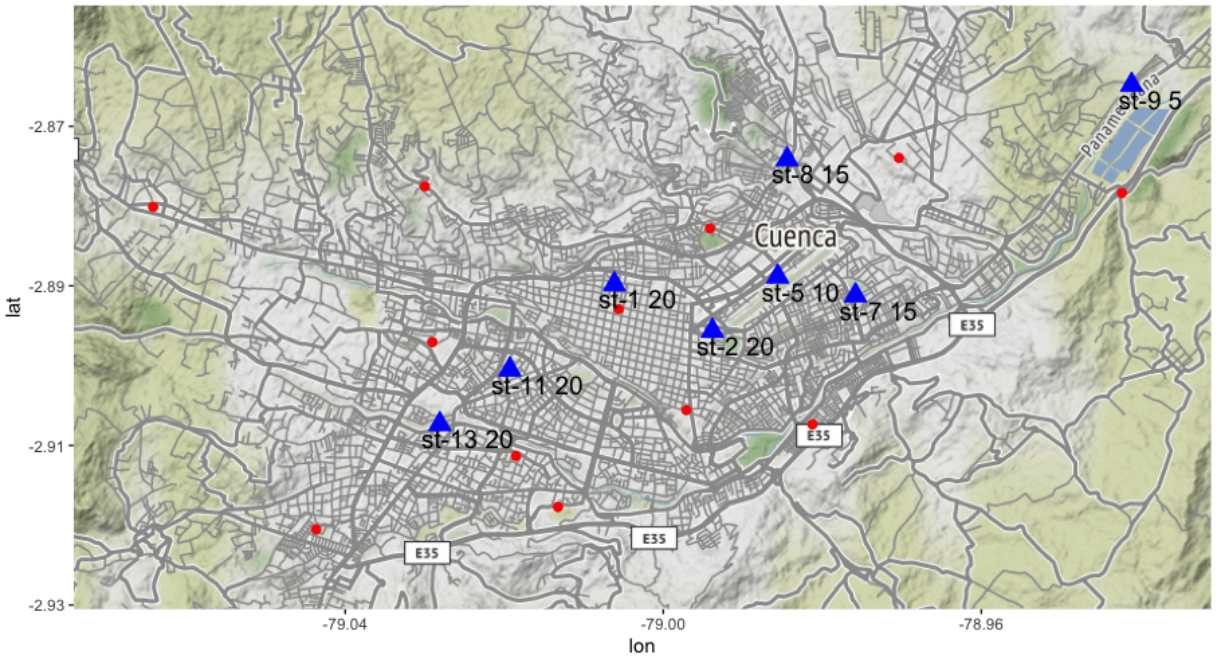


Figure 5.11: 3D Pareto optimal set - Spatial localization of Solution 106

Table 5.3 shows the configuration values for this solution, and we can observe that in this case, unlike the previous solutions, the quality of service is superior. This is due to the availability of more stations and chargers, which reduce waiting times within the stations. The travel time remains acceptable and is similar to the 6-station configuration, while the total number of chargers increases to 125.

Genetic configuration (solution 106)			
4 4 0 0 2 0 3 3 1 0 4 0 4 0 0 0 0 0 0 0			
tt	**nst**	**QoS**	**Chargers**
804.7109097	8	177.3448276	125

Table 5.3: Configuration of solution 106

Figure 5.12 shows the 3D spatial localization of solution 112 in the Pareto Optimal Set according to the fitness values.

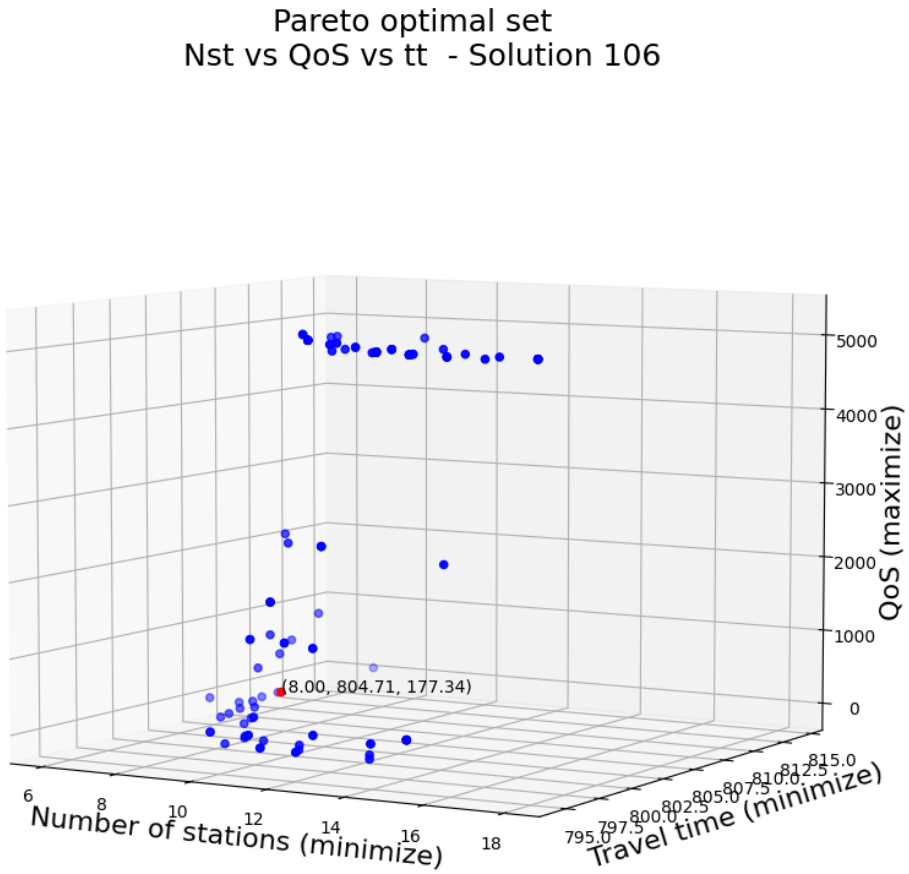


Figure 5.12: 2D plot of Pareto Optimal Set tt vs nst

5.5 Simulation of the solutions in Via-Simunto

Vía-Simunto is a software that provides us with tools for simulating transportation systems [105]. With Vía-Simunto, we can import and visualize the data generated by MATSim, allowing us to simulate the movement of vehicles on the road network of Cuenca, according to established mobility plans and the configuration of charging stations.

