T.C.

ALINBAS UNIVERSITY

Electrical and Computer Engineering

USING MODIFIED GREY WOLF OPTIMIZATION TO SOLVE TRAVELING SALESMAN PROBLEM

Hamzah Ghazi Abdulfattah

Master Thesis

Asst. Prof. Dr. Sefer Kurnaz

İstanbul 2019

USING MODIFIED GREY WOLF OPTIMIZATION TO SOLVE TRAVELING SALESMAN PROBLEM

by

Hamzah Ghazi Abdulfattah

Electrical and Computer Engineering

Submitted to the Graduate School of Science and Engineering

in partial fulfillment of the requirements for the degree of

Master of Science

ALTINBAŞ UNIVERSITY

2019

DEDICATION

A huge appreciation to almighty god for supporting me in finishing my thesis. The one who gave me gave me everything I wish for, and was my main inspiring to get to the top. And a big thanks to my dad who I ask god to give him a long healthy life who supported me in everything that he possibly can. He was my main inspiring in this life and my first school. And also a big thanks to my mother who cared about me in anything and from everything. And she always wished me the best. She was just like home to me. And to my brothers and sisters who supported me in everything in my life. And also I give my appreciation to Asst. Prof. Dr. Sefer Kurnaz who was my light in the darkness and was library of knowledge and my support in information. And to all the professors and teachers in computer and electronics engineering department. And to everyone who believed in me and my ability in doing this thesis.

ACKNOWLEDGEMENTS

I wish to express my acknowledgements to my supervisor, Asst. Prof. Dr. Sefer Kurnaz who was abundantly helpful and offered invaluable support with his sincerity and belief in me.

ABSTRACT

USING MODIFIED GREY WOLF OPTIMIZATION TO SOLVE TRAVELING SALESMAN PROBLEM

Hamzah Ghazi Abdulfattah

M.Sc., Electrical and Computer Engineering, Altınbaş University,

Supervisor: Asst Prof. Dr. Sefer kurnaz

Date: March 2019

Pages: 59

The main problem of the network is that the wrong designs and incorrect connection lead to major problems such as cost impact problems and network efficiency, the most famous being Fiber Optical network problems that still considered a problem for various types of networking. Travel Salesmen's Problem (TSP) is one of the most traditional methods for solving this type of problem, depending today on optimization. Modified algorithms may be regarded as one of the most resourceful and effective ways to solve animal-based TSP problems. Grey Wolf Optimization (GWO) and Genetic Algorithm (GA) focuses this paper on TSPLIB-type issues. This work showed a greatly promising performance with Gray Wolf Optimization and was better than the genetic algorithm.

Keywords: Genetic Algorithm, Gray Wolf Optimization, Travel Salesman Problem, Euclidean distance, optimization.

ÖZET

SEYAHAT SALESMAN PROBLEMİNİ ÇÖZMEK İÇİN MODİFİYE GRİ KURT OPTİMİZASYONU KULLANMA

Hamzah Ghazi Abdulfattah

M.Sc., Electrical and Computer Engineering, Altınbaş University,

Danışman: Asst Prof. Dr. Sefer kurnaz

Tarih: March 2019

Sayfalar: 59

Ağın ana sorunu, yanlış tasarımların ve yanlış bağlantının maliyet etki sorunları ve ağ verimliliği gibi büyük sorunlara yol açmasıdır. Bunların en ünlüsü, çeşitli ağ türleri için hala bir sorun olarak görülen Fiber Optik ağ sorunlarıdır. Seyahat Satıcılarının Sorunu (TSP), bugün optimizasyona bağlı olarak bu tür bir sorunu çözmek için en geleneksel yöntemlerden biridir. Modifiye edilmiş algoritmalar, hayvan bazlı TSP sorunlarını çözmek için en becerikli ve etkili yollardan biri olarak kabul edilebilir. Gri Kurt Optimizasyonu (GWO) ve Genetik Algoritma (GA), bu çalışmayı TSPLIB tipi konular üzerinde yoğunlaştırmaktadır. Bu çalışma Gray Wolf Optimization ile büyük umut vaat eden bir performans gösterdi ve genetik algoritmadan daha iyiydi.

Anahtar kelimeleri: Genetik Algoritma, Gri Kurt Optimizasyonu, Seyahat Satıcısı Sorunu, Öklid mesafesi, optimizasyon.

TABLE OF CONTENTS

Pages

LIST OF TABLES

Pages

LIST OF FIGURES

Pages

1. INTRODUCTION

It doesn't have to be so evident, but it is a difficult problem for the travel vendor to optimize. This is a mathematical problem which generalizes the ability of a set of vertexes to find the shortest route around all points.

- N points and paths are known in length (that means that the shortest route between two is easy to find). The aim is to find the shortest route to be followed simultaneously. The objective is therefore to find a round trip as fast as possible.

It's not the easiest way to find the shortest round trip – you find the perfect round trip. There's a big problem. In that case, all possible ways between all points must be tested, and this is exactly the problem. Such a solution even a "layman" must take an unusual amount of time. The problem is not that complicated with a few points (e.g. 4 or 5). Some arithmetical experts can resolve this problem, though they can solve it with a pen, paper and even without instruments or help. If there are more points (e.g. 8 or 9), even this algorithm can be used in order to test all possible options. What if thousands or tens of thousands of travelers are to solve the problem? Can the calculation be solved with the actual computer (in real time when the computer is going to calculate for a long time)?

In Table 1, if you have only ten points with the salesman's problem, you can see that the possible number of ways is really large. The reader is invited to try to figure out how long it takes to experiment with everything possible for "just" 20 points. The quantity of calculations is so great that it cannot be solved in actual time. History has led to the finding of a universal algorithm for optimizing this issue by many scientists and others.

Table 1.1: Count of TSP resolution permutations with accurate algorithm

Composition of points	Complete possibilities
4	24
5	120
6	720
7	5040
8	40320
9	362 880
10	3 628 800
11	39 916 800
12	479 001 600

1.1 SALESMAN PROBLEM THEORY

A tour salesman needs to come back to all his customers as quickly as possible. Hamilton models the way that visits all points of the graph. Graph G is the G-graph sequence with all graph G points. G-graph sequence. The path of Hamilton and the cycle of Hamilton are defined, because both are specific Hamilton cases. Figure G is the Hamilton graph if the Hamilton cycle is found in Figure G. The aim is to find in the Hamilton graph a short cycle / the shortest sequence closed.

The traveling salesman problem has an optimization version of NP-hard (NP means non-polynomial) issues. It is uncommon to find an optimal solution in real time for every input. If there can be a complex polynomial algorithm.

1.2 HEURISTIC ALGORITHMS

A way to solve a problem that is not entirely accurate but can be found in a (real) short time. The search process is not the exact solution. Heuristic algorithms are usually used when other and better algorithms (e.g. exact algorithms) cannot be used to provide a general proof optimum solution.

With the advent of computers, heuristic algorithms have been started. Even very complicated problems are still being developed and used to solve. Useful for solving such functions, for example with numerous parameters, complex and extreme processes. They are used when the most important drawback of the precise algorithms is not time-useful. There are often ways to solve heuristically and accurate behavior. The fundamental difference between heuristic and precise algorithms is in this period that heuristic algorithms are not optimal and offer (more or less) approximate solutions.

Some problem may not have an accurate algorithm. Exact algorithms are so demanding and in polynomial time they can't solve the problem. In the case of a sales representative who is traveling, the exact algorithm means all possible permutations to be tried, however it really demands the calculation time even if it is a problem with a couple of points.

1.3 HISTORY

Laporte (2006) is the basis for this sub - chapter. The roots of the problem of traveling salespeople are unclear. Undoubtedly, there was a problem for people worldwide when they visited more sites and walked the shortest road. This problem started to be resolved between the 18th and 19th centuries at scientific level. In 1835 instruction was developed for travel agents from Germany and Switzerland, with practical examples. No mathematical documents or other supporting documents were contained in this manual.

First, around 1800, the Irish mathematician W formally defined the traveling salesman problem. Kirkman Thomas R. R. British math and Hamilton. Hamilton developed a kind of game called Hamilton to solve the traveling supplier’s problem. This game has a simple principle, with the

"player" having a certain number of points on the circle of Hamilton. About 1930 professors at the Universities of Vienna and Harvard began to learn about this problem as it took shape. This issue became popular not only in Europe, but also in the USA in the 1950s and 1960s in scientific circles.

This issue has been turned into a linear programmed, and a method "slicing planes" has been developed to resolve it. With this new method, the problem of the travel salesman in 49 cities could be solved with an optimum solution, which enables this method to reach all points as soon as possible. This has been investigated in the past decades by many other scientists, mathematicians, computer scientists, physicist and many others. The problem of the Hamilton circle was proven by Richard M. Karp to be NP - hard in 1972. This detection made it harder to find the optimal route. In the 1970s and 1980s, the method and branch and linking of' cutting planes,' in 2.392 city centers, were important developments when Grötschel, Padberg, Rinaldi and team settled this issue. Applegate, Bixby, Chvatal and Cook developed a program for solving the problem in the 1990s. In 1991, Gerhard Reinelt published TSPLIB with examples of various travel agent problem solving issues. Many researchers around the world use these examples and solutions are shared.

Cook and team presented a solution for 33,810 municipalities in 2005. In this time it was the greatest (re)decided problem. They ensured that the solution to any other kind of problem would not be lower than 1 percent compared to the optimal tour.

1.4 CURRENT SITUATION

Even in this time the number of unbelievable experiments to solve the problem of the traveler and not only computer usage is very interesting. The problem with a bee can be solved one of the most interesting ones. Even when he first read it, the author of this thesis was really surprised. The principle behind this experiment was the rearrangement of flowers and adaptation of bees to new situations. The shortest route every bee wanted to fly. Naturally it cannot be accepted as an excellent scientific experiment, but with such a weird experience new inspiration and ideas can also be gained, in order to search for possible algorithms.

1.4.1 Optimal Solution

As the thesis author has said, it is unknown whether any common travel dealer problems can be solved in an optimum way by a complex polynomial algorithm. This algorithm cannot even be simply confirmed or rejected by the intelligent people of our age. This problem has long been solved and still stands for perfection, even when good progress is achieved (optimum algorithm).

1.5 FAMOUS SOLUTIONS

The thesis addresses possible and known algorithms to solve the problem of traveling salespersons but even an algorithm for the resolution is presented by the author. This section consists of a few algorithms known and used.

1.5.1 2-approximation Algorithm

In polynomial time, this algorithm resolves the problem of a metric soldier. The main idea is to build a bigger tree. The cost (minimum ring) definition is called "altocost," because there are U1 limits of the minimum span and U- circle limits. In the second step, the first algorithm on this tree is researched in depth and all routes are saved over all vertexes – some are handled twice. The final step means that from step 2 through all vertexes, all duplication is ignored. A cycle will be made by it. The triangular inequality is also valid here, and the circle cost will therefore be max twice the initial tour.

1.5.2 Christofides Algorithm

The solution of this algorithm is maximum 1.5 times the optimal solution to solve the traveling salesman's problem. This is not a "free" solution. Real figures show that this solution is not much better than a 2-related algorithm and it is very demanding to apply. The first step is to create a small tree spanning the diagram and construct the entire diagram using a careful search for the first algorithm from the random vertex with a peculiar number of borders. The second step is to ensure these new edges are consistent with the minimum range. This graph is known as the graph of Eulerian. This means that there is a move containing all the edges straight away. Eulerian's move is the last step.

