DOI: 10.14750/ME.2023.026

UNIVERSITY OF MISKOLC
FACULTY OF MECHANICAL ENGINEERING AND INFORMATICS

The generalization of the Vehicle Routing Problem: Mathematical model, ontology, algorithms, fitness landscape analysis and applications

Ph.D. dissertation

Anita Agárdi
MSc in Information Engineering

József Hatvany Doctoral School of Information Science, Engineering and Technology
Research Area
Applied Computational Science
Research Group
Data and Knowledge Bases, Knowledge Intensive Systems

Supervisor
Prof. Dr. László Kovács
professor

Co-supervisor
Prof. Dr. Tamás Bányai
professor

Miskolc, 2023

Declaration

The author hereby declares that this thesis has not been submitted, either in the same or in different form, to this or to any other university for obtaining Ph.D. degree. The author confirms that the submitted work is her own and the appropriate credit has been given where reference has been made to the work of others.

Nyilatkozat

Alulírott Agárdi Anita kijelentem, hogy ezt a doktori értekezést magam készítettem, és abban csak a megadott forrásokat használtam fel. Minden olyan részt, amelyet szó szerinti vagy azonos tartalomban, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Miskolc, 2023. január 16.

Agárdi Anita

A disszertáció bírálatai és a védésről készült jegyzőkönyv megtekinthető a Miskolci Egyetem Gépészmérnöki és Informatikai Karának Dékáni Hivatalában, valamint a doktori iskola weboldalán az Értekezések menüpont alatt: http://www.hjphd.iit.uni-miskolc.hu

Supervisor recommendation

I have known Ms. Anita Agárdi for nearly six years from the time that she joined our program in Computer Engineering. She has consistently achieved excellence in her studies, achieving the best grades. She worked hard also during her MSc studies, she participated several times in the local and national level TDK competitions where I was the supervisor or co-supervisor of her TDK essays. She achieved special awards at the national level TDK.

In her PhD study, she was an excellent and very talented student with a lot of ambition. She passed the required courses in a short time and she could turn to the scientific research work at a very early stage. She had a very strong interest in the integration of computer science with the field of logistic. She worked hard to analyse the related background literature, to synthesize the different approaches and to develop novel application frameworks. She proved her excellent talent in scientific publications in her selected fields.

As the MTMT publication and citation summary show, Anita Agárdi has 55 scientific publications: 39 scientific journal articles, 9 conference papers and 7 other scientific works. The Hirsch-index of her publications is 4 and they have 39 independent citations. The total impact factor of her publications is 18.190, while her weighted impact factor is 6.291.

During her PhD study, she won the ÚNKP awards several times. The excellent publication activity and the awards show her ability to perform high level research work on the field of applied computer science, especially focusing on the optimization of the Vehicle Routing Problem.

As the Head of Department of Computer Technology where Ms. Anita Agárdi is working as an assistant lecturer, I would like to highlight the activity of Anita in participation of different department level tasks, like organization of students' events or administration of our local scientific journal.

I am completely convinced that Ms. Anita Agárdi, based on her accomplishment and talent, is capable of carrying out independent scientific research work, I fully support her application for the PhD degree.

Miskolc, January 16, 2023.

....

Prof. Dr. László Kovács

PhD supervisor

Co-supervisor recommendation

I have known Anita Agárdi for a few years now: I initially came to know her during her postgraduate coursework studies in Information Engineering at the University of Miskolc, in which I was the co-supervisor of her TDK essay. This TDK essay was presented at the both Local and National conference of Student Research Societies. I was her PhD co-supervisor and this enabled me to gain, professionally and personally, a very thorough impression of Anita Agárdi.

It is so easy to provide a recommendation for a person of Ms. Agárdi's calibre. One only needs to glimpse at her professional qualifications to be convicted that she is one of those rare individuals who are never satisfied with what they have achieved and who are continuously striving for excellence. She can be described as a perfectionist, in the best possible sense of this notion. The works that she submitted were always very thoroughly researched and immaculately presented, demonstrating a remarkable capacity to think analytically and to provide a balanced, penetrating and persuasive argument. The same is even more obvious in her PhD thesis, which is undeniable the most highly literate and well-written dissertation that I have supervised in the whole of my academic career. One of the finest achievement of her thesis is the presentation of novel model, ontology and algorithm of generalized Vehicle Routing Problems.

Her excellent scientific results can be validated by her excellent publications. As the MTMT publication and citation summary show, Anita Agárdi has 55 scientific publications: 39 scientific journal articles, 9 conference papers and 7 other scientific works. The Hirsch-index of her publications is 4 and they have 39 independent citations. The total impact factor of her publications is 18.190, while her weighted impact factor is 6.291.

She has been actively involved in the international relations of the Institute of Logistics, participating several times in meetings of international projects, for example in Germany and France.

In light of the above I am completely convinced that a person of Ms. Anita Agárdi's accomplishment and talent is capable of carrying out independent scientific research work.

Miskolc, January 16, 2023.

....
Prof. Dr. Tamás Bányai
PhD co-supervisor

Acknowledgement

I would like to thank all those who contributed in connection with the completion of my dissertation with their support. First, I would like to thank my supervisor, **Prof. Dr. László Kovács**, who has provided me with professional help and support for nearly six years. I would like to thank **Prof. Dr. Tamás Bányai**, who helped my work as a co-supervisor since my MSc studies. The dissertation could not have been completed without their professional guidance and help. I would like to also thank the József Hatvany Doctoral School for supporting my work.

I would like to also thank my Family, who trusted and stood by me during the compilation of my research work and this dissertation.

Table of contents

List of abbreviations

- VRP: Vehicle Routing Problem

- ACS: Ant Colony System, where the initial solution is generated randomly.
- ACS_C: Ant Colony System, where the initial solution is generated with construction algorithms and randomly.
- AS: Ant System, where the initial solution is generated randomly.
- AS_C: Ant System, where the initial solution is generated with construction algorithms and randomly.
- ESAS: Elitist Strategy of Ant System, where the initial solution is generated randomly.
- ESAS_C: Elitist Strategy of Ant System, where the initial solution is generated with construction algorithms and randomly.
- FA: Firefly Algorithm, where the initial solution is generated randomly.
- FA_C: Firefly Algorithm, where the initial solution is generated with construction algorithms and randomly.
- FCHC: First Choice Hill Climbing, where the initial solution is generated randomly.
- FCHC_AI: First Choice Hill Climbing, where the initial solution is generated with Arbitrary Insertion algorithm.
- FCHC_CI: First Choice Hill Climbing, where the initial solution is generated with Cheapest Insertion algorithm.
- FCHC_FI: First Choice Hill Climbing, where the initial solution is generated with Farthest Insertion algorithm.
- FCHC_NI: First Choice Hill Climbing, where the initial solution is generated with Nearest Insertion algorithm.
- FCHC_NN: First Choice Hill Climbing, where the initial solution is generated with Nearest Neighbour algorithm.
- FCHC_G: First Choice Hill Climbing, where the initial solution is generated with Greedy algorithm.
- GA: Genetic Algorithm, where the initial solution is generated randomly.
- GA_C: Genetic Algorithm, where the initial solution is generated with construction algorithms and randomly.
- HS: Harmony Search, where the initial solution is generated randomly.
- HS_C: Harmony Search, where the initial solution is generated with construction algorithms and randomly.
- MMAS: MAX-MIN Ant System, where the initial solution is generated randomly.
- MMAS_C: MAX-MIN Ant System, where the initial solution is generated with construction algorithms and randomly.
- PSO: Particle Swarm Optimization, where the initial solution is generated randomly.
- PSO_C: Particle Swarm Optimization, where the initial solution is generated with construction algorithms and randomly.
- RBVAS: Rank Based Version of Ant System, where the initial solution is generated randomly.
- RBVAS_C: Rank Based Version of Ant System, where the initial solution is generated with construction algorithms and randomly.
- SA: Simulated Annealing, where the initial solution is generated randomly.
- SA_AI: Simulated Annealing, where the initial solution is generated with Arbitrary Insertion algorithm.
- SA_CI: Simulated Annealing, where the initial solution is generated with Cheapest Insertion algorithm.

- SA_FI: Simulated Annealing, where the initial solution is generated with Farthest Insertion algorithm.
- SA_NI: Simulated Annealing, where the initial solution is generated with Nearest Insertion algorithm.
- SA_NN: Simulated Annealing, where the initial solution is generated with Nearest Neighbour algorithm.
- SA_G: Simulated Annealing, where the initial solution is generated with Greedy algorithm.
- TS: Tabu Search, where the initial solution is generated randomly.
- TS_AI: Tabu Search, where the initial solution is generated with Arbitrary Insertion algorithm.
- TS_CI: Tabu Search, where the initial solution generated with Cheapest Insertion algorithm.
- TS_FI: Tabu Search, where the initial solution generated with Farthest Insertion algorithm.
- TS_NI: Tabu Search, where the initial solution generated with Nearest Insertion algorithm.
- TS_NN: Tabu Search, where the initial solution generated with Nearest Neighbour algorithm.
- TS_G: Tabu Search, where the initial solution generated with Greedy algorithm.
- AI: Arbitrary Insertion.
- CI: Cheapest Insertion.
- FI: Farthest Insertion.
- NI: Nearest Insertion.
- NN: Nearest Neighbour.
- G: Greedy.

- BOFM: Bounded Objective Function Method.
- PR: Pareto Ranking.
- WESM: Weighted Exponential Sum Method.
- WGCM: Weighted Global Criterion Method.
- WPM: Weighted Product Method.
- WSM: Weighted Sum Method.

- OX: Order Crossover.
- CX: Cycle Crossover.
- PMX: Partially Matched Crossover.

- BSS: Basic Swap Sequence.

- OWL: Web Ontology Language.
- XML: eXtensible Markup Language.
- RDF: Resource Description Framework.

List of figures

List of tables

1. Introduction

One of the most important tasks of logistics is the cost-effective delivery of the right products to the right place at the right time. The Vehicle Routing Problem (VRP) is a transportation problem, one of the most common tasks in logistics. The objective of the basic Vehicle Routing Problem (VRP) is to serve the demands of customers in a given position. Vehicles deliver products from one or more depots to customers and then return to the depot. Over the years, several variants (components) of the task have developed, thus adapting to the needs of the industry. Examples of such components are Time Window, Multi-Echelon (when products are first transported from the depot to distribution points - called satellites - and then from there to the customers), Open Route, Multiple Product, Multiple Vehicle, Vehicle Routing with Electric Vehicles, and so on. The Vehicle Routing Problem can be in-plant, out-of-plant transport depending on the components used, but it can even model an entire supply chain. As transport tasks are becoming more and more complex nowadays (simultaneous delivery of several products, several types of vehicles, several types of distribution points, etc.), computer support is necessary for the design and operation tasks.

The aim of the dissertation is to develop a general model and prototype transportation system. Based on the literature, current proposals of this area are focusing only on a subsystem. There is no problem covering a wide area in the literature. The main goal of my research work is an integrated general system. The proposed general system should contain all the major components currently known in practice. Having this general model, almost all types of transportation tasks that appear in logistics systems can be analyzed and solved as a subtype of the general model.

The first part of the dissertation is a literature review, which presents the components of the Vehicle Routing Problem (VRP) and the algorithms with which the Vehicle Routing Problem (VRP) can be solved.

In the next chapter the mathematical model of the Vehicle Routing Problem is presented. The model includes among others the following components: number of levels, number of vehicles, number of product types, periodicity, attributes between nodes (such as distance between nodes, safety factor, etc.), vehicle attributes (such as vehicle capacity limit, charging time, the rental fee for rented vehicles, etc.), time components (packing, unpacking time, time window, etc.), product attributes (such as the capacity limit of the node, fixed order of arrival of products, etc.), costs (such as packing and unpacking costs, quality control cost, etc.) and finally operational parameters (such as inter-depot route, delivery, collection, open route, etc.). Regarding the parameter values the model allows to use static, stochastic, fuzzy, and forecasted data types.

After the mathematical model, I present an ontology knowledge base that helps to design the Vehicle Routing Problem (which types of products can be transported with which vehicles, which types of products can be transported together, what types of products may be needed by each customer, what components should be used in the system design).

Next, I outline the representation related to the applied optimization algorithms and evaluation. The representation used is based on permutations of customers, but I also had to consider other factors in the design, such as different types of products, vehicles, and the levels (Multi-Echelon).

Detailed analyzes of optimization algorithms are also presented in the dissertation in the context of fitness landscape analysis. Algorithms can be divided into two parts: construction and improvement algorithms. Construction algorithms create one possible solution. They take the best steps locally, but global optimum is most often not achieved with their exclusive use. Their running time is usually low. In my dissertation I used 6 different construction algorithms. Improvement algorithms iteratively improve one or more possible solutions. Their running time is usually high. The global optimum can be achieved with their application. In the dissertation, I used 11 different improvement algorithms. Since the model contains several objective functions, I tested 6 different multi-objective optimization techniques. I also implemented the “pre-filtering” of customers, so not all incoming needs must be fulfilled, only those that are economically profitable. This can be done using a classification known in data mining. I have used 5 different classification data mining algorithms.

I have applied the fitness landscape analysis method to extended analysis of the search space. For the generation of the optimum points I used either the optimum of the optimization algorithms or the random selection method. Using the fitness landscape I performed the investigation of the operators and the efficiency of the optimization algorithms. For the analysis I have used 6 different walk techniques. I performed other analyzes for random walk also from the view point of information theory.
In the last chapter I show, that the general system is suitable for solving both in-plant and out-of-plant transport processes. I presented 16 prototype systems to illustrate the versatility of the proposed system. Based on the theoretical and practical test I stated that the proposed model covers wide range of transportation models.

1.1. Research goals

The focus of my research is the generalization of the Vehicle Routing Problem. My goal is to perform a literature review and create a general Vehicle Routing Problem model based on this. Based on the general model, specific transportation tasks can be solved. The advantage of the general model is that it only needs to be created once. It is not necessary to write a new model or a new program for every transportation tasks, it is enough to parameterize the general model. My research covers the followings:

- Performing a literature review on the Vehicle Routing Problem, systematization of the different tasks (components, constraints, objective functions).
- Creation of a general Vehicle Routing Problem model, mathematical description of the model based on the literature and logistics tasks.
- Creation of the ontology of the generalized Vehicle Routing Problem model. The ontology must contain the elements of the mathematical model and the additional elements of the main components (nodes, vehicles, customers). The system should help the user in which of the components and metrics of my generalized VRP model to use during the delivery of a specific product to a specific node with a specific vehicle. I also saw it necessary for the system to help to decide which products can be delivered to the nodes with which vehicles.
- Creation of the representation model of the general VRP model. With the help of the representation, the real problem can be mapped for the optimization algorithm.
- Analysis of the search space of the VRP. The analysis of the search space can be linked to the study of optimization algorithms. The aim of my research is to analyze the effectiveness of optimization algorithms known in the literature. Most of the improvement heuristics use neighborhood operators, so my research goal is also to analyze the efficiency of the operators. The Vehicle Routing Problems may contain several objective function components, so it is also necessary to analyse the multi-objective optimization techniques. My research goal is the application of a wide range of search space analysis techniques in the VRP.
- Developing software that solves the general VRP. Based on the representation, the software can solve the Vehicle Routing Problems with different heuristics, answer the user's questions in connection with the VRP based on the implemented ontology, and analyze the search space.
- The success of a general model lies in how wide a range of problems it covers. I analyze which transportation tasks can be solved with the created general model.

2. Survey of the literature background of the Vehicle Routing Problem (VRP)

The aim of this chapter is to present the informal definition, components and the related literature background of the Vehicle Routing Problem (VRP).
In case of the VRP, all (potentially) places to be visited are called as nodes. The node can be a depot, customer, charger station, satellite. Any transport unit that transports products between nodes is called a vehicle.
During the basic Vehicle Routing Problem [1], the customers can be visited by vehicles. Customers have a fix demand for products. All demands must be served; each node can only be visited once. The vehicles start from depot, visit the customers assigned to them and then return to the depot. Vehicles cannot exceed their capacity limit. If the vehicles visited the customers, then they must return to the depot. The objective function is the minimization of the distance travelled by vehicles. The problem is illustrated in *Figure 1*, where the customers are denoted with integer numbers, the product demands are denoted with PDN.

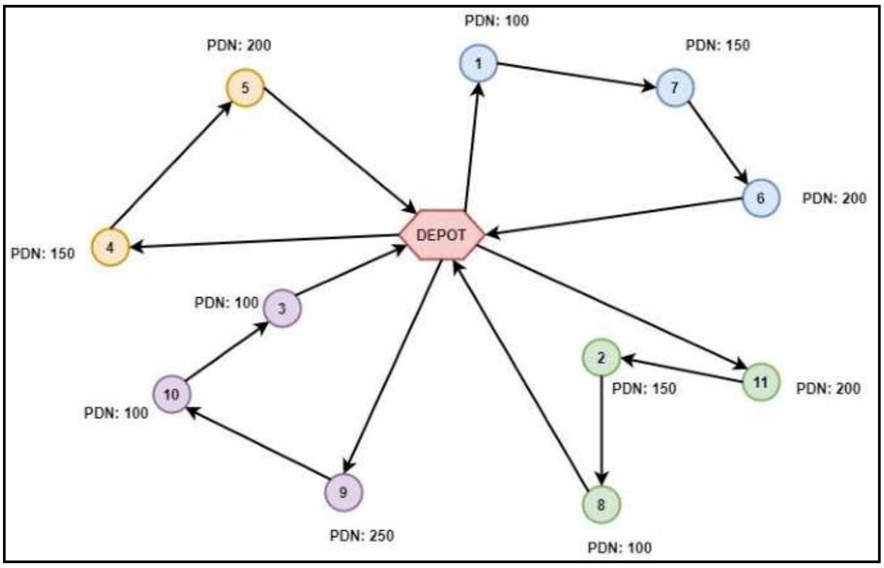

Figure 1.: The Vehicle Routing Problem (VRP)

2.1. Literature survey

The Vehicle Routing Problem is a current and important area of logistics. The first Vehicle Routing Problem articles appeared in 1959 by Dantzig & Ramser [1]. After that, in the 60s and 70s, many articles appeared for example [2-3] publications. Over the years, several components of the task have evolved that are trying to meet the growing needs of the industry.
I performed a literature analysis to discover the key trends and achievements in the research area of the Vehicle Routing Problems. I have used the Google Scholar database to retrieve the papers on VRP.

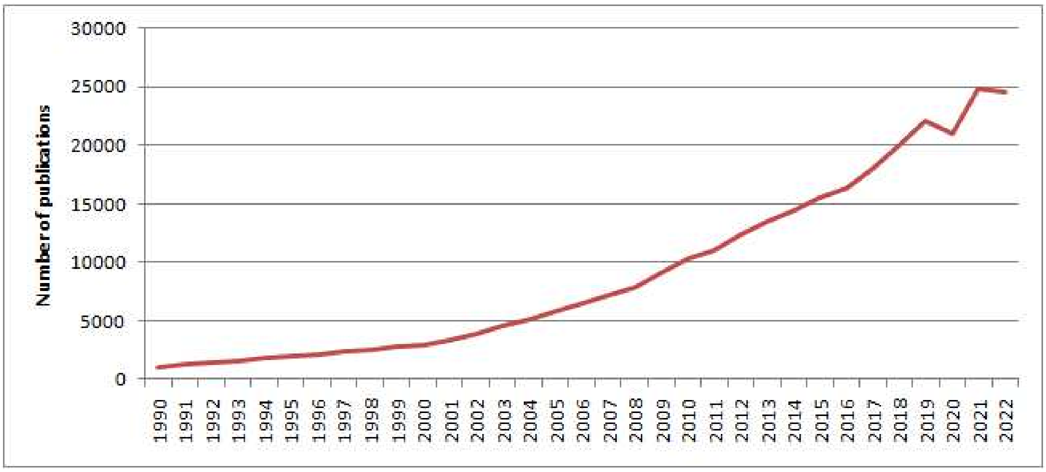

Figure 2.: Annual appearance of publications

As *Figure 2.* shows the number of articles published in Vehicle Routing Problem is increasing. Since 2010, the number of publications has shown a drastic increase. The search was considered from the 90s.

2.2. Categories of the research topic

The transportation problem may have many parameters. I have selected the followings to perform the categorization of the research topic: the number of nodes, their type (depot, customer, charger station, satellite and their levels). In the case of transportation, it is also necessary to decide on the number of vehicles and products. The types of vehicles to be used may involve the use of several components, such as the use of charger stations, the rental fee (if the vehicles are rented). Types of products can also determine components, the relationship between products and suitable vehicles for transportation, but also that the objective should not be to minimize the distance travelled but to make the vehicle cross the safest route possible. Business logic can also define components such as the time window (only individual nodes can be visited in a certain time interval) or the handling of uncertain (stochastic, fuzzy) or past data (forecasting). In the following, I will present the components of the Vehicle Routing Problems.

- Time Window (Vehicle Routing Problem with Time Window) [4]: each customer is given a time window within the demand must be served.
- Multiple Time Windows (Vehicle Routing Problem with Multiple Time Windows) [5]: multiple time windows have been added to customers, one time window must be met.
- Soft Time Window (Vehicle Routing Problem with Soft Time Window) [6]: customers can also be visited outside the time window, but then there is a penalty point.
- Single Depot (Vehicle Routing Problem with Single Depot) [7]: vehicles leave a single depot and then return to the depot when customers are visited.
- Multiple Depots (Vehicle Routing Problem with Multiple Depots) [7]: the vehicles leave one of several depots and then return here.
- Open Route (Open Vehicle Routing Problem) [8]: the vehicles leave the depot but do not return there after visiting the customers.
- Inter-Depot Routes (Multi-Depot Vehicle Routing Problem with Inter-Depot Routes) [9]: vehicles leave one depot at a time and then return to any other depot if they have visited customers.
- Two-Echelon (Two-Echelon Vehicle Routing Problem) [10]: the products are first transported from the depot to intermediate locations (satellites) and then transported to customers from there. In this problem, different types of vehicles can be used for transportation between the depot-satellite and the satellite-customer.

- Homogeneous Fleet (Homogeneous Fleet Vehicle Routing Problem) [11]: the products are transported by vehicles of the same type.
- Heterogeneous Fleet (Heterogeneous Fleet Vehicle Routing Problem) [11]: different types of vehicles transport the products.
- Capacity Constraint (Capacitated Vehicle Routing Problem) [12]: the vehicles have a capacity limit for the products to be transported.
- Pickup and Delivery (Vehicle Routing Problem with Pickup and Delivery) [13]: a combination of delivery and pickup. The products are delivered from the depot to the customers, while the products are picked up from the customers to the depot.
- Multiple Products (Vehicle Routing Problem with Multiple Products [14]: there are several types of products in the system.
- Environmentally Friendly Vehicles (Environmentally Friendly Vehicle Routing Problem) [15]: environmentally-friendly vehicles transport the products.
- Electric Vehicles (Electric Vehicle Routing Problem) [16]: vehicles are electric vehicles, require charging after a certain distance, so they must also visit the charger station.
- Periodic Problem (Periodic Vehicle Routing Problem) [17]: customers do not have to be visited once, but periodically, several times in a period.
- Profits and Consistency Constraints (Vehicle Routing Problem with Profits and Consistency Constraints) [18]: the system contains two types of customers. One type of customer needs to be visited regularly and is a must-visit. The other type can be visited once and is not mandatory, but these customers make a profit. The objective is to achieve maximum profit.
- Feeder Vehicle Routing Problem [19]: given a large and a small capacity vehicle. They visit customers together so that the low-capacity vehicle occasionally encounters the high-capacity to get fuel.
- Stochastic Demand (Vehicle Routing Problem with Stochastic Demand) [20]: the probability distribution of commodity demands is known only.
- Shared Customer Collaboration (Shared Customer Collaboration Vehicle Routing Problem) [21]: there are several types of vehicles that deliver different products. Some customers need one type of products (then only one vehicle visits) some customers need more types of products (then more vehicles visit).
- Cumulative Capacitated Vehicle Routing Problem [22]: the objective function is not to minimize the distance travelled, but to minimize the waiting time.
- Inventory Routing Problem [23]: the quantity of products to be delivered to customers depends on the customer's stock level and consumption level. All that is required is that a certain quantity is always in stock. Delivery is periodic, the quantity of products to be delivered in each period is determined based on the mentioned factors. The objective function is the minimization of the travelled distance.
- Allocation Problem (Vehicle Routing with Allocation Problem) [24]: not all customers need to be visited; missed customers will receive a penalty point.
- Cross-Docking (Vehicle Routing Problem with Cross-Docking) [25]: cross-docking is a new logistics strategy in which it goes from sellers to one place (cross-dock) and then shipped from there to the consumer as quickly as possible. The products are not stored for a long time, in practice approx. within 24 hours are shipped. In this problem, first, the products are collected and then delivered.
- Location Routing Problem [25]: determination of depots from a set (containing possible depots).
- Site-Dependent Vehicle Routing Problem [25]: each customer can only be visited by a certain type of vehicle.
- Selective Vehicle Routing Problem [25]: the products are delivered to certain places that are most favourable in terms of profitability.

- Open Vehicle Routing Problem with Decoupling Points [26]: it is worth applying during transport to unsafe places. A vehicle delivers the products to certain locations (decoupling point) and then transports them from there to other vehicles.
- Occasional Drivers (Vehicle Routing Problem with Occasional Drivers) [27]: given a certain number of vehicles used by the company, and the company may use other vehicles to serve the product demands of customers. These extra vehicles have extra costs.
- Traffic Congestion (Vehicle Routing Problems with Traffic Congestion) [28]: for the task, the time between nodes is not fixed, it varies at each time depending on the traffic.
- Synchronized Visits (Vehicle Routing Problem with Synchronized Visits) [29]: customers must be visited at a certain time interval. Several types of vehicles are used in this problem. Some customers need to be visited by more than one type of vehicles at the same time (the vehicles work together, for example, some maintenance activity at the customer).
- Fuel-Efficient Green Vehicle Routing Problem [30]: the objective function is the minimization of the fuel emission.
- Rollon-rolloff Vehicle Routing Problem [31]: vehicles have large containers. Customers can use containers for 3 reasons: they order a full container (need product demand), they order an empty container (the customers would like to deliver products) and they change containers.
- Stochastic (Stochastic Vehicle Routing Problem) [32]: some factor (e.g. demand for products, presence of customers) is given with probability distribution only.
- Swap-Body (Swap-Body Vehicle Routing Problem) [33]: two types of vehicles are distinguished, the truck and the train. The train consists of a track and a swap body. Some customers can be visited only by truck, some only by train, while others can be visited by truck and train. There are also swap locations where the train can put down the truck body to serve customers that only a truck can visit. Then you have to take the swap body back from there because at the very end of your journey you have to get back to the depot with a swap body.
- Incompatible Goods (Vehicle Routing Problem with Incompatible Goods) [34]: certain products cannot be transported together.
- Generalized Vehicle Routing Problem [35]: customers are divided into clusters, and only one customer in a cluster needs to be visited.
- Perishable Food Products Delivery (Vehicle Routing Problem with Perishable Food Products Delivery) [36]: the delivered products have an expiration date, which must be taken into account.
- Risk Constrained Vehicle Routing Problem [37]: the safety/status of each road section is indicated by a quantity. The objective is the safest road section.
- Delivery and Installation Vehicles (Vehicle Routing Problem with Delivery and Installation Vehicles) [38]: products must either be delivered to each node or some operation must be performed (possibly both). Some vehicles carry products, while others carry out operations at the nodes.
- Clustered Vehicle Routing Problem [39]: each node is clustered. All nodes of one cluster need to be visited after the nodes of another cluster started to be visited.
- Trailers and Transshipments (Vehicle Routing Problem with Trailers and Transshipments) [40]: certain types of vehicles have a higher capacity and are intended to serve as a "depot" for other vehicles - from where other vehicles take the products - while other vehicles deliver the products to the right place.
- Balanced Vehicle Routing Problem [41]: between a minimum and a maximum number of customers must be visited during a trip.
- Cumulative Vehicle Routing Problem [42]: the objective is to minimize customer waiting time.

- Risk-constrained Cash-in-Transit (Risk-constrained Cash-in-Transit Vehicle Routing Problem) [43]: vehicles collect money from each node. Each number on each route describes how safe the route is.
- Fuzzy Vehicle Routing Problem [44]: certain factors, e.g. time window, customers' demands are given in fuzzy numbers.
- Dynamic Vehicle Routing Problem [45]: customers are not given in advance, they will show up along the way.
- Travelling Salesman Problem [46]: the simplest version of the Vehicle Routing Problem, where a single agent (vehicle) visits all the cities (customers) once and takes a tour.

2.3. Optimization methods

The Vehicle Routing Problem is an NP hard discrete optimization problem. In the following, the algorithms are presented, which are generally used to solve the problem (*Figure 3.*).

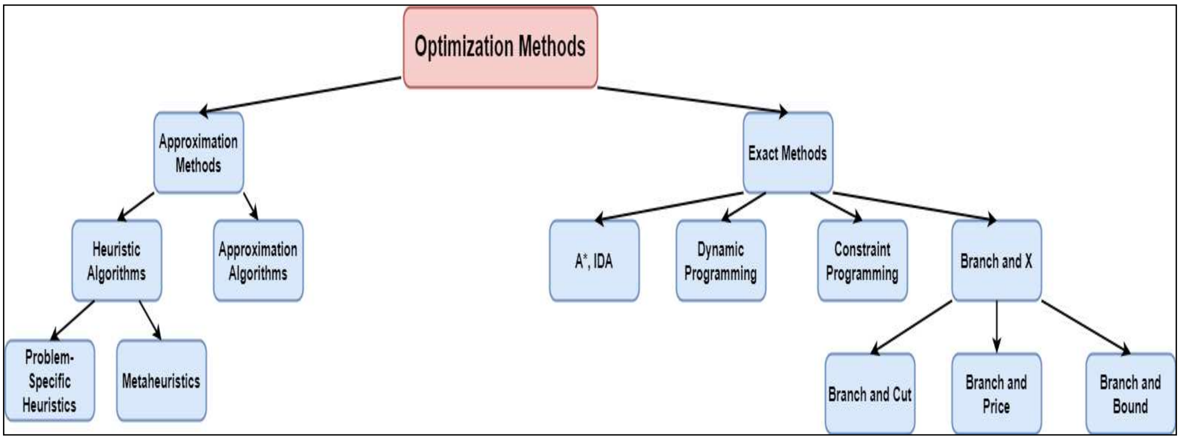

Figure 3.:Optimization Methods [47]

In the dissertation, I have chosen Approximation Methods, which give a relatively "good" solution to complicated problems. VRP is most often solved with the following metaheuristics: Genetic Algorithm, Simulated Annealing, Tabu Search, Ant Colony Optimization, Firefly Algorithm, Harmony Search, Memetic Algorithm, Variable Neighbourhood Search, Evolutionary Algorithm, Artifical Bee Colony, Greedy Randomized Adaptive Search Procedure.

Based my literature survey, the diversity of the problem can be seen. We can find, that there is not a general model for the problem. A general model would like many benefits: management of a complex system, unified framework, flexibility. In reality, the complexity of each system varies, so all smaller models can be covered by a single large model.
Therefore, I found it necessary to develop a system that includes all the major VRP components. The dissertation aims to present this integrated Vehicle Routing Problem model. First, the mathematical model is presented, then the ontological model, then the representation of the heuristic algorithms. Then an analysis of the search space of the task is performed, and finally, some case studies to prove that the developed general model is suitable for the main transport tasks.

3. The mathematical model of the generalization of the Vehicle Routing Problem

It can be seen in Chapter 2., that the Vehicle Routing Problem attracted many researchers over the years. In most of the publications, the mathematical model of the Vehicle Routing Problem was also presented in the articles. These mathematical models were focusing to the investigated transportation problems. The purpose of this chapter is to present generalized mathematical model of a generalized Vehicle Routing Problem.

In the description of the mathematical model the following notations are used:

- i: level index,
- j: position index (positions within the level),
- t: time in a period,
- k: vehicle index (vehicles within the level),
- m: product index,
- q: time window index,
- M: positive large constant,
- a: start time of node service,
- b: arrival time at the node,
- d: time window penalty point.

3.1. Base parameters

<u>Position graph</u>

The topology of the Vehicle Routing Problem can be described by a graph, where the nodes denote positions (customers, depots, etc.) and edges denote transport relationships between positions.

The graph contains the position components and the relationships between them. I assumed, that the positions can be decomposed into several levels. The levels separate the transportation tasks. Several levels are assumed in the graph:

$$LEVEL = \{level_1, \dots, level_{n_{level}}\}. \tag{1}$$

The first level contains only depots, the intermediate levels are local service centres (satellites) and the last level is customers. The number of levels is denoted by n_{level}. In this case, the graph G of the model can be written as follows

$$G = \{G_i\}, \qquad i = 1 \dots n_{level}, \tag{2}$$

where

$$G_i = \{POS_i\}. \tag{3}$$

POS_i denotes the positions in level i. Each position is denoted by the symbol pos_j^i, where i denotes the level and j denotes the sequence number index within the level:

$$POS_i = \{pos_1^i, \dots, pos_{nol_i}^i\}. \tag{4}$$

Assume that the positions correspond to the points of a two-dimensional Euclidean space (given by a coordinate pair), i.e.

$$pos_j^i = [x, y], \tag{5}$$

where $x, y \in \mathbb{R}$.

The set of all positions for the entire model graph:

$$POS = \bigcup_i POS_i \tag{6}$$

and the number of positions:

$$n_{position} = |POS|. \tag{7}$$

Vehicles

Different types of vehicles can be used in the system. The number of vehicles is denoted by $n_{vehicletype}$.

The possible vehicle types are described globally with

$$VEHICLE = \left\{vehicle_1, \dots, vehicle_{n_{vehicletype}}\right\}. \tag{8}$$

The number of vehicles varies by level and type, their values are described with the following equation, where i indicates the index of the level, and k indicates the index of the vehicle within a level:

$$CNTVEHICLE = \left\{cntvehicle_k^i\right\}. \tag{9}$$

All vehicles in the system are marked with

$$n_{vehicle} = \sum_i \sum_k cntvehicle_k^i . \tag{10}$$

Products, services

In the model, several different products are delivered. The number of product types is specified with $n_{producttype}$.

The possible types of products are indicated globally with

$$PRODUCTTYPE = \left\{product_1, \dots, product_{n_{producttype}}\right\}. \tag{11}$$

Time

In the model, the operation is modelled in time. Time is measured at different units A period is the repetition time. Assume that the plane of the schedule is one-level. There are repeating parts that consist of n_{period} repeating sections:

$$TIME = \left\{time_1, time_2, \dots, time_{n_{period}}\right\}. \tag{12}$$

Value

Different attribute values (properties) can be assigned to the building blocks of the model. Based on the value set, the following attribute types can be distinguished:

Static:
$VALUE$: a real number, so $VALUE \in \mathbb{R}^+$
Stochastic:
$VALUE = \{value_1, value_2, \ldots, value_n\}$, where n indicates the number of values for each (pos_j^i, pos_j^i) arc.
$value_n = \{real\_number_n, probability_n\}$, where $real\_number \in \mathbb{R}^+, probability \in \{0,1\}$, and $\sum probability = 1$.
Fuzzy:
$VALUE = \{number_1, number_2, number_3, number_4\}$ where $number \in \mathbb{R}^+$.
Forecasted:
$VALUE$ is forecasted from the following set $\{value_1, value_2, \ldots, value_p\}$, where $value_p = \{time_p, real\_number_p\}$, $real\_number \in \mathbb{R}^+$.

3.2. Attributes

Based on the meaning, several attributes can be defined, which are summarized in *Table 1*. In the table i means the level index, j means the position index within the level, k means the vehicle index, and t means the time index.

Meaning of attribute	Notation	Value type
Node attributes		
Travel time between the nodes	$TTBN(i_1, j_1, i_2, j_2, k, t)$	static, stochastic, fuzzy, forecasted
Travel distance between the nodes	$TDBN(i_1, j_1, i_2, j_2, k, t)$	static, stochastic, fuzzy, forecasted
Safety between the nodes	$SBN(i_1, j_1, i_2, j_2, k, t)$	static, stochastic, fuzzy, forecasted
Route quality between the nodes	$RSBN(i_1, j_1, i_2, j_2, k, t)$	static, stochastic, fuzzy, forecasted
Type of the node	$TN(i_1, j_1)$	customer, depot, satellite, charger station
Vehicle attributes		
The capacity constraint of each vehicle types per product type	$CCVTPT(k, m)$	static, stochastic, fuzzy, forecasted
Fuel consumption	$FC(k)$	static, stochastic, fuzzy, forecasted
Charger time	$RTV(k)$	static, stochastic, fuzzy, forecasted
Own vehicle or rented vehicle	$OBV(k)$	{true, false}
Rental fee per vehicle types	$RFVT(k)$	static, stochastic, fuzzy, forecasted
Maximum distance with a full tank / full charge	$MDFT(k)$	static, stochastic, fuzzy, forecasted
Time attributes		
Service time	$ST(i_1, j_1, m, k, t)$	static, stochastic, fuzzy, forecasted
Packing time	$PT(i_1, j_1, m, k, t)$	static, stochastic, fuzzy, forecasted
Unpacking time	$UPT(i_1, j_1, m, k, t)$	static, stochastic, fuzzy, forecasted
Loading time	$LT(i_1, j_1, m, k, t)$	static, stochastic, fuzzy,

		forecasted
Unloading time	$ULT(i_1, j_1, m, k, t)$	static, stochastic, fuzzy, forecasted
Product downtime	$PDT(i_1, j_1, m, k, t)$	static, stochastic, fuzzy, forecasted
Administration time	$AT(i_1, j_1, m, k, t)$	static, stochastic, fuzzy, forecasted
Quality control time	$QCT(i_1, j_1, m, k, t)$	static, stochastic, fuzzy, forecasted
Time window	$TW(i_1, j_1, m, t)$	$\{[twe_1, twl_1], \ldots [twe_q, twl_q]\}$ where twe_q, twl_q can be static, stochastic, fuzzy, forecasted. A twe_q indicates the earliest time at which the products can be at the node, while twl_q indicates the latest time at which the products must be processed at the node.
Product attributes		
The capacity constraint of the node	$CCN(i_1, j_1, m, t)$	static, stochastic, fuzzy, forecasted
Product demand of the node	$PDN(i_1, j_1, m, t)$	static, stochastic, fuzzy, forecasted
Prices- of product	$PP(i_1, j_1, m, t)$	static, stochastic, fuzzy, forecasted
Processing order	$PO(i_1, j_1, m, t)$	{0,1, ... ,noproducttype}
Processing constraint	$PC(i_1, j_1, m_1, m_2)$	{true, false}
Storage level at the locations	$DSSL(i_1, j_1, m, t)$	static, stochastic, fuzzy, forecasted
Cost attributes		
Packaging costs	$PC(i_1, j_1, m, k, t)$	static, stochastic, fuzzy, forecasted
Unpacking cost	$UPC(i_1, j_1, m, k, t)$	static, stochastic, fuzzy, forecasted
Loading costs	$LC(i_1, j_1, m, k, t)$	static, stochastic, fuzzy, forecasted
Unloading costs	$ULC(i_1, j_1, m, k, t)$	static, stochastic, fuzzy, forecasted
Administrative costs	$AC(i_1, j_1, m, k, t)$	static, stochastic, fuzzy, forecasted
Quality control cost	$QCC(i_1, j_1, m, k, t)$	static, stochastic, fuzzy, forecasted
Operational parameter attributes		
Inter-depot route	$IDR(k)$	{true,false}
Delivery	$D(i_1, j_1, m)$	{true,false}
Pickup	$PU(i_1, j_1, m)$	{true,false}
Soft time window	$STW(i_1, j_1, m)$	{true,false}
Open route	$OR(k)$	{true,false}

Table 1.: List of attributes

3.3. Decision variable

The decision variable can have a value of 0 or 1. It has 7 indexes which are the followings: i_1: level of the starting point, i_2: level of the ending point, j_1: index of the starting point in level i_1, j_2: index of the ending point in level i_2, k: index of vehicle, t: index of the time, and m: index of the product.

$$\lambda_{j_1,j_2}^{i_1,i_2,k,t,m} = \begin{cases} 1, if\ vehicle\ k\ in\ period\ t\ \ after\ node\ j_1\ in\ level\ i_1\ travels\ to \\ node\ j_2\ in\ level\ i_2\ inmediately \\ and\ transport\ product\ m, \\ 0, else. \end{cases} \tag{13}$$

3.4. Constraints

Constraints are an important feature of optimization. In order to get a valid solution, various constraints need to be defined.

<u>Constraint 1:</u>

A node at level i only needs to be served maximum once a period by a vehicle with product m (here $n_{position_i}$ is the number of nodes at level i):

$$\forall j_2 \in POS, \forall i_2 \in LEVEL, \forall t \in TIME, \forall m \in PRODUCTTYPE:$$

$$\sum_{i_1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{k=1}^{n_{vehicle}} \lambda_{j_1,j_2}^{i_1,i_2,k,t,m} \leq 1, \tag{14}$$

$$\forall j_1 \in POS, \forall i_1 \in LEVEL, \forall i_2 \in LEVEL, \forall t \in TIME, \forall m \in PRODUCTTYPE:$$

$$\sum_{i_2}^{n_{level}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \lambda_{j_1,j_2}^{i_1,i_2,k,t,m} \leq 1. \tag{15}$$

<u>Constraint 2:</u>

Time window:

This constraint is optional and should only be considered if a time window has been defined. If a time window is defined, two cases are possible, the hard time window and the soft time window. In the case of a hard time window, the time window must be taken into consideration, while the soft time window is an optimization parameter.

1.) Hard time window case

This part of the constraint is also optional, even if a time window is defined because it can be only considered if $STW(i_1, j_1, m)$=false so in case of the hard time window. In this case, the time window must be observed, so service can take place within the given interval.

$a_{j_1}^{i_1,k,t,m} \in \mathbb{R}^+, a_{j_1}^{i_1,k,t,m} \in [twe_{j_1}^{i_1,t,m}, twl_{j_1}^{i_1,t,m} - st_{j_1}^{i_1,k,t,m}]$ the date of service of the product m in level i_1 position j_1 with vehicle k in time t. The following limits can be defined:

The earliest service to each position:

$$\forall j_1 \in POS, \forall i_1 \in LEVEL, \forall k \in VEHICLE, \forall t \in TIME, \forall m \in PRODUCTTYPE, \forall twe_{j_1}^{i_1,t,m} \in TW(i_1, j_1, m, t), \forall\, t \in TIME: \quad twe_{j_1}^{i_1,t,m} \le a_{j_1}^{i_1,k,t,m} \tag{16}$$

The latest service to each position:

$$\forall twl_{j_1}^{i_1,t,m} \in TW(i_1, j_1, m, t), \forall j_1 \in POS, \forall i_1 \in LEVEL, \forall k \in VEHICLE, \forall t \in TIME, \forall m \in PRODUCTTYPE: \quad a_{j_1}^{i_1,k,t,m} + st_{j_1}^{i_1,k,t,m} \le twl_{j_1}^{i_1,t,m}, \tag{17}$$

where

$$st_{j_1}^{i_1,k,t,m} = ST(i_1, j_1, k, m, t) + PT(i_1, j_1, m, k, t) + UPT(i_1, j_1, m, k, t) + LT(i_1, j_1, m, k, t) + ULT(i_1, j_1, m, k, t) + PDT(i_1, j_1, m, k, t) + AT(i_1, j_1, m, k, t) + QCT(i_1, j_1, m, k, t). \tag{18}$$

If $a_{j_1}^{i_1,k,t,m} = 0$, then the given products are not transported by the given vehicle to the given node.