Figure 5.13 shows a snapshot of the mobility of 500 vehicles at 7 am, with a configuration of 6 charging stations. The green arrows represent the agents (vehicles), and the red dots are the charging stations taken from solution 106.

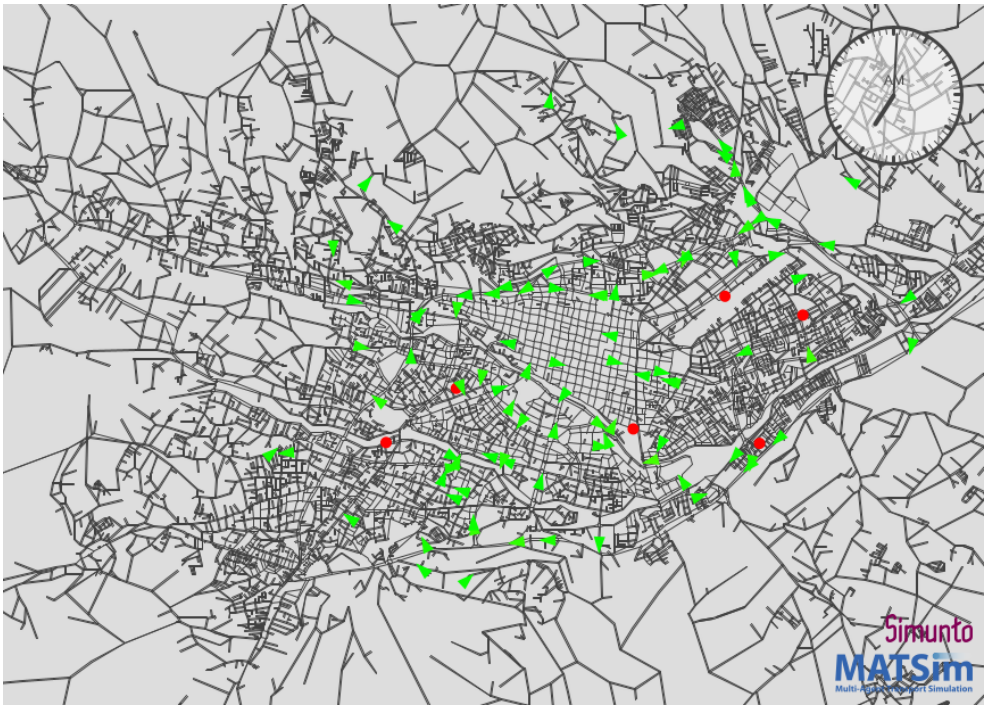
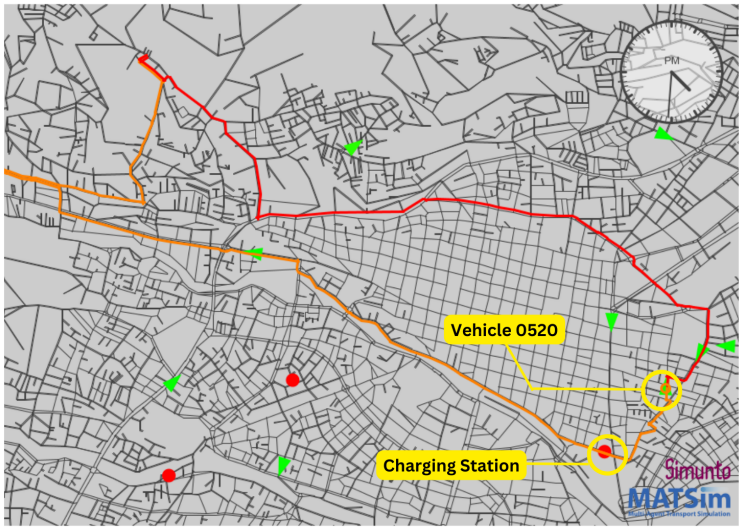
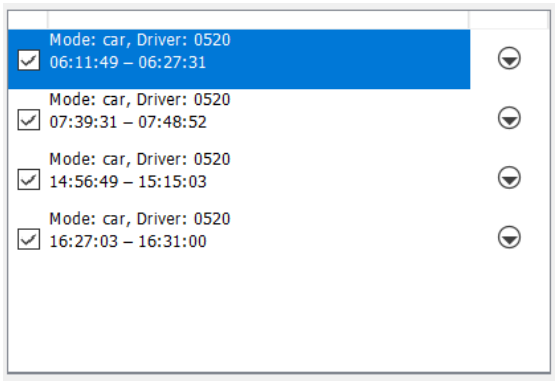


Figure 5.13: Vehicles movement over the Cuencas Map)

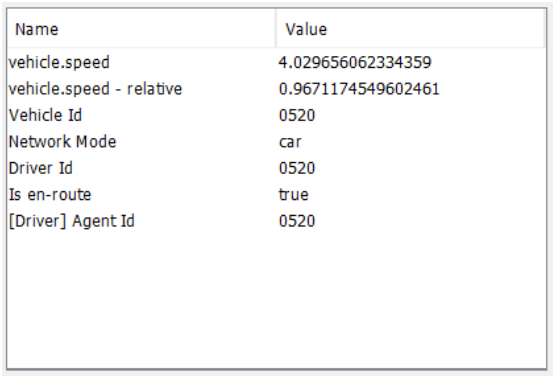
Figure 5.14a shows the travel trajectory of vehicle 0520 until 3 pm, which needed to pass through one of the charging stations. We can also see one of the vehicle's trajectories highlighted in red. Figure 5.14b shows the trips by the vehicle during the day, and Figure 5.14c shows some vehicle data such as ID, speed, type of car, etc.