1.5.3 Nearest Neighbor

It is usually known as the Greedy algorithm. Perhaps the easiest algorithm, but not good solutions. The algorithm principle is very straightforward. The first vertex is selected and the next vertex is iterated until all vertexes have been chosen. It is very easy to apply and depends on the choice of first vertex for the solution's quality. Many upgrades to this method are currently known.

1.5.4 Random Improvement

A short tour with up to 700 or 800 vertexes is possible close to an optimal example tour. You can use this procedure. Such algorithms can be said to be the best in the current time. A 100 000 vertex problem can be resolved in real life. The basic principle is arbitrary, four vertexes are chosen, order changes and a new tour takes place. This algorithm searches for a minimum local level. With thousands of vertexes this algorithm can help solve this problem.

1.6 FUTURE

At least it is not clear what this problem will be in the future. However, as has been said, it is not yet possible to find someone with a complex polynomial algorithm to resolve this problem. In order to find the best solution, numerous organizations throughout the world organize competitions and grants every year. At present, the IT industry's development is so rapid and software system developments go hand in hand, not excluding the travel salespeople's problem. It is possible to compare how powerful computers were ten years ago. Computers can be said to be stronger every day. Perhaps we need to consider how powerful a computer may be in the (close) future, so that only without an accurate algorithm of revolutionary optimization can the problem with tour dealers (and many more) be solved. This is only a speculation and the author of his thesis is probably unable to agree to scientific fiction and many learned people.

2. LITERETURE REVIEW

2.1 NP-COMPLETE ROUTE QUERY PROBLEMS

To a certain extent, the structure of this survey was inspired by [59] which was taken from a book.

2.1.1 Problems of Vehicle Routing

In the last three to four decades the scientific community has paid immense attention to the Vehicle Routing Problem (VRP), which often plays a key role in designing distributors.

The VRP tackles the problem of roads design for several trained vehicles that are least expensive for a wide range of spread customers. Often a common home base, namely the depot, is assumed by vehicles. Travel expenses shall be calculated between the customer and the repository and each customer. In real - life circumstances, limitations like time windows are important lateral constraints on the problem. Include, for example, goods and pick outs, school bus service, road cleaning, the transportation of persons with disability, salesmen's and maintenance operations. Typical real-world applications include

For the vehicle routing problem, several objectives may be considered. Typical objectives are:

- Reduce global cost of transport, depending on traveling time or distance.
- Reduced the number of vehicles to provide service to all clients.
- The vehicle's road balance, time and load.

More than 45 years ago, Dantzig and Ramser [25] launched the VRP. The first mathematical and algorithmic problems were described in a real - world application. Some years later the effective greedy heuristic suggested by Clarke and Wright [19]. As a consequence, many models and accurate and heuristic algorithms for a solution of the various versions of the VRP were present. There are already a number of surveys [1, 4, 11, 17, 26, 31, 59, 62] which are used to provide a thorough overview of this fertile area of research to readers of such surveys.

2.1.2 Other VRP Classes

The purpose of this section is to present some of the key issues with regard to routing vehicles.

Traveling Salesman Problem, the traveler's problem is one of the most important VRP versions. This problem has been researched over decades because of its simplicity and applicability. We mainly concentrate on this VRP class in this thesis. In Section 2.1.3, we present this issue in more detail. We examine three different approaches used to cope with this problem in Sections 2.2, 2.3 and 2.4. We refer readers to [54], [27] and [57] for further explanation for the TSP.

Compared with TSP, it is heuristically and precisely much harder to solve more general problems like capacities routing or pickup and delivery problems with time windows.

Problem of Capacitated Vehicle Routing, VRP is training in CVRP as a basic version. He is the simplest and learned family member. All customers know the CVRP beforehand. The vehicles are identical and installed in one main repository, with the vehicles only having capacities limits. The goal is to reduce the overall cost of customer service.

Distance-Constrained Vehicle Routing Problem, for each route a maximum length (or time) restriction is repositioned by a CVRP primary variant-DVRP (distance controlled vrp). Specifically, each arc cannot have a total length of the maximum length of the vehicle. If the cars are different, the maximum length may vary. The matrices usually match the cost and the length; therefore, the difficulty is to minimize the total length of the route. The problem is called Distance Contracted CVRP (DCVRP), where both vehicle capacity and maximum distance constraints are considered.

Vehicle's Time Windows Routing Problem, the VRP Time Venue (VRPTW) is the CVRP extension, where capabilities are restricted and a time interval [ai, bi] is associated with every customer. The service must be started for each customer and the vehicle must stop for a period of

time at the customer's location. The car should come and go in the window. The vehicle can wait until the instant to be served by that customer when the customer arrives at the location early.

The Attended Home Delivery Services [2] are one of VRPTW's current applications. Depending on the loss (for instance of foodstuffs), extent of the goods (for example furniture) or the provision of a service (for example repair or installation), it may be necessary to deliver. Goods security (e.g. electronics).

Pickup and Delivery Problem, VRP Pickup and Delivery (VRPPD) is a subset of road problems. Every Customer I in this category of problems has two quantities, di and pi. For each customer I sometimes only one request is used. di-pi (no place). di-pi. The problem is that roads are to be found for pick-ups and deliveries and that the same vehicle is to be collected and delivered on request. There are a number of further constraints, with the most typical limitations on capacity and timescales.

Heterogeneous Vehicle Routing Problem, The following features have been modified in particular:

The fleet contains an unlimited number of cars per type. Fixed vehicle costs will not be taken into account and automatic routing costs will be taken into account. A literature overview of the approaches to the resolution of heterogeneous VRPs is provided by the Baldace et al. survey [5]. They classify the different versions of heterogeneous VRPs and because there is no accurate algorithm, especially as described in the literature. The lower limits and heuristic algorithms are also examined.

Most built-in VRP formulations use binary variables to determine whether a car travels as best as possible between two customers. It combines assignment limitations, modeling of vehicle routes, restriction of commodity flows and modeling of commodity movements with decision variables. When you increase the VRP Set Partitioning (SP) model and provide a viable binary variable, you can also receive a major formula for heterogeneous VRPs.

Dynamic Vehicle Routing Problem, a complete Dynamic Vehicle Routing Problem (DVRP) survey is conducted by Madsen et al. [46]. In the last few years the majority of new vehicles have state-of-the-art GPS/GIS systems, due to technical developments. Therefore, distribution companies are in a position at all times to monitor vehicles' position and status.

The fundamental VRP is for clients with previous design knowledge. In addition, other information, like driving time between customers and customer service times, is known before schedule. This ensures the full configuration of advanced mathematical optimistic methods such as setup. However, information is often uncertain or even unknown when planning for real-life applications. It may be said that the traditional VRP is both static and disruptive. The DVRP, however, considers the VRP, in which after the day of operation there is a subset of customers (or the whole set). The DVRP will need to consider how the new applications can be included in the already designed routes.

Two types of requests are used in many DVRP's:

1. Advance demands that can also be called static customers as these service requests were received before the routing process began.

2. Immediate requests, also known as dynamic customers, will be made during the route execution in real time.

Ideally, the planned routes should integrate new customers with the minimum time without changing the order for non-visited customers. The introduction of new customers in practice is however often a much more difficult task and involves either a partial or a comprehensive overhaul of the unavailable part of the route.

Inventory Routing, from [9] we encouraged the following description. One major and challenging extension to car routing problems is the problem of inventory routing in order to control inventory and routing decisions simultaneously. The aim is to reduce overall costs, such as stock and shipping costs, while avoiding inventories and storage capacity. There are numerous

inventory routing problems in the literature and they have some common features. All inventory routing issues have certain basic features. Everybody considers the stock of products to be supplied to one or more customers by a qualified vehicle supplier. The cost of the cars is defined by their distance. It is part of the goal role. To avoid any inventory, the supplier must manage its customers ' product stock. For example, Bertazzi et al [9] noted different other features that may alter the routing stock structure:

- The horizon of planning can be endless or endless;
- The cost of holding an inventory may or cannot be taken into account;
- The costs of inventory holding may be borne solely by the supplier, the supplier and clients or customers only;
- Deterministic or stochastic production and consumption rates may be.
- Manufacturing and consumption take place at discreet times;
- Rates of production and consumption over time or time are constant;
- The optimal supply policy can be selected from any policy or must be chosen from a certain policy category.

2.1.3 Traveling Salesman Problem

The question is how to define different versions of the distances between towns. The problem is said to be symmetrical in that the distance from the town I is the same as the distance from the city I for all towns I and j. The problem is said to be asymmetrical if the property is not present. The Euclidians are the problem if the towns are located within the Euclid distance and between two towns. We will concentrate on symmetrical TSP in this thesis. But also for asymmetric TSP. Our work is suitable. For many practical purposes, such as x-rays [9, 10] or the making of the VLSI chip [41], a symmetrical TSP is useful. As a mathematical model, TSP can be formulated. The number G= (V, A, w) was given. We define the xij variable of binary decision which is set only with the arc I j).

The triangular inequality is an important restriction often placed on instances. This applies in particular to all I j, k, 1— t, j, k— t, d (ci, cj) — tu (ci, d) — t— d (cj). It says that the shortest road is always the direct path between two towns. Most theoretical papers on TSP heurism suppose that the inequality of the triangle exists.

More than one salesperson, called MTSP (m-TSP), is generalized through the TSP process. The m-TSP offers n cities, retailers and one store or base. In one of the m-tours from and to the repository, each city should be visited exactly once. There are no free tours. The triangular inequality can be easily determined if the route TSP is less or equal to any m from the shortest m-TSP solution at cities n+ depot.

Every n town m - TSP is a TSP m+ n town. First, in the repository node, you create m copies. The deposit nodes will be defined as a sufficient number and the distances between repository nodes and common nodes will be copied from the TSP m. There is no vendor tours because of the large distance between the nodes. TSP does not meet the inequality of the triangles. Due to the close connection of the m - TSP with the TSP the literature was not thoroughly studied. Bektas studied the problem literature and its precise methods [6].

TSP has been studied and is one of the worst NP issues to be addressed. More general problems with routing, such as the capacity vehicular routing problem or time windows collection and delivery problems, are much more difficult than the TSP. Solution methods for the TSP mean that solutions for more general problems are developed significantly. In the three following sections, we describe the three different approaches to the TSP.

2.2 EXACT METHODS

The optimum solution guarantees exact methods when time and space are enough. No simple listing is possible, so smarter techniques are required in precise methods. The worst - case time for NP - hard issues is still high. We cannot expect to build accurate algorithms in polynomial time to solve NP-hard problems, except NP= P. For some sort of problems, it is hoped that algorithms can be found that solve problems in practice in reasonable time.

2.2.1 Branch and Bound Algorithm

The entire solution room is searched for the best solution by a branch and a bundled algorithm. Explicit listing, however, is usually impossible as the number of possible solutions increases

exponentially. The algorithm searches for the solution implied by the use of limitations for optimizing the feature with the best solution.

A proven upper and lower limit for the ideal objective value (globally), the non-heuristic branch and the bonds retain their certification that shows the suboptimal point to be sub-optimal. Yet the slow (and often slow) branch and binding algorithms. Efforts, but methods converge in some cases with much less effort, are needed in the worst case.

The industry and binding methods in TSP generally depend on the relaxation of the TSP integrated linear programming model. A proper relaxation, e.g. by removing the restrictions on removal of a subtour, determines the lower limit for the cycle length. This results in a task issue and is called relaxation of the assignment. The task problem for the mathematical optimization or operational research area is one of the fundamental challenges. It is a diagram that divides vertices into two U- and V - sets to connect the vertical U vertex to the vertical V. The U and V sets are distinct. The weight matches maximum in one bipartite weight graph. The problem is as follows in its most general form:

A number of officers and various tasks are involved. Any agent is able to carry out any task, with certain costs that may vary according to the assignment of the agent task. All tasks must be done in so far as the total cost of the assignment is minimized by assigning exactly one representatives to each task.