2.) Soft time window case

This part of the constraint is also optional, it is taken into account only if $STW(i_1, j_1, m)$=true, so a soft time window is given. In this case, the time window does not have to be observed, so service can take place outside the given interval, then the solution get a penalty point:

$a_{j_1}^{i_1,k,t,m} \in \mathbb{R}^+$ means the service time of position j_1 in level i_1 with product m in time t with vehicle k:

$$st_{j_1}^{i_1,k,t,m} = ST(i_1, j_1, k, m, t) + PT(i_1, j_1, m, k, t) + UPT(i_1, j_1, m, k, t) + LT(i_1, j_1, m, k, t) + ULT(i_1, j_1, m, k, t) + PDT(i_1, j_1, m, k, t) + AT(i_1, j_1, m, k, t) + QCT(i_1, j_1, m, k, t). \tag{19}$$

If $a_{j_1}^{i_1,k,t,m} = 0$, then the given products are not transported by the given vehicle to the given node.

The value of the penalty point:

If $a_{j_1}^{i_1,k,t,m} \le twe_{j_1}^{i_1,t,m}$ then $d_{j_1}^{i_1,k,t,m} = twe_{j_1}^{i_1,t,m} - a_{j_1}^{i_1,k,t,m}$,

if $twl_{j_1}^{i_1,t,m} - st_{j_1}^{i_1,k,t,m} \le a_{j_1}^{i_1,k,t,m}$ then $d_{j_1}^{i_1,k,t,m} = a_{j_1}^{i_1,k,t,m} - twl_{j_1}^{i_1,t,m} - st_{j_1}^{i_1,k,t,m}$,

else $d_{j_1}^{i_1,k,t,m} = 0$.

<u>Constraint 3</u>

Vehicles start from an aggregated level position and then terminate at a aggregated-level position after visiting the lower-level positions. This constraint is optional, it is only not met if there is a single level. This constraint consists of three parts.

1.) First, there is a level change, namely from the upper level to the lower level. This sub-constraint within the constraint is not optional.

$$\forall i_2 - i_1 = 1, \forall i_1 \in LEVEL, \forall i_2 \in LEVEL, \forall k \in VEHICLE, \forall t \in TIME, \forall m \in PRODUCTTYPE:$$

$$\sum_{j_2=1}^{n_{position_{i_2}}} \lambda_{j_1,j_2}^{i_1,i_2,k,t,m} = 1. \tag{20}$$

2.) After the level change, we stay at the level below. This sub-constraint within the constraint is not optional.

$$\forall i_2 = i_1, \forall i_1 \in LEVEL, \forall i_2 \in LEVEL, \forall k \in VEHICLE, \forall t \in TIME, \forall m \in PRODUCTTYPE:$$

$$\sum_{j_2=1}^{n_{position_{i_2}}} \lambda_{j_1,j_2}^{i_1,i_2,k,t,m} = 1. \tag{21}$$

3.) Then there must be a change of level, namely to the aggregated level from which we started. This sub-constraint is optional within the constraint, only to be considered if $OR(vehicle_k)$=false, so it is not an open route. If $OR(vehicle_k)$=true, this step does not occur.

$$\forall i_2 - i_1 = 1, \forall i_1 \in LEVEL, \forall i_2 \in LEVEL, \forall k \in VEHICLE, \forall t \in TIME, \forall m \in PRODUCTTYPE:$$

$$\sum_{j_2=1}^{n_{position_{i_2}}} \lambda_{j_1,j_2}^{i_1,i_2,k,t,m} = 1. \tag{22}$$

Within this sub-constraint, two additional sub-constraints can be distinguished according to whether the vehicle should return to the aggregated-level node from which it started or may return to another aggregated-level node:

a.) This sub-constraint means that the vehicle must return to the same aggregated-level node from which it started. This is optional, only to be considered if $IDR(k)$=false, so there is no inter-depot route:

$$\forall i_2 - i_1 = 1, \forall i_1 \in LEVEL, \forall i_2 \in LEVEL, \forall k \in VEHICLE, \forall t \in TIME, \forall m \in PRODUCTTYPE: \tag{23}$$

If

$$\lambda_{j_1,j_2}^{i_1,i_2,k,t,m} = 1, \tag{24}$$

then

$$\forall i_3 - i_1 = 1, \forall i_1 \in LEVEL, \forall i_3 \in LEVEL, \forall k \in VEHICLE, \forall t \in TIME, \forall m \in PRODUCTTYPE:$$

$$\lambda_{j_3,j_1}^{i_3,i_1,k,t,m} = 1\,. \tag{25}$$

b.) This sub-constraint is also optional, it is taken into account only if there is an inter-depot route, so $IDR(vehicle_k)$=true:

$$\forall i_2 - i_1 = 1, \forall i_1 \in LEVEL,, \forall i_2 \in LEVEL, \forall k \in VEHICLE, \forall t \in TIME, \forall m \in PRODUCTTYPE \quad (26)$$

If

$$\lambda_{j_1,j_2}^{i_1,i_2,k,t,m} = 1, \quad (27)$$

then

$$\forall i_3 - i_4 = 1, \forall i_4 \in LEVEL, \forall i_3 \in LEVEL, \forall k \in VEHICLE, \forall t \in TIME, \forall m \in PRODUCTTYPE:$$

$$\lambda_{j_3,j_4}^{i_3,i_4,k,t,m} = 1\,. \quad (28)$$

Here we allow $j_4 = j_1$, but also that $j_4 \neq j_1$.

Constraint 4

This constraint is not optional and should always be considered when transporting products. Vehicles must comply with their capacity limit:

$$\forall j_2 \in POS, \forall i_1 \in LEVEL, \forall i_2 \in LEVEL, \forall k \in VEHICLE, \forall t \in TIME, \forall m \in PRODUCTTYPE:$$

$$\sum_{j_1=1}^{n_{position_{i_1}}} PDN(i_1, j_1, m, t) \cdot \lambda_{j_1,j_2}^{i_1,i_2,k,t,m} \leq CCVTPT(k, m). \quad (29)$$

Constraint 5

This constraint is not optional and always be considered when transporting products. Positions must take into account their capacity limit:

$$\forall j_2 \in POS, \forall i_1 \in LEVEL, \forall i_2 \in LEVEL, \forall k \in VEHICLE, \forall t \in TIME, \forall m \in PRODUCTTYPE:$$

$$\sum_{j_1=1}^{n_{position_{i_1}}} PDN(i_1, j_1, m, t) \cdot \lambda_{j_1,j_2}^{i_1,i_2,k,t,m} \leq CCN(i_1, j_1, m, t) + DSSL(i_1, j_1, m, t). \quad (30)$$

Constraint 6

This constraint is not optional. For each level the number of transport edges must not exceed the number of vehicles available at each level:

$$\forall j_2 \in POS, \forall i_2 \in LEVEL, \forall t \in TIME, \forall m \in PRODUCTTYPE:$$

$$\sum_{i_1=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{k=1}^{n_{vehicle}} \lambda_{j_1,j_2}^{i_1,i_2,k,t,m} \leq n_{vehicle}, \quad (31)$$

$$\forall j_1 \in POS, \forall i_1 \in LEVEL, \forall t \in TIME, \forall m \in PRODUCTTYPE:$$

$$\sum_{i_2=1}^{n_{level}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \lambda_{j_1,j_2}^{i_1,i_2,k,t,m} \leq n_{vehicle}. \tag{32}$$

Constraint 7

The following constraint is not optional. It relates to subpath elimination:

$$\forall j_1 \in POS, \forall j_2 \in POS, \forall i_1 \in LEVEL, \forall i_2 \in LEVEL, \forall k \in VEHICLE, \forall t \in TIME, \forall m \in PRODUCTTYPE:$$

$$a_{j_2}^{i_2,k,t,m} - st_{j_1}^{i_1} - a_{j_1}^{i_1,k,t,m} \leq M\left(1 - \lambda_{j_1,j_2}^{i_1,i_2,k,t,m}\right). \tag{33}$$

Constraint 8

The constraint is not optional. It relates to route continuity. It says, that the number of incoming must be equal with the number of outcoming edges in each position j:

$$j_3 \in POS, \forall i_3 \in LEVEL, \forall k \in VEHICLE, \forall t \in TIME, \forall m \in PRODUCTTYPE:$$

$$\sum_{j_1=1}^{n_{position_{i_1}}} \lambda_{j_1,j_3}^{i_1,i_3,k,t,m} = \sum_{j_2=1}^{n_{position_{i_2}}} \lambda_{j_3,j_2}^{i_3,i_2,k,t,m}. \tag{34}$$

Constraint 9

The constraint is not optional. Vehicles require charging after a certain distance, so the vehicles need to visit the charger station:

$$\forall k \in VEHICLE, \forall t \in TIME, \forall m \in PRODUCTTYPE:$$

$$\begin{aligned} &\sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} TDBN(i_1, j_1, i_2, j_2, k, t) \cdot \lambda_{j_1,j_2}^{i_1,i_2,k,t,m} \\ &\quad + \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{j_3=1}^{n_{position_{i_3}}} TDBN(i_2, j_2, i_3, j_3, k, t) \cdot \lambda_{j_2,j_3}^{i_2,i_3,k,t,m} \\ &\quad \leq MDFT(k), \\ &j_1 = j_2, j_2 - j_1 = 1, j_3 - j_1 = 1. \end{aligned} \tag{35}$$

The value of $MDFT(vehicle_k)$ can also be ∞, in which case the vehicle does not need to charge.

Constraint 10

The constraint is optional, only to be considered when defining a fixed order of products. The products arriving in each position can have a fixed order, they must arrive one after the other:

$$\forall i_1 \in LEVEL, \forall j_1 \in POS, \forall i_2 \in LEVEL, \forall j_2 \in POS, \forall k \in VEHICLE, \forall t \in TIME, \forall m_1 \in PRODUCTTYPE, \forall m_2 \in PRODUCTTYPE.$$

$$\text{If } PO(i_1, j_1, m_1, t) \leq PO(i_1, j_1, m_2, t) \text{ then } a_{j_1}^{i_1,k,t,m_1} \leq a_{j_1}^{i_1,k,t,m_2}. \tag{36}$$

Constraint 11

The constraint is optional and will only be considered if there are products that cannot be shipped together:

$$\forall i_1 \in LEVEL, \forall j_1 \in POS, \forall i_2 \in LEVEL, \forall j_2 \in POS, \forall t \in TIME, \forall m_1 \in PRODUCTTYPE, \forall m_2 \in PRODUCTTYPE. \tag{37}$$

If $PC(i_1, j_1, m_1, m_2)$=false and $\lambda_{j_1,j_2}^{i_1,i_2,k,t,m_1} = 1$, then $\lambda_{j_1,j_2}^{i_1,i_2,k,t,m_2} = 0$.

Figure 4. illustrates the optional and obligatory constraints.

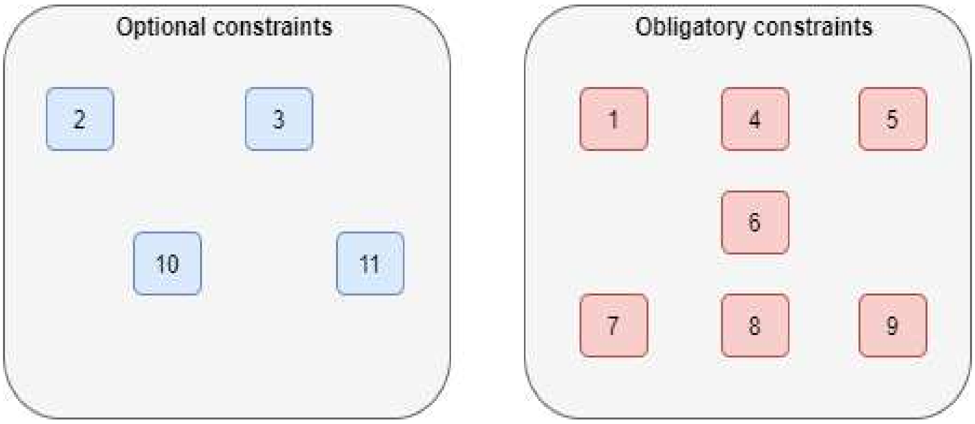

Figure 4.: Optional and obligatory constraints

3.5. Metrics, objective function components

Each objective function component can be formed from the metrics. In my case, the goal is to minimize for most metrics. In this section, I use the following notations:

- w_i: metrics,
- z_i: objective function components,
- p_i: penalty point elements,
- c_i: weights of each metrics.

Length of the route:

Minimizing the length of the route is an important factor, profit is greatly influenced by how far the vehicles travel. It is related to fuel consumption, human resource costs, the actual number of vehicles, etc. therefore, it is a very important factor to consider:

$$\forall m \in PRODUCTTYPE:$$

$$c_{MR} = \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} TDBN(i_1, j_1, i_2, j_2, k, t) \cdot \lambda_{j_1,j_2}^{i_1,i_2,k,t,m}, \tag{38}$$

$$z_{MR} = min\, c_{MR}. \tag{39}$$

Transported value:
The main factor of profit depends on the values sold. The objective is to sell more and valuable products to a higher profit. The long-term objective of the company is also to satisfy the demands of the customers and to fulfil the order, as this expands the company's customer:

$$c_{TV} = \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} \sum_{m=1}^{n_{producttype}} PP(i_2, j_2, m, t) \cdot \lambda_{j_1,j_2}^{i_1,i_2,k,t,m}, \quad (40)$$

$$z_{TV} = max\, c_{TV}. \quad (41)$$

Packaging cost:
Products can be packaged in the nodes, in which case there is a packing cost:

$$c_{PC} = \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} \sum_{m=1}^{n_{producttype}} PC(i_1, j_1, m, k, t) \cdot \lambda_{j_1,j_2}^{i_1,i_2,k,t,m}, \quad (42)$$

$$z_{PC} = min\, c_{PC}. \quad (43)$$

Unpacking cost:
Products can be unpacked in the nodes, in which case there is an unpacking cost:

$$c_{UC} = \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} \sum_{m=1}^{n_{producttype}} UPC(i_2, j_2, m, k, t) \cdot \lambda_{j_1,j_2}^{i_1,i_2,k,t,m}, \quad (44)$$

$$z_{UC} = min\, c_{UC}. \quad (45)$$

Loading cost:
Products are loaded into vehicles in the nodes, then loading costs are incurred:

$$c_{LC} = \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} \sum_{m=1}^{n_{producttype}} LC(i_1, j_1, m, k, t) \cdot \lambda_{j_1,j_2}^{i_1,i_2,k,t,m}, \quad (46)$$

$$z_{LC} = min\, c_{LC}. \quad (47)$$

Unloading cost:
Products are unloaded from the vehicles in the nodes, then an unloading cost is incurred:

$$c_{UC} = \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} \sum_{m=1}^{n_{producttype}} ULC(i_2, j_2, m, k, t) \cdot \lambda_{j_1,j_2}^{i_1,i_2,k,t,m}, \tag{48}$$

$$z_{UC} = min\, c_{UC}. \tag{49}$$

Administrative costs:
There may be an administrative cost when arriving products at each node or transporting products from the nodes:

$$c_{AC} = \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} \sum_{m=1}^{n_{producttype}} [AC(i_1, j_1, m, k, t) + AC(i_2, j_2, m, t)] \cdot \lambda_{j_1,j_2}^{i_1,i_2,k,t,m}, \tag{50}$$

$$z_{AC} = min\, c_{AC}. \tag{51}$$

Quality control cost:
There may be a quality control cost when arriving products in the nodes or transporting products from the nodes to test the quality of the products:

$$c_{QCC} = \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} \sum_{m=1}^{n_{producttype}} [QCC(i_1, j_1, m, k, t) + QCC(i_2, j_2, m, k, t)] \cdot \lambda_{j_1,j_2}^{i_1,i_2,k,t,m}, \tag{52}$$

$$z_{QCC} = min\, c_{QCC}. \tag{53}$$

Fuel consumption:
One of the main transportation costs is the cost of fuel. This depends on the route taken by the vehicles, the type of vehicles, and may depend on the number of products delivered by the vehicles:

$$\forall m \in PRODUCTTYPE:$$

$$c_{FC} = \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} TDBN(i_1, j_1, i_2, j_2, t) \cdot FC(k, t) \cdot \lambda_{j_1,j_2}^{i_1,i_2,k,t,m}, \tag{54}$$

$$z_{FC} = min\, c_{FC}. \tag{55}$$

Vehicle rental fee:

In the case that the company is unable to serve the demands of all customers with its fleet of vehicles, it will have to rent vehicles. The rental cost is an extra cost to the company, so the objective may be to pay as little rental fee as possible for the company:

$$c_{VRF} = \sum_{k=1}^{n_{vehicle}} RFVT(k), \tag{56}$$

$$z_{VRF} = min\, c_{VRF}. \tag{57}$$

Safety between nodes:

An important factor is that the route along which the vehicle travels is safe. If the route is not very safe, the products may be stolen so they do not arrive at their destination:

$$\forall m \in PRODUCTTYPE:$$

$$c_R = \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} SBN(i_1, j_1, i_2, j_2, k, t) \cdot \lambda_{j_1, j_2}^{i_1, i_2, k, t, m}, \tag{58}$$

$$z_S = max\, c_S. \tag{59}$$

Route quality:

An important factor is that the road the vehicle travels along is in good condition. If the road is not in very good condition, the products may be damaged and the vehicle may be also damaged:

$$\forall m \in PRODUCTTYPE:$$

$$c_{RQ} = \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} RQBN(i_1, j_1, i_2, j_2, k, t) \cdot \lambda_{j_1, j_2}^{i_1, i_2, k, t, m}, \tag{60}$$

$$z_{RS} = max\, c_{RQ}. \tag{61}$$

Route time:

It presents, how long the route time takes.

$$\forall m \in PRODUCTTYPE:$$

$$c_{RT} = \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} TTBN(i_1, j_1, i_2, j_2, k, t) \cdot \lambda_{j_1, j_2}^{i_1, i_2, k, t, m}, \tag{62}$$

$$z_{RT} = min\, c_{RT}. \tag{63}$$

Packing time:

Products can be packed in the nodes, the time of which is shown by the packing time.

$$c_{PT} = \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} \sum_{m=1}^{n_{producttype}} PT(i_1, j_1, m, k, t) \cdot \lambda_{j_1,j_2}^{i_1,i_2,k,t,m}, \tag{64}$$

$$z_{PT} = min\, c_{PT}. \tag{65}$$

Unpacking time:

Products can be unpacked in the nodes, the time of which is shown by the unpacking time.

$$c_{UP} = \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} \sum_{m=1}^{n_{producttype}} UPT(i_2, j_2, m, k, t) \cdot \lambda_{j_1,j_2}^{i_1,i_2,k,t,m}, \tag{66}$$

$$z_{UP} = min\, c_{UP}. \tag{67}$$

Loading time:

Products can be loaded in the nodes, the time of which is shown by the loading time.

$$c_{LT} = \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} \sum_{m=1}^{n_{producttype}} LT(i_1, j_1, m, k, t) \cdot \lambda_{j_1,j_2}^{i_1,i_2,k,t,m} + ULT(i_2, j_2, m, k, t) \cdot \lambda_{j_1,j_2}^{i_1,i_2,k,t,m}, \tag{68}$$

$$z_{LT} = min\, c_{LT}. \tag{69}$$

Product downtime:

For the product, downtime can also occur in the nodes:

$$c_{CT} = \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} \sum_{m=1}^{n_{producttype}} PDT(i_2, j_2, m, k, t) \cdot \lambda_{j_1,j_2}^{i_1,i_2,k,t,m}, \tag{70}$$

$$z_{PDT} = min\, c_{PDT}. \tag{71}$$

<u>Administrative time:</u>
For the products, administration time is also possible in the nodes:

$$c_{AT} = \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} \sum_{m=1}^{n_{producttype}} [AT(i_1, j_1, m, k, t) + AT(i_2, j_2, m, k, t)] \cdot \lambda_{j_1,j_2}^{i_1,i_2,k,t,m}, \tag{72}$$

$$z_{AT} = min\, c_{AT}. \tag{73}$$

<u>Quality control time:</u>
Quality control time in the nodes is also possible for the products:

$$c_{QCT} = \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} \sum_{m=1}^{n_{producttype}} [QCT(i_1, j_1, m, k, t) + QCT(i_2, j_2, m, k, t)] \cdot \lambda_{j_1,j_2}^{i_1,i_2,k,t,m}, \tag{74}$$

$$z_{QCT} = min\, c_{QCT}. \tag{75}$$

<u>Fuel filling time:</u>
At the charger station, the vehicles are charged, the time of which is indicated by the fuel filling time:

$$\forall m \in PRODUCTTYPE:$$
$$c_{RTV} = \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} RTV(i_2, j_2, k, t) \cdot \lambda_{j_1,j_2}^{i_1,i_2,k,t,m}, \tag{76}$$

$$z_{RTV} = min\, c_{RTV}. \tag{77}$$

<u>Waiting time of the nodes:</u>
Vehicles may arrive at nodes sooner than the demand could be served. In this case, the products are forced to wait in the nodes:

$$c_{WTN} = \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} \sum_{m=1}^{n_{producttype}} (a_{j_2}^{i_2,k,t,m} - b_{j_2}^{i_2,k,t,m}) \cdot \lambda_{j_1,j_2}^{i_1,i_2,k,t,m}, \tag{78}$$

$$z_{WTN} = min\, c_{WTN}. \tag{79}$$

Exceeding the time window:

Vehicles may arrive at nodes sooner than the demand should be served. In this case, they can process earlier, but they can get extra penalty point:

$$p_{ETW} = \sum_{i_1=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} \sum_{m=1}^{n_{producttype}} d_{j_1}^{i_1,k,t,m}, \tag{80}$$

$$z_{ETW} = min\ p_{\text{ETW}}. \tag{81}$$

Unvisited customers:

It is possible that the vehicles will not be able to serve all product demands, in which case certain product demands will not be satisfied, and customers will be not visited:

$$\forall m \in PRODUCTTYPE:$$

$$\begin{aligned} p_{UC} = & \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{\infty} \sum_{t=1}^{n_{period}} \lambda_{j_1,j_2}^{i_1,i_2,k,t,m} \\ & - \sum_{i_1=1}^{n_{level}} \sum_{i_2=1}^{n_{level}} \sum_{j_1=1}^{n_{position_{i_1}}} \sum_{j_2=1}^{n_{position_{i_2}}} \sum_{k=1}^{n_{vehicle}} \sum_{t=1}^{n_{period}} \lambda_{j_1,j_2}^{i_1,i_2,k,t,m}. \end{aligned} \tag{82}$$

$$z_{UC} = min\ p_{UC}. \tag{83}$$

Objective function

The objective function is a function of the metrics.

$$\begin{aligned} Z = F\big(& z_{MR}, z_{TV}, z_{PC}, z_{UC}, z_{LC}, z_{UC}, z_{AC}, z_{QCC}, z_{FC}, z_{VRF}, z_S, z_{RQ}, z_{RT}, z_{PT}, z_{UT}, z_{LT}, \\ & z_{PDT}, z_{AT}, z_{QCT}, z_{RTV}, z_{WTN}, z_{ETW}, z_{UC}\big). \end{aligned} \tag{84}$$

The *Equation 84.* is a function of the metrics. The following multi-objective optimization techniques are most commonly used: Weighted-sum method, Weighted-exponential sum method, Weighted global criterion method, Exponentially weighted criterion, Weighted product method, Bounded objective function method, Pareto ranking. I will write more about this in *Appendix 10.*

Thesis 1:

I have developed a novel general Vehicle Routing Problem model that is suitable (both in terms of objective function and constraints) for solving different types of transportation problems. In the model, there are 4 basic components between which different attributes can be defined. The basic components are graph descriptors, vehicles, products, services, and time. I defined the relations between the basic components with functions. In this novel model the relations can be assigned to 6 large groups, including nodes, vehicles, time, products, costs, and operational parameters. The novelty of this complex VRP model is the one complex decision variable, which includes a wide range of decision variables of conventional VRP models. The solutions of the defined complex VRP problems are limited by different constraints. The model includes 11 constraints and 23 metrics.

Related publications: [P/1], [P/5], [P/6], [P/8], [P/16], [P/17], [P/22], [P/23], [P/24], [P/25]

4. The ontology model of the generalized Vehicle Routing Problem

Ontology is a standard tool for knowledge representation. The term comes from philosophy, where it means a systematic description of existence. Ontology describes a domain, its concepts, and the relationships between concepts. Its purpose is not only the presentation of information for humans but the machine processing of the content of the information. The most common general ontology language is OWL, developed by the World Web Consortium (W3C). OWL helps automatic machine interpretation of web content to a greater extent than XML, RDF, and RDF Schema (RDF-S). OWL also offers extended vocabulary and formal semantics [48,49].

Ontologies are used mainly in distributed applications, where users need to work together. The ontology defines the concepts necessary for the representation and description of the application area, helps to unify their management. Ontologies are used by databases, applications, users who need to work together in a particular area of knowledge (e.g. medicine, financial management, etc.). Ontologies contain the basic concepts of the knowledge area and their contexts in a computer-interpretable form.

Considering an ontology description, it should contain the following components [50]:

- classes (general things) in a wide range of subject areas,
- relevant relationships between things,
- the properties (or attributes) that these things may have.

Ontologies are formulated to make precise, clear, meaningful distinctions between classes, attributes, and relationships between things [50].

The main role of the ontology is to perform the following operations in efficient way:

- conclusion,
- specialization,
- validation,
- reasoning.

4.1. OWL (Web Ontology Language)

The most common general ontology language is OWL, developed by the World Web Consortium (W3C). OWL also offers an extended vocabulary and formal semantics. The first version was developed in 2004 and the second version was created in 2012. OWL is a commonly used, widespread format, whose three levels are distinguished: OWL Lite, OWL DL and OWL Full.

4.1.1. Components of the OWL ontology

<u>Classes:</u>

Classes are abstract set of similar individuals belonging to the same category. Classes can be considered as a set of individuals. Each class has a logical formula that describes whether an individual belongs to a particular class. Classes can be organized in a subclass-parent class hierarchy, this is also called a taxonomy. For example, if the animal and cat classes are considered, then the animal is the parent class, the cat is the descendant. This means that every cat is an animal. Every element of the cat class is also the animal class. [48]

In the terminology of OWL, every individual is a member of the top class called 'owl: Thing' thus, every defined class is a subclass of the 'owl:Thing' class. There is also an empty class in OWL called 'owl:Nothing'. [50]

'Rdfs:subClassOf' defines a more specific class. It classifies this more specific class under a more general class. If 'X' is a subclass of 'Y', then every individual of 'X' is also an individual of 'Y'. [50]

OWL assumes an open approach. That is, descriptions are not limited to a single file. For example, if class ‘C1’ was originally defined in the ‘O1’ ontology, it can be extended in other ontologies. [50]

Properties:
OWL properties represent relationships. There are two main property types: object properties and datatype properties. Object properties describe the relationship between two objects. Object properties link one object to another. Data properties connect individuals with literals. There is another type of property in OWL, it is the annotation property, which is used to add metadata to classes, individuals, object/datatype properties. [48]
Object properties can be further broken down, as shown in the following category [48]:

- Inverse property: The object property is inverted. If one property points from ’A’ to ’B’, its inverse points from ’B’ to ’A’. For example, the parent (’hasParent’) property is the inverse of the child (’hasChild’).
- Functional property: A property is functional for a given individual, then there is maximum one individual that is related to the individual through the property. The relationship of her/his mother (’hasBirthMother’) is a functional property because an individual can have a single mother.
- Inverse functional property: It means that the inverse of the property is functional. For a given individual, there must be maximum one individual that is associated with that individual with that property. The inverse functional property of his/her mother (’hasMother’) is his/her mother (’isBirthMotherOf’).
- Transitive property: If the property applies to an individual ’B’ from an individual ’A’, and also to an individual ’C’ from an individual ’B’, it can be concluded that it also applies to an individual ’C’ from an individual ’A’.
- Symmetric property: If the property applies from ’A’ to ’B’, then it also applies from ’B’ to ’A’. For example, Matthew's sibling (’hasSibling’) is ’Gemma’, and Gemma's sibling (’hasSibling’) is ’Matthew’.
- Asymmetric property: If entity ’A’ is associated with entity ’B’ by property ’P’, entity ’B’ cannot be associated with entity ’A’ by property ’P’.
- Reflexive property: If the property is associated with an individual ’A’ itself.
- Irreflexive property: If the property connects an individual ’A’ and ’B’, where individuals ’A’ and ’B’ cannot be the same.

Property domain and range: Properties as special mapping have domains and ranges. Properties link individuals from the domain entity to the range entity. For example, for a ‘Pizza’ ontology, the domain of the ‘hasTopping’ property is ‘Pizza’ and the range is ‘PizzaTopping’. [48]
Property restrictions: Restrictions describe a class of individuals based on the relationships in which class members participate. So the restrictions can be seen as a kind of class.
There are three main categories of restrictions in OWL [48]:

- *Quantifier restrictions:* Quantifier restrictions can be further broken down into existential and universal constraints.
    - *Existential restrictions:* describe classes of individuals that have at least one relationship with individuals in a particular class along with a property.
    - *Universal restrictions:* describe classes of individuals that are associated with only one property in a particular class. In Protege, the ’only’ keyword is used for this.
- *Cardinality restrictions:* In OWL, individuals can be described as: at least, at most, exactly or a certain number of relationships between another entity or data type. or a cardinality restriction. The following cardinalities can be distinguished: maximum cardinality, minimum cardinality, or exactly cardinality restriction.
- *hasValue restrictions:* links a restriction class to a value.

Individuals:
They represent domain objects. These represent the specific individuals. It also has specific properties. [48]

Namespaces
The namespace is one of the identification toolbars. Ontologies begin with a series of XML namespace declarations. These make the ontology easier to read and the interpretation of the identifiers clear [51].

Usage of the reasoner
An important feature of ontologies is that they can be processed with a reasoner. Reasoner is a special machine, that infer logical operations. One of the main tasks of a reasoner is to decide for each class whether or not it is a subclass of another class, whether each class can be instantiated, and whether the system contains a contradiction. In the ontology, by performing such tests on the classes, the reasoner is the inferred ontology class hierarchy. [48]

4.2. Using ontology in logistics

In this subsection, I will summarize the ontology systems of the logistics area.
One of the earliest work is the *ontology for constructing scheduling systems*, which was detailed in *1994* in the publication of *Smith & Becker* [52]. The building blocks of the ontology system are the following five classes: 'Demand', 'Activity', 'Resource', 'Product', and 'Constraint'. 'Demand' is the demand for products and services. The 'Product' is associated with 'Demand'
A *multi-agent logistics scheduler for road transportation* system was proposed by *Himoff, Rzevski & Skobelev* in *2006* [53]. In this paper, a model of intelligent road transport scheduling is presented, which includes several functions such as real-time scheduling, scheduling improvement, scheduling analysis. The scheduler is event-driven, which means that schedules are compiled to take into account all events that affect the schedule (e.g., the arrival of a new order, cancellation of an order, truck failure, lack of travel). When an event occurs, the scheduler reschedules only the affected part, thereby minimizing the change to resources that have already been allocated. Their system consists of three main components: 'Basic', 'Operations', 'Scheduling'.
In an article by *Lian, Park, & Kwon* in [54] published in *2007*, an *ontology for semantic representing of situation in logistics* is performed.
Dong, Hussain, & Chang report on a *transport service ontology* in *2008* [55]. The authors divide the ontological system into layers. The first layer is the root of the hierarchy, which includes all delivery services. The second layer is a specialization of the concept of abstract transport service. The third layer is a further specialization of the second layer, in this service the service concepts can be considered as concrete or abstract concepts. During the third layer, the abstract concepts have additional specializations in the fourth layer.
In their publication [56], in *2010,* the authors *Hoxha, Scheuermann, & Bloehdorn* reported an *ontology model for flexibility in supply chain configuration and management*. The approach combines loosely coupled logistics services and semantic technologies to provide a unified representation of different logistics data and service functions. The authors have created a framework that uses automated, intelligent techniques to discover, rank, and execute efficient assemblies of services.
An *ontology model for exception monitoring service* is presented by *Xu, Wijesooriya, Wang & Beydoun* in *2012* [57]. The system models logistics exceptions. The authors focus on the type of exceptions and the reason for their creation.
Authors *Anand, Yang, van Duin, & Tavasszy* specifically focused on the *ontology model for city logistics* in their article [58], published in *2012*. Their main classes are 'Activity', 'KPI', 'Objective', 'Stakeholder', 'Resource', 'Measure', 'R&D'. Several subclasses and properties have been created for the classes. In their article, they write about the following numbers: number of

classes: 263, number of object properties: 56, number of data properties: 102, number of individuals: 170.
A *transportation ontology* was developed by the author's *De Oliveira, Bacha, Mnasser & Abed* in *2013* [59].
An example of an *ontological approach to* general *logistics* systems can also be seen in the literature. These systems are good because other ontological scientists can supplement the general system with additional classes, relationships, and instances. Such a system was presented by *Daniele & Pires* in *2013* [60].
Ontology model has also been developed *for emergency logistics*, such a system is presented by *Zhang, Jiang, Zeng, Ning, & Wanga* in their paper in *2014* [61].
Ontology model for the Manufacturing Execution System was discussed in *2014* by *Fumagalli, Pala, Garetti & Negri* [62]. The main class of their system is the 'Component' class, which contains 'Storage', 'Transporter', 'Processor', 'Sensor' classes. Each class was also given datatype properties such as 'ID', 'name', 'type', 'capacity', and so on.
An ontology-based system for transporting refrigerated goods will be introduced in *2015* by the authors *Wang, Yi, Zhu, Luo, & Ji* [63], as the supply chain of refrigerated goods is becoming more complex and challenging today.
The LoSe ODP, ontological design pattern sample will be reported in *2017* by *Glöckner & Ludwig* [64]. The composition of the logistics services of different service providers is a difficult task due to the different formulations, descriptions and IT systems. Such logistics service building blocks can be easily implemented with a central ontological design pattern. Data from different service providers (services) can be easily accessed, connected and exchanged within the network.
A detailed description of the presented systems can be found in *Appendix 13 [A/13].*

4.3. The ontology system of the vehicle routing problem

In this subsection, the ontology system of the generalized model of the Vehicle Routing Problem (VRP) is presented. Some of the ontology systems presented in the literature are limited to specific logistics tasks, for example, monitoring service, city logistics, emergency logistics, manufacturing execution system, refrigerated goods. Some ontology models can also be found, that try to cover the whole logistics process. I developed a more general system, which is different from these systems. The system should help the user choose which of the components and metrics of my generalized Vehicle Routing Problem model to use when transporting a given product, to a given node, with a given vehicle. I also saw the need for the system to help decide which products can be shipped with which vehicles to nodes.

4.3.1. The main components of the ontology system

The advantage of ontologies is that ontology systems can work together, so my system can be supplemented with a logistics model developed by others. The developed sample system includes the graph descriptive, vehicles, products, period, set of values (static, stochastic, fuzzy, forecasted values) and attributes (nodes, vehicles, period, products, costs, operational parameter) detailed in the mathematical model. Also, metrics are included in the model. In the sample system, ontological concepts allow for a rule-based validation. The main components of the ontology model are shown in *Figure 5.*

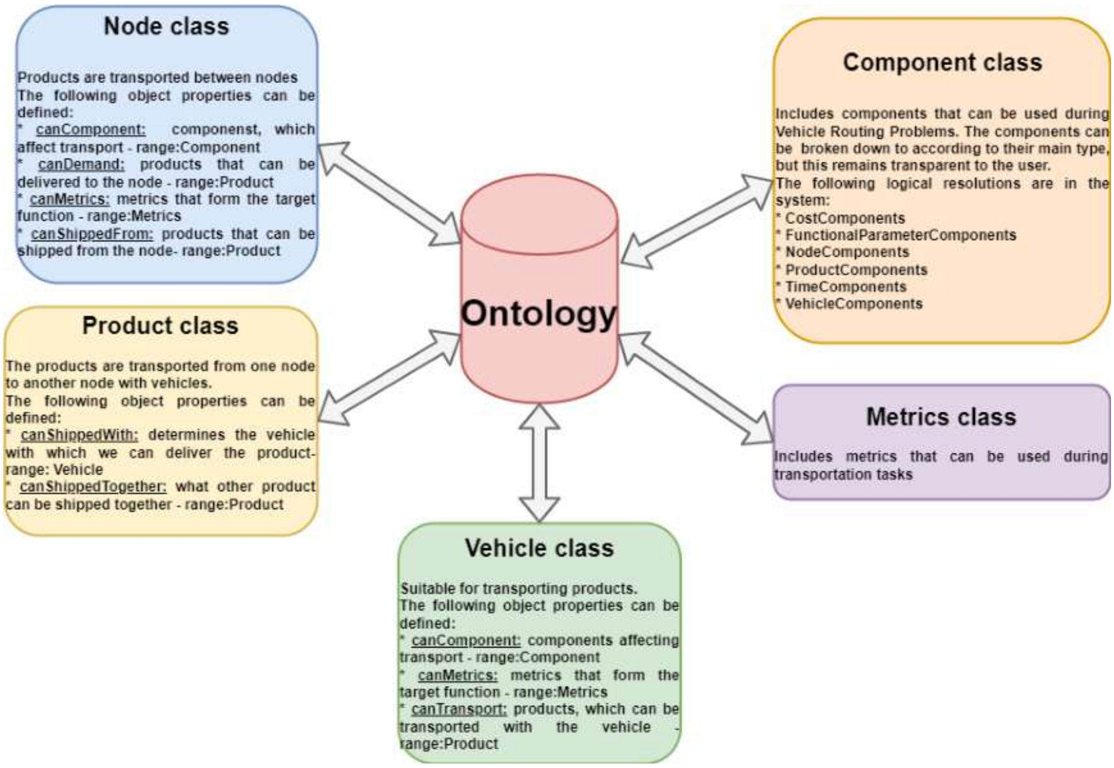

Figure 5.: The ontology system

Due to rule-based validation the subclasses are assigned to certain classes of components detailed in the generalized model. The subclasses of the `Node` class are shown in *Figure 10. ontology.figure [A/2]*. The subclasses of the `Product` are shown in *Figure 11-12. ontology.figure [A/2]*, the subclasses of the `Vehicle` class is illustrated in *Figure 13-17. ontology.figure [A/2]*.
In the ontology, association relationships between classes are specified with an object property. The relationships between the classes of components (*Table 1-2 vrp.table [A/1])* detailed in the generalized model can be seen in *Figure 18-23. ontology.figure [A/2]*. *Figure 18. ontology.figure [A/2]* shows the list of object properties as administration cost. According to the generalized model (*Table 1. vrp.table [A/1]*), the administration cost is a node, product, vehicle, and period dependent, and has a value that can otherwise be static, stochastic, fuzzy or forecasted). Therefore, `administrationCost` object properties are as follows:

- `administrationCost_hasCost`,
- `administrationCost_hasNode`,
- `administrationCost_hasPeriod`,
- `administrationCost_hasProduct`,
- `administrationCost_hasVehicle`.

The domain of these properties is the `AdministrationCost` class and the ranges are `Value`, `Node`, `Period`, `Product` and `Vehicle` class, respectively. For easier query, I also created inverses of object properties:

- `administrationCost_inverse_hasCost`,
- `administrationCost_inverse_hasPeriod`,
- `administrationCost_inverse_hasNode`,
- `administrationCost_inverse_hasProduct`,
- `administrationCost_inverse_hasVehicle`.

The `TimeComponent` is the same, the `administrationTimeOfTheNode` object properties are the followings:

- `administrationTimeOfTheNode_hasAdministrationTime`,
- `administrationTimeOfTheNode_hasNode`,
- `administrationTimeOfTheNode_hasPeriod`,
- `administrationTimeOfTheNode_hasVehicle`.

Because administration time is a function of node, vehicle, period, and means a value. Here, too, I created the inverses of the properties:

- `admininstrationTimeOfTheNode_inverse_hasAdministrationTime`,
- `admininstrationTimeOfTheNode_inverse_hasNode`,
- `admininstrationTimeOfTheNode_inverse_hasPeriod`,
- `admininstrationTimeOfTheNode_inverse_hasVehicle`.

The ontology model does not only include the basic taxonomy of the generalized model, as the aim is a rule-based validation. I had to add different subclasses to the `BaseComponents` class because my goal was to help model outdoor material handling tasks. So, in addition to the `Node`, `Product` and `Vehicle` classes (*Figure 10-17. ontology.figure [A/2]*), I also had to add object properties to the system.
A list of object properties of a `Node` subclass (`Bank`) is shown in *Figure 24. ontology.figure [A/2]*:

- `bank_canComponent`: shows what components a bank can have. (`CostComponent`, `OperationalParameterComponent`, `TimeComponent`, etc. and their subclasses).
- `bank_canDemand`: shows what kind of product demand a bank node can have. This is the subclass of the `Product` class.
- `bank_canMetrics`: shows what metrics can be considered during bank delivery. Here the range is a subclass of `MetricsComponent`.
- `bank_canShippedFrom`: shows what product can be delivered during bank delivery. Here, the range is a subclass of the `Product` class.

A list of object properties of a `Product` subclass (`Book`) is shown in *Figure 25. ontology.figure [A/2]:*

- `book_canShippedWith`: shows what vehicles (subclasses of the `Vehicle` class) can deliver a book (`Book` class).
- `book_shippedTogether`: shows what other product (subclasses of the `Product` class) a book (`Book` class) can be shipped with.

A list of object properties of a `Vehicle` subclass (`Bus` class) is shown in *Figure 26. ontology.figure [A/2]*:

- `bus_canComponent`: shows what components can be used for transport by bus (`Bus` class). (`CostComponent`, `OperationalParameterComponent`, `TimeComponent`, etc. and their subclasses).
- `bus_canMetrics`: shows what metrics can be considered during a bus transport. Here the range is a subclass of `MetricsComponent`.
- `bus_canTransport`: shows what products can be transported in bus transport. Here, the range can be subclasses of the `Product` class.

4.3.2. The architecture of the ontology system

The ontology model was also integrated into the optimization module, as the model allows rule-based validation for outdoor material handling tasks. Thus, I made the application of the model known to all programmers, because instead of ontology-specific queries (DL Query, SPARQL Query), the user only needs to specify a JSON descriptor, which is known to all programmers, and is an easy-to-read format for people to read.
The system is illustrated in the figure below, where the expert user steps can also be found in publication [65]. The ontology system is created so, that it can be handled by two types of users. One of the users (expert user) who has adequate IT and logistics knowledge. This kind of user performs the following: purpose, scope and requirement identification, then concepts collection, ontology creation and analysis. This user accesses the ontology through the ontology editor, where he can implement it. The other user is the end-user who has adequate logistical knowledge. The user asks questions to the system using a Java program and a JSON descriptor. The query engine layer of the Java program generates SPARQL queries, thereby extracting information from the ontology, and the Java program responds to the user. The ontology layer consists of the RDF / OWL file itself and the reasoner. In *Figure 6.*, I marked each layer in red and each operation in blue.

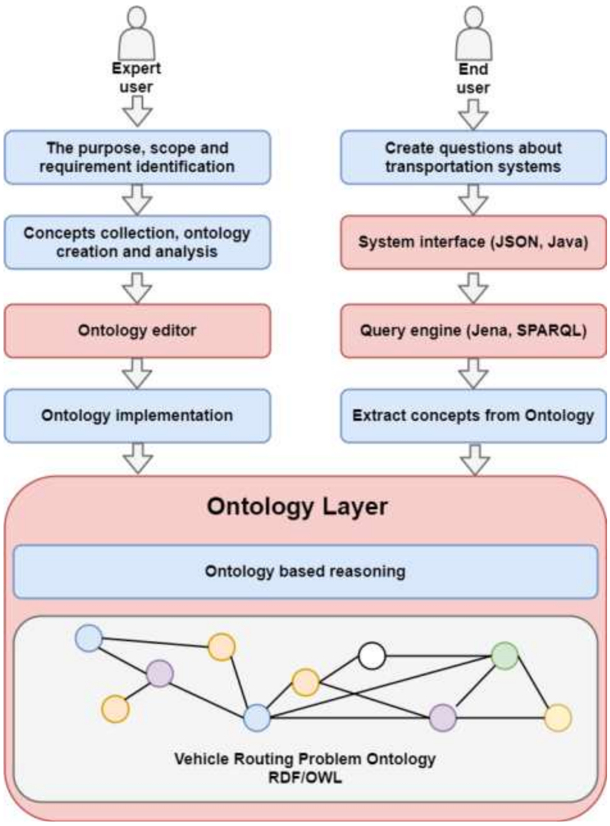

Figure 6.: The architecture of the ontology-based system

The ontology system does not have to specify all the members of the JSON descriptor, it is possible to omit the individual components (Node, Product, Vehicle). It is illustrated in *Figure 7*. Examples of these can be found also in *Appendix 3 [A/3]*.

