A SOLUTION FOR STORAGE SPACE ALLOCATION PROBLEM IN CONTAINER TERMINALS

Kathaluwa Liyana Kankanamge Suranga Deshapriya

168216L

Dissertation submitted in partial fulfillment of the requirements for the Degree of Master of Science in Computer Science

Department of Computer Science and Engineering

University of Moratuwa

Sri Lanka

March 2020

DECLARATION

I declare that this is my own work and this dissertation does not incorporate without acknowledgement any material previously submitted for degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, I hereby grant to University of Moratuwa the non-exclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Signature: ……..……. Date:.....

Name: K.L.K. Suranga Deshapriya

I certify that the declaration above by the candidate is true to the best of my knowledge and that this report is acceptable for evaluation of the Masters' thesis.

Signature of the supervisor: ……. Date: ……..

Name: Dr. Indika Perera

ABSTRACT

Container terminal operations at a container port form an important part of worldwide goods trade. These facilities/operations are usually involved with expensive and limited resources and must, therefore, be planned carefully to ensure effective usage of the limited resources. The main roles of a container terminal are the transfer of inbound and outbound containers as well as their storage within the container yard of the container terminal.

Focusing more on inbound containers, we study the container storage space allocation problem, being one of the major problems in container terminals, and presents a solution with an implementation of a Genetic algorithm. In our solution, we aim to minimize the number of containers that have to be re-handled both when a container is fetched from vessel in order to store in the terminal and when a container is to be dispatched to a customer from the container terminal. In addition, the total Yard crane movements across the bays are aimed to be minimized. We take into account also the different container types such as Regular, Open Top and Reefer containers, which require special storage space allocations. Furthermore, we adapt our solution such that it can provision for the changes in the environment and configurations, etc. with minimal code changes.

For the evaluation of our work, it was compared against both the standard LIFO approach as well as the Optimized LIFO approach, which is an optimization of the manual process used. For this, results of 50 sample shipments were evaluated against these two approaches and the results indicated that Optimized LIFO produced better results than the LIFO approach and that both the LIFO and the Optimized LIFO produced better results than the results of GA's initial generations. But with the generations to pass by, GA results got improved and went past the fitness of LIFO and Optimized LIFO results, even though the rate of improvements declined. Thus, for all 50 samples, the results of our solution could go past the results of LIFO and Optimized LIFO approach, within an average of 21.32 generations.

ACKNOWLEDGEMENTS

I would like to express profound gratitude to my advisor, Dr. Indika Perera, for his invaluable support by providing relevant knowledge, materials, advice, supervision and useful suggestions throughout this research work. His expertise and continuous guidance enabled me to complete my work successfully.

I am grateful for the support and advice given by Dr. Shehan Perera and Dr. Malaka Walpola, by encouraging continuing this research till the end. Further I would like to thank all my colleagues for their help on finding relevant research material, sharing knowledge and experience and for their encouragement.

I am as ever, especially indebted to my parents for their love and support throughout my life. I also wish to thank my loving wife, who supported me throughout my work.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Description
CT	Container Terminal
CY	Container Yard
YC	Yard Crane
QC	Quay Crane
GA	Genetic algorithm
SSAP	Storage space allocation problem
HS	Harmony Search
B&B	Branch-and-Bound
TEU	Twenty-Foot Equivalent Unit

CHAPTER 1
INTRODUCTION

Worldwide goods trade contributes a significant portion to the world economy and most of the worldwide goods trade occurs in the form of sea freight trade. The worldwide sea freight trade, in turn, occurs in the form of container shipping and container ports, thus, plays an important role in the worldwide goods trade. Containerized shipping, on the other hand, has been growing rapidly and has thus imposed increasing demand in container port facilities. The part of the container port which facilitates the container handling is the Container Terminal (CT) and is a very important part of the container port. There has been increasing competition between CTs, especially among the ones within close geographic proximity. Due to this competition, the increased workload and the limited resources in the container terminals, it is essential for the CTs to optimize their operations and schedules in order to remain competitive. The container Storage Space Allocation Problem (SSAP), which leads to a significant container re-handling and unproductive crane movements, is a major problem the CTs face and it is mostly due to the storage space limitations in the CTs. In addition, requirements such as Container Yard (CY) configurations and layouts tend to change frequently in CTs and having to respond to them promptly by their systems is yet another problem that the CTs face. In this research, we study the SSAP, explore existing literature and adopt a solution including a suitable software architecture and presents our solution such that it can easily be accommodated to specific configurations and needs. We evaluate the performance of our solution in terms of the number of container re-handlings both at vessel and at yard as well as the total Yard Crane (YC) movements against the LIFO and the Optimized LIFO approaches.

1.1 Container Yards and Yard operations

Most of the worldwide goods trade today occurs in the form of shipping containers using container vessels. Thus container ports become a central point of importance in the container transportation. When a container vessel arrives at a port the inbound containers have to be unloaded from the vessel, temporarily storing them in the storage area whilst the outbound containers have to be loaded in the vessel from where they are stored. Figure 1.1 depicts the loading and unloading process of containers. These

operations are handled by the CT and the containers are stored within the terminal storage areas. CTs provide facilities to transship cargo containers between different transport vehicles, such as container vessels and land vehicles. When container vessels are involved, it is called a maritime CT. The transshipment can also be between land vehicles such as trains and trucks, in which case it is called an inland CT. In this study, we focus our study only on the container storage at maritime CTs but the same solution we propose in this work is also applicable for the inland CTs.

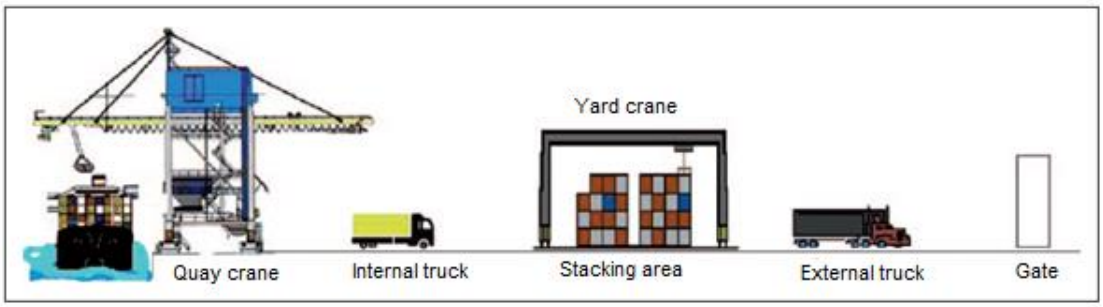


Figure 1.1: Container loading and unloading process [12]

There are primarily three types of instruments used in a CT. Quay Cranes (QC), Yard Cranes (YC) and Trucks/Tractors. QCs are used to load/unload containers on/off the vessels. They are placed close to the vessel and are used to unload inbound containers off the ship and load them on the trucks. Outbound containers, on the other hand, are unloaded from the trucks and loaded on the vessel using these QCs. Most of these QCs have the ability to rotate 360 degrees and move in any direction and at any angle. The operations of QCs are relatively quicker and cheaper to the YCs

YCs are used at the CY for placing and taking the containers on/off the container stacks. They are usually placed above the container blocks of stacks. These YCs have the movements in three directions. Up and down along the stack, Left and right along the bay and back and forth across the bays. When the movements are across the bays, the entire structure has to be moved and hence takes more time and energy than the movements in other directions.

The trucks are used to transport containers between QCs and YCs. Railway tracks are also used by some terminals for the transportation. Figure 1.2 depicts the basic layout

of the CT with the QCs in Red, YCs in Blue and Trucks/Tractors in Green. In addition to the QCs and YCs, some terminals use fork lifts and straddle carriers, which are capable of both transporting containers and storing them on the stack. Manned equipment are used in most of the terminals. There are, however, some semi-automated terminals such as in Port of Rotterdam, in Netherlands. At such terminals, automated guided vehicles are used for the container transportation and the automated stacking cranes are used for the stacking process [8]. While these semi automation reduces operational costs, the initial costs for setting up these semi-automated systems are very high and this is one of the reasons not most of the CTs are semi-automated yet.

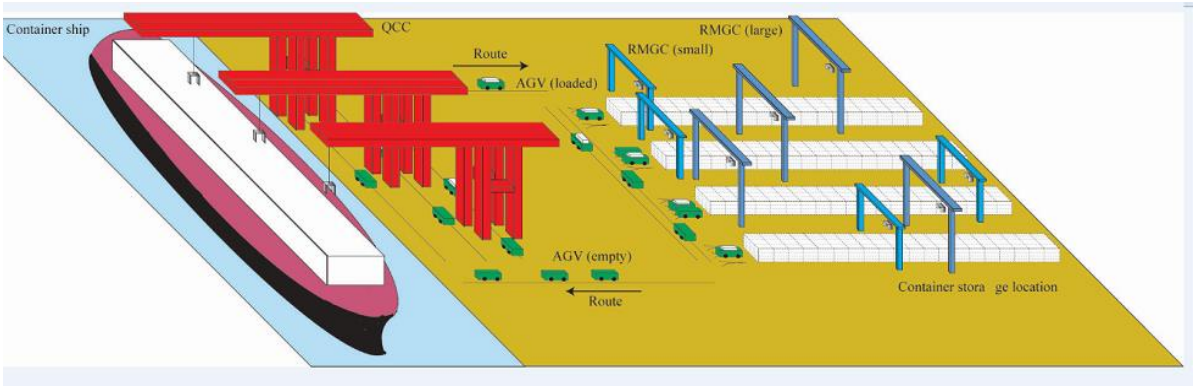


Figure 1.2: Basic layout of a Container terminal with QCs (red), YCs (blue) and Trucks/Tractors (green)

1.2 Container Storage at Container Yard

Inbound containers are stored in the CY within the terminal. This storage area is divided into multiple blocks and each block, in turn, is organized as bays and lanes/rows. On these bays and rows, containers are stacked usually, but not necessarily, around up to 4-5 tiers. The height of the stack (no of tiers) is usually determined by height at which the YCs are installed above the container blocks. With this kind of organization, the containers are stored in multiple three dimensional blocks. The following figure 1.3 depicts the organization of containers within a block.

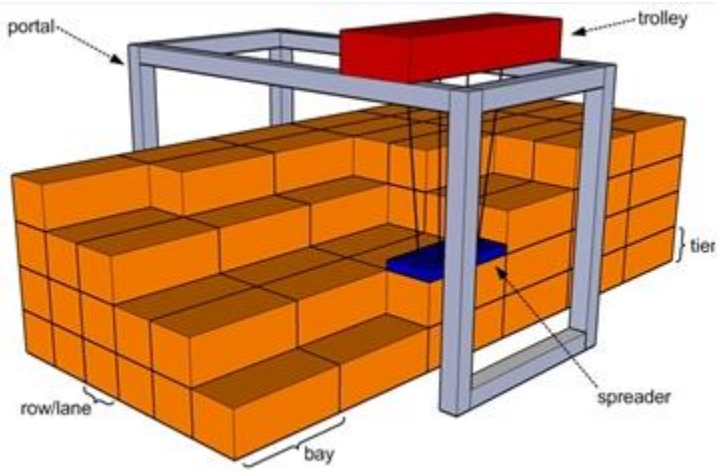


Figure 1.3: Organization of containers within a block

Thus, to locate a particular container, four parameters need to be known. The block number, bay number, row number, and the tier define the exact location of a particular container. Following figure 1.4 depicts the location of a particular container in the CY

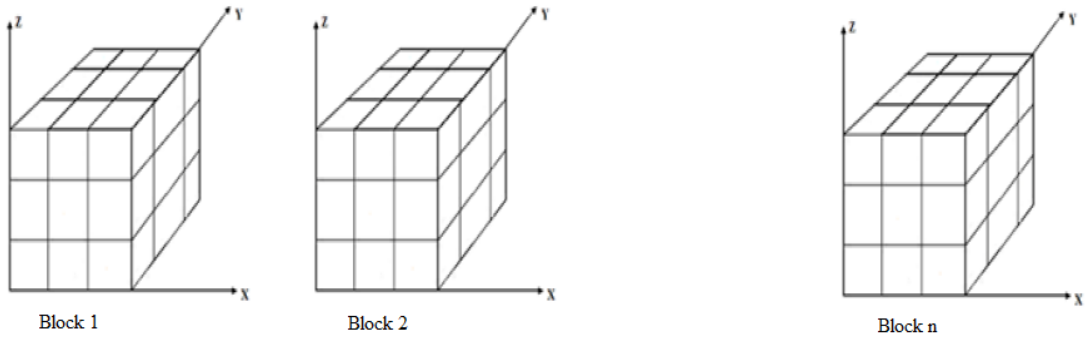


Figure 1.4: Location of a particular container with the parameters: block no x, y and z

There are no standard figures as to the number of containers along each dimension (axis) of the block and different CTs may have different figures, usually based on the height of the overhead YCs, the shape and the layout of the storage area. But different blocks within the same CT usually have the same x, y, and z maximum values. In this figure, x axis represent the row/lane number, while y and z represent the bay number and tier number respectively.

1.3 Different types of containers

Today, different kinds of goods are transported in the shipping containers and certain goods may need special type of containers to avoid damages to the goods inside, as goods/items may be remained within the container for several weeks to couple of months before the container is finally released and the goods are unloaded from the container. Thus, in addition to the regular containers, to prevent damages to the goods inside, based on the nature of the goods, special container types such as Refrigerated/Heated, Ventilated (open top), etc are used. To standardize the different types and properties of the containers, ISO 6346 standard has emerged [16], [17] and unlike in the early stages of container manufacture, now the containers are manufactured in compliance with these ISO standards.

Amongst these different types of containers, there may be certain types that requires special storage allocations. Reefer containers requires electricity to maintain the required temperature inside. But the power supply sockets which provide access to electricity are not available near every storage locations. Thus, it is important to make sure that reefer containers are allocated the storage locations within a close proximity to power supplies. Another container type that requires special storage allocation is Open Top containers. In this container type, the top of the containers are kept open. Therefore, when stacking containers, containers cannot be placed on top of an Open Top container. Thus, Open top containers must always be at the top of the stack. Figure 1.5 depicts an Open Top container.

There can also be empty containers in the inbound containers. Container carriers/Liners import and export empty containers too in order to balance the containers in their yards located in different countries. These empty containers are usually stored separately from other containers in the CT storage area. A Reefer container can be either a full container or an empty container. The constraints we discussed earlier are only applicable when the Reefer container is a full container. When it is an empty container, it does not have goods inside, hence it is not necessary to maintain a particular temperature inside. An Open Top container can also be a full or an empty container. The constraints for the Open Top containers too are only

applicable when the container is a full container. When an Open Top container is an empty container, there are no goods inside and it is thus not necessary to allow ventilation and to keep it open/unblocked. Thus, when a container is an empty container, irrespective of its type (whether it is a reefer container, an Open Top container, etc.), the container is stored separately in the Empty container storage area, without requiring special storage allocations that would otherwise require (if it were a Reefer container etc..)

Figure 1.5: An Open Top container

1.4 The Problem/Opportunity

While organizing containers in stacks helps reduce the space requirements, it yields a new problem when retrieving containers from the stack. The containers in the stacks placed in the CY are on a temporary basis and each container is associated with a delivery/departure deadline. Problem arises when the container to be retrieved is not at the top of the container stack, in which case all the containers above have to be re-

handled to retrieve the required container. This can lead to a significant financial loss as it is possible that this can occur at a considerable rate, particularly when the height of the stack gets higher. These re-handling actually are unnecessary (non-productive) operations and waste valuable time, energy, resources, etc. and ultimately will lead to adding up unnecessary costs.

Furthermore, based on the storage location allocations for the containers, the YCs might have to undergo unproductive movements when retrieving containers in the order of their deadlines. This happens when the containers are scattered, in terms of their deadlines, across the storage block.

This problem is known as container Storage Space Allocation Problem (SSAP) and can get more complicated at the availability of different types of containers, as some container types are in need of special storage space allocations and impose additional constraints. Thus, when providing a solution, these constraints have also to be taken into account.

This container SSAP can present both in the container vessel as well as in the CT yard. When this is in the container vessel, it concerns about the arrangement of containers in the vessel such that the containers which are supposed to be unloaded in the first port that the vessel stops at during its course are to be at the top of the container stack and the containers that are supposed to be unloaded at the vessel's final destination are to be at the bottom of the container stacks. With this container arrangement, it is expected to minimize the number of container re-handlings required when unloading containers from the vessel and thus to minimize the waiting time of the vessel at each port during its course. When addressing the container SSAP in the vessel, it is also very important to account for the stability of the ship. Since it is possible that different containers have different weights and these weight differences can be significant. To maintain the balance and stability of the vessel, it is thus important to distribute containers such that the weights of the containers are also accounted for. This will add additional complexity to the problem. Most of the previous researches on the container SSAP are either in the interest of the vessel or in the CT and have considered the re-handlings

only in either vessel or in the CY and not both. In this work, our interest is in the CY and we have taken into account the re-handlings both in the vessel and the CY. Furthermore, the minimizing of the total YC movements is also accounted for, in this work.

This is an optimization problem and falls under the NP-Hard class of problems and due to the nature of the problem, we will use a Genetic Algorithm (GA) for our solution.

Furthermore, the CY operations have been improving rapidly and will continue to be so in the future as well. Thus, most of the times, the CTs have to take significant time and efforts in upgrading their systems to support the potential changes in the container operations, changes in the configurations of the environment and in introducing new constraints. This is yet another problem the CTs have to address and we, in our work, design our solution such that these can easily be provisioned for.

1.5 Motivation

The operations and facilities involved with the equipment at the CTs are usually expensive as these equipment consume significant energy, time, experienced human labor etc. Furthermore, this equipment is limited resources in the CT and the operations expecting their services may form a queue at busy times. Thus, delay in one operation leads to delays in subsequent operations, which may result in the trucks, vessels etc. waiting. These ultimately affect the costing and the reputation of the CT and it is therefore very important to plan and schedule these operations carefully to ensure efficient use of these equipment as well as to avoid unnecessary operations, as they consume valuable time, experienced human labor (for operating the equipment), energy etc.

As pointed out in Steenken, D. et al‘s work [4], in the last 30 years, the efficiency of world-wide trade has increased by around 9.5% per year due to the improvement in container handling. With this growth in container handling, it becomes more and more competitive between CTs, especially among the ones within close geographic areas,

and thus it will necessitate the further optimizations. Table 1.1 below depicts the top 20 CTs and their throughput from 2012–2014 in Twenty-Foot Equivalent Unit (TEU) and percentage change [15].

Also, due to the nature of this problem, the number of containers can get large enough and the good results shown in the similar previous researches, which will be discussed in the literature review section, have motivated us to carry out this research.

Table 1.1: Top 20 container terminals and their throughput, 2012–2014 (TEUs and percentage change) [15]

Port Name	2012	2013	2014	Percentage change 2013–2012	Percentage change 2014–2013
Shanghai	32 529 000	36 617 000	35 290 000	12.57	-3.62
Singapore	31 649 400	32 600 000	33 869 000	3.00	3.89
Shenzhen	22 940 130	23 279 000	24 040 000	1.48	3.27
Hong Kong	23 117 000	22 352 000	22 200 000	-3.31	-0.68
Ningbo	15 670 000	17 351 000	19 450 000	10.73	12.10
Busan	17 046 177	17 686 000	18 678 000	3.75	5.61
Guangzhou	14 743 600	15 309 000	16 610 000	3.83	8.50
Qingdao	14 503 000	15 520 000	16 580 000	7.01	6.83
Dubai	13 270 000	13 641 000	15 200 000	2.80	11.43
Tianjin	12 300 000	13 000 000	14 060 000	5.69	8.15
Rotterdam	11 865 916	11 621 000	12 298 000	-2.06	5.83
Port Klang	10 001 495	10 350 000	10 946 000	3.48	5.76
Kaohsiung	9 781 221	9 938 000	10 593 000	1.60	6.59
Dalian	8 064 000	10 015 000	10 130 000	24.19	1.15
Hamburg	8 863 896	9 258 000	9 729 000	4.45	5.09
Antwerp	8 635 169	8 578 000	8 978 000	-0.66	4.66
Xiamen	7 201 700	8 008 000	8 572 000	11.20	7.04
Tanjung Pelepas	7 700 000	7 628 000	8 500 000	-0.94	11.43
Los Angeles	8 077 714	7 869 000	8 340 000	-2.58	5.99
Jakarta	6 100 000	6 171 000	6 053 000	1.16	-1.91
Total top 20	284 059 418	296 791 000	310 116 000	4.48	4.49

1.6 Research Problem

The primary research problem this work is trying to address is the container SSAP in CT. When a container vessel arrives at a CT, the containers are unloaded from the vessel and stored in the CY as multi-dimensional blocks of stacks. They are stacking due to the limitation of the storage space. This is how the containers are stored in the

container vessel too. The containers stored in the CY are on the temporary basis and each container has a deadline, according to which containers are retrieved from the CY. When a container to be retrieved is not on the top of the stack, container re-handling occurs when retrieving the required container. If a storage plan is done without the occurrences of re-handling in the CY, this in turn require containers in the vessel to be unloaded in an order compatible with the CY storage plan, as containers are directly move from the vessel to the CY. This introduces re-handlings in the container vessel. The re-handling in the vessel and in the CY are not mutually exclusive and minimizing at one end may lead to increased re-handling at the other. Furthermore, when containers are scattered in the CY locations in terms of their deadlines, the YC movements gets unnecessarily higher. The YC would have to move from one end of the block all the way to the other, if two containers of adjacent deadlines are in either side of the container block. Thus the re-handling at the vessel, at the CY and the unnecessary movements in the CY are all unproductive operations and should be avoided or minimized. Furthermore, the cost of single re-handling in container vessel may not be equal to that of CY. Similarly the cost of single YC movement is different from the costs of the re-handlings at vessel and the CY. This leads to the problem of minimizing all three operations such that the total cost is minimal. This is called container SSAP and is an optimization problem falling in the NP-Hard class of problems.

Furthermore, the configurations and environments in the CY tend to change fairly frequently and any solution should be able to tolerate these changes.

1.7 Objectives

The primary objective of this research is to explore the existing approaches/solutions to the SSAP, adopt and improve them to produce a solution to suit more realistic situations.

- To adopt a good GA capable of providing quality solutions (the GA approach is selected due to the unavailability of exact analytical solution due to the nature of the problem and the good results shown in previous researches, etc.)

- To produce a prototype version of a software tool to implement and evaluate our proposed solution, with the capability of handling potential changes in the environments such as Yard configurations.

- Consider the special storage requirements of container types such as Reefer and Open Top and make our solution and the software tool capable of handling these types.