The branch is simple, but ineffective due to its bordering technique and relaxation. The tree can become too large for sub-problems. Other relaxation measures, e.g. minimum tight tree relaxation, leading to a relatively limited branching, can be used for a stricter lower limit. These and other variants are listed in [42]. The following variants are listed.

The analysis of branches and bonds until the late 1980s was completed by Laporte and Norbert in [43]. Since CVRP generalizes the TSP, numerous precise CVRP approaches are used to find the correct CVRP solution. The degree-limited shortest span tree, such as AP assignment problem, is used by branch and binding algorithms. These algorithms were the most effective exact approaches to CVRP till the end of the 1980s. More complicated limits have been proposed recently, which increase the problems which can be solved by connecting branches and algorithms.

The literature branches and algorithms are not exactly compared, as the authors have either a slightly different problem or resolved a whole new set of instances.

2.2.2 Algorithms for Branch and Cut

"The IP linear relaxation is the linear program obtained via IP since all variables must be integer." IP is the linear program. As indicated under [59]. The optimum ZLP relaxation value is therefore lower to the ideal ZIP value of the linear integer (ZLP ie ZIP) program. The classic way to solve this problem is to link the linear programming if an integer linear program has a small number of constraints to fill in linear relief into a LP resolver. Instead, if the number of linear LP restrictions or exponential sizes is large, it is impossible to supply a limit system to an LP solver and to resolve a linear program using a cutting plane technique. The IP is an integer program and the LP (De) is very limited in linear relaxation.

For h=0, let LP(h) consist of a sufficient subset of LP(demon) as a linear program. The solution is ideal if IP is workable. A blackbox algorithm is known as the separation algorithm, which gives a minimum of one LP(Teing) limit violated by Lp(h) solution if one is available. LP(h+1) then adds the contravened restrictions. For each h > > 0, if ZLP(h) is the LP(h) optimum, it is ZLP(h).

With the regular simplex algorithm the branch and the methods of cutting solve the linear app without the integer limit. When an optimum solution with a non-integer value for an integrated variable is obtained, an algorithm is used to identify additional linear limits that comply with the existing partial solution but violates all workable entities. This will be done until a whole solution (then called the optimum solution) is found or the interconnecting planes no longer found. If this has been found, it will be added to resolve the issue and will hopefully be "less broken". In practice, however, there may not be an accurate separation algorithm and, although there are some, there can be no injustice violation. If the optimum IP solution is not finished, we

are in the industry. The problem is divided into two new problems, namely adding a variable with a fractional current value at the top and bottom. There are two different versions of the problem. The first one has an additional limit of the original variable or equal to it. Each new issue is solved again and again using the same method and the original problem is solved in the best way. At the heart of the branch and the cutting method is the listing with the trimming plane.

Thus STSP was able to find optimal solutions for major situations of close-knit problems. The problem of Symmetric Traveling Salesman However, compared to TSP, branch research and the cutting of CVRP is still very limited. Branch and cut are important issues, namely that the tree produced during the branching process is too wide, and it appears that termination is impossible, in a reasonable time, just as with the branch and bound algorithms.

2.3 CLASSICAL HEURISTICS

Since TSP is an NP problem, finding heuristics quickly is one way to resolve this problem. Heuristics are practices which generally provide a viable and reasonable quality solution relatively quickly. However it can be arbitrarily bad, there are no guarantees of quality of the solution. Heuristics are empirically tested and comments on the quality of heuristic can be made on the basis of this experiment. Because of its speed and capacity to deal with large cases, heuristics are usually used to solve real problems.

Several heuristics families for the TSP were proposed. The classical heuristics, mainly developed from1960 to 1990 and metaheuristics, have grown in the last 10 years. They are divided into two important categories. Most standard buildings and upgrade procedures currently available are first class. These methods scan the search area relatively small and generate usually good solutions in small computational times. Further, the majority of limitations can easily be extended to reflect diversity in real-life contexts. Therefore in commercial packages they are still widely used.

2.3.1 A Good Heuristic Character

In the beginning, we explained four key software transfer and acceptance features, described by Cordeau et al. [20], for end - users. Most heuristics usually rely on two criteria: precise and fast. But simple and flexible characteristics of good heuristics are also important. Those four criteria should be explained.

Accuracy measures the degree of departure from the optimal value of a heuristic solution. Optima is usually unavailable because of the sharp low boundaries. Optima and lower limits are generally not available for the TSP. This is why most comparisons with the best known values have to be made. There are many difficulties in analyzing heuristic results. The author usually reports research results for the best or several rounds of algorithms. When authors do not apply the same rounding techniques to present their results, the problem increases. This could lead to huge discrepancies.

Coherence is another precise problem. Instead of one which can do even better as often as possible, users prefer a heuristic which works well all the time. This solution discredits the algorithm. Users often prefer an early solution algorithm and display solutions of increased quality when performing, which can only be replied to in the end, perhaps after a very long period. Give users a better idea of how many more investments the evolutionary solution value should be taken into account.

Speed, based on plan level and the accuracy of the problem must be calculated. For real-time applications like explicit mail pick-up, delivery or re-deployment, efficient, quick and sometimes near-instant action is needed. On the other hand, spending hours or even days in computational plans every few months in the long term, such as fresh sizes, would not be a problem. Somewhere between those two extremes can be placed most of these applications. It does not appear unreasonable to calculate 10 or 20 minutes on a daily routing problem. Of course, interactive systems have to react much faster.

Simplicity, Some heuristics are not commonly used since they are too difficult to comprehend and code. In addition, heuristic systems have to be sufficiently robust to make sure that they function properly even if not in full detail. Many algorithm descriptions are not sufficient or too comprehensive. Simpler, preferably short and self-contained codes are more likely to be adopted, although good results can be expected to be achieved with minimum complexity.

It is difficult to understand and unlikely to use algorithms which contain too many parameters. The majority of meta-heuristics developed in the last 20 years have experienced this problem. In its algorithms, the number of parameters increased considerably beyond what is acceptable, especially during the testing process in relatively few cases. In search of better solutions. There must be no limit to the number of parameters in the Algorithm but also for the end-user. There are two easy ways of generating parameters, as suggested by Cordeau et al [20]. One value of this kind must be defined for good, in particular when testing shows that the algorithm is very uncomfortable with a selection of certain parameters. The automatically adjusted parameters can also be used during the entire algorithm.

Flexibility, In many real - life applications a good heuristic should be flexible enough for various lateral restrictions. Most documents concentrate on precision and speed in the literature and the manner in which further constraints can be resolved is not clear. The result can influence the algorithm performance.

Conventional TSP heuristic systems can usually be divided into two categories: construction heuristics gradually produce a workable solution, taking into account the cost of the solution. Improvement methods try to upgrade any viable solution via edge or vertex sequence exchange. But, in some constructive algorithms the distinction between constructive methods and enhancement methods is frequently blurred.

2.3.2 Constructive Methods

The heuristics are designed to proceed until a viable solution is found from the start. This usually involves a gullible algorithm. In addition, constructive algorithms are important since the initial tours required by local search algorithms can be generated. "In the case of the TSP, many heuristic touring tours are surprisingly good in practice, because of the successive increase in the number of problems with combined optimisation". Their paper stated Johnson and McGeoch [38]. In relatively little time the best can usually be found in about 10 - 15 percent of optimal.

The remaining four important tours heuristics and their algorithmic conduct are discussed in this subdivision.

Nearest Neighbor, at the beginning of the starting city, the algorithm for this TSP visit is created and added to the city tour not visited by the next town. O(N2) is used as its runtime by this algorithm. NN(I)/OPT(I) N(0, 5)(log2N+ 1) is the best tour quality guarantee. The distance metric, however, meets triangular disparities. But the cases for which that ratio grows as a function have been identified by Rosenkrantz et al [55].

Greedy, the algorithm is entered into a complete diagram of the cities, which means vertices and ends for each pair. The tour creates a rim from the shorter edge and repeatedly adds one of the remaining rims. If you are not yet on a tour and if you add a grade-3 vertex, or a length cycle is less than that of the total number of vertices. With working time - lib (N2logN), you may use this algorithm. You might have noticed that the algorithm is a slower algorithm than the next algorithm. However, the worst case tour quality is Greedy (Log N)/ (3 Log N) [28] for all cases which meet the unequal triangle [48], as in the nearest algorithm to the site.

Clarke-Wright savings heuristic, a more general vehicle routing algorithm [19] is used to derive Clarke-Wright's savings savings heure (Clarke-Wright or simply short CW). We select a random

city as the hub in this algorithm. After each visit to another town, the salesman returns to the hub. We have several graphs and every non - stroke vertex is connected to the hub through two angles. If a salesman goes straight to the other town and passes the hub, let the savings be as large as the tour length for each of the non-hub towns is reduced. On all non-hub city pairs we are calculating savings. In the next step, all savings are ordered and applied. Savings that make up a non-stroke or non-stroke cycle adjacent to more than two other non-stroke tops are not taken into account, like a hateful algorithm. The construction process ends when the hub is linked to only two non-continental cities, so we have a real tour. Like Greedy, it is possible to quickly implement this algorithm (N2 logN). It is currently known as CW(I)/OPT(I) (log2N + 1)'s best-known performance guarantee (factor 2 higher than Greedy) [47]. But Greedy's performance ratios (logN)/(3 log N) are the worst known example [27].

Christofides, the Christofides algorithm has a consistent and, worse still, only three - and - a - half case performance ratio assuming unequal triangularity. In the TSP tour using heuristic Christofides, Johnson and McGeoch described [38] the way to build:

"First, for a set of cities we are building a minimum spanning tree T. Note that a tree's length cannot be longer than OPT (I), as the removal of an edge from the optimal tour results in a tree spanning. Then, on the vertices of an odd grade in T, we calculated a minimum length matching M. A simple argument can be shown if OPT(I)/2 does not exceed this unfair triangle. We have a graph connected to each vertex when M is combined with T. This diagram must be toured by Euler, i.e. a cycle which goes exactly at every edge once and is easy to find. A long tour for a traveling salesman is possible during this cycle while using shorts so that visited vertices do not multiply. The direct road cannot be any more than the path replaced by the triangular inequality. A direct path can be no longer, but only a shortcut replaces the path between two towns.

In practice, this heuristic TSP is superior to the nearest neighbor, Greedy and Clarke Wright, as well as better guarantee in the worst case. However, Christofides takes heuristic times (N3) in comparison to the others three heuristics. Heuristic times are considerable. The heuristic Christofides are amended by Gabow & Tarjan [29] to take the same bad guarantee during the O(N2.5) period. This result has been achieved by using a matching algorithm based on the

scaling and stopping when the match is not greater than 1 + (1/N) times optimal. For those who wish to visit other touring areas, such as Bentley [7, 8], Reinelt [53], Junger, Reinelt and Rinaldi [40] readers can also carry out more extensive investigations.

2.3.3 Improvement Methods

The most popular classic TSP, op, 2-op and 3-op optimizer algorithms are presented in this section. Johnson and McGeoch [38] begins in their papers that they are normally able to choose three to four percent with a simple heurtical choice, and variable "algorithm" for Lin and Kernighan, 1 to 2 percent yardstick (as constructive as methods) of traditional Local Optimization techniques for PSD [43].