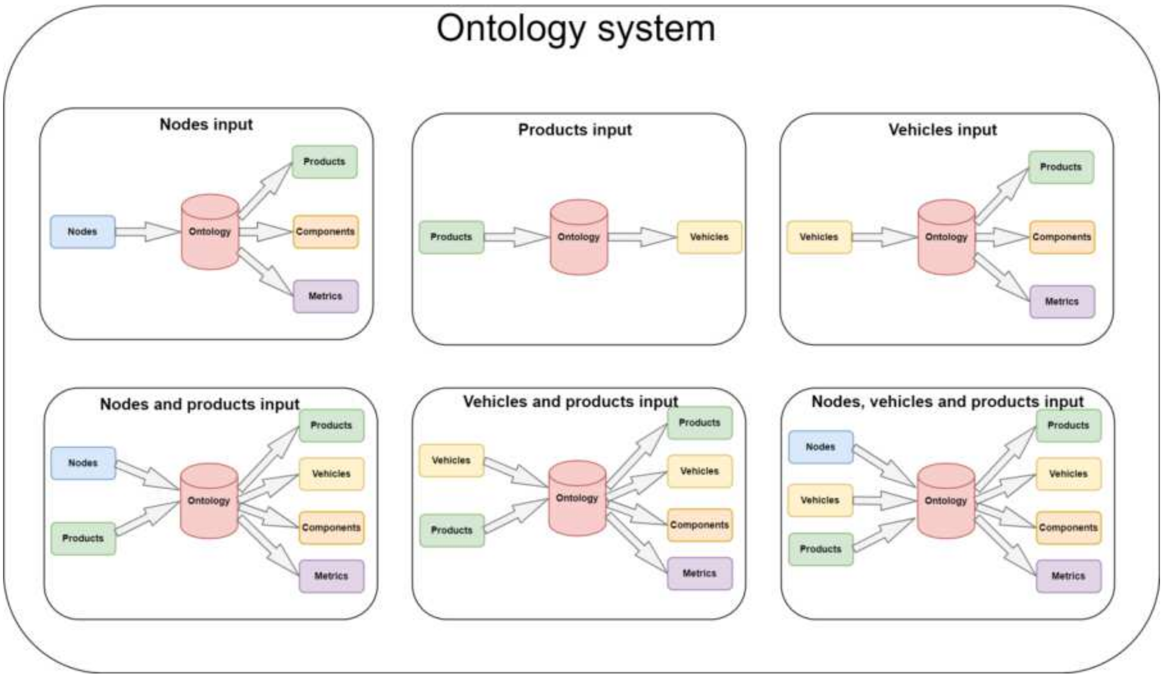

Figure 7.: Ontology system input-output values

The ontology sample system supports transport tasks. If the user knows what types of nodes to visit during the transport task, the system helps to tell which products can be transported to or from the node and what constraint and objective function components can be used during the transport task (*Figure 27. ontology.figure [A/3]*). If the user knows which products the delivery is limited to, the system helps to decide what type of vehicles can be used (*Figure 28. ontology.figure [A/3]*). If the type of vehicles is known during the delivery task, the system tells us in response what products can be delivered with the vehicle, what constraint components and objective function components can be included in the optimization (*Figure 29. ontology.figure [A/3]*). If the system input parameter is the node and the vehicle type, the response is the type of possible products to be transported to and from the node, the vehicles, the possible vehicle routing components, and the possible objective function components (*Figure 30. ontology.figure [A/3]*). If the system receives nodes and products as input, it returns other products, vehicle types, vehicle routing components, and objective function components that can be transported to and from the nodes (*Figure 31. ontology.figure [A/3]*). If a list of vehicles and products are provided, the system returns the products that can be shipped to the node, the products that can be shipped from the node, the list of possible other vehicles, the components, and metrics (*Figure 32. ontology.figure [A/3]*). If a node, a vehicle, and a product are also specified, the system responses with products, vehicles, components, and objective function metrics (*Figure 33. ontology.figure [A/3]*).

In the following, I will generally show how to generate answers to user questions using the reasoning engine of the ontology system. In the description the questions are related to the following elements:
$NODE = \{node_1, \dots, node_n\}$, transporting $PRODUCT = \{product_1, \dots, product_m\}$ with $VEHICLE = \{vehicle_1, \dots, vehicle_k\}$:
I have collected the most typical decision situations and converted into ontological form.

Question 1.:
What other products can be shipped from $NODE = \{node_1, \dots, node_n\}$ with $VEHICLE = \{vehicle_1, \dots, vehicle_k\}$ if $PRODUCT = \{product_1, \dots, product_m\}$ are also transported?

Answer 1:

$$\{canShippedFrom(node_1, x) \vee \dots \vee canShippedFrom(node_n, x)\})$$
$$\wedge\{canShippedTogether(product_1, x) \vee \dots \vee canShippedTogether(product_m, x)\}$$
$$\wedge\{canTransport(vehicle_1, x) \vee \dots \vee canTransport(vehicle_k, x)\}$$

Question 2.:
What other products can be shipped to $NODE = \{node_1, \dots, node_n\}$ with $VEHICLE = \{vehicle_1, \dots, vehicle_k\}$ if $PRODUCT = \{product_1, \dots, product_m\}$ } are also transported?

Answer 2.:

$$\{canDemand(node_1, x) \vee \dots \vee canDemand(node_n, x)\}$$
$$\wedge\{canShippedTogether(product_1, x) \vee \dots \vee canShippedTogether(product_m, x)\}$$
$$\wedge\{canTransport(vehicle_1, x) \vee \dots \vee canTransport(vehicle_k, x)\}$$

Question 3.:
What vehicles can be used to transport $PRODUCT = \{product_1, \dots, product_m\}$ to $NODE = \{node_1, \dots, node_n\}$?

Answer 3.:

$$canShippedWith(product_1, x) \vee \dots \vee canShippedWith(product_m, x)$$

Question 4.:
What Vehicle Routing components can be used when visiting $NODE = \{node_1, \dots, node_n\}$ with $VEHICLE = \{vehicle_1, \dots, vehicle_k\}$ if $PRODUCT = \{product_1, \dots, product_m\}$ are transported?

Answer 4.:

$$\{canComponent(node_1, x) \vee \dots \vee canComponent(node_n, x)\}$$
$$\vee\{canComponent(vehicle_1, x) \vee \dots \vee canComponent(vehicle_k, x)\}$$

Question 5.:
What Vehicle Routing metrics can be used when visiting $NODE = \{node_1, \dots, node_n\}$ with $VEHICLE = \{vehicle_1, \dots, vehicle_k\}$ if $PRODUCT = \{product_1, \dots, product_m\}$ are transported?

Answer 5.:

$$\{canMetrics(node_1, x) \vee \dots \vee canMetrics(node_n, x)\}$$
$$\vee\{canMetrics(vehicle_1, x) \vee \dots \vee canMetrics(vehicle_k, x)\}$$

Questions about visiting a Supermarket node *(Figure 27-28. ontology.figure [A/3]):*

Question 1.:
What products can be shipped from the Supermarket node?

Answer 1.:

$$canShippedFrom(Supermarket, x)$$

Question 2.:
What products can be delivered to the Supermarket node?

Answer 2.:

$$canDemand(Supermarket, x)$$

Question 3.:
What Vehicle Routing components can be used when visiting a Supermarket node?

Answer 3.:

$$canComponent(Supermarket, x)$$

Question 4.:
What Vehicle Routing metrics can be used when visiting a Supermarket node?

Answer 4.:

$$canMetrics(Supermarket, x)$$

Questions about Food delivery (*Figure 29-30. ontology.figure [A/3]*):

Question 1.:
What vehicles can be used to transport Food products?

Answer 1:

$$canShippedWith(Food, x)$$

Question about Train vehicles (*Figure 31-32. ontology.figure [A/3]*):

Question 1.:
What products can be picked up with the Train vehicle?

Answer 1:

$$\forall y\text{: } canShippedFrom(y, x) \land canTransport(Train, x)$$

Question 2.:
What products can be delivered with the Train vehicle?

Answer 2.:

$$\forall y\text{: } canDemand(y, x) \land canTransport(Train, x)$$

Question 3.:
What Vehicle Routing components can be used to transport with a Train?

Answer 3.:

$$canComponent(Train, x)$$

Question 4.:
What Vehicle Routing metrics can be used to transport with a Train?

Answer 4.:

$$canMetrics(Train, x)$$

Questions about visiting a Supermarket node and using Train vehicle (*Figure 33-34. ontology.figure [A/3]*):

Question 1.:
What products can be shipped from a Supermarket node with a Train?

Answer 1.:

$$canShippedFrom(Supermarket, x) \land canTransport(Train, x)$$

Question 2.:
What products can be delivered to a Supermarket node with a Train?

Answer 2.:

$$canDemand(Supermarket, x) \land canTransport(Train, x)$$

Question 3.:
What Vehicle Routing components can be used when visiting a Supermarket node with a Train?

Answer 3.:

$$canComponent(Supermarket, x) \lor canComponent(Train, x)$$

Question 4.:
What Vehicle Routing metrics can be used when visiting a Supermarket node with a Train?

Answer 4.:

$$canMetrics(Supermarket, x) \lor canMetrics(Train, x)$$

Questions about visiting a supermarket node and shipping a Food product (*Figure 35-36. ontology.figure [A/3]*):

Question 1.:
What other products can be shipped from a Supermarket node if Food product is also shipped?

Answer 1.:

$$canShippedFrom(Supermarket, x) \land canShippedTogether(Food, x)$$

Question 2.:
What products can be delivered to a Supermarket node if Food product is also delivered?

Answer 2.:

$$canDemand(Supermarket, x) \land canShippedTogether(Food, x)$$

Question 3.:
What vehicles can be used to deliver Food products to a Supermarket node?

Answer 3.:

$$canShippedWith(Food, x)$$

Question 4.:
What Vehicle Routing components can be used when visiting a Supermarket node if Food product is also transported?

Answer 4.:

$$canComponent(Supermarket, x)$$

Question 5.:
What Vehicle Routing metrics can be used when visiting a Supermarket node if Food product is also transported?

Answer 5.:

$$canMetrics(Supermarket, x)$$

Questions about shipping a Food product with a Train (*Figure 37-38. ontology.figure [A/3]*):

Question 1.:
What other products can be transported with a Train if Food product is also transported?

Answer 1.:

$$\forall y: canShippedFrom(y, x) \land canShippedTogether(Food, x) \land canTransport(Train, x)$$

Question 2.:
What other products can be transported with a Train if Food product is also transported?

Answer 2.:

$$\forall y: canDemand(y, x) \land canShippedTogether(Food, x) \land canTransport(Train, x)$$

Question 3.:
With what other vehicles can Food be transported?

Answer 3.:

$$canShippedWith(Food, x)$$

Question 4.:
What Vehicle Routing components can be used in connection with transporting a Food product with a Train?

Answer 4.:

$$canComponent(Food, x) \vee canComponent(Train, x)$$

Question 5.:
What Vehicle Routing metrics can be used in connection with transporting a Food product with a Train?

Answer 5.:

$$canMetrics(Food, x) \vee canMetrics(Train, x)$$

Questions about visiting a Supermarket node, transporting a Food product and using a Train vehicle (*Figure 39-40. ontology.figure [A/3]*).:

Question 1.:
What other products can be transported from the Supermarket node with Train if Food product is also transported?

Answer 1.:

$$canShippedFrom(Supermarket, x) \wedge canShippedTogether(Food, x) \wedge canTransport(Train, x)$$

Question 2.:
What other products can be transported from the Supermarket node with Train if Food product is also transported?

Answer 2.:

$$canDemand(Supermarket, x) \wedge canShippedTogether(Food, x) \wedge canTransport(Train, x)$$

Question 3.:
What vehicles can be used to deliver Food products to the Supermarket node?

Answer 3.:

$$canShippedWith(Food, x)$$

Question 4.:
What Vehicle Routing components can be used when visiting a Supermarket node with a Train if Food product is also transported?

Answer 4.:

$$canComponent(Supermarket, x) \vee canComponent(Train, x)$$

Question 5.:
What Vehicle Routing metrics can be used when visiting a Supermarket node with a Train if Food product is also transported?

Answer 5.:

$$canMetrics(Supermarket, x) \vee canComponent(Train, x)$$

The advantages of the above queries include the separation of business logic from the source code, the system can be flexibly changed, the ontology model can be easily integrated into other systems.

The presented ontology model was integrated into my Java optimization system. My system thus includes an optimization module and an ontology module. *Figure 8.* shows the relationship between the ontology and the optimization module. The user first uses the ontology module, asking questions to the ontology. The questions are written in JSON format and then answered in a txt format from the system. After that, the optimization module can be used, where the result of the ontology system can be taken into account to create the Vehicle Routing parameters. It then sets the parameters for each optimization algorithm (here too, it specifies a JSON file). Then the system gives the result of the optimization in JSON format.

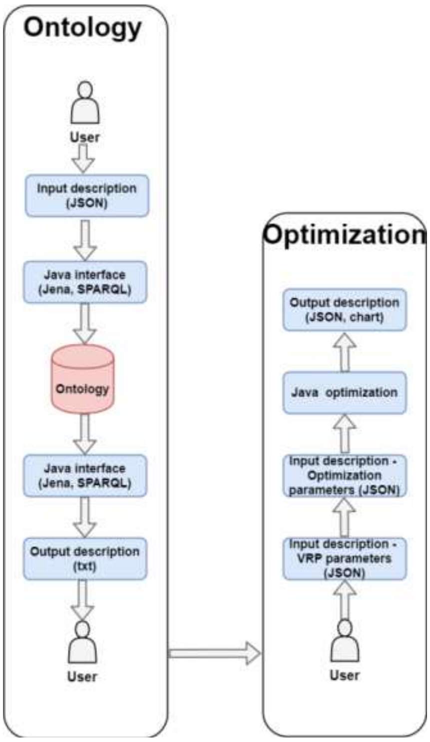

Figure 8.: The relationship between the ontology and optimization module

Thesis 2.:

I have proposed a novel ontology system approach for the generalized Vehicle Routing Problem using the OWL modelling language. The generated ontology system was used for rule-based validation. The system contains the basic components of the general model of the Vehicle Routing Problem, the possible relations between them and the metrics. I extended the basic components of the system with additional subclasses (subcategories of product types, vehicle types and node types). I have created an ontology system, which supports flexibly and effectively the decision in the following topics:

1. Selection of Vehicle Routing components that can be used during transport to/from nodes.
2. Selection of products, which can be transported to the nodes.
3. Selection of Vehicle Routing metrics, which can be used during transport to/from nodes.
4. Selection of products, which can be transported from nodes.
5. Selection of vehicles, which can transport each product.
6. Selection of products, which can be delivered together.
7. Selection of Vehicle Routing components, which can be used when using each vehicle.
8. Selection of Vehicle Routing metrics, which can be used when using each vehicle.
9. Selection of products, which can be delivered by the given vehicles.

Related publications: [P/2], [P/24].

5. Representation model and operators of integrated Vehicle Routing Problem

In this chapter, I present the representation model and evaluation strategy used by the implemented optimization algorithms for solving the general model of the Vehicle Routing Problem. Using the representation, the real problem can be mapped for the optimization algorithm. To solve each type of Vehicle Routing Problem, there are a number of representation modes in the literature. I would like to detail these modes of representation now, some of which have helped me to create my own representation. Then, I present my representation strategy.

Kachitvichyanukul et al. [66] solved the multi-depot vehicle routing problem with multiple pickup and delivery requests. There are more vehicles, more depots, and in addition to the delivery of products from the depots to the customers, the pick up from the customers to the depots is also involved. *Figure 1. authors.representation [A/4]* illustrates the representation mode of the problem. The first part of *Figure 1. authors.representation (a) [A/4]* indicates the order of customers and the second part the vehicles. *Figure 1. authors.representation (b) [A/4]* illustrates which customers are pickup and which are delivery customers. It also sets a priority order. *Figure 1. authors.representation (c) [A/4]* illustrates the vehicle-customer binding. The authors also illustrate which should be the starting and ending station (depot) of the vehicles, as illustrated in *Figure 1. authors.representation (d) [A/4]*. An operating sequence has also been created, as shown in *Figure 1. authors.representation (e) [A/4]*. *Figure 1. authors.representation (f) [A/4]* was then obtained as an evaluation of the representation in *Figure 1. authors.representation (a-e) [A/4].*
The representation of the solution of the Multiple Depot Vehicle Routing Problem (MDVRP) was created by Wang et al. [67]. During the task, several depots and vehicles were provided. The vehicles leave one warehouse at a time and then return to the warehouse from which they left after visiting the customers. *Figure 2. authors.representation [A/4]* illustrates the MDVRP problem and the clash detection and repair. In *Figure 2. authors.representation (a) [A/4]* can be seen the representation of the problem. The first part (depot-vehicle part) indicates how many customers will belong to each depot and vehicle. There are two depots, the depot 1 part belongs to the first and the depot 2 part to the second depot. 2-4-3-3 means that 2 + 4 =6 customers will belong to the first depot. Of these, 2 are customers for the first vehicle and 4 are customers for the second. The second depot will have 3 + 3 =6 customers, of which 3 for the third and 3 for the fourth vehicle. The second part of the representation is a sequence of numbers, the order of the customers. The authors used the classical permutation operators (crossover, mutation) in the customers' part, while their clash detection and repair operators were used in the depot vehicle part.
Wang et al. [68] solved the vehicle-routing problem with soft time windows and multiple depots. During the task, several depots and vehicles were involved. Time window were also assigned to the customers, which is the time interval during which the customer can be visited. *Figure 3. authors.representation [A/4]* illustrates the representation of the task. The authors used a three-part representation. The first part indicates how many customers a vehicle will visit. The second part is the order of the customers. The third part is which vehicle belongs to which depot. The solution for the representation is also illustrated in the *Figure 3. authors.representation [A/4]*. The authors apply the classical permutation operators in the order of the customers, and any complete change in the first and third parts.
Zhou et al. [69] detailed the Capacitated Vehicle Routing Problem. The number of customers and vehicles given during the task. The vehicles start from a single, common depot. The representation is shown in *Figure 4. authors.representation [A/4]*. Each vehicle has a separate part, the example contains 2 vehicles. The number 0 indicates the central depot and the other numbers indicate each customer.

Based on the presented representations and the operators performed on them, I developed my own representation method for my general VRP model and I will also detail, what kind of operators can be applied to them. My representation model of the optimization problem contains the followings:

- the solution description vector,
- vector describing level and period,
- the vector describing the assignment of vehicle - portal node,
- vector describing the order of nodes (depot, satellite, customer),
- matrix describing the assignment of vehicle-node-products,
- vector describing the assignment of the vehicle-charger station.

Examples of the representation and operators of Vehicle Routing Problem optimization algorithms are shown in *Appendix 5 [A/5].*
The components of the general model can be divided into two groups in terms of optimization. One of the groups includes the components, the order and assignment of which give a solution. Changing the order and assignment results in another (possible) solution. This group includes nodes, vehicles, products, charger stations, levels, periods. The other group includes components that cannot be associated with each other. These system constraints (e.g. time window) are either parameters describing the functional operation of the system (delivery of products, collection) or can even serve as metrics (distance between nodes, safety factor, cost of loading and unloading products, etc.)
In the following, I will present the developed representation model and then the operators.

5.1. Representation model elements

5.1.1. Vector describing the order of nodes

A permutation is used to describe the order of nodes in the transportation. In the list the nodes are denoted by position index. The length of the permutation is equal to the number of nodes. The vector of the nodes of level i. is given by:

$$\bar{P}_i = [p_1^i, p_2^i, \dots, p_{nol_i}^i], \tag{85}$$

where nol_i denotes the number of nodes in the level i. and $p_1^i, p_2^i, \dots, p_{nol_i}^i \in POS_i,\ p_1^i, p_2^i, \dots, p_{nol_i}^i \neq rechargerstation$. An example of a vector describing the order of nodes is shown in *Figure 1. representation.figure [A/5].* This vector does not fully represent a visit sequence, together with a vector describing the assignment of vehicle-nodes-products, it represents a service sequence for a vehicle.

5.1.2. The matrix describing the assignment of vehicles-nodes-products

Unlike the previous permutation vector, it is an assignment. This means that the individual numbers here indicate the individual vehicles in the line, only here the numbers appear more than once. The length of the assignment is equal to the product of the number of nodes and the number of types of products in the system. This specifies which vehicle will deliver each item at each node:

$$V_i = \begin{bmatrix} v_{1,1}^i & \cdots & v_{1,n_{producttype}}^i \\ \vdots & \cdots & \vdots \\ v_{nol_i,1}^i & \cdots & v_{nol_i,n_{producttype}}^i \end{bmatrix}. \tag{86}$$

In the equation V_i is the matrix describing the association of vehicles-nodes-products at level i. and v_{jm}^i denotes the service in level i. for node j. and for product m. For $v_{jm}^i \in \{1, \dots, cntvehicle_i\}$, $cntvehicle_i = \sum_k cntvehicle_i^k$.
An example of a vector describing the assignment of vehicles-nodes-products is shown in *Figure 2. representation.figure [A/5].*

5.1.3. Vector describing the assignment of vehicle - charger stations

Here it is necessary to determine to which charger station vehicle will belong, so a charger station-vehicle assignment must be introduced. Its length is equal to the number of vehicles:

$$\bar{R}_i = \left[r_1^i, r_2^i, \dots, r_{cntvehicle_i}^i\right]. \tag{87}$$

In the equation $\bar{R}_i$ is a vector, describing the association of vehicle – charger stations in level i., and $r_1^i, r_2^i, \dots, r_{cntvehicle_i}^i \in POS_i,\ r_1^i = chargerstation, \dots, r_1^i = chargerstation.$
An example of a vector describing the association of vehicle - charger stations is shown in *Figure 3 .representation.figure [A/5].*

5.1.4. Vector describing the vehicle - portal node assignment

The length of the assignment vector is equal to the number of vehicles at the current level since a start node and end node is assigned to the vehicles:

$$\overline{UL}_i = \left[ul_1^i, ul_2^i, \dots, ul_{cntvehicle_i}^i\right]. \tag{88}$$

In the equation $\overline{UL}_i$, describes the vehicle portal node assignment at the level i. $ul_1^i, ul_2^i, \dots, ul_{cntvehicle_i}^i \in POS_{i-1}$.
An example of a vector describing a vehicle - portal node assignment is shown in *Figure 4. representation.figure [A/5].*

5.1.5. Vector describing level

The Vehicle Routing Problem has a level structure. Vehicles start from one node in each of the above levels (and deliver/collect the products) at the lower levels and then return to the aggregated level (if no open route component is used):

$$\bar{L}_i = [\bar{P}_i, \bar{V}_i, \bar{R}_i, \overline{UL}_{i-1}]. \tag{89}$$

The vector L contains vectors describing the order of the nodes, matrices describing the assignment of the vehicle-nodes-products, vectors describing the assignment of the vehicle-charger stations, and vectors describing the assignment of the vehicle - portal node.
An example of a matrix describing a level is shown in *Figure 5. representation.figure [A/5].*

5.1.6. Vector describing period

I designed the representation mode so that the descriptive vectors can handle the period as well:

$$\bar{T} = \left[\bar{L}_1, \bar{L}_2, \dots, \bar{L}_{n_{level}}\right]. \tag{90}$$

In the equation $\bar{T}$ describes the period, and contains matrices describing the levels.
An example of a vector describing the period is shown in *Figure 5. representation.figure [A/5].*

5.1.7. Solution description vector

The solution description vector contains the period description vectors:

$$\bar{S} = \left[\bar{T}_1, \bar{T}_2, \dots, \bar{T}_{n_{period}}\right]. \tag{91}$$

In the equation $\bar{S}$ is the vector, which describes the solution. It contains the vectors describing the period.
An example of a vector describing the solution is shown in *Figure 6. representation.figure [A/5].*

5.2. Optimization operators

During optimization, operators create one or more new solutions from one or more existing solutions. The operators used greatly influence the efficiency of the optimization algorithms, which I will also present in the fitness landscape analyzes.

5.2.1. Vector describing the order of nodes

Here, I use a permutation representation mode, the sequence numbers of the nodes are included in the representation mode. Here I used a neighbourhood operator and crossover operators.

<u>Neighbourhood operator (2-opt)</u>

This practically means exchanging two edges. Two elements of the permutation are randomly selected and the elements of the section are "reversed" between them. The procedure is generalized to k-opt, then k edges are replaced [70]:

$$\bar{P}_i^{c1} = O_{2-opt}(\bar{P}_i^{p1}), \tag{92}$$

where $O_{2-opt} = [o_1, o_2]$ denotes the two selected positions, $\bar{P}_i^{p1}$ is the vector before 2-opt, $\bar{P}_i^{c1}$ is the vector after 2-opt.
It is a unary operator, and it has unit element: if $o_1 = o_2$ then for $\forall O_{2-opt}$ $\bar{P}_i^{p1} = \bar{P}_i^{c1}$. The number of neighbours of the operator: $|\bar{P}_i| \cdot |\bar{P}_i|$ where $|\bar{P}_i|$ denotes the length of $\bar{P}_i$.
An example of a neighbourhood operator (2-opt) is shown in *Figure 7. representation.figure [A/5].*

<u>Partially Matched Crossover (PMX)</u>

Two positions in the permutation are selected. The section between the two positions is called the fitting section. The genes are paired. And the pairs are exchanged [71]:

$$(\bar{P}_i^{c1}, \bar{P}_i^{c2}) = O_{PMX}(\bar{P}_i^{p1}, \bar{P}_i^{p2}), \tag{93}$$

where $O_{PMX} = [o_1, o_2]$ denotes the two selected positions, $(\bar{P}_i^{p1}, \bar{P}_i^{p2})$ the parent pairs –vectors-, $(\bar{P}_i^{c1}, \bar{P}_i^{c2})$ denotes child pairs - vectors.
It is a binary operator, and it has unit element: if $o_1 = o_2$ then $\forall O_{PMX}$ $\bar{P}_i^{p1} = \bar{P}_i^{p2} = \bar{P}_i^{c1} = \bar{P}_i^{c2}$. If $\bar{P}_i^{p1} = \bar{P}_i^{p2}$ then $\forall O_{PMX}$ $\bar{P}_i^{p1} = \bar{P}_i^{p2} = \bar{P}_i^{c1} = \bar{P}_i^{c2}$. The number of neighbours of the operator: $|\bar{P}_i| \cdot |\bar{P}_i|$ where $|\bar{P}_i|$ denotes the length of $\bar{P}_i$. The operator is a commutative operator, because $O_{PMX}(\bar{P}_i^{p1}, \bar{P}_i^{p2}) = O_{PMX}(\bar{P}_i^{p2}, \bar{P}_i^{p1})$.
An example of a partially matched crossover is shown in *Figure 8. representation.figure [A/5].*

<u>Order Crossover (OX)</u>

Also this operator uses fitting section. This operator works with parent and child solutions. A copy of the parents is made, which will be the children. Elements of the fitting section in the second individual (child) must be deleted from the first individual (child) and vice versa. This creates holes. The holes should fall into the fitting section in both children. Once the movement is completed, the elements in the fitting section of the first parent will be elements of the fitting section of the second child, while the elements in the fitting section of the second parent will be elements of the fitting section of the first child [71]:

$$(\bar{P}_i^{c1}, \bar{P}_i^{c2}) = O_{OX}(\bar{P}_i^{p1}, \bar{P}_i^{p2}), \tag{94}$$

where $O_{OX} = [o_1, o_2]$ denotes the two selected positions, $(\bar{P}_i^{p1}, \bar{P}_i^{p2})$ the parent pairs - vectors -, $(\bar{P}_i^{c1}, \bar{P}_i^{c2})$ denotes child pairs - vectors.
It is a binary operator, and it has got a unit element: if $o_1 = o_2$, then $\forall O_{OX}$ $\bar{P}_i^{p1} = \bar{P}_i^{p2} = \bar{P}_i^{c1} = \bar{P}_i^{c2}$. If $\bar{P}_i^{p1} = \bar{P}_i^{p2}$ then $\forall O_{OX}$ $\bar{P}_i^{p1} = \bar{P}_i^{p2} = \bar{P}_i^{c1} = \bar{P}_i^{c2}$. The number of neighbours of the operator: $|\bar{P}_i| \cdot |\bar{P}_i|$ where $|\bar{P}_i|$ denotes the length of $\bar{P}_i$. The operator is commutative, because $O_{OX}(\bar{P}_i^{p1}, \bar{P}_i^{p2}) = O_{OX}(\bar{P}_i^{p2}, \bar{P}_i^{p1})$.
An example of order crossover is shown in *Figure 9. representation.figure [A/5].*

<u>Cycle crossover – CX</u>

This procedure starts from the fact that the child individuals receive their items from either the first or the second parent.
Let us first look at the first element of the first parent and the first element of the second parent. The child individuals are initially empty, then the first element of the first child should be the first element of the first parent, and the first element of the second child should be the first element of the second parent.
The first element of the second child must also be included by the first child, so we need to look at where this element is found in the first parent. Thus, a pair must be formed again, where the element of the first parent is received by the first child (to the position corresponding to the position of the parent) and the element of the second parent is received by the second child (according to the position of the parent). This procedure is continued until there is a step. Then, the empty elements of the first child are filled with the elements of the second parent (taking care of the positions) and the empty elements of the second child with the elements of the first parent [71]:

$$(\bar{P}_i^{c1}, \bar{P}_i^{c2}) = O_{CX}(\bar{P}_i^{p1}, \bar{P}_i^{p2}). \tag{95}$$

O_{CX} is not a position-based operator. $(\bar{P}_i^{p1}, \bar{P}_i^{p2})$ denotes the parent pairs –vectors-, and $(\bar{P}_i^{c1}, \bar{P}_i^{c2})$ denotes the child pairs –vectors.
It is binary operator: if $\bar{P}_i^{p1} = \bar{P}_i^{p2}$ then $\forall O_{CX}$ $\bar{P}_i^{p1} = \bar{P}_i^{p2} = \bar{P}_i^{c1} = \bar{P}_i^{c2}$.
Commutative because $O_{CX}(\bar{P}_i^{p1}, \bar{P}_i^{p2}) = O_{CX}(\bar{P}_i^{p2}, \bar{P}_i^{p1})$.
An example of a cycle crossover is shown in *Figure 10. representation.figure [A/5].*

5.2.2. The matrix describing the assignment of vehicles-nodes-products

A "regeneration" is used here, so the neighbour will be a randomly generated new matrix:

$$V_i^{c1} = O_{VNP}(V_i^{p1}), \tag{96}$$

where V_i^{p1} denotes the initial matrix and V_i^{c1} denotes the matrix after the operation.
It is a unary operator. There exists such that $V_i^{c1} = V_i^{p1} = O_{VNP}(V_i^{p1})$. The number of neighbours is $cntvehicle_i^{|V_i^{p1}|}$.
An example of regenerating a matrix describing vehicle-node-product assignment is shown in *Figure 11. representation.figure [A/5].*

5.2.3. Vector describing the assignment of vehicle - charger stations

A "regeneration" is used here, so the neighbour will be a randomly generated new vector:

$$\bar{R_i}^{c1} = O_{VR}\left(\bar{R_i}^{p1}\right), \tag{97}$$

where $\bar{R_i}^{p1}$ is the initial vector and $\bar{R_i}^{c1}$ is the vector after the operation.
It is a unary operation. There exists such that $\bar{R_i}^{c1} = \bar{R_i}^{p1} = O_{VR}(\bar{R_i}^{p1})$. The number of neighbours: $n_{chargerstation}{}^{|\bar{R_i}^{p1}|}$ where $n_{chargerstation}$ is the number of charger stations.
An example of the regeneration of a vector describing the assignment of vehicle-charger stations is shown in *Figure 12. representation.figure [A/5].*

5.2.4. Vector describing the vehicle - portal node assignment

A "regeneration" is used here, so the neighbour will be a randomly generated new assignment:

$$\overline{UL_i}^{c1} = O_{VR}\left(\overline{UL_i}^{p1}\right), \tag{98}$$

where $\overline{UL_i}^{p1}$ is the initial vector and $\overline{UL_i}^{c1}$ is the vector after the operation.
It is unary operator. There exists such that $\overline{UL_i}^{c1} = \bar{R_i}^{p1} = O_{VR}(\overline{UL_i}^{p1})$. The number of neighbours are: $|POS_i|^{|\overline{UL_i}^{p1}|}$ where $n_{chargerstation}$ is the number of charger stations.
An example of a regeneration vector describing a vehicle - portal node assignment is shown in *Figure 13. representation.figure [A/5].*

A detailed evaluation of the representation mode for each component of the Vehicle Routing Problem is shown in the tables in *Appendix 1 [A/1]* (*Table 1-2. vrp.table [A/1]*).

Thesis 3.:
I created a novel representation of the general model of the Vehicle Routing Problem. The representation model includes the following main components:

1. ***Vector describing the order of nodes (depot, satellite, customer).***
2. ***The matrix describing the assignment of vehicles-nodes-products.***
3. ***Vector describing the assignment of vehicle - charger stations.***
4. ***Vector describing the vehicle - portal node assignment.***
5. ***Vector describing the level.***
6. ***Vector describing the period.***

I introduced and analyzed the search operators of heuristic algorithms.
Operators applied to the vector describing the order of the nodes (based on the literature) are the followings:

1. ***2-opt,***
2. ***Partially Matched Crossover – PMX,***
3. ***Order Crossover – OX,***
4. ***Cycle crossover – CX.***

Operator applied to the matrix describing the assignment of vehicles-nodes-products, vector describing the assignment of vehicle - charger stations and the vehicle - portal node assignment vector:

1. ***Regeneration operator.***

I also created an evaluation of the representation of the general model of the Vehicle Routing Problem.
Related publications: [P/1], [P/5], [P/6], [P/8], [P/10], [P/15], [P/16], [P/17], [P/21], [P/22], [P/23].

6. Fitness landscape analysis

In this chapter, I present the relationship between the representation form and optimization efficiency in search space. Relatively few researchers have addressed this topic in fitness landscape analysis over the years. This is probably because researchers typically focus solely on investigating the surface effectiveness, but they do not analyze the internal optimization processes. Another important reason of topic selection is that in most cases the (local) optimum value is searched by an ad hoc method, do not analyze the quality of the located optimum. In the journal publications, the comparisons are made only practical test. Unfortunately, most researchers do not investigate nor demonstrate why the developed method (algorithm tuning, good choice of parameters, implementation tricks, etc.) is better than the existing methods. This technique is also called "competitive testing". In contrast, behavioural models demonstrate what makes a particular local search metaheuristic more effective than the others. Extensive behavioural models allow us to improve the efficiency of existing local search algorithms. [77]

6.1. The literature of fitness landscape analysis

The fitness landscape analysis for optimization tasks first appeared in Sewall Wright's publication. Here, he described the search space as a set of solutions and fitness functions [72].
In the survey part I performed analysis in 3 different aspects. The first comparison is based on the landscape model. The second approach investigates the applied methods on surface analysis. The third aspect is the application.

Table 2. summarizes the different data models, concepts and measurements related to the landscape analysis.

Landscape model	Publication	Landscape model	Publication
local and global optimum	[73,77,74,78,88,82]	fitness value	[87]
walk	[73,76,74,78]	connectivity	[73]
neighbourhood	[76,77,79]	navigation strategy	[77]
fitness function	[73,74,87]	plateau	[73]
evolvability	[73,89,83]	neutral area	[73]
search space	[73,74]	isotropic landscape	[90]
encoding and representation of the solution	[73,87]	rugged landscape	[90]
neutrality	[76,89]	ruggednes	[88]
basin	[73,88]	local search	[82]
barrier	[73,74]	neutral neighbor	[83]
local optimum	[83,87]	diameter of the fitness landscape	[91]

Table 2.: Landscape models

Regarding the applied methods three main groups can be distinguished: fitness landscape methods, optimization algorithms, and neighborhood operators. *Table 3.* illustrates the results.

Methods	Publication	Methods	Publication
Fitness landscape methods			
auto-correlation	[73,75,93,78,90,79, 94,80,81,83,87,84,85,86]	ruggedness	[74,87]
fitness distance correlation	[89,78,90,79,96,97] [91,98,81,87,84,99,86]	time to local optimum	[94,100]
correation length	[73,89,74,80,91,92,87,85]	number of local optima	[94,100]
random walk	[73,76,74,101,92,87,99]	distance of optima	[94,100]
basin of attraction	[76,77,74,82,92,87]	reverse adaptive walk	[74,87]
fitness cloud	[73,74,78,102,98,87]	information stability	[73,74]
information analysis	[73,76,74,87,85]	bordering fitness	[102,87]
adaptive walk	[73,74,88,87]	time series methods	[80,87]
neutral walk	[73,74,81,87]	probability of visiting an optimum	[100]
partial information content	[73,74,87,85]	cost density	[101]
information content	[73,76,74,87]	neutral degree	[83]
density basin information	[73,76,74,87]	partial information content	[76]
auto-correlation coefficient	[73,95,85]	regularity	[73]
uphill-downhill walk	[73,74,87]	evolvability portrait	[73]
fitness distance correlation	[73,74,103]	negative slope coefficient	[73]
Optimization algorithms			
genetic algorithm	[76,104,95,92,106]	tabu search	[104,78]
memetic algorithm	[90,79,107,91]	evolution algoritm	[98]
simulated annealing	[105,77,92]	differential evolution algorithm	[89]
hill climbing algorithm	[104,78]		
Neighbourhood operators			
swap	[76,96,92]	displacement	[76]
inversion	[76,96]	cycle crossover	[76]
partially matched crossover	[76,92]	edge recombination	[95]
insertion	[76]	maximal preservative crossover	[95]

Table 3.: Applied methods on surface analysis

Table 4. presents the main application areas where the landscape analysis can significantly improve the efficiency.

Application	Publication	Application	Publication
Traveling Salesman Problem	[74,104,90,94, 96,95,100]	Maximal Constraint Satisfaction Problem	[101]
Vehicle Routing Problem	[76,105]	Quadratic Assignment Problem	[92]
Job Shop Scheduling Problem	[75]	Flowshop Scheduling Problem	[83]
Quadratic Assignment Problem	[74]	Optimum Multiuser Detection Problem	[99]
No-Wait Job Scheduling	[94]	Knapsack Problem	[86]
Multiuser Detection Problem	[97]		

Table 4.: Applications in connection with fitness landscape

6.2. Fitness landscape overview

The subject of the fitness landscape analysis is the examination of configuration optimization algorithms, in the search space containing all the allowed configurations.

Optimization metaheuristics are often based on some kind of search or navigation iterations. The body of the iteration cycle is based on the following elements [73,108]:

- Algorithm independent elements:
  - The set of possible states denoted by S. The search space can be discrete or continuous.
  - The distance-based neighbourhood, defined by the following operator $N : x \to P(x)$. The operator is applied to the current status point to generate the candidates for the next state point. An example for the neighbourhood operator is the edge swap (2-opt) operator.
  - Fitness, objective function denoted by $f : S \to R$. This gives the fitness value for each possible state point (solution). Usually, fitness is given by a real number. For most optimization tasks, a single fitness value is used. However, in multi-object optimization, the fitness value may be a vector.
  - Encoding and representation: Although encoding and representation are not formally part of the fitness landscape, they are an important factor. This is because representation is part of the evaluation of fitness value, and mutation operators depend on representation.
- Algorithm specific elements:
  - Transition rule (pivoting rule = selection strategy) that selects the next state point from potential neighbouring state points.
  - Termination condition, which determines when the algorithm stops.
  - An initial state point is either a randomly generated solution (state point) or a solution given by some construction heuristic (state point).

Based on the previous introduced symbols, the formal description of the search space [73]:

$$\mathcal{F} \coloneqq (S, f, N). \tag{99}$$

<u>Local and global optimum:</u>

One of the most important factors in the search space analysis is the detection of the local and global optima.
The global optimum is given by in case of maximization problem [73]:

$$O(\mathcal{F}) \coloneqq \{x \mid x \in S, \forall (y \in S)\, f(x) \geq f(y)\}. \tag{100}$$

The global optimum is given by in case of minimization problem [73]:

$$O(\mathcal{F}) \coloneqq \{x \mid x \in S, \forall (y \in S)\, f(x) \leq f(y)\}. \tag{101}$$

(The maximization task can be easily converted to a minimization task, we only need to take -1 times the fitness function).
In the case of a neighbourhood-based search space, the local optimum is a solution where none of its neighbours is smaller (larger) than itself.
In the case of maximization task [73]:

$$\widehat{O_N}(\mathcal{F}) \coloneqq \{x \mid x \in S, \forall (y \in N(x))\, f(x) \geq f(y)\}. \tag{102}$$

In the case of minimization task [73]:

$$\widehat{O_N}(\mathcal{F}) \coloneqq \{x \mid x \in S, \forall (y \in N(x))\, f(x) \leq f(y)\}. \tag{103}$$

The local optimum can be also defined in recombination spaces.

In the case of maximization task [73]:

$$\widehat{\mathcal{O}_P}(\mathcal{F}) \coloneqq \{x \mid x \in S, \forall (y \in S, z \in \mathcal{R}(x,y))\ f(x) \geq f(z)\},$$
$$\mathcal{R}: S \times S \rightarrow P(S). \tag{104}$$

In the case of minimization task [73]:

$$\widehat{\mathcal{O}_P}(\mathcal{F}) \coloneqq \{x \mid x \in S, \forall (y \in S, z \in \mathcal{R}(x,y))\ f(x) \leq f(z)\},$$
$$\mathcal{R}: S \times S \rightarrow P(S). \tag{105}$$

The following questions/tasks are known in the analysis of local and global optima [73,100]:

- The number of local optima: If the problem is unimodal, there is only one local optimum. This kind of problem is easier to solve than problems that have multiple local optima.

<u>Fitness landscape analysis:</u>

Optimization algorithms search for a good search space state in a relatively low (minimal) time, while fitness landscape analysis provides a „good" insight into the problem in a minimal amount of time. Insight means "good" (acceptable) and "optimal" solutions. The fitness landscape analysis can be a good technique to develop new optimization algorithms. [73]

Fitness landscape analysis can be used to compare the efficiency of the different search space representation modes. It helps to make the following conclusions [109]:

- Comparison of differences between two search spaces: a problem with two or more different representation methods: different representation, different mutation operator, different objective function, etc.
- Algorithm selection: analysis of the navigation differences of the search space.
- Tuning the parameters: determination of the most appropriate parameters for the selected algorithm (for example the neighbourhood operators, size of populations, number of restarts).
- Controlling parameters during runs: what is the optimal neighbourhood operator based on structure estimation.

<u>Plateaus and neutral areas:</u>

Before defining the plateau and the neutral area, I will define the definition of connectivity, i.e. the connected subset [73]:

- A single-element set with a single solution x is a connected set, denoted $connected(\{x\})$.
- The union of the connected set S and a solution candidate x (which is the neighbour of the connected set S) is also a connected set [73]:

$$connected(\{x\} \cup S) :\Leftrightarrow \exists (y \in S')\ N(x,y) \wedge connected(S'), S' \subseteq S. \tag{106}$$

From the definition of the connected set, it is now easy to define the neutral area and the plateau. The neutral area is defined by the following formula [73]:

$$neutral(S') :\Leftrightarrow connected(S') \wedge \forall (x,y \in S')\ \ f(x) = f(y). \tag{107}$$

A neutral area is a connected subset of the search space where the fitness value of the elements is the same. To define the plateau, it is also necessary to define the outer border of the connected set. These are the points that delimit the connected set [73]:

$$border(S') \coloneqq \{x \mid x \in S, x \notin S', \exists (y \in S'), x \in N(y)\}. \tag{108}$$

The definition of a plateau can be given as follows [73]:

- for maximization problem: $plateau(S) :\Leftrightarrow neutral(S) \wedge \forall(x \in border(S))\, f(x) < f(S)$,
- for minimization problem: $plateau(S) :\Leftrightarrow neutral(S) \wedge \forall(x \in border(S))\, f(x) > f(S)$.