CHAPTER 2

LITERATURE REVIEW

When we explore the previous work related to the container SSAP done by the researchers, it can be identified that various approaches such as Mathematical models, simulation based and heuristic methods have been used to solve this problem in various related researches, which will be discussed in the following sections.

2.1 Genetic Algorithm Approach

In computer science and operations research, GA is a meta-heuristic inspired by the process of natural selection that belongs to the larger class of evolutionary algorithms (EA) [31]. Genetic algorithms are commonly used to generate high-quality solutions to optimization and search problems by relying on biologically inspired operators such as mutation, crossover and selection [31].

The algorithm repeatedly modifies a population of individual solutions and at each step it randomly selects individuals from the current population and uses them as parents for producing children for the next generation [22].

The population for the first iteration, called the initial population, is usually generated randomly and within each iteration, for producing children, the genetic operators called selection, crossover and mutation are applied. Fitness is calculated for all the solutions in the population and solutions with higher fitness are allowed higher probability to retain in the population while weaker solutions are allowed lower probability. The selection operation selects the solutions with higher fitness values for the crossover and mutation operations required for the offspring. Mutation performs a small random change in the same solution while crossover combines two solutions to produce two new solutions.

Most of the previous authors have taken this approach due to the NP-Hard nature of the problem as well as the due to the possibility of getting medium to fairly large number of containers. GA is a well-known meta-heuristic approach that its efficiency is verified for many problems in the literature [23]. The encouraging results shown with this approach in previous researches has been yet another reason that most of the researchers were inspired to use this approach.

I. Ayachi et al. [1] presented a solution for the container SSAP at the CT, using a GA. In their work, they have also considered the different container types such as open top, refer, empty containers etc. Their work is not associated with any software/tool and they have presented the GA, claiming that the consideration of different container types as the main contribution of their work. Their main objective is to determine a valid container assignment in the CY in order to reduce the container unloading time as well as their re-handlings, while respecting the customer delivery deadlines. By inspecting the fitness function, we can observe that they comparatively prefer a re-handling in a later date than in a sooner date (based on the delivery deadline of the container).

In their GA implementation, they have used the roulette-wheel method for selecting the parents from the initial population and for generating new solutions, the two point crossover operator and mutation is used. The crossover operation has been produced with a probability fixed to 70%.

Somewhat similar work as of I. Ayachi et al [1] has been presented by Phatchara Sriphrabu et al. [2]. In their work, they have used a simulation model for container lifting with the aim of minimizing the total container lifting time. They have applied a GA for the allocation of containers to the storage locations. Their work is also for solving the SSAP at the CY of the CT, instead of the SSAP at the container vessel. But, their focus is more towards the outbound containers instead of inbound containers. They show, in their work, that the container storage space allocation at the CT is correlated with the storage plan of the container vessel.

The total lifting time was observed using the simulation model based on a GA for finding the best configuration, as well as solution of simulation based on the First-In, First-Storage. They claim that the results show that the simulation model based on the GA is more efficient.

The work presented by Riadh Moussi et al. [3] addresses the SSAP at the CT by providing two GAs: one-cut-point and two-cut-point. As a case study, they have used the CT of Normandy, in Le Havre port, in France. In their work, they modeled the seaport system with the intention of determining optimum storage space allocation for

various container handling schedules. The authors have developed a new container location model with the objective of minimizing the total distance of transport containers between the vessel berthing locations and their storage positions.

To prove the efficiency of the proposed GA, they have also solved the problem using Branch-and-Bound (B&B) method for the small scale problems, using the commercial software ILOG CPLEX. For the large scale problems, they have used the two GAs proposed. Computational results in the real application in the CT of Normandy show good quality of the solutions obtained by GA.

In the work from Mohammad Bazzazi et al [23], a GA approach is used to extend the work by Zhang et al and as the case study they have used the CT located at the south of Iran namely Shahid Rajaei terminal. They have also considered the different container types and have pointed out that the balancing workload between blocks is a critical element of the efficiency of the CT. They have aimed to minimize the vessel berthing times by balancing the workloads of YCs and QCs. With workloads of vessel dispersing in different blocks, the YCs in the blocks serve as parallel servers processing jobs for the vessel and the deberthing time of the vessel is the maximum processing time of these parallel jobs [23].

Using 22 numerical examples of different sizes, they have verified the performance of the proposed model and the developed GA. For the experiment, they have used LINGO 8.0 software on a personal computer with two Intel CoreTM2 T5600@1.83 GHz processors and 512 GB RAM and it was found that the small-sized examples were optimally solved by the B&B method. It was, however, not possible to solve the large-sized examples optimally within a reasonable CPU time with this B&B method. Then all the examples are solved with the GA too and the results were compared in terms of the objective function value and CPU time. Each example is solved by GA 20 times and the mean of OFV and CPU time are reported. A feasible solution, however, could not be found even after 3 hours with the B&B method. The figure 2.1 below depicts the comparison between B&B and GA methods, in terms of CPU Times.

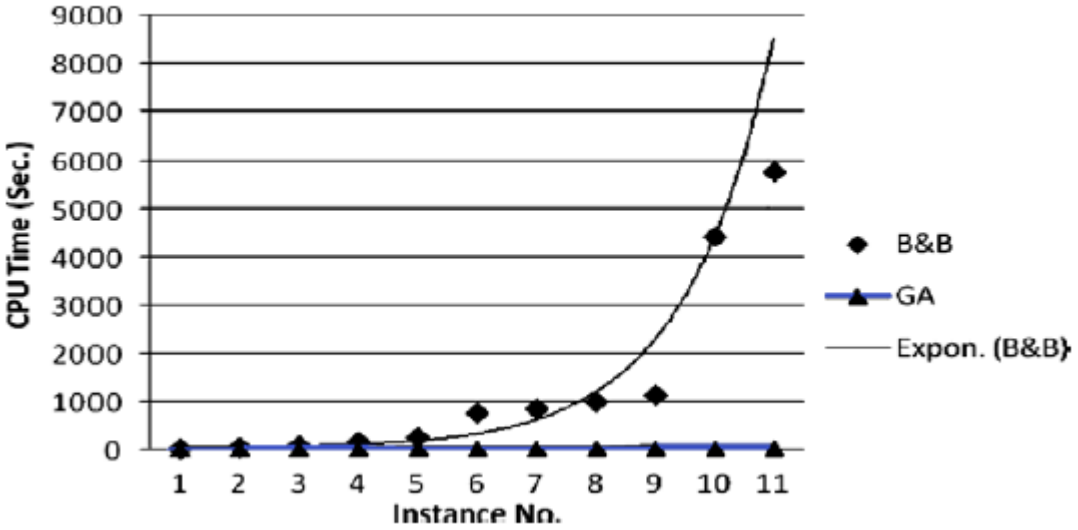


Figure 2.1: Comparison between B&B and GA's CPU times

2.2 Harmony Search Approach

Harmony search is a phenomenon-mimicking metaheuristic introduced in 2001 by Zong Woo Geem, Joong Hoon Kim, and G. V. Loganathan. Harmony search is inspired by the improvisation process of jazz musicians [6]. When musicians compose the harmony, they usually try various possible combinations of the music pitches stored in their memory, which can be considered as an optimization process of adjusting the input (pitches) to obtain the optimal output (perfect harmony). Harmony search draws the inspiration from harmony improvisation, and has gained considerable results in the field of optimization [5]. Figure 2.2 depicts the Flow of the HS algorithm.

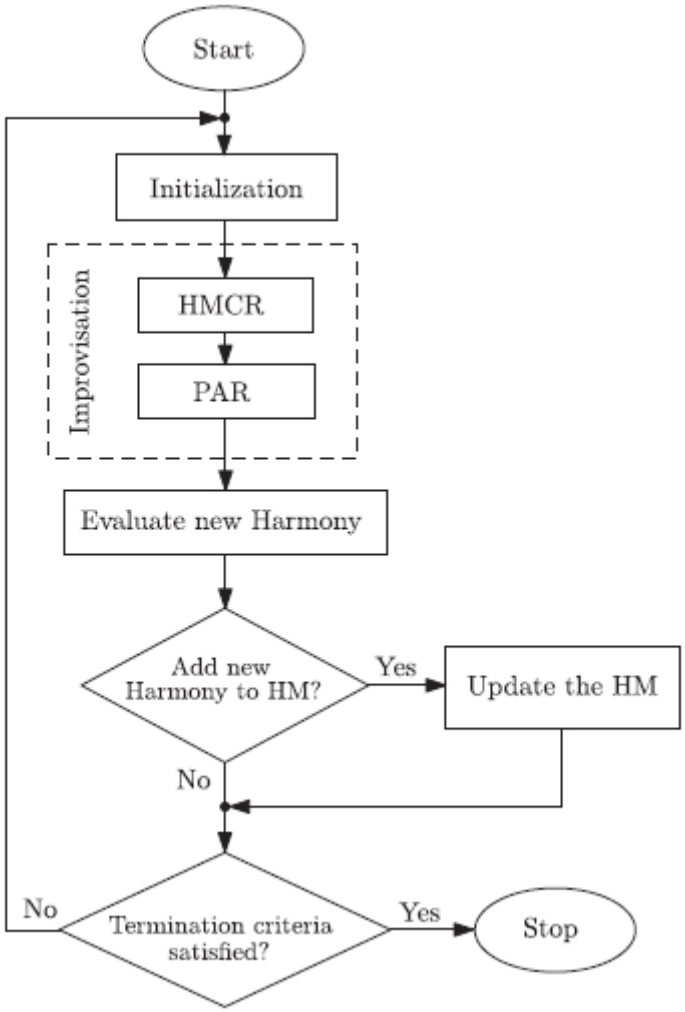


Figure 2.2: Flowchart of the HS algorithm [10]

I. Ayachi et al [7] have applied an adaptation of the Harmony Search (HS) algorithm to the container SSAP and have presented some experimental results. They claim that good results were obtained.

2.3 Simulation Based Approach

Although this is discussed as a separate approach, this approach can be combined with other approaches such as GAs. Simulation-based approach is yet another popular approach for CT related researches and have shown good results. With this approach, it allows discrete event simulation model (amongst others) for the real life processes. Use of this approach allows studying the behavior of the operations without actually

disturbing or interfering with it. It may also help find unexpected behaviors of the system and easy to perform the 'what if' analysis. However, some of the apparent problems associated with this approach are that being it expensive and time consuming and also the difficulty in interpreting the results.

Gamal Abd El-Nasser et al, in their work [11], have used this approach. Their paper propose a developed approach using discrete-event simulation modeling to optimize solution for SSAP, taking into account all various interrelated CT handling activities. They claim that this approach of discrete event simulation is suitable since the problem being studied is stochastic, dynamic and discrete. Flexsim software was used as the discrete-event simulation tool for their research. The openness and flexibility provided by Flexsim as well as the ability to model as discrete [13] are some motivation factors for the researcher to use Flexsim as their simulation tool. The figure 2.3 below depicts a snapshot of container Terminal from the simulation model run

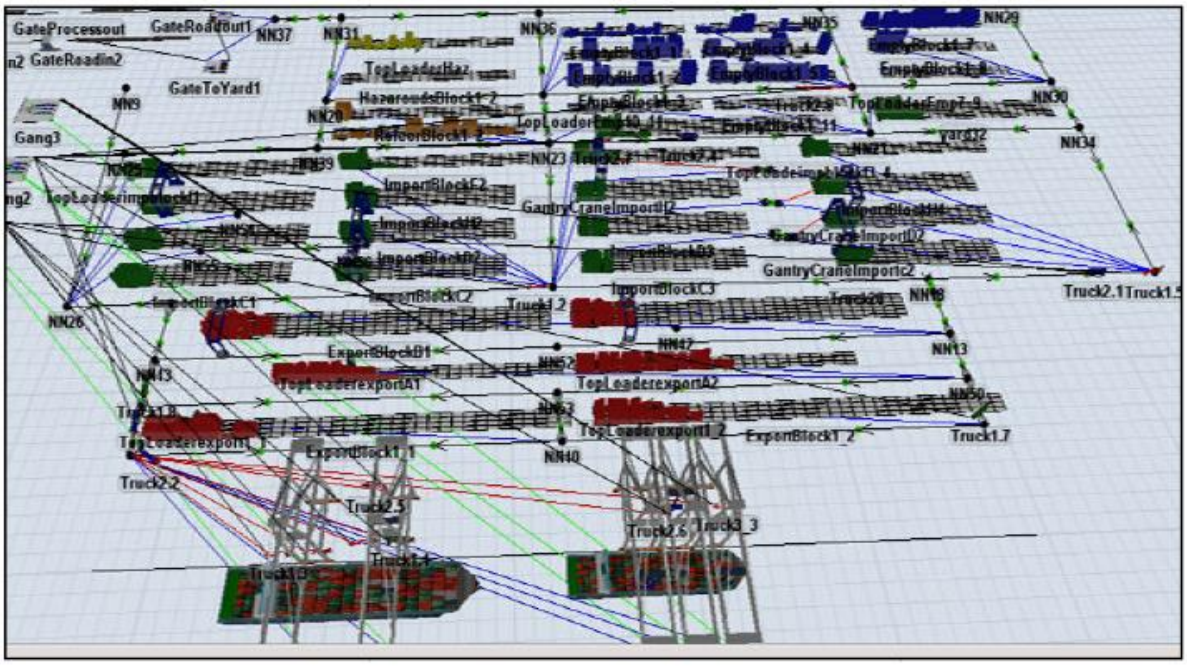


Figure 2.3: A snapshot of container Terminal from the simulation model run

The proposed approach is applied on a real case study data of CT at Alexandria port. One week's data from the Alexandria port was used as the input data for the

simulation model. As for the outcome, they claim that their model for optimization of storage space allocation showed up around 54% reduction in container handling time.

2.4 Rolling-Horizon Approach

This approach is also used by some researches in addressing the container SSAP and have shown good results. With the Rolling-Horizon approach, it solves the problem in a sequence of iterations, and the part of the planning horizon will be modeled in detail within each iteration. And the rest of the time horizon is represented in an aggregated manner [20], [21]

Chuqian Zhang et al, in their work [19], have used the Rolling-Horizon approach and their work is also for the CY instead of at the container vessel. As a case study, they have used Hong Kong container terminals. They have, for each horizon, decomposed the problem into two levels. Then each level was formed as a mathematical model and in the first level, the total number of containers that are supposed to be put in each block for each time period of the planning horizon is made to balance the two types of workloads amongst blocks. The number of containers associated with each vessel that constitutes the total number of containers in each block in each period is then determined by the second level with the intent of minimizing the total container transport distance between vessel berthing area and storage areas. They claim that their experimental results show that the workload imbalances in the storage blocks could be significantly reduced with a short computational time.

2.5 Multitier architecture

Multi-tier architecture is a client-server architecture in which data access, business logic and presentation functionalities are separated [26]. In Multi Layered architecture this separation is logical while in the Multi-Tier architecture this is separated physically, allowing higher scalability by distributing the workload of different tiers in different servers.

This architecture has been in the industry for a long time and time tested. While having many benefits such as enhanced maintainability, readability, scalability etc., this architectural style is CRUD based and is also database centric style, in which databases or data access layer is at the core and other layers (directly or indirectly) depend on this. Due to this drawback, new architectures (or its variants) such as Onion Architecture, Clean Architecture, etc. have been evolved which allow to hold the enterprise business rules and application business rules at the core and to be independent and for other parts to depend on these. The dependencies between these layers are managed using the Inversion of Control mechanisms.

In terms of our work, even though the proposed software tool is of prototype implementation with only the core functionality integrated, as a future work, it is expected to be evolved to a comprehensive tool with enterprise integration facilities to be operative with other systems. When considering the interaction with the outside world – with other local and overseas companies such as shipping liners, import and export companies, insurance agents, local transportation companies, government bodies and authorities such as customs, it makes sense to use Multi-Tier web based architecture including a Restful service layer and the presentation layer being decoupled from the server side implementations using client side frameworks such as in Single Page Application architecture.

Despite the benefits of the Multi-Tier and similar architectures, which are usually suited for business applications, they, in turn, introduces performance penalties due to the communication between different tiers. This makes it not suitable for systems which require high performance or for systems which involve intensive data processing/computations. Since this system is also involved with intensive data processing while executing the GA, this architecture is not suitable for the implementation of the proposed work, despite its advantages.

2.6 Pipe and Filter Architecture

This architecture is suitable for systems when a single event triggers a sequence of processing steps, each performing a specific function [24]. It is used to divide a larger processing task into a sequence of smaller, independent processing steps (Filters) that are connected by channels (Pipes) [24]. In other words, this architecture is suitable when there is a continuous data flow which needs a chain of processes. The pipes and the filters are the main components of this architecture and the responsibilities of a pipe component are to transfer data between filters, sometimes to buffer data or to synchronize activity between neighboring filters [27], while the responsibilities of a filter component are to get input data from a pipe, to perform an operation on its local data, and to send output result data to one or several pipes [27]. As shown in [28] and [29], Unix Shell is a good example where the pipes and filters are used very effectively.

Considering the amount of data processing and the ability to separate the implementation of GA into individual set of processing (functions), this architectural style will be adopted for the implementation of the proposed solution in this work.

Decoupling the components from one another and being able to maintain independently, being able to be tested in isolation, ability to reuse components, and being able to add, remove, reorder or reorganize are some of the benefits gained from this architecture. Figure 2.4 depicts an example of pipe and filter architecture.

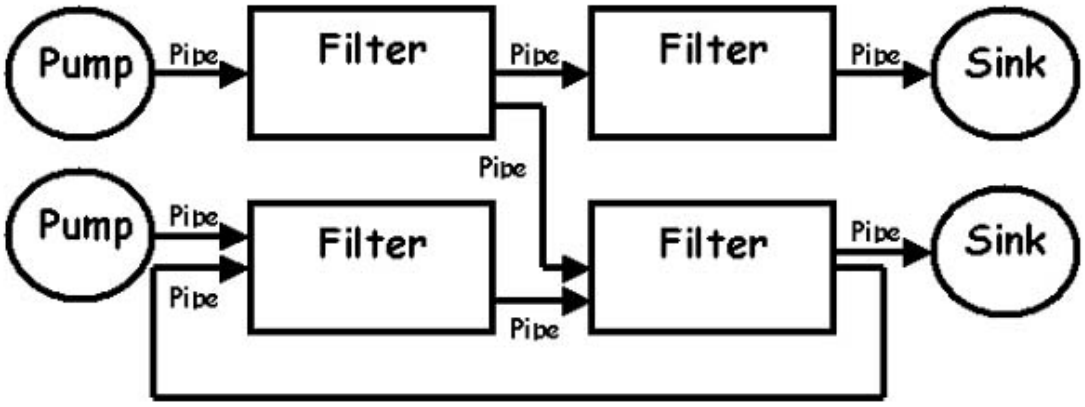


Figure 2.4: An example of Pipe and Filter architecture [14]

2.7 Summary of Related Work

The related work specified above have been summarized into tabular form and are shown in the table 2.1

Table 2.1: Summary of Related Work.

Author/Date	Topic/Focus/Question	Approach	Context/Setting	Findings
I. Ayachi, R. Kammarti, M. Ksouri and P. Borne, 2010.	A Genetic algorithm to solve the container storage space allocation problem Main contribution is it solves the problem with different containers types	GA	Applied in CY instead of Vessel	Compared against LIFO and better results found with GA based solutions.
Phatchara Sriphrabu, Kanchana Sethanan, and Banchar Arnonkijpanich, 2013	A Solution of the Container Stacking Problem by Genetic Algorithm The main objective is to minimizing the total container lifting time	Simulation Model with GA	Applied in CY instead of Vessel	Compared against the first in first storage and found better results with GA based solutions.
Riadh MOUSSI, Ndèye Fatma NDIAYE, Adnan YASSINE, 2011	A Genetic Algorithm And New Modeling To Solve Container Location Problem In Port The main objective is to minimizing the total distance between the vessel and storage positions	Two GAs: one-cut-point and two-cut-point	Applied in CY instead of Vessel. As a case study, they have used the CT of Normandy l	Compared against Branch-and-Bound and found GA results better
Mohammad Bazzazi, Nima Safaei, Nikbakhsh Javadian, 2009	A genetic algorithm to solve the storage space allocation problem in a container terminal Considered different container types. Aiming to minimize the vessel berthing times by balancing the workloads of YCs and QCs	GA	As the case study, used at Shahid Rajaei terminal, Iran	Compared against B & B and GA produced better results. Balancing workload between blocks is a critical element of the efficiency of the CT

I. Ayachi, R. Kammarti, M. Ksouri, P. Borne, 2010	Harmony Search Algorithm For The Container Storage Problem. The objective is to determine a valid containers arrangement, which meets customers' delivery deadlines, reduces the number of container re-handlings	Harmony Search	Applied in CY instead of Vessel	Compared against a metaheuristics algorithm and claim good results found.
Gamal Abd El-Nasser A. Said, El-Sayed M. El-Horbaty, 2015	A Simulation Modeling Approach for Optimization of Storage Space Allocation in Container Terminal	Discrete-event simulation modeling. Flexsim software was used as the discrete-event simulation tool	Applied as a case study at Alexandria port	Applied to 1week's real data set and claim that reduction of container handling time is achieved.
Chuqian Zhang, Jiyin Liu, Yat-wah Wan, Katta G. Murty, Richard J. Linn, 2003	Storage space allocation in container terminals	Rolling-Horizon	Applied in CY As a case study, they have used Hong Kong container terminals	Claim that the workload imbalances in the storage blocks could be significantly reduced

CHAPTER 3

METHODOLOGY

As discussed in the literature review section, there exists several related researches of different approaches. As far as the author knows, none of them aim to minimize the re-handling in both the CY and the vessel. Moreover, none of them aim to minimize both the re-handling as well as the YC movements.

3.1 Proposed Solution

In our solution, a GA to the SSAP is provided, implemented and will be associated with a software tool that allows dynamic yard configurations. As an additional contribution of our work, the software tool allows the tuning of the algorithm parameters dynamically, such that it can be easily experimented. Accordingly, the output of our work will be in the form of a software tool with the core functionality implemented.

This software tool is implemented as a desktop application and even though this is for the demonstration and evaluation purposes of the effectiveness of the algorithm, the Pipe and Filter Architecture is used considering its nature of operations.