(a) Travel trajectory of vehicle 0520 within the map of Cuenca.



(b) Table of vehicle travel trajectories.



Name	Value
vehicle.speed	4.029656062334359
vehicle.speed - relative	0.9671174549602461
Vehicle Id	0520
Network Mode	car
Driver Id	0520
Is en-route	true
[Driver] Agent Id	0520

(c) Table of vehicle attributes.

Figure 5.14: Simulation of vehicle trips through Vía-Simunto.

Chapter 6

Conclusions

The study addresses the complex challenge of determining the optimal locations and capacities of electric vehicle charging stations in the Cuenca scenario. By integrating the DEAP evolutionary framework with the MATSim traffic simulator, we have developed an innovative approach that considers the interdisciplinary nature of the problem and effectively balances conflicting objectives. Through analyzing various mobility scenarios and collecting relevant data, we have achieved a comprehensive understanding of the charging infrastructure. Furthermore, the integration of evolutionary algorithms and traffic simulation allows us to evaluate different solution options and provide recommendations for improving electric vehicle charging infrastructure performance and implementation. This research contributes to the advancement of transportation engineering, electrical engineering, computer science, and data analytics but also promotes the adoption of electric cars and fosters the development of sustainable transportation systems. Below we expose our main conclusions and to outline the possible tasks that can emerge from this research study.

1. Integration of DEAP and MATSim: We successfully integrated the DEAP evolutionary framework with the MATSim traffic simulator by developing a Python-based interface. This integration enabled a comprehensive analysis of mobility scenarios and collecting relevant data, laying the foundation for determining optimal charging station locations and capacities.

2. Multi-objective Optimization: By utilizing the NSGA-II algorithm within the DEAP framework, we achieved a multi-objective optimization that effectively balanced conflicting objectives. Factors such as travel time, the number of charging stations, and

service quality were considered, resulting in a set of 112 optimal solutions plotted on the Cuenca scenario map.

3. Performance Evaluation: Through simulations in the MATSim environment, we evaluated the performance of the electric vehicle charging infrastructure. By considering various factors, including vehicle types, road networks, peak traffic hours, and areas of high human activity, we obtained valuable insights into the effectiveness of the infrastructure.

4. Visualization of Solutions: Through the use of Vía-Simunto, we successfully visualized the movement of agents based on mobility plans and the selected charging station configuration. This visualization allowed us to analyze and understand the obtained solutions for the optimal location and capacity of electric vehicle charging stations in Cuenca.

5. Recommendations for Improvement: The thorough analysis of simulation results allowed us to provide valuable suggestions for future improvements or implementations. These recommendations will guide decision-makers and stakeholders involved in developing electric vehicle charging infrastructure, facilitating more efficient and effective implementation strategies.

6.0.1 Future Work

The integration of MATSim with NSGA-II has shown promising results in simulating and optimizing large-scale travel behavior scenarios. However, there are still several areas where the research can be extended.

One of the most important future directions is the use of real data on transportation in Cuenca, including the number of cars circulating, the different types of transport, and the actual traffic flows. By incorporating real-world data, we can create more accurate simulations that better reflect the behavior of travelers in the area.

Additionally, we can further improve the simulation by adding other objectives such as building costs, energy efficiency, and environmental sustainability. By incorporating these additional objectives, we can better evaluate the trade-offs between different transportation options and help inform policy decisions.

Finally, we can explore creating additional scenarios that consider the use of electric buses as a public transportation option. This could involve evaluating the impact of electric buses on the transportation system, as well as analyzing their potential benefits in terms of environmental sustainability and energy efficiency. By incorporating this additional factor into the simulation, we can better evaluate the trade-offs between different transportation options and provide more comprehensive insights for decision-making

6.0.2 Limitations

This research faced several limitations throughout the development of the project. One of the main limitations was related to the version of the MATSim simulator used. Initially, we attempted to use version 0.14.0, which was the most recent version at the time of the study. However, we encountered a bug in the source code that prevented us from generating a correct charging stats file. This problem was further compounded by compatibility issues between the different versions of Java and Python.

Another limitation was the computational resources required to run the experiments. Due to the complexity and size of the simulation, it was necessary to run the experiments using Docker in a cluster for parallelization. This required a significant amount of computing power, which was not readily available.

In addition, the study focused solely on the travel time, number of stations, and quality of service as the objectives to optimize. Other important factors, such as building costs, energy efficiency, and the number of cars circulating in Cuenca were not considered. Future research could explore the integration of these factors into the optimization process.

Overall, while the study was successful in integrating the simulator with the genetic algorithm and obtaining the Pareto optimal set with the configurations, these limitations highlighted the need for further research and development in this area.

Bibliography

[1] M. M. Soares and G. E. Vieira, "A new multi-objective optimization method for master production scheduling problems based on genetic algorithm," *The International Journal of Advanced Manufacturing Technology*, vol. 41, no. 5-6, pp. 549–567, 2009.

[2] M. Kumar, D. Husain, N. Upreti, D. Gupta *et al.*, "Genetic algorithm: Review and application," *Available at SSRN 3529843*, 2010.

[3] D. E. Goldberg, *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley Professional, 1989.

[4] C. A. C. Coello, G. B. Lamont, D. A. Van Veldhuizen *et al.*, *Evolutionary algorithms for solving multi-objective problems*. Springer, 2007, vol. 5.

[5] I. E. Agency, *Global EV outlook 2021: Driving the transition to electric mobility*. Paris: International Energy Agency, 2021.

[6] S. N. Zealand, "Hybrid tech: How does it work?" https://www.suzuki.co.nz/cars/suzuki-life/news/article/hybrid-tech-how-does-it-work/592920, 2021, [Accessed: April 19, 2023].

[7] J. Tanomaru, "Motivação, fundamentos e aplicações de algoritmos genéticos," in *II Congresso Brasileiro de Redes Neurais*, 1995, pp. 373–403.