In the equations, $f(S)$ is a fitness value in the neutral area.
A plateau consists of a neutral area and a boundary where there is no higher (in the case of a maximization task) or less (in the case of a minimization task) fitness value at the boundary of the connected set. *Figure 9.* shows the neutral area and border for a plateau. In the figure, I marked the neutral area of the plateau in green and the border in red. The numbers show the serial number of the solutions. The x-axis indicates the number of iterations (iteration of creating neighbouring solutions) and the y-axis the fitness value.

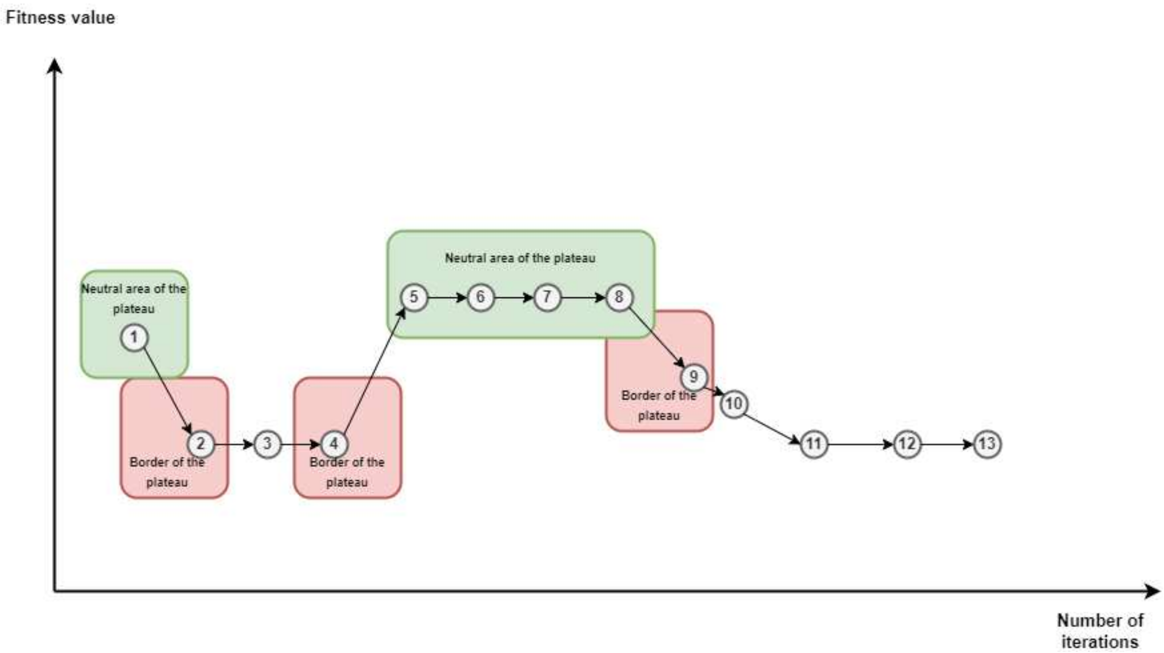

Figure 9.: Plateau [110]

Neutral areas can easily hinder optimization strategies. Due to the lack of slope, optimization methods often stick to a large neutral area. But a carefully selected optimization scheme can easily exploit neutral areas so that the algorithm finds other solution candidates (outside the neutral area). [73]

6.2.1. Measures

Search space size

A key measure is the size of the search space. For a finite, countable search space e.g. combinatorial optimization problems, the size of the search space is one of the most important metrics to estimate the size of the computation cost. The size of the search space can be given by the number of dimensions, the number of possible state points, or the diameter of the search space. [73]
The diameter of the search space is calculated with the following equation [73]:

$$diam(S) := \max\{d(x,y) \mid x, y \in S\}, \tag{109}$$

where $d: S \times S \rightarrow \mathbb{R}$ is the distance between two individuals.

Autocorrelation

Autocorrelation function

Ruggedness is based on the correlation of adjacent solution vectors. Therefore, the autocorrelation of the fitness function is one measure of search space ruggedness. If the fitness landscape has a relatively large number of local optima, it is rugged. If there are few local optima, it is smooth or flat. *Figure 10* shows the rugged search space and *Figure 11* shows the smooth search space. [73]
In a rugged search landscape it is a large variance of adjacent fitness values, and it is difficult to search in such a space. [110]

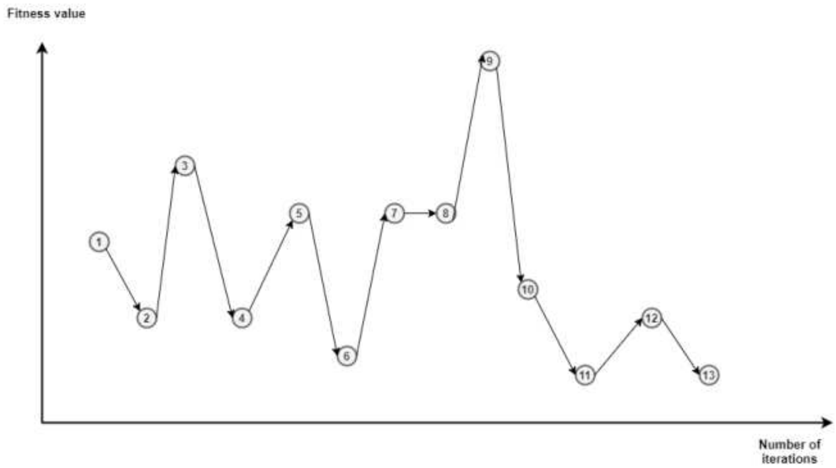

Figure 10.: Rugged landscape

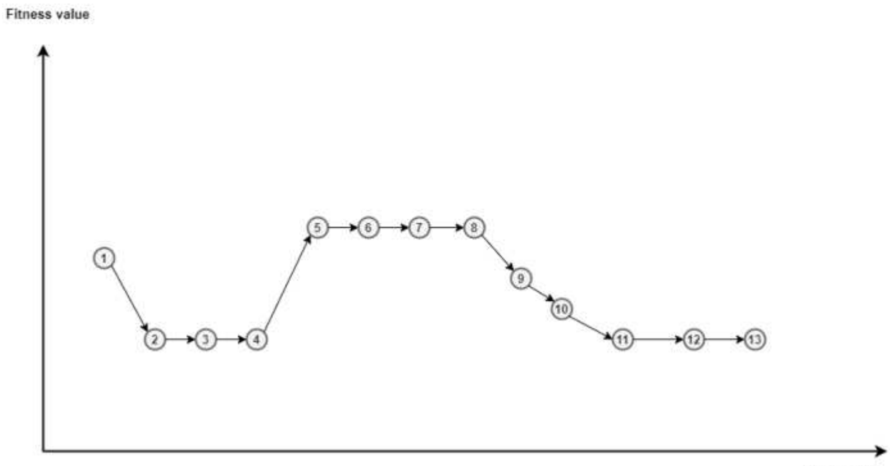

Figure 11.: Smooth landscape

The following formula is used to calculate the autocorrelation, where τ means shift [73]:

$$\rho(\tau) := \frac{E[f_i \cdot f_{i+\tau}] - E[f_i]E[f_{i+\tau}]}{Var[f_i]}. \tag{110}$$

In the *Equation 110.*, $E[x]$ and $Var[x]$ are the expected values and standard deviations of $\{f(x_i)\}_{i=0}^{n}$. The $\{f(x_i)\}_{i=0}^{n}$ is abbreviated with f_i. [73]

Autocorrelation coefficient

This measure gives the average correlation between solution candidates for the entire search space. This measure is denoted with $\rho(1)$. The actual autocorrelation coefficient can be calculated by the following formula [73]:

$$\lambda := \frac{1}{1-\rho(1)}. \tag{111}$$

If λ is small, than the landscape is rugged, otherwise the landscape is smooth.

Fitness cloud

A fitness cloud is a figure that contains the fitness value of the "base" state point and the fitness values of the neighbouring states of the "base" state point (bordering fitness). The fitness cloud consists of each fitness pair (f_i, g_i), where f_i is the "base" state point fitness value, while g_i is the "adjacent" (bordering) state fitness value. [73,74]
For each search state point (solution candidate) it means one point in the coordinate system, where the x-axis represents the fitness value of the "base" state and the y-axis represents the fitness value of the neighbour state of the "base" state point. Formally, $FC = \{(f(x), \tilde{f}(x)) | x \in S\}$, where the fitness value of the "base" state is denoted by f, while the fitness value of the "adjacent" (bordering) state is denoted by $\tilde{f}$. The $FC_{min}, FC_{mean}, FC_{max}$ can be introduced as follows [102]:

$$FC_{min} := \{(\phi, \tilde{\phi}) | \phi \in f(S), \tilde{\phi} = \min \tilde{f}(x)\}, \tag{112}$$

$$FC_{mean} := \{(\phi, \tilde{\phi}) | \phi \in f(S), \tilde{\phi} = \operatorname{mean} \tilde{f}(x)\}, \tag{113}$$

$$FC_{max} := \{(\phi, \tilde{\phi}) | \phi \in f(S), \tilde{\phi} = \max \tilde{f}(x)\}. \tag{114}$$

Let φ be the fitness value of the "base" state point, α be the associated FC_{min} value, β be the associated FC_{mean} value and γ be the associated FC_{max} value.
Using the previously introduced variables, the following 4 cases can be distinguished for the maximization task [73]:

- $\varphi < \alpha$: If the fitness value of the "base" state point is below α, then the bordering fitness values are better, so it is called as strictly advantageous.
- $\alpha < \varphi \leq \beta$: If the fitness value of the "base" state point is between α and β, it is called average advantageous because the bordering fitness values are higher than theirs.
- $\beta < \varphi \leq \gamma$: The "base" fitness point value between β and γ is called deleterious because their average bordering fitness value is lower than the "base" fitness point value.
- $\gamma < \varphi$: A "base" fitness value above γ is called strictly deleterious because their bordering fitness is always lower than themselves.

Negative slope coefficient

The slope defined by successive point can be calculated with the following formula [73]:

$$S_i = \frac{g_{i+1} - g_i}{f_{i+1} - f_i}. \tag{115}$$

The negative slope coefficient can be defined as the sum of all negative slopes [73]:

$$nsc = \sum_i \min(S_i, 0). \tag{116}$$

The negative slope coefficient gives a wide picture of the search space where misleading structures can be identified. When the optimization process has to go down, it usually achieves a higher fitness value (or vice versa in a minimization task), which hinders the optimization process because it can get stuck. This idea is well reflected in the negative slope coefficient, where all downward slopes are summed to represent the severity of the problem. [73]

Fitness Distance Correlation

The goal of the optimization is usually to find the global optimum, where an exhaustive search need to find the global optimum. The basic idea of fitness distance correlation is to analyze the correlations of fitness distances between the global optimum and any other points. In the case of high fitness distance correlation, higher fitness values are closer to the global optimum than lower fitness values. In the case of a local search, it can be useful to know this value, because if the search space (landscape) has a high correlation, the local search quickly navigates to solution with better approximation of the global optimum. [73,111]
The following equation gives the fitness distance correlation value, where $E[x]$ and $Var[x]$ are the expected value and the standard deviation [73,111]:

$$FDC(\mathcal{F}) := \frac{E[f_i \cdot d_i] - E[f_i] \cdot E[d_i]}{Var[f_i] \cdot Var[d_i]}. \tag{117}$$

- $FDC(\mathcal{F}) \geq 0.15$ misleading landscape: fitness increases with distance.
- $-0.15 \leq FDC(\mathcal{F}) \leq 0.15$ difficult landscape: no correlation between distance and fitness value.
- $FDC(\mathcal{F}) \leq -0.15$ straightforward landscape: fitness value increases as we get closer to the global optimum.

6.2.2. Methods of analysis

Analytical methods for investigation of the search space can be divided into two categories, exhaustive search and stochastic, sampling-based techniques [73].

Exhaustive search

This analysis gives the most complete picture of the search space, but they can only be applied to smaller problems (no longer for larger problems) and it is difficult to apply in practice. The main advantage of an exhaustive search is completeness. Search time (runtime) is high, but search space analysis is complete. [73]

Stochastic, sampling-based techniques

Stochastic methods are the most commonly used in analyzes. Their advantage over exhaustive search is their speed. On the other hand the disadvantage is that they are biased, we do not get complete results with their use. It is comparable to metaheuristics in the sense that metaheuristics search for a “relatively good” solution, while stochastic search offers a “relatively good” mapping. [73]
The basic problem of stochastic analysis methods is the selection of a representative sample. This task is difficult if we do not know the complete set of solutions. So far, two types of sample generation techniques have been used [73]:

- The trajectory-based sampling, which is the path of optimization methods and produces continuous solution candidates.
- The discovery based sampling strategies which generates scattered samples that can be supplemented with recombination operators or local neighbourhood operators.

Trajectory-based sampling

Sampled trajectories create a path in the search space or, in other words, a sequence of adjacent state points (solution candidates). The method is also called a walk. Formally, the sequence $\{x\}_{i=0}^{n}\ x_i \in S$ describes the sequence of state points (solution candidates) and $f(\{x\}_{i=0}^{n}) = \{f(x)\}_{i=0}^{n} = \{v\}_{i=0}^{n}, v_i \in \mathbb{R}$ describes the sequence of fitness values for the result. [73]
There are several walks to analyze the search space. The walk starts from a random solution candidate and uses the neighbourhood search to "walk" to neighbouring solution candidates. Depending on the type of neighbourhood search, several different types of walks are possible [74]:

- During the random walk, a state point is randomly selected from a set of neighbours.
- During the adaptive walk, the better state point (neighbour) is selected. Here several strategies can be used to choose which better neighbour state point to choose, for example, any solution (any ascent), steepest ascent, minimum ascent.
- During the reverse adaptive walk, the worse neighbour is selected. This is the reverse of the adaptive walk.
- During the uphill-downhill walk, first, an adaptive walk is performed, and then if a better fitness point (solution) is not found during the step, a reverse adaptive walk is performed until a better state point is found.
- In the neutral walk, a neighbour is selected, whose fitness value is equal to the fitness value of the current state (and try to increase the distance from the starting solution).

Operators of population-based algorithms (e.g., genetic algorithms) can also be used for sampling. Here, a recombination operator is used, which can give samples that are very different from the parent samples. [73]

Information analysis

Information analysis attempts to quantify the frequency of relative sequences between neighbours from an information-theoretical perspective. The measurement of information content can be linked-to Shannon's name. It can be described as the uncertainty of the state of the system. The higher the information content, the harder it is to make an estimate. The more information, the more complex the fitness landscape. By analogy, it analyses the sequence of fitness values. [73]
Starting from a series of fitness values (e.g., taking random steps $\{f(x_i)\}_{i=0}^{n}$), the first is to calculate the relative fitness difference. So it first subtracts the values of the random walk, which can be calculated with the following formula [73]:

$$\{d_i\}_{i=1}^{n} = \{f(x_i) - f(x_{i-1})\}_{i=1}^{n}. \tag{118}$$

These fitness differences can be ascending slopes, denoted by ↗ or 1, descending slopes, denoted by ↘ or -1, and a straight slope, denoted by ⟶ or 0. The following function (relaxed sing) can be defined using a threshold value of ε [73]:

$$\widetilde{sgn}_{\varepsilon}(x) := \begin{cases} x < \varepsilon \ : \ -1 \ (i.e. \searrow), \\ x > \varepsilon \ : \ 1 \ (i.e. \nearrow), \\ otherwise \ : \ 0 \ (i.e. \longrightarrow). \end{cases} \tag{119}$$

The result sequence, i.e. $\{S_i\}_{i=1}^{n} = \{\widetilde{sgn}_{\varepsilon}(d_i)\}_{i=1}^{n}$, is used to form multisets of consecutive symbol pairs, this given by the following formula [73]:

$$P := \{[S_i S_{i+1}] \big| i \in \{1, \dots, n-1\}\}. \tag{120}$$

The multiset of symbol pairs represents the slope change between all intermediate points on the trajectory and serves as a basis for information analysis. *Figure 12* illustrates all possible consecutive slope combinations. [73]

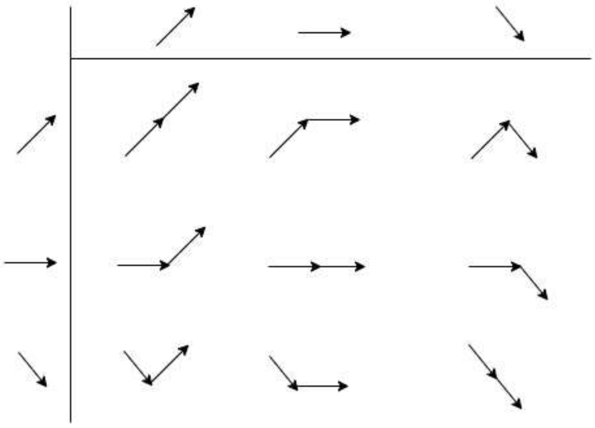

Figure 12.: The combinations [73]

Partial information content

This measure is the number of slope changes in the trajectory. Given the sequence of slopes, that is, $S = \{S_i\}_{i=1}^{n} = \{\widetilde{sgn}_\varepsilon(d_i)\}_{i=1}^{n}$. The new sequence is constructed using a σ function that first removes the → slopes. Then σ ensures that the successive slopes are different. So the sequence of alternating slopes σ is created by ↗↘ ⋯ ↘↗or ↘↗ ⋯ ↗↘. This sequence is the relative length of the original sequence which is important. The greater the relative length, the more rugged the trajectory. This sequence is based not only on the trajectory itself but also on the value of ε, which categorizes straight or non-straight successive steps. The partial information content thus shows the number of slope changes [73]:

$$M(\varepsilon) \coloneqq \frac{|\sigma(S)|}{|S|}. \tag{121}$$

In the formula $|S|$ denotes the length of the sequence S. If $M(\varepsilon) = 0$, it means that there are no slopes during the walk, so the landscape is flat or smooth. Similarly, if $M(\varepsilon) = 1$, it means that the path is very rugged. The number of expected optima during n long random steps can be calculated by the following formula [76]:

$$\mathbb{E}[M(\varepsilon)] = \left\lfloor \frac{n \cdot M(\varepsilon)}{2} \right\rfloor. \tag{122}$$

Information stability

Information stability gives the maximum size of a random step and is shown by the following formula [73]:

$$\varepsilon^* \coloneqq \max\{f(x_i) - f(x_{i-1})\}_{i=1}^{n}. \tag{123}$$

Information content

For this measure, the consecutive symbols are used, i.e. $P \coloneqq \{[S_i S_{i+1}] | i \in \{1, \dots, n-1\}\}$. The entropy can be calculated as follows, where i denotes the different states and P_i denotes the probability that state i. is the current state [73]:

$$H(S) := -\sum_{i} P_i \cdot lnP_i. \tag{124}$$

For a given entropy reaches its maximum when all states have the same probability. The formula can also be applied directly to our microstates, which are different combinations of slopes, illustrated in the table already presented (*Figure 15*) as follows [73]:

$$H(\varepsilon) := -\sum_{p \neq q} P_{[pq]} \cdot log_6 P_{[pq]}. \tag{125}$$

in the formula, $p, q \in \{S_i\}_{i=1}^{n}$ are slopes and $[pq] \in P$ are symbol pairs.

Density Basin Information

Density Basin Information is defined by the following formula [73]:

$$h(\varepsilon) := -\sum_{p = q} P_{[pq]} \cdot log_3 P_{[pq]}. \tag{126}$$

Regularity

Another measure of the sampled trajectory is regularity. This represents a series of differences in fitness values given by the following formula [73]:

$$r(\varepsilon) := |\{d_i | d_i \in \{d_i\}_{i=1}^{n}\}|. \tag{127}$$

Evolvability

The possibility of evolution is one of the most important in the nature, and since many metaheuristics have been inspired by evolution, it is also an important factor in the optimization. It is also called the ability of a population to produce a better individual than before. Continuous evolvability is important in evolutionary algorithms due to the variety of state points (solutions). The more diverse the individuals in the metaheuristics population, the better the algorithm traverses the search space. [73]

Based on the above, evolvability means the probability of finding better neighbours, given by the following formula [73,112]:

$$E_a = e(x) := \frac{|\{n \in N(x) | f(n) \geq f(x)\}|}{|N(x)|}. \tag{128}$$

So neighbours that are better than the selected state point are divided by all the neighbours of the state point.

The *evolvability portrait* can be defined as follows [73,112]:

$$E_b = \frac{|\{\sum f(n) \, | n \in N(x)\}|}{|N(x)|}. \tag{129}$$

The formula gives the average of the fitness values of the neighbouring state points.

6.2.3. Calculate distances between solutions

The distance between two possible solutions in the search space can be determined in several ways. In the dissertation, I used three different techniques.

1. Fitness distance: the absolute value of the difference in fitness value between two solutions.
2. Hamming distance [113]: the number of different elements in each position in the representation of two solutions. For example, *Figure 13.* represents the two solutions where the Hamming distance between the two solutions is 4.

First solution								
1	**3**	**2**	**5**	**4**	**6**	**8**	**7**	**9**

Second solution								
1	**3**	**8**	**6**	**4**	**5**	**2**	**7**	**9**

Figure 13.: Two example solutions

3. Basic Swap Sequence (BSS) distance [114]: the number of exchanges between two solution permutations, i.e. how many pieces can be produced from one solution to the other with a minimum edge change. In the example above, the BSS distance between the two solutions is 1, because with a single edge change we can get solution 2 from solution 1.

7. Fitness landscape analysis for optimization tasks

In this chapter, an analysis of the search space of the Vehicle Routing Problem is presented. My goal is to investigate different multi-objective optimization techniques, heuristic optimization algorithms, and neighbourhood operators from the view point of fitness landscape characteristics. In generally, the quality and representation of the search space depends on the nature of the optimization task. For example, in the case of a complex Vehicle Routing Problem, all the components are included (*Chapter 3.*). The task presented in this chapter is a Multi-Echelon VRP with the following properties: 4 levels, 5 nodes in the depot-level, 10 nodes in the first level satellite, 10 nodes in the second level satellite and 15 nodes in the customer level. In total, therefore, the number of nodes is 40. The system contains 1 type of product and I assume, that there is only one period. Each level contains 2-2 vehicles, for a total of 8 vehicles. Among the node properties, the following factors are taken into account: travel time between the nodes, travel distance between the nodes and route quality between the nodes. Among the attributes of the vehicles, I considered the following components: capacity constraint, fuel consumption and rental fee. Among the temporality attributes, the sample system includes the following: loading time, unloading time, administration time. Among the attributes of the products, I used the product demand of the node factor. Among the cost attributes, the following were included in the system: loading costs, unloading costs, administrative costs and quality control cost. The delivery parameter is used in addition to the operational parameters. The following metrics were used: length of the route, fuel consumption, vehicle rental fee, route quality, route time and unvisited customers.

During the analysis of the search space, I also used the following methods: *analysis of the heuristic optimization algorithms* and the *sampling methods (trajectory-based sampling, scattered sampling).*

During the analysis of the search space, first, the *analysis of the heuristic algorithms* is presented. This section includes also the analysis of the results of multi-objective optimization techniques for each optimization algorithm. I analyzed also each heuristic algorithm and their solution set (*analysis of the solutions of the heuristic algorithms*). and both the results of the iteration of the heuristic algorithms (*analysis of the iteration of the iterative heuristic algorithms*).

In the work also the trajectory-based sampling (*operator analysis (with walks)*) and the random walk were also analyzed in terms of *information theory (with random content analysis)*.

Then, I also presented scattered sampling and analyzed the resulting samples (*analysis of the filtered search space*). Finally, a *fitness cloud* analysis is presented in my work.

The fitness landscape analysis used in my investigation is implemented in Java. In the analysis for the above metrics not only the running time but the iteration count was taken into account.

7.1. Analysis of heuristic optimization algorithms

In this subsection I present the analysis of the heuristic optimization algorithms. This subsection contains the followings: the analysis of the results of multi-objective optimization techniques, the analysis of the solutions of heuristic algorithms and the analysis of the iteration of the iterative heuristic algorithms.

7.1.1. Analysis of the results of multi-objective optimization techniques

First, a general overview of the multi-objective optimization are compared. Simultaneously with the multi-objective optimization methods, the efficiency of each algorithm is also analyzed.
In my analysis I worked with the following objective function components: length of the route, unloading cost, administrative cost, fuel consumption, vehicle rental fee, route quality, route time, loading time, unloading time, administrative time.
The following algorithms were included into my analysis: ACS, ACS_C, AS, AS_C, ESAS, ESAS_C, FA, FA_C, FCHC, FCHC_AI, FCHC_CI, FCHC_FI, FCHC_NI, FCHC_NN, FCHC_G, GA, GA_C, HS, HS_C, MMAS, MMAS_C, PSO, PSO_C, RBVAS, RBVAS_C, SA, SA_AI, SA_CI, SA_FI, SA_NI, SA_NN, SA_G, TS, TS_AI, TS_CI, TS_FI, TS_NI, TS_NN, TS_G, AI, CI, FI, NI, NN, G.

The following multi-objective optimization techniques are analyzed: Bounded Objective Function Method (BOFM), Pareto Ranking (PR), Weighted Exponential Sum Method (WESM), Weighted Global Criterion Method (WGCM), Weighted Product Method (WPM), Weighted Sum Method (WSM).

The purpose of the test experiment is to examine the results of each component of the effect of the components of the objective function. The purpose of the measurement was to determine the best multi-objective optimization technique and the best heuristic algorithm. During the evaluations, I first examined which multi-objective optimization technique was best on average (regardless of heuristics). I then examined which heuristic algorithm provided the best solution (independent of multi-objective optimization methods). I then examined which combination of multi-objective optimization technique and heuristic algorithm gave the best results. During the measurement, the results of each objective function component are evaluated. The objective is to minimize most of the components, but for some components, the objective is the maximization.

Figure 1. multi.objective.optimization.figure [W/1] and *Table 1. multi.objective.optimization.table [W/1]* show the effect of the <u>length of the route</u> metric. On average, the best multi-objective optimization technique for the length of the route metric was the BOFM technique. From the heuristics, the TS_G technique proved to be the best on average. Comparing multi-objective optimization techniques and heuristics, the combination of WESM and TS_NI provided the best solution.
Figure 2. multi.objective.optimization.figure [W/1] and *Table 2. multi.objective.optimization.table [W/1]* show the results of the <u>unloading cost</u> component. Based on this, the best multi-objective optimization technique on average was BOFM. The best heuristic algorithm is FCHC_NN. The best combination is a combination of WPM multi-objective optimization and FCHC_NN heuristics.
Figure 3. multi.objective.optimization.figure [W/1] and the *Table 3. multi.objective.optimization.table [W/1]* show the results for the <u>administrative cost</u> component. The best multi-objective optimization technique on average was also the BOFM technique. According to heuristic algorithms, on average, the AS_C technique was the best. The best combination of multi-objective optimization technique and heuristic algorithm was given by the combination of WESM and ACS_C.
According to the *Figure 4. multi.objective.optimization.figure [W/1]* and the *Table 4. multi.objective.optimization.table [W/1]*, the best multi-objective optimization technique for the <u>fuel consumption</u> component was on average the WGCM technique. According to the heuristic algorithms, on average, the TS_G algorithm was the best. The best combination was given by the combination of WESM multi-objective optimization technique and TS_NI heuristics.
According to *Figure 5. multi.objective.optimization.figure [W/1]* and *Table 5. multi.objective.optimization.table [W/1]*, the best multi-objective optimization technique for the <u>vehicle rental fee</u> component was WESM on average. Of the heuristic algorithms, SA_AI was the best. Considering the combination of the multi-objective optimization technique and the heuristic

algorithm, the WESM multi-objective optimization technique and the FCHC_G heuristic gave the best results.
According to the *Figure 6. multi.objective.optimization.figure [W/1]* and the *Table 6. multi.objective.optimization.table [W/1]*, the best multi-objective optimization technique for the route quality component was on average the WSM technique. And in terms of heuristic algorithms, HS was the best. Looking at the combination of multi-objective optimization technique and heuristic algorithms, BOFM and HS_C gave the best results.
According to *Figure 7. multi.objective.optimization.figure [W/1]* and *Table 7. multi.objective.optimization.table [W/1]*, the best multi-objective optimization technique for the route time component was on average the PR technique. According to algorithms, SA_NI was the best. According to the combination of multi-objective optimization technique and heuristic algorithms, the combination of PR and SA_CI gave the best results.
Figure 8. multi.objective.optimization.figure [W/1] and *Table 8. multi.objective.optimization.table [W/1]* illustrate the results for the loading time component. The best multi-objective optimization technique on average was the WSM technique. The best heuristic algorithm was SA_CI. In terms of the combination of the multi-objective optimization technique and the heuristic algorithm, the combination of WSM and SA gave the best results.
According to the *Figure 9. multi.objective.optimization.figure [W/1]* and the *Table 9. multi.objective.optimization.table [W/1],* the best multi-objective optimization technique for the unloading time component was the WSM technique on average. According to the heuristic algorithms, RBVAS_C was the best. Taking the multi-objective optimization technique and the heuristic algorithm together, the combination of BOFM and TS_NN gave the best results.
According to *Figure 10. multi.objective.optimization.figure [W/1]* and the *Table 10. multi.objective.optimization.table [W/1]*, the best multi-objective optimization technique for the administrative time component was the WGCM technique on average. And in terms of algorithms, FCHC_CI was the best. The best result in terms of the combination of the multi-objective optimization technique and the heuristic algorithm was given by the combination of WGCM and HS_C.

7.1.2. Analysis of the solutions of heuristic algorithms

In this section, I first run each heuristic algorithm and then performed analyzes on the solutions provided by the heuristic algorithms, so my solution set contains the results of each heuristic algorithm. The following heuristic algorithms are used: Ant Colony System, Ant System, Elitist Strategy of Ant System, Firefly Algorithm, First Choice Hill Climbing, Genetic Algorithm, Harmony Search, MAX-MIN Ant System, Particle Swarm Optimization, Rank Based Version of Ant System, Simulated Annealing, Tabu Search, Nearest Neighbour, Greedy, Nearest Insertion, Farthest Insertion, Cheapest Insertion and Arbitrary Insertion.
The following analysis methods were considered: distance of solutions, the distance of best solution, the distance of best solution and solutions, cost density, distance of best solution with filtered global optima, the distance of best solution and solutions with filtered global optima, cost density with filtered global optima.
During the measurements, the distances between the elements of the solution set are analyzed, where the following distances are used: fitness distance, Hamming distance and basic swap sequence distance.

According to *Figure 1. heuristic.algorithm.analysis.figure [W/2]*, average fitness distances describe a parabolic-like function. The x-axis contains fitness values that range from ≈110,000 to ≈140,000. The average fitness distance values are on the y-axis, these values are between ≈10,000 and ≈21,000. It can be seen that solutions with low or high fitness value have a large average distance from other solutions, while solutions with a fitness value moving in the middle of the scale have a small average distance. According to *Figure 2. heuristic.algorithm.analysis.figure [W/2],* the average Hamming distances range from 33 to 36, and the fitness value of each solution does not affect the average Hamming distance from the other solutions. According to *Figure 3.*

heuristic.algorithm.analysis.figure [W/2], the average basic swap sequence distances range from 25 to 28 and the fitness value of each solution does not affect the average basic swap sequence distance from the other solutions. According to *Figure 4. heuristic.algorithm.analysis.figure [W/2],* the fitness distances related to the best solution (the best of the solutions given by the heuristic algorithms) represent a linear function, as the fitness value of the solutions increases, so does the best solution (the best of the solutions provided by the heuristic algorithms). According to the *Figure 5. heuristic.algorithm.analysis.figure [W/2]*, the Hamming distances related to the best solution (the best of the solutions provided by the heuristic algorithms) range from 33 to 39, and the fitness value of the solution does not affect the best solution took Hamming distance. According to the *Figure 6. heuristic.algorithm.analysis.figure [W/2]*, the basic swap sequence distances related to the best solution (the best of the solutions provided by the heuristic algorithms) range from 25 to 32, and the fitness value of the solution does not affect the best solution (the best of the solutions given by the heuristic algorithms). According to *Figure 7. heuristic.algorithm.analysis.figure [W/2]*, the function of the fitness distances related to the best solution (the best of the solutions given by the heuristic algorithms) and the average distances related to the other solutions are also a parabolic function. The Hamming distances (*Figure 8. heuristic.algorithm.analysis.figure [W/2]*) and basic swap sequence distances (*Figure 9. heuristic.algorithm.analysis.figure [W/2]*) move around one point each. *Figure 10. heuristic.algorithm.analysis.figure [W/2]* shows the cost density values, according to which the fitness values of each solution are very different. The summary table is *Table 1. fitness.landscape.table [A/12].*

<table>
<tr><th colspan="2">Analysis of the solutions of heuristic algorithms</th></tr>
<tr><th>Type</th><th>Optimal value</th></tr>
<tr><td>Fitness values</td><td>The greater the difference between the lower and upper limits, the better. And the lower limit should be small because the goal is to minimize fitness value</td></tr>
<tr><td>Average of fitness distances</td><td rowspan="3">The greater the distances, the better the search space is mapped.</td></tr>
<tr><td>Average of Hamming distances</td></tr>
<tr><td>Average of basic swap sequence distances</td></tr>
<tr><td>Fitness distances of best solution</td><td rowspan="3">Small lower bound and large upper bound. Then the operator explores the space well (large upper bound), but the operator also found a very good solution because of the small lower bound.</td></tr>
<tr><td>Hamming distances of best solution</td></tr>
<tr><td>Basic swap sequence distances of best solution</td></tr>
<tr><td>Cost density</td><td>For many, small values, it maps space well. So the objective is to create many different fitness value solutions.</td></tr>
</table>

Table 5.: Analysis of the solutions of heuristic algorithms: evaluation strategy

7.1.3. Analysis of the iteration of the iterative heuristic algorithms

In this subsection, I examine the results of each iterative (improvement) heuristic. The set of solutions for each algorithm means the solutions of each iteration. I used 10 iterations for the population algorithms and 100 iterations for the algorithms that improve a single solution.

The following algorithms are analyzed: Ant Colony System, Ant System, Elitist Strategy of Ant System, Firefly Algorithm, First Choice Hill Climbing, Genetic Algorithm, Harmony Search, MAX-MIN Ant System, Particle Swarm Optimization, Rank Based Version of Ant System, Simulated Annealing and Tabu Search.

The analysis techniques are the following: distance of solutions, the distance of best solution, the distance of best solution and solutions, cost density, distance of best solution with filtered global

optima, the distance of best solution and solutions with filtered global optima, cost density with filtered global optima.
During the measurements, I analyzed the distances between the elements of the solution set. I used the following distances: fitness distance, Hamming distance and basic swap sequence distance.

7.1.3.1. Ant Colony System

According to *Figure 1. iterative.heuristic.analysis.figure [W/3]*, the higher the fitness value of one solution, the higher the average of the fitness distances related to the other solutions. This shows a linear function. The averages of the Hamming and basic swap sequence distances are not overly influenced by the fitness value (*Figure 2-3. iterative.heuristic.analysis.figure [W/3]).* The fitness distance from the best solution naturally increases linearly as a function of the fitness value of the solutions (*Figure 4. iterative.heuristic.analysis.figure [W/3]*). The distances related to the best solution during Hamming and basic swap sequence distances are not affected by the fitness values of the solutions (*Figure 5-6. iterative.heuristic.analysis.figure [W/3]*). Plotting the averages of the distances as a function of the distance from the best solution gives a function that increases linearly over the fitness distance (*Figure 7. iterative.heuristic.analysis.figure [W/3]*). The functions of the Hamming and basic swap sequence distances (*Figure 8-9. iterative.heuristic.analysis.figure [W/3]*) also show an increasing trend in terms of the distance from the best solution and the average of the distances of the solutions. Cost density values (*Figure 10. iterative.heuristic.analysis.figure [W/3]*) range from 1 to 5, which means that some solutions are the same. Fitness distances from the filtered global optima result (*Figure 11. iterative.heuristic.analysis.figure [W/3]*) decrease with increasing fitness values, which means that the filtered global optima has a higher fitness value than Ant Colony System solutions. The Hamming and basic swap sequence distances (*Figure 12-13. iterative.heuristic.analysis.figure [W/3]*) are condensed into one node.
The summary table is illustrated in *Table 8. fitness.landscape.table [A/12]*. The results of the iteration of the algorithm are illustrated in *Figure 1. iterative.heuristic.analysis.iteration.figure [W/4]*.

7.1.3.2. Ant System

The method provides a fast convergence. Depending on the fitness values, considering the averages of the fitness values related to the solutions, it can be said that the increase of the fitness value increases with the average of the fitness distances (*Figure 18. iterative.heuristic.analysis.figure [W/3]*). The same result is valid for Hamming and basic swap sequence distances (*Figure 19-20. iterative.heuristic.analysis.figure [W/3]*), only here the values are relatively scattered in the coordinate system. The distance from the best solution increases linearly with the fitness value (*Figure 21. iterative.heuristic.analysis.figure [W/3]*). In case of Hamming and basic swap sequence distances (*Figure 22-23. iterative.heuristic.analysis.figure [W/3]*) the situation is similar, here some solutions have a small distance, while others have a relatively large distance. Considering the averages of the fitness distance from the best solution and the fitness distances from the other solutions, we get a parabolic function (*Figure 24. iterative.heuristic.analysis.figure [W/3]*). The Hamming and basic swap sequence distances (*Figure 25-26. iterative.heuristic.analysis.figure [W/3]*) are scattered in the coordinate system. In terms of cost density values (*Figure 27. iterative.heuristic.analysis.figure [W/3]*), some of the solutions obtained during the iteration are identical. The function of the fitness distance from the filtered global optima solution (*Figure 28. iterative.heuristic.analysis.figure [W/3])* shows a decreasing and then an increasing trend. This means that during the iteration, the algorithm managed to find a better one than the filtered optima solution. The Hamming and basic swap sequence distances (*Figure 29-30. iterative.heuristic.analysis.figure [W/3]*) are condensed into one node as a function of fitness value, no significant difference is noticeable.
The summary table is illustrated in *Table 9. fitness.landscape.table [A/12]*. The results of the iteration of the algorithm are illustrated in *Figure 2. iterative.heuristic.analysis.iteration.figure [W/4]*.

7.1.3.3. Elitist Strategy of Ant System

Depending on the fitness values, the fitness distances from the other solutions are condensed into one point (*Figure 35. iterative.heuristic.analysis.figure [W/3]*). But the averages of the Hamming and basic swap sequence distances (*Figure 36-37. iterative.heuristic.analysis.figure [W/3]*) are large. The distance from the best solution is a linear function of the fitness values (*Figure 38. iterative.heuristic.analysis.figure [W/3]*), from the Hamming and basic swap sequence distances (*Figure 39-40. iterative.heuristic.analysis.figure [W/3]*) this can also be said. Hamming and basic swap sequence distances range widely. The distance from the best solution and the averages of the distance from the solutions describe a parabolic function (*Figure 41. iterative.heuristic.analysis.figure [W/3]*). The Hamming and basic swap sequence distances (*Figure 42-43. iterative.heuristic.analysis.figure [W/3]*) are scattered. In terms of cost density values (*Figure 44. iterative.heuristic.analysis.figure [W/3]*), almost all solutions have unique fitness values. Fitness distances from the filtered global optima solution are small (*Figure 45. iterative.heuristic.analysis.figure [W/3]*), but Hamming and basic swap sequence distances (*Figure 46-47. iterative.heuristic.analysis.figure [W/3]*) no, they are compressed into a node as shown.
The summary table is illustrated in *Table 10. fitness.landscape.table [A/12]*. The results of the iteration of the algorithm are illustrated in *Figure 3. iterative.heuristic.analysis.iteration.figure [W/4]*.

7.1.3.4. Firefly Algorithm

During the firefly algorithm, all the solutions of the iteration gave the same solution, so the algorithm could not improve on the initial value. This initial value is about 140,000 fitness values. Thus, the values of each distance (fitness, Hamming, basic swap sequence) are 0, and each solution of the iteration has the same fitness value. Approximately 20,000 fitness values away from the filtered optima solution (*Figure 62. iterative.heuristic.analysis.figure [W/3]*), and 36 Hamming (*Figure 63. iterative.heuristic.analysis.figure [W/3]*) and 30 basic swap sequences distance (*Figure 64. iterative.heuristic.analysis.figure [W/3]*) is the solution of the iteration.
The summary table is illustrated in *Table 11. fitness.landscape.table [A/12]*. The results of the iteration of the algorithm are illustrated in *Figure 4. iterative.heuristic.analysis.iteration.figure [W/4]*.

7.1.3.5. First Choice Hill Climbing

Considering the fitness values and the means of the fitness distances (*Figure 69. iterative.heuristic.analysis.figure [W/3]*), it results a parabolic function. Thus, for low and high fitness value solutions, the averages of fitness distances are high, while for medium fitness value solutions, they are low. Also for Hamming and basic swap sequence distances (*Figure 70-71. iterative.heuristic.analysis.figure [W/3]*), the higher the fitness value, the higher the averages of the distances. Fitness distances from the best solution increase linearly as a function of fitness values *Figure 72. iterative.heuristic.analysis.figure [W/3]*. Hamming and basic swap sequence distances (*Figure 73-74. iterative.heuristic.analysis.figure [W/3]*) also show an increase, and these distances range on a relatively wide scale. The cost density (*Figure 78. iterative.heuristic.analysis.figure [W/3]*) ranges from 1 to 18, and many solutions have the same fitness value. The fitness distance of filtered global optima describes a V-shaped function as a function of fitness values, so as the fitness values increase, the distance decreases and then when it is close to 0, it starts to increase. This means that the first choice hill climbing algorithm found a better solution than the filtered global optima solution. Hamming and basic swap sequence distances (*Figure 80. iterative.heuristic.analysis.figure [W/3]*) are relatively concentrated in one place, not much different depending on fitness values.
The summary table is illustrated in *Table 12. fitness.landscape.table [A/12]*. The results of the iteration of the algorithm are illustrated in *Figure 5. iterative.heuristic.analysis.iteration.figure [W/4]*.

7.1.3.6. Genetic Algorithm

The fitness values of the solutions of the genetic algorithm are close to each other. The averages of fitness distances are also small (*Figure 86. iterative.heuristic.analysis.figure [W/3]*). The Hamming and basic swap sequence distances (*Figure 87-88. iterative.heuristic.analysis.figure [W/3]*) are also small. Therefore, fitness distances from the best solution are also small (*Figure 89. iterative.heuristic.analysis.figure [W/3]*). The Hamming and basic swap sequence distances (*Figure 90-91. iterative.heuristic.analysis.figure [W/3]*) are also small. According to the cost density (*Figure 95. iterative.heuristic.analysis.figure [W/3]*), 7 solutions also have the same fitness value, which can be said to be a high number. Fitness distances from the filtered global optima solution (*Figure 96. iterative.heuristic.analysis.figure [W/3]*) are small. The Hamming and basic swap sequence distances from the filtered global optima (*Figure 97-98. iterative.heuristic.analysis.figure [W/3]*), on the other hand, are no longer so small, the distance values fall within a narrow range.
The summary table is illustrated in *Table 13. fitness.landscape.table [A/12]*. The results of the iteration of the algorithm are illustrated in *Figure 6. iterative.heuristic.analysis.iteration.figure [W/4]*.