This software tool, when run, shows the user the existing yard configurations as well as the parameters of the algorithm and allows the user to change them if required. If run for the first time certain parameters such as yard configurations are shown as blank as user is required to provide them. Once all the parameters required are provided, user can enter the details of a container shipment and run the solution such that the tool can find the optimum or near optimum solution as per the algorithm. This solution is designed per shipment basis and for each shipment the solution can be run to find the optimum/near optimum solution for that shipment. Amongst the shipment data expected to be fed into the system are the number of containers in each type (Regular, Open Top and Reefer), minimum and maximum deadline dates (lower and upper boundaries) and stack heights of Regular/Open Top and Reefer container stacks. When running the solution after all the required data is fed into, the system execute the GA by starting with creating the Initial Population. The GA used in our work is described step by step in the section 3.2

3.1.1 Container Shipment Representation

Container shipment is the source of data for our proposed system and hence it's of great importance to study in detail. Container vessels usually, but not necessarily, transport containers not only to one destination container port; hence, vessels usually store the containers according to their own plan to suit their multiple destinations, not considering or respecting the order preferred by each and every CYs to support their deadline based storage plan. Vessel planners also consider the weight of the containers when stacking them in the vessel such that the weight is evenly distributed to maintain the ship stability. Moreover, these deadlines associated with containers are internal property/aspect agreed between the importers and the CT and are not something the vessels, though they may be visible to them, are interested in.

When storing the containers in the vessels, they are stored in stacks, just as in CYs, since it would otherwise find that the storage space is insufficient. Vessels of different size and container capacities arrive at the CTs and container storage area dimensions may vary from vessel to vessel. That is, one vessel may have stacks with four tiers whilst it is five or six tiers for another. Similarly a vessel may have twelve rows/lanes in the storage block whilst another may have eighteen or twenty rows. But when it comes to different types of container, such as Reefer containers, vessels too, just like in the CTs, have to store them in separate blocks in which access to the power supply is available.

Thus, when modeling the container data in a vessel, unlike the modeling for the containers in the yard, the three dimensional modeling (with Bays, Rows, and Tiers) are avoided and only Block Type, a Stack No and a Tier No were used instead to represent the stacks of containers in a linear way. That is, a unique number is assigned for each stack and the (same) convention used for numbering the stacks should be used across the entire operations until all the containers are unloaded. Figure 3.1 shows how containers are assigned to stacks in a shipment.

Figure 3.1: Container assignment in the shipment.

3.1.2 Container Representation in the Yard

The storage of containers in the CY is modeled as three dimensional structure, in contrast to the linear representation of the containers in the vessel. This is due the fact that the layout of the CY is pre known and the number of bays, rows and tiers in a storage block of a particular CT change less often compared to the vessels, even though the proposed solution accommodates the changes in these parameters. Hence the containers in the CY are modeled using the attributes Block Type, Block No, Bay No, Row No, Tier No and The Container itself. The CYs maintain two types of blocks, one for the Regular and Open Top containers and the other for the Reefer containers where access to power is available. The figure 3.2 shows the layout of the CY and the container assignment in a block in the CY.

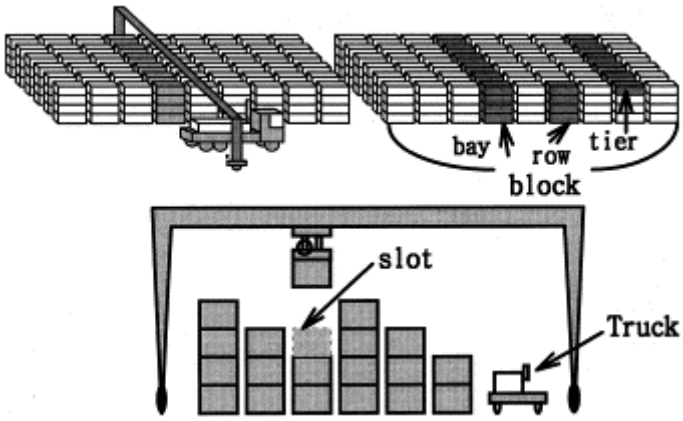


Figure 3.2: Container assignment in the Container Yard

3.2 The Genetic Algorithm Used in the Proposed Solution

The Figure 3.3 shows the GA used in our proposed solution.

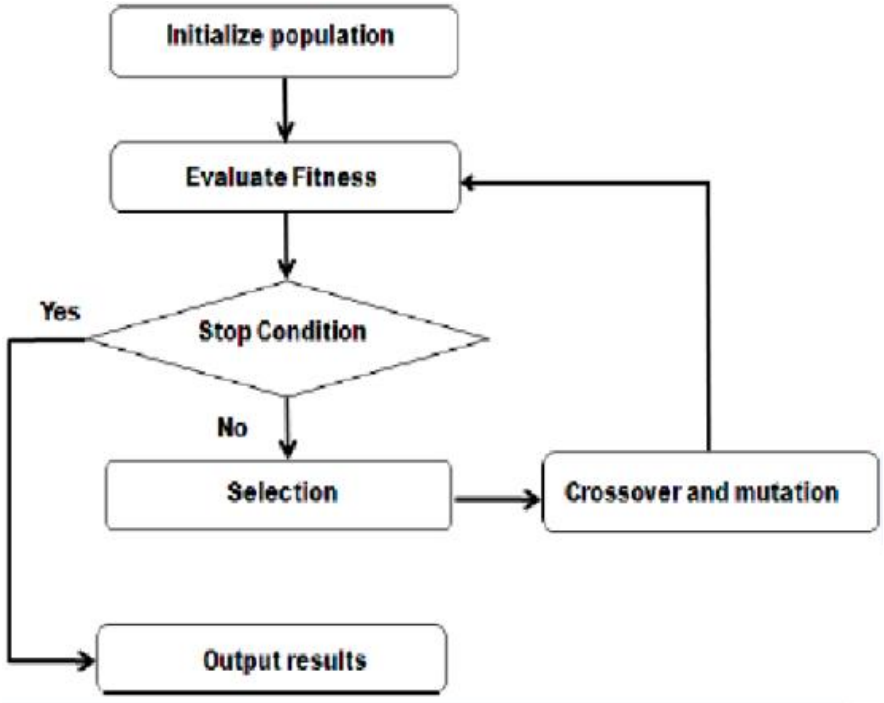


Figure 3.3: Genetic Algorithm [25]

3.2.1 Initial Population

Creating the initial population is the very first step in the GAs and the initial population is a collection of different solutions usually, but not necessarily, generated randomly. Although some GAs may use Heuristics to create initial population, in this work we generate the initial population randomly since the container assignment in the vessel is of random nature in terms of the deadline and the initial population is based on this shipment data. One solution amongst the population is also called a chromosome and this chromosome (one chromosome) is a list of Yard Container Location objects. One chromosome, in other terms, is one representation of the container assignment in the CY for a single shipment. One Yard Container Location object, which represent a container placed in a particular location of the CY as described in 3.1.2, in turn, is an object that reflects both a single container with its details (including deadline, container type, container number etc) and the location of that container in the CY, denoted by Block Type, Block No, Bay No, Row No, Tier No. Below is an example of a single chromosome that represent n number of containers in the CY. The population size is made as a parameter set by the user, subject to validations, based on the number of containers to be assigned.

Example of a JSON representation of a single chromosome for n containers assigned in the CY

```
{
    [
    {
        Container: { No: “1”, Type: “Regular”, Deadline: “2019-11-20”},
        BlockType: “RegAndOpt”,
        BlockNo: “1”,
        BayNo: “1”,
        RowNo: “1”,
        TierNo: “1”
    },
    {
        Container: { No: “2”, Type: “Regular”, Deadline: “2019-11-22”},
```

BlockType: “RegAndOpt”,
BlockNo: “1”,
BayNo: “1”,
RowNo:”1”,
TierNo: “2”
},
.
.
.,
{
Container: { No: “n”, Type: “Regular”, Deadline: “2019-13-15”},
BlockType: “RegAndOpt”,
BlockNo: “2”,
BayNo: “23”,
RowNo: “4”,
TierNo: “5”
}
]
}

The following steps are executed in the order they are specified when creating the Initial Population.

I) Get a copy of Regular container list and randomize

II) Get a copy of Open Top container list and randomize

III) Get a copy of Reefer container list and randomize

IV) Create the list of stacks required for Regular and Open Top containers, and randomize the list.

This step can be decomposed into two steps – Calculating the number of stacks required, the creating of that number of stacks and keeping in a list of stacks objects, in which containers are not added yet (empty list of stack objects) and finally

randomizing that list of stacks except the incomplete stacks. When randomizing, it is made sure that the incomplete stacks stay at the tail of the list of stacks.

As for the former, if the following notations are used:

Number of stacks = Nb(stacks),
Number of tiers per stack = Nb(tiers),
Number of Regular containers = Nb(reg),
Number of Open Top Containers = Nb(opt)

The stacks required can be found as:

$$Nb(stacks) = \begin{cases} Nb(opt) & \text{if } Nb(reg) <= (Nb(tiers) - 1) * Nb(opt) \\ \begin{cases} Nb(opt) + Nb(excess) / Nb(tiers) & \text{if } Nb(excess) \,\%\, Nb(tiers) = 0 \\ Nb(opt) + Nb(excess) / Nb(tiers) + 1 & \text{Otherwise} \end{cases} \end{cases}$$

where $Nb(excess) = Nb(reg) - (Nb(tiers) - 1) * Nb(opt)$

As for the latter, stacks should start with a new stack location (in a new Row) next to the one ended up with the previous shipment and then advance to the next Row and so on until the end of the Row of that Bay. Once the end of the Row is reached, start with the first Row of next Bay and so on and if the end of a Bay is reached, start with the first Row of the first Bay of the next Block.

V) Create the list of stacks required for Reefer containers and randomize the list.

This is similar to the step in IV) except that only the Reefer containers are applicable in this step whereas Regular and Open Top containers are applicable in the previous step.

If the following notations are used:

Number of stacks = Nb(stacks),
Number of tiers per stack = Nb(tiers),
Number of Reefer containers = Nb(reefer)

The stacks required can be found as:

$$Nb(stacks) = \begin{cases} Nb(reefer) / Nb(tiers) & \text{if } Nb(reefer) \% Nb(tiers) = 0 \\ Nb(reefer) / Nb(tiers) + 1 & \text{Otherwise} \end{cases}$$

The creating the list of stacks is also similar as in the Regular and Open Top containers except that the stacks are in the Reefer blocks instead of Regular and Open Top blocks. When randomizing, the incomplete stacks are placed at the tail of the list of stacks.

VI) Adding Regular and Open Top Containers to the stacks

For each stack add regular containers from the list of regular containers taken from the shipment starting from bottom tier until the top tier but one (add Nb(tiers) – 1 containers), keeping the top tier for the Open Top container. If there are no Open Top containers, fill this tier too with a Regular container, otherwise place an Open Top container at the top from the list of Open Top containers taken from the shipment. The randomness is ensured since the list of stacks is randomized. The incomplete stacks, if any, will be at the end since the incomplete stacks are ignored by the randomize function.

VII) Adding Reefer containers to the stacks.

This is similar but less complicated than in the Regular and Open Top containers and is only applicable to Reefer containers. For each stack, add Reefer containers from the list of Reefer containers taken from the shipment starting from bottom tier all the way up to the top tier. The randomness is ensured since the list of stacks are randomized and the incomplete stacks, if any, stay at the end.

3.2.2 Fitness Evaluation

Fitness function/evaluation is one of the most important features of a GA and for our solution, being an implementation of a GA, the Fitness evaluation, hence, becomes an

integral aspect of the solution. For the fitness evaluation of this work, three values are calculated separately for the three operations used, which are re-handlings in the vessel, re-handlings in the CY and total YC movements across the Bays, in evaluating the fitness. By establishing a relationship between these three values, we aim to aggregate these three values to obtain a single value representing the quality of the solution in terms of all three operations. All these three values indicate the costs or overheads of three operations used and the relationship between these values represents the relative complexities (complexity ratio/percentage) between these operations. This relationship is specified as a percentage and percentage values, in fact, may vary from CY to CY depending on the instruments used. Hence, this relationship values are set as runtime parameters that can be specified by the users.

All these values are costs and hence the aggregated single value depicts a cost instead of a quality or a fitness. Thus, the multiplicative inverse (Reciprocal) is obtained to get the fitness value.

I) Getting the Total number of re-handlings in the vessel.

Out of three values calculated to find the fitness of a particular solution (chromosome) is the total number of re-handlings required in the vessel when unloading the containers off the vessel in the order required to maintain the storage plan in the CY.

When calculating, the re-handlings in two different block types (Regular and Open Top and Reefer block types) are calculated separately and added together, since the re-handling in a Regular and Open Top block is similar to a re-handling in a Reefer block.

II) Getting the total number of re-handlings in the container yard

This is another value out of the three values calculated to find the fitness of a particular solution and this refers to the total number of re-handlings required when unloading the containers from the CY in the order of their deadlines. As in the calculation of re-handlings in the vessel, the calculation of re-handlings in here too is done separately for different block types and are added together to get the total

number of re-handlings in the CY. A one re-handling in a Regular and Open Top block equals to a re-handling in a Reefer block.

III) Getting the total Yard Crane movements across the Bays.

This is the last value of the three values required when calculating the fitness of a particular solution. This refers to the required total number of movements of the YC across the Bays when unloading the containers from the CY in the order of their deadlines. One movement represents the distance the YC has to travel to arrive at an adjacent Bay (previous or next Bay).

If we represent any container by Ct(i) and the container with the next deadline in the same block by Ct(i+1) then the number of movements between two containers is

| Ct(i).BayNo – Ct(i+1).BayNo |

Thus, if the total movements is represented by M, the number of blocks by Nb(B), container i in block b by Ct(b,i) (containers ordered by deadline such that container i+1 has the deadline next the deadline of container i), and the number of containers in block b by Nb(Cb), then:

$$M = \sum_{b=1}^{Nb(B)} \sum_{i=1}^{Nb(Cb)-1} |Ct(b,i).BayNo - Ct(b,i+1).BayNo|$$

3.2.3 Selection

The selection is an important step in any GA and it allows choosing the parents in the population and usually putting in a mating list for the offspring process. Usually the selection methods choose the parents with higher fitness values or give them the higher probability to be selected for the offspring. In our solution, the 50% of the population having the best fitness is selected and added to the mating list, deleting the worst 50% of the solution from the population. The population size is kept constant by adding the new chromosomes resulted in the process of mating of the selected parents. Since there is no guarantee that the offspring give the better solutions always,

keeping the solutions used to generate the offspring in the population makes sure at least the existing good solutions are remained in the population.

When selecting the parents for the offspring, even number of parents are selected since the crossover is done in pairs. The mating list is in the descending order of the quality placing the best solution at the first index and the worst at the last.

When selecting for the cross over operation, the mating list is divided in two from the middle and select the first solution of the first list with the first solution of the second list and so on until the last solution of the first list is with the last solution of the second list. The figure 3.4 shows the divided mating list and the selection of solutions for the cross over operation.

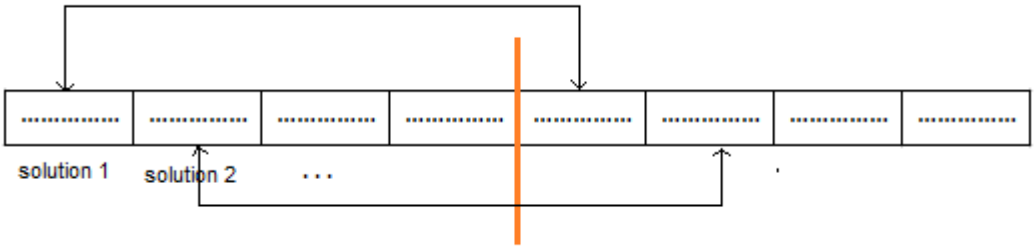


Figure 3.4: Divided mating list and the selection for the crossover operation.

3.2.4 Crossover

Crossover is yet another important genetic operation used in GAs and is also called re-combination. In our solution, two solutions are re-combined such that a part of a solution joins a part of another solution to form a new solution. The remaining parts of the two solutions are joined together to form the other solution. We, in our work, use the single point crossover method since we are using the best fifty percent of the population for the crossover and too much mixing of two good solutions may spoil the good solutions. Thus, in single point crossover it is expected that while allowing the inheritance of good features from two parents, also to make sure that a good portion of good features of each parent is maintained in the new children.

Cross over operation is not applied to all the solutions pairs in the mating list and based on the cross over probability provided, it is decided as to apply this operation or not for a given pair. To find out if the given pair of solutions falls within the probability, a random floating point number r, where 0 < = r < 1, is generated and if r is less than the cross over probability, then the given pair of solutions is crossed over. The figure 3.5 shows how parts of two solutions are used in the single point cross over operation.

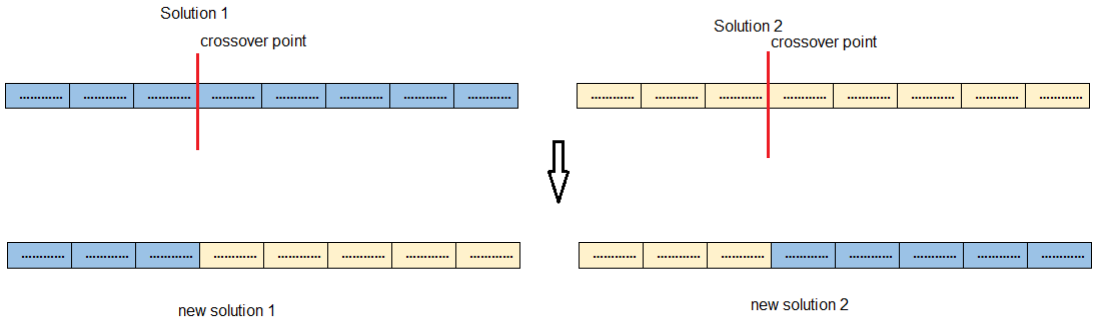


Figure 3.5: Crossover of two solutions

Crossover is applied separately for different block types since it would otherwise mix the Regular/Open Top container with the Reefer containers, which is practically not possible.

When selecting the cross over point, a random point is selected ignoring some percentage for the margins in both sides of the solution, in order for the crossover point not to be fallen on or very near the edges. One of the biggest challenges the crossover operation introduces is the duplication and while it is acceptable for certain GA applications, for this work, it is not acceptable since the duplication of containers is not at all possible. Thus when executing the cross over, the duplicates are checked for and if found they are replaced with the Genes of the same solution.

3.2.5 Mutation

This is another genetic operation used in GAs and this is a small random change done within a single chromosome. This is usually allowed at a low probability since it may otherwise spoil good solutions. Since, in this work, the quality of the solution is determined by the location of the containers (assignment), a small random change in the solution means changing the locations of a few containers randomly. Thus, the locations of a small percentage of containers are changed randomly to perform the mutation. Similar to the crossover, this step too is executed separately for the different block types to avoid the mixing of Regular/Open Top containers with the Reefer containers.

Unlike in the crossover operation, in the mutation, in this work, it is important to consider the any potential incomplete stacks. If special attention is not given for the incomplete stacks when mutating, it may be possible that the incomplete stacks are placed in the middle of the solution. For each shipment, usually, the incomplete stacks are placed at the last stack (or stacks if Open Top containers are more than the number of stacks) of that shipment and the container stacking for a new shipment will be started in a new stack. Hence, the incomplete stacks are ignored when randomly selecting the stacks of containers for mutating. Just like in crossover operation, a random floating point number between 0 and 1 is generated to apply the probability for the mutation operation. Below are the steps used in mutation process briefly.

I) Get a copy of the original solution

II) Select two stacks randomly and assign to two variables

III) Remove those two stacks from the copy of the solution (to avoid the same stacks being selected again)

IV) Locate those selected two stacks in the original solution (location coordinates – bay no, etc.) and interchange

V) Repeat from (including) the step II until required percentage of the stacks are mutated.

3.3 Solution Architecture and Implementation

While there exists numerous architectural styles, patterns and the methods for developing software solutions, there is no such thing as universally accepted best architecture that suits all type of projects. It is rather based on the project/solution in hand and its functional and non-functional requirements that a suitable architecture is adopted or selected.

Even though the implementation of our solution is of prototype level, the integration of a suitable software architecture is very important considering the amount of data processing and the computations this software tool is involved with. Thus the pipe and filter architecture is used for our software tool due to its nature of operations.

3.3.1 Model for the Container in the Vessel

Thus, the figure 3.6 describes a single container with its location in the vessel:

```
shipContLocation: {
        shipmentNo: int,
        blockType: { RegularAndOpenTop: 0, Reefer: 1},
        stackNo: int,
        tierNo: int,
        container: { contNo: int, contType: { Regular: 0,
          Reefer: 1, OpenTop: 2},
        deadline: datetime
}
```

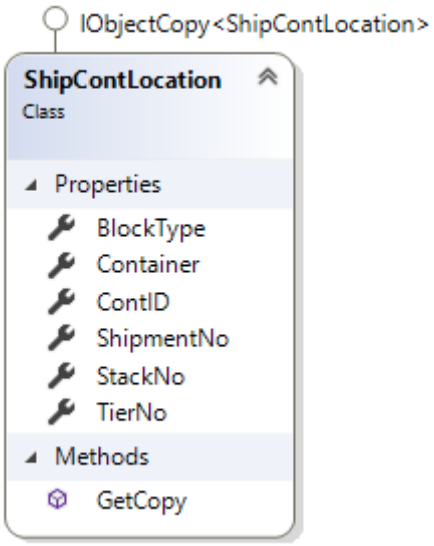


Figure 3.6: Model for the Container in the Vessel

3.3.2 Model for the Container in the Yard

The figure 3.7 shows the model for the container in the yard.

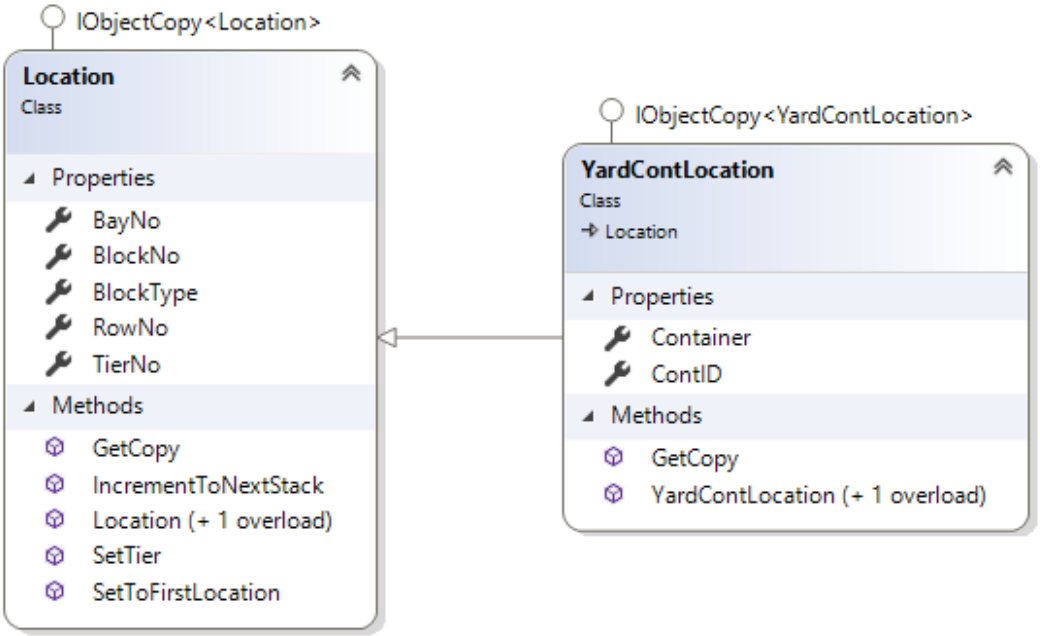


Figure 3.7: Model for the Container in the Yard

3.3.3 Integration of Pipe and Filter Architecture

As described in the literature review chapter, Pipe and Filter architecture is well suited for our software tool due to its nature of operations and the number of computations it is involved with.