Variable-Opt Algorithm, Lin and Kernighan [45] made a new journey involving the replacement of pairs of borders. It's a common K-Opt. In a shorter journey between cities, Lin-Kernighan is adaptive and decides on the number of borders. It starts with an excellent solution, but it is feasible. It's not optimal as k borders are missing on the tour; the tour needs to be optimized with k new edges replaced. From the edges of place and k new edges, K can be easily identified. The difference between the old rim length and the new one rim length is defined in its algorithm.

This begins with a random initial and an interchangeable blanket. To optimize the swap set, you select the most popular pair and add it to the suggested swap set. This step is repeated until no additional gain is achieved (under the appropriate stop rule). In the following step, a range of exchanges are selected and implemented. These two steps are repeated until there is no further improvement.

2-Opt Algorithm, a local TSP upgrade algorithm based on tour changes is regarded as 2 -Opt Algorithm. When the tour is feasible, an algorithm will then repeatedly perform an operation sequence, as long as the tour is reduced by one to the next (a locally optimal tour), until a tour is completed that does not improve operation.

Croes proposed the 2-Opt algorithm [22] first. Two edges are removed by the 2-Opt – Algorithm, which divides the route into two routes and reconnect newly added edges (only if the sum of new edges is less than their length). The first tour was shown on the left in Figure 2.1. Edges 'v1v5' and 'v2v6' have been taken off with edges 'v1v2' and 'v5v6'.

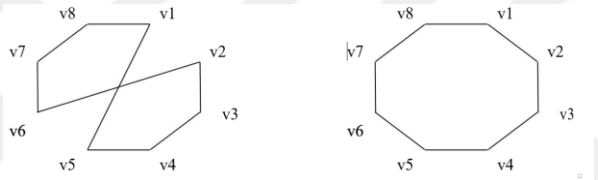


Figure 2.1: The First Left – Hand Tour and Resulting Right – Hand Tour is a 2-op Move.

3-Opt Algorithm, the exchange replaces up to three edges of this tour in 3-Opt [44] as shown in Figure 2.2.

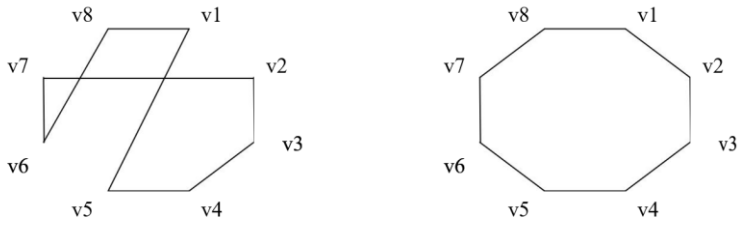


Figure 2.2: A 3-op Move: Original Left - Hand Tour, Potential Resulting Right - Hand Tour.

Theoretical bounds on local search algorithms, these algorithms are limited in terms of tour quality and local search algorithms to the two theorems referred to in section 2.1.3 for arbitration. Papadimitriou & Steiglitz [50] in 1977 showed the local search algorithm which does not take the Polynomial times per shift provided by p / np by A(I)/OPT(i). 2-Opt, 3-Opt and k-

Opt with k<3N/8 is a significantly worse situation. In 1978, they demonstrated that there were instances where there were an optimal single tours but many local tours exponentially, each with an exponential factor that is lengthier than optimal [51]. The result is not applicable to graphs that fulfill the unequal triangle.

In view of the triangular inequality of the 2-Opt, Chandra and others [24] have shown that (1/4) to 1/N and (1/4) to 1/6) are the best guarantees. In general, k - Opt's best guaranteed output is to meet the three - way inequality at least (1/4) N 1/2k. If cities with Rd dots for a fixed d and Euclidean standard compute distances, the worst rate of output is the limited number

In contrast, we can improve the worst-case conduct significantly using a good heuristic to generate our start tours. For instance, the start of the tour with Christofides will be never worse than 3/2 times better if the tour resulted from 2-Opt (as long as the triangle inequality is assumed). Because Christofides algorithm's worst-case ratio is 3/2.

The number of movements on local search algorithms can be significantly large before ideal 2 and 3 opt rounding. For K before stop [24], Chandra et al, for 2-Opt's worse binding case, have k-Opt movements. Ça (2N/2) may be the 2-opt number. But these results are based on a random beginning.

We also need to be conscious of time per step in addition to the number of movements that are time to improve and time to move. The data structure used depends on both. The cost of an assessment is, for instance, to calculate a structure of an array while calculating the cost of performing a change (1). The results show that 2- and 3-Opt algorithms are far better than theoretical limitations could imply, as indicated in [38].

2.4 METAHEURISTICS

During the last two decades a special class of heuristic was metaheuristic. Metaheuristics offers general heuristic frameworks for many problem classes. High quality solutions are often used with metaheuristics.

The focus in metaheuristics is on the most promising space for the solution to be explored thoroughly. In general, these methods combine advanced research standards, memory structures and solutions for neighborhood research that produce high - quality solutions, but increase computer time prices. Moreover, the procedures typically depend on the context and need finely defined parameters, which can make it difficult to reach other circumstances. Metaheuristic methods in one sense are simply sophisticated improvements and can be viewed in classic heuristic terms simply as natural improvements. During the search process, metaheuristics make possible the main difference between conventional approaches to the deteriorating and even inviolable intermediate solutions. Typically the most well-known metaheuristics identify better local optimize than earlier, but also take longer.

The six major types of VRP meta - heuristics are provided by Gendreau et al. [31]. From her article you had inspired the following overview of these metaheuristics.

The initial solution starts with simulated annealing (SA), taboo search (TA), and deterministic annealing (DA). Each time a stop is satisfied, care needs to be taken to prevent cycling, to be moved in the area. At every step, a population of alternatives is examined by genetic algorithms (GA). Each population comes from the first by bringing together the best and the worst. In every iteration, Ant Systems (AS) uses the data collected in previous iterations to create a series of new solutions. Taillard et al. have pointed out [59] that TS, GA and AS are methods for recording and using information on solutions. Neural networks (NN) are an apprenticeship that progressively adapts certain weights to an acceptable solution. The search rules are different and need to be adjusted to the problem. The search rules are different in each case. There is also a lot of creativity and testing. Some of the most representative local searching algorithms are being studied by Gendreau et al. [31]. In their survey they conclude that with several hundred customers the best methods can find good and sometimes ideal solutions, but significant computer costs. The most effective approaches are now offered by taboo search. Pure genetic and neural algorithms are much poorer than other algorithms, while simulated rinsing and Ant systems are not competitive. Hybrid AS and GA can, however, be achieved with the successive implementation of a given approach taking into account performance improvements. In future, since approaches have not been fully used, we could match the effectiveness of existing TS heuristic products. Another observation is that the currently used data sets consist of cases that

are too small to make a clear difference between different implementation of some of these metaheuristics, particularly TS. Data sets are therefore required that correspond to larger instances. You can also ask how these metaheuristics would work in much larger cases, which often occur in practices. Given their computer needs, heuristic devices with these levels of sophistication may not be able, especially where real-time applications are envisaged, to resolve satisfactory large cases in a reasonable time. Metaheuristics are quite time consuming, but in comparison with classical heuristics they offer better solutions. Classical methods usually produce 2% to 10% higher than the optimum (or best known solution value); whereas best methods usually have a value less than 0.5%.

New solutions for traditional approaches like tabo searching, simulated renovation and so on have less room. However, you have at least one new method of genetic algorithm that needs to contribute to pay a large, but O (N2) time - price [38].

2.5 EXPERIMENTAL EVALUATION OF TSP

The work on TSP algorithms has been extensive. The experimental results for TSP are briefly addressed in this section. In [36] the writers discuss algorithms experimentally in a comprehensive informal way. It is based on the lessons learn from experiments, paper writing, arbitrary examinations and lively discussions by the author over the past decade.

2.5.1 Theoretical Results

The theorem of Sahni & Gonzalez offers the behavior of every heurtist TSP a good limitation [56]. The following is the theorem:

Theory: A(I) is a heuristic TSP tour A and I and an OPT tour I which is identical to that optimal to a TSP tour. Theorem theorem. Theorem. Theorem, theorem. Theorem. If P is assumed – A(I)/OP(I) can not be assumed to be 2p(N) for any fixed polynomial p or instance.

They regard no limitations on the instances in the theorem. However, in most applications, circumstances like the triangle inequality are restricted. The authors presented a theorem in [1] that gives TSP heuristics an extremely limited limit. Theorem: If P is раи>0, so that A(I)/OPT(I) cannot be guaranteed by any heuristic TSP as opposed to 1 +¨ if I satisfy a triangular inequality. On the other hand, the size of μ was not even shown.

Based on both of these theorems, how can TSP heuristics guarantee the performance of polynomial time? There are, as Rosenkrantz, Stearns and Lews observed, three simple generations of heuristic trials: dual minimum span, nearest inclusion, and nearest supplementation which are the worst of all, in triangular inequality[55]. In other words, under that limitation they guarantee A(I)/OPT(I) para 2. Although these three algorithms have a good theory in the worst cases, the four tournament heuristics (Next neighbor, Greedy, Clarke - Wright, Christofides) discussed [38] are much looser and better in practice.

2.5.2 Lower Bound

The optimal tour length is often not available to assess the empirical achievements of the TSP method, because the usual tour length is not the ideal for the largest instances. Therefore the authors often compare heuristic findings to what can be calculated in large cases: the lower bound of the optimal tours of Held and Karp [33]. They define 1-tree as a tree that has vertex set in their paper {2, 3... n}, along with two different vertex edges 1. Their approach makes it easy to calculate a minimum 1-tree. Each vertex is only one tree, and each tour is a low tree journey if the weight of one tree is the least. The minimum tour weight is equal to or above the minimum 1-tree weight. Their bond is generated by linear TSP relaxation programming. It can be precisely calculated by linear programming in moderate size instances. But it is not trivial, because the number of limits of the linear programme, in the number of cities, is exponential. The Held-Karp connections seem to be a constant approximation of Johnson and McGeoch's optimum tours [38]. Given the three-way inequality [63, 58], the Held-Karp boundary can be no less than (2/3) OPT(I) from the worst possible perspective. In reality it is usually much better even if the triangle inequality has not been observed.

2.5.3 Experimental Results

Until 1980, 318 cities were resolved in the largest TSP instance [23]. In 1987 the TSP proceeding with 2392 cities was resolved by Padberg & Rinaldi [49]. In 1994, Applegate et al.[3], taking 3-4 years of computing, could find the optimum TSP tour in 7397 cities, for example. Applegate, Bixby, Cvátal, Cook and Helsgaun research team has worked on the largest Euclidean instance, comprising 24,978 cities [65) with the branch-and-cut method. For examples as large as 85900 cities the same authors find the optimal TSP tour. In a case with over 1.9 million cities, they have reportedly applied heuristic methods for the TSP [66].

In the field of TSP heuristics, the 8th DIMACS Implementation Challenge is a great source of advanced algorithms. This challenge is aimed at continuously updating the newest TSP heuristics industry. The authors give extensive Heuristic Experimental Analyses for the symmetric and asymmetrical TSP in both [39] and [37]. These findings show that, as compared to the optimal results of the TSPLIB instance, the typical excess carp excess is under 1 percent.