7.1.3.7. Harmony Search

The results of the Harmony Search iterations are the same, so the average fitness, Hamming, and basic swap sequence distance is 0. The cost density chart also shows a single fitness value. The fitness distance from the filtered global optima solution is small, but the Hamming and basic swap sequence distances are large.
The summary table is illustrated in *Table 14. fitness.landscape.table [A/12]*. The results of the iteration of the algorithm are illustrated in *Figure 7. iterative.heuristic.analysis.iteration.figure [W/4]*.

7.1.3.8. MAX-MIN Ant System

During iterations of the MAX-MIN Ant System, the averages of the fitness distances between the solutions (*Figure 120. iterative.heuristic.analysis.figure [W/3]*) move in a small interval. The averages of the Hamming and basic swap sequence distances (*Figure 121-122. iterative.heuristic.analysis.figure [W/3]*) also move in small intervals. Fitness distances from the best solution (*Figure 123. iterative.heuristic.analysis.figure [W/3]*) are small. The Hamming and basic swap sequence distances (*Figure 124-125. iterative.heuristic.analysis.figure [W/3]*), on the other hand, are already moving in larger intervals. The cost density (*Figure 129. iterative.heuristic.analysis.figure [W/3]*) shows that multiple solutions have the same fitness value. The fitness distance from the filtered global optima solution (*Figure 130. iterative.heuristic.analysis.figure [W/3]*) is not large either, but the Hamming and basic swap sequence distances (*Figure 131-132. iterative.heuristic.analysis.figure [W/3]*) already show a higher value.
The summary table is illustrated in *Table 15. fitness.landscape.table [A/12]*. The results of the iteration of the algorithm are illustrated in *Figure 8. iterative.heuristic.analysis.iteration.figure [W/4]*.

7.1.3.9. Particle Swarm Optimization

In the case of the Particle Swarm Optimization, the averages of fitness distances (*Figure 137. iterative.heuristic.analysis.figure [W/3]*) are relatively large. The averages of the Hamming distances (*Figure 138. iterative.heuristic.analysis.figure [W/3]*) are also large, but the basic swap sequence distances (*Figure 139. iterative.heuristic.analysis.figure [W/3]*) are already smaller. Fitness distances related to the best solution (*Figure 140. iterative.heuristic.analysis.figure [W/3]*) increases linearly as a function of fitness values. The Hamming and basic swap sequence distances related to the best solution (*Figure 141-142. iterative.heuristic.analysis.figure [W/3]*) range widely. Cost density values (*Figure 146. iterative.heuristic.analysis.figure [W/3]*) range from 1 to 3. The

distance from the filtered global optima solution (*Figure 147. iterative.heuristic.analysis.figure [W/3]*) decreases with increasing fitness functions, which means that the filtered global optima has a higher fitness value than the results of Particle Swarm Optimization iterations. Hamming and basic swap sequence distances (*Figure 148-149. iterative.heuristic.analysis.figure [W/3]*) are large and move in small intervals.
The summary table is illustrated in *Table 16. fitness.landscape.table [A/12]*. The results of the iteration of the algorithm are illustrated in *Figure 9. iterative.heuristic.analysis.iteration.figure [W/4]*.

7.1.3.10. Rank Based Version of Ant System

Averages of fitness distances move in small intervals (*Figure 154. iterative.heuristic.analysis.figure [W/3]*), averages of Hamming and basic swap sequence distances (*Figure 155-156. iterative.heuristic.analysis.figure [W/3]*) are small. Fitness distances from the best solution (*Figure 157. iterative.heuristic.analysis.figure [W/3]*) are small, Hamming and basic swap sequence distances (*Figure 158-159. iterative.heuristic.analysis.figure [W/3]*) however, they move in larger intervals, and as the fitness value increases, the Hamming and basic swap sequence distances from the best solution also increase. The cost density values (*Figure 163. iterative.heuristic.analysis.figure [W/3]*) range from 1 to 4, so several solutions have the same fitness value. The fitness distance from the filtered global optima solution (*Figure 164. iterative.heuristic.analysis.figure [W/3]*) moves in a larger interval, but the Hamming and basic swap sequence distances (*Figure 165-166. iterative.heuristic.analysis.figure [W/3]*) move at small intervals, but these distances are large.
The summary table is illustrated in *Table 17. fitness.landscape.table [A/12]*. The results of the algorithm iteration are illustrated in *Figure 10. iterative.heuristic.analysis.iteration.figure [W/4]*.

7.1.3.11. Simulated Annealing

The averages of the fitness distances of the solutions and the fitness value (*Figure 171. iterative.heuristic.analysis.figure [W/3]*) describe a parabolic function. The averages of the Hamming and basic swap sequence distances (*Figure 172-173. iterative.heuristic.analysis.figure [W/3]*) range widely. The fitness distance from the best solution (*Figure 174. iterative.heuristic.analysis.figure [W/3]*) increases as a function of fitness values. Even for Hamming and basic swap sequence distances related to the best solution (*Figure 175-176. iterative.heuristic.analysis.figure [W/3]*), this distance increases with increasing fitness values and these distances move at large intervals. The cost density (*Figure 180. iterative.heuristic.analysis.figure [W/3]*) ranges from 1 to 36, with several solutions with the same fitness value. Fitness distances related to the filtered global optima solution (*Figure 181. iterative.heuristic.analysis.figure [W/3]*) decrease with increasing fitness distances, so the results of the Simulated Annealing iteration provided a better solution than the filtered optima solution. The Hamming and basic swap sequence distances (*Figure 182-183. iterative.heuristic.analysis.figure [W/3]*) related to the filtered optima solution move on a small scale and are compressed into one place.
The summary table is illustrated in *Table 18. fitness.landscape.table [A/12]*. The results of the algorithm iteration are illustrated in *Figure 11. iterative.heuristic.analysis.iteration.figure [W/4]*.

7.1.3.12. Tabu Search

Average fitness distances (*Figure 188. iterative.heuristic.analysis.figure [W/3]*) show a first decreasing and then an increasing trend as a function of fitness values. The averages of the Hamming and basic swap sequence distances (*Figure 189-190. iterative.heuristic.analysis.figure [W/3]*) are very small, around 1-4. The fitness distance from the best solution (*Figure 191. iterative.heuristic.analysis.figure [W/3]*) increases as a function of fitness value. The Hamming and basic swap sequence distances from the best solution (*Figure 192-193. iterative.heuristic.analysis.figure [W/3]*) are very small, around 0-4. The cost density values (*Figure 197. iterative.heuristic.analysis.figure [W/3]*) are very high, there are 20-30 solutions for

several fitness values. Fitness distances related to the filtered global optima solution (*Figure 198. iterative.heuristic.analysis.figure [W/3]*) decrease as a function of fitness distances. This means that the Tabu Search algorithm found a better solution than the filtered optima solution. The Hamming and basic swap sequence distances from the filtered global optima solution (*Figure 199-200. iterative.heuristic.analysis.figure [W/3]*) are large and move in small intervals.
The summary table is illustrated in *Table 19. fitness.landscape.table [A/12]*. The results of the iteration of the algorithm are illustrated in *Figure 12. iterative.heuristic.analysis.iteration.figure [W/4]*.

Analysis of the iteration of the iterative heuristic algorithms	
Type	**Optimal value**
Fitness values	The greater the difference between the lower and upper bound, the better. And the lower limit should be small because the goal is to minimize fitness value.
Average of fitness distances	The greater the distances, the better the algorithm maps the search space.
Average of Hamming distances	
Average of basic swap sequence distances	
Fitness distances of best solution	Small lower bound and large upper bound. Then the operator explores the space well (large upper bound), but it also found a very good solution because of the small lower bound.
Hamming distances of best solution	
Basic swap sequence distances of best solution	
Cost density	For many, small values, it maps space well. So the goal is to create many different fitness value solutions.
Fitness distance of filtered global optima	Small lower bound and large upper bound. Then the operator explores the space well (large upper bound), but it also found a very good solution because of the small lower bound.
Hamming distance of filtered global optima	
Basic swap sequence distance of filtered global optima	

Table 6.:Analysis of the iteration of the iterative heuristic algorithms: evaluation strategy

Analysis of the iteration of the iterative heuristic algorithms												
Type	ACS	AS	ESAS	FA	FCHC	GA	HS	MMAS	PSO	RBVAS	SA	TS
Fitness values	✓	✓	✓	x	✓	✓	x	✓	✓	✓	✓	✓
Average of fitness distances	✓	✓	✓	x	✓	✓	x	✓	✓	✓	✓	✓
Average of Hamming distances												
Average of basic swap sequence distances												
Fitness distances of best solution	✓	✓	✓	x	✓	✓	x	✓	✓	✓	✓	x
Hamming distances of best solution												
Basic swap sequence distances of best solution												
Cost density	✓	✓	✓	x	x	✓	x	✓	✓	✓	✓	x
Fitness distance of filtered global optima	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Hamming distance of filtered global optima												
Basic swap sequence distance of filtered global optima												

Table 7.: Analysis of the iteration of the iterative heuristic algorithms: evaluation

Analysis of the iteration of the iterative heuristic algorithms	
Effective algorithm	**Not effective algorithm**
ACS, AS, ESAS, FCHC, GA, MMAS, RBVAS, SA, TS	FA, HS, PSO

Table 8.: Analysis of the iteration of the iterative heuristic algorithms: summary

7.2. Trajectory-based sampling

Trajectory-based sampling provides a general characterization of the search space. The method does not measure the efficiency of the algorithms but determine the neighbourhood relationships. Trajectory-based sampling initially starts from a random state point (initially, this state point is the current state point). The next current state point is selected from the neighbours of the current state point. Sampling is continued until a certain iteration.

7.2.1. Operator analysis (with walks)

In this procedure the following walk operators are analyzed: Random Walk, Adaptive Walk, Reverse Adaptive Walk, Uphill-Downhill Walk, Neutral Walk, Reverse Neutral Walk.
I used 100 iterations of each walk operator during the test runs. I also examined the relationship between the individual solutions of the iteration with the following methods: distance of solutions, the distance of best solution, the distance of best solution and solutions, cost density, distance of best solution with filtered global optima, the distance of best solution and solutions with filtered global optima, cost density with filtered global optima. This means that the elements of the solution set are the solutions of the iteration.
The following neighbourhood techniques are analyzed: 2-opt, Order Crossover (OX), Cycle Crossover (CX), Partially Matched Crossover (PMX).
During the measurements, I analyzed the distances between the elements of the solution set. I used the following distances: fitness distance, Hamming distance, basic swap sequence distance.

7.2.1.1. Random Walk

Figure 1-3. operator.analysis.figure [W/5] illustrate the results of the 2-opt operator, where distances are calculated from the elements of the solution set. According to *Figure 1. operator.analysis.figure [W/5]*, the average fitness distances during the random walk and 2-opt operator describe a parabolic function. Fitness values range from ≈120,000 to ≈140,000, while average fitness distances range from ≈2,500 to ≈9,000. If a solution has a low or high fitness value, the average fitness distance from the other solutions is high, while if it has a medium fitness value, the average distance is low. According to *Figure 2. operator.analysis.figure [W/5]*, average Hamming distances are not affected by fitness value. Average Hamming distances range from 28 to 34. According to *Figure 3. operator.analysis.figure [W/5]*, average basic swap sequence distances are not affected by fitness value. The average basic swap sequence distances are between 20 and 26.
Figure 4-6. operator.analysis.figure [W/5] illustrate the results of the order crossover operator, where distances are calculated from the elements of the solution set. According to *Figure 4. operator.analysis.figure [W/5]*, the average fitness distances during the random walk and order crossover operators describe a parabolic function. Fitness values range from ≈120,000 to ≈140,000, while average fitness distances range from ≈2,000 to ≈10,000. If a solution has a low or high fitness value, the average fitness distance from the other solutions is high, while if it has a medium fitness value, the average distance is low. According to *Figure 5. operator.analysis.figure [W/5]*, average Hamming distances are not affected by fitness value. Average Hamming distances range from 30 to 34. According to *Figure 6. operator.analysis.figure [W/5]*, average basic swap sequence distances are not affected by fitness value. The average basic swap sequence distances are between 23 and 28.

Figure 7-9. operator.analysis.figure [W/5] illustrate the results of the cycle crossover operator, where the distances are calculated from the elements of the solution set. The distances describe here a parabolic function. Fitness values range from ≈120,000 to ≈140,000, while average fitness distances range from ≈1,800 to ≈5,000. If the standard deviation of the distributions is large, the distance between the points is large. According to *Figure 8. operator.analysis.figure [W/5]*, average Hamming distances are not affected by fitness value. Average Hamming distances range from 25 to 30. According to *Figure 9. operator.analysis.figure [W/5]*, average basic swap sequence distances are not affected by fitness value. The average basic swap sequence distances are between 20 and 24.

Figure 10-12. operator.analysis.figure [W/5] illustrate the results of the partially matched crossover operator, where distances are calculated from the elements of the solution set. According to *Figure 10. operator.analysis.figure [W/5]*, the average fitness distances during the random walk and partially matched crossover operators describe a parabolic function. Fitness values range from ≈120,000 to ≈140,000, while average fitness distances range from ≈2,000 to ≈7,500. According to *Figure 11. operator.analysis.figure [W/5]*, average Hamming distances are not affected by fitness value. As different properties are measured, the average Hamming distances range from 30 to 34. According to *Figure 12. operator.analysis.figure [W/5]*, average basic swap sequence distances are not affected by fitness value. The average basic swap sequence distances are between 23 and 27.

Figure 13-15. operator.analysis.figure [W/5] illustrate the results of the 2-opt operator, where the distances are calculated from the best element of the solution set. *Figure 13. operator.analysis.figure [W/5]* shows the fitness values and the fitness distance from the best solution for random walk and 2-opt. The higher the fitness value of a solution, obviously the farther away from the best solution. According to *Figure 14. operator.analysis.figure [W/5]*, Hamming distances range from 2 to 38. The greater the fitness value of a solution, the greater the Hamming distance from the best solution. According to *Figure 15. operator.analysis.figure [W/5]*, the basic swap sequence distances are between 1 and 32. The higher the fitness value of a solution, the greater the distance of the basic swap from the best solution.

Figure 16-18. operator.analysis.figure [W/5] illustrate the results of the order crossover operator, where distances are calculated from the best element in the solution set. *Figure 16. operator.analysis.figure [W/5]* shows the fitness values and the fitness distance from the best solution for the random walk and order crossover. The higher the fitness value of a solution, obviously the farther away from the best solution. According to *Figure 17. operator.analysis.figure [W/5]*, Hamming distances range from 2 to 40. The greater the fitness value of a solution, the greater the Hamming distance from the best solution. According to *Figure 18. operator.analysis.figure [W/5]*, the basic swap sequence distances are between 2 and 34. The higher the fitness value of a solution, the greater the distance of the basic swap from the best solution.

Figure 19-21. operator.analysis.figure [W/5] illustrate the results of the order crossover operator, while *Figure 22-24. operator.analysis.figure [W/5]* illustrate the results of a partially matched crossover, where distances are calculated from the best element of the solution set. These analyzes also show that the higher the fitness value of a solution, the greater the distance of the fitness from the best solution, and the Hamming and basic swap sequence distances also increase as a function of the fitness value. For a cycle crossover, the Hamming distance is 0-38, while the basic swap sequence distance is 0-29, and for a partially matched crossover, the Hamming distance is 2-40 and the basic swap sequence distance is 1-34.

Figure 25-36. operator.analysis.figure [W/5] illustrate the distance from the best solution and the average of the distances from the other solutions. *Figure 25-27. operator.analysis.figure [W/5]* illustrate the results for the 2-opt operator, *Figure 28-30. operator.analysis.figure [W/5]* for order crossover, *Figure 31-33. operator.analysis.figure [W/5]* for the cycle crossover operator, while *Figure 34-36. operator.analysis.figure [W/5]* for the partially matched crossover operator. According to *Figure 25. operator.analysis.figure [W/5]*, this distance is a parabolic function, so for large and small distances, the average distances are large, while for medium distances the average distances are small for a random walk and 2-opt. According to *Figure 26. operator.analysis.figure [W/5]*, the distance from the best solution during Hamming distances does not affect the average distance here. *Figure 27. operator.analysis.figure [W/5]* shows the results for the basic swap sequence, it is true also here that the average basic swap sequence distance is not affected by the distance from the best solution to the given result. According to *Figure 28. operator.analysis.figure [W/5]*, in the case of order crossover, it results a parabolic function for fitness distances. The results for cycle crossover and partially matched crossover are similar to those for 2-opt and cycle crossover.

Figure 37-40. operator.analysis.figure [W/5] show the cost density results, i.e., how many solutions with the same fitness values are present between each solution. Based on the results, in the case of 2-opt, almost every solution has a unique fitness value. In the case of order crossover and cycle crossover, some solutions have the same fitness value, while in the case of partially matched crossover, each solution also has a unique fitness value.

Figure 41-52. operator.analysis.figure [W/5] show the results of the distance of best solution with filtered global optima, which is similar to the results of the distance of best solution, only here I consider the filtered global optima solution.

Figure 53-64. operator.analysis.figure [W/5] illustrate the results of the distance of best solution and solutions with filtered global optima, which are similar to the results of the distance of best solution and solutions, only the filtered global optima solution is compared.

Figure 65-68. operator.analysis.figure [W/5] illustrate the cost density with filtered global optima values, which differs in that the filtered global optima is also included in the solution set.

The summary results for the random walk are illustrated in *Table 2. fitness.landscape.table [A/12]*. The results of the iterations for the 2-opt operator is illustrated in *Figure 1. operator.analysis.iteration.figure [W/6],* for order crossover operator in *Figure 2. operator.analysis.iteration.figure [W/6]*, for cycle crossover operator in *Figure 3. operator.analysis.iteration.figure [W/6]* and a partially matched crossover operator is illustrated in *Figure 4. operator.analysis.iteration.figure [W/6].*

Operator analysis (Random Walk, Adaptive Walk, Reverse Adaptive Walk, Uphill-Downhill Walk, Neutral Walk, Reverse Neutral Walk)	
Type	**Optimal value**
Fitness values	The greater the difference between the lower and upper bound, the better. And the lower limit should be small because the objective is to minimize fitness value.
Average of fitness distances	The greater the distances, the better the algorithm maps the search space.
Average of Hamming distances	
Average of basic swap sequence distances	
Fitness distances of best solution	Small lower bound and large upper bound. Then the operator explores the space well (large upper bound), but it also found a very good solution because of the small lower bound.
Hamming distances of best solution	
Basic swap sequence distances of best solution	
Cost density	For many, small values, it maps the space well. The goal is to create many different fitness value solutions.
Fitness distance of filtered global optima	Small lower bound and large upper bound. Then the operator explores the space well (large upper bound), but it also found a very good solution because of the small lower bound.
Hamming distance of filtered global optima	
Basic swap sequence distance of filtered global optima	

Table 9.: Operator analysis (with walk): Evaluation strategy

Random walk				
Type	**2-opt**	**OX**	**CX**	**PMX**
Fitness values	✓	✓	✓	✓
Average of fitness distances Average of Hamming distances Average of basic swap sequence distances	✓	✓	✓	✓
Fitness distances of best solution Hamming distances of best solution Basic swap sequence distances of best solution	✓	✓	✓	✓
Cost density	✓	✓	✓	✓
Fitness distance of filtered global optima Hamming distance of filtered global optima Basic swap sequence distance of filtered global optima	✓	✓	✓	✓

Table 10.: Operator analysis (with walk): Random walk analysis: evaluation

Information content analysis

I used the following techniques for information theory analysis: partial information content, expected partial information content, information stability, entropy, information content, density basin information, regularity, evolvability portrait.

In the test the following neighbourhood techniques are analyzed: 2-opt, order crossover (OX), cycle crossover (CX), partially matched crossover (PMX).

Considering the test results, I would like to highlight the following test results: the partial information content is 0.55 for 2-opt, 0.56 for order crossover, 0.47 for cycle crossover and 0.63 for partially matched crossover. Since the partial information content shows a flat landscape for 1 and a rugged landscape for 0, so in my case, I get neither a flat nor a rugged landscape for all neighbourhood techniques. The landscape was the flattest is in the case of a partially matched crossover, so this operator may be the most efficient, while in the case of a cycle crossover it may be the most rugged, so this operator may be the worst.

The measure expected partial information content values are 27 for 2-opt, 28 for OX, 23 for CX and 31 for PMX.

Considering the information stability measure, the following values were given with the test runs: 7111.15 for 2-opt, 5809.92 for OX, 2728.38 for CX and 5696.12 for PMX. According to my test experiments, the 2-opt operator provided the largest change, while CX yields the smallest change values.

The lower the entropy, information content, density basin information values, the better is the optimization surface, because this means that the search is moving in the same direction (increasing or decreasing the fitness value of the neighbour of the current solutions).

The measured entropy values are the followings: for 2-opt 1.5809 for OX 2.5792 for CX 2.9458 and for PMX 2.1070. According to this, the 2-opt has the smallest entropy and the CX has the largest.

The information content values are 0.4101 for 2-opt, 0.8252 for OX, 0.8831 for CX, and 0.6380 for PMX. According to this, the highest value was given by CX and the lowest by 2-opt.

The density basin information values are 0.3285 for 2-opt, 0.2814 for OX, 0.4183 for CX, and 0.2888 for PMX.

The diagram of regularity values for each operator (2-opt, OX, CX, PMX) is shown in *Figure 1-4. random.walk.analysis.figure [W/7]*, which shows ruggedness.

A graph of evolvability portrait values for each operator is shown in *Figure 5-8. random.walk.analysis.figure [W/7]*. It results a rugged function along with the iterations. However, if we plot the values along with fitness, which is shown in *Figure 9-12. random.walk.analysis.figure [W/7]* we get a level result (function) that is independent of the fitness value. The higher the value of evolvability, the better the solution, because the more neighbours of the current solution are better than the current solution, so it is more likely to improve during the iteration. Evolvability measures the probability of improvement.

Information content analysis	
Type	**Optimal value**
Partial Information Content	The higher its value, the better because it means flat space.
Expected Partial Information Content	
Information Stability	The higher its value, the better. The value shows the largest change in fitness of the neighbors, which means that the higher the value for an operator, the more worthwhile it is to use.
Entropy	The lower the value, the better, because it means that the search is moving in the same direction (increasing or decreasing the fitness value of the neighbor of the current solutions).
Information Content	
Density Basin Information	
Regularity	Small solution differences, small jump
Evolvability portrait	The higher the value of evolvability, the better the solution, because the more neighbors of the current solution are better than the current solution, so it is more likely to improve during the iteration. In fact, evolvability measures the probability of improvement.

Table 11.: Operator analysis (with walk): Random walk analysis: Information content analysis: evaluation strategy

Information content analysis				
Type	**2-opt**	**OX**	**CX**	**PMX**
Partial Information Content	✓	✓	x	✓
Expected Partial Information Content	✓	✓	x	✓
Information Stability	✓	✓	x	✓
Entropy	✓	x	x	x
Information Content	✓	x	x	x
Density Basin Information	✓	✓	x	✓
Regularity	✓	✓	x	✓
Evolvability portrait	✓	✓	x	✓

Table 12.: Operator analysis (with walk): Random walk analysis: Information content analysis: evaluation

7.2.1.2. Adaptive walk

Figure 69-71. operator.analysis.figure [W/5] illustrate the results of the 2-opt operator, *Figure 72-74. operator.analysis.figure [W/5]* present the results for the order crossover operator, *Figure 75-77. operator.analysis.figure [W/5]* illustrate the cycle crossover results, *Figure 78-80. operator.analysis.figure [W/5]* show the results of a partially matched crossover, where distances are calculated from the elements of the solution set. According to *Figure 69. operator.analysis.figure [W/5]*, fitness values range from ≈120,000 to ≈150,000 for 2-opt, and the average fitness distances related to solutions range from ≈4,000 to ≈21,000. It is a parabolic function. The *Figure 70. operator.analysis.figure [W/5]* represents the average of the Hamming distances for 2-opt, the higher this value, the higher the average of the distances in general. Here the Hamming distances are between 20 and 34. The same can be said for the basic swap sequence (here the distances are between 13 and 26), this is shown in *Figure 71. operator.analysis.figure [W/5]*. According to *Figure 72. operator.analysis.figure [W/5]*, the analysis also gives a parabola-like function for fitness distances in case of order crossover (here the fitness values are ≈110,000-≈140,000, while the averages of the fitness differences range from ≈2,000 to ≈17,000), while the Hamming (here the averages of the differences are between 16 and 34) and basic swap sequence distances (where the averages of the distances are between 12 and 26) also depend on the fitness value of the solution (*Figure 73-74. operator.analysis.figure [W/5]*). The figures are also similar during a cycle crossover, here the fitness values are between ≈110,000 and ≈130,000, while the averages of the distances for a fitness distance are ≈2,000-≈15,000 (*Figure 75. operator.analysis.figure [W/5]*), while basic swap sequence distance during 8-24 is this value. For a partially matched crossover, fitness values (*Figure 78. operator.analysis.figure [W/5]*) are ≈110,000 – ≈140,000 and fitness differences averages between ≈2,000 and ≈21,000. Hamming distances are between 22 and 36 and also depend on the fitness value, the higher the fitness value the higher the averages of the Hamming distances, and I also got this result in the case of basic swap sequence, here the values are between 16 and 28.

Figure 81-92. operator.analysis.figure [W/5] show the distance from the best solution for an adaptive walk. *Figure 81-83. operator.analysis.figure [W/5]* show the results of the 2-opt operator, *Figure 84-86. operator.analysis.figure [W/5]* show the results of the order crossover, *Figure 87-89. operator.analysis.figure [W/5]* for cycle crossover results, *Figure 90-92. operator.analysis.figure [W/5]* illustrate the results of a partially matched crossover. Basic swap sequence distances also depend on the fitness values of the solutions. The higher the fitness value, the greater the distances from the best solution. The situation is also similar in the case of order crossover, this case is illustrated in *Figure 84-86. operator.analysis.figure [W/5]*. The same can be said for cycle crossover and partially matched crossover, these are shown in *Figure 87-89. operator.analysis.figure [W/5]* and *Figure 90-92. operator.analysis.figure [W/5]*.

Figure 93-104. operator.analysis.figure [W/5] show the results of the distance of the best solution and solutions. *Figure 93-95. operator.analysis.figure [W/5]* illustrate the 2-opt operator results, *Figure 96-98. operator.analysis.figure [W/5]* the order crossover results, *Figure 99-101. operator.analysis.figure [W/5]* show the results of the cycle crossover, while *Figure 102-104. operator.analysis.figure [W/5]* show the results of the partially matched crossover. The fitness distance from the best solution and the average of the fitness distances of the other solutions describe a parabolic function. If the distance from the best solution is small or large, then the average distance from the other solutions is large when using a 2-opt operator (*Figure 93. operator.analysis.figure [W/5]*). There are no similarities between the Hamming and basic swap sequence distances, the distance of the best solution does not affect the average of the distances related to the other solutions (*Figure 94-95. operator.analysis.figure [W/5]*). In the case of order crossover, if the fitness distances from the best solution are too large, the averages of the distances will also be very large (*Figure 96. operator.analysis.figure [W/5]*), and this is also true for Hamming and basic swap sequence distances (*Figure 97-98. operator.analysis.figure [W/5]*). In the case of a cycle crossover, the averages of fitness distances are also high when the fitness distance related to the best solution is high (*Figure 99. operator.analysis.figure [W/5]*). The same is true for Hamming and basic swap sequence distances (*Figure 100-101. operator.analysis.figure [W/5]*). In the case of a partially matched crossover, both fitness, Hamming and basic swap sequence distances related to the best solution also greatly affect the average distances (*Figure 102-104. operator.analysis.figure [W/5]*).

The cost density diagrams are shown in *Figure 105-108. operator.analysis.figure [W/5]*. In the case of 2-opt (*Figure 105. operator.analysis.figure [W/5]*), each solution has a different fitness value, but in the case of order crossover (*Figure 106. operator.analysis.figure [W/5]*) and cycle crossover (*Figure 107. operator.analysis.figure [W/5]*) some solutions have the same fitness value. In the case of a partially matched crossover (*Figure 108. operator.analysis.figure [W/5]*), almost every solution has a unique fitness value.

The distances related to the filtered global optima result are shown in *Figure 109.-120. operator.analysis.figure [W/5]*. As fitness distances first decrease and then increase, depending on the fitness value, this means that the 2-opt (*Figure 109. operator.analysis.figure [W/5]*), order crossover (*Figure 112. operator.analysis.figure [W/5]*), cycle crossover (*Figure 115. operator.analysis.figure [W/5]*) and partially matched crossover (*Figure 118. operator.analysis.figure [W/5]*) solutions were better than the filtered global optima. Hamming distances and basic swap sequence distances do not depend on fitness values for either operator.

Figure 121-132. operator.analysis.figure [W/5] illustrate the average of the distances related to the filtered global optima result and the other solutions. During 2-opt and fitness distances, as the

distance from the best solution increases, so does the average of the distances (*Figure 121. operator.analysis.figure [W/5]*). Hamming and basic swap distances, on the other hand, do not depend on the filtered global optima distance for 2-opt. In the case of order crossover, fitness distances show both increasing and decreasing tendencies, increasing the distance from the filtered global optimum at the same time (*Figure 124. operator.analysis.figure [W/5]*). The Hamming and basic swap sequence distances (*Figure 125-126 operator.analysis.figure [W/5]*) do not depend on the filtered global optima distance for order crossover. In the case of a cycle crossover, fitness distances also show an increasing and decreasing trend, increasing the distance related to the filtered global optimum simultaneously (*Figure 127. operator.analysis.figure [W/5]*). The Hamming and basic swap sequence distances do not depend on the filtered global optima distance for cycle crossover. In the case of a partially matched crossover, the mean of the fitness distances also shows an increasing and decreasing trend as a function of the filtered global optima distance (*Figure 130. operator.analysis.figure [W/5]*), the averages of the Hamming and basic swap sequence distances (*Figure 131-132. operator.analysis.figure [W/5]*) do not depend on the distance of the filtered global optima.
The cost density with filtered global optima (*Figure 133-136. operator.analysis.figure [W/5]*) is similar to the cost density, only the filtered global optima solution has been included in the set of solutions.
The summary results for the adaptive walk technique are illustrated in *Table 3. fitness.landscape.table [A/12]*. The results of the iterations are for operator 2-opt operator are presented in *Figure 5. operator.analysis.iteration.figure [W/6]*, for order crossover operator *Figure 6. operator.analysis.iteration.figure [W/6]*, for cycle crossover operator *Figure 7. operator.analysis.iteration.figure [W/6]* and partially matched crossover operator are illustrated in *Figure 8. operator.analysis.iteration.figure [W/6]*.

<table>
<tr><th colspan="5">Adaptive walk</th></tr>
<tr><th>Type</th><th>2-opt</th><th>OX</th><th>CX</th><th>PMX</th></tr>
<tr><td>Fitness values</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td></tr>
<tr><td>Average of fitness distances</td><td rowspan="3">✓</td><td rowspan="3">✓</td><td rowspan="3">✓</td><td rowspan="3">✓</td></tr>
<tr><td>Average of Hamming distances</td></tr>
<tr><td>Average of basic swap sequence distances</td></tr>
<tr><td>Fitness distances of best solution</td><td rowspan="3">✓</td><td rowspan="3">✓</td><td rowspan="3">✓</td><td rowspan="3">✓</td></tr>
<tr><td>Hamming distances of best solution</td></tr>
<tr><td>Basic swap sequence distances of best solution</td></tr>
<tr><td>Cost density</td><td>✓</td><td>x</td><td>x</td><td>x</td></tr>
<tr><td>Fitness distance of filtered global optima</td><td rowspan="3">✓</td><td rowspan="3">✓</td><td rowspan="3">✓</td><td rowspan="3">✓</td></tr>
<tr><td>Hamming distance of filtered global optima</td></tr>
<tr><td>Basic swap sequence distance of filtered global optima</td></tr>
</table>

Table 13.: Operator analysis (with walk): Adaptive walk analysis: evaluation

7.2.1.3. Reverse adaptive walk

The averages of the distances between the solutions for a 2-opt operator are shown in *Figure 137-139. operator.analysis.figure [W/5]*, order crossover operator in *Figure 140.-142. operator.analysis.figure [W/5]*, cycle crossover operator in *Figure 143-145. operator.analysis.figure [W/5]*, and for a partially matched crossover operator in *Figure 143-145. operator.analysis.figure [W/5]*. Averages of fitness distances (*Figure 137. operator.analysis.figure [W/5]*) describe a parabolic function, Hamming (*Figure 138. operator.analysis.figure [W/5]*) and basic swap sequence (*Figure 139. operator.analysis.figure [W/5]*) averages decrease with increasing fitness value. Order crossover fitness distances (*Figure 140. operator.analysis.figure [W/5]*) also describe a parabolic function, while Hamming (*Figure 141. operator.analysis.figure [W/5]*) and basic swap sequence (*Figure 142. operator.analysis.figure [W/5]*) distances do not depend on fitness values. The situation is similar in the case of the cycle and partially matched crossover, the averages of the difference in fitness values can be represented by a parabolic function, and the distances between Hamming and basic swap sequence do not depend on the fitness value, they are located close to each other.

The distances from the best solution to the 2-opt operator are shown in *Figure 149-151. operator.analysis.figure [W/5]*, in the case of order crossover *Figure 152-154. operator.analysis.figure [W/5]* illustrate, in the case of cycle crossover *Figure 155-157. operator.analysis.figure [W/5]*, while in the case of a partially matched crossover *Figure 158-160. operator.analysis.figure [W/5]*. Fitness distance from best solution in the case of 2-opt (*Figure 149. operator.analysis.figure [W/5]*), order crossover (*Figure 152. operator.analysis.figure [W/5]*), cycle crossover (*Figure 155. operator.analysis.figure [W/5]*) and partially matched crossover (*Figure 158. operator.analysis.figure [W/5]*) also describe a linear function. Hamming and basic swap sequence distances occur in a large interval, from a small distance of 1-2 distances to a very large distance of 30-40 between solutions.

The function of the distance from the best solution and the averages of the distances from the other solutions are presented in the following. The fitness distance for 2-opt (*Figure 161. operator.analysis.figure [W/5]*) is a parabolic function, but shows a prolonged, long-decreasing trend and then starts to increase slightly by the end of the scale. The Hamming (*Figure 162. operator.analysis.figure [W/5]*) and basic swap sequence (*Figure 163. operator.analysis.figure [W/5]*) distances show wider average distances for larger distances. In the case of the order crossover, the fitness distances (*Figure 164. operator.analysis.figure [W/5]*) also describe a parabolic function, only here a larger increase can be observed after a small decrease. For Hamming (*Figure 165. operator.analysis.figure [W/5]*) and basic swap sequence (*Figure 166. operator.analysis.figure [W/5]*) distances, the scale of the average distances increases as the distances from the best solution increase. In the case of the cycle and partially matched crossover, the analysis gives a parabolic function for fitness distance, and the Hamming and basic swap sequence distances increase with increasing distance from the best solution, and the scale of the averages of the distances also increases.

The cost density values are low, which means that the fitness values of the solutions are different during the steps. 1-2 are the cost density for 2-opt (*Figure 173. operator.analysis.figure [W/5]*), 1-4 for order crossover (*Figure 174. operator.analysis.figure [W/5]*). Cycle crossover (*Figure 175. operator.analysis.figure [W/5]*), on the other hand, provided several solutions with the same fitness value, where 1-5 is the number of identical fitness values. Partially matched crossover (*Figure 176. operator.analysis.figure [W/5]*) is 1-3, but here too almost all solutions have different fitness values.

The fitness distance from the filtered global optima solution describes a linear function for all examined operators (2-opt - *Figure 177. operator.analysis.figure [W/5]*-, order crossover - *Figure 180. operator.analysis.figure [W/5]* -, cycle crossover - *Figure 183. operator.analysis.figure [W/5]* - and partially matched crossover - *Figure 186. operator.analysis.figure [W/5]*), which means that the filtered global optima solution proved to be better as the solutions obtained during each step (walk). The Hamming and basic swap sequence distances do not depend on the fitness values, these distances move on approximately the same scale for each operator.

The function of the distances from the best solution and the averages of the distances describes a parabolic function for fitness distance. In the case of 2-opt (*Figure 189. operator.analysis.figure [W/5]*) is an elongated parabola, the values decrease for a long time and then increase slightly at the end of the scale. The Hamming and basic swap sequence distances do not depend on the fitness values here either.

The cost density with filtered optima (*Figure 201-204. operator.analysis.figure [W/5]*) figures are similar to the cost density figure, only here the solution set also includes the filtered optima solution.

The summary table for the reverse adaptive walk technique is illustrated in *Table 4. fitness.landscape.table [A/12]*. The results of the iterations for the 2-opt operator are in *Figure 9. operator.analysis.iteration.figure [W/6]*, for the order crossover operator in *Figure 10. operator.analysis.iteration.figure [W/6]*, for the cycle crossover operator in *Figure 11. operator.analysis.iteration.figure [W/6]* and partially matched crossover operator is illustrated in *Figure 12. operator.analysis.iteration.figure [W/6]*.

<table>
<tr><th colspan="5">Reverse adaptive walk</th></tr>
<tr><th>Type</th><th>2-opt</th><th>OX</th><th>CX</th><th>PMX</th></tr>
<tr><td>Fitness values</td><td>✓</td><td>x</td><td>x</td><td>✓</td></tr>
<tr><td>Average of fitness distances</td><td rowspan="3">✓</td><td rowspan="3">✓</td><td rowspan="3">✓</td><td rowspan="3">✓</td></tr>
<tr><td>Average of Hamming distances</td></tr>
<tr><td>Average of basic swap sequence distances</td></tr>
<tr><td>Fitness distances of best solution</td><td rowspan="3">✓</td><td rowspan="3">✓</td><td rowspan="3">✓</td><td rowspan="3">✓</td></tr>
<tr><td>Hamming distances of best solution</td></tr>
<tr><td>Basic swap sequence distances of best solution</td></tr>
<tr><td>Cost density</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td></tr>
<tr><td>Fitness distance of filtered global optima</td><td rowspan="3">✓</td><td rowspan="3">✓</td><td rowspan="3">✓</td><td rowspan="3">✓</td></tr>
<tr><td>Hamming distance of filtered global optima</td></tr>
<tr><td>Basic swap sequence distance of filtered global optima</td></tr>
</table>

Table 14.: Operator analysis (with walk): Reverse adaptive walk analysis: evaluation

7.2.1.4. Uphill-downhill walk

During the uphill-downhill walk, the average of the distances related to the solutions to the 2-opt operator is shown in *Figure 205-207. operator.analysis.figure [W/5]*, the *Figure 208-210. operator.analysis.figure [W/5]* illustrate order crossover operator, *Figure 211-213. operator.analysis.figure [W/5]* the cycle crossover operator, while a partially matched crossover operator is shown in *Figure 214-216. operator.analysis.figure [W/5]*. Fitness values range from ≈130,000 to ≈150,000, while fitness distances average between ≈2,500 and ≈9,000. This function is parabolic because small and large fitness values have a large average fitness distance, while medium fitness values have a small average distance (*Figure 205. operator.analysis.figure [W/5]*). The averages of Hamming distances do not depend on fitness values. They move in a relatively small interval, and the same can be said for the basic swap sequence distance (*Figure 206-207. operator.analysis.figure [W/5]*). In the case of an order crossover, the average of the distances according to fitness values is shown in *Figure 208-210. operator.analysis.figure [W/5]*. The fitness values of the solutions range from ≈120,000 to ≈140,000. The averages of the fitness distances related to the solutions are large for small and large fitness values, while they are small for average fitness values, so their function is parabolic (*Figure 208. operator.analysis.figure [W/5]*). The Hamming and basic swap sequence distances are condensed into a single point, the averages of the distances do not depend on the fitness values (*Figure 209-210. operator.analysis.figure [W/5]*). Cycle crossover values are also similar to 2-opt and order crossover values, here fitness values range from ≈120,000 to ≈140,000, fitness distances average between ≈2,000 and ≈8,500, Hamming distances range from 28 to 34, and basic swap sequence distances range from 23 to 26 values. Partially matched crossover results are also similar to those of the other three operators.

The distances related to the best solution according to each operator are shown in *Figure 217-228. operator.analysis.figure [W/5]*. For a 2-opt operator, the results are shown in *Figure 217-219. operator.analysis.figure [W/5]*, in the case of order crossover *Figure 220-222. operator.analysis.figure [W/5],* in the case of cycle crossover *Figure 223-225. operator.analysis.figure [W/5]*, while in the case of a partially matched crossover *Figure 226-228. operator.analysis.figure [W/5]*. Fitness distances related to the best solution for the 2-opt operator (*Figure 217. operator.analysis.figure [W/5]*) increase as a function of fitness distance, Hamming (*Figure 218. operator.analysis.figure [W/5]*) and basic swap sequence (*Figure 219.*

operator.analysis.figure [W/5]) distances, on the other hand, show no correlation with fitness values, these distances are in a large range. For order crossover, fitness distances (*Figure 220. operator.analysis.figure [W/5]*) also increase linearly as a function of fitness value, and Hamming (*Figure 221. operator.analysis.figure [W/5]*) and basic swap sequence (*Figure 222. operator.analysis.figure [W/5]*) distances do not correlate with fitness values, and these distances are also in a large interval here. The results of the cycle crossover are similar to the results of 2-opt and order crossover, the fitness distances (*Figure 223. operator.analysis.figure [W/5]*) here are ≈500 - ≈16,000, the Hamming distances (*Figure 224. operator.analysis.figure [W/5]*) are 2-42 and basic swap sequence distances (*Figure 225. operator.analysis.figure [W/5]*) range from 1-34. The situation is similar in the case of a partially matched crossover, here the fitness distances (*Figure 226. operator.analysis.figure [W/5]*) are between ≈0 and ≈14,000, the Hamming distances are (*Figure 227. operator.analysis.figure [W/5]*) 6-40, and the basic swap sequence distances (*Figure 228. operator.analysis.figure [W/5]*) are 4-32.

The distance of the best solution and other solutions for 2-opt are shown in *Figure 229-231. operator.analysis.figure [W/5]*, in the case of order crossover in *Figure 232-234. operator.analysis.figure [W/5]*, in the case of cycle crossover in *Figure 235-237. operator.analysis.figure [W/5]* while in the case of partially matched crossover in *Figure 238-240. operator.analysis.figure [W/5]*. For 2-opt, the fitness distance from the best solution and the average of the fitness distances describe a parabolic function (*Figure 229. operator.analysis.figure [W/5]*). However, the averages of Hamming distances do not depend on the Hamming distances related to the best solution (*Figure 230. operator.analysis.figure [W/5]*). The results are similar for the basic swap sequence (*Figure 231. operator.analysis.figure [W/5]*). For order crossover fitness distances (*Figure 232. operator.analysis.figure [W/5]*) the analysis results a parabolic function, Hamming (*Figure 233. operator.analysis.figure [W/5]*) and basic swap sequence (*Figure 234. operator.analysis.figure [W/5]*) distances do not depend on the distance from the best solution. In the case of cycle crossover and partially matched crossover, similar results are obtained for fitness distance, Hamming distance, and basic swap sequence distance.

The cost density values move at a small interval during each operator, which means that the fitness values are different for each solution. The cost density value for 2-opt (*Figure 241. operator.analysis.figure [W/5]*) is from 1 to 2, for order crossover (*Figure 242. operator.analysis.figure [W/5]*) is from 1 to 3, cycle crossover is (*Figure 243. operator.analysis.figure [W/5]*) from 1 to 3 and in the case of partially matched crossover (*Figure 244. operator.analysis.figure [W/5]*) is from 1 to 3.