For the Pipes, we will create a Generic Pipe class which can be reused for all type of pipes. This allows deferring specification of the type of the pipe while still providing the compile time type safety due to the strongly typed nature of this generic pipe implementation. This pipe is inherited from the generic List class and hence having all the features of the list class.

The figure 3.8 shows the process of fitness calculation, modeled as per the pipe and filter architecture.

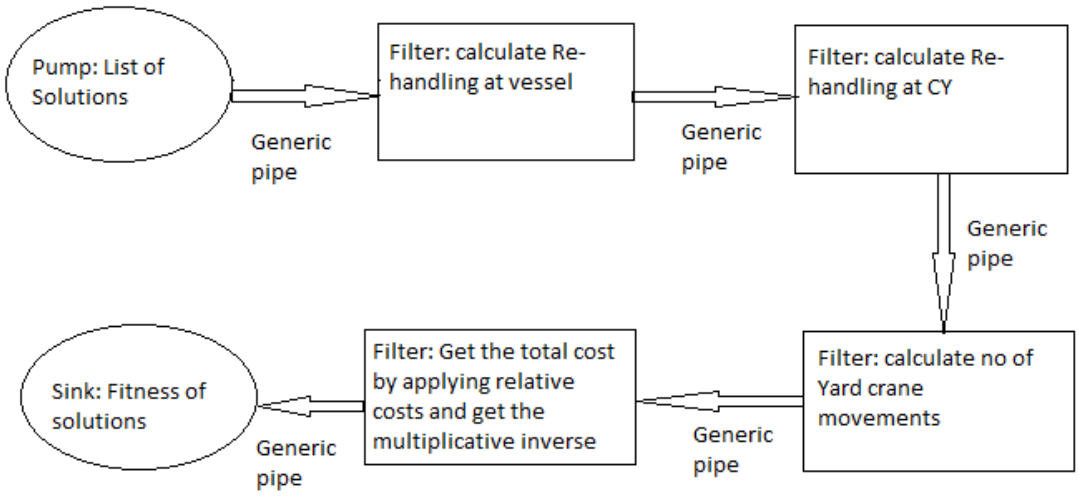


Figure 3.8: Fitness calculation process in Pipe and Filter Architecture

CHAPTER 4

SYSTEM EVALUATION

As specified in the Introduction chapter and as would be discussed in detail later in 4.2 section of this chapter, we, in the evaluation of our system, compare the fitness of the output of our work against both the standard LIFO approach and against the optimized LIFO approach, which is one of the best approaches that can be used in manual process and which resembles close to an analytical approach.

Once the evaluation mechanism is determined, the next important step towards the evaluation of the proposed solution is to prepare a set of test data to be used as input for the system. For this, random data generated programmatically is used, as described in the section 4.1 below. Once the test data is fed into the system, run and experimented, the analysis is done based on the observation and the interpretation of the test results.

4.1 Generation of Shipment Data as the Test Data

The input for our proposed solution is the container shipment data. Hence, the data required to evaluate the solution presented in this work is the container shipment data. Even though there is a lot of data (attributes) associated with a container shipment, for our proposed solution, only a few attributes of shipment data is sufficient as our optimization algorithm requires only few of them. Thus, in the interest of our solution, the data attributes related to container shipment has been identified as follows:

- Container Number: Integer – A unique number to distinguish a container
- Container Type: Enumeration of (Regular, Reefer or Open Top) – The type of the container.
- Deadline: Date – This is the deadline date at which the container is to be taken from the CT by the party imported the container.

Even though there are many data attributes related to a container shipment, only few attributes are of interest in terms of evaluating our system and since even those few data attributes are simple to artificially/programmatically produce with close

resemblance to the real shipment data, we generate those data programmatically with some randomness.

Out of the above data attributes, container number is only for the identification of a container uniquely and other than the identification purposes, in terms of evaluation, it is not necessary to represent an actual (real) container number. Hence, we do believe that assigning a sequential integer value programmatically is fair and reasonable for this field.

The Container Type field is important in terms of the distribution of containers in each type. While the Regular containers make up most of the shipment, studies show that Reefer and Open Top containers contribute to around 8 percent and 1 percent respectively. Thus, for the evaluation, around 8 percent of Reefer containers and around 1 percent of Open Top containers were used while Regular containers were used for the rest.

Deadline is the most important field for the evaluation since it is the data attribute the whole optimization process is based on. Deadlines are assigned randomly subject to upper and lower boundaries and this assignment of random dates are justifiable since the deadlines in a real container shipment are of random nature.

When generating the shipments as Test input for the evaluation, containers of different types are programmatically created and assign a random deadline date for each container. For creating random dates, a list of distinct dates between min and max limits are added to a list and the dates are picked from the list one by one based on the index which is generated as a random integer between zero and one below the list length, using the Random class provided in the application program framework.

4.2 Evaluation Techniques

While there are various evaluation techniques, we use measurement based evaluation technique for our solution. Since our work is incorporated with a software tool, even though of prototype implementation, the measurement based evaluation is made conveniently feasible. As always the case with GA based solutions, the fitness value

is used as the evaluation criteria/metrics. Since our evaluation criteria is the fitness value and since the calculation of fitness value is well defined and produces exactly the same value over repeated calculation and does not deviate from the true value, the statistical procedures to assess measurement errors such as Standard Error of Measurements are not necessary to be accounted for.

As described earlier, in the evaluation process, we compare the results of GA against both the LIFO as well as against the optimized LIFO approach, which is not only aiming to minimize or eliminate the re-handlings but also to minimize the YC movements, that can be used in the manual process. The suitability of the used evaluation technique for this work can be justified as follows:

- Measurements give the most reliable results and there are no measurement errors.
- The Fitness function represents the all actual costs (ex: true calculations of re-handlings and movements and considering the level of relative overhead of each operation, produces a single figure that represents all costs)
- Compare against not just LIFO but against optimized LIFO, which not only minimizes the re-handlings, but also the YC movements.

4.2.1 Optimized LIFO and LIFO Approaches

While LIFO is self-explanatory, when there are multiple stacks in the vessel to retrieve the containers from, the stacks are selected by ascending order of the stack number. Here, we get the same shipment as was used in the GA and retrieve containers from the first stack of the vessel according to the order governed by the LIFO approach. Once all the containers of that stack are retrieved, the adjacent stack is selected for retrieving the containers from. Since we have two block types – Regular and Open Top blocks and Reefer blocks, the container assignment is done separately for two block types, as was the case in GA approach.

Unlike in LIFO approach, in the Optimized LIFO approach, when selecting the stacks, they are selected such that the number of Re handlings and the YC movements are minimal. In this approach too, we get the same shipment as was used in the GA and generate a new container assignment in the CY in a way the re-handlings and YC movements are minimized, instead of randomly assigning containers to locations and allowing to evolve as in the GA. As in the LIFO approach and in GA, the container assignment is done separately for two block types

In this approach, a list of empty stacks are taken from the yard ordered by Block No, Bay No and Row No. Then the topmost regular containers, taken by the descending order of their deadlines, are added to the yard stacks one by one, up until there is one location (top most) left in the stack. Once the yard stack is short of one container to fill to the completion, an Open Top container, taken from the vessel is placed on to fill the stack. When selecting an Open Top container from the vessel, the first Open Top container from the list ordered by, descending, the number of containers below and then descending by the deadline is selected. The above steps are repeated until all containers are assigned. Thus, in addition to minimize the re-handlings at both ends, the aim is to place the containers with higher deadlines towards one end in the CY and the ones with the lowest deadlines towards the other end in order to reduce the movements without sacrificing the re-handlings.

During this process, any potential re-handlings at the vessel, due to having too many Open Top Containers, are counted and any potential re-handlings in the CY is also calculated. Then the total YC movements are calculated and based on the relative costs of each operation, the final cost is calculated as a single figure and takes the multiplicative inverse of this figure as the fitness.

The assignment of Reefer containers is very straight forward and much less complicated due to not having Open Top containers in the top of stacks. Hence, it aims to fill the yard stacks with the containers from the vessel by retrieving in the descending order of their deadlines.

The calculation of the fitness of the Optimized LIFO approach is integrated to our solution in order to be used in the evaluation process.

4.2.2 Test Objectives, Design and Sampling

It is well known that GA based solutions produce poor fitness at initial population and at early generations and the fitness only improves along with the generations, as solution evolves.

Due to this nature of the GA based solutions, when designing our test, we anticipate that the fitness of the initial generations of our GA based solution to be lower than the LIFO and the Optimized LIFO approaches and expect to be improved and to go past the LIFO and Optimized LIFO fitness after certain number of generations.

Test Objectives:

- To find out the number of the Generation at which the fitness of our solution go past the fitness of LIFO and Optimized LIFO solution.
- To find out, at the above number of the Generation, whether the improvement of fitness is still kept on (and if the solution is evolving more), by inspecting the upwards trend of the graph plotted based on the Fitness over Generations.

Test Design and Sampling:

To meet the above objectives, our tests were designed to take fifty random samples of different shipments. In getting the samples, the ratio between Regular, Reefer and Open Top containers was maintained. Different shipments of five sizes were used, each having ten different shipments, totaling to fifty different shipments.

- 2400 Containers
    - Shipment 1 created, LIFO and Optimized LIFO fitness calculated and recorded, and GA fitness values (when improved) and their generations were recorded.
    - [repeated above for total of ten times for the ten shipments of size 2400 containers]
- 2000 Containers

- Shipment 1 created, LIFO and Optimized LIFO fitness calculated and recorded, and GA fitness values (when improved) and their generations were recorded.
- [repeated above for total of ten times for the ten shipments of size 2000 containers]

Similar to the above, test were design for 1600, 1200 and 800 containers. Recording the fitness values, when improved, and the generation number at which each fitness improvement occurred is required for plotting the graphs of fitness over generations as specified in the test objective 2. The fitness over generations are recorded for each and every tests. By this, the generation number at which the GA solution fitness went past the LIFO and Optimized LIFO fitness can be obtained.

All the tests will be shown along with their results in the 4.3 Test Results Section below. We believe that the above is reasonable test coverage since it includes different shipments of different sizes and having included fifty samples.

4.3 Test Results

The results for the tests described above are shown below. The results are shown both in a summarized form as well as in detail form. In the summarized form, the shipment, its LIFO and Optimized LIFO fitness, the number of the generation at which the LIFO and Optimized LIFO fitness were gone past (Achieved at Gen.), and the direction of the trend of the graph are specified. In the detail form, the fitness improvements and their corresponding generations are shown for each test. The Table 4:1 shows the summarized results. The Regular, Reefer and Open Top containers were distributed with the percentages of 91%, 8% and 1% respectively of total containers [30].

Table 4.1: Summarized Results for each Shipment

Shipment	Total Cont. Count	LIFO Fitness	Achieved at Gen. (against LIFO)	Optimized LIFO Fitness	Achieved at Gen. (against Optimized LIFO)	Still Improving
1	2400	0.038523	10	0.039011	25	TRUE
2	2400	0.039502	11	0.039892	21	TRUE
3	2400	0.038433	7	0.039176	30	TRUE
4	2400	0.039491	15	0.039843	28	TRUE
5	2400	0.038418	9	0.039205	24	TRUE
6	2400	0.039728	12	0.039853	18	TRUE
7	2400	0.038960	19	0.039346	27	TRUE
8	2400	0.039215	10	0.039916	24	TRUE
9	2400	0.038382	8	0.038589	19	TRUE
10	2400	0.039283	9	0.039461	23	TRUE
11	2000	0.046280	8	0.046815	18	TRUE
12	2000	0.047211	10	0.047493	14	TRUE
13	2000	0.046436	11	0.048294	25	TRUE
14	2000	0.047598	6	0.048851	20	TRUE
15	2000	0.046482	10	0.046581	15	TRUE
16	2000	0.047155	11	0.047682	24	TRUE
17	2000	0.047178	16	0.047562	28	TRUE
18	2000	0.047590	9	0.047921	18	TRUE
19	2000	0.046428	11	0.046486	14	TRUE
20	2000	0.047739	3	0.048599	18	TRUE
21	1600	0.058791	11	0.059168	19	TRUE
22	1600	0.058979	14	0.059464	31	TRUE
23	1600	0.058701	7	0.059305	18	TRUE
24	1600	0.058803	8	0.059582	25	TRUE
25	1600	0.058931	13	0.059819	25	TRUE
26	1600	0.059191	10	0.059632	23	TRUE
27	1600	0.059240	14	0.059365	21	TRUE
28	1600	0.058775	9	0.059169	19	TRUE
29	1600	0.058765	10	0.059313	20	TRUE
30	1600	0.058915	11	0.059480	20	TRUE
31	1200	0.079546	11	0.080072	19	TRUE

32	1200	0.079612	15	0.080291	27	TRUE
33	1200	0.079762	12	0.080569	21	TRUE
34	1200	0.080111	11	0.080813	24	TRUE
35	1200	0.079860	6	0.081673	22	TRUE
36	1200	0.079823	8	0.079932	12	TRUE
37	1200	0.079582	9	0.079842	14	TRUE
38	1200	0.079644	9	0.080228	17	TRUE
39	1200	0.079912	14	0.081132	23	TRUE
40	1200	0.079674	8	0.080099	16	TRUE
41	800	0.079625	8	0.124738	23	TRUE
42	800	0.123380	11	0.123544	15	TRUE
43	800	0.122566	8	0.124519	23	TRUE
44	800	0.123932	12	0.125686	22	TRUE
45	800	0.122356	9	0.125921	32	TRUE
46	800	0.123543	9	0.124078	14	TRUE
47	800	0.124304	12	0.125589	18	TRUE
48	800	0.123512	10	0.126384	22	TRUE
49	800	0.124605	11	0.127644	28	TRUE
50	800	0.123224	8	0.125288	20	TRUE

The detailed results for few shipments from 1 to 6 are shown in the tables from 4.2 to 4.7. Detailed results for all other the shipments can be found on the Appendix

Table 4.2: Detailed Results for Shipment 1

Gen No	Fitness
1	0.038105
1	0.038130
2	0.038174
3	0.038218
5	0.038291
5	0.038478
9	0.038507
10	0.038555
12	0.038568
13	0.038610
14	0.038623
17	0.038700
18	0.038733
19	0.038736
19	0.038755
20	0.038757
20	0.038809
21	0.038848
21	0.038963
22	0.038982
23	0.038988
25	0.039082
26	0.039156
27	0.039237
27	0.039326
29	0.039391
31	0.039468
34	0.039483
35	0.039500
36	0.039545
39	0.039584

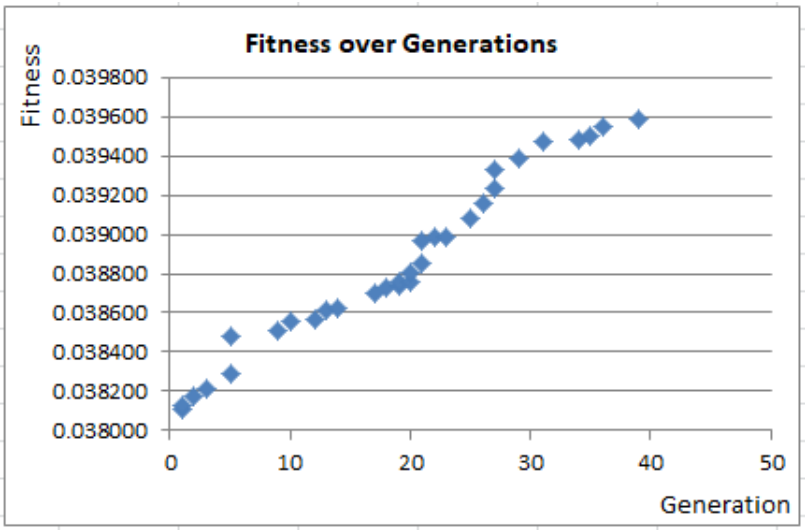


Table 4.3: Detailed Results for Shipment 2

Gen No	Fitness
1	0.038784
2	0.038923
2	0.039081
4	0.039095
4	0.039214
5	0.039344
8	0.039365
10	0.039441
11	0.039476
11	0.039533
15	0.039617
17	0.039691
18	0.039767
20	0.039823
20	0.039843
21	0.039919
21	0.039975
22	0.040006
22	0.040060
24	0.040133
24	0.040134
25	0.040141
26	0.040152
26	0.040164
27	0.040187
28	0.040207
31	0.040225
33	0.040258
35	0.040262
36	0.040283
38	0.040296
39	0.040308
40	0.040312

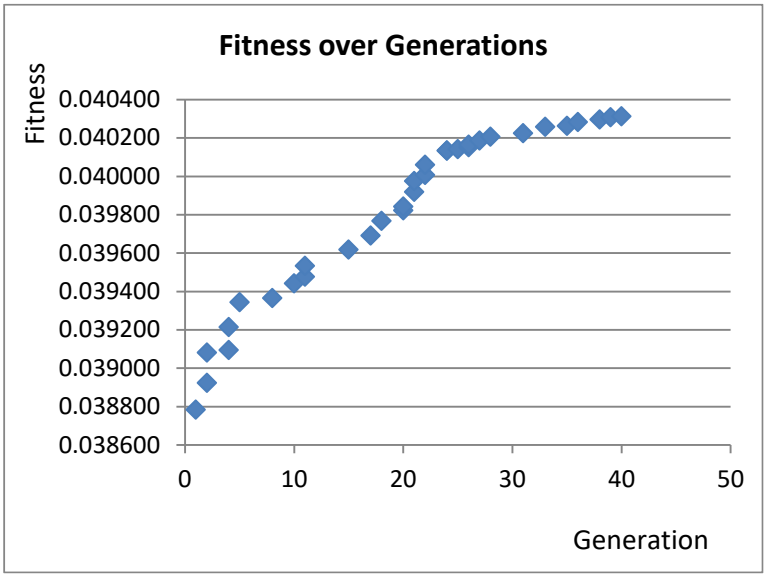


Table 4.4: Detailed Results for Shipment 3

Gen No	Fitness
2	0.038182
2	0.038197
4	0.038204
4	0.038287
6	0.038363
7	0.038515
12	0.038542
13	0.038660
19	0.038662
19	0.038763
20	0.038778
20	0.038815
21	0.038842
22	0.038933
25	0.038941
26	0.039019
27	0.039091
27	0.039135
29	0.039148
30	0.039194
32	0.039238
39	0.039276
39	0.039281

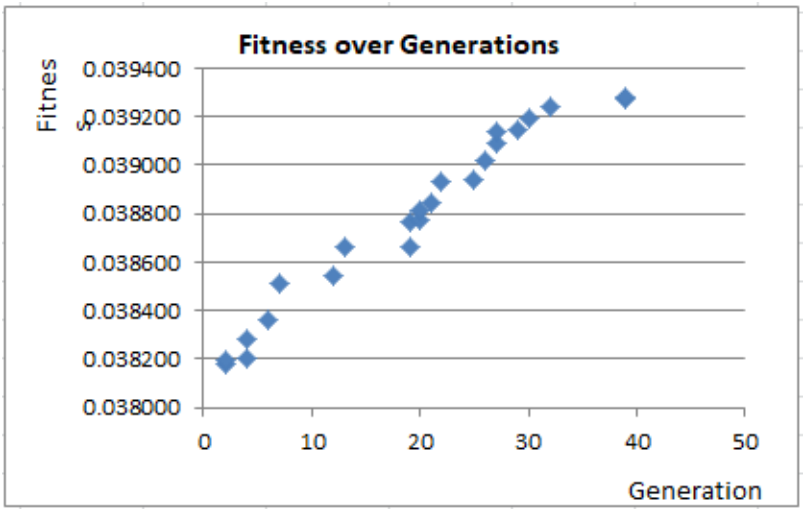


Table 4.5: Detailed Results for Shipment 4

Gen No	Fitness
1	0.038982
1	0.039127
2	0.039185
4	0.039196
5	0.039232
6	0.039356
8	0.039416
11	0.039472
15	0.039553
20	0.039590
20	0.039666
22	0.039716
23	0.039740
24	0.039769
25	0.039800
27	0.039822
28	0.039877
29	0.039930
30	0.039956
30	0.039974
33	0.040006
38	0.040021
39	0.040036
40	0.040042
40	0.040056

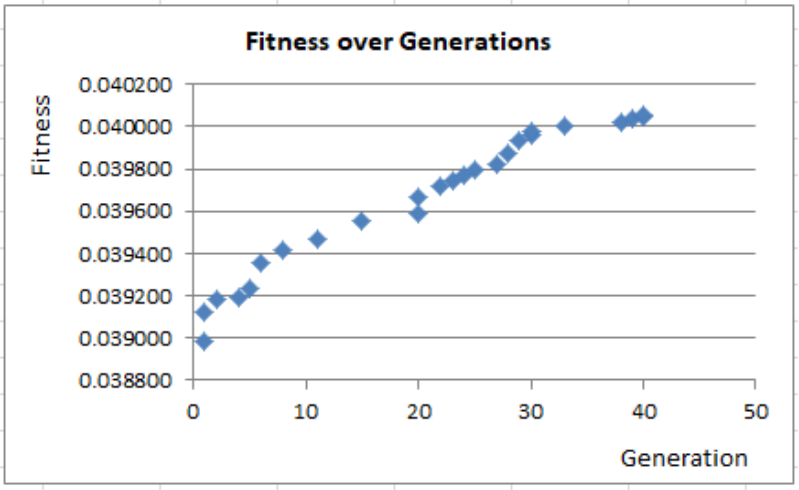


Table 4.6: Detailed Results for Shipment 5

Gen No	Fitness
1	0.038156
3	0.038221
3	0.038223
4	0.038269
6	0.038335
8	0.038353
9	0.038372
9	0.038384
9	0.038429
10	0.038450
12	0.038462
12	0.038511
12	0.038548
14	0.038673
18	0.038688
19	0.038734
20	0.038844
20	0.038960
24	0.039130
24	0.039180
24	0.039270
25	0.039370
30	0.039450
33	0.039590
34	0.039710
40	0.039800