The high level description of the algorithms and the results is a disadvantage of the information provided by the published papers. It is difficult, especially when time is compared, to compare different algorithms. In addition it is too pessimistic to tell us much about typically algorithmic behavior in the worst case of many theoretical outcomes. Taking these problems into account, Johnson and McGeoch [39] are carrying out a thorough TSP case study. The paper focuses on the relation between what they studied theoretically and what can be empirically observed. You have succeeded in providing the reader with a more generally applicable source of solutions. The findings in their papers demonstrate that the Christofides algorithm is about 9.5-9.9% lower than the one of Held Karp. The resulting TSP tour is in the range of 9.2-12.2% for Clarke-Wright algorithms over Held karp. Greedy took about 14.2-19.5% of the TSP rounds through the lower Held-Karp and the TSP tours found the nearest algorithm over the lower Held Karp with 23.3-25%.

You are all supposed to provide a full graph for state-of-the-art algorithms. No complete chart was found for the TSP. No work. No work. As a result, only a fraction of our experimental results can be compared using 100 percent of the diagram's edges. In this thesis, however, our

main focus is to see how much the tour quality reduces compared to the full diagram of a subset of borders.

3. METHODOLOGY

3.1 SOFTWARE

This makes it difficult, especially when comparing times, to compare different algorithms. The author has many experiences, for instance, from the Faculty of Management Science, the University of Zilina. The programming language has been chosen. Java-development of languages and apps, data structure 2, discrete simulation, etc. The software can be used on all other platforms (Windows 7,Windows 8 and Ubuntu features tested etc.), as the language Java can be used on many platforms.

Netbeans, Netbeans is a development kit for open source software. The development environment is relatively fast and offers many opportunities for not only developing programs for Java (the main purpose), but many more (e.g., PHP, C++, etc.). This is a Sun Microsystems (Oracle) product. This is a product. This environment complies with commonly recognized standards and can be used on any commonly used Java virtual machine operating system. Netbeans are also available for the graphical user interface designer, which can say that a nice user interface with a little "click" has been created.

Java, java is a programming language oriented towards objects. The American company Sun Microsystems, now known as Oracle, developed and introduced Java on 23 May 1995. The Java programming language is one of the most popular and used. It is also used for mobile and mobile devices (Wiki Ubuntu) and is not just used to build desktops and notebooks. Java is the primary programming language for the platform, and has boomed in recent months and years thanks to its new Android platform.

It's very simple to control the basic functions. The user must select the data input file of the computer, all points shall be drawn automatically by the given co - ordinates, and the distance matrix shall be calculated. After that, users can select which algorithm solves the input data problem of the traveling salesman. Once all the algorithm steps are complete, the app opens a tab

with a graphical display showing a solution with all the points and the final tour. The last distance, point sequence, time and algorithm time of processing is displayed in the "Results" tab.

3.2 INPUT DATA

For this thesis, the input data is actually necessary. All algorithms have a good source of test data, the http website / www.tsp.gatech.edu/. This is a website aimed at solving the travel seller problem. Infinite information can be found throughout the world by every researcher, other scientists contacted, their example, their algorithms and their solutions shared. It has been chosen as real examples of true states with the best solution aren't only fictional examples. The best solution for each investigator can be shared. This information is useful to evaluate powerful algorithms implemented in the last chapter.

Structure of input data, the input data structure remains unchanged. Each thesis data comes from http/www.tsp.gatech.edu/. Firstly the ID for the vertex, secondly the x-coordinate and thirdly the y-coordinate for the vertex. A sample of input data is shown under this paragraph. This example is from the vertex site with 10 639 vertexes from Finland.

```
5883 62466.6667 29050.0000
5884 62466.6667 29200.0000
5885 62466.6667 29250.0000
5886 62466.6667 29916.6667
5887 62466.6667 30000.0000
5888 62466.6667 30100.0000
5889 62466.6667 31116.6667
5890 62483.3333 21266.6667
5891 62483.3333 21350.0000
5892 62483.3333 21483.3333
5893 62483.3333 21733.3333
5894 62483.3333 21933.3333
5895 62483.3333 22016.6667
5896 62483.3333 22050.0000
5897 62483.3333 22300.0000
```

Figure 3.1: Example of Cities Coordinates

3.3 STRUCTURE OF DATA OF TEXT AND SOLUTION APPLICATION

Memory data - objects (point) - is loaded. These are all objects with five parameters:

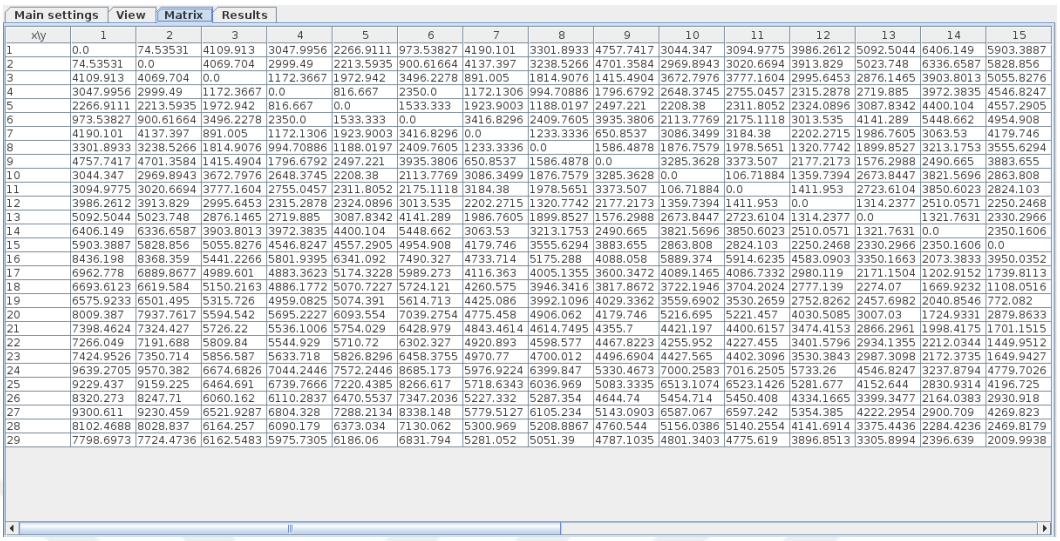
- ID – Point identification.
- X – x-coordinate.
- Y – y-coordinate.
- dX – regulated x-coordinated.
- dY – regulated y-coordinate.

The drawing coordinates have been regulated by each point. The reason is simple: as shown in this input data example, the coordinate range can be quite wide and the drawing can be difficult (Drawing field starts at coordination (0,0), so that all of these coordinates are difficult to trace and uncomfortable with co-ordinates of about 5,000, etc.). That is why regulated design co-ordinates are used, which are adjusted in relation to the diagram area size.

It is not so easy to draw in Java as many people believe, and work with many other elements is just as important. Jpanel was used to draw the Java component. The option to send data with static variable coordinates and end solution has been selected. The drawCircle method draws points with the dX and dY coordinates. DrawLine with the dX and dY co points should draw the final solution. The final sequence shows all the points on the system, so that the tour must be completed by different directions from the last point in the sequence to the first point.

3.4 DESTINATIONS' MATRIX

For computing and working with a destination matrix, the distance from Euclid is used. It means that what units have input data (or map scale) does not matter because only comparisons of paths are made in the same units.



Main settings | View | Matrix | Results

x\y	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0.0	74.53531	4109.913	3047.9956	2266.9111	973.53827	4190.101	3301.8933	4757.7417	3044.347	3094.9775	3986.2612	5092.5044	6406.149	5903.3887
2	74.53531	0.0	4069.704	2999.49	2213.5935	900.61664	4137.397	3238.5266	4701.3584	2969.8943	3020.6694	3913.829	5023.748	6336.6587	5828.856
3	4109.913	4069.704	0.0	1172.3667	1972.942	3496.2278	891.005	1814.9076	1415.4904	3672.7976	3777.1604	2995.6453	2876.1465	3903.8013	5055.8276
4	3047.9956	2999.49	1172.3667	0.0	816.667	2350.0	1172.1306	994.70886	1796.6792	2648.3745	2755.0457	2315.2878	2719.885	3972.3835	4546.8247
5	2266.9111	2213.5935	1972.942	816.667	0.0	1533.333	1923.9003	1188.0197	2497.221	2208.38	2311.8052	2324.0896	3087.8342	4400.104	4557.2905
6	973.53827	900.61664	3496.2278	2350.0	1533.333	0.0	3416.8296	2409.7605	3935.3806	2113.7769	2175.1118	3013.535	4141.289	5448.662	4954.908
7	4190.101	4137.397	891.005	1172.1306	1923.9003	3416.8296	0.0	1233.3336	650.8537	3086.3499	3184.38	2202.2715	1986.7605	3063.53	4179.746
8	3301.8933	3238.5266	1814.9076	994.70886	1188.0197	2409.7605	1233.3336	0.0	1586.4878	1876.7579	1978.5651	1320.7742	1899.8527	3213.1753	3555.6294
9	4757.7417	4701.3584	1415.4904	1796.6792	2497.221	3935.3806	650.8537	1586.4878	0.0	3285.3628	3373.507	2177.2173	1576.2988	2490.665	3883.655
10	3044.347	2969.8943	3672.7976	2648.3745	2208.38	2113.7769	3086.3499	1876.7579	3285.3628	0.0	106.71884	1359.7394	2673.8447	3821.5696	2863.808
11	3094.9775	3020.6694	3777.1604	2755.0457	2311.8052	2175.1118	3184.38	1978.5651	3373.507	106.71884	0.0	1411.953	2723.6104	3850.6023	2824.103
12	3986.2612	3913.829	2995.6453	2315.2878	2324.0896	3013.535	2202.2715	1320.7742	2177.2173	1359.7394	1411.953	0.0	1314.2377	2510.0571	2250.2468
13	5092.5044	5023.748	2876.1465	2719.885	3087.8342	4141.289	1986.7605	1899.8527	1576.2988	2673.8447	2723.6104	1314.2377	0.0	1321.7631	2330.2966
14	6406.149	6336.6587	3903.8013	3972.3835	4400.104	5448.662	3063.53	3213.1753	2490.665	3821.5696	3850.6023	2510.0571	1321.7631	0.0	2350.1606
15	5903.3887	5828.856	5055.8276	4546.8247	4557.2905	4954.908	4179.746	3555.6294	3883.655	2863.808	2824.103	2250.2468	2330.2966	2350.1606	0.0
16	8436.198	8368.359	5441.2266	5801.9395	6341.092	7490.327	4733.714	5175.288	4088.058	5889.374	5914.6235	4583.0903	3350.1663	2073.3833	3950.0352
17	6962.778	6889.8677	4989.601	4883.3623	5174.3228	5989.273	4116.363	4005.1355	3600.3472	4089.1465	4086.7332	2980.119	2171.1504	1202.9152	1739.8113
18	6693.6123	6619.584	5150.2163	4886.1772	5070.7227	5724.121	4260.575	3946.3416	3817.8672	3722.1946	3704.2024	2777.139	2274.07	1669.9232	1108.0516
19	6575.9233	6501.495	5315.726	4959.0825	5074.391	5614.713	4425.086	3992.1096	4029.3362	3559.6902	3530.2659	2752.8262	2457.6982	2040.8546	772.082
20	8009.387	7937.7617	5594.542	5695.2227	6093.554	7039.2754	4775.458	4906.062	4179.746	5216.695	5221.457	4030.5085	3007.03	1724.9331	2879.8633
21	7398.4624	7324.427	5726.22	5536.1006	5754.029	6428.979	4843.4614	4614.7495	4355.7	4421.197	4400.6157	3474.4153	2866.2961	1998.4175	1701.1515
22	7266.049	7191.688	5809.84	5544.929	5710.72	6302.327	4920.893	4598.577	4467.8223	4255.952	4227.455	3401.5796	2934.1355	2212.0344	1449.9512
23	7424.9526	7350.714	5856.587	5633.718	5826.8296	6458.3755	4970.77	4700.012	4496.6904	4427.565	4402.3096	3530.3843	2987.3098	2172.3735	1649.9427
24	9639.2705	9570.382	6674.6826	7044.2446	7572.2446	8685.173	5976.9224	6399.847	5330.4673	7000.2583	7016.2505	5733.26	4546.8247	3237.8794	4779.7026
25	9229.437	9159.225	6464.691	6739.7666	7220.4385	8266.617	5718.6343	6036.969	5083.3335	6513.1074	6523.1426	5281.677	4152.644	2830.9314	4196.725
26	8320.273	8247.71	6060.162	6110.2837	6470.5537	7347.2036	5227.332	5287.354	4644.74	5454.714	5450.408	4334.1665	3399.3477	2164.0383	2930.918
27	9300.611	9230.459	6521.9287	6804.328	7288.2134	8338.148	5779.5127	6105.234	5143.0903	6587.067	6597.242	5354.385	4222.2954	2900.709	4269.823
28	8102.4688	8028.837	6164.257	6090.179	6373.034	7130.062	5300.969	5208.8867	4760.544	5156.0386	5140.2554	4141.6914	3375.4436	2284.4236	2469.8179
29	7798.6973	7724.4736	6162.5483	5975.7305	6186.06	6831.794	5281.052	5051.39	4787.1035	4801.3403	4775.619	3896.8513	3305.8994	2396.639	2009.9938