The distances from the filtered optima solution for 2-opt are as follows: the fitness value and the distances from the filtered optima solution increase linearly (*Figure 245. operator.analysis.figure [W/5]*), the Hamming distances do not depend on the fitness value (*Figure 246. operator.analysis.figure [W/5]*), as well as basic swap sequence distances from the filtered optima solution (*Figure 247. operator.analysis.figure [W/5]*). In the case of an order crossover, the fitness distance also describes a linear function, and the Hamming and basic swap sequence distances do not depend on the fitness value. Cycle crossover results are also similar to 2-opt and order crossover results. In the case of a partially matched crossover, on the other hand, the fitness distance function (*Figure 254. operator.analysis.figure [W/5]*) is V-shaped, which means that the walk found a better fitness value solution than the filtered global optima solution. However, the distances between Hamming (*Figure 255. operator.analysis.figure [W/5]*) and basic swap sequence (*Figure 256. operator.analysis.figure [W/5]*) do not depend on the fitness value here either.

Regarding the distance from the filtered optima solution and the averages of the distances from the other solutions, the following can be stated. For 2-opt (*Figure 257. operator.analysis.figure [W/5]*) and order crossover (*Figure 260. operator.analysis.figure [W/5]*), the fitness distances describe a parabolic function. Cycle crossover (*Figure 263. operator.analysis.figure [W/5]*) and partially matched crossover (*Figure 266. operator.analysis.figure [W/5]*) do not have a completely parabolic function, because in the case of cycle crossover the average fitness initially increases, then decreases, while in the case of a partially matched crossover, a filtered optima fitness distance has several average distances of the same value. The averages of the Hamming and basic swap sequence distances do not depend on the distance related to the filtered global optima solution. In the case of 2-opt, these distances are more scattered than in the case of order crossover, cycle crossover, and partially matched crossover. The cost density with filtered global optima values (*Figure 269-272. operator.analysis.figure [W/5]*) are similar to the cost density values, only here the filtered global optima values were included in the solution set. The summary results for the uphill-downhill walk technique are illustrated in *Table 5. fitness.landscape.table [A/12]*. The result of the iterations for the 2-opt operator is illustrated in *Figure 13. operator.analysis.iteration.figure [W/6]*, for the order crossover operator is *Figure 14. operator.analysis.iteration.figure [W/6]*, for the cycle crossover operator is *Figure 15. operator.analysis.iteration.figure [W/6]* and partially matched crossover operator is illustrated in *Figure 16. operator.analysis.iteration.figure [W/6]*.

Uphill-downhill walk				
Type	**2-opt**	**OX**	**CX**	**PMX**
Fitness values	✓	✓	✓	x
Average of fitness distances	✓	✓	✓	✓
Average of Hamming distances				
Average of basic swap sequence distances				
Fitness distances of best solution	✓	✓	✓	✓
Hamming distances of best solution				
Basic swap sequence distances of best solution				
Cost density	✓	✓	✓	✓
Fitness distance of filtered global optima	✓	✓	✓	✓
Hamming distance of filtered global optima				
Basic swap sequence distance of filtered global optima				

Table 15.: Operator analysis (with walk): Uphill-downhill walk analysis: evaluation

7.2.1.5. Neutral walk

Since the neutral walk searches for the solutions from the neighbours of the previous solution, during which the fitness distance is as small as possible, the results of this fitness landscape analysis technique show a slightly different result from the previous ones.

Figure 273-275. operator.analysis.figure [W/5] for 2-opt operator, *Figure 276-278. operator.analysis.figure [W/5]* for the order crossover operator, *Figure 279-281. operator.analysis.figure [W/5]* for cycle crossover operator, while *Figure 282-284. operator.analysis.figure [W/5]* for partially matched crossover operator shows the average distances related to the other solutions as a function of fitness values. Fitness values are around 140,000. Average fitness distances are ≈500 - ≈1,900 values (*Figure 273. operator.analysis.figure [W/5]*). Hamming distances (*Figure 274. operator.analysis.figure [W/5]*) range from 23-30, while basic swap sequence distances (*Figure 275. operator.analysis.figure [W/5]*) range from 16-23. *Figure 276.-278. operator.analysis.figure [W/5]* for order crossover shows the average distances from other solutions as a function of fitness values. Fitness values range from ≈120,000 to ≈130,000. The average fitness distances (*Figure 276. operator.analysis.figure [W/5]*) are between ≈1,300 and ≈2,400, the Hamming distances (*Figure 277. operator.analysis.figure [W/5]*) are 24-34 and the basic swap sequence distances (*Figure 278. operator.analysis.figure [W/5]*) between 19-27. For cycle crossover, fitness values are also around 140,000, fitness distances (*Figure 279. operator.analysis.figure [W/5]*) averages ≈400 - ≈950, and Hamming distances (*Figure 280. operator.analysis.figure [W/5]*) averages 12-26, for basic swap sequence (*Figure 281. operator.analysis.figure [W/5]*) 10-20. For a partially matched crossover, fitness values are around 140,000, while fitness distances (*Figure 282. operator.analysis.figure [W/5]*) averages ≈800 - ≈3,600, and Hamming distances (*Figure 283. operator.analysis.figure [W/5]*) are between 30-34 and the basic swap sequence distances (*Figure 284. operator.analysis.figure [W/5]*) are 22-27.

The results of the distance of the best solution are shown in *Figure 285-296. operator.analysis.figure [W/5]*. For a 2-opt operator, the results are shown in *Figure 285-287.*

operator.analysis.figure [W/5], order crossover operator in *Figure 288-290. operator.analysis.figure [W/5]* while for cycle crossover *Figure 291-293. operator.analysis.figure [W/5]* and, in the case of a partially matched crossover *Figure 294-296. operator.analysis.figure [W/5]*. Based on the figures, each distance can move on a fairly wide scale. For 2-opt, fitness distances are ≈0 - ≈3,000, Hamming distances are 2-38, and basic swap sequence distances are 1-32. In the case of order crossover (*Figure 288-290. operator.analysis.figure [W/5]*), the fitness distances are ≈200 - ≈4,800, the Hamming distances are 12-38, in the case of the basic swap sequence 10-30. In the case of cycle crossover (*Figure 291-293. operator.analysis.figure [W/5]*), the averages of the fitness distances are about 50-1,700, the Hamming distances are 10-28, and the basic swap sequence is 9-22. For partially matched crossover (*Figure 294-296. operator.analysis.figure [W/5]*), fitness distances are ≈0 - ≈5,500, Hamming distances are 2-40, and basic swap sequence is 1-34.

The averages of the distance from the best solution and the distances related to the other solutions are shown in *Figure 297-308. operator.analysis.figure [W/5]*. For a 2-opt operator, the results are shown in *Figure 297-299. operator.analysis.figure [W/5]. Figure 300-302. operator.analysis.figure [W/5]* show the results of the order crossover, *Figure 303-305. operator.analysis.figure [W/5]* shows the results of the cycle crossover, *Figure 306-308. operator.analysis.figure [W/5]* illustrate the results of the partially matched crossover. For fitness distances for all operators, i.e. 2-opt (*Figure 297. operator.analysis.figure [W/5]*), order crossover (*Figure 300. operator.analysis.figure [W/5]*), cycle crossover (*Figure 303. operator.analysis.figure [W/5]*), partially matched crossover (*Figure 306. operator.analysis.figure [W/5]*) operator, the averages of the Hamming and basic swap sequence distances do not depend on the distance related to the best solution, 2-opt and in the case of an order crossover they are close to each other, while in the case of a cycle crossover they are scattered in the coordinate system and in the case of a partially matched crossover they are also more scattered.

Based on the cost density diagrams, it can be stated that in the case of 2-opt (*Figure 309. operator.analysis.figure [W/5]*) almost every solution has a unique fitness value, while in the case of order crossover (*Figure 310. operator.analysis.figure [W/5]*), almost every fitness value has 2-3 solutions. In the case of cycle crossover (*Figure 311. operator.analysis.figure [W/5]*), some fitness values come with a relatively large number (10-20) of solutions. In the case of a partially matched crossover (*Figure 312. operator.analysis.figure [W/5]*), almost every solution has a unique fitness value.

The distances from the filtered global optima solution (*Figure 313-324. operator.analysis.figure [W/5]*) are condensed into one place in the coordinate system. For 2-opt, the averages of fitness distances (*Figure 313. operator.analysis.figure [W/5]*) range from ≈22,000 to ≈26,000, for Hamming distances (*Figure 314. operator.analysis.figure [W/5]*) 28-38, for basic swap sequence (*Figure 315. operator.analysis.figure [W/5]*) 20-32. In case of order crossover, the fitness distances (*Figure 316. operator.analysis.figure [W/5]*) are between ≈4,000 - ≈9,000, Hamming distances (*Figure 317. operator.analysis.figure [W/5]*) are 32-40, the basic swap sequence distances (*Figure 318. operator.analysis.figure [W/5]*) are 24-34. The values of the distances range between ≈20,000 - ≈22,000 for cycle crossover and fitness distances, 36-40 for Hamming distances, and 28-32 for basic swap sequences. In the case of partially matched crossover and fitness distance the values are between ≈18,000 - ≈23,000, Hamming distance values are 30-40, basic swap sequence values are 24-34.

As a function of the distances related to the filtered global optima result and the averages of the distances (*Figure 325-336. operator.analysis.figure [W/5]*) for fitness distance for each operator, so 2-opt (*Figure 325. operator.analysis.figure [W/5]*), order crossover (*Figure 328. operator.analysis.figure [W/5]*), cycle crossover (*Figure 331. operator.analysis.figure [W/5]*), and partially matched crossover (*Figure 334. operator.analysis.figure [W/5]*) describes a parabolic function. For Hamming distances for each operator, the distance from the filtered global optima result does not affect the averages of the distances for either 2-opt (*Figure 326. operator.analysis.figure [W/5]*) or order crossover (*Figure 329. operator.analysis.figure [W/5]*), nor for cycle crossover (*Figure 332. operator.analysis.figure [W/5]*) or partially matched crossover

(*Figure 335. operator.analysis.figure [W/5]*). The situation is similar to basic swap sequence distances.
The cost density with filtered global optima results (*Figure 337-340. operator.analysis.figure [W/5]*) are the same as the cost density results, they only include the filtered global optima solution.
For the neutral walk technique, the summary table is shown in *Table 6. fitness.landscape.table [A/12]*. The results of the iterations for the 2-opt operator are *Figure 17. operator.analysis.iteration.figure [W/6]*, for the order crossover operator *Figure 18. operator.analysis.iteration.figure [W/6]*, for the cycle crossover operator *Figure 19. operator.analysis.iteration.figure [W/6]* and partially matched crossover operator is illustrated in *Figure 20. operator.analysis.iteration.figure [W/6]*.

Neutral walk				
Type	**2-opt**	**OX**	**CX**	**PMX**
Fitness values	x	✓	x	x
Average of fitness distances	x	✓	x	x
Average of Hamming distances				
Average of basic swap sequence distances				
Fitness distances of best solution	✓	✓	✓	✓
Hamming distances of best solution				
Basic swap sequence distances of best solution				
Cost density	✓	x	x	✓
Fitness distance of filtered global optima	✓	✓	✓	✓
Hamming distance of filtered global optima				
Basic swap sequence distance of filtered global optima				

Table 16.: Operator analysis (with walk): Neutral walk analysis: evaluation

7.2.1.6. Reverse neutral walk

The reverse neutral walk is the opposite of the neutral walk, it selects the neighbour from the neighbours of the current solution that is the furthest (the difference in fitness value is the largest) from its current solution. The results of the distance of solutions (*Figure 341-352. operator.analysis.figure [W/5]*) represent the average of the distances related to the other solutions as a function of the fitness value. The fitness distances describe a parabolic function for each operator (2-opt, order crossover, cycle crossover, partially matched crossover), so for small and high fitness values, the average of the distances from the other solutions is high, while for medium fitness the average of the fitness distances is small. In the case of Hamming and basic swap sequence distances, the averages of the distances do not depend on the fitness value, they are all concentrated in one node in each case, so the average basic swap sequence distance of almost all solutions is the same.
The results of the distance of the best solution are illustrated in *Figure 353-364 operator.analysis.figure [W/5]*. For the 2-opt operator, the *Figure 353-355 operator.analysis.figure [W/5],* for order crossover operator the *Figure 356-358 operator.analysis.figure [W/5],* for cycle crossover operator the *Figure 359-361 operator.analysis.figure [W/5]*, while for a partially matched crossover operator it is shown in *Figure 362-364 operator.analysis.figure [W/5]*. For fitness distance, the analysis results linearly increasing functions for all 2-opt (*Figure 353. operator.analysis.figure [W/5]*), order crossover (*Figure 356. operator.analysis.figure [W/5]*), cycle crossover (*Figure 359. operator.analysis.figure [W/5]*) and partially matched crossover (*Figure 362. operator.analysis.figure [W/5]*). At first, the fitness distance from the best solution is very small, but the difference between high fitness value solutions and the best solution is already large. Hamming and basic swap sequence distances no longer depend on the fitness value, but these distances also move on a wide scale, not condensing into one place in the coordinate system. For example, for the 2-opt and Hamming distance the distance values are between 4-40 (*Figure 354. operator.analysis.figure [W/5]*), for the 2-opt and basic swap sequence 2-34 (*Figure 355.*

operator.analysis.figure [W/5]), 6-40 for Hamming distance (*Figure 357. operator.analysis.figure [W/5]*), and 6-34 for basic swap sequence (*Figure 358. operator.analysis.figure [W/5]*).

The results of the distance of the best solution and solutions (*Figure 365-376. operator.analysis.figure [W/5]*) illustrate the distance from the best solution and the averages of the distances from the other solutions. For a 2-opt operator the *Figure 365-367. operator.analysis.figure [W/5]*, in the case of order crossover the *Figure 368-370. operator.analysis.figure [W/5]*, in the case of cycle crossover the *Figure 371-373. operator.analysis.figure [W/5]*, while for partially matched crossover the *Figure 374-376. operator.analysis.figure [W/5]* show the results. Fitness distances (average distance as a function of distance from the best solution) describe a parabolic function for each operator (*Figure 365. operator.analysis.figure [W/5], Figure 368. operator.analysis.figure [W/5], Figure 371. operator.analysis.figure [W/5], Figure 374. operator.analysis.figure [W/5]*). For Hamming and basic swap sequence distances, the points are scattered in the coordinate system, and the average of the distances from the solutions does not depend on the distance from the best solution.

Cost density values for 2-opt (*Figure 377. operator.analysis.figure [W/5]*) indicate that some solutions have the same fitness value. The situation is similar for other operators (order crossover, cycle crossover, partially matched crossover).

The distances related to the filtered global optima solution as a function of fitness values are shown in *Figure 381-392. operator.analysis.figure [W/5]*. Fitness distances show a linearly increasing function for 2-opt (*Figure 381. operator.analysis.figure [W/5]*). Order crossover (*Figure 384. operator.analysis.figure [W/5]*) and partially matched crossover (*Figure 390. operator.analysis.figure [W/5]*) describe a parabolic function, while cycle crossover (*Figure 387. operator.analysis.figure [W/5]*) is also a linearly increasing function. The Hamming and basic swap sequence distances are relatively concentrated in one place in the case of 2-opt, and even closer to each other in the coordinate system in the case of order crossover. In the case of a cycle crossover and a partially matched crossover, the points are also close.

The distances from the filtered global optima solution and the averages of the distances related to the solutions are illustrated for different operators and distances in *Figure 393-404. operator.analysis.figure [W/5]*. Regarding fitness distances, the values describe a parabolic function for each operator (2-opt, order crossover, cycle crossover, partially matched crossover). The Hamming and basic swap sequence distances are close to each other and do not affect the average distances from the filtered global optima solution. For the reverse neutral walk technique, the summary table is illustrated in *Table 7. fitness.landscape.table [A/12]*. The results of the iterations for the 2-opt operator are *Figure 21. operator.analysis.iteration.figure [W/6]*, for the order crossover operator are *Figure 22. operator.analysis.iteration.figure [W/6]*, for the cycle crossover operator are *Figure 23. operator.analysis.iteration.figure [W/6]* and partially matched crossover operator is illustrated in *Figure 24. operator.analysis.iteration.figure [W/6]*.

Reverse neutral walk				
Type	**2-opt**	**OX**	**CX**	**PMX**
Fitness values	✓	✓	✓	✓
Average of fitness distances	✓	✓	✓	✓
Average of Hamming distances				
Average of basic swap sequence distances				
Fitness distances of best solution	✓	✓	✓	✓
Hamming distances of best solution				
Basic swap sequence distances of best solution				
Cost density	✓	✓	✓	✓
Fitness distance of filtered global optima	✓	✓	✓	✓
Hamming distance of filtered global optima				
Basic swap sequence distance of filtered global optima				

Table 17.: Operator analysis (with walk): Reverse neutral walk analysis: evaluation

7.3. Scattered sampling

7.3.1. Analysis of the filtered search space

This means that the search space is sampled. So I randomly generate solutions and analyze them. I produced 100 solutions at random. The following analysis methods were considered: distance of solutions, the distance of the best solution, the distance of the best solution and solutions, cost density.

Figure 1. filtered.search.space.analysis.figure [W/8] shows fitness values and average fitness distances. Fitness values range from ≈120,000 to ≈160,000, while average fitness distances range from ≈5,000 to ≈19,000. These values also show a parabolic function because small and large fitness values are paired with a large average distance, while medium fitness values are associated with a medium average distance. *Figure 2. filtered.search.space.analysis.figure [W/8]* shows fitness values and average Hamming distances. The average Hamming distances in each case range from about 35 to 36. *Figure 3. filtered.search.space.analysis.figure [W/8]* shows fitness values and average basic swap sequence distances. The average basic swap sequence distances in each case range from about 28-29. *Figure 4. filtered.search.space.analysis.figure [W/8]* illustrates fitness values and fitness distances from the best solution (the best of the generated solutions). The values show a linear function, the higher the fitness value of the solution, the greater the distance from the best solution (the best of the generated solutions). Fitness values range from ≈120,000 to ≈160,000, and fitness distances range from ≈4,000 to ≈36,000. *Figure 5. filtered.search.space.analysis.figure [W/8]* shows fitness values and Hamming distance from the best solution (the best of the generated solutions). Hamming distances range from 30 to 40. *Figure 6. filtered.search.space.analysis.figure [W/8]* shows the fitness values and the basic swap sequence distance from the best solution (the best of the generated solutions). Basic swap sequence distances range from 24 to 35. *Figure 7. filtered.search.space.analysis.figure [W/8]* shows the fitness distance from the best solution (the best of the generated solutions) and the average of the fitness distances from the other solutions. The values are described by a parabolic function, thus, low and high distances mean high average distances from the best (best of generated solutions) solution, while medium distances mean low average distances from the best (best of generated solutions) solution. *Figure 10. filtered.search.space.analysis.figure [W/8]* shows that almost every solution has a different fitness value.

The summary table is illustrated in *Table 20. fitness.landscape.table [A/12]*.

Analysis of filtered search space	
Type	**Optimal value**
Fitness values	The greater the difference between the lower and upper bound, the better. And the lower limit bound be small because the objective is to minimize fitness value.
Average of fitness distances	The greater the distances, the better the algorithm maps the search space.
Average of Hamming distances	
Average of basic swap sequence distances	
Fitness distances of best solution	Small lower bound and large upper bound. Then the operator explores the space well (large upper bound), but it also found a very good solution because of the small lower bound.
Hamming distances of best solution	
Basic swap sequence distances of best solution	
Cost density	For many, small values, it maps space well. The goal is to create many different fitness value solutions.

Table 18.: Analysis of filtered search space evaluation strategy

Analysis of filtered search space	
Type	**Value**
Fitness values	✓
Average of fitness distances	✓
Average of Hamming distances	
Average of basic swap sequence distances	
Fitness distances of best solution	✓
Hamming distances of best solution	
Basic swap sequence distances of best solution	
Cost density	✓

Table 19.: Analysis of filtered search space evaluation

7.3.2. Fitness Cloud

During the creation of the fitness cloud, I randomly selected solutions and then created their neighbours. The representation of the solutions and the neighbours, their relation to them is important in the analysis. The following analysis techniques were used: Fitness Cloud, Fc-max, Fc-min, Fc-mean, Strictly Advantageous Count, Average Advantageous Count, Average Deleterious Count, Strictly Deleterious Count.

The following operators are analyzed: 2-opt, Order Crossover (OX), Cycle Crossover (CX), Partially Matched Crossover (PMX).

Figure 1. fitness.cloud.figure [W/9] illustrates the fitness cloud for the 2-opt operator. Fitness values range from ≈100,000 to ≈150,000. The fitness values of the neighbours of solutions with lower fitness values are also lower, while the fitness values of the neighbours of solutions with higher fitness values are also higher

Figure 2. fitness.cloud.figure [W/9] illustrates the fitness cloud for the OX operator. Fitness values range from ≈120,000 to ≈150,000. Here, too, it is typical that the fitness values of the neighbours of solutions with lower fitness values are also lower, while the fitness values of the neighbours of solutions with higher fitness values are also higher.

Figure 3. fitness.cloud.figure [W/9] illustrates the fitness cloud for the CX operator. Fitness values range from ≈110,000 to ≈160,000. Here, too, it is typical that the fitness values of the neighbours of solutions with lower fitness values are also lower, while the fitness values of the neighbours of solutions with higher fitness values are also higher.

Figure 4. fitness.cloud.figure [W/9] illustrates the fitness cloud for the PMX operator. Fitness values range from ≈120,000 to ≈160,000. Here, too, it is typical that the fitness values of the neighbours of solutions with lower fitness values are also lower, while the fitness values of the neighbours of solutions with higher fitness values are also higher.

The FC-max values for each operator (2-opt, OX, CX, PMX) are shown in *Figure 5-8. fitness.cloud.figure [W/9]*. Here again, FC-max values are lower for solutions with lower fitness values, while FC-max values are also higher for solutions with higher fitness values. For 2-opt, the FC-max values range from ≈110,000 to ≈150,000, for OX ≈120,000 to ≈150,000, for CX ≈110,000 to ≈150,000, and for PMX ≈120,000 to ≈150,000.

The FC-mean values for each operator (2-opt, OX, CX, PMX) are shown in *Figure 9.-12. fitness.cloud.figure [W/9]*. Here, too, it is typical that the FC-mean values are lower for solutions with lower fitness values, while the FC-mean values are also higher for solutions with higher fitness values. For 2-opt, FC-mean values range from ≈110,000 to ≈150,000, for OX between ≈120,000 and ≈150,000, for CX ≈120,000 - ≈160,000, and for PMX ≈120,000 - ≈150,000.

The FC-min values for each operator (2-opt, OX, CX, PMX) are shown in *Figure 13-16. fitness.cloud.figure [W/9]*. Here, too, it is typical that the FC-min values are lower for solutions with lower fitness values, while the FC-min values are also higher for solutions with higher fitness values. ≈110,000 - ≈150,000 for 2-opt, ≈120,000 - ≈150,000 for OX, ≈110,000 - ≈150,000 for CX, and ≈120,000 - ≈150,000 for PMX.

According to *Figure 17. fitness.cloud.figure [W/9]*, the 2-opt operator has a strictly advantageous count of 3, an average advantageous count of 97 and a strictly deleterious count of 0. This means that the fitness values of the neighbours are lower, as the fitness value of the given solution, which proves the goodness of the operator since our task is a minimization problem.
According to *Figure 18. fitness.cloud.figure [W/9]*, the OX operator has a strictly advantageous count of 7, an average advantageous count of 93 and a strictly deleterious count of 0. This means that the fitness values of the neighbours are less than the fitness value of the given solution, which proves the goodness of the operator since our task is a minimization problem.
According to *Figure 19. fitness.cloud.figure [W/9]*, the CX operator has a strictly advantageous count of 2, an average advantageous count of 98 and a strictly deleterious count of 0. This means that the fitness values of the neighbours are less than the fitness value of the given solution, which proves the goodness of the operator since our task is a minimization problem.
According to *Figure 20. fitness.cloud.figure [W/9]*, the CX operator has a strictly advantageous count of 5, an average advantageous count of 95 and a strictly deleterious count of 0. This means that the fitness values of the neighbours are less than the fitness value of the given solution, which proves the goodness of the operator since our task is a minimization problem.

Fitness cloud	
Type	**Optimal value**
Fitness Cloud	Low value.
Fc-max	
Fc-mean	
Fc-min	Great value
Strictly Deleterious Count	Low value. Then there are few neighbor solutions whose fitness values are worse than the basic solution.
Average Deleterious Count	
Average Advantageous Count	High value. Then, on average, the fitness values of the neighboring solution are better than those of the basic solution.
Strictly Advantageous Count	

Table 20.: Fitness cloud: evaluation strategy

Fitness cloud				
Type	**2- opt**	**OX**	**CX**	**PMX**
Fitness Cloud	✓	✓	✓	✓
Fc-max	✓	x	✓	x
Fc-mean	✓	x	x	x
Fc-min	x	✓	x	✓
Strictly Deleterious Count	✓	✓	✓	✓
Average Deleterious Count	✓	✓	✓	✓
Average Advantageous Count	✓	✓	✓	✓
Strictly Advantageous Count	✓	✓	✓	✓

Table 21.: Fitness cloud: evaluation

7.4. Summary analyzes

In this subsection, I summarize the results of my measurement and give general conclusions about which multi-objective optimization techniques, which heuristic algorithms, and which operators are worth using. Also, I show which multi-objective optimization techniques, which heuristic algorithms, and which operators are not worth using.
In *Table 22.*, I summarize which are the effective multi-objective optimization techniques and which are the effective heuristics in my measurement. Also, very weak multi-objective optimization techniques and weak heuristics are included in the table. The results show that the simulated annealing and tabu search algorithms provided the best results.

Analysis of multi-objective optimization technique	
Effective multi-objective optimization technique	WGCM, WPM, WESM
Effective heuristics	SA_FI, SA_G, SA_NI, SA_NN, TS_AI, TS_FI, TS_G, TS_NN, FCHC_NN, AS_C, RBVAS_C
Weak multi-objective optimization technique	WSM, PR
Weak heuristics	FA, FA_C, HS, HS_C

Table 22.: Analysis of multi-objective optimization technique

Table 23. shows which heuristics are effective and which are weak heuristics. Efficient iterative heuristic algorithms are effective when the solutions are variable, so space is mapped well. Fitness values vary, with averages of distances between solutions. The results show that the simulated annealing and tabu search and ant algorithms provided the best results.

Iterative heuristic analysis	
Effective iterative heuristic algorithm	ACS, AS, ESAS, FCHC, GA, MMAS, RBVAS, SA, TS
Weak iterative heuristic algorithm	FA, HS

Table 23.: Iterative heuristic analysis

Table 24. shows the analysis of operators with walk techniques. In analyzing each walk operator, I concluded that the 2-opt and partially matched crossover (PMX) operators are much more efficient than the cycle crossover (CX) and order crossover (OX) operators.

Operator analysis (with walks)		
	Effective operator	**Weak operator**
Random walk	2-opt, OX, PMX	
Adaptive walk	2-opt	CX, OX
Reverse adaptive walk	2-opt	CX
Uphill-downhill walk		
Neutral walk	PMX	CX, OX
Reverse neutral walk	PMX, 2-opt	

Table 24.: Operator analysis (with walks)

In the case of random walk, the average difference in fitness values of the solutions received by the operators is almost the same, the Hamming and basic swap sequence distances are very similar, in the case of order crossover and partially matched crossover the distances are slightly larger than in 2-opt and cycle crossover. Cost density values are from 1 to 2, so fitness values are also unique. In the case of a cycle crossover, the cost density diagram shows that several solutions have the same fitness value, although the number of solutions with the same fitness value is only between 1 and 3. The distances from the best solution also move in large intervals, which also means that space is well mapped by these operators.

In the case of an adaptive walk and the case of a cycle crossover result the smallest average fitness distance. The average Hamming and basic swap sequence distances are also the smallest here. The cost density value became the highest in the case of cycle crossover, which means that the analysis got several solutions with the same fitness value during the walk. According to the order crossover cost density diagram, several solutions also have the same fitness value. The cost density values are the lowest for 2-opt, so this operator produced the most varied solutions. Since the objective is to map the search space as well as possible, the 2-opt operator proved to be the best in this measurement experiment.

The reverse adaptive walk analysis has the greatest distance between the solutions of the 2-opt operator. The Hamming and basic swap sequence distances are approximately the same for each operator. Cost density values are low for all operators, but lowest for 2-opt. In the case of a cycle crossover, several solutions also have the same fitness value.
The results of the uphill-downhill walk are nearly the same for each operator. The cost density values are small for all operators, and the figures also show that I obtained varied results using the operators.
During the neutral walk, I got the highest value of the average distances of the fitness values in the case of the partially matched crossover. Mean Hamming and basic swap sequence distances were greatest for the partially matched crossover operator. This means that this operator has produced the most varied solutions. Order crossover distance values are also high but cycle crossover and 2-opt fitness distances are low. Based on the cost density diagrams, the cycle crossover solutions are the most unchanged, but the order crossover solutions are also very unchanged. The 2-opt and partially matched crossover solutions are varied, with almost every solution having a unique fitness value.
For the reverse neutral walk measurement, the 2-opt operator gave the highest average fitness distances, while the Hamming and basic swap sequence distances were higher than the other operators. High fitness value differences also resulted in the partially matched crossover, where Hamming and basic swap sequence distances were also high. Here, however, the cost density value was slightly higher, with several fitness values having the same fitness value. According to the cost density diagrams, all operators produced different solutions, the fitness values of each solution are different.

Table 25. analyzes the efficiency of the operators (2-opt, order crossover, cycle crossover, partially matched crossover) from the information content analysis perspective. The table shows that 2-opt and partially matched crossover (PMX) are more effective than cycle crossover (CX) and order crossover (OX).

Information content analysis		
	Efficient operator	**Weak operator**
Partial Information Content	PMX	CX
Expected Partial Information Content	PMX	CX
Information Stability	2-opt	CX
Entropy	2-opt	CX
Information Content	2-opt	CX
Density Basin Information	OX	CX
Regularity	2-opt	CX
Evolvability portrait	2-opt, PMX	CX, OX

Table 25.: Information content analysis

The partial information content value was the highest for the partially matched crossover, which means a flat landscape, so this technique proved to be the best during the measurement. In the case of the cycle crossover, the partial information content value is the lowest, so this technique proved to be the worst during the measurement.
The expected partial information content value is also the highest for the partially matched crossover, so this technique proved to be the best.
The information stability value is highest for 2-opt and lowest for cycle crossover. The value shows the largest change in fitness of the neighbours, which means that the higher the value for an operator, the more worthwhile it is to use. In the case of order crossover and partially matched crossover, the value between 2-opt and cycle crossover was obtained, which is almost the same for the two operators.

The lower the entropy, information content, density basin information values, the better the landscape is because this means that the search is moving in the same direction (increasing or decreasing the fitness value of the neighbour of the current solutions).
2-opt has the lowest entropy and CX has the largest. The order crossover and partially matched crossover values are nearly the same, between 2-opt and order crossover.
The 2-opt has the lowest information content value and the cycle crossover has the largest. The order crossover and partially matched crossover values are nearly the same, between 2-opt and order crossover operators.
Of the density basin information values, CX gave the highest value and OX the lowest.
According to the regularity diagram, the difference between the solutions of the cycle crossover is the largest. In the case of 2-opt, the jumps are the smallest.
Evolvability values are much higher on average for 2-opt and partially matched crossover than for cycle crossover and order crossover, so it is better to use 2-opt and partially matched crossover operators. The higher the value of evolvability, the better the solution, because the more neighbours of the current solution are better than the current solution, so it is more likely to improve during the iteration. Evolvability measures the probability of improvement.
For the fitness cloud measurement, I summarize the results in *Table 26*. According to the table, the 2-opt operator became efficient, the partially matched crossover operator according to this measurement did not become as efficient as the other operators (order crossover, partially matched crossover).

Fitness Cloud		
	Efficient operator	**Weak operator**
Fitness Cloud	2-opt	PMX
Fc-max	2-opt	PMX
Fc-mean	2-opt	PMX
Fc-min	2-opt	PMX
Strictly Deleterious Count	2-opt, OX, CX, PMX	
Average Deleterious Count	2-opt, OX, CX, PMX	
Average Advantageous Count	2-opt, OX, CX, PMX	
Strictly Advantageous Count	2-opt, OX, CX, PMX	

Table 26.: Fitness Cloud

According to the fitness cloud, 2-opt fitness values are the lowest values and partially matched crossover values are the highest.
Typically, solutions with lower fitness values also have lower FC-max values, while solutions with higher fitness values also have higher FC-max values. For solutions with a lower fitness value, the FC-mean values are also lower, while for solutions with a higher fitness value, the FC-mean values are also higher. For solutions with lower fitness values, the FC-min values are also lower, while for solutions with higher fitness values, the FC-min values are also higher. According to them, the 2-opt operator became the best based on these measurements, the partially matched operator the worst.
The strictly deleterious count, average deleterious cont, average advantageous count, and strictly advantageous count values are nearly the same, the average advantageous count values are high for all operators, so each operator is efficient.

Based on the measurements, I made the following summary evaluation: Efficient multi-objective optimization techniques are the following: WGCM, WPM, WESM. Efficient heuristics are the following algorithms: SA, TS, FCHC, GA, RBVAS. During the measurements, I also observed that the improvement of the solutions provided by the construction algorithms proved to be more effective than the improvement of the purely randomly generated solutions. During the measurements, I also determined the efficient operators, which are as follows: 2-opt, partially matched crossover (PMX).

Thesis 4.:

I analytically analyzed the search space belonging to the optimization task of the generalized model of the Vehicle Routing Problem and I made a detailed analysis. In the search space study, I first analyzed the efficiency of multi-objective optimization techniques and heuristic algorithms. After I analyzed the efficiency of the operators (2-opt, cycle crossover, order crossover, and partially matched crossover). During the analysis of the operators I used different walk techniques like the random walk, adaptive walk, reverse adaptive walk, uphill-downhill walk, neutral walk and the reverse neutral walk. I also analyzed the search space based on information theory, which is also used to analyze the efficiency of operators. For the analysis of the operators, I also prepared the fitness cloud analysis.

Based on the examined sample, it can be concluded that the following multi-objective optimization techniques are effective: Weighted Global Criterium Method (WGCM), Weighted Product Method (WPM), Weighted Exponential Sum Method (WESM). Based on the performed analyzes, it can be concluded that the following heuristics are effective for Vehicle Routing Problems: Simulated Annealing (SA), Tabu Search (TS), First Choice Hill Climbing (FCHC), Genetic Algorithm (GA), Rank Based Version of Ant System (RBVAS). During the measurements, I also observed that the improvement of the solutions provided by the construction algorithms proved to be more effective than the improvement of the purely randomly generated solutions. In the measurements, I also found that the 2-opt and partially matched crossover operators are more efficient than the order crossover and cycle crossover operators.

Related publications: [P/3], [P/4], [P/7], [P/9], [P/11], [P/12], [P/13], [P/18].

8. The applications of the generalized model of the Vehicle Routing Problem

In this chapter, case studies are presented to demonstrate the practical applicability of the generalized model for optimization design of the Vehicle Routing Problem.
I first examined what transport case studies were presented in the literature for vehicle routing tasks. The results are summarized in *Table 27*.

Transportation problem	Publication	Transportation problem	Publication
Waste collection	[115,116]	Delivery of beverages	[116,132]
Newspaper delivery	[117,118,119,116]	Food delivery	[116]
Electrical wholesaler distribution system	[120]	Tank transport	[133]
Perishable food delivery	[121,122,123,124]	Remittances	[134]
Frozen food	[125]	Parcels	[135]
Cash delivery	[126,127]	Patient transport	[136,137]
In - plant delivery	[128]	Maintenance	[138,139, 140,141]
Delivery of bakery products	[127,130,131,116]		

Table 27.: Case studies for the Vehicle Routing Problems

Based on the study, it can be said that a wide range of transportation tasks can be solved as a special Vehicle Routing Problem.

8.1. Transport of bakery products

By transporting bakery products, I mean that bread and pastries (non-durable baked products) are delivered from bakeries to shops. The characteristic of this delivery task is that in the case of the tiered system (depot-satellite-customer), only one depot and several customers (approx. [from 5 to 20 locations]) are distinguished. There is no need for satellites because the bakeries are located near the shops.
Among the relations between the nodes, the travel time between the locations (approx. [from 5 to 10 minutes]), the travel distance between the locations (approx. [from 1 to 5 km]), and the route quality between the locations can be taken into account. There are several types of vehicles that can deliver bakery product, but the companies usually have a single type of vehicle. Regarding the attributes, the following components can be taken into account: the capacity constraints of the vehicles (approx. [from 30 to 200 pieces of bakery product]), and fuel consumption of the vehicles (approx. [from 10 to 25 liter/100 km]). The status of the vehicle is own or rented vehicle and the rental fee component can also be taken into account.
Among the temporality attributes, the service time at the locations is 0, because no service is performed. When packing time at the location is taken into account, it only makes sense in the depot (bread, bakery) (approx. [from 2 to 5 minutes]). The loading time at the locations only makes sense in the depot (approx. [from 2 to 5 minutes]). Unloading time of the location makes sense for customers (approx. [from 2 to 5 minutes]). Product downtime at the locations is not taken into account, there is no product downtime. The administration time at the locations can be considered for the nodes (approx. [from 2 to 5 minutes]), as like the quality control time at the locations (approx. [from 2 to 5 minutes]). Fuel filling time (approx. [from 5 to 10 minutes]) can also be considered, but this is not a significant factor during transport. The time window, on the other hand, is important because both the depot (approx. [from 3 to 15 hour] in Hungary) and the shops (customers) (approx. [from 4 to 20 hour] in Hungary) have opening hours. There can be multiple time windows, as the store (customers) may not accept goods at all times (approx. [from 4 to 11 hour] [from 13 to 20 hour] in Hungary).

Among the attributes of products and services, the capacity constraint of each location per product type, their demand for products by product type (approx. [from 20 to 300 pcs]) can be taken into account. The storage level at the locations may be worth considering. It is also worth considering the prices of the product (approx. [from 300 to 3000 HUF/ pcs] in Hungary).
There is no inter-depot route among the operational parameter, only a single depot can be defined. Delivery, of course, appears as an operational parameter, but pickup can also appear because any leftovers can be returned to the plant (where, for example, bread crumbs are made from them). The soft time window can also be used.
Among the metrics, minimization of the length of the route is an important factor and the maximization of profit is the most important factor. The following parameters can be also taken into account: minimization of loading costs, minimization of administrative costs, minimization of quality control cost, minimization of fuel consumption, minimization of rental fee can be taken into account. Route quality maximization (minimization of the depreciation of the vehicle) can be also important. Time minimization can be also an important metric, it can contain the followings: minimization of route time, minimization of packaging time, minimization of loading time, minimization of administrative time, minimization of quality control time, minimization of waiting time for nodes. Among the penalty points, the following can be taken into account: exceeding the number of vehicles.
Each component depends on the temporality (period). The base unit of the period is seven. Stores should be served 7 times a week. The attributes may be different during each period, e.g. the store's bakery demands will be different on weekdays than on weekends, and opening hours will be different.
I found the following publication for this type of transportation problem: [127,130,131,116].

8.2. Transport of short-term food

It is very similar to the delivery of baked products (Chapter 8.1.), it is characterized by the same attributes, the only difference is that here it is not frequent the pick-up, only delivery, and the period time can be given a higher value e.g. the products are delivered weekly. Fruits and vegetables can be a pick-up if they have remained in the supermarkets.
I found the following publication for this type of transportation problem: [121,122,123,124].

8.3. Transport of refrigerated products (e.g. dairy products, meat)

They are very similar to short-term foods (Chapter 8.2.), but here only a special vehicle must transport the products. Furthermore, here the period time is small (e.g. daily transport) because they are perishable products.
I found the following publication for this type of transportation problem: [125].

8.4. Transport of durable food

This task is also very similar to the delivery of bakery products (Chapter 8.1.), the difference here is that there is no collection (pick-up), only delivery, and the period time can be given an even higher value (e.g. monthly or annual transport). Here, however, there is no longer just one depot, but several depots, and the satellites can be also distinguished (even on several levels), for example in the case of products coming from abroad. For the operational parameter, the open route is also possible, this can be mainly for rail transport, and the inter-depot route can therefore also be the inter-depot route.
Processing constraint is an important factor, some products may not be able to be delivered at the same time.
I found the following publication for this type of transportation problem: [116].

8.5. Transport of beverages (soft drinks, alcohol)

This task is also similar to bakery products (Chapter 8.1.) and durable foods (Chapter 8.4.). There are only a few differences, which I will detail below. Processing constraint, a component is no need to pay attention if only beverages (soft drinks, alcohol) is transported. Here, too, as in the case of bakery products, care must be taken with the combination of pickup and delivery, as well as multi-level transport of products with different types of vehicles.
I found the following publication for this type of transportation problem: [116,132].

8.6. Tank transport

During tank transport, products are transported in containers on the liquid. Such liquid products may be water, gasoline, milk, oil, etc. This requires special vehicles. In this type of transport, usually, only one type of products can be transported at a time. The route quality between nodes is important. Packing means pouring into the container, unpacking means pouring out of the container. Delivery and pickup of liquid material are also possible. The following metrics can be used during tank transport task: length of the route, transported value, fuel consumption (approx. [from 25 to 40 liter / 100 km]), vehicle rental fee, route quality, route time (approx. [from 30 to 3000 minutes]), packaging time (approx. [from 30 to 60 minutes]), unpacking time, administrative time (approx. [from 10 to 60 minutes]), fuel filling time (approx. [from 10 to 30 minutes]), waiting time of the nodes, exceeding time window, a penalty point for missed customers.
I found the following publication for this type of transportation problem: [133].

8.7. Transport of durable products

The transport of durable products is the same as the transport of the beverages (Chapter 8.5.), the only difference being that most of the time there is only a delivery, the collection (pick-up) is not frequent.

8.8. Treatment of waste

Garbage collection is different from previous transportation tasks. This task may mean removing garbage from households or garbage from individual factories and office buildings to the garbage dump.
There is only collection (pick-up), no delivery. The system has only one depot and multiple customers. Among the relationships between nodes, the travel time between locations can be an important factor, as well as travel distance between locations. Perhaps the status of the roads can also be a factor as much as possible in trying to find roads that are gentle on vehicles.
For vehicles, the capacity constraint of each vehicle (approx. [from 10 to 40 m^3]) can be an important factor. Fuel consumption of the vehicles is also an important factor (approx. [from 15 to 40 liter / 100 km]).
Timeliness (period) is relatively not so important here. There is no service, so the service time at the locations is 0. Since there is no delivery of products, so there is no packing time and unpacking time, but loading (approx. [from 0.5 to 10 minutes]) and unloading time (approx. [from 5 to 10 minutes]) can be expected but the time is similar in all nodes. Neither product downtime of the location, nor the cost of administration time of the location, nor the quality control time of the location can be a component. Fuel filling time can be a factor, although, for non-electric vehicles, this time is low (approx. [from 5 to 10 minutes]). There is also a time window only in the depot (approx. [from 5 to 20 hour] in Hungary), you have to leave the garbage truck at a certain time and come back, this is due to the working hours of the human resource.

There are no special products, services, the positions do not have a capacity constraint of the location, there is a product demand of the location, other amounts of garbage are generated in a 10-storey tenement house and a family house must be collected from all places. There is no need to handle certain products at the same time (because there is a waste product in the system), neither the storage level at the locations nor the type of the products. nor the prices of the products. The operating parameter is also simple, there is only a collection (pick-up) process, and the garbage collection can be on a round trip, so there is no inter-depot route, neither delivery nor open route.
Among the metrics, minimization of the route is important, but there is no maximization of the profit except minimization of the vehicle rental fee. Of the time minimizations, the only factor is the minimization of the route time. Penalty points are exceeded if the number of suppliers is exceeded, which is an important factor.
I found the following publication for this type of transportation problem: [115,116].