[8] A. Kumar, "Encoding schemes in genetic algorithm," *International Journal of Advanced Research in IT and Engineering*, vol. 2, no. 3, pp. 1–7, 2013.

[9] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*. Springer, 2015.

[10] A. Lavin, "A pareto front-based multiobjective path planning algorithm," *arXiv preprint arXiv:1505.05947*, 2015.

[11] M. T. Emmerich and A. H. Deutz, "A tutorial on multiobjective optimization: fundamentals and evolutionary methods," *Natural computing*, vol. 17, pp. 585–609, 2018.

[12] K. Deb, N. Srinivas, and S. Bandaru, "Genetic algorithms," in *Handbook of Genetic Programming Applications*. Cham: Springer, 2019, pp. 79–116.

[13] J. Bischoff, F. J. Márquez-Fernández, G. Domingues-Olavarría, M. Maciejewski, and K. Nagel, "Impacts of vehicle fleet electrification in sweden–a simulation-based assessment of long-distance trips," in *2019 6th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*. IEEE, 2019, pp. 1–7.

[14] A. Horni, K. Nagel, and K. W. Axhausen, *The Multi-Agent Transport Simulation MATSim*. Ubiquity Press, 2016.

[15] F.-A. Fortin, F.-M.-D. Rainville, M.-A. Gardner, M. Parizeau, and C. Gagné, "DEAP framework documentation," https://deap.readthedocs.io/en/master/, 2012, accessed: [Insert date here].

[16] G. A. D. M. de Cuenca. (fecha desconocida) Se realizó presentación de plan de electromovilidad e-cuenca. [Online]. Available: https://www.cuenca.gob.ec/content/se-realizo-presentacion-de-plan-de-electromovilidad-e-cuenca

[17] E. Baker, H. Y. Chon, and G. A. Keoleian, "Environmental impacts of electric vehicles in the united states," *Environmental Science Technology*, vol. 55, no. 2, pp. 855–866, 2021.

[18] International Energy Agency, *Global EV Outlook 2021: Scaling up the transition to electric mobility*. Paris, France: International Energy Agency, 2021. [Online]. Available: https://www.iea.org/reports/global-ev-outlook-2021

[19] BloombergNEF, "Bloombergnef electric vehicle outlook 2021," 2021. [Online]. Available: https://about.bnef.com/electric-vehicle-outlook/

[20] K. S. Kurani and T. S. Turrentine, "The future of electric vehicles," *Annual Review of Environment and Resources*, vol. 44, pp. 289–316, 2019.

[21] L. Lu, X. Han, J. Li, J. Hua, and M. Ouyang, "A review on the key issues for lithium-ion battery management in electric vehicles," *Journal of Power Sources*, vol. 408, pp. 166–180, 2020.

[22] X. Zhang, G. Li, J. Chen, and Z. Liu, "Life cycle assessment of electric vehicles: a review," *Renewable and Sustainable Energy Reviews*, vol. 107, pp. 1–13, 2019.

[23] 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.

[24] M. K. Hidrue, G. R. Parsons, W. Kempton, and M. P. Gardner, "Vehicle-to-grid power: Battery, hybrid, and fuel cell vehicles as resources for distributed electric power in california," *Applied Energy*, vol. 261, p. 114304, 2020.

[25] C. Bauer, E. Helmers, and L. Hirth, "Electromobility: The impact of regulatory and policy measures on the development of electric vehicles," *Journal of Cleaner Production*, vol. 212, pp. 716–728, 2019.

[26] R. A. Acheampong, R. J. Hafner, and J.-P. Belieres, "Electric vehicle charging infrastructure deployment in germany: A quantitative analysis," *Transportation Research Part A: Policy and Practice*, vol. 129, pp. 163–184, 2019.

[27] K. Schmidt-Rohr, "Lithium-ion batteries: Solid-electrolyte interphase," *Nature*, vol. 571, no. 7765, pp. 478–479, 2019.

[28] W. Liu, F. Zhao, L. Guo, C. Liu, Q. Zhu, and B. Zhang, "Life cycle environmental impact of electric vehicles in china," *Environmental Science and Pollution Research*, vol. 27, no. 15, pp. 17 668–17 678, 2020.

[29] P. Gao, T. Tang, X. Cheng, H. Chen, and S. Zhu, "Impact of electric vehicle adoption on urban travel demand: A case study of beijing," *Transportation Research Part D: Transport and Environment*, vol. 94, p. 102783, 2021.

[30] X. Li, P. Liu, Y. Li, H. Li, B. Li, and X. Chen, "Integrated operation of electric vehicle charging station and renewable energy source considering uncertainty," *Energy*, vol. 230, p. 120662, 2021.

[31] X. Wang, M. Hu, Y. Shi, and X. Li, "Research on the coordinated development of new energy vehicle and charging infrastructure in china," *Renewable and Sustainable Energy Reviews*, vol. 123, p. 109745, 2020.

[32] S. Sivanandam and S. Deepa, "Genetic algorithms," in *Introduction to genetic algorithms*. Springer, 2008, pp. 15–37.

[33] J. Thanura, R. D. Rathnayake, S. Dewasurendra, and L. Rajakaruna, "Generating optimised mrp lot sizes using genetic algorithm: Considering supplier deals."

[34] W. Ip, Y. Li, K. Man, and K. Tang, "Multi-product planning and scheduling using genetic algorithm approach," *Computers & Industrial Engineering*, vol. 38, no. 2, pp. 283–296, 2000.

[35] S. Chakraborty, S. Ghosh, and S. Das, "Effective population initialization techniques for genetic algorithms: A review," *Information Sciences*, vol. 501, pp. 20–44, 2019.

[36] J. Singh and S. S. Yadav, "Comparative analysis of selection techniques in genetic algorithm for engineering optimization," *Applied Soft Computing*, vol. 75, pp. 517–531, 2019.

[37] Y. T. Le and K. Nakamatsu, "Genetic algorithms applied in dynamic environments: A survey," *Applied Soft Computing*, vol. 77, pp. 275–294, 2019.