Figure 3.2: Matrix of destinations

The user selects and calculates the matrix of a file with loaded input data. Computing distances is easy because of the known coordinates. It reminds us of some basic mathematics – the Pythagoras sentence

$$c^2 = a^2 + b^2 \tag{3.1}$$

The distances between two points with co - ordinates are calculated in this case

$$Distance(i,j) = (X_i - X_j)^2 + (Y_i - Y_j)^2 \tag{3.2}$$

Where the first point is (dXi, Dyi) the co-ordain, and the second (dXj, dYj) the co-ordained. The author of the thesis was interested in determining if the matrix of destinations before the import data (including more memory claims) is better or if only if any algorithm is required would be better for each destination. There is insufficient memory, according to the author, for current computers, so no problem will occur before the whole matrix is computed and saved in memory. If a user only wants to solve one problem, options for computer destinations may be more convenient on demand, but it is assumed in this thesis that the input data should be used repeatedly in various settings and that different solutions should be seen and compared.

3.5 GRAY WOLF OPTIMIZATION

3.5.1 Inspiration

Gray wolf (Canis lupus) is part of the family of Canidae. Gray wolves are regarded as apex predators, which means they are at the top of the food chain. Mostly gray wolves prefer to live in a pack. The average group size is 5 - 12. They have a very strict social dominant hierarchy, as shown in Figure 1, of particular interest.

The leaders are called alphas, a male and a female. The alpha is mostly responsible for hunting, sleeping, waking time, and so on. The bundle dictates the alpha decisions. However, there was a sort of democratic behavior in which the other Wolves followed an Alpha. The package recognizes the alpha in meetings by holding its tails. The dominant wolf is also named the Alpha Wolf, as the pack should follow his commands. The pack is for the alpha wolves only. Instead, it is not necessarily the Alpha that is the most important part of the package management. This shows that a package is much more organized and disciplined than its strength.

The second phase of the gray wolves ' hierarchy is beta. Betas are subordinate wolves who support alpha in decisions or other packaging. The beta wolf is male or female. It is probably the most successful candidate for the alpha when a wolf dies or grows old. Both the alpha and other lower levels wolves should be respected by the beta wolf. He is an alpha advisor and discipliner. The Beta reinforces the Alpha commands and gives the whole pack alpha feedback.

The omega rating is the lowest gray wolf. The omega has a role to play. All other dominant wolves have to submit to omega wolves. It's the last wolves that can eat. The omega does not appear to be an important person in the pack, but the whole package is confronted with internal problems and struggles if the omega is lost. This is because the omega(s) of all wolves are violent and frustrative. This helps the whole pack to satisfy and maintain the dominant structure. In some cases, Omega is also the childcare center in the pack.

If an alpha wolf, beta or omega isn't, it's a subordinate (or delta) in some references. Alphas and betas must be submitted to delta wolves, but omega is dominant. Scouts, sentinels, elders, hunters and carers are included in this category. Scouts monitor the borders of the area and warn the pack if there is a danger. Sentinels protect and guarantee the safety of the pack. Elders are

Alpha or Beta wolves experienced. Juniors help hunt prey alphas and betas and provide food for the pack. At last, carers take care of the wolves who are weak, sick, and injured.

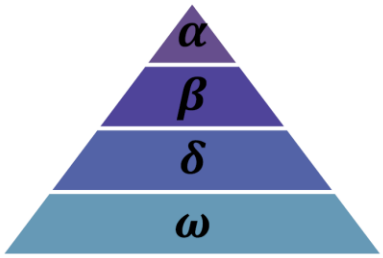


Figure 3.3: Grey wolfs hierarchy

3.5.2 Mathematical model and algorithm

A) Social hierarchy:

In order to mathematically model the social hierarchy of wolves in the design of GWO, we consider alpha (a) to be the most appropriate solution. The second and third best solutions are therefore called beta (b) and delta (d). The remaining solutions for the candidate are considered to be omega (x). Hunting (optimization) is guided by a, b and d in the GWO algorithm. These three wolves follow the x wolves.

B) Encircling prey

As mentioned above, during the hunt, gray wolves surround the prey. The following equations are proposed to mathematically model encircling behavior:

$$D = \left|A.X_{(l)} - C.Xp_{(l)}\right| \tag{3.3}$$

$$X_{(l+1)} = Xp_{(l)} - A.D \tag{3.4}$$

In l as the most recent iteration, the coefficient vectors are C and A ; in Xp and X, the prey and the gray wolves position vectors. In (3) and (4), the C and A vectors are calculated.

$$A = 2c.r_1 - c \tag{3.5}$$

$$C = 2.r_2 \tag{3.6}$$

If c decreases linearly during the iteration process from 2 to 0 (in both phases of exploitation and exploration) and r1, r2 are random vectors in [0, 1].

C) Hunting

Grey wolves can recognize and encircle the location of the prey. Usually the alpha leads the hunt. Beta and delta can also be involved in hunting occasionally. However, in an abstract search space we have no idea how to position the optimum (prey) location. We presume that beta and delta alpha (the best candidate solution) better know the potential location of the prey to mathematically simulate the hunting behavior of gray wolves. Therefore we have saved the first three optimal solutions so far and require the other search agents (including omegas) to update the search agents ' positions accordingly. The following formulas are proposed in this respect, pursuant to (5), (6) and (7).

$$\begin{aligned} D_\alpha &= |C_1.X_\alpha - X| \\ D_\beta &= |C_2.X_\beta - X| \\ D_\delta &= |C_\delta.X_\delta - X| \end{aligned} \tag{3.7}$$

$$\begin{aligned} X_1 &= |X_\alpha - A_1(D_\alpha)| \\ X_2 &= \left|X_\beta - A_2\left(D_\beta\right)\right| \\ X_3 &= |X_\delta - A_3(D_\delta)| \end{aligned} \tag{3.8}$$

$$X_{(l+1)} = (X_1 + X_3 + X_3)/3 \tag{3.9}$$

As soon as the prey stops change its position, the hunting process ends. The interval of A [-2c, 2c] and c from 2 to 0 must be a random number.

What is in 3.4 displays how a 2D search agent updates the alpha, beta and delta location of the search agent. The end position can be seen inside an alpha, beta or delta position in a random

location in the search area. Which means, they estimate the position of a beast by alpha, beta and delta and other wolves randomly update its position around the beast.

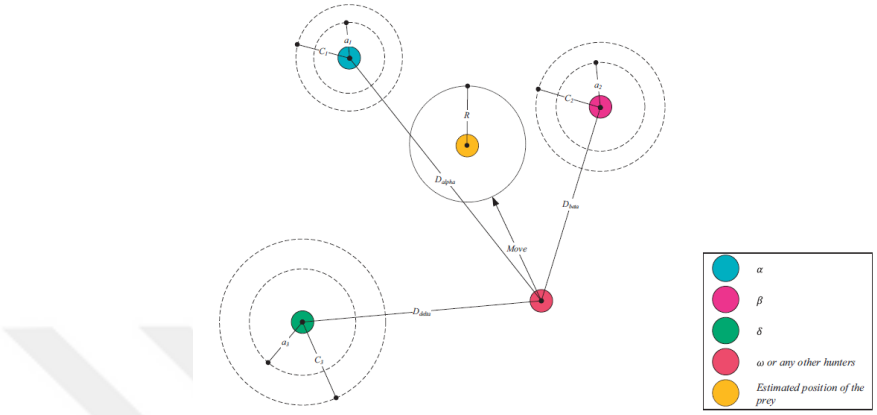


Figure 3.4: Position updating in GWO

D) Attacking prey

The gray wolves end hunting when they stop moving. As mentioned earlier. We lower the value of a in order to model the prey mathematically. Note that the range of fluctuation of A is also reduced by one. ~A is a random value in the [-2a, 2a] interval where a value is lowered from 2 to 0 during iterations. The next search agent position between its current position and the position of the prey can be anywhere when a random A values lie within [-1, 1]. 3(a) shows that wolves must attack the prey.

The operators proposed so far enable the GWO algorithm to update its position on the basis of alpha, beta and delta locations for their search agents and attack against the projected individual. The GWO algorithm with these operators, however, is likely to stagnate in local solutions. It is true that some scanning is demonstrated by the encircling mechanism, but GWO needs more operating agencies to underline scanning.

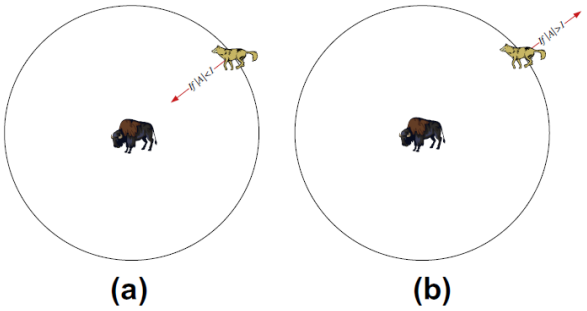


Figure 3.5: Attacking prey versus searching for prey

E) Search for prey

Grey wolves mainly look for positions of alpha, beta and delta. They vary from searching for beasts to attacking animals. In order to mathematically model the divergence, we use ~A with random values of 1 or less than 1 forced the search agent to diverge the prey. This emphasizes research and permits a global GWO algorithm. Figure 3(b) shows also that forces gray wolves to differentiate from the animal and, hopefully, find a more adequate beast.

The GWO positions drive the Wolves to search for the animals and ensure that the animals are best attacked. Mathematically, the scanning process is modeled on A with random numbers when the scanner's members are forced by A < -1 or A>1 to differ.

The following is the GWO described.

Step 1: Start the population of the wolf.

Step 2: Make A, C and a

Step 3: Compute the fitness of the agent and determine the best three first agents X_α, X_β and X_δ.

Step 4: The current agent position is updated using Eq (7).

Step 5: Update A, C and a

Step 6: Fitness calculation and update of all agents X_α, X_β and X_δ

Step 7: Use mutation in pairs of swap

Step 8: Population optimization.

Step 9: Go to step 3 until the end criteria are fulfilled.

Step 10: Take the best solution X_α as a result.