8.9. Transport of mail items (domestic, foreign)

The handling of mail is also a completely different case. This task means that the mail is delivered from the sender to the recipient by the post office.
The number of levels can be important here, especially for foreign mail. From the post office, the mail items are sent to a central sorter, then to other sorters, after that to a post office close to the recipient, and finally from there to the recipient. So a multi-level satellite system is important here (approx. [from 3 to 5 levels]).
Among the attributes between nodes, the followings are important: travel time between the locations (approx. [30 minutes to 10 hours]), travel distance between locations (approx. [from 1 to 10,000 km]), and route quality between the locations.
Different types of vehicles (land, water, air) can also be used. It also depends on the levels, what kind of vehicles are used. Attributes for vehicles are also important, several attributes need to be considered. Several types of vehicles can be used, so these attributes are very important: capacity constraint of each vehicle (thousands, tens of thousands pieces of mail), fuel consumption of vehicles (approx. [from 15 to 25 liter / 100 km]) and charger time (approx. [from 60 to 300 minutes]) can also be calculated because electric vehicles can also transport mail items. Whether the vehicle is owned or rented can also be an important factor. Rental fee per vehicle types is also an important factor when using rented vehicles.
From a time point of view, the service time at the locations in the node does not make sense because there is no service. Packing time at the locations is low (approx. [from 5 to 30 minutes]), letters are only bagged, just as unpacking time at the locations is low (approx. [from 1 to 30 minutes]). However, there is a loading time at the locations (approx. [from 5 to 30 minutes]), as well as an unloading time at the locations (approx. [from 1 to 30 minutes]). There is no product downtime at the locations, there is an administration time at the locations (approx. [from 1 to 10 minutes]), the letters have to be sorted by area. There is no quality control time at the locations. There is also a time window, each sending post office has opening hours (approx. [from 5 to 20 hour] in Hungary), which can vary depending on the size of the post office. Each sorting location may also have opening hours. A multiple time window (approx. [from 5 to 11 hour] and [from 13 to 20 hour] in Hungary) can also be used, each post can close e.g. due to lunch break in the village post offices.
There are only products, but not services. A capacity constraint at the locations is usually not assigned to positions. The positions do not have any special product demands, the only purpose is to deliver the given letter. Profit per sent letter can be also important. There is no need for the processing order. There is no means of the storage level at the locations.
In terms of the operational parameter, the vehicles can return to the same locations (aggregated level), or even to another location of the aggregated level, so that the inter-depot route can be used. Delivery and pickup are also important, but only those products can be delivered which are collected to an aggregated level, products at the same levels cannot be delivered. The soft time window can also be used, e.g. the post office receives letters and parcels even after it is open, but

then the postmen have to pay overtime, so the objective function gets a penalty point. The open route can also be used, especially in the case of transport by train, boat or plane, and in the case of road transport, round-trip transport is common.
Among the metrics, minimization of the route is an important factor. Profit maximization is not usually used, so minimizing sold values, minimizing packaging costs, minimizing unpacking cost, minimizing loading costs, minimizing unloading costs, minimizing administrative costs, minimizing quality control cost does not make sense. Minimization of the fuel consumption, minimization of the vehicle rental fee can be considered. Maximization of road safety is not usually considered, but maximizing route quality can be considered. From the time minimization factor, the minimization of route time can be taken into account, but the minimization of packaging time, minimization of unpacking time factors do not make sense. Minimization of loading time can be useful, but minimization of product downtime has no sense. Minimization of administrative time, minimization of quality control time is not taken into account. Of the penalty points, the exceeding the number of suppliers and unvisited customers can be important, because it is important that all customers are served by a given vehicle.
The problem is not periodic, a specific period cannot be defined, because mail deliveries are not fixed demands, they can occur at any time, so the problem has one period.

8.10.Transport of parcels (domestic, foreign)

The case of parcels is very similar to the case of the letters (Chapter 8.9.). The difference here is that route quality between the locations is a very important case due to the potential for fragile, bulky, valuable packets. The capacity constraint of each vehicle is also a very important attribute, especially if the packages are bulky. The capacity constraint at the locations can also be a useful attribute. The other attributes are the same as the letter delivery.
Parcels and letters are usually collected from the post office at the same time, and during delivery, the packages are delivered with the letters to the nearest post office next to the consignee, where they are delivered separately to the consignee. Packages are delivered by car and letters are delivered by a courier.
I found the following publication for this type of transportation problem: [135].

8.11.Money transportation

The transportation of money is a special case of vehicle routing, it may be different from the previous cases. Cash transit means that cash is collected from post offices, shops, banks or just deliver it there. The money must be transported by a special money carrier, with security guards, equipment.
During this problem the tiered system (depot-satellite-customer) can be applied. The customers are the bank, post office, gas station, shop, etc.
Among the relationships between nodes, the travel time between the locations, the travel distance between the locations, the safety between the locations, and possibly the status of the route are important.
Among the attributes of the node, the type of the node (depot, satellite, customer) is an important factor, no other factor plays a role. The number of customers (bank, post office, gas station, shop, etc) can be approx. [from 1 to 1000 count].
Among the attributes of vehicles, the capacity constraint can be important (transporting too much money is risky), the vehicles can only transport a certain amount of money. The amount of money is determined by the money transportation insurance. Fuel consumption (approx. [from 10 to 25 liter / 100 km]) can also be important. Electric vehicles may not carry the money, so the charger time does not matter. It may also be important whether the vehicle is own or rented and the rental fee per vehicle types.

The attributes of the temporalities can be packing time (approx. [from 10 to 30 minutes]) at the locations. Product downtime at the locations is not there. Fuel filling time (approx. [from 10 to 30 minutes]) is low as it is not supplied by electric vehicles. The time window (approx. [from 5 to 20 hour] in Hungary), on the other hand, plays an important role due to the opening hours of certain places (e.g. post offices and shops have opening hours).
Among the attributes of products, the product demand at the locations can be different whether it is a distribution (approx. [from 40 to 600 million HUF]) or a general location (approx. [from 5 to 50 million HUF]). There is no fixed order of products, nor is it such that certain products must / can be delivered together. The number of products (banknotes/coins) in the warehouse (storage level at the locations) can be an important factor and possibly the type of products (banknotes/coins).
Among the operational parameters, the inter-depot route can also be used. And the delivery is important, but pickup can also be applied, the old worn-out money is collected for destruction, or the excess money is also handed over for storage security. A soft time window can also be used. The open route, so the use of open route is not common, money carriers take a tour.
Among the metrics, the minimization of the route is an important metric. Some of the factors of maximization of the profit may also be worth considering, such as minimization of packaging costs and minimization of unpacking costs, minimization of unloading costs and minimization of unloading costs, minimization of administrative costs, minimization of quality control cost, minimization fuel consumption, minimization of rental fee. Maximization of road safety is a very important factor, but so is the maximization of route quality. Minimization of route time, minimization of packaging time and unpacking time, minimization of loading time, minimization of administrative time, minimization of quality control time and minimization of waiting time for nodes can be important metrics. Penalty points can include exceeding the number of suppliers and unvisited customers.
I found the following publication for this type of transportation problem: [126,127].

8.12. Transport of advertising papers

In most cases, advertising papers are not delivered by the post office. Here a few levels can be distinguished, depending on what kind of advertising paper it is (e.g. large mall, local store). Advertising papers are delivered periodically, e.g. weekly, monthly. They are usually transported with one type of vehicle. Travel distance between nodes (approx. [from 5 to 1000 km]) is an important factor, other node properties do not matter. Vehicle capacity is important and fuel-related components (approx. [from 5 to 10 liter / 100 km]). There is packing time (approx. [from 10 to 30 minutes]), unpacking time (approx. [from 10 to 30 minutes]), loading time (approx. [from 10 to 30 minutes]) and unloading time (approx. [from 1 to 30 minutes]), administration time (approx. [from 10 to 30 minutes]) but no quality control time. The time window is not applied on the last level (for households), but in depot (approx. [from 5 to 20 hour] in Hungary) and satellite locations (approx. [from 5 to 20 hour] in Hungary). The advertising papers are only delivered, there is no collection in the system. Among the metrics the following can be taken into account: length of the route, packaging cost, packaging time, unpacking time, loading time.

8.13. Transport of a newspaper

There are two major cases in the delivery of newspapers. One is when you subscribe to the newspaper, it is then delivered to your home. The other case is when the newspaper is delivered to newspaper vendors (or shops).
The number of levels is one during the delivery of the subscription and during the delivery to the newspaper vendors. The number of periods is an important factor, as some newspapers are published weekly, monthly, others daily. Travel time between the nodes and travel distance between the nodes (approx. [from 5 to 1000 km]) are important factors. The capacity constraint of each vehicle is also important, as is fuel consumption (approx. [from 5 to 10 liter / 100 km]).

Among the temporality components, the following are important: packing time (approx. [from 10 to 30 minutes]), unpacking time (approx. [from 10 to 30 minutes]), loading time (approx. [from 10 to 30 minutes]), unloading time (approx. [from 10 to 30 minutes]), administration time (approx. [from 10 to 30 minutes]), time window (approx. [from 5 to 20 hour] in Hungary). Among the product components, the followings are important: product demand of the node (in case of shops approx [from 5 to 20 count] in case of home 1 count), prices of the product (approx. [from 50 to 5,000 HUF/count] in Hungary). The following components can be also important in terms of cost: packaging cost, unpacking cost. Among the operational parameters, the following are important: delivery, pickup, but pickup only if the newspaper is delivered to a newsagent or shop. Among the metrics, the following are important: length of the route, transported value, packaging cost, fuel consumption, route time, packaging time, unpacking time, loading time, administrative time.
I found the following publication for this type of transportation problem: [116,117,118,119].

8.14. Travel agency tour planning

The development of travel agency tour planning can be considered as a completely special vehicle routing. There are no levels (depot-satellite-customer), all nodes are equal, so it is a one-tier case.
The travel time between the locations is a key important factor. Safety between the locations is also important, allowing passengers to travel in a safer area. Route quality between the locations can also be important, the better the quality of the road. The node has no type, all nodes are equal.
From the attributes of vehicles also some factors can be considered. Such is the capacity constraint of each vehicle, i.e. how many people it can carry (approx. [from 7 to 500 people]). There is no service. Whether the vehicle is own or rented is also an important feature. Thus, the rental fee per vehicle types is also important. If the system has several different types of vehicles, fuel consumption (in case of bus approx. [from 15 to 30 liter / 100 km]) is also important.
Some attributes of temporality may be important. There is no service and no material handling, so the following factors are not taken into account: service time at the locations, packing time at the locations, unpacking time at the locations, loading time, unloading time, administrative time, quality control time. However, it is possible to use fuel filling time (approx. [from 10 to 30 minutes]) and the time window (approx. [from 8 to 24 hour]), as individual attractions, hotels and restaurants can have their opening hours.
There are no products or services, so any of these types of attributes are not considered.
The inter-depot route is omitted from the operational parameters. Delivery, pickup attributes are also omitted as there is no transport of products. The soft time window can be used.
Also, only a few types of metrics can be used because there is no transportation of products, only passenger transport. Minimization of the route can be an important factor, maximization of the profit, minimization vehicle rental fee are also important.

8.15. In-plant material handling: between warehouse and production

In-plant material handling processes are very different from outdoor material handling processes (e.g. Chapters 8.1.- 8.13.). Here, the tiered system corresponds to the depot and the individual production sites to the customers. There is no satellite, only material handling between depot and customers takes place.
Among the relationships between nodes, travel time between the locations can be an important attribute. The travel distance between the locations (approx. [from 100 m to 10 km]) is also taken into account. There is no need in defining safety between locations and route quality between locations, as there is in-plant material handling. Travel time between the locations (approx. [from 2 to 30 min]) can also be taken into account. Safety between the locations does not make sense, because there are no thefts or bad roads, all roads are of the same quality. The type of the node is distinguished: depot (this means warehouse), the customer (these are the production sites), and

there can also be charger stations because electric vehicles can also be used for the transportation of products and raw materials. Transport can also be done by electric vehicles.
In the case of vehicles, the capacity constraint of each vehicle types per the product type, fuel consumption and charger time (approx. [from 1 to 5 hour]) can also be important, as electric vehicles are also used here. Only products are transported to the nodes, no service is provided, so this attribute is not taken into account. Whether the vehicle is owned or rented and, if rented, the rental fee per vehicle types can also be an important attribute. Fuel consumption can also be important.
Of the temporal attributes, the service time at the locations attribute is not taken into account due to the lack of service provision, but in many cases, individual raw materials, semi-finished products or finished products have to be packaged separately during this transportation, such as packaging. Loading (approx. [from 5 to 30 min]) and unloading time (approx. [from 5 to 30 min]) in the locations can also play a big role. However, administration time (approx. [from 5 to 10 min]) in the locations and quality control time (approx. [from 5 to 10 min]) in the locations are both at stock and at the production sites. The time window is also an important factor, each production site requests the products at a certain time, so the Just In Time (JIT) principle appears, so not sooner, not later, at exactly that time.
The capacity constraint at the locations is an important factor, as is individual demand for the locations. There may be a fixed order of delivery of products at each production site. It may also be important whether certain products can be transported at the same time.
Among the operational parameters, the inter-depot route can also be important, in which case the vehicles start from one warehouse and then arrive at another location after visiting the production sites. Delivery and pickup are also present, as well as a combination of pickup and delivery. The soft time window can also be displayed so that the product can be delivered beyond the time window, but then it gets a penalty point. The open route is not displayed.
The main metric in this problem is the minimization of the route. Profit maximization is also important, but some factors may be left out, such as maximization of the sold values. Minimization of loading and unloading costs, minimization of administrative costs, and minimization of quality control cost, minimization of fuel consumption and rental fee can be important metrics. Maximization of road safety is not interpreted because the routes are uniform, just as there is no maximization of route quality. Time minimization is also an important factor, including the minimization of route time, the minimization of unpacking time, product downtime, administrative time and quality control time. Minimization of waiting time in the nodes is an important factor. Penalty points can be exceeding the number of suppliers or unvisited customers.

8.16. In-plant material handling: in the warehouse

In-plant material handling (rearrangement) in the warehouse is similar to material handling between a warehouse and production units (Chapter 8.15.) but also differs in many ways. There are no privileged nodes here, so there is no level case (neither depot nor satellite), all positions are considered equal because transport can take place from any position to any position. From the relations between the nodes, the travel time between the locations (approx. [from 1 to 5 min]) and the travel distance between the locations (approx. [from 10 to 500 m]) is important here as well, but the route safety and the route quality are irrelevant. The type of nodes can only be a customer or a charger station, no other type (depot, satellite). The number of vehicles that can be charged at one time can be an important factor because electric vehicles can also transport products.
For vehicles, the attributes capacity constraint, fuel consumption charger time also correspond to the material transportation between storage and production units. The ownership status component and the rental fee per vehicle types, fuel consumption components can be also useful in case of this type of transportation problem.
Of the temporality, there is no service provision here either, so the service time at the locations is not interpreted here either. It can be packing and unpacking time, as well as loading and unloading time in the locations. Product downtime and quality control time in the locations and administration

time are also required parameter. Fuel filling time can also be important, but the time window is irrelevant.
Among the attributes of the products, the capacity constraint at the locations can be important, there is no special product demand at the locations, the capacity limit can determine how many products are moved. There is no profit of the product and given the order of products components is also not important.
Among the operational parameters, the inter-depot route does not make sense because there are no particularly preferred nodes, split-delivery can be used, and also the combination of delivery and pickup. In the absence of a time window, the soft time window is not interpreted.
Among the metrics, the followings can be very important: minimization of the route and maximization of profit. The following factors may occur in profit maximization: minimization of the fuel consumption, minimization of the rental fee. Time minimization components can be also important.
I found the following publication for this type of transportation problem: [128].

8.17. In-plant material handling: transportation to other locations

This case differs from the transportation of material between the warehouse (Chapter 8.16.) and the production units (Chapter 8.15.) in that pickup does not appear here, as the aim here is to transport the products from different places in the warehouse to a specific point from which they will be transported to third party locations.

8.18. Patient transport

Patient transport is different from the cases discussed so far, here it is not the transportation of products, but the transport of patients from hospitals to patients' homes. The case where the patient is transported from the home to the hospital is separate, there the time between the nodes is important alone, the objective is to minimize the time taken so that the patient is transported to the hospital as soon as possible.
During the transport of patients discussed here (i.e., from hospitals, transporting patients to their own homes), the travel time between the locations (approx. [from 5 to 120 min]), the travel distance between the locations (approx. [from 1 to 200 km]), and the route quality between the locations of the nodes which may be important. There is only a single depot and a single "customer" here, no satellites are in the system. The depot means the hospital and the "customers" mean the patients 'homes.
The capacity constraint of each vehicle is an important factor, it can only carry a certain number of people (approx. [from 1 to 10 person]). Fuel consumption (approx. [from 8 to 12 liter / 100 km]) is also important. Charger time as these are not electric vehicles is not considered. Whether the vehicle is owned or rented is also an important factor, and if rented, the rental fee.
Of the temporality attributes, there are only a few factors that can be important, as it is not product transport but passenger transport. Thus, the service time and the packing and unpacking time in the node are not taken into account. There is no product downtime, administration and quality control time in the locations. Fuel filling time is not usually taken into account. It is not customary to define a time window either.
There are no products or services, so any of its attributes are not taken into account. Of the operational parameters, there is only delivery, the other attributes are not taken into account.
I found the following publications for this type of transportation problem: [136,137].

8.19.Maintenance

Maintenance is also very different from the cases presented so far. Here, each location needs to be visited, it takes a different amount of time for some maintenance activity to be done by the maintenance team.
In this case, the tiered system consists only of a depot and customers, there are no satellites. The depot is the starting point for the maintainers (the location of the maintenance company) and the customers are the places to be maintained.
Among the time relations, the following can be considered: the travel time (approx. [from 5 to 300 min]), the travel distance (approx. [from 5 to 500 km]), the safety and the route quality between the locations.
Among the attributes of the vehicles, the capacity constraint is important, as the vehicle can also supply the materials needed for maintenance. Fuel consumption (approx. [from 5 to 15 liter / 100 km]) and possibly charger time (approx. [from 1 hour to 5 hour]), even with electric vehicles can be also important components. The services that can be provided by each vehicle (maintenance team) are important. Vehicles can even be rented, so whether the vehicles are own or rented and rental fee of the vehicles are also determining factors.
Among the time components, the service time in the locations (approx. [from 30 min to several days]) is important, but it can include also packing time (approx. [from 5 to 30 min]) and unpacking time (approx. [from 5 to 30 min]). There is a loading (approx. [from 5 to 30 min]) and an unloading time (approx. [from 5 to 30 min]) because if the service (maintenance) requires certain devices, they need to be delivered. There is no product downtime in the locations, but it can also be an administration time in the locations and a quality control time in the locations if maintenance is reviewed at the node. The time window (approx. [from 5 to 20 hour] in Hungary) is an important factor, the individual nodes where maintenance is performed have opening hours, and the maintainers also have a working time to get back to the starting node.
Among the attributes of products the following can be considered: nodes do not have a capacity constraint due to the provision of services. The nodes can have a product demand, and the profit of the product can be also a key factor. It is possible to determine the processing order of the product, only one maintenance operation can be followed by another maintenance operation. It is also possible to define whether certain services can be provided at the same time on a single vehicle. The problem does not define how many products are on the nodes because the transportation of products is also only due to the provision of the service. The problem does not distinguish between product types, but service types.
Among the operational parameters, the inter-depot route can be a useful parameter. There is no delivery or pickup. The soft time window makes sense, so if the time window is exceeded, a penalty point is given, but the service is still performed. The open route does not appear most of the time because it is customary to plan a tour with vehicles.
The most important of the metrics here are the minimization of the route and some components of the maximization of the profit. Maximization of sold values, minimization of packaging, unpacking and unloading costs is important. It can also be an administrative cost as well as a quality control cost. Minimization of fuel consumption can also be an important consideration, as can minimizing vehicle rental fees, because certain vehicles can be rented. Road safety factor can be also taken into account. Maximization of route quality, route time, loading time, administration time and waiting time in the nodes may be important. Penalty points can be the followings: exceeding the number of vehicles and unvisited customers.
I found the following publication for this type of transportation problem: [138,139,140,141].

8.20. Test results

In this section, I present the test results of the program, that implements my general Vehicle Routing Problem model through two case studies. I have tested my model with 2 special cases. I have chosen 2 cases of *mail items* and *in-plant material handling between warehouse and production*. I have chosen these two tasks because both tasks are a common transportation problem and the parameters of the two tasks are very different.

8.20.1. Mail items

Table 28. summarizes the basic parameters of my data set for mail items.

Parameter	Value	Parameter	Value
Number of levels	4	Location of fourth level nodes	[600,700]
Number of nodes belonging to the first level	5	Number of charging stations	2
Location of first level nodes	[0,100]	Location of charging stations	[1000,1100]
Number of second-level nodes	10	Number of periods	1
Location of second level nodes	[200,300]	Number of product types	1
Number of third level nodes	10	Number of vehicles (per level)	2
Location of third level nodes	[400,500]	Number of rented vehicles	0
Number of nodes belonging to the fourth level	15		

Table 28.: Basic parameters data set for mail items

According to *Table 28.*, the nodes are located at four different levels. I have chosen so many levels because each mail (mostly foreign mail) goes through multiple distribution points while being sent from the sender to the recipient. Letters can be delivered by different vehicles, they will definitely be delivered by a car close to the recipient, so a charging station is also needed. The problem does not differentiate between letter types, so the system contains practically only one type of goods.

Parameter	Value	Parameter	Value
Administration cost	static, [10,50]	Quality control cost	static, [10,50]
Loading cost	static, [10,50]	Unloading cost	static, [10,50]

Table 29.: Cost-related parameters of my data set for mail items

Table 29. lists the cost parameters that can occur during a mail delivery. These are administration, loading, unloading and quality control.

Parameter	Value	Parameter	Value
Route quality between nodes	static, [100,500]	Travel time between nodes	static, [10,100]
Travel distance between nodes	static, coordinate based		

Table 30.: Node-related parameters of mail items

The parameters of the nodes are shown in *Table 30*. Distance and time are very important factors in almost every transport task. The condition of the route is mainly important for car transport (not by plane and rail).

Parameter	Value	Parameter	Value
Product demand of the node	static, [10,100]	Loading time	static, [30, 50]
Administration time	static, [30, 50]	Unloading time	static, [30, 50]

Table 31.: Product-related parameters of mail items

Of the attributes that can be associated with products (*Table 31.*), the most important is the product demand of the node, i.e., how many letters need to be delivered to a node. Furthermore, the administrative, loading and unloading time at the hub is also very important.

Parameter	Value	Parameter	Value
Capacity constraint of the vehicle	static, [10000, 50000]	Fuel consumption of the vehicle	static, [10, 100]

Table 32.: Vehicle-Related parameters of the data set of mail items

Among the parameters that can be linked to vehicles (*Table 32.*), the vehicle capacity limit is important. The capacity limit is relatively large here because the leaves themselves are small. Fuel consumption is also very important, it can be considered as the main parameter of almost all transport tasks.
Among the metrics (objective function components), my system includes: length of the route range, fuel consumption range, route quality range, route time range, unvisited customers range. These are the parameters that occur during most other transport tasks.
The result of a test run is illustrated in *Figure 14*, where I used the genetic algorithm (as an optimizing heuristic algorithm) and the Weighted Product Method (WPM) as a multi-objective optimization method. I also performed a fitness landscape analysis for this data set in *Chapter 7*.
In the figure, each color indicates the route of each vehicle. In the interval [50,100], there are nodes in the depot. These are visited by a single vehicle. From here, the products are transported by the vehicles to the nodes in the interval [200,300]. The products are then transported to the nodes in the interval [400,500] and from there to the nodes in the interval [600,700]. We can see a bit tangled result in the figure, because not only the minimization of the distance travelled is important, but also other constraints must be taken into consideration (e.g. vehicle capacity limitation, time costs).

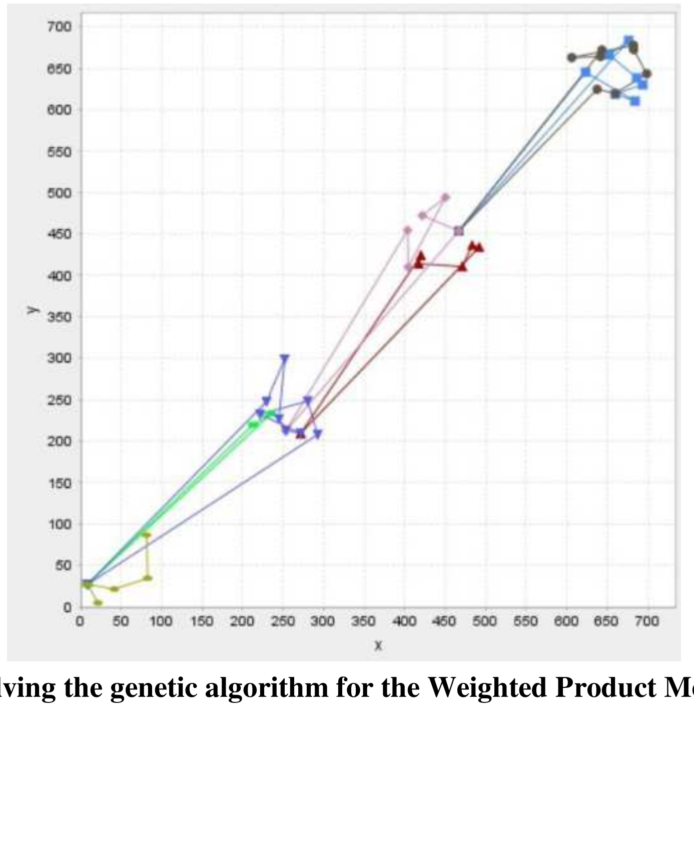

Figure 14.: Solving the genetic algorithm for the Weighted Product Method (WPM)

8.20.2. In-plant material handling between warehouse and production

In-plant material handling between warehouse and production is also a common transport task, occurring during all production.

Parameter	Value	Parameter	Value
Number of levels	2	Location of charging stations	[1000,1100]
Number of first level nodes	1	Number of periods	1
Location of first level nodes	[0,100]	Number of product types	1
Number of second-level nodes	25	Number of vehicles (per level)	2
Location of second level nodes	[0,100]	Number of rented vehicles	0
Number of charging stations	2		

Table 33.: Basic parameters of my in-plant material handling data set between warehouse and production

The basic parameters of the task are illustrated in *Table 33*. There are only two levels of the system, the warehouse on the first level and the production units on the second level. Vehicles transporting within the plant may require charging, these may be electric vehicles, so my system also includes a charging station.

Parameter	Value	Parameter	Value
Administration cost	static, [10,50]	Quality control cost	static, [10,50]
Loading cost	static, [10,50]	Unloading cost	static, [10,50]
Packing cost	static, [10,50]	Unpacking cost	static, [10,50]

Table 34.: Cost parameters of my in-plant material handling data set between warehouse and production

The cost parameters are illustrated in *Table 34*. There may be administration, loading and unloading costs, and quality control costs during transportation.

Parameter	Value	Parameter	Value
Travel distance between nodes	static, coordinate based	Travel time between nodes	static, [100,500]

Table 35.: Node parameters of the in-plant material handling data set between warehouse and production

Of the node parameters (*Table 35.*), I considered only two (these are important for all transport tasks): distance between nodes and route time.

Parameter	Value	Parameter	Value
Product demand of the node	static, [10,100]	Quality control time	static, [30,50]
Administration time	static, [30,50]	Service time	static, [30,50]
Loading time	static, [30,50]	Unloading time	static, [30,50]
Packing time	static, [30,50]	Unpacking time	static, [30,50]

Table 36.: Node parameters of the in-plant material handling data set between warehouse and production

The parameters related to the products are shown in *Table 36*. Here, too, I considered the classic transportation-related parameters. Demand for products, administration, loading and unloading time, quality control. The time of service actually means the time of production.

Parameter	Value	Parameter	Value
Capacity constraint of the vehicle	static, [10000,50000]	Maximum distance with full tank / full charge	static, [50000,100000]
Fuel consumption of the vehicle	static, [10,100]	Charger time of the vehicle	static, [10,30]

Table 37.: Vehicle parameters of the in-plant material handling data set between warehouse and production

In relation to vehicles (*Table 37.*), the parameters that can be used in classical transport tasks were also included. Vehicle capacity limit, fuel consumption (delivered with smaller vehicles within the plant), distance to be filled with tank (fuel level), and charging time.
Of the metrics, I considered only the length of the route.
Figure 15. illustrates the result of a test run. The figure shows the solution of the simulated annealing in case of improving the result of the random point insertion algorithm using the Weighted Product Method (WPM).
The figure shows the depot with a black triangle. From here, the two vehicles deliver the products. The route of one vehicle is shown in pink and the path of the other vehicle is shown in green. The route of vehicles is not tangled here, but if only the route of vehicles were important, there would be no constraint factors then we would get a much nicer route.

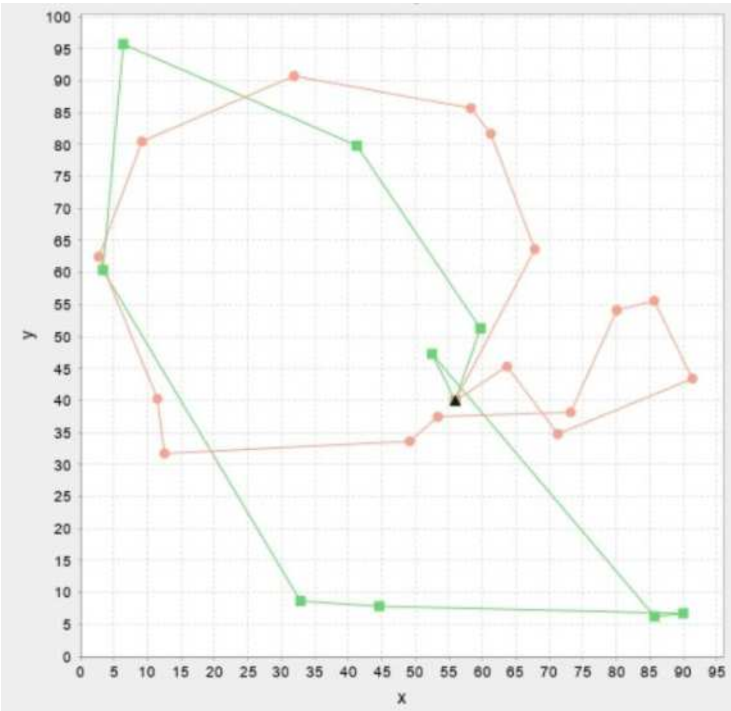

Figure 15.: Solution of simulated annealing in case of improving the result of the random point insertion algorithm using the Weighted Product Method (WPM)

<u>Thesis 5:</u>
I have demonstrated, that the proposed general model is able to manage a wide range of transportation task. I obtained result solutions that meet the right quality, result for practice.
Related publications: [P/1], [P/2], [P/15].

9. Summary

The focus of my dissertation is on a logistics task, the Vehicle Routing Problem (VRP). In the case of the basic VRP, the objective is to fulfil the demands of customers in a transportation network. Vehicles deliver products from one or more depots to customers and then return to the depot. Based on the literature many variants of the task were investigated over the years and many solution variants were developed for the special cases. In the first part of my dissertation, I conducted a literature search, where I presented the Vehicle Routing Problem components published so far. In the next section I analyzed and presented a general requirements against a general model. I presented the mathematical model of the problem including the variables, constraints, objective functions. After the mathematical model, I presented an ontological knowledge to design the control of the optimization of the VRP task. Ontology can be used among others to describe which types of products can be transported with which vehicles, which types of products can be transported together, what types of products may be needed by each customer, what components should be used in designing the system. Next, I outlined the formalism related to the applied optimization algorithms and evaluation. In the thesis I have analyzed the efficiency of different optimization algorithms. Heuristic algorithms can be divided into two parts: construction and improvement algorithms. Construction algorithms create one possible solution. They take the best steps locally, but most often the global optimum cannot be reached with their exclusive use. Their running time is usually low. The following construction algorithms are used in the dissertation: Arbitrary Insertion, Cheapest Insertion, Farthest Insertion, Greedy, Nearest Insertion, Nearest Neighbor. Improvement algorithms iteratively improve one or more possible solutions. Their running time is usually high. The global optimum can be achieved with their application. In the dissertation, I used the following improvement algorithms: Ant Colony System, Ant System, Elitist Strategy of Ant System, Firefly Algorithm, First Choice Hill Climbing, Genetic Algorithm, Harmony Search, MAX-MIN Ant System, Particle Swarm Optimization, Rank Based Version of Ant System, Simulated Annealing, Tabu Search. In this step I analyzed the efficiencies of each multi-objective optimization technique and the heuristic algorithms several of my objective functions are possible, I used the following multi-objective optimization in the dissertation: Bounded Objective Function Method, Pareto Ranking, Weighted Exponential Sum Method, Weighted Global Criterion Method, Weighted Product Method, Weighted Sum Method. I also performed the analysis of each neighbourhood operator (2-opt, order crossover, cycle crossover, partially matched crossover). The analysis showed that improving construction algorithms proved to be better than improving randomly generated solutions. In the analysis of operators, the 2-opt and partially matched crossover operators proved to be more efficient than cycle crossover and order crossover. I have also demonstrated, that the proposed general model is able to manage a wide range of transportation task. I obtained result solutions that meet the right quality, result for practice.

I summarized the scientific results in 5 theses.

Thesis 1:

I have developed a novel general Vehicle Routing Problem model that is suitable (both in terms of objective function and constraints) for solving different types of transportation problems. In the model, there are 4 basic components between which different attributes can be defined. The basic components are graph descriptors, vehicles, products, services, and time. I defined the relations between the basic components with functions. In this novel model the relations can be assigned to 6 large groups, including nodes, vehicles, time, products, costs, and operational parameters. The novelty of this complex VRP model is the one complex decision variable, which includes a wide range of decision variables of conventional VRP models. The solutions of the defined complex VRP problems are limited by different constraints. The model includes 11 constraints and 23 metrics.

Related publications: [P/1], [P/5], [P/6], [P/8], [P/16], [P/17], [P/22], [P/23], [P/24], [P/25].

<u>**Thesis 2.:**</u>

I have proposed a novel ontology system approach for the generalized Vehicle Routing Problem using the OWL modelling language. The generated ontology system was used for rule-based validation. The system contains the basic components of the general model of the Vehicle Routing Problem, the possible relations between them and the metrics. I extended the basic components of the system with additional subclasses (subcategories of product types, vehicle types and node types). I have created an ontology system, which supports flexibly and effectively the decision in the following topics:

1. ***Selection of Vehicle Routing components that can be used during transport to/from nodes.***
2. ***Selection of products, which can be transported to the nodes.***
3. ***Selection of Vehicle Routing metrics, which can be used during transport to/from nodes.***
4. ***Selection of products, which can be transported from nodes.***
5. ***Selection of vehicles, which can transport each product.***
6. ***Selection of products, which can be delivered together.***
7. ***Selection of Vehicle Routing components, which can be used when using each vehicle.***
8. ***Selection of Vehicle Routing metrics, which can be used when using each vehicle.***
9. ***Selection of products, which can be delivered by the given vehicles.***

Related publications: [P/2], [P/24].

<u>**Thesis 3.:**</u>

I created a novel representation of the general model of the Vehicle Routing Problem. The representation model includes the following main components:

1. ***Vector describing the order of nodes (depot, satellite, customer).***
2. ***The matrix describing the assignment of vehicles-nodes-products.***
3. ***Vector describing the assignment of vehicle - charger stations.***
4. ***Vector describing the vehicle - portal node assignment.***
5. ***Vector describing the level.***
6. ***Vector describing the period.***

I introduced and analyzed the search operators of heuristic algorithms.
Operators applied to the vector describing the order of the nodes (based on the literature) are the followings:

1. ***2-opt,***
2. ***Partially Matched Crossover – PMX,***
3. ***Order Crossover – OX,***
4. ***Cycle crossover – CX.***

Operator applied to the matrix describing the assignment of vehicles-nodes-products, vector describing the assignment of vehicle - charger stations and the vehicle - portal node assignment vector:

1. ***Regeneration operator.***

I also created an evaluation of the representation of the general model of the Vehicle Routing Problem.
Related publications: [P/1], [P/5], [P/6], [P/8], [P/10], [P/15], [P/16], [P/17], [P/21], [P/22], [P/23].

<u>Thesis 4.:</u>
I analytically analyzed the search space belonging to the optimization task of the generalized model of the Vehicle Routing Problem and I made a detailed analysis. In the search space study, I first analyzed the efficiency of multi-objective optimization techniques and heuristic algorithms. After I analyzed the efficiency of the operators (2-opt, cycle crossover, order crossover, and partially matched crossover). During the analysis of the operators I used different walk techniques like the random walk, adaptive walk, reverse adaptive walk, uphill-downhill walk, neutral walk and the reverse neutral walk. I also analyzed the search space based on information theory, which is also used to analyze the efficiency of operators. For the analysis of the operators, I also prepared the fitness cloud analysis.
Based on the examined sample, it can be concluded that the following multi-objective optimization techniques are effective: Weighted Global Criterium Method (WGCM), Weighted Product Method (WPM), Weighted Exponential Sum Method (WESM). Based on the performed analyzes, it can be concluded that the following heuristics are effective for Vehicle Routing Problems: Simulated Annealing (SA), Tabu Search (TS), First Choice Hill Climbing (FCHC), Genetic Algorithm (GA), Rank Based Version of Ant System (RBVAS). During the measurements, I also observed that the improvement of the solutions provided by the construction algorithms proved to be more effective than the improvement of the purely randomly generated solutions. In the measurements, I also found that the 2-opt and partially matched crossover operators are more efficient than the order crossover and cycle crossover operators.
Related publications: [P/3], [P/4], [P/7], [P/9], [P/11], [P/12], [P/13], [P/18].

<u>Thesis 5:</u>
I have demonstrated, that the proposed general model is able to manage a wide range of transportation task. I obtained result solutions that meet the right quality, result for practice.
Related publications: [P/1], [P/2], [P/15].

Literature

[1] Dantzig, G. B., & Ramser, J. H. (1959). The Truck Dispatching Problem. Management Science, 6(1), 80–91
[2] Stricker, R. (1970). Public sector vehicle routing: the Chinese Postman Problem (Doctoral dissertation, Massachusetts Institute of Technology).
[3] Christofides, N. (1976). The vehicle routing problem. Revue française d'automatique, informatique, recherche opérationnelle. Recherche opérationnelle, 10(V1), 55-70.
4] Cordeau, J. F., Laporte, G., & Mercier, A. (2001). A unified tabu search heuristic for vehicle routing problems with time windows. Journal of the Operational research society, 52(8), 928-936.
[5] Belhaiza, S., Hansen, P., & Laporte, G. (2014). A hybrid variable neighborhood tabu search heuristic for the vehicle routing problem with multiple time windows. Computers & Operations Research, 52, 269-281.
[6] Taillard, É., Badeau, P., Gendreau, M., Guertin, F., & Potvin, J. Y. (1997). A tabu search heuristic for the vehicle routing problem with soft time windows. Transportation science, 31(2), 170-186.
[7] Salhi, S., & Nagy, G. (1999). A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling. Journal of the operational Research Society, 50(10), 1034-1042.
[8] Brandão, J. (2004). A tabu search algorithm for the open vehicle routing problem. European Journal of Operational Research, 157(3), 552-564.
[9] Crevier, B., Cordeau, J. F., & Laporte, G. (2007). The multi-depot vehicle routing problem with inter-depot routes. European journal of operational research, 176(2), 756-773.
[10] Hemmelmayr, V. C., Cordeau, J. F., & Crainic, T. G. (2012). An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics. Computers & operations research, 39(12), 3215-3228.
[11] Gendreau, M., Laporte, G., Musaraganyi, C., & Taillard, É. D. (1999). A tabu search heuristic for the heterogeneous fleet vehicle routing problem. Computers & Operations Research, 26(12), 1153-1173.
[12] Ralphs, T. K., Kopman, L., Pulleyblank, W. R., & Trotter, L. E. (2003). On the capacitated vehicle routing problem. Mathematical programming, 94(2-3), 343-359.
[13] Ai, T. J., & Kachitvichyanukul, V. (2009). A particle swarm optimization for the vehicle routing problem with simultaneous pickup and delivery. Computers & Operations Research, 36(5), 1693-1702.
[14] Kabcome, P., & Mouktonglang, T. (2015). Vehicle routing problem for multiple product types, compartments, and trips with soft time windows. International Journal of Mathematics and Mathematical Sciences, 2015.
[15] Soysal, M., Bloemhof-Ruwaard, J. M., & Bektaş, T. (2015). The time-dependent two-echelon capacitated vehicle routing problem with environmental considerations. International Journal of Production Economics, 164, 366-378.
[16] Schneider, M., Stenger, A., & Goeke, D. (2014). The electric vehicle-routing problem with time windows and recharging stations. Transportation Science, 48(4), 500-520.
[17] Vidal, T., Crainic, T. G., Gendreau, M., Lahrichi, N., & Rei, W. (2012). A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. Operations Research, 60(3), 611-624.
[18] Stavropoulou, F., Repoussis, P. P., & Tarantilis, C. D. (2019). The Vehicle Routing Problem with Profits and consistency constraints. European Journal of Operational Research, 274(1), 340-356
[19] Huang, Y. H., Blazquez, C. A., Huang, S. H., Paredes-Belmar, G., & Latorre-Nuñez, G. (2019). Solving the Feeder Vehicle Routing Problem using ant colony optimization. Computers & Industrial Engineering, 127, 520-535
[20] Bertazzi, L., & Secomandi, N. (2018). Faster rollout search for the vehicle routing problem with stochastic demands and restocking. European Journal of Operational Research, 270(2), 487-497
[21] Fernández, E., Roca-Riu, M., & Speranza, M. G. (2018). The shared customer collaboration vehicle routing problem. European Journal of Operational Research, 265(3), 1078-1093
[22] Nucamendi-Guillén, S., Angel-Bello, F., Martínez-Salazar, I., & Cordero-Franco, A. E. (2018). The cumulative capacitated vehicle routing problem: New formulations and iterated greedy algorithms. Expert Systems with Applications, 113, 315-327
[23] Archetti, C., Fernández, E., & Huerta-Muñoz, D. L. (2017). The flexible periodic vehicle routing problem. Computers & Operations Research, 85, 58-70
[24] Reihaneh, M., & Ghoniem, A. (2019). A branch-and-price algorithm for a vehicle routing with demand allocation problem. European Journal of Operational Research, 272(2), 523-538

[25] Baniamerian, A., Bashiri, M., & Tavakkoli-Moghaddam, R. (2019). Modified variable neighborhood search and genetic algorithm for profitable heterogeneous vehicle routing problem with cross-docking. Applied Soft Computing, 75, 441-460
[26] Atefi, R., Salari, M., Coelho, L. C., & Renaud, J. (2018). The open vehicle routing problem with decoupling points. European Journal of Operational Research, 265(1), 316-327
[27] Archetti, C., Savelsbergh, M., & Speranza, M. G. (2016). The vehicle routing problem with occasional drivers. European Journal of Operational Research, 254(2), 472-480
[28] Sabar, N. R., Bhaskar, A., Chung, E., Turky, A., & Song, A. (2019). A self-adaptive evolutionary algorithm for dynamic vehicle routing problems with traffic congestion. Swarm and evolutionary computation, 44, 1018-1027
[29] Ran LIU, Yangyi TAO, Xiaolei XIE (2019): An adaptive large neighborhood search heuristic for the vehicle routing problem with time windows and synchronized visits Computers & Operations Research Vol 101, 250-262
[30] Poonthalir, G., & Nadarajan, R. (2018). A fuel efficient green vehicle routing problem with varying speed constraint (F-GVRP). Expert Systems with Applications, 100, 131-144.
[31] Bodin, L., Mingozzi, A., Baldacci, R., & Ball, M. (2000). The rollon–rolloff vehicle routing problem. Transportation Science, 34(3), 271-288.
[32] Salavati-Khoshghalb, M., Gendreau, M., Jabali, O., & Rei, W. (2019). An exact algorithm to solve the vehicle routing problem with stochastic demands under an optimal restocking policy. European Journal of Operational Research, 273(1), 175-189
[33] Toffolo, T. A., Christiaens, J., Van Malderen, S., Wauters, T., & Berghe, G. V. (2018). Stochastic local search with learning automaton for the swap-body vehicle routing problem. Computers & Operations Research, 89, 68-81
[34] Hu, Z. H., Sheu, J. B., Zhao, L., & Lu, C. C. (2015). A dynamic closed-loop vehicle routing problem with uncertainty and incompatible goods. Transportation Research Part C: Emerging Technologies, 55, 273-297
[35] Pop, P. C., Kara, I., & Marc, A. H. (2012). New mathematical models of the generalized vehicle routing problem and extensions. Applied Mathematical Modelling, 36(1), 97-107
[36] Song, B. D., & Ko, Y. D. (2016). A vehicle routing problem of both refrigerated-and general-type vehicles for perishable food products delivery. Journal of Food Engineering, 169, 61-71
[37] Talarico, L., Sörensen, K., & Springael, J. (2015). Metaheuristics for the risk-constrained cash-in-transit vehicle routing problem. European Journal of Operational Research, 244(2), 457-470
[38] Bae, H., & Moon, I. (2016). Multi-depot vehicle routing problem with time windows considering delivery and installation vehicles. Applied Mathematical Modelling, 40(13-14), 6536-6549
[39] Defryn, C., & Sörensen, K. (2017). A fast two-level variable neighborhood search for the clustered vehicle routing problem. Computers & Operations Research, 83, 78-94
[40] Drexl, M. (2013). Applications of the vehicle routing problem with trailers and transshipments. European Journal of Operational Research, 227(2), 275-283
[41] Bektaş, T., Gouveia, L., Martínez-Sykora, A., & Salazar-González, J. J. (2019). Balanced vehicle routing: Polyhedral analysis and branch-and-cut algorithm. European Journal of Operational Research, 273(2), 452-463
[42] Ke, L., & Feng, Z. (2013). A two-phase metaheuristic for the cumulative capacitated vehicle routing problem. Computers & Operations Research, 40(2), 633-638
[43] Talarico, L., Springael, J., Sörensen, K., & Talarico, F. (2017). A large neighbourhood metaheuristic for the risk-constrained cash-in-transit vehicle routing problem. Computers & Operations Research, 78, 547-556
[44] Cheng, R., Gen, M., & Tozawa, T. (1995). Vehicle routing problem with fuzzy due-time using genetic algorithms. Journal of Japan Society for Fuzzy Theory and Systems, 7(5), 1050-1061
[45] Bent, R. W., & Van Hentenryck, P. (2004). Scenario-based planning for partially dynamic vehicle routing with stochastic customers. Operations Research, 52(6), 977-987
[46] Lin, S. (1965). Computer solutions of the traveling salesman problem. Bell System Technical Journal, 44(10), 2245-2269.
[47] Caceres-Cruz, J., Arias, P., Guimarans, D., Riera, D., & Juan, A. A. (2014). Rich vehicle routing problem: Survey. ACM Computing Surveys (CSUR), 47(2), 1-28.
[48] Matthew Horridge, Holger Knublauch, Alan Rector, Robert Stevens, Chris Wroe, Simon Jupp, Georgina Moulton, Nick Drummond, Sebastian Brandt (2011): A Practical Guide To Building OWL Ontologies Using Protege 4 and CO-ODE Tools, Edition 1.3, The University Of Manchester.
[49] OWL: http://www.w3c.hu/forditasok/OWL/REC-owl-features-20040210.html (Last accessed: 2022.12.30.)