[38] W. Hua, J. Li, Y. Li, J. Li, and Z. Gao, "Dynamic mutation probability strategy for multi-objective genetic algorithm," *Engineering Applications of Artificial Intelligence*, vol. 80, pp. 88–101, 2019.

[39] X.-y. Tan, J.-h. Hu, X.-y. Zhao, R. Zhang, W. Yang, and X. Chen, "An improved genetic algorithm for solving the maximum clique problem," *Engineering Applications of Artificial Intelligence*, vol. 78, pp. 121–130, 2019.

[40] R. Sharma, M. Dwivedy, R. Shankar, and S. S. Mahapatra, "A hybrid genetic algorithm for optimization of machining parameters for sustainable manufacturing," *Journal of Cleaner Production*, vol. 231, pp. 440–453, 2019.

[41] K. V. Krishna, "Genetic algorithms: An overview," *Journal of Physics: Conference Series*, vol. 1244, no. 1, p. 012073, 2019.

[42] A. Rad, J. Aghaei, S. M. Mousavi, and N. Javadian, "Application of genetic algorithms in scheduling," *Computers & Industrial Engineering*, vol. 136, pp. 691–703, 2019.

[43] R. Wadhwa and G. Singh, "Genetic algorithms in transportation engineering: A review," *Transportation Research Part C: Emerging Technologies*, vol. 103, pp. 392–412, 2019.

[44] A. Sobhani and S. Hasani, "Optimizing urban transport: A hybrid method based on genetic algorithm and artificial bee colony algorithm," *Sustainable Cities and Society*, vol. 44, pp. 246–257.

[45] W. Chen, S. Chen, and M. Wang, "A multi-objective genetic algorithm based on pareto boundary detection for energy-efficient task scheduling," *Applied Soft Computing*, vol. 98, p. 106898, 2021.

[46] A. Jain, P. Vashistha, P. Kumar, and S. Chandra, "Multi-objective optimization of grinding process using hybrid genetic algorithm," *Computers & Industrial Engineering*, vol. 146, p. 106566, 2020.

[47] S. K. Singh, S. Gupta, and J. C. Bansal, "A new hybrid genetic algorithm for the optimization of surface grinding process," *Expert Systems with Applications*, vol. 116, pp. 169–180, 2019.

[48] J. Miao, Y. Zhang, X. Wang, and B. Liu, "A multi-objective genetic algorithm based on random local search for the vehicle routing problem," *IEEE Access*, vol. 8, pp. 197 679–197 689, 2020.

[49] S. Li, H. Xu, and X. Du, "Multichannel image processing algorithm based on genetic algorithm," *Journal of Physics: Conference Series*, vol. 1341, no. 1, p. 012092, 2019.

[50] H. Li, Q. Li, and Y. Li, "A genetic algorithm based on phenotype diversity and nearest-neighbor for the job shop scheduling problem," *Computers & Industrial Engineering*, vol. 156, p. 107277, 2021.

[51] A. Gomez and D. Wilson, "Optimizing protein folding energy landscapes using genetic algorithms," *PLoS computational biology*, vol. 16, no. 6, p. e1007948, 2020.

[52] D.-H. Lee and B.-S. Yang, "A genetic algorithm-based feature selection method for text classification," *Journal of Ambient Intelligence and Humanized Computing*, vol. 9, no. 6, pp. 1801–1811, 2018.

[53] Z. Michalewicz, *Genetic algorithms + data structures = evolution programs*. Springer Science Business Media, 2013.

[54] R. L. Haupt and S. E. Haupt, *Practical genetic algorithms*. John Wiley Sons, 2019.

[55] O. Kramer, "Genetic algorithms," in *Genetic algorithm essentials*. Springer, 2017, pp. 11–19.

[56] D. Whitley, "A genetic algorithm tutorial," *Statistics and Computing*, vol. 4, no. 2, pp. 65–85, 1994.

[57] I. Hussain, M. A. Sabir, and S. Hussain, "A novel hybridization of grey wolf optimizer and genetic algorithm for constrained optimization problems," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 11, pp. 4757–4773, 2020.

[58] S. N. Sivanandam and S. N. Deepa, *Introduction to Genetic Algorithms*. Springer, 2020.

[59] M. Ali, S. Asghar, S. A. Butt, W. Shahzad, and J.-H. Kim, "Performance analysis of objective functions in genetic algorithm for vehicle routing problem," *Computers & Industrial Engineering*, vol. 145, p. 106545, 2020.

[60] R. Cheng, Q. Zhang, and Y. Zhu, "A novel constraint handling method for moea/d based on multi-objective surrogate constraint satisfaction," *Soft Computing*, vol. 24, no. 19, pp. 14 711–14 730, 2020.

[61] K. Deb, *Multi-objective optimisation using evolutionary algorithms: an introduction*. Springer, 2011.

[62] J. D. Knowles and D. W. Corne, "On metrics for comparing nondominated sets," in *Congress on Evolutionary Computation (CEC-2002)*. Piscataway, NJ: IEEE Press, 2002, pp. 711–716.

[63] A. Custódio, M. Emmerich, and J. Madeira, "Recent developments in derivative-free multiobjective optimization," *Computational Technology Reviews*, vol. 5, no. 1, pp. 1–31, 2012.

[64] Y. Yusoff, M. S. Ngadiman, and A. M. Zain, "Overview of nsga-ii for optimizing machining process parameters," *Procedia Engineering*, vol. 15, pp. 3978–3983, 2011.

[65] N. Srinivas and K. Deb, "Multiobjective optimization using nondominated sorting in genetic algorithms," *Evolutionary computation*, vol. 2, no. 3, pp. 221–248, 1994.

[66] P. W. Shaikh, M. El-Abd, M. Khanafer, and K. Gao, "A review on swarm intelligence and evolutionary algorithms for solving the traffic signal control problem," *IEEE transactions on intelligent transportation systems*, vol. 23, no. 1, pp. 48–63, 2020.