Figure 3.6: GWO's pseudo code

3.6 SWAP MUTATION

In its global and local models PSO always solves continuous optimizing problems for optimal solutions, which does not make the algorithm ideal for the resolution of discreet problems such as TSP. Recently, however, researchers have been able to use PSO to solve TSP through swap operators. The resolution of the TSP by GWO is therefore done with the addition of a pair - side swap mutation (PSM).

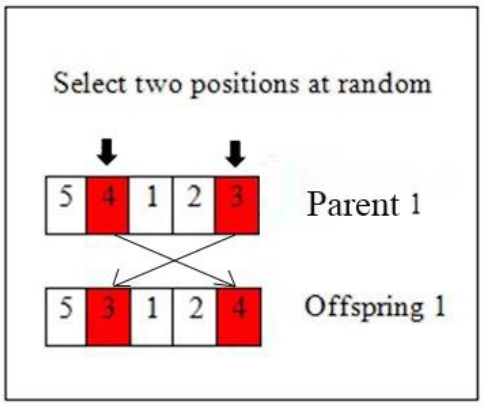


Figure 3.7: Mutation of pair-way swap

4. IMPLEMENTATION AND RESULTS

4.1 APPLYING GWO FOR TRAVEL SALESMAN PROBLEM

TSP implements all three algorithms. Set to 200 population, set to 2000 iterations maximum number. In this paper, the TSPLIB [32] uses 6 benchmark problems with a range of 30 to 100 cities. Table 1 summarizes the results of the experiments, averaging 30 runs of each data set of all models. The first column of the table shows the name of the instance of the right solution length. The second column of the "Known Optimal Solution" shows the best solution length. The third column of the GA is the result of the genetic algorithm capability. The' Modified GWO ' column is the outcome of the Grey Wolf Optimizer that can solve TSP after being modified.

The algorithm has been applied to the GWO population parameter TSP by 500 search agents. Three benchmark issues have been identified in this document. The references TSPLIB to cities 51, 42, and 14 are dantzig 42 and burma 14. Table 1 summarizes the GA performance comparison in these cities as well as the GWO performance in line with the GA performance and changed GWO performance. The GA and GWO iterations for burma14 were 100, for dantzig 42 were 1000 and for eli51 were 1000.

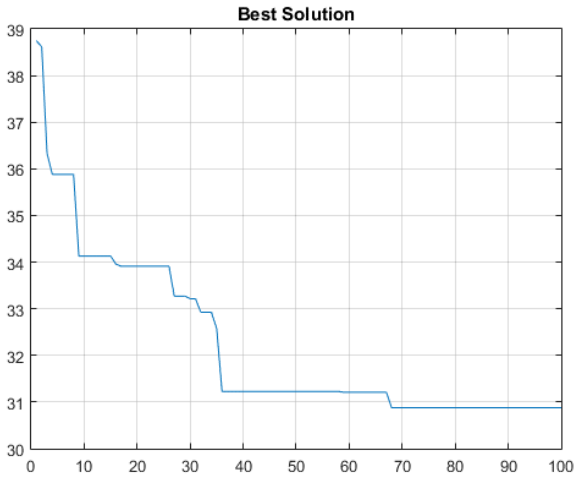


Figure 4.1: GWO's error chart for the burma14

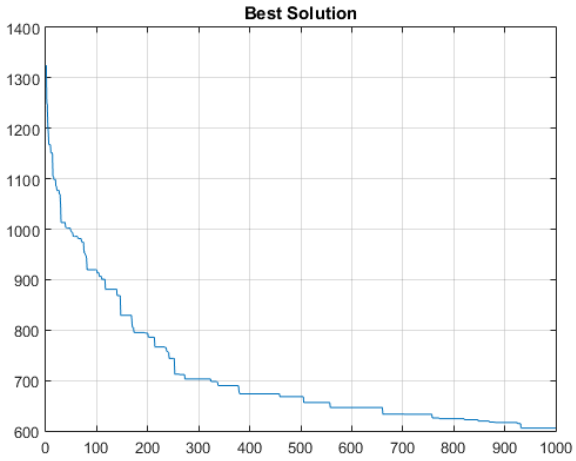


Figure 4.2: GWO's error chart for the dantzig42

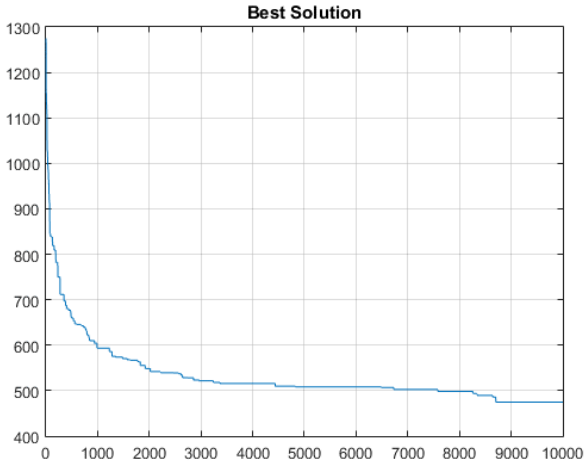


Figure 4.3: GWO's error chart for the eil51

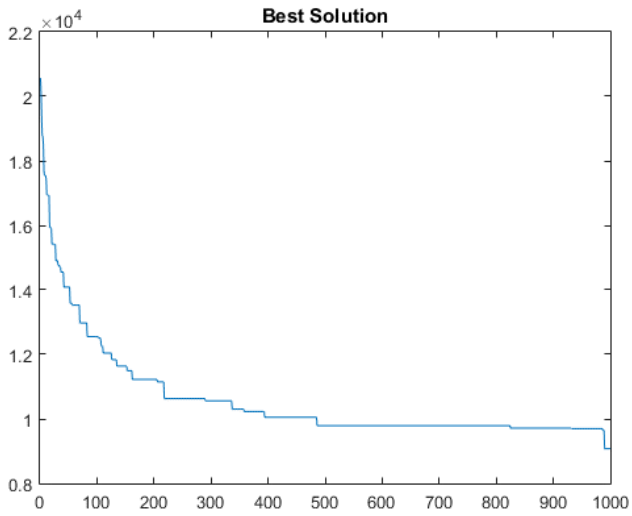


Figure 4.4: GWO's error chart for the bay29

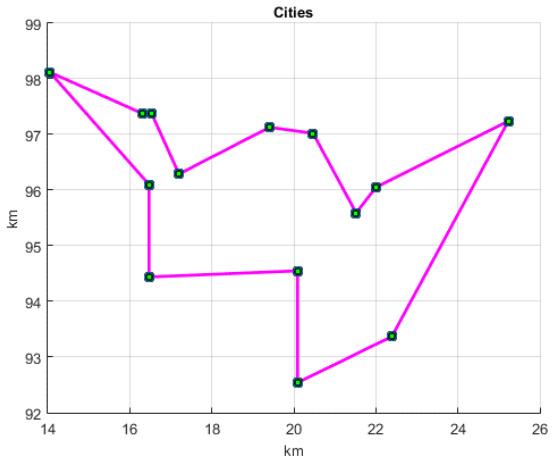


Figure 4.5: The ideal solution for burma14

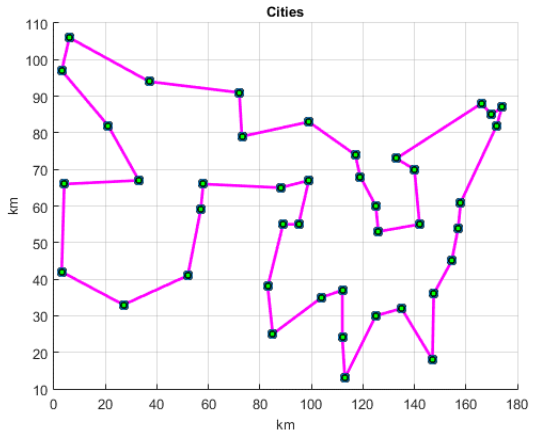


Figure 4.6: The ideal solution for dantzig42

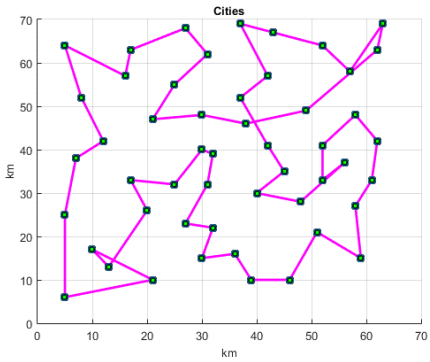


Figure 4.7: The ideal solution for eil51

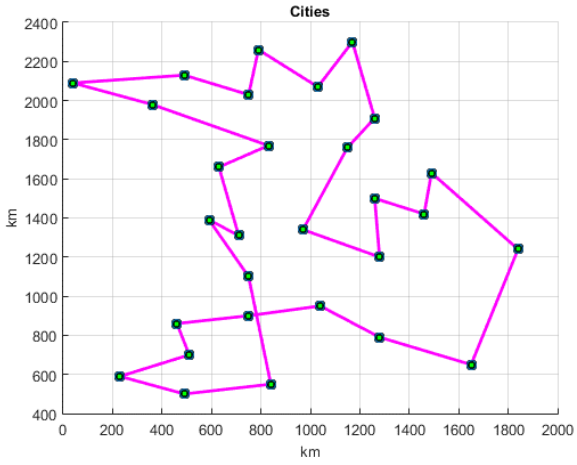


Figure 4.8: The ideal solution for bay29

4.2 RESULTS

The algorithm has been applied to the GWO population parameter TSP by 500 search agents. Three benchmark issues have been identified in this document. The references TSPLIB to cities 51, 42, and 14 are dantzig 42 and burma 14. Table 1 summarizes the GA performance comparison in these cities as well as the GWO performance in line with the GA performance and changed GWO performance. The GA and GWO iterations for burma14 were 100, for dantzig 42 were 1000 and for eli51 were 1000.

Table 4.1: The GA Compared with GWO

Dataset	Burma 14	Dantzig 42	Eil 51	Bay 29
The Required Value	30.8785	679	425	9074
GA	30.8785	679	430	9074
GWO	30.8785	679	429	9074

5. CONCLUSION

Summary of Results, saving time and costs are one of the requirements for finding the optimum route. This document was perfectly used to solve TSP problems with a GWO modified wolf optimization system for TSP that has been shown to be good in comparison with genetic algorithms. Comparisons of the performance of GA and modified GWW based on performance of GA and GWO based on four key datasets (burma14, dantzig42, eil51, and bay29). In comparison to the GA results, GWO showed slightly better results.

Future Work, future work would largely focus on adding advanced data filtering techniques to the software package. One of the issues that could be addressed immediately is the detection of markers which have high correlation to multiple chromosome groups. This kind of detection is clearly impossible to do before the chromosome groups are formed, and since the procedure is so labor-intensive in GWO it is time-prohibitive.

5.1 SUGGESTIONS

In this research work, we perform the GWO using modified PSO's swap mutation using the tsp research datasets. For future suggestions we want to perform below points:

- The current evaluation is based on optimization for solving TSP, however under the real time scenario this will not be the case always. So we suggest evaluating the performance of proposed model using bigger datasets.

Second point is, the consideration of more real time will be the interesting future direction for this research work.

REFERENCES

[1] A E.H.L Aarts and J.K. Lenstra, editors, *Local Search in Combinatorial Optimization*, Wiley, Chichester, UK, 1997.

[2] N. Agatz, A.M. Campbell, M. Fleischmann, and M. Savelsbergh. *Challenges and opportunities in attended home delivery*. In B. Golden, S. Raghavan, and E. Wasil, editors, *The Vehicle Routing Problem: Latest Advances and New Challenges*. 2007.

[3] D. Applegate, R. Bixby, V. Chvatal, and W. Cook, *private communication* (1994).