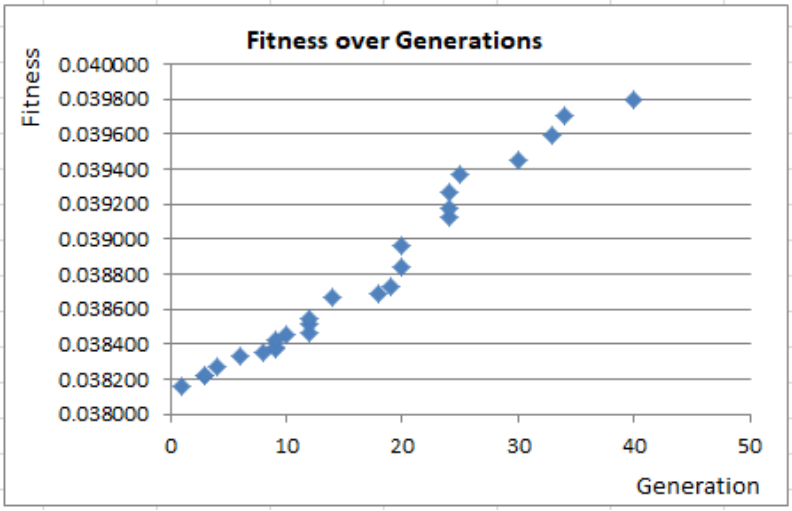
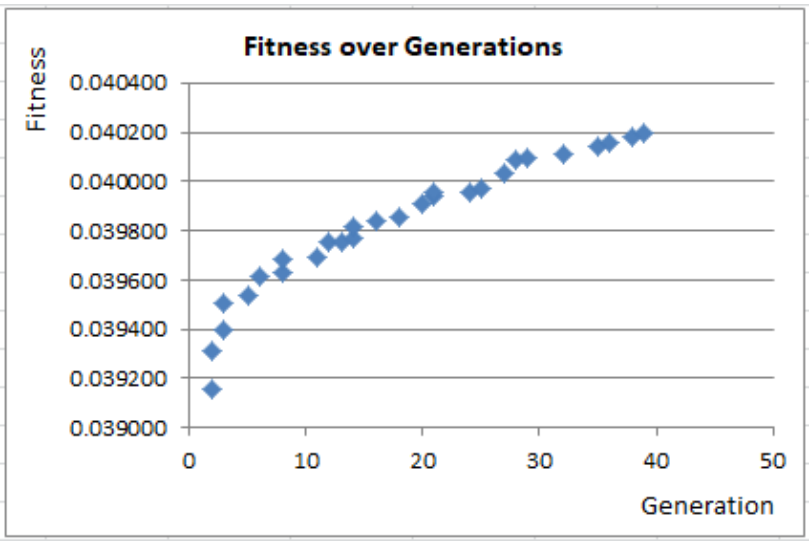


Table 4.7: Detailed Results for Shipment 6

Gen No	Fitness
2	0.039156
2	0.039315
3	0.039401
3	0.039507
5	0.039536
6	0.039617
8	0.039630
8	0.039681
11	0.039695
12	0.039755
13	0.039757
14	0.039771
14	0.039817
16	0.039841
18	0.039857
20	0.039913
21	0.039939
21	0.039955
24	0.039958
25	0.039969
27	0.040030
28	0.040086
29	0.040095
32	0.040114
35	0.040141
36	0.040161
38	0.040178
39	0.040193



For executing the above tests, we used a computer with Intel Celeron 1.10GHz, 1101 MHz 2 Core Processor with x64 architecture and 4GB Physical Ram on a Windows 10 Operating system. The average time taken for the algorithm to complete for each shipment size is specified in the tables 4.8

Table 4.8: Average Time Taken For Each Shipment Size

Shipment Size in number of containers	Average time taken in hours
800	1.2
1200	1.4
1600	1.7
2000	2.2
2400	2.9

By looking at the fitness of LIFO, Optimized LIFO and GA, the following points could be identified as common to all the tests.

- For all the samples, the fitness of the solutions produced by the Optimized LIFO approach was better than the fitness of the solutions produced by the standard LIFO approach.
- At the initial population and at initial generations, the LIFO and the Optimized LIFO approaches produced better results than the GA solution and while at higher generations, the GA solution gives better results.
- After some time, the improvement rate of the fitness over generations are getting reduced and by looking at the trend in the graph we could predict that, at certain moment in the future, improvements will almost halt.

The Figure 4.1 shows the Fitness over Generations for the shipment 31, shown as an example. The logarithmic trend line was added using Excel spreadsheets and the decline of the improvement rate can be observed clearly.

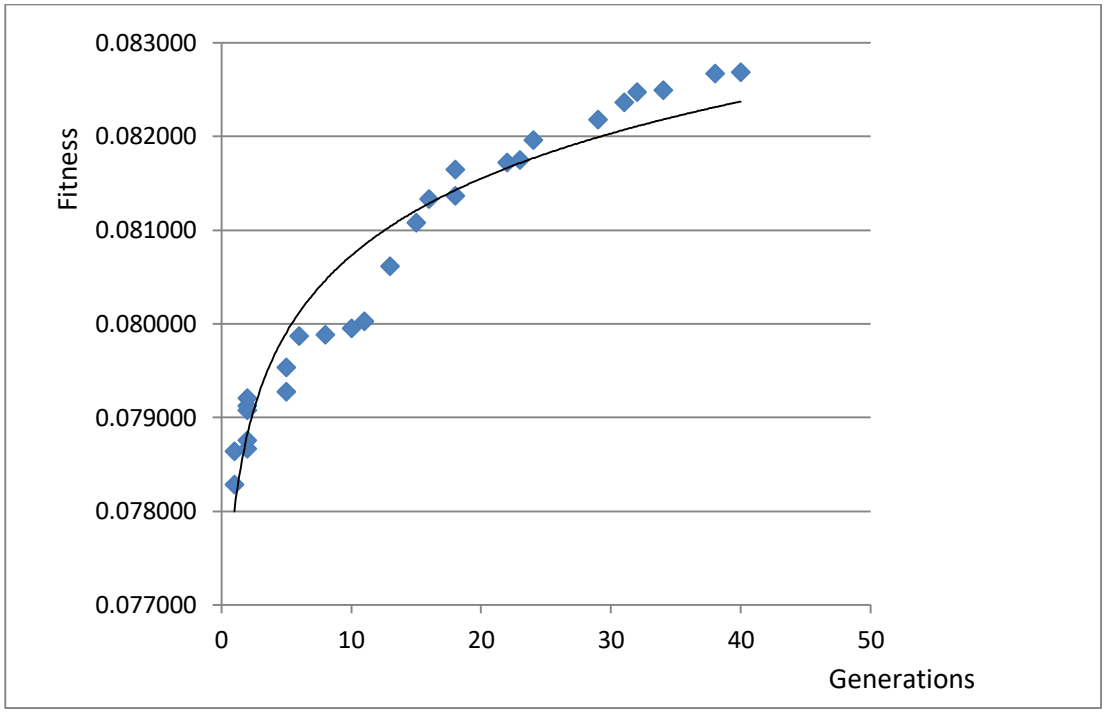


Figure 4.1: Fitness over Generations for Shipment 31

As specified in point one above, for our entire sample of fifty shipments, the fitness of the GA solution could reach and go beyond both the fitness of the LIFO and the Optimized LIFO approach. While there are some variations, the minimum and maximum generations at which the fitness of the GA Solutions went pass the fitness of whichever the higher of LIFO and Optimized LIFO solutions are 12 and 32 respectively. A histogram for the frequency distribution of Generations at which the GA fitness went pass Optimized LIFO fitness (Optimized LIFO since its fitness is always higher than the LIFO fitness) is shown in the figure 4.2.

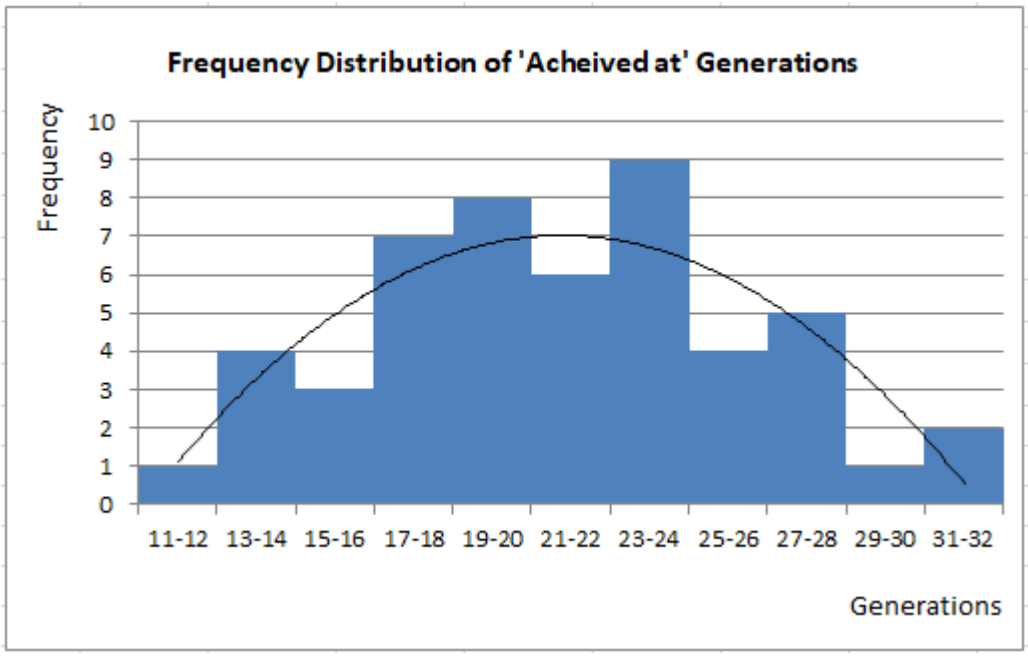


Figure 4.2: Histogram for the Frequency distribution of 'Achieved at' Generations.

The above distribution has the arithmetic mean of 21.32, variance of 22.5 and standard deviation of 4.74 and the trend line plotted in the graph indicates that it looks approaching a normal distribution.

CHAPTER 5

DISCUSSION

As specified in the Introduction chapter, our research problem, in brief, is the occurrence of Re-handlings at CY, Re-handlings at vessel and the unnecessary YC movements in CY. In our solution, we aim to minimize all these three operations such that the total cost is minimal.

From the test results, we had three major findings and out of which, first one being that the Optimized LIFO approach produced better solutions than the LIFO approach and this can be explained as, in the Optimized LIFO, when selecting a stack to retrieve the containers from, it selects the stack having the highest deadline container as the topmost container. This helps reduce both the Re-handlings at CY since the containers with higher deadlines are retrieved first from vessel and hence sent to bottom at CY, as well as the YC movements in CY since the Optimized LIFO directs the containers with higher deadlines towards one end of the CY and the containers with the lower deadlines towards the other end of the CY.

The second finding, where LIFO and Optimized LIFO produced better fitness than GA initial and early generations and later GA fitness got better and went pass the LIFO and Optimized LIFO, can be explained as the initial population of the GA based solution just being a set of random solutions and have not inherited good features of any previous solutions since this is the first generation. But in the LIFO and the Optimized LIFO approach, they are of somewhat analytical approaches and the solutions are obtained by applying some rules when creating the solutions. Hence this LIFO and the Optimized LIFO solutions already are rich in fitness to some extent. In the GA base solution, with the generations pass by, solutions get more and more good features inherited of previous solutions and only the good solutions retained while weaker solutions are removed for future generations. Hence, with time, GA based solution fitness go past the LIFO and the Optimized LIFO solutions fitness.

The third finding, in which the GA improvement rate is declined, is due to the fact that after already inheriting significant portion of the good features through the past generations, the number of good features left to be inherited is getting low and hence the rate gets decreased. And in the future when the solution approaches the optimum solution, the fitness improvements will almost come to a halt.

For all the samples, GA achieved the fitness of the Optimized LIFO, which always produced higher fitness than standard LIFO, within reasonable number of generations and even the recorded maximum number of generations required, which was 32, is reasonable and since the progress of all the samples are still on at the time of these generations, without the need of applying the statistical methods such as Hypothesis Tests to emphasize the statistical significance, we can say that the GA based solution will produce better results for the entire population. Thus, since the progress is not halted and is still on, even if the above statistics are only true for the sample and not true for the entire population due to sampling errors, we can still say that within next few generations the GA solution will produce better results.

While our GA based solution produces better results, there are strengths and major contributions of our work. One being that the work we produce is not only minimizing re-handling just at the CY (or vessel) but in both vessel and CY. All the previous work, to the best of our knowledge, consider re-handlings either in the vessel or in the CY and not in both. That means if they aim to store the containers in the CY such that they can be retrieved, in the order of their deadline, with the minimum number of re-handlings, they assume that, when retrieving containers from the vessel in order to store them in the CY, the containers can be retrieved from the vessel in any order they wish, ignoring the re-handlings that have to be executed when retrieving from the vessel as per the order required by the CY storage plan. For this assumption, the containers have to be unloaded from the vessel into an intermediate storage area and temporarily store in flat (without stacking), before transferring to the CY storage blocks/area. But, in practice, this is very inefficient and unproductive and most of the CTs avoid this approach and, instead, transfer the containers directly from the vessel to the CY's storage blocks/area.

This assumption of intermediate storage is not only the not real situation but also simplifies the solution while introducing additional overhead and costs for the operations, if adopted. This assumption of intermediate storage makes the solution simplified since it eliminates the need of having to consider both the re-handling at

the vessel as well as in the CY. The minimizing the re-handlings both in the vessel and at the CY are not mutually exclusive and one has the effect over the other and vice versa.

In addition to aiming to minimize the number of re-handlings in both vessel and CY, we, on our work, aim to minimize the total YC movements across the Bays. Minimizing across the Bays YC movements is very important since, unlike in the along-the-Bays YC movements, in which only the trolley has to be moved, the entire crane has to be moved in the event of YC movements across the Bays. This is very costly and time consuming due to its heavy structure and, hence, minimizing these movements, in addition to the Re handlings in both the CY and the Vessel, is a major contribution of our work.

Furthermore, the different container types are also considered and the full implementation of a GA with a prototype version of a software with the capability of accommodating dynamic yard configurations as well as changing algorithm parameters.

Hence, our solution with GA implementation is not only capable of producing better results and is effective, but also it produces better results in more realistic situations, which is also the main objective of our work.

CHAPTER 6

CONCLUSION

Summary

As described in the introduction chapter, when a container vessel arrives at a CT, the containers are unloaded from the vessel and stored in the CY as stacks in multi-dimensional blocks. These containers stored in CY are on a temporary basis and each container has an associated deadline. According to the order of these deadlines the containers are retrieved from the CY. This leads to re-handling of containers when the container to be retrieved is not at the top of the stack. When adapting a storage plan with zero or minimum re-handling, it will enforce a retrieval order in the container vessel and will result in re-handling at the container vessel. Thus, the re-handlings in both ends cannot be minimized independently since they are co-related. Furthermore, containers can be scattered in the CY in relation to their deadlines and this will result in additional YC movements across the Bays just to locate the container. Moreover, the cost of single re-handling in container vessel may not be equal to the cost of single re-handling in the CY. Similarly the cost of single YC movement is different from the cost of other two types of re-handlings.

Thus, as mentioned in the research problem, we must optimize all three operations such that the total cost is minimal. This is known as the container SSAP and is an optimization problem that belongs to the NP-Hard class of problems.

Furthermore, potential future changes in the CY environments such as the configurations etc. is yet another problem the CT face and must therefore be solved.

On the other hand, most of the world wide goods trade occurs in the form of the shipping containers and this has been in continuous growth and leaves the CT with the more and more work load to be handled. This, in turn, demands CT to be more efficient, quick in operations and to ensure optimum usage of their limited resources. Since the equipment used in container handling, as described in this work, is limited resources and the re-handling and the additional movements of the YC are unproductive and inefficient operations, and since these re-handlings, both in the vessel and in the CY, and the additional YC movements can occur to significant extent, this problem becomes a very important problem and must therefore be solved.

In addition to solving the container SSAP, it is also important to be able to respond to the changes in the CY such as configurations quickly, due to the increased competition amongst the CTs, especially within close geographical proximity.

Our primary objective was to explore the existing approaches/solutions to the SSAP, adopt and improve them to produce a solution to suit more realistic situations. From this objective, a few specific objectives were identified and one being to adopt a good GA, another being to produce a prototype version of a software tool to implement and evaluate our proposed solution and to be able to accommodate any potential future changes to yard environments, and the last being the capability of our solution and the software tool to handle the special storage requirements of certain container types such as Reefer and Open Top.

As for the solution to our problem, we proposed a GA, along with a prototype version of software tool in which the GA was implemented. While the cost of a single re-handling at the vessel, at the CY and a single movement in YC can each be different to one another, our aim is to minimize the all three operations such that the total cost is minimal. For this, we defined a relationship between the costs of single operation of each type. Once a relationship is built which defines the relative cost of each operation as a percentage, the total cost could be calculated using the number of operations of each type. Thus the multiplicative inverse of this total cost was used as the fitness of our GA used. Also, in our solution, we architectured it such that the potential future changes can easily be responded to.

When evaluating the solution, measurement based evaluation techniques was used and fitness was used as the evaluation criteria. For this, fifty random shipment samples were taken and experimented with the software tool we produced and the improvements of fitness over generations were recorded for the result of each shipment in the sample.

Also the fitness were calculated for the results of each shipment of the sample for both the LIFO and the Optimized LIFO approaches and compared against the fitness of the GA approach. As the results, it was observed that the fitness were better for the

LIFO and the Optimized LIFO results when compared to the initial generations, but GA fitness got better and went past the fitness of both the LIFO and the Optimized LIFO results with the generations to pass by. The fitness improvements over the generations were plotted into graphs individually for each shipment and the trend (direction) of the curves at the points of crossing the fitness of GA and Optimized LIFO approaches were inspected and observed to be upwards, meaning that the improvements are still on, even though at a decreasing rate.

Thus, we could show that the solution we presented was capable of producing better solutions, even in more realistic situations, and state that all our objectives were met and solved our research problem, leaving no questions unanswered.

Contributions

While there exists several previous attempts to solve the Container SSAP in Container Terminals, the work we presented here is different in several ways and is of significant contribution.

To the best of our knowledge, all of the previous work, as pointed out in the discussions chapter, is dependent on the intermediate storage locations. In our work, however, we consider the more practical approach of transferring the containers to the CY storage blocks directly. Thus, in our work, we make it more realistic by avoiding this assumption of intermediate storage and this has been one of the significant contributions of our work.

Another significant contribution of our work is to minimize the re-handling at vessel, re-handling at CY and the number of YC movements across the Bays. We, in our solution, minimize not just one or two but all these three operations such that the total cost of all operations are minimal. Most of the previous works have attempted to minimize only one or two operations. All these three operations use limited resources and experienced human skills and have big impact on the total time, energy and cost

and hence minimizing all these operations is important as they save time, energy, costs and increase the reputation of CT.

Considering and facilitating the special storage space allocations for the special type of containers such as Reefer and Open Top can be specified as yet another contribution of our work.

Thus, our work has significant contribution and numerous parties can benefit from it including the CT, Transporters, Shipping Liners and general society as it help reduce the cost of importation and hence, ultimately the cost of the products imported, even though indirectly.

Moreover, since our solution help reduce operations involved with machinery, it enables saving energy and indirectly helps reduce emissions and is friendly for the environment, especially when the electricity used for these machinery is generated from fossil fuels.

Limitations

Crossover and Mutation are two of the most important genetic operations in GAs. However, depending on the nature of the problem the GA is applied to solve, certain problems may introduce complications in applying Crossover and Mutations operations. Duplication when applying Crossover is one such complication that certain problems do not tolerate with. In our problem, a chromosome represents a container location assignment for a shipment and hence same containers cannot be duplicated in multiple locations. Thus, when applying Crossover, the duplications should be checked for and avoided. This involves significant additional computational processing and other resources. This can be specified as one of the main limitations in our work.

Future Work

As specified earlier in the introduction, as part of our work we presented a prototype version of a software tool with the implementation of the GA. The purpose of this tool is to implement, test and evaluate our GA based solution and hence includes only the core functionality required for the GA based solution. This prototype version of the tool successfully served its intended purpose and it would be of more practical use if this tool can be further extended by including the enterprise integration facilities. On the other hand, having and maintaining multiple stand- alone information systems is difficult, time consuming and hence rare. Since it is difficult and unpractical to change the CT main information system to include the functionality of this tool, the most suitable option is to integrate this tool with the system of CT and other systems. For the potential systems to integrate with, it can be identified as the information systems in Shipping Liners, Container Terminal, Port Authorities/Customs. Moreover, having this as a separate tool with integration capabilities helps increase the scalability by allowing this tool to run in separate node/s due to its demanding computational resources.

Thus, as future work, we suggest to adopt this prototype as a production software tool with the Enterprise Integration facilities such as the use of Enterprise Service Bus.

REFERENCES

[1] I. Ayachi, R. Kammarti, M. Ksouri and P. Borne, "A Genetic algorithm to solve the container storage space allocation problem," presented at the 2010 Int. conf. on Computational Intelligence and Vehicular System (CIVS), France, 2010.

[2] P.Sriphrabu, K. Sethanan, and B. Arnonkijpanich, "A Solution of the Container Stacking Problem by Genetic Algorithm," *IACSIT International Journal of Engineering and Technology*, Vol. 5, No. 1, Feb. 2013

[3] R. Moussi, N.F. Ndiaye, A. Yassine, "A Genetic Algorithm and new Modeling to Solve Container Location Problem In Port," presented at the Dec. 2011 Int. Maritime Transport and Logistics Conf., Laboratory of Applied Mathematics of Le Havre (LMAH), Le Havre University, France 18 - 20 Dec. 2011

[4] D. Steenken, S. Vob, and R.Stahlbock, "Container terminal operation and operations research - A classification and literature review," *OR Spectrum*,2004, doi: 10.1007/s00291-003-0157-z

[5] X. Wang et al., "An Introduction to Harmony Search Optimization Method," *SpringerBriefs in Computational Intelligence, 2015*, doi: 10.1007/978-3-319-08356-8_2

[6] Geem, Kim and Loganathan. "Harmony search." Wikipedia. https://en.wikipedia.org/wiki/Harmony_search [Accessed: 02-Jan-2017].