[67] Y. Wang, J. Shi, R. Wang, Z. Liu, and L. Wang, "Siting and sizing of fast charging stations in highway network with budget constraint," *Applied Energy*, vol. 228, pp. 1255–1271, 2018.

[68] F. Goodarzian and H. Hosseini-Nasab, "Applying a fuzzy multi-objective model for a production–distribution network design problem by using a novel self-adoptive evolutionary algorithm," *International Journal of Systems Science: Operations & Logistics*, vol. 8, no. 1, pp. 1–22, 2021.

[69] H. A. Daham and H. J. Mohammed, "An evolutionary algorithm approach for vehicle routing problems with backhauls," *Materials Today: Proceedings*, 2021.

[70] G. Zhou, Z. Zhu, and S. Luo, "Location optimization of electric vehicle charging stations: Based on cost model and genetic algorithm," *Energy*, vol. 247, p. 123437, 2022.

[71] B. Mohammadi, Y. Guan, R. Moazenzadeh, and M. J. S. Safari, "Implementation of hybrid particle swarm optimization-differential evolution algorithms coupled with multi-layer perceptron for suspended sediment load estimation," *Catena*, vol. 198, p. 105024, 2021.

[72] S. Nesmachnow, J. Muraña, G. Goñi, R. Massobrio, and A. Tchernykh, "Evolutionary approach for bus synchronization," in *Latin American High Performance Computing Conference*. Springer, 2020, pp. 320–336.

[73] M. Jahandideh-Tehrani, O. Bozorg-Haddad, and H. A. Loáiciga, "A review of applications of animal-inspired evolutionary algorithms in reservoir operation modelling," *Water and Environment Journal*, vol. 35, no. 2, pp. 628–646, 2021.

[74] L. Adenaw and M. Lienkamp, "Multi-criteria, co-evolutionary charging behavior: An agent-based simulation of urban electromobility," *World Electric Vehicle Journal*, vol. 12, no. 1, p. 18, 2021.

[75] D. Rizopoulos and D. Esztergár-Kiss, "Heuristic time-dependent personal scheduling problem with electric vehicles," *Transportation*, pp. 1–40, 2022.

[76] F. Ahmad, A. Iqbal, I. Ashraf, M. Marzband *et al.*, "Optimal location of electric vehicle charging station and its impact on distribution network: A review," *Energy Reports*, vol. 8, pp. 2314–2333, 2022.

[77] S. R. Gampa, K. Jasthi, P. Goli, D. Das, and R. Bansal, "Grasshopper optimization algorithm based two stage fuzzy multiobjective approach for optimum sizing and placement of distributed generations, shunt capacitors and electric vehicle charging stations," *Journal of Energy Storage*, vol. 27, p. 101117, 2020.

[78] L. Pan, E. Yao, Y. Yang, and R. Zhang, "A location model for electric vehicle (ev) public charging stations based on drivers' existing activities," *Sustainable Cities and Society*, vol. 59, p. 102192, 2020.

[79] H. Zhang, C. J. Sheppard, T. E. Lipman, T. Zeng, and S. J. Moura, "Charging infrastructure demands of shared-use autonomous electric vehicles in urban areas," *Transportation Research Part D: Transport and Environment*, vol. 78, p. 102210, 2020.

[80] D. Ji, Y. Zhao, X. Dong, M. Zhao, L. Yang, M. Lv, and G. Chen, "A spatial-temporal model for locating electric vehicle charging stations," in *National Conference on Embedded System Technology*. Springer, 2017, pp. 89–102.

[81] A. Raith, A. Aish, G. Covic, P. Jochem, and M. Reuter-Oppermann, "New simulation tool of electric taxi services for analysis of charging infrastructure placement."

[82] T.-Y. Ma and S. Xie, "Optimal fast charging station locations for electric ridesharing with vehicle-charging station assignment," *Transportation Research Part D: Transport and Environment*, vol. 90, p. 102682, 2021.

[83] T. Peng, Z. Liu, Y. Yang, S. Wu, and Y. Gao, "Comparison of five optimization frameworks for multi-objective design optimization of offshore wind turbines," *Energy Conversion and Management*, vol. 218, p. 113381, 2020.

[84] Y. Oyama, T. Kuriyama, Y. Ueda, and A. Tokuhiro, "Comparison of multi-objective optimization algorithms for water distribution network design," *Water Resources Management*, vol. 34, no. 9, pp. 2903–2916, 2020.

[85] M. A. Mariscal, A. Saenz-Otero, and A. Alvarez, "Comparison of multi-objective optimization frameworks for the design of aircraft wings," *Aerospace Science and Technology*, p. 107231, 2021.

[86] N. Rodríguez-Fernández, J. A. Aguado, R. Pino-Mejías, and A. Moreno-Munoz, "Comparison of optimization frameworks for the design of photovoltaic systems," *Energies*, vol. 13, no. 21, p. 5723, 2020.

[87] C. Chen, C. Liu, and C. Kang, "Comparison of two evolutionary optimization algorithms for optimal design of power systems," *Energies*, vol. 13, no. 14, p. 3673, 2020.

[88] W. Liu, B. Hu, Q. Lin, and J. Chen, "Multi-objective optimization of electric vehicle charging station locations in urban areas," *Transportation Research Part C: Emerging Technologies*, vol. 111, pp. 340–358, 2020.

[89] X. Gao, Y. Liu, and H. Chen, "A hybrid genetic algorithm-based approach for optimal electric vehicle charging station placement in large-scale transportation networks," *Journal of Cleaner Production*, vol. 287, p. 125322, 2021.

[90] W. Niu, W. Du, and K. Wang, "Optimal placement of ev charging stations in rural areas with multi-objective optimization," *Journal of Cleaner Production*, vol. 267, p. 122090, 2020.

[91] X. Li, Y. Wei, J. Wang, X. Zhang, and W. Li, "Clustering-based genetic algorithm for electric vehicle charging station location allocation in limited urban space," *Journal of Advanced Transportation*, vol. 2020, p. 8842969, 2020.