[4] L.D. Aroson, Algorithms for vehicle routing – A survey. 1995

[5] R. Baldacci, M. Battarra and D. Vigo. Routing a Heterogeneous Fleet of Vehicles. Technical report. 2007.

[6] T. Bektas. *The multiple traveling salesman problem: an overview of formulations and solution procedures*. Omega, 34:209–219, 2006.

[7] J. L. Bentley, *Experiments on traveling salesman heuristics*, in Proc. 1st Ann. ACMSIAM Symposium. On Discrete Algorithms, SIAM, Philadelphia, PA, 1990, 91-99.

[8] J. L. Bentley, *Fast algorithms for geometric traveling salesman problems*, ORSA J. Comput.4 (1992), 387-411.

[9] L. Bertazzi, M.G. Speranza, M.W.P. Savelsbergh, (2008), *Inventory Routing, in The Vehicle routing Problem: Latest Advances and New Chalenges*, B. Golden, R. Raghavan, E. Wasil, (eds.), Operations Research/Computer Science Interfaces Series, 2008.

[10] R. G. Bland and D. F. Shallcross, *Large traveling salesman problems arising from experiments in X-ray crystallography: A preliminary report on computation*, Operations Research. Lett. 8 (1989), 125-128.

[11] L.D. Bodin, B.L. Golden, A.A. Assad, and M. Ball. Routing and scheduling of vehicles and crews, the state of the art. *Computers and Operations Research*, 10(2):63-212, 1983.

[12] J. A. Bondy and U. S. R. Murty, *Graph Theory with Applications*. Elsevier North-Holland, 1976.

[13] A. R. Butz, Convergence with Hilbert's space filling curve, J. Comput. Sys. Sci., vol. 3, May 1969, pp 128-146.

[14] E.P.F. Chan, and H. Lim, Optimization and Evaluation of Shortest Path Queries,VLDB Journal (16:3) July 2007, pp.343-369.

[15] E.P.F. Chan, and J. Zhang, , A Fast Unified Optimal Route Query Evaluation Algorithm, Proceedings of ACM 16th Conference on Information and Knowledge Management (CIKM 07), pp. 371-380, Lisboa, Portugal, Nov. 2007.

[16] N. Christofides, *Worst-case analysis of a new heuristic for the traveling salesman problem,* Report No. 388, GSIA, Carnegie-Mellon University, Pittsburgh, PA, 1976.

[17] N. Christofides. Vehicle routing. In E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys, editors, *The Traveling Salesman Problem*, Wiley, Chichester, UK, 1985, pp. 431-448.

[18] N. Christofides, A. Mingozzi, and P. Toth. The vehicle routing problem. In N. Christofides, A. Mingozzi, P. Toth and C. Sandi, editors, *Combinatorial Optimization*, Wiley, Chichester, UK, 1979, pp. 315-338

[19] G. Clarke and J.V. Wright. *Scheduling of vehicle from a central depot to a number of delivery points*. Operations Research, 12:568-581, 1964.

[20] J-F Cordeau, M Gendreau, G Laporte, J-Y Potvin and F Semet. *A guide to vehicle routing heuristics*. Journal of Operational Research Society. (2002) 53, 512-522.

[21] T.H. Cormen, C.E. Leiserson, R.L.Rivest, and C. Stein. *Introduction to Algorithms*. McGraw-Hill Book Company, 2001.

[22] G. A. Croes, *A method for solving traveling salesman problems*, Operations Research 6 (1958), 791-812.

[23] H. Crowder and M. Padberg, *Solving large-scale symmetric travelling salesman problems to optimality*, Mgmt. Sci. 26 (1980), 495-509.

[24] B. Chandra, H. Karloff, and C. Tovey, *New results on the old k-opt algorithm for the TSP*, in „„„Proceedings 5th ACM-SIAM Symposium on Discrete Algorithms,““ Society for Industrial and Applied Mathematics, Philadelphia, 1994, 150-159.

[25] G.B. Dantizg and J.H. Ramser. *The truck dispatching problem*. Management Science, 6:80, 1959.

[26] M. Desrochers, J.K. Lenstra, and M.W.P. Savelsbergh. A classification scheme for vehicle routing and scheduling problems. Journal of Operational Research Society, 46:322-332, 1990.

[27] M.L. Fisher. Vehicle routing. In M.O. Ball, T.L. Magnanti, C.L. Monma, and G.L. Nemhauser, editors, *Network Routing, Handbooks in Operations Research and Management Science 8*, North-Holland, Amsterdam, 1995, pp. 1-33.

[28] A. M. Frieze, *Worst-case analysis of algorithms for traveling salesman problems*, Methods of Operations Research 32 (1979), 97-112.

[29] H. N. Gabow and R. E. Tarjan, *Faster scaling algorithms for general graph-matching problems*, J. Assoc. Comput. Mach. 38 (1991), 815-853.

[30] M. Gendreau, A. Hertz, and G. Laporte, *New insertion and post-optimization procedures for the traveling salesman problem*, Operations Research 40 (1992), 1086-1094.

[31] M. Gendreau, G. Laporte, and J.-Y. Potvin. *Vehicle routing: Modern heuristics.* In P. Toth and D. Vigo, editors, *The Vehicle Routing Problem*. Siam, Bologna, Italy, 1997, pp. 129-154.

[32] B.L. Golden, E.A. Wasil, J.P. Kelly, and I.M. Chao. Metaheuristics for the capacitated VRP. In T.G Crainic and G. Laporte, editors, *Fleet Management and Logistics*, Kluwer, Boston, MA, 1998, pp. 33-56.

[33] M. Held and R. M. Karp, *The traveling-salesman problem and minimum spanning trees*, Operations Research 18 (1970), 1138-1162.

[34] M. Held and R. M. Karp, *The traveling-salesman problem and minimum spanning trees: Part II*, Math. Programming 1 (1971), 6-25.

[35] D. Hilbert, "Ueber die stetige Abbildung einer Line auf ein Flächenstück", *Mathematische Annalen* **38**: (1891) 459–460

[36] D. S. Johnson. *A Theoretician's Guide to the Experimental Analysis of Algorithms* To appear in *Proceedings of the 5th and 6th DIMACS Implementation Challenges*, M. Goldwasser, D. S. Johnson, and C. C. McGeoch, Editors, American Mathematical Society, Providence, 2002.

[37] D. S. Johnson, G. Gutin, L. A. McGeoch, A. Yeo, W. Zhang, and A. Zverovich, *Experimental Analysis of Heuristics for the ATSP* in *The Traveling Salesman Problem and its Variations*, G. Gutin and A. Punnen, Editors, Kluwer Academic Publishers, 2002, Boston, 445-487

[38] D.S. Johnson and L.A. McGeoch. *The traveling salesman problem: A case study*. In E.H.L Aarts and J.K. Lenstra, editors, *Local Search in Combinatorial Optimization*, Wiley, Chichester, UK, 1997, pp. 215-310.

[39] D. S. Johnson and L. A. McGeoch, *Experimental Analysis of Heuristics for the STSP* in *The Traveling Salesman Problem and its Variations*, G. Gutin and A. Punnen, Editors, Kluwer Academic Publishers, 2002, Boston, 369-443

[40] M. Junger, G. Reinelt, and G. Rinaldi, *The Traveling Salesman Problem*, Report No. 92.113, Angewandte Mathematik und Informatik, Universit.a. t zu K.o. ln, Cologne, Germany,1994.

[41] B. Korte, *Applications of combinatorial optimization*, talk at the 13th International Mathematical Programming Symposium, Tokyo, 1988.

[42] G. Laporte: *The Traveling Salesman Problem: An overview of exact and approximate algorithm*. European Journal of Operations Research 59 (1992), pp. 231-247.

[43] G. Laporte and Y. Nobert. *Exact algorithms for the vehicle routing problem*. Annals of Discrete Mathematics, 31:147-184, 1987.

[44] S. Lin, *Computer solutions of the traveling salesman problem,* Bell Syst. Tech. J. 44 (1965), 2245-2269.

[45] S. Lin and B. W. Kernighan, *An Effective Heuristic Algorithm for the Traveling Salesman Problem*, Operations Research 21 (1973), 498-516.

[46] B.G. Madsen, A. Larsen, M. M. Solomon, *Dynamic Vehicle Routing Systems – Survey and Classification,* Proceeding of the Tristan IV Conference (2007)

[47] H. L. Ong and J. B. Moore, *Worst-case analysis of two traveling salesman heuristics*, Operations Research Letter 2 (1984), 273-277.

[48] I. OR, *Traveling Salesman-Type Combinatorial Problems and their Relation to the Logistics of Regional Blood Banking*, Ph.D. Thesis, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL, 1976.

[49] M. Padberg and G. Rinaldi, *Optimization of a 532-city symmetric traveling salesman problem by branch and cut,* Operations Research Letter 6 (1987), 1-7.

[50] C. H. Papadiamitriou and K. Steiglitz, *On the complexity of local search for the traveling salesman problem,* SIAM J. Comput. 6 (1977), 76-83.

[51] C. H. Papadiamitriou and K. Steiglitz, *Some examples of difficult traveling salesman problems,* Operations Research 26 (1978), 434-443.

[52] C. Potts and S. Van de Velde, *Dynasearch—Iterative local improvement by dynamic programming: Part I, The traveling salesman problem,* manuscript (1995).

[53] G. Reinelt, *The Traveling Salesman Problem: Computational Solutions for TSP Applications*, Lecture Notes in Computer Science 840, Springer-Verlag, Berlin, 1994.

[54] H. G. Rinnooy Kan E. L. Lawler, J. K. Lenstra and D. B. Shmoys, editors. *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*. John Wiley & Sons, 1985.

[55] D. J. Rosenkrantz, R. E. Stearns, and P. M. Lewis, II, *An analysis of several heuristics for the traveling salesman problem,* SIAM J. Comput. 6 (1977), 563-581.

[56] S. Sahni and T. Gonzalez, *P-complete approximation problems*, J. Assoc. Comput. Mach. 23 (1976), 555-565.

[57] Schrijver. *On the history of combinatorial optimization (till 1960).* In K. Aardal, G.L. Nemhauser, and R. Weismantel, editors, Discrete Optimization, volume 12 of Handbooks in Operations Research and Management Science, chapter 1. Elsevier, 2005.

[58] D. B. Shmoys and D. P. Williamson, *Analyzing the Held-Karp TSP bound: A monotonicity property with applications,* Inform. Process. Lett. 35 (1990), 281-285.

[59] E.D. Tailard, L.M. Gambardella, M. Gendreau, and J.-Y. Potvin. *Adaptive memory programming: A unified view of metaheuristics.* Research Report IDSIA/19-98, IDSIA, Lugano, Switazerland, 1998.

[60] P. Toth and D. Vigo, editors, *The Vehicle Routing Problem*. Siam, Bologna, Italy.

[61] P.Toth and D.Vigo. Exact algorithms for vehicle routing. In T.G Crainic and G. Laporte, editors, *Fleet Management and Logistics*, Kluwer, Boston, MA, 1998, pp. 1-31.

[62] P.Toth and D.Vigo. Models, relaxations and exact approaches for the capacitated vehicle routing problem. *Discrete Applied Mathematics*.

[63] L. Wolsey, *Heuristic analysis, linear programming, and branch and bound*, Math. Prog. Stud. 13 (1980), 121-134.

[64] http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/

[65] http://www.tsp.gatech.edu/sweden/index.html

[66] http://www.tsp.gatech.edu/world/index.html

[67] http://www.research.att.com/~dsj/chtsp/