[7] I. Ayachi, R. Kammarti, M. Ksouri, and P. Borne, "Harmony Search Algorithm For The Container Storage Problem," presented at the 8th Int. Conf. of Modeling and Simulation, Hammamet-Tunisia, May 10-12, 2010.

[8] A. Stehouwer, "Celebration of 20 years AGV's." Port of Rotterdam. https://www.portofrotterdam.com/en/news-and-press-releases/celebration-of-20-years-agv%E2%80%99s [Accessed: 20-Dec-2016].

[9] Container Transportation Systems, "Robotic Seaport Container Terminals" http://www.mech.utsunomiya-u.ac.jp/~hosino/member/hosino/research/research-e.htm [Accessed: 25-Dec-2016].

[10] D. Manjarres, I.Landa-Torres, S.Gil-Lopez, J.DelSer, M.N.Bilbao, S. Salcedo-Sanz, Z.W.Geem, "A survey on applications of the harmony search algorithm," *Engineering Applications of Artificial Intelligence* 26(2013)1818–1831, Bizkaia, Spain, May 2013

[11] A. Said, M. El-Horbaty, "A Simulation Modeling Approach for Optimization of Storage Space Allocation in Container Terminal," *World Academy of Science, Engineering and Technology International Journal of Computer, Electrical, Automation, Control and Information Engineering*, Vol:9, No:1, 2015

[12] E.U. Guldogan, "Simulation-based analysis for hierarchical storage assignment policies in a container terminal," *Transactions of the Society for Modeling and*

Simulation International, 87(6) 523–537, Izmir University of Economics Sakarya Cad, BalcovaIzmir/Turkey, 2010

[13] flexsim. "3D Simulation Modeling and Analysis Software." Flexsim Software. https://www.flexsim.com [Accessed: 28-Dec-2016].

[14] Garfixia software architectures. "Pipe-And-Filter." http://www.dossier-andreas.net/software_architecture/pipe_and_filter.html [Accessed: 08-Aug-2017].

[15] Unctad, "Reviews of Maritime Transport 2015," presented at United Nations Conference on Trade And Development, Oct. 2015

[16] ISO. "ISO 6346:1995." Iso.org https://www.iso.org/obp/ui/#iso:std:iso:6346:ed-3:v1:en [Accessed: 05-Jan-2017].

[17] Wikipedia. "ISO 6346." Wikipedia.org https://en.wikipedia.org/wiki/ISO_6346 [Accessed: 06-Jan-2017].

[18] Singamas. "20' OPEN TOP OFFSHORE CONTAINER (HC)" http://www.singamas.com/en-us/products/detail/100 [Accessed: 06-Jan-2017].

[19] C. Zhang, J. Liu, Y. Wan, K.G. Murty and R.J. Linn, "Storage space allocation in container terminals," *Transportation Research Part B* 37 883–903, Dec. 2003.

[20] M. Zamarripa, P.A. Marchetti et al., "Rolling Horizon Approach for Optimal Production-Distribution Coordination of Industrial Gases Supply-chains," presented at the Center for Advanced Process Decision-making Enterprise Wide Optimization Meeting, Sep. 18-19, 2014.

[21] S. Sethi and G. Sorger, "A Theory of Rolling Horizon Decision Making," Annals of Operations Research 29 (1991) 387-416

[22] MathWorks. "Genetic Algorithm" https://www.mathworks.com/discovery/genetic-algorithm.html [Accessed: 03-Jan-2017].

[23] M. Bazzazi, N.Safaei, N. Javadian, "A genetic algorithm to solve the storage space allocation problem in a container terminal," *Computers & Industrial Engineering* 56, 2009

[24] Enterprise Integration Patterns. "Pipes and Filters" http://www.enterpriseintegrationpatterns.com/patterns/messaging/PipesAndFilters.html [Accessed: 08-Aug-2017].

[25] A. Abdeslam, F. Bouanani, H.Benazza, "Four Parallel Decoding Schemas of Product BlockCodes," *Society for Science and Education,* Vol. 2, United Kingdom, Jun. 2014.

[26] Wikipedia. “Multitier architecture.” Wikipedia.org https://en.wikipedia.org/wiki/Multitier_architecture [Accessed: 14-Mar-2020].

[27] J.L.Arjona, “The Parallel Pipes and Filters Pattern,” presented at the 2005 EuroPLoP Int. conf. on Architectural Pattern for Parallel Programming, Kloster Irsee in Bavaria, Germany, 2005.

[28] Steve Easterbrook. “Software Architectures.” University of Toronto http://www.cs.toronto.edu/~sme/CSC444F/slides/L19-SoftwareArchitectures.pdf [Accessed: 15-Mar-2020].

[29] Syed Hasan. “Pipe and Filter Architecture.” medium.com https://medium.com/@syedhasan010/pipe-and-filter-architecture-bd7babdb908 [Accessed: 15-Mar-2020].

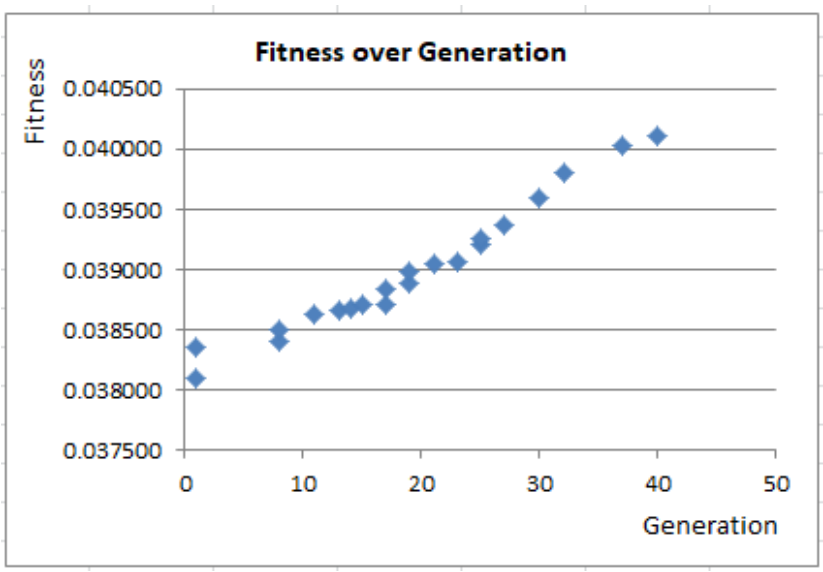
[30] H.R. Morley. “Growing demand to keep ocean reefer rates on the rise.” joc.com https://www.joc.com/international-logistics/ocean-reefer-rate-increases-forecast-keep-outpacing-dry-prices_20180927.html [Accessed: 15-Mar-2020].

[31] Wikipedia. “Genetic algorithm.” Wikipedia.org https://en.wikipedia.org/wiki/Genetic_algorithm [Accessed: 22-May-2020].

APPENDIX

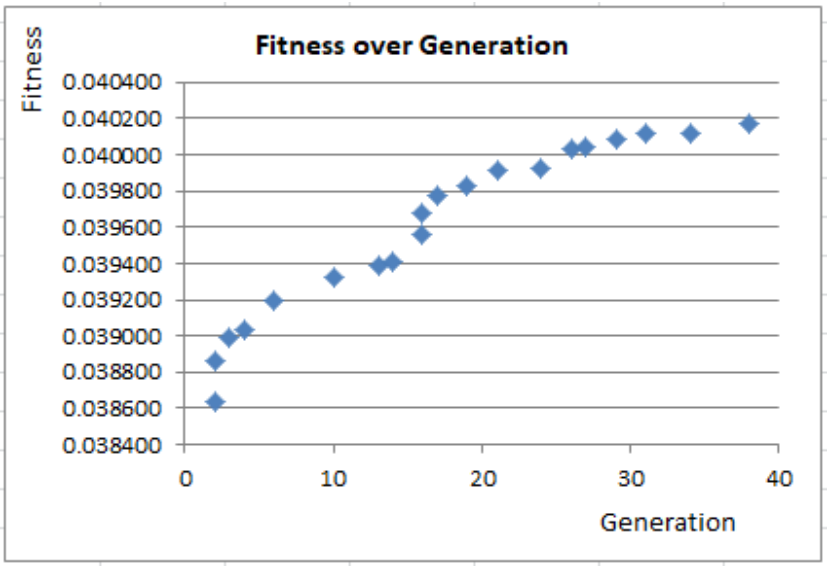
Detailed Results for Shipment 7

Gen No	Fitness
1	0.038092
1	0.038355
8	0.038400
8	0.038505
11	0.038626
13	0.038656
14	0.038671
15	0.038706
17	0.038713
17	0.038835
19	0.038892
19	0.038978
21	0.039041
23	0.039061
25	0.039207
25	0.039262
27	0.039376
30	0.039590
32	0.039808
37	0.040029
40	0.040104



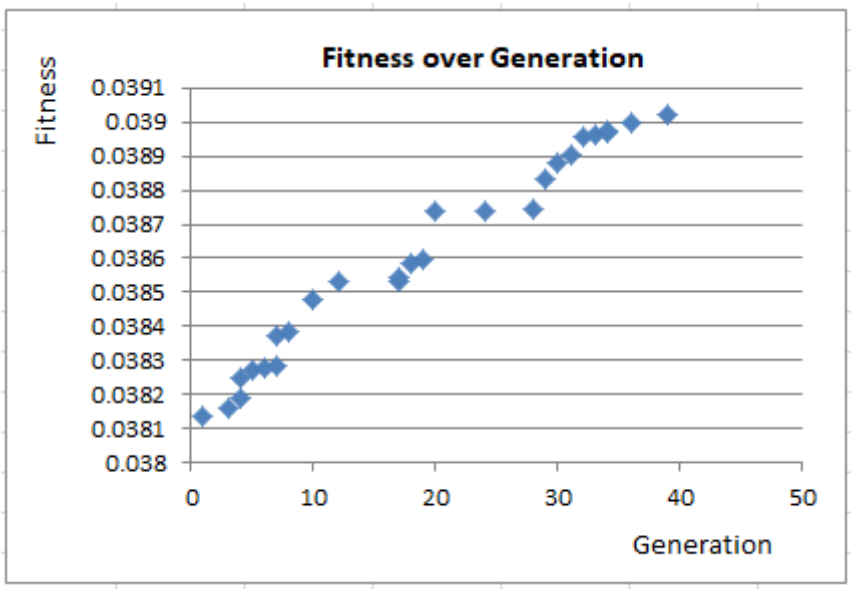
Detailed Results for Shipment 8

Gen No	Fitness
2	0.038641
2	0.038863
3	0.038988
4	0.039032
6	0.039196
10	0.039327
13	0.039384
14	0.039413
16	0.039559
16	0.039676
17	0.039779
19	0.039832
21	0.039913
24	0.039926
26	0.040035
27	0.040044
29	0.040087
31	0.040118
34	0.040119
38	0.040170



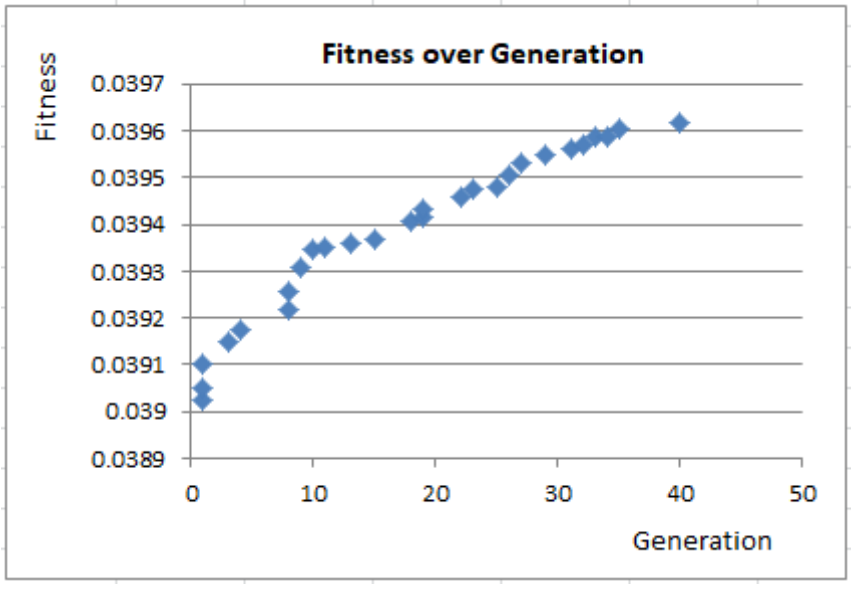
Detailed Results for Shipment 9

Gen No	Fitness
1	0.038139
3	0.03816
4	0.038188
4	0.038247
5	0.038273
6	0.038276
7	0.038284
7	0.038371
8	0.038386
10	0.038476
12	0.03853
17	0.038534
17	0.038542
18	0.038585
19	0.038599
20	0.038737
24	0.038739
28	0.038745
29	0.03883
30	0.038883
31	0.038905
32	0.038956
33	0.038965
34	0.038966
34	0.038972
36	0.038996
39	0.039022



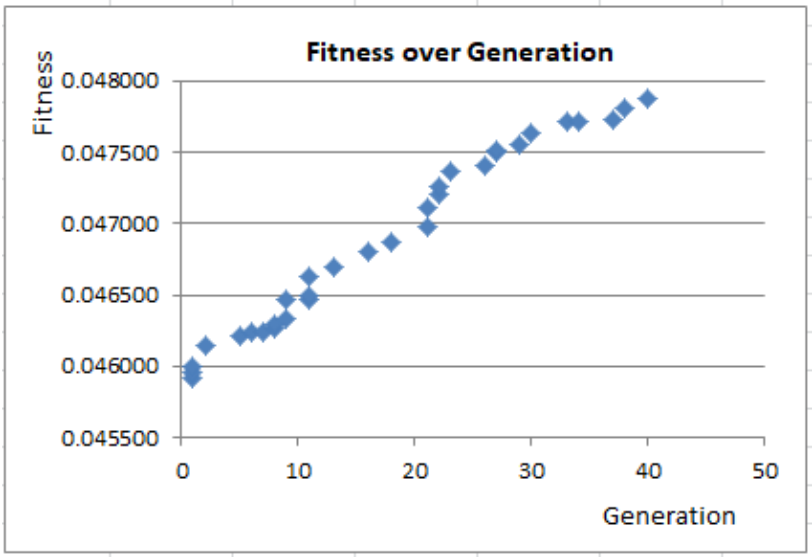
Detailed Results for Shipment 10

Gen No	Fitness
1	0.039024
1	0.039052
1	0.039104
3	0.03915
4	0.039176
8	0.039218
8	0.039258
9	0.039309
10	0.039347
11	0.039349
13	0.03936
15	0.039369
18	0.039406
19	0.039415
19	0.039434
22	0.039458
23	0.039474
25	0.03948
26	0.039506
27	0.039531
29	0.039548
31	0.039562
32	0.039569
33	0.039586
34	0.039589
35	0.039606
40	0.039619



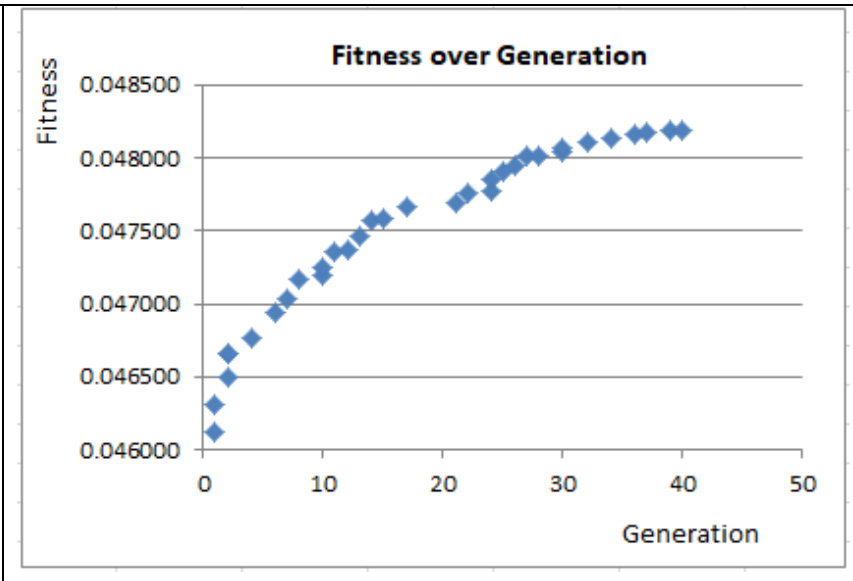
Detailed Results for Shipment 11

Gen No	Fitness
1	0.045921
1	0.045956
1	0.045995
2	0.046151
5	0.046217
6	0.046234
7	0.046245
8	0.046262
8	0.046289
9	0.046330
9	0.046466
11	0.046468
11	0.046492
11	0.046624
13	0.046700
16	0.046803
18	0.046868
21	0.046974
21	0.047112
22	0.047209
22	0.047261
23	0.047365
26	0.047400
27	0.047496
27	0.047518
29	0.047554
30	0.047629
33	0.047710
34	0.047716
37	0.047728
38	0.047806
40	0.047874



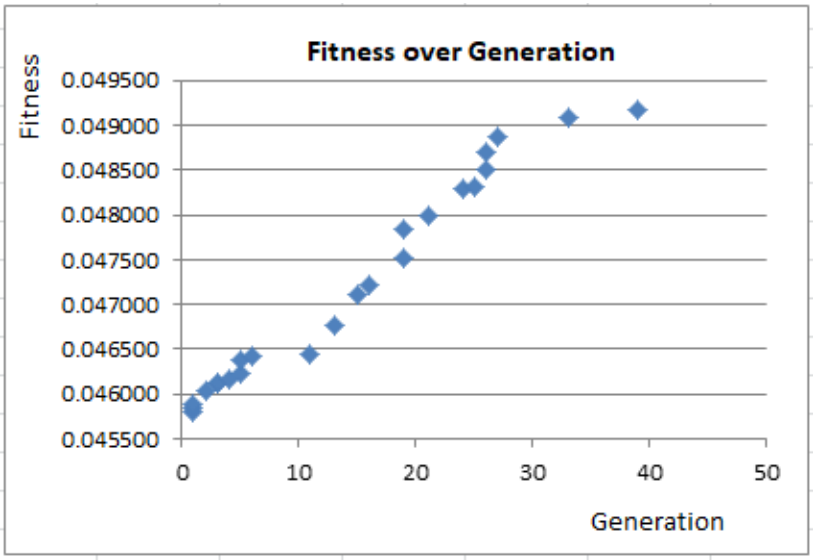
Detailed Results for Shipment 12

Gen No	Fitness
1	0.046123
1	0.046305
2	0.046495
2	0.046653
2	0.046665
4	0.046771
6	0.046936
7	0.047034
8	0.047173
10	0.047195
10	0.047250
11	0.047361
12	0.047363
13	0.047464
14	0.047572
15	0.047590
17	0.047663
21	0.047693
22	0.047753
24	0.047773
24	0.047856
25	0.047900
26	0.047945
27	0.048013
28	0.048015
30	0.048038
30	0.048063
32	0.048102
34	0.048134
36	0.048156
37	0.048168
39	0.048182
40	0.048191



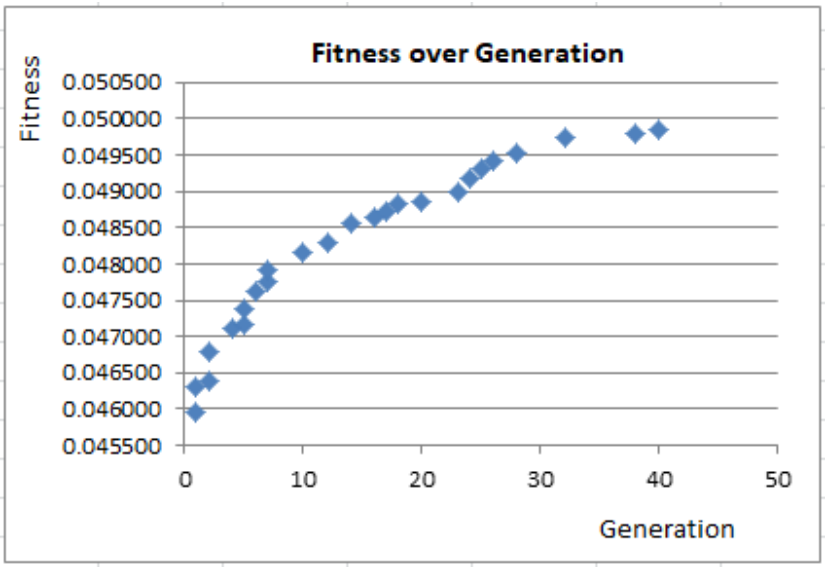
Detailed Results for Shipment 13

Gen No	Fitness
1	0.045812
1	0.045835
1	0.045886
2	0.046047
3	0.046121
3	0.046123
4	0.046160
5	0.046234
5	0.046378
6	0.046432
11	0.046443
13	0.046765
15	0.047108
16	0.047211
19	0.047518
19	0.047847
21	0.048001
24	0.048288
25	0.048322
26	0.048498
26	0.048704
27	0.048865
33	0.049082
39	0.049164



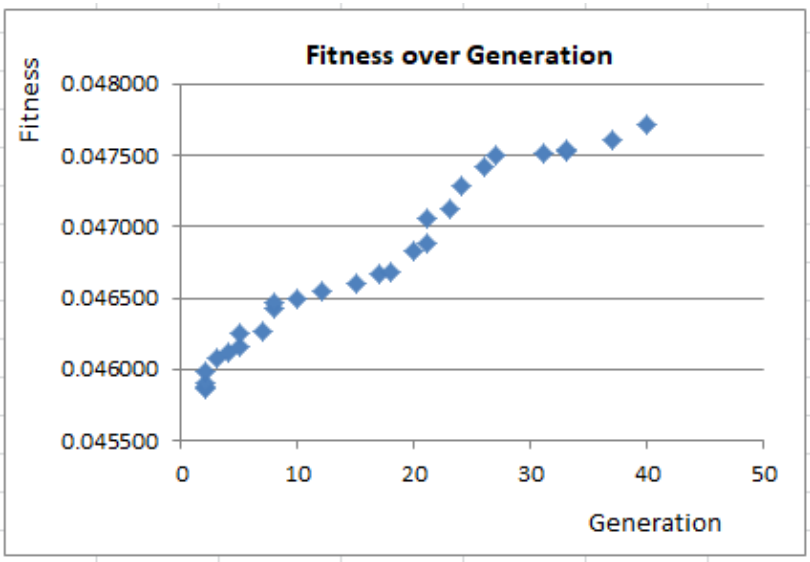
Detailed Results for Shipment 14

Gen No	Fitness
1	0.045946
1	0.046310
2	0.046397
2	0.046792
4	0.047113
5	0.047177
5	0.047382
6	0.047624
7	0.047745
7	0.047921
10	0.048168
12	0.048290
14	0.048550
16	0.048643
17	0.048711
18	0.048842
20	0.048866
23	0.048990
24	0.049177
25	0.049300
26	0.049421
28	0.049520
32	0.049741
38	0.049809
40	0.049857