[50] IRI, URI, URL, URN and their differences https://fusion.cs.uni-jena.de/fusion/blog/2016/11/18/iri-uri-url-urn-and-their-differences/ (Last accessed: 2022.12.30.)
[51] OWL: http://www.w3c.hu/forditasok/OWL/REC-webont-req-20040210.html (Last accessed: 2022.12.30.)
[52] Smith, S. F., & Becker, M. A. (1997, March). An ontology for constructing scheduling systems. In Working Notes of 1997 AAAI Symposium on Ontological Engineering (pp. 120-127). AAAI Press.
[53] Himoff, J., Rzevski, G., & Skobelev, P. (2006, May). Magenta technology multi-agent logistics i-Scheduler for road transportation. In Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems (pp. 1514-1521).
[54] Lian, P., Park, D. W., & Kwon, H. C. (2007, December). Design of logistics ontology for semantic representing of situation in logistics. In Second Workshop on Digital Media and its Application in Museum & Heritages (DMAMH 2007) (pp. 432-437). IEEE.
[55] Dong, H., Hussain, F. K., & Chang, E. (2008, October). Transport service ontology and its application in the field of semantic search. In 2008 IEEE International Conference on Service Operations and Logistics, and Informatics (Vol. 1, pp. 820-824). IEEE.
[56] Hoxha, J., Scheuermann, A., & Bloehdorn, S. (2010, September). An approach to formal and semantic representation of logistics services. In Proceedings of the Workshop on Artificial Intelligence and Logistics (AILog), 19th European Conference on Artificial Intelligence (ECAI 2010), Lisbon, Portugal (pp. 73-78).
[57] Xu, D., Wijesooriya, C., Wang, Y. G., & Beydoun, G. (2011). Outbound logistics exception monitoring: A multi-perspective ontologies' approach with intelligent agents. Expert Systems with Applications, 38(11), 13604-13611.
[58] Anand, N., Yang, M., van Duin, J. R., & Tavasszy, L. (2012). GenCLOn: An ontology for city logistics. Expert Systems with Applications, 39(15), 11944-11960.
[59] De Oliveira, K. M., Bacha, F., Mnasser, H., & Abed, M. (2013). Transportation ontology definition and application for the content personalization of user interfaces. Expert Systems with Applications, 40(8), 3145-3159.
[60] Daniele, L., & Pires, L. F. (2013). An ontological approach to logistics. Enterprise interoperability, research and applications in the service-oriented ecosystem, IWEI, 13, 199-213.
[61] Zhang, L., Jiang, D., Zeng, Y., Ning, Y., & Wang, Q. (2014). Exploring Ontology-driven Modeling Approach for Multi-agent Cooperation in Emergency Logistics. JCP, 9(2), 285-294.
[62] Fumagalli, L., Pala, S., Garetti, M., & Negri, E. (2014, September). Ontology-based modeling of manufacturing and logistics systems for a new MES architecture. In IFIP International Conference on Advances in Production Management Systems (pp. 192-200). Springer, Berlin, Heidelberg.
[63] Wang, Y., Yi, J., Zhu, X., Luo, J., & Ji, B. (2015). Developing an ontology-based cold chain logistics monitoring and decision system. Journal of Sensors, 2015.
[64] Glöckner, M., & Ludwig, A. (2017). LoSe ODP-an ontology design pattern for logistics services. Advances in Ontology Design and Patterns, 32, 131.
[65] Li, S. T., Hsieh, H. C., & Sun, I. W. (2003, May). An Ontology-based Knowledge Management System for the Metal Industry. In WWW (Alternate Paper Tracks)
[66] Kachitvichyanukul, V., Sombuntham, P., & Kunnapapdeelert, S. (2015). Two solution representations for solving multi-depot vehicle routing problem with multiple pickup and delivery requests via PSO. Computers & Industrial Engineering, 89, 125-136.
[67] Wang, S., Lu, Z., Wei, L., Ji, G., & Yang, J. (2016). Fitness-scaling adaptive genetic algorithm with local search for solving the Multiple Depot Vehicle Routing Problem. Simulation, 92(7), 601-616.
[68] Wang, S., Wang, X., Liu, X., & Yu, J. (2018). A bi-objective vehicle-routing problem with soft time windows and multiple depots to minimize the total energy consumption and customer dissatisfaction. Sustainability, 10(11), 4257.
[69] Zhou, L., Baldacci, R., Vigo, D., & Wang, X. (2018). A multi-depot two-echelon vehicle routing problem with delivery options arising in the last mile distribution. European Journal of Operational Research, 265(2), 765-778.
[70] Englert, M., Röglin, H., & Vöcking, B. (2007, January). Worst case and probabilistic analysis of the 2-Opt algorithm for the TSP. In SODA (pp. 1295-1304).
[71] Hussain, A., Muhammad, Y. S., Nauman Sajid, M., Hussain, I., Mohamd Shoukry, A., & Gani, S. (2017). Genetic algorithm for traveling salesman problem with modified cycle crossover operator. Computational intelligence and neuroscience, 2017.
[72] Wright, S. (1932). The roles of mutation, inbreeding, crossbreeding, and selection in evolution (Vol. 1, pp. 356-366).
[73] Erik Pitzer (2013): Applied Fitness Landscape Analysis, PhD dissertation, JOHANNES KEPLER UNIVERSITÄT LINZ, Technisch-Naturwissenschaftliche Fakultät

[74] Pitzer, E., & Affenzeller, M. (2012). A comprehensive survey on fitness landscape analysis. In Recent advances in intelligent engineering systems (pp. 161-191). Springer, Berlin, Heidelberg
[75] Mattfeld, D. C., Bierwirth, C., & Kopfer, H. (1999). A search space analysis of the job shop scheduling problem. Annals of Operations Research, 86, 441-453.
[76] Ventresca, M., Ombuki-Berman, B., & Runka, A. (2013, April). Predicting genetic algorithm performance on the vehicle routing problem using information theoretic landscape measures. In European Conference on Evolutionary Computation in Combinatorial Optimization (pp. 214-225). Springer, Berlin, Heidelberg.
[77] Watson, J. P. (2010). An introduction to fitness landscape analysis and cost models for local search. In Handbook of metaheuristics (pp. 599-623). Springer, Boston, MA.
[78] Humeau, J., Liefooghe, A., Talbi, E. G., & Verel, S. (2013). ParadisEO-MO: From fitness landscape analysis to efficient local search algorithms. Journal of Heuristics, 19(6), 881-915.
[79] Merz, P., & Freisleben, B. (1999). Fitness landscapes and memetic algorithm design. New ideas in optimization, 245-260.
[80] Hordijk, W. (1996). A measure of landscapes. Evolutionary computation, 4(4), 335-360.
[81] Verel, S., Collard, P., & Clergue, M. (2007). Measuring the evolvability landscape to study neutrality. arXiv preprint arXiv:0709.4011.
[82] Ochoa, G., Tomassini, M., Vérel, S., & Darabos, C. (2008, July). A study of NK landscapes' basins and local optima networks. In Proceedings of the 10th annual conference on Genetic and evolutionary computation (pp. 555-562).
[83] Marmion, M. E., Dhaenens, C., Jourdan, L., Liefooghe, A., & Verel, S. (2011, January). On the neutrality of flowshop scheduling fitness landscapes. In International Conference on Learning and Intelligent Optimization (pp. 238-252).
[84] Ochoa, G., Qu, R., & Burke, E. K. (2009, July). Analyzing the landscape of a graph based hyper-heuristic for timetabling problems. In Proceedings of the 11th Annual conference on Genetic and evolutionary computation (pp. 341-348).
[85] Vassilev, V. K., Fogarty, T. C., & Miller, J. F. (2000). Information characteristics and the structure of landscapes. Evolutionary computation, 8(1), 31-60
[86] Tavares, J., Pereira, F. B., & Costa, E. (2008). Multidimensional knapsack problem: A fitness landscape analysis. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 38(3), 604-616.
[87] Pitzer, E., & Affenzeller, M. (2012). A comprehensive survey on fitness landscape analysis. In Recent advances in intelligent engineering systems (pp. 161-191).
[88] Stadler, P. F. (2002). Fitness landscapes. In Biological evolution and statistical physics (pp. 183-204). Springer, Berlin, Heidelberg
[89] Uludağ, G., & Uyar, A. Ş. (2009, September). Fitness landscape analysis of differential evolution algorithms. In 2009 Fifth International Conference on Soft Computing, Computing with Words and Perceptions in System Analysis, Decision and Control (pp. 1-4).
[90] Merz, P., & Freisleben, B. (2001). Memetic algorithms for the traveling salesman problem. Complex Systems, 13(4), 297-346
[91] Merz, P. (2004). Advanced fitness landscape analysis and the performance of memetic algorithms. Evolutionary Computation, 12(3), 303-325
[92] Chicano, F., Daolio, F., Ochoa, G., Vérel, S., Tomassini, M., & Alba, E. (2012, September). Local optima networks, landscape autocorrelation and heuristic search performance. In International Conference on Parallel Problem Solving from Nature (pp. 337-347).
[93] Pitzer, E., & Affenzeller, M. (2012). A comprehensive survey on fitness landscape analysis. In Recent advances in intelligent engineering systems (pp. 161-191).
[94] Tayarani-N, M. H., & Prügel-Bennett, A. (2016). An analysis of the fitness landscape of travelling salesman problem. Evolutionary computation, 24(2), 347-384.
[95] Mathias, K., & Whitley, D. (1992). Genetic operators, the fitness landscape and the traveling salesman problem. In PPSN (pp. 221-230).
[96] Fonlupt, C., Robilliard, D., & Preux, P. (1997, April). Fitness landscape and the behavior of heuristics. In Evolution Artificielle(Vol. 97, p. 56).
[97] Wang, S., Zhu, Q., & Kang, L. (2006, May). Landscape properties and hybrid evolutionary algorithm for optimum multiuser detection problem. In International Conference on Computational Science (pp. 340-347).
[98] Lu, G., Li, J., & Yao, X. (2011, April). Fitness-probability cloud and a measure of problem hardness for evolutionary algorithms. In European Conference on Evolutionary Computation in Combinatorial Optimization (pp. 108-117).
[99] Wang, S., Zhu, Q., & Kang, L. (2006, May). Landscape properties and hybrid evolutionary algorithm for optimum multiuser detection problem. In International Conference on Computational Science (pp. 340-347).

[100] Tayarani-N, M. H., & Prügel-Bennett, A. (2016). An analysis of the fitness landscape of travelling salesman problem. Evolutionary computation, 24(2), 347-384.
[101] Belaidouni, M., & Hao, J. K. (2000, September). An analysis of the configuration space of the maximal constraint satisfaction problem. In International Conference on Parallel Problem Solving from Nature (pp. 49-58).
[102] Collard, P., Verel, S., & Clergue, M. (2007). Local search heuristics: Fitness cloud versus fitness landscape.
[103] Müller, C. L., & Sbalzarini, I. F. (2011, April). Global characterization of the CEC 2005 fitness landscapes using fitness-distance analysis. In European Conference on the Applications of Evolutionary Computation (pp. 294-303).
[104] Preux, P., Robilliard, D., & Fonlupt, C. (1997). Fitness Landscapes of Combinatorial Problems And The Performance Of Search Algorithms
[105] Czech, Z. J. (2008, April). Statistical measures of a fitness landscape for the vehicle routing problem. In 2008 IEEE International Symposium on Parallel and Distributed Processing(pp. 1-8).
[106] Jones, T., & Forrest, S. (1995). Fitness distance correlation as a measure of problem difficulty for genetic algorithms. In Proc. 6th Internat. Conf. on Genetic Algorithms.
[107] Merz, P., & Freisleben, B. (1998, September). Memetic algorithms and the fitness landscape of the graph bi-partitioning problem. In International Conference on Parallel Problem Solving from Nature (pp. 765-774).
[108] Fonlupt, C., Robilliard, D., Preux, P., & Talbi, E. G. (1999). Fitness landscapes and performance of meta-heuristics. In Meta-Heuristics (pp. 257-268).
[109] Sébastien Verel (2013): Fitness Landscapes and Graphs: Multimodularity, Ruggedness and Neutrality, Université Nice Sophia Antipolis / CNRS, France, DOLPHIN team - INRIA Lille-Nord Europe, http://www-lisic.univ-littoral.fr/~verel/talks/2tut16-verel.pdf (Last accessed: 2022.12.30.)
[110] HEURISTIC OPTIMIZATION, Search Space Analysis http://iridia.ulb.ac.be/~stuetzle/Teaching/HO12/Slides/Lecture12.pdf (Last accessed: 2022.12.30.)
[111] Rogier Hans Wuijts (2018): Investigation of the Traveling Thief Problem, Utrecht University, Master Thesis
[112] Smith, T., Husbands, P., & O'Shea, M. (2002). Fitness landscapes and evolvability. Evolutionary computation, 10(1), 1-34.
[113] Lokketangen, A., Oppen, J., Oyola, J., & Woodruff, D. L. (2012). An attribute based similarity function for VRP decision support. Decision making in manufacturing and services, 6(1-2).
[114] Wang, K. P., Huang, L., Zhou, C. G., & Pang, W. (2003, November). Particle swarm optimization for traveling salesman problem. In Proceedings of the 2003 international conference on machine learning and cybernetics (IEEE cat. no. 03ex693) (Vol. 3, pp. 1583-1585). IEEE.
[115] Moustafa, A., Abdelhalim, A. A., Eltawil, A. B., & Fors, N. (2013). Waste collection vehicle routing problem: case study in Alexandria, Egypt. In The 19th International Conference on Industrial Engineering and Engineering Management (pp. 935-944). Springer, Berlin, Heidelberg.
[116] Golden, B. L., Assad, A. A., & Wasil, E. A. (2002). Routing vehicles in the real world: applications in the solid waste, beverage, food, dairy, and newspaper industries. In The vehicle routing problem (pp. 245-286). Society for Industrial and Applied Mathematics.
[117] Boonkleaw, A., Suthikarnnarunai, N., & Srinon, R. (2009, October). Strategic planning and vehicle routing algorithm for newspaper delivery problem: Case study of morning newspaper, bangkok, thailand. In Proceedings of the world congress on engineering and computer science (Vol. 2, pp. 1067-1071).
[118] Osaba, E., Yang, X. S., Diaz, F., Onieva, E., Masegosa, A. D., & Perallos, A. (2017). A discrete firefly algorithm to solve a rich vehicle routing problem modelling a newspaper distribution system with recycling policy. Soft Computing, 21(18), 5295-5308.
[119] Boonkleaw, A., Suthikarnnarunai, N., & Srinon, R. (2010). Strategic planning for newspaper delivery problem using vehicle routing algorithm with time window (VRPTW).
[120] Maden, W., Eglese, R., & Black, D. (2010). Vehicle routing and scheduling with time-varying data: A case study. Journal of the Operational Research Society, 61(3), 515-522.
[121] Hsu, C. I., Hung, S. F., & Li, H. C. (2007). Vehicle routing problem with time-windows for perishable food delivery. Journal of food engineering, 80(2), 465-475.
[122] Chen, H. K., Hsueh, C. F., & Chang, M. S. (2009). Production scheduling and vehicle routing with time windows for perishable food products. Computers & operations research, 36(7), 2311-2319.
[123] Song, B. D., & Ko, Y. D. (2016). A vehicle routing problem of both refrigerated-and general-type vehicles for perishable food products delivery. Journal of food engineering, 169, 61-71.
[124] Osvald, A., & Stirn, L. Z. (2008). A vehicle routing algorithm for the distribution of fresh vegetables and similar perishable food. Journal of food engineering, 85(2), 285-295.

[125] Zhang, Y., & Chen, X. D. (2014). An optimization model for the vehicle routing problem in multi-product frozen food delivery. Journal of applied research and technology, 12(2), 239-250.
[126] Talarico, L., Sörensen, K., & Springael, J. (2015). Metaheuristics for the risk-constrained cash-in-transit vehicle routing problem. European Journal of operational research, 244(2), 457-470.
[127] Ghannadpour, S. F., & Zandiyeh, F. (2020). A new game-theoretical multi-objective evolutionary approach for cash-in-transit vehicle routing problem with time windows (A Real life Case). Applied Soft Computing, 106378.
[128] Bocewicz, G., Nielsen, P., & Banaszak, Z. (2018, September). Declarative modeling of a milk-run vehicle routing problem for split and merge supply streams scheduling. In International Conference on Information Systems Architecture and Technology (pp. 157-172). Springer, Cham.
[129] Keskinturk, T., & Yildirim, M. B. (2011, June). A genetic algorithm metaheuristic for bakery distribution vehicle routing problem with load balancing. In 2011 International Symposium on Innovations in Intelligent Systems and Applications (pp. 287-291). IEEE.
[130] Tunjongsirigul, B., & Pongchairerks, P. (2010, May). A genetic algorithm for a vehicle routing problem on a real application of bakery delivery. In 2010 2nd International Conference on Electronic Computer Technology (pp. 214-217). IEEE.
[131] Derigs, U., & Grabenbauer, G. (1993). Intime-A new heuristic approach to the vehicle routing problem with time windows, with a bakery fleet case. American Journal of Mathematical and Management Sciences, 13(3-4), 249-266.
[132] Privé, J., Renaud, J., Boctor, F., & Laporte, G. (2006). Solving a vehicle-routing problem arising in soft-drink distribution. Journal of the Operational Research Society, 57(9), 1045-1052.
[133] Stolk, J., Mann, I., Mohais, A., & Michalewicz, Z. (2013). Combining vehicle routing and packing for optimal delivery schedules of water tanks. OR Insight, 26(3), 167-190.
[134] Hollis, B. L., Forbes, M. A., & Douglas, B. E. (2006). Vehicle routing and crew scheduling for metropolitan mail distribution at Australia Post. European Journal of Operational Research, 173(1), 133-150.
[135] Wasner, M., & Zäpfel, G. (2004). An integrated multi-depot hub-location vehicle routing model for network planning of parcel service. International journal of production economics, 90(3), 403-419.
[136] Pérez, E. S., Yepez, L. A., & de la Mota, I. F. (2010, July). Simulation and optimization of the pre-hospital care system of the National University of Mexico using travelling salesman problem algorithms. In SummerSim (pp. 364-370).
[137] Xiao, L., Dridi, M., Hajjam El Hassani, A., Fei, H., & Lin, W. (2018). An improved cuckoo search for a patient transportation problem with consideration of reducing transport emissions. Sustainability, 10(3), 793.
[138] Li, J., Li, T., Yu, Y., Zhang, Z., Pardalos, P. M., Zhang, Y., & Ma, Y. (2019). Discrete firefly algorithm with compound neighborhoods for asymmetric multi-depot vehicle routing problem in the maintenance of farm machinery. Applied Soft Computing, 81, 105460.
[139] Rashidnejad, M., Ebrahimnejad, S., & Safari, J. (2018). A bi-objective model of preventive maintenance planning in distributed systems considering vehicle routing problem. Computers & Industrial Engineering, 120, 360-381.
[140] Dhahri, A., Zidi, K., & Ghedira, K. (2015). A variable neighborhood search for the vehicle routing problem with time windows and preventive maintenance activities. Electronic Notes in Discrete Mathematics, 47, 229-236.
[141] Xie, B., Li, Y., & Jin, L. (2013). Vehicle routing optimization for deicing salt spreading in winter highway maintenance. Procedia-Social and Behavioral Sciences, 96, 945-953.
[142] Kizilateş, G., & Nuriyeva, F. (2013). On the nearest neighbor algorithms for the traveling salesman problem. In Advances in Computational Science, Engineering and Information Technology (pp. 111-118). Springer, Heidelberg.
[143] Rosenkrantz, D. J., Stearns, R. E., & Lewis, II, P. M. (1977). An analysis of several heuristics for the traveling salesman problem. SIAM journal on computing, 6(3), 563-581.
[144] Nilsson, C. (2003). Heuristics for the traveling salesman problem. Linkoping University, 38, 00085-9.
[145] Braun, H. (1990, October). On solving travelling salesman problems by genetic algorithms. In International Conference on Parallel Problem Solving from Nature (pp. 129-133). Springer, Berlin, Heidelberg.
[146] Zhou, J., Xiao, H., Wang, H., & Dai, H. N. (2016, July). Parallelizing simulated annealing algorithm in many integrated core architecture. In International Conference on Computational Science and Its Applications (pp. 239-250). Springer, Cham.
[147] Tsubakitani, S., & Evans, J. R. (1998). Optimizing tabu list size for the traveling salesman problem. Computers & Operations Research, 25(2), 91-97.

[148] Marini, F., & Walczak, B. (2015). Particle swarm optimization (PSO). A tutorial. Chemometrics and Intelligent Laboratory Systems, 149, 153-165.
[149] Wang, K. P., Huang, L., Zhou, C. G., & Pang, W. (2003, November). Particle swarm optimization for traveling salesman problem. In Proceedings of the 2003 international conference on machine learning and cybernetics (IEEE cat. no. 03ex693) (Vol. 3, pp. 1583-1585). IEEE.
[150] Zhang, L., Liu, L., Yang, X. S., & Dai, Y. (2016). A novel hybrid firefly algorithm for global optimization. PloS one, 11(9), e0163230.
[151] Osaba, E., Carballedo, R., Yang, X. S., & Diaz, F. (2016). An evolutionary discrete firefly algorithm with novel operators for solving the vehicle routing problem with time windows. In Nature-Inspired Computation in Engineering, pp. 21-41. Springer, Cham.
[152] Jati, G. K. (2011). Evolutionary discrete firefly algorithm for travelling salesman problem. In Adaptive and intelligent systems, pp. 393-403. Springer, Berlin, Heidelberg.
[153] Bouzidi, M., & Riffi, M. E. (2014). Adaptation of the Harmony Search Algorithm to solve the Travelling Salesman Problem. Journal of Theoretical & Applied Information Technology, 62(1), pp. 154-160.
[154] Boryczka, U., & Szwarc, K. (2019). The adaptation of the harmony search algorithm to the ATSP with the evaluation of the influence of the pitch adjustment place on the quality of results. Journal of Information and Telecommunication, 3(1), 2-18.
[155] Blum, C. (2005). Ant colony optimization: Introduction and recent trends. Physics of Life reviews, 2(4), 353-373.
[156] Stützle, T., & Dorigo, M. (1999). ACO algorithms for the traveling salesman problem. Evolutionary algorithms in engineering and computer science, 4, 163-183.
[157] Bullnheimer, B., Hartl, R. F., & Strauss, C. (1997). A new rank based version of the Ant System. A computational study.
[158] Ökonometria /Elméleti jegyzet/ Nagy Lajos, Balogh Péter (2013): https://dtk.tankonyvtar.hu/xmlui/handle/123456789/3619 (Last accessed: 2022.12.30.)
[159] Barta Gergő: Idősorok elemzése, http://www.cs.bme.hu/nagyadat/timeseries-gergo_barta.pdf (Last accessed: 2022.12.30.)
[160] Nwogu, E. C., Iwueze, I. S., & Nlebedim, V. U. (2016). Some tests for seasonality in time series data. Journal of Modern Applied Statistical Methods, 15(2), 24.
[161] Applied Time Series Analysis, https://online.stat.psu.edu/stat510/ (Last accessed: 2022.12.30.)
[162] Tratar, L. F., & Strmčnik, E. (2016). The comparison of Holt–Winters method and Multiple regression method: A case study. Energy, 109, 266-276.
[163] Genre, V., Kenny, G., Meyler, A., & Timmermann, A. (2013). Combining expert forecasts: Can anything beat the simple average?. International Journal of Forecasting, 29(1), 108-121.
[164] Ostertagová, E., & Ostertag, O. (2011, September). The simple exponential smoothing model. In The 4th International Conference on Modelling of Mechanical and Mechatronic Systems, Technical University of Košice, Slovak Republic, Proceedings of conference (pp. 380-384).
[165] Rasmussen, R. (2004). On time series data and optimal parameters. Omega, 32(2), 111-120.
[166] Bai, Y., Jin, X., Wang, X., Su, T., Kong, J., & Lu, Y. (2019). Compound autoregressive network for prediction of multivariate time series. Complexity, 2019.
[167] Géczi-Papp Renáta: Autoregresszív és mozgóátlag folyamatok, http://gtk.uni-miskolc.hu/files/8791/D%C3%B6nt%C3%A9sel%C5%91k%C3%A9sz%C3%ADt%C3%A9s+m%C3%B3dszertana+2.pdf (Last accessed: 2022.12.30.)
[168] Wei, W. W. (2018). Multivariate time series analysis and applications. John Wiley & Sons.
[169] Jason Brownlee: A Gentle Introduction to Autocorrelation and Partial Autocorrelation, https://machinelearningmastery.com/gentle-introduction-autocorrelation-partial-autocorrelation/
[170] Prof. Dr. habil. Kosztyán Zsolt Tibor: Kvantitatív módszerek, https://kmt.gtk.uni-pannon.hu/kzst/oktatas/km/levelezo/KM01.pptx (Last accessed: 2022.12.30.)
[171] Wei, W. W. (2006). Time series analysis. In The Oxford Handbook of Quantitative Methods in Psychology: Vol. 2.
[172] Jason Brownlee: Time Series Forecasting Performance Measures With Python, https://machinelearningmastery.com/time-series-forecasting-performance-measures-with-python/ (Last accessed: 2022.12.30.)
[173] Adhikari, R., & Agrawal, R. K. (2013). An introductory study on time series modeling and forecasting. arXiv preprint arXiv:1302.6613.
[174] Bodon Ferenc, Buza Krisztián (2014): Adatbányászat, http://www.cs.bme.hu/nagyadat/bodon.pdf (Last accessed: 2022.12.30.)

[175] Pang-Ning Tan, Michael Steinbach, Vipin Kumar. Bevezetés az adatbányászatba. https://gyires.inf.unideb.hu/KMITT/a04/ (Last accessed: 2022.12.30.)
[176] Safavian, S. R., & Landgrebe, D. (1991). A survey of decision tree classifier methodology. IEEE transactions on systems, man, and cybernetics, 21(3), 660-674.
[177] Szabó Adrienn: Random Forests - Véletlen erdők, https://docplayer.hu/11299587-Random-forests-veletlen-erdok.html (Last accessed: 2022.12.30.)
[178] Zhao, Y., & Zhang, Y. (2008). Comparison of decision tree methods for finding active objects. Advances in Space Research, 41(12), 1955-1959.
[179] Zhang, S., Li, X., Zong, M., Zhu, X., & Wang, R. (2017). Efficient knn classification with different numbers of nearest neighbors. IEEE transactions on neural networks and learning systems, 29(5), 1774-1785.
[180] Ren, J., Lee, S. D., Chen, X., Kao, B., Cheng, R., & Cheung, D. (2009, December). Naive bayes classification of uncertain data. In 2009 Ninth IEEE International Conference on Data Mining (pp. 944-949). IEEE.
[181] Powers, D. M. (2020). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. arXiv preprint arXiv:2010.16061.
[182] Cheng, R., Gen, M., & Tozawa, T. (1995). Vehicle routing problem with fuzzy due-time using genetic algorithms. Journal of Japan Society for Fuzzy Theory and Systems, 7(5), 1050-1061.
[183] Bansal, S., & Katiyar, V. (2014). Integrating Fuzzy and Ant Colony System for Fuzzy Vehicle Routing Problem with Time Windows. arXiv preprint arXiv:1411.3806
[184] Dror, M., & Trudeau, P. (1986). Stochastic vehicle routing with modified savings algorithm. European Journal of Operational Research, 23(2), 228-235.)
[185] Balogh Sándor (2009) Többszempontú gazdasági döntéseket segítő Genetikus Algoritmus kidolgozása és alkalmazásai, Phd értekezés, Kaposvári Egyetem, Gazdaságtudományi Kar, Informatika Tanszék
[186] Kuang-HuaChang (2015) Chapter 5 - Multiobjective Optimization and Advanced Topics, Design Theory and Methods Using CAD/CAE, The Computer Aided Engineering Design Series, 325-406
[187] Arora, J. S. (2004). Multiobjective Optimum Design Concepts and Methods. Introduction to Optimum Design, 543-563.
[188] Marler, R. T., & Arora, J. S. (2004). Survey of multi-objective optimization methods for engineering. Structural and multidisciplinary optimization, 26(6), 369-395.
[189] Ghoseiri, K., & Ghannadpour, S. F. (2010). Multi-objective vehicle routing problem with time windows using goal programming and genetic algorithm. Applied Soft Computing, 10(4), 1096-1107.

Author's scientific works

Publications related to the dissertation

I. Scientific journal article:
in international journal:

[P/1] A Agárdi, L Kovács, T Bányai (2022). Mathematical Model for the Generalized VRP Model, SUSTAINABILITY 14 (18) Paper: 11639 **SCOPUS INDEXED Q1, IF=3,889**
[P/2] A Agárdi, L Kovács, T Bányai (2022). Ontology Support for Vehicle Routing Problem, APPLIED SCIENCES-BASEL, 12(23), Paper:12299, **SCOPUS INDEXED Q2, IF=2,838**
[P/3] L Kovács, A Agárdi, T Bányai (2020). Fitness Landscape Analysis and Edge Weighting-Based Optimization of Vehicle Routing Problems, PROCESSES 8 (11), Paper: 1363 **SCOPUS INDEXED Q2, IF=2,753**
[P/4]A Agárdi, L Kovács, T Bányai (2021). An Attraction Map Framework of a Complex Multi-Echelon Vehicle Routing Problem with Random Walk Analysis, APPLIED SCIENCES-BASEL, 11 (5), Paper: 2100 **SCOPUS INDEXED Q2, IF=2,474**
[P/5] A Agárdi, L Kovács, T Bányai (2019). Two - Echelon Vehicle Routing Problem with Recharge Stations, TRANSPORT AND TELECOMMUNICATION 20 (4), 305–317 **SCOPUS INDEXED Q2**
[P/6] A Agárdi, L Kovács, T Bányai (2019). Optimization of Multi-Depot Periodic Vehicle Routing Problem with Time Window, ACADEMIC JOURNAL OF MANUFACTURING ENGINEERING 17 (4), 96-108 **SCOPUS INDEXED Q3**
[P/7] A Agárdi, L Kovács, T Bányai (2021). Comparision of the walk techniques for fitness state space analysis in vehicle routing problem, ACTA POLYTECHNICA 61, 672-683. **SCOPUS INDEXED Q3**
[P/8] A Agárdi, L Kovács, T Bányai (2021). Ant Colony Algorithms For The Vehicle Routing Problem With Time Window, Period And Multiple Depots, MANUFACTURING TECHNOLOGY 21, 422-433. **SCOPUS INDEXED Q3**
[P/9] A Agárdi, L Kovács, T Bányai (2021). The Fitness Landscape Analysis of The Ant Colony System Algorithm in Solving a Vehicle Routing Problem, ACADEMIC JOURNAL OF MANUFACTURING ENGINEERING 19 (2), 85-89. **SCOPUS INDEXED Q3**
[P/10] A Agárdi, L Kovács, T Bányai (2021). Using Time Series and Classification in Vehicle Routing Problem, INTERNATIONAL JOURNAL OF PERFORMABILITY ENGINEERING 17 (1), 14-25. **SCOPUS INDEXED Q4**
[P/11] A Agárdi (2023). Fitness Landscape Analysis of Population-Based Heuristics in Solving a Complex Vehicle Routing Problem, LECTURE NOTES IN MECHANICAL ENGINEERING Vehicle and Automotive Engineering 4, 667-677. **SCOPUS INDEXED Q4**
[P/12] A Agárdi (2023). Analysis of the Multi-Objective Optimisation Techniques in Solving a Complex Vehicle Routing Problem, LECTURE NOTES IN MECHANICAL ENGINEERING Vehicle and Automotive Engineering 4, 678-693. **SCOPUS INDEXED Q4**
[P/13] A Agárdi, L Kovács, T Bányai (2021). Neutrality of Vehicle Routing Problem, INTERNATIONAL JOURNAL OF PERFORMABILITY ENGINEERING 17 (10), 848-857. **SCOPUS INDEXED Q4**

in hungarian journal in a foreign language:

[P/14] A Agárdi, L Kovács, T Bányai (2019). Investigation of Convergence Properties of Ant Colony Based Algorithms, ADVANCED LOGISTIC SYSTEMS: THEORY AND PRACTICE 13 (2), 5-20.
[P/15] A Agárdi (2022). Software for the generalization of the Vehicle Routing Problem, PRODUCTION SYSTEMS AND INFORMATION ENGINEERING 10 (3), 41-52.

<u>in hungarian journal in hungarian:</u>

[P/16] A Agárdi, L Kovács, T Bányai (2019). Időablakos járatszervezési probléma megoldása populációs heurisztikus algoritmusokkal, GÉPGYÁRTÁS 58 (1-2), 17-25
[P/17] A Agárdi, L Kovács, T Bányai (2018). Az időablakos járatszervezési probléma optimalizálása, GÉPGYÁRTÁS 57 (1-2), 6-14
[P/18] A Agárdi, L Kovács, T Bányai (2020). Fitness landscape elemzési technikák áttekintése, MULTIDISZCIPLINÁRIS TUDOMÁNYOK: A MISKOLCI EGYETEM KÖZLEMÉNYE 10 (4), 214-218.
[P/19] A Agárdi (2019). A többlerakatos időablakos járatszervezési feladat matematikai és ontológiai modellje, MULTIDISZCIPLINÁRIS TUDOMÁNYOK: A MISKOLCI EGYETEM KÖZLEMÉNYE 9 (4), 293-300.
[P/20] A Agárdi (2018). A többügynökös utazó ügynök probléma megoldása lokális optimalizálással
MULTIDISZCIPLINÁRIS TUDOMÁNYOK: A MISKOLCI EGYETEM KÖZLEMÉNYE 8 (1), 3-8.

<u>II. Conference paper</u>
<u>in a foreign language:</u>

[P/21] A Agárdi, L Kovács, T Bányai (2019). Optimization of Complex Vehicle Routing Problems, 12th International Doctoral Students Workshop on Logistics, 107-112.
[P/22] A Agárdi, L Kovács, T Bányai (2019). Optimization of automatized picking process, SACI 2019 : IEEE 13th International Symposium on Applied Computational Intelligence and Informatics : PROCEEDINGS, 364-369.
[P/23] A Agárdi, L Kovács, T Bányai (2019). Vehicle routing in drone-based package delivery services, Solutions for Sustainable Development: Proceedings of the 1st International Conference on Engineering Solutions for Sustainable Development (ICESSD 2019), 151-159.
[P/24] A Agárdi, L Kovács, T Bányai (2020). The mathematical and ontology model of the Two-Echelon Vehicle Routing Problem with Time Window, 13th International Doctoral Students Workshop on Logistics, 23-30.

<u>in hungarian:</u>

[P/25] A Agárdi, L Kovács, T Bányai (2019). A hangya kolónia optimalizáció hatékonyságának vizsgálata a járatszervezési probléma megoldásában, Műszaki tudomány az Észak-kelet Magyarországi Régióban 2019 : konferencia előadása, 5-8
[P/26] A Agárdi (2018). A többlerakatos járat szervezési probléma optimalizálási lehetőségei, 15 éves PEME XVI. PhD - Konferenciájának előadásai, 5-14
[P/27] A Agárdi (2018). Optimális lerakat meghatározása a több ügynökös egy lerakatos utazó ügynök probléma megoldásában, Tavaszi Szél 2018 = Spring Wind 2018 : Tanulmánykötet, 525-534

III. Other scientific works:

[P/28] A Agárdi (2019). A járatszervezési probléma megoldása autonóm, elektromos járművek esetén, DIÁKTUDOMÁNY: A MISKOLCI EGYETEM TUDOMÁNYOS DIÁKKÖRI MUNKÁIBÓL XII, 50-55
[P/29] A Agárdi (2018). A több ügynökös utazó ügynök probléma megoldása lokális optimalizálással, DIÁKTUDOMÁNY: A MISKOLCI EGYETEM TUDOMÁNYOS DIÁKKÖRI MUNKÁIBÓL XI, 53-58
[P/30] A Agárdi (2018). A több lerakatos, periodikus, időablakos járatszervezési probléma optimalizálása, Diplomamunka, Miskolci Egyetem
[P/31] A Agárdi (2017). Klaszterezési és evolúciós technikák alkalmazása az utazó ügynök probléma megoldásában, Szakdolgozat, Miskolci Egyetem

Author's web resources

Appendices:

[A/1] Meaning of the components of the general Vehicle Routing Problem model, the ontology model and evaluation of the representation mode:
https://drive.google.com/file/d/1qBL1AiwFd1U4wQ6kgEML0o7Amzj_cja1/view?usp=sharing
[A/2] Additional components of the ontology system:
https://drive.google.com/file/d/1fg2aXSRKkgkS_w9TG-VsgqqZDdxVTbfI/view?usp=sharing
[A/3] Test results of the ontology knowledge base:
https://drive.google.com/file/d/1MFMryefOadfkCDoheFZxnAYrIH9b1h_Z/view?usp=sharing
[A/4] The representation of the Vehicle Routing Problem in the literature:
https://drive.google.com/file/d/1wb-H6gUJiCTWeeY1wQmkbCXzPwnLaS5n/view?usp=sharing
[A/5] The representation of the generalized route description model of the Vehicle Routing Problem (examples):
https://drive.google.com/file/d/13R8yHlaf81NbPUOWOsZ65Cj51u5axhRU/view?usp=sharing
[A/6] Optimization algorithms:
https://drive.google.com/file/d/1NlmQD2DfKnYfmc9mcPzhlRwn2BJ8OBhs/view?usp=sharing
[A/7] Time series algorithms:
https://drive.google.com/file/d/1VKL_3stlJy_4VHDAiFycVHQBo86Z_f3e/view?usp=sharing
[A/8] Classification algorithms:
https://drive.google.com/file/d/1k3gdSaW8zCGDoUM93mc90m9iBtzF6YYC/view?usp=sharing
[A/9] Handling fuzzy and stochastic data:
https://drive.google.com/file/d/1A63gQHhb796TZ-flX8b7_tlNOqEXy5ft/view?usp=sharing
[A/10] Single - and Multi-Objective Optimization:
https://drive.google.com/file/d/1RJRLei6kxC87GQZoPbsSuqcz67OsOx59/view?usp=sharing
[A/11] Applications of the general model of the Vehicle Routing Problem: tables:
https://drive.google.com/file/d/1PjtEiBdrr29MCgEy8Qx3qoeOXvctNHL4/view?usp=sharing
[A/12] The tables of the analysis of fitness landscape:
https://drive.google.com/file/d/1fZFQujHEdX3hita-ivDRKxW6HcZhGAVJ/view?usp=sharing
[A/13] Overview of ontology systems:
https://drive.google.com/file/d/1HHbRFCwHzlO7yDJmLptxpeRispsWt3ls/view?usp=sharing

Test results:
[W/1] Multi-Objective Optimization:
https://drive.google.com/file/d/1gy-5n8TBwT8IwcthYJN-4Y1HmhYyUWOl/view?usp=sharing
[W/2] Analysis of the solutions of heuristic algorithms:
https://drive.google.com/file/d/1x48Nbdo2_bF6iNin5fZ9zNLSKTunRBSZ/view?usp=sharing
[W/3] Analysis of the iteration of the iterative heuristic algorithms:
https://drive.google.com/file/d/1SnLvoznz0lBkto0XDn9jZsLcufzHXDZx/view?usp=sharing
[W/4] Analysis of the iteration of the iterative heuristic algorithms:
https://drive.google.com/file/d/1c2EMGmu2UtxjR13R6bKdzUIuGtRlYE05/view?usp=sharing
[W/5] Operator analysis (with walks):
https://drive.google.com/file/d/1ZQisWItH9NUcbj8REhBtNAQbGNQ7b4tc/view?usp=sharing
[W/6] Operator analysis (with walks):
https://drive.google.com/file/d/1VKUL5nhpdxd-ACdQMABcXUdBAp_FXu4L/view?usp=sharing
[W/7] Random walk analysis (with information content analysis):
https://drive.google.com/file/d/1MQFVW1GNhC-o_pa-l4zx6aZHs8OBn99l/view?usp=sharing
[W/8] Analysis of the filtered search space:
https://drive.google.com/file/d/1mPXBOa-8oB1oOoZJvNt7wHDsNv-wWgNf/view?usp=sharing
[W/9] Fitness Cloud:
https://drive.google.com/file/d/1S_Jk9hGA7huE9-M6beRB5-0BAAvOF6rW/view?usp=sharing