[92] J. Zhou, Y. Li, K. Li, and J. Wang, "Multi-objective optimization of fast charging station allocation for electric vehicles with a novel ga-based approach," *Energies*, vol. 13, no. 15, p. 3857, 2020.

[93] X. Zhang, Y. Zhang, H. Wang, and C. Liu, "A robust optimization approach to the optimal deployment of ev charging stations in large-scale transportation networks with uncertain demand," *Transportation Research Part C: Emerging Technologies*, vol. 124, p. 103050, 2021.

[94] Z. Huang, Y. Liu, and L. Ma, "A parking-based ga optimization for electric vehicle charging station placement in urban areas with limited parking spaces," *Transportation Research Part D: Transport and Environment*, vol. 81, p. 102238, 2020.

[95] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[96] A. Qin, V. Huang, and P. Suganthan, "Analysis and optimization of mutation probabilities in differential evolution," *IEEE Transactions on Cybernetics*, vol. 50, no. 6, pp. 2387–2399, 2019.

[97] M. Rostami and A. Bahreininejad, "Mutation operators in evolutionary algorithms: a review," *Neural Computing and Applications*, vol. 32, no. 21, pp. 15 989–16 021, 2020.

[98] S. De, S. Saha, and U. Chakraborty, "Multimodal optimization using adaptive binary differential evolution with mutation probability-based adaptation," *Swarm and Evolutionary Computation*, vol. 65, p. 100902, 2021.

[99] M. B. Alaya, W. Louati, M. Hammami, and A. M. Alimi, "Comparison of some metaheuristic algorithms for the flexible job-shop scheduling problem," *Journal of Intelligent Manufacturing*, vol. 30, no. 5, pp. 2145–2161, 2019.

[100] D. E. Goldberg, L. Wang, X. Yao, and S. Liu, "The effects of parameter tuning on genetic algorithm performance," *Frontiers of Computer Science*, vol. 13, no. 2, pp. 330–341, 2019.

[101] F. Lobo, L. de Castro, and M. Chica, "Parameter tuning in evolutionary algorithms: A survey of the state of the art," *Swarm and Evolutionary Computation*, vol. 57, p. 100710, 2020.

[102] A. Mohr, J. Weidendorfer, and O. Hohlfeld, "Docker for reproducible research: An empirical study of researchers' practices, challenges, and needs," in *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, 2021, pp. 1–14.

[103] D. Merkel, "Docker: lightweight linux containers for consistent development and deployment," *Linux journal*, vol. 2014, no. 239, p. 2, 2014.

[104] C. Boettiger, "An introduction to docker for reproducible research," *ACM SIGOPS Operating Systems Review*, vol. 49, no. 1, pp. 71–79, 2015.

[105] Simunto, "Simunto-vía user guide," Simunto, Zurich, Switzerland, Technical Report SV-2018-UG, 2018.

Appendices

.1 Appendix 1.

.1.1 MATSIM

MATSim (Multi-Agent Transport Simulation) is an open-source, large-scale, multi-modal, agent-based transportation simulation framework [14]. It was developed to model complex transport systems in a realistic way, including the behavior of individual travelers and the interaction with the transport network. One of the main features of MATSim is its modular architecture, which allows users to easily customize the simulation according to their needs. One of these modules is the EV (Electric Vehicle) module, which simulates the behavior of electric vehicles in the transport system.

Ev Module

The EV module in MATSim allows for the simulation of electric vehicles as a part of the transport system. The module enables the inclusion of electric vehicles in the simulation and the modeling of their charging behavior, including charging times and locations. It also allows for the modeling of range anxiety and other constraints that are specific to electric vehicles [14]. The EV module can be combined with other modules in MATSim, such as the network module, to simulate the interaction between electric vehicles and the transport network.

The EV module is based on the concept of agent-based modeling, which represents individual travelers and their decision-making processes. In the EV module, electric vehicle users are represented as agents, who make decisions about when and where to charge their vehicles based on their individual preferences and constraints. These agents interact with the transport network, other agents, and charging infrastructure to reach their destination in an efficient way.

The EV module in MATSim provides a flexible and customizable framework for simulating the behavior of electric vehicles in the transport system. It allows for the modeling of complex charging behavior and range anxiety, and can be integrated with other modules in MATSim to simulate the interaction between electric vehicles and the transport network. Overall, the EV module in MATSim is a useful tool for studying the potential impact of electric vehicles on the transport system and for evaluating policies related to

electric vehicle adoption.

.1.2 matsim_ev environment

The matsim_ev environment contains the jar file, which we have named ev_esenarios, and the subdirectory with the configurations, network, and mobility plans. The simulation of the EV MATSim module generates the JAR file. Through Eclipse IDE, we can import, as a Maven project, the MATSim contribs that we need for our simulation project. This research needs the EV module, which contains the necessary files for simulating electric cars. Python calls the JAR file with the required configurations, meaning the Cuenca scenario with its network and mobility plans. The JAR file contains the libraries with scenarios classified according to the vehicles' capacities and the agents' mobility plans.

.1.3 ev_interface environment

Once we have obtained the JAR file, which we have named ev_scenarios, we can integrate it with our Python project. The python environment, which we have named ev_interface, has several classes that call the call JAR file and pass it the necessary parameters to run the simulation and get the results. The python classes connect the simulator with any multi-objective evolutionary optimization framework. The simulation receives the configuration of an individual or population and reports the evaluation. Some of the classes we need to integrate the simulation are the following.

- **ev_runI.sh:** The ev_runI.sh script contains the call to the ev_interface environment. The script captures the input parameters and sets them to a series of environment variables. The input parameters it receives are ¡scenario name¿ ¡number of agents in the simulation¿ ¡individual¿.

- **ev_cliI.py:** This class, through a classification, assign the simulation parameters to different variables. With the simulation parameters, the simulation is pre-configured (ev_simulatorI.py), and then ev_callMianI.py is called, which receives the simulation object for execution through ev_problemI.py.