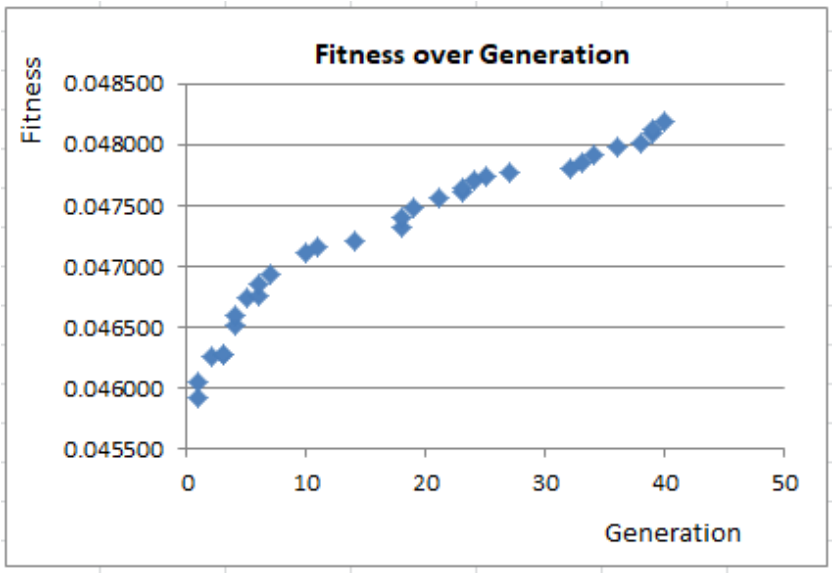
Detailed Results for Shipment 15

Gen No	Fitness
2	0.045860
2	0.045875
2	0.045901
2	0.045980
3	0.046072
4	0.046123
5	0.046152
5	0.046250
7	0.046263
8	0.046427
8	0.046473
10	0.046488
12	0.046542
15	0.046607
17	0.046672
18	0.046676
20	0.046832
21	0.046880
21	0.047060
23	0.047130
24	0.047290
26	0.047420
27	0.047500
31	0.047516
33	0.047530
33	0.047540
37	0.047610
40	0.047720



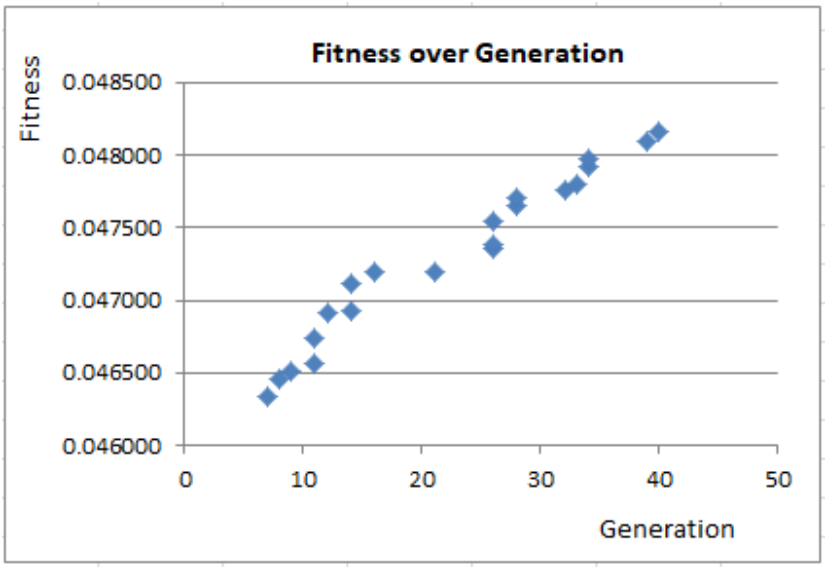
Detailed Results for Shipment 16

Gen No	Fitness
1	0.045925
1	0.046048
2	0.046254
3	0.046270
3	0.046281
4	0.046510
4	0.046599
5	0.046733
6	0.046763
6	0.046855
7	0.046938
10	0.047119
11	0.047158
14	0.047206
18	0.047323
18	0.047406
19	0.047475
21	0.047559
23	0.047615
23	0.047638
24	0.047707
25	0.047747
27	0.047777
32	0.047806
33	0.047854
34	0.047913
36	0.047983
38	0.048021
39	0.048097
39	0.048118
40	0.048186



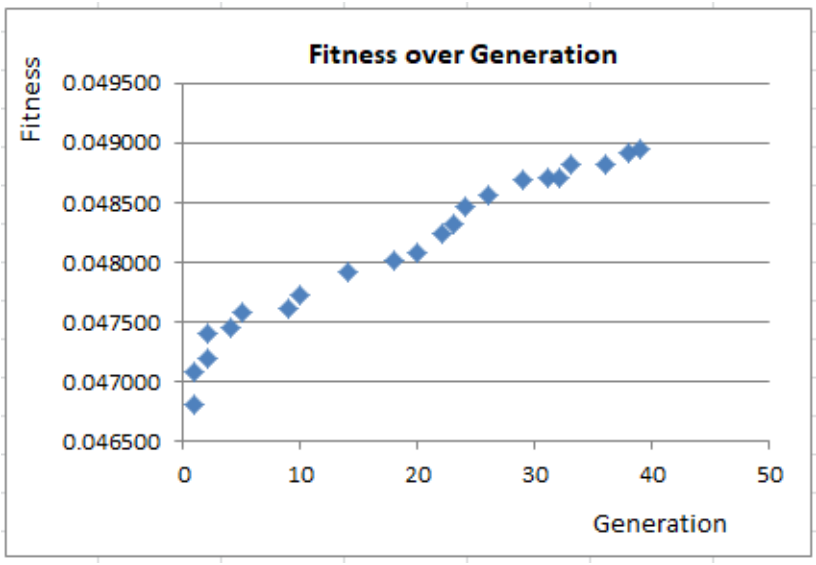
Detailed Results for Shipment 17

Gen No	Fitness
7	0.046335
8	0.046452
9	0.046514
11	0.046565
11	0.046740
12	0.046910
14	0.046920
14	0.047120
16	0.047190
21	0.047200
26	0.047350
26	0.047380
26	0.047540
28	0.047650
28	0.047700
32	0.047752
33	0.047800
34	0.047920
34	0.047980
39	0.048090
40	0.048160



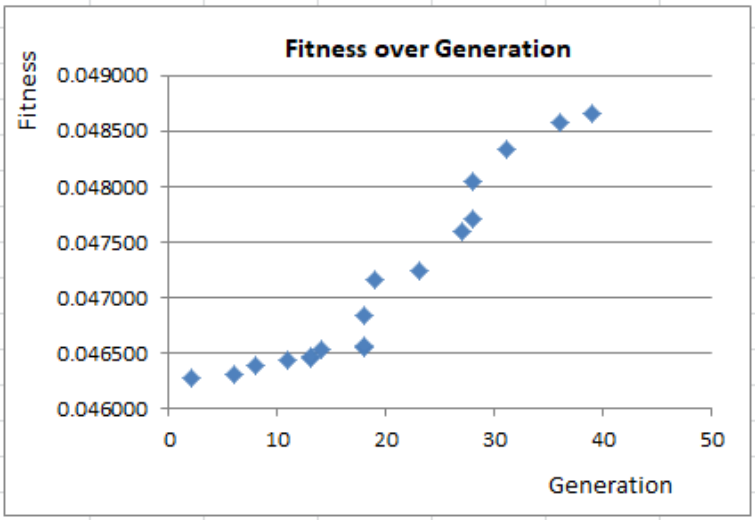
Detailed Results for Shipment 18

Gen No	Fitness
1	0.046813
1	0.047079
2	0.047189
2	0.047406
4	0.047449
5	0.047577
9	0.047617
10	0.047722
14	0.047919
18	0.048022
20	0.048082
22	0.048240
23	0.048325
24	0.048471
26	0.048560
29	0.048691
31	0.048699
32	0.048703
33	0.048819
36	0.048826
38	0.048924
39	0.048953



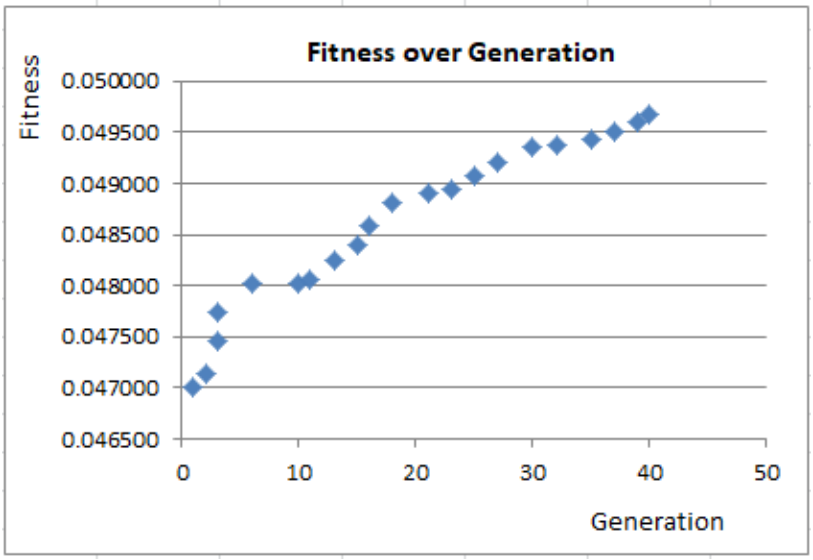
Detailed Results for Shipment 19

Gen No	Fitness
2	0.046273
6	0.046307
8	0.046381
11	0.046432
13	0.046445
13	0.046463
14	0.046534
18	0.046547
18	0.046571
18	0.046844
19	0.047160
23	0.047240
27	0.047600
28	0.047710
28	0.048050
31	0.048340
36	0.048570
39	0.048650



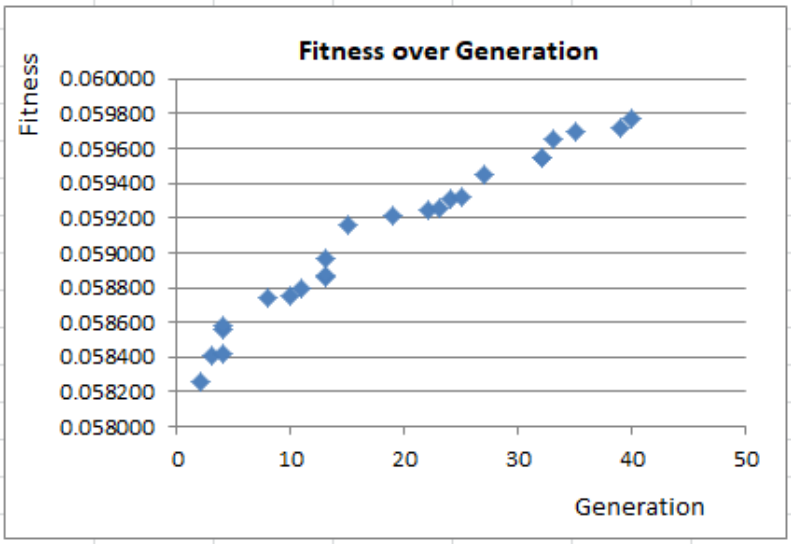
Detailed Results for Shipment 20

Gen No	Fitness
1	0.047005
2	0.047139
3	0.047459
3	0.047743
6	0.048019
10	0.048030
11	0.048059
13	0.048251
15	0.048405
16	0.048588
18	0.048809
21	0.048906
23	0.048939
25	0.049070
27	0.049198
30	0.049359
32	0.049369
35	0.049427
37	0.049512
39	0.049598
40	0.049668



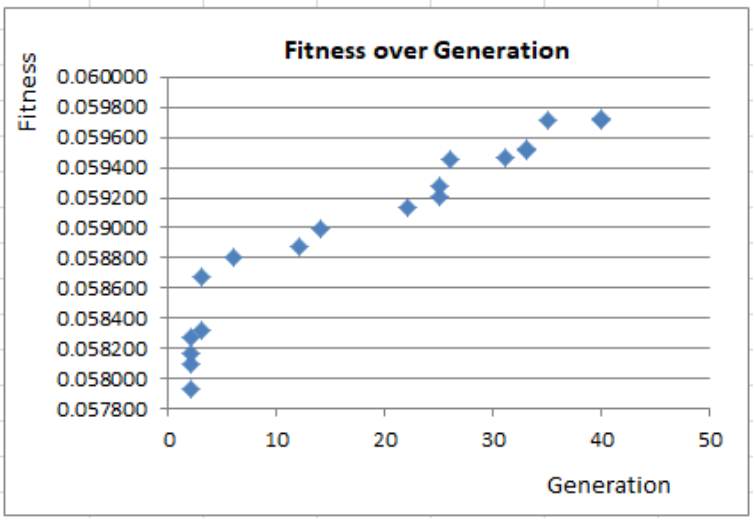
Detailed Results for Shipment 21

Gen No	Fitness
2	0.058261
3	0.058411
4	0.058418
4	0.058557
4	0.058584
8	0.058744
10	0.058753
11	0.058799
13	0.058856
13	0.058875
13	0.058966
15	0.059156
19	0.059217
22	0.059245
23	0.059258
24	0.059312
25	0.059317
27	0.059451
32	0.059547
32	0.059549
33	0.059658
35	0.059702
39	0.059715
40	0.059777



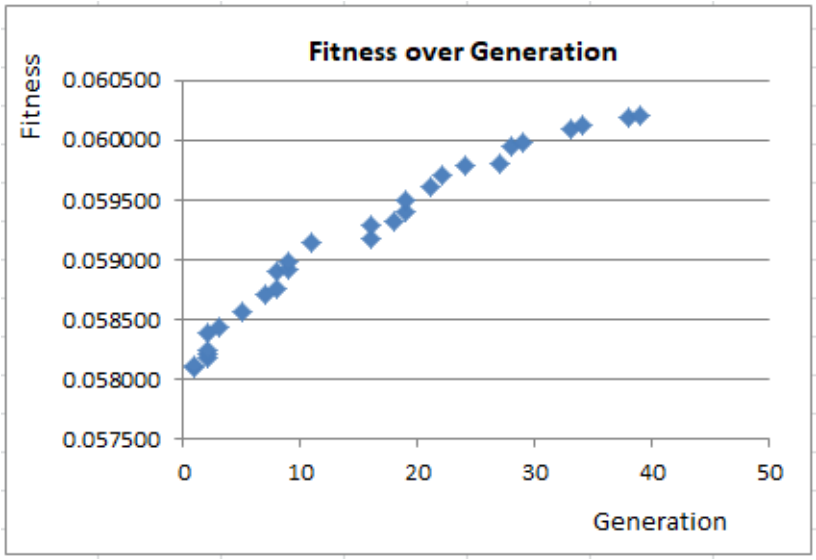
Detailed Results for Shipment 22

Gen No	Fitness
2	0.057934
2	0.058099
2	0.058168
2	0.058268
3	0.058323
3	0.058672
6	0.058801
12	0.058877
14	0.058994
22	0.059137
25	0.059205
25	0.059275
26	0.059458
31	0.059471
33	0.059510
33	0.059520
35	0.059712
40	0.059719
40	0.059730



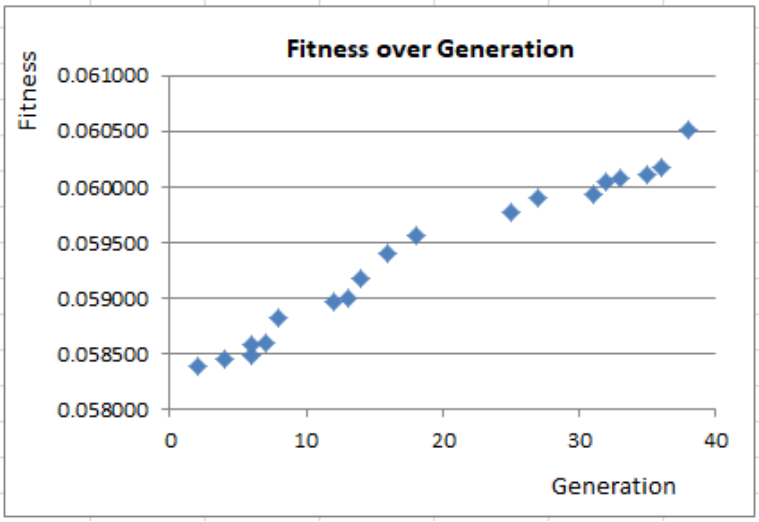
Detailed Results for Shipment 23

Gen No	Fitness
1	0.058102
1	0.058113
2	0.058177
2	0.058216
2	0.058234
2	0.058381
3	0.058427
5	0.058563
7	0.058708
8	0.058751
8	0.058896
9	0.058922
9	0.058981
11	0.059144
16	0.059179
16	0.059293
18	0.059319
19	0.059405
19	0.059504
21	0.059616
22	0.059714
24	0.059795
27	0.059810
28	0.059949
29	0.059982
33	0.060097
34	0.060121
38	0.060190
39	0.060199



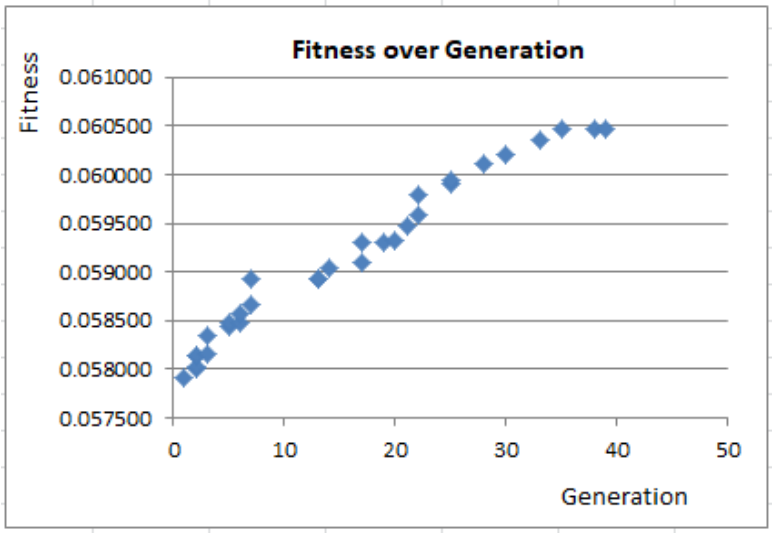
Detailed Results for Shipment 24

Gen No	Fitness
2	0.058382
4	0.058454
6	0.058478
6	0.058582
7	0.058596
8	0.058815
12	0.058964
13	0.058992
14	0.059173
16	0.059407
18	0.059565
25	0.059769
27	0.059900
31	0.059929
32	0.060038
33	0.060074
35	0.060103
36	0.060174
38	0.060514



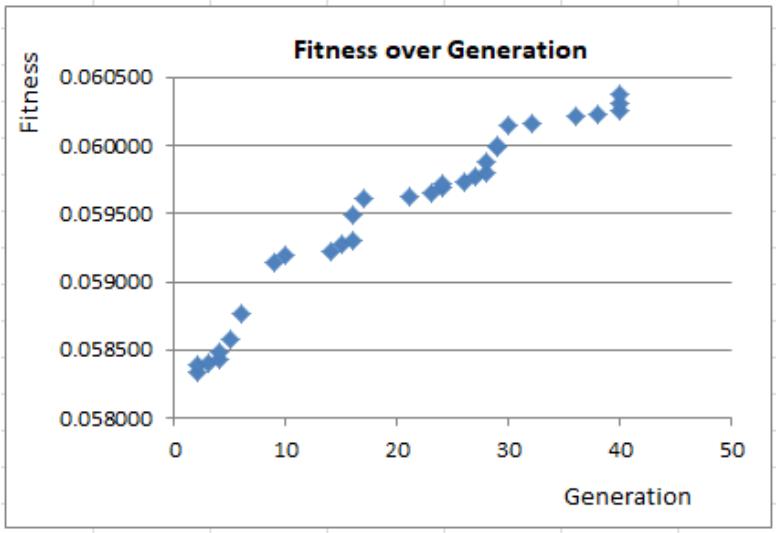
Detailed Results for Shipment 25

Gen No	Fitness
1	0.057909
2	0.058000
2	0.058028
2	0.058140
2	0.058148
3	0.058151
3	0.058353
5	0.058447
5	0.058476
6	0.058486
6	0.058579
7	0.058658
7	0.058928
13	0.058929
13	0.058934
14	0.059037
17	0.059096
17	0.059301
19	0.059311
20	0.059326
21	0.059477
22	0.059595
22	0.059801
25	0.059911
25	0.059943
28	0.060111
30	0.060210
33	0.060365
35	0.060461
38	0.060469
39	0.060475



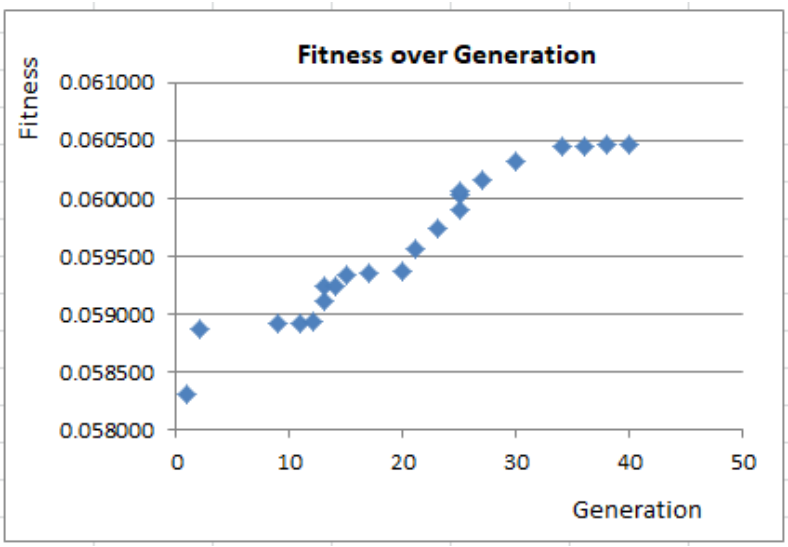
Detailed Results for Shipment 26

Gen No	Fitness
2	0.058340
2	0.058391
3	0.058398
4	0.058432
4	0.058490
5	0.058579
6	0.058768
9	0.059137
10	0.059194
14	0.059228
15	0.059277
16	0.059305
16	0.059497
17	0.059616
21	0.059623
23	0.059657
24	0.059698
24	0.059719
26	0.059730
27	0.059778
28	0.059793
28	0.059882
29	0.059983
29	0.059997
30	0.060143
32	0.060165
36	0.060214
38	0.060228
40	0.060255
40	0.060306
40	0.060370