- **ev_simulatorI.py:** This class takes care of all the configuration of the simulation. The configuration, network, and mobility plans (config.xml, network.xml, plan.xml,

etc.) are necessary files for running the simulation. This class has two main functions: sim and run. The sim class deals with the configurations, and the run class deals with executing the simulation. It takes some values through the ev_settingsI.py class that contains some constants of simulator configuration values, for example, the locations of the charging stations associated with the network links of each scenario. Use ev_parserI.py to get the values from the simulation output files so you can set the fitness variables.

- **ev_callMainI.py:** This class is in charge of instantiating the ev_problemI.py class, which contains the fitness function that allows evaluating the individual.

- **ev_problem.py:** This class allows the configuration of the fitness function, necessary for the evolutionary algorithm to obtain the evaluations of the solutions. This class calls the ev_initializeI.py class which allows, in the particular case of electric mobility, to instantiate the solution through the ev_solutionI.py class which defines the names of the fitness functions among other attributes. The ev_initializeI.py class creates some files to complete the simulation setup (for example, creating the charging stations file in xml format) and calls the ev_simulatorI.run function to get the fitness values of the solutions. The ev_problemI.py class uses the ray library to parallelize the execution of the simulations (see fitnessPop method).

.1.4 DEAP

The Distributed Evolutionary Algorithms in Python (DEAP) library is an open-source software designed for implementing various types of evolutionary computation algorithms in Python. DEAP provides several tools for implementing evolutionary algorithms, such as genetic algorithms, genetic programming, and swarm intelligence algorithms. One of the main features of DEAP is its flexibility, which allows users to easily customize the algorithms for their specific needs.

DEAP toolbox

The DEAP toolbox is the core of the library and contains all the necessary components for building evolutionary algorithms. The toolbox provides a wide range of functions,

classes, and algorithms for various tasks, including individual and population creation, fitness evaluation, selection, and genetic operators. These tools can be used to create a custom evolutionary algorithm by combining them in different ways.

DEAP individuals definition

Individuals in DEAP are represented as Python objects that contain a genome, which is a collection of parameters that define a solution. The toolbox provides several pre-defined data types, such as floats, integers, and lists, to represent the genome. Users can also define their own data types to represent more complex solutions.

DEAP fitness function definition

The fitness evaluation function in DEAP is a user-defined function that calculates the fitness of an individual. The toolbox provides several types of fitness functions, such as maximizing or minimizing a single objective, maximizing or minimizing multiple objectives, and constraint satisfaction. Users can also define their own fitness functions to suit their specific needs.

DEAP Selection methods

Selection in DEAP is performed using a variety of methods, such as tournament selection, roulette wheel selection, and rank-based selection. The toolbox also provides several genetic operators, such as mutation, crossover, and reproduction, which can be used to create new individuals from the current population.

DEAP also includes several advanced features, such as parallelization, island models, and checkpointing, which can be used to speed up the optimization process and improve the quality of the solutions.

DEAP (Distributed Evolutionary Algorithms in Python) is an evolutionary computing framework written in python which is used to create projects involving evolutionary algorithms quickly. This framework enables researchers to develop custom projects since the authors provide the necessary libraries and the source code. The developers explicitly provide the algorithms and structures to be adaptable, unlike other software encapsulating the algorithms with the black box approach. DEAP is a framework that enables paral-

lelization mechanisms such as multiprocessing and SCOOP modules. The source code is available at `https://github.com/DEAP/notebooks`.

According to Andres, the DEAP framework has the following principles

- The data structures are key to using this framework because they make writing algorithms easy and clean.
- Depending on the type of problem, the selection operators and the parameters can be chosen. This choice significantly influences the evolution, so the authors recommend defining the problem as close to reality as possible to obtain the most accurate results.
- DEAP provides the necessary mechanisms to solve evolutionary algorithms frequently requiring parallel execution.

.1.5 DEAP operators

The DEAP framework provides a variety of genetic operators to create new offspring from parent individuals in evolutionary algorithms.

1. `stepMut`: This function performs the mutation operation on the offspring population. It applies a specific mutation method to each offspring with a given probability. The mutation operator can be customized to meet the needs of a specific problem. The `stepMut` function is called by the `eaSimple` or `eaMuPlusLambda` algorithms in the DEAP framework.
2. `xOnePoint`: This function performs the crossover operation on the parents to generate offspring. It applies the one-point crossover method to each parent with a given probability. This function can be replaced with other crossover methods, depending on the specific requirements of a problem. The `xOnePoint` function is also called by the `eaSimple` or `eaMuPlusLambda` algorithms in DEAP.
3. `selNSGA2`: This function performs the selection operation on the population using the NSGA-II algorithm. The NSGA-II algorithm is a widely used multi-objective optimization algorithm that ranks individuals based on their non-dominance level and their crowding distance. The `selNSGA2` function is called by the `eaMuPlusLambda` algorithm in DEAP.

.2 Appendix 2

.2.1 Coordinates of charging stations

Table 1: Coordinates of Charging Stations

station	x	y
st-1	721631.4855313053	9680391.180570882
st-2	723000.4311702102	9679744.778843224
st-3	721688.6788659381	9680050.486500792
st-4	720836.4889870753	9677307.93879515
st-5	723914.2040172062	9680495.377165154
st-6	728731.5746737241	9681646.518565264
st-7	725001.7400978936	9680233.125461085
st-8	724050.1781458601	9682120.102357747
st-9	728869.93	9683144.0
st-10	724397.0563394848	9678445.49127423
st-11	720165.8724970899	9679212.32718246
st-12	718982.5543238329	9681754.794208251
st-13	719185.9448167619	9678456.61874815
st-14	722634.4878721512	9678645.908995178
st-15	717454.8125	9677002.0
st-16	719083.25	9679597.0
st-17	720248.5625	9678016.0
st-18	722969.25	9681165.0
st-19	715181.625	9681477.0
st-20	725614.375	9682138.0