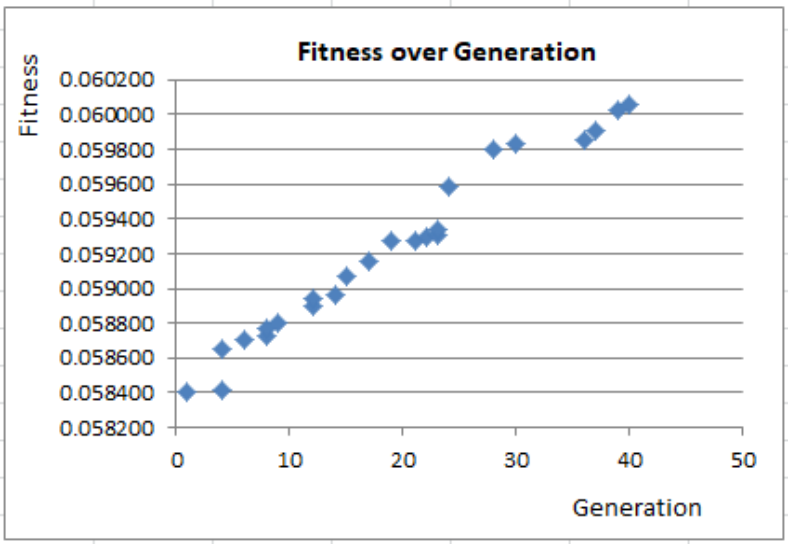
Detailed Results for Shipment 27

Gen No	Fitness
1	0.058301
2	0.058867
9	0.058917
11	0.058922
12	0.058938
13	0.059119
13	0.059238
14	0.059245
15	0.059345
17	0.059347
20	0.059363
21	0.059557
23	0.059748
25	0.059906
25	0.060024
25	0.060056
27	0.060154
30	0.060320
34	0.060445
36	0.060454
38	0.060459
40	0.060462



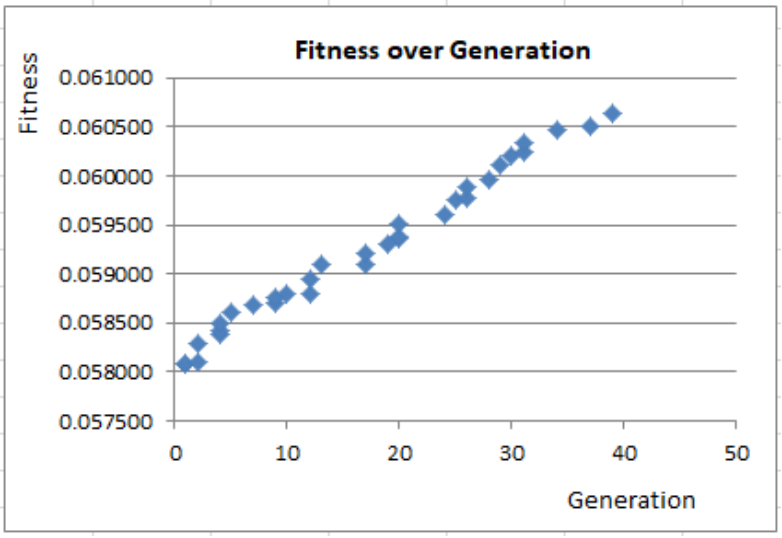
Detailed Results for Shipment 28

Gen No	Fitness
1	0.058408
4	0.058420
4	0.058648
6	0.058710
8	0.058723
8	0.058770
9	0.058799
12	0.058895
12	0.058941
14	0.058964
15	0.059067
17	0.059159
19	0.059270
21	0.059279
22	0.059293
23	0.059301
23	0.059340
24	0.059584
28	0.059800
30	0.059832
36	0.059852
37	0.059905
39	0.060029
40	0.060058



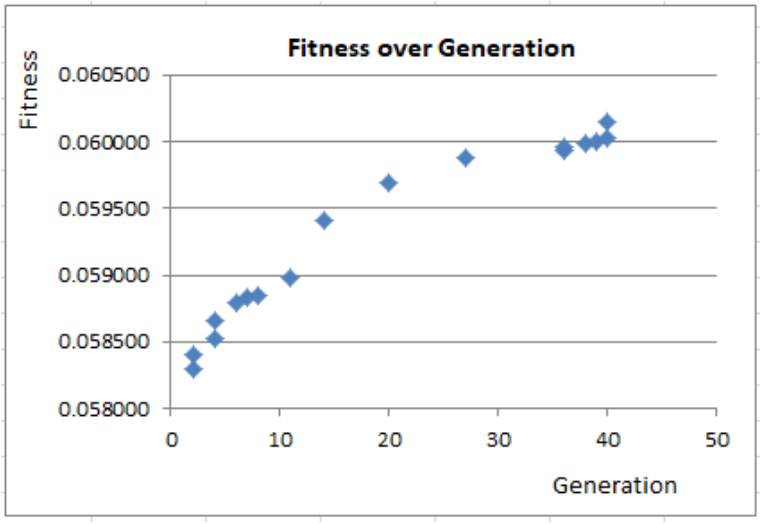
Detailed Results for Shipment 29

Gen No	Fitness
1	0.058082
1	0.058087
2	0.058106
2	0.058292
4	0.058382
4	0.058428
4	0.058493
5	0.058603
7	0.058694
9	0.058708
9	0.058751
10	0.058791
12	0.058796
12	0.058947
13	0.059093
17	0.059103
17	0.059214
19	0.059303
20	0.059368
20	0.059381
20	0.059517
24	0.059602
25	0.059759
26	0.059779
26	0.059884
28	0.059956
29	0.060116
30	0.060212
31	0.060252
31	0.060331
34	0.060462
37	0.060505
39	0.060632



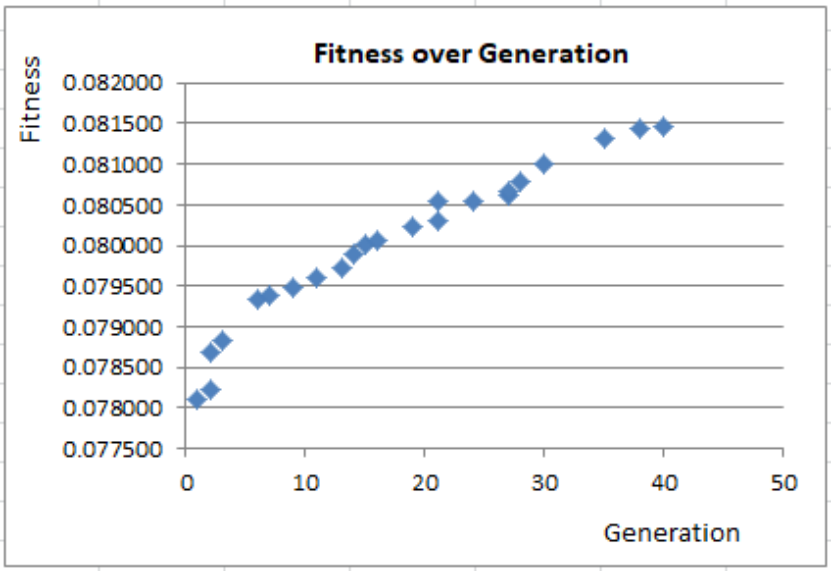
Detailed Results for Shipment 30

Gen No	Fitness
2	0.058295
2	0.058406
4	0.058522
4	0.058661
6	0.058791
7	0.058836
8	0.058846
11	0.058974
14	0.059411
20	0.059685
27	0.059875
36	0.059938
36	0.059957
38	0.059992
39	0.060002
40	0.060031
40	0.060149



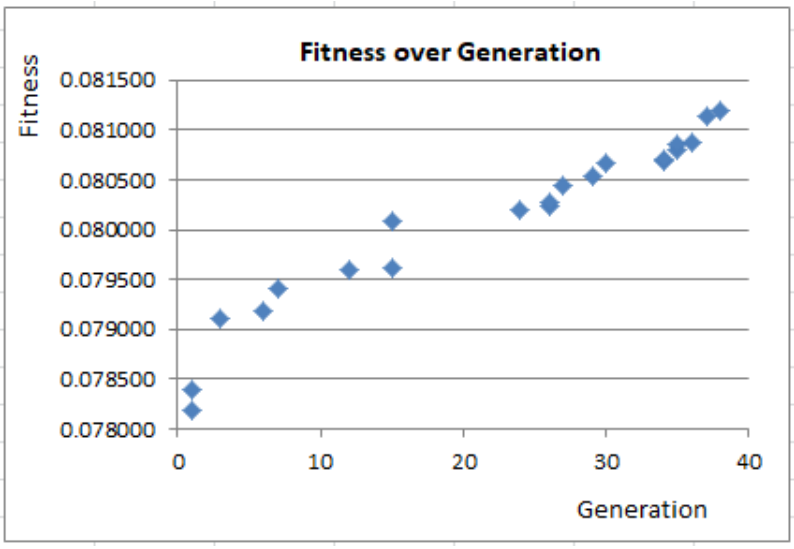
Detailed Results for Shipment 31

Gen No	Fitness
1	0.078104
2	0.078220
2	0.078687
3	0.078824
6	0.079340
7	0.079375
9	0.079494
11	0.079599
13	0.079729
14	0.079898
15	0.080022
16	0.080064
19	0.080231
21	0.080292
21	0.080538
24	0.080548
27	0.080613
27	0.080661
28	0.080782
30	0.081014
35	0.081311
38	0.081427
40	0.081460



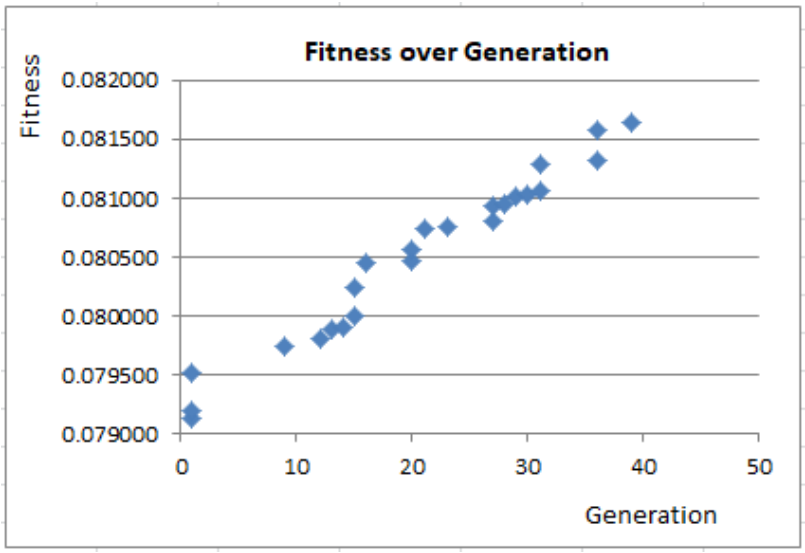
Detailed Results for Shipment 32

Gen No	Fitness
1	0.078189
1	0.078391
3	0.079105
6	0.079183
7	0.079406
12	0.079602
15	0.079621
15	0.080093
24	0.080192
26	0.080231
26	0.080266
27	0.080441
29	0.080541
30	0.080678
34	0.080694
34	0.080700
35	0.080798
35	0.080854
36	0.080867
37	0.081139
38	0.081202



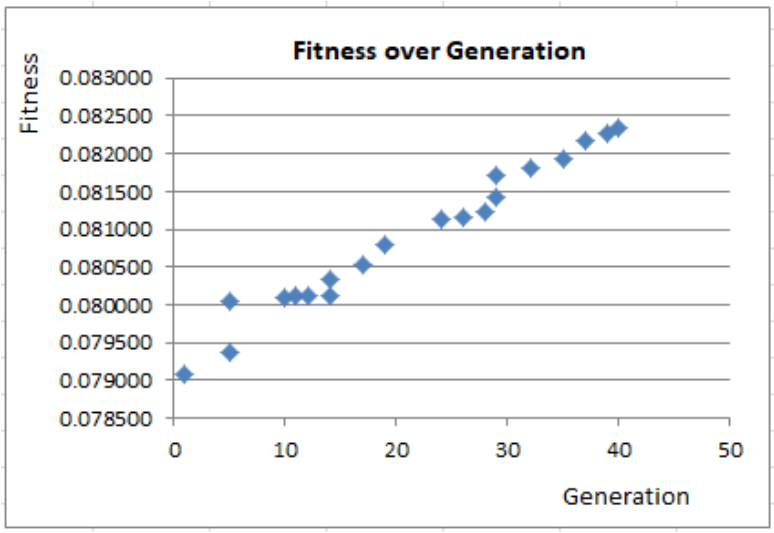
Detailed Results for Shipment 33

Gen No	Fitness
1	0.079130
1	0.079199
1	0.079520
9	0.079738
12	0.079805
13	0.079882
14	0.079895
15	0.080000
15	0.080247
16	0.080447
20	0.080473
20	0.080561
21	0.080740
23	0.080756
27	0.080805
27	0.080926
28	0.080952
29	0.081011
30	0.081027
31	0.081060
31	0.081281
36	0.081317
36	0.081583
39	0.081649



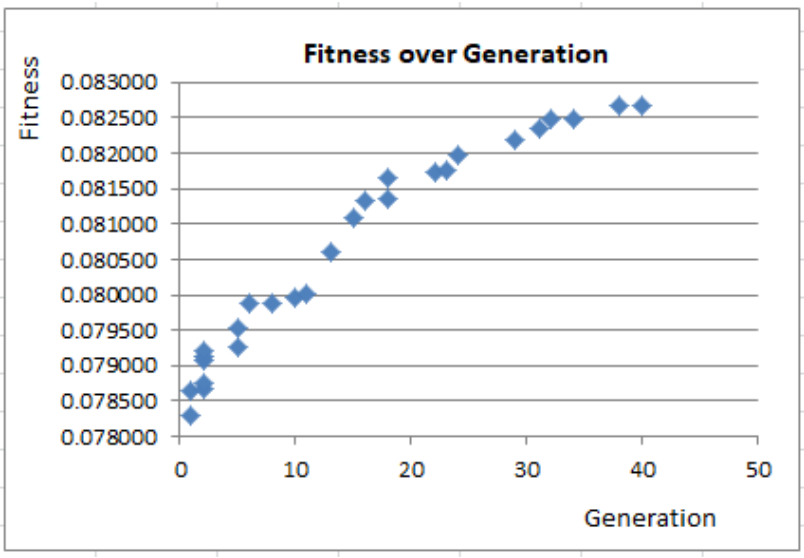
Detailed Results for Shipment 34

Gen No	Fitness
1	0.079076
5	0.079365
5	0.080051
10	0.080103
11	0.080122
12	0.080125
14	0.080131
14	0.080328
17	0.080532
19	0.080802
24	0.081126
26	0.081149
28	0.081222
29	0.081427
29	0.081709
32	0.081813
35	0.081927
37	0.082179
39	0.082274
40	0.082345



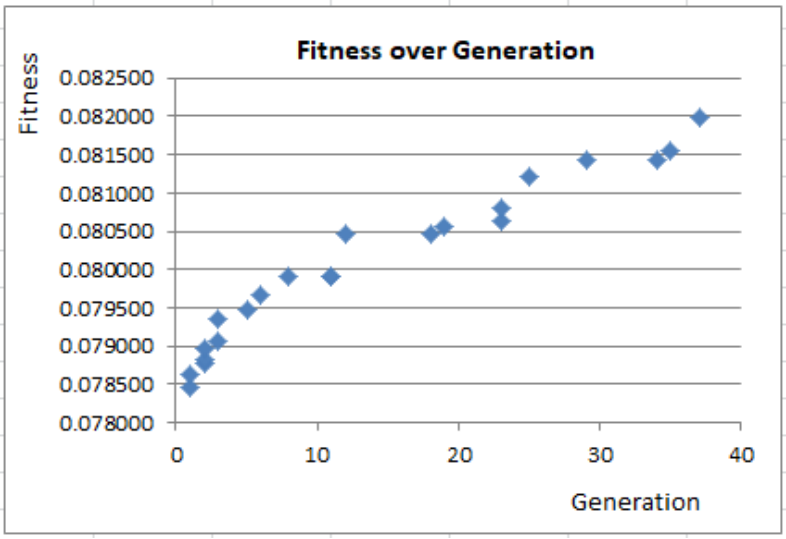
Detailed Results for Shipment 35

Gen No	Fitness
1	0.078287
1	0.078641
2	0.078666
2	0.078756
2	0.079076
2	0.079123
2	0.079208
5	0.079274
5	0.079532
6	0.079869
8	0.079882
10	0.079952
11	0.080026
13	0.080613
15	0.081077
16	0.081334
18	0.081368
18	0.081645
22	0.081722
23	0.081750
24	0.081963
29	0.082182
31	0.082360
32	0.082471
34	0.082495
38	0.082668
40	0.082681



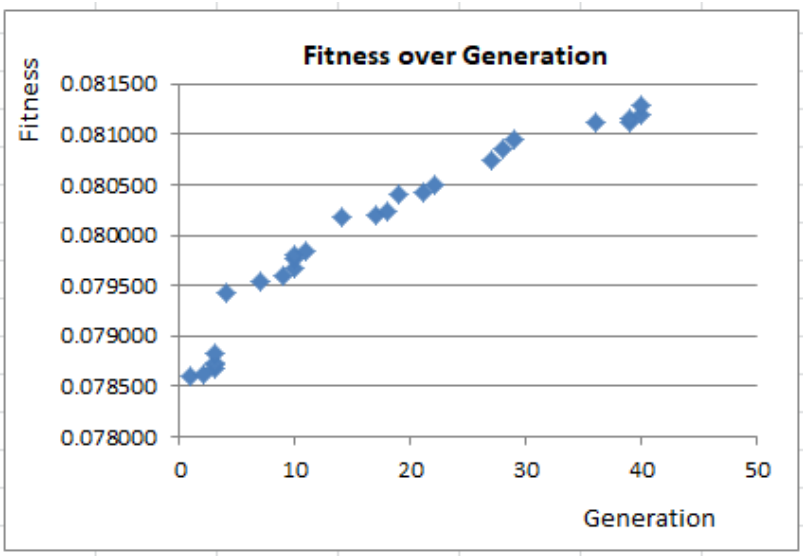
Detailed Results for Shipment 36

Gen No	Fitness
1	0.078468
1	0.078623
2	0.078777
2	0.078818
2	0.078970
3	0.079067
3	0.079365
5	0.079472
6	0.079659
8	0.079914
11	0.079917
11	0.079920
12	0.080463
18	0.080470
19	0.080554
23	0.080629
23	0.080811
25	0.081215
29	0.081427
34	0.081440
35	0.081556
37	0.081994



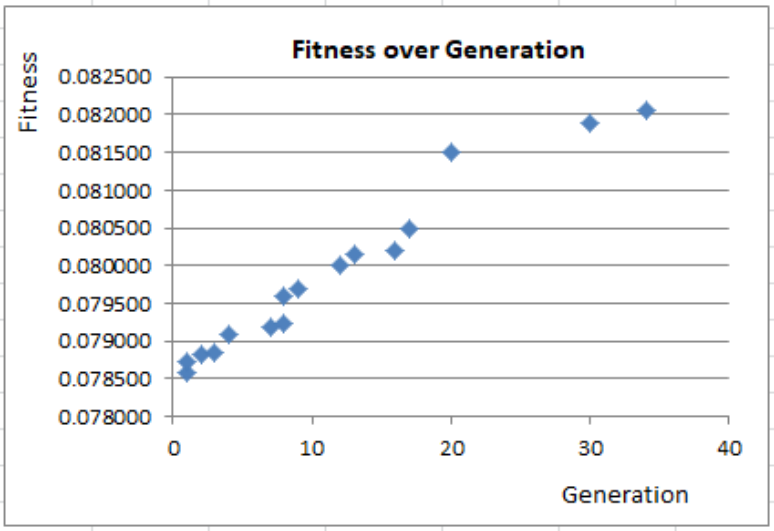
Detailed Results for Shipment 37

Gen No	Fitness
1	0.078604
2	0.078629
3	0.078672
3	0.078718
3	0.078743
3	0.078821
4	0.079434
7	0.079536
9	0.079596
10	0.079681
10	0.079770
10	0.079812
11	0.079837
14	0.080186
17	0.080199
18	0.080241
19	0.080402
21	0.080428
22	0.080509
27	0.080736
28	0.080860
29	0.080952
36	0.081119
39	0.081129
39	0.081159
40	0.081195
40	0.081281



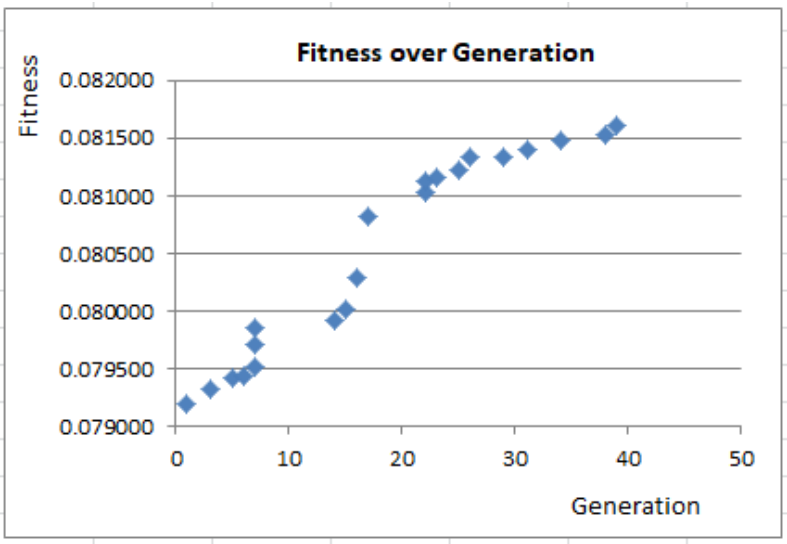
Detailed Results for Shipment 38

Gen No	Fitness
1	0.078570
1	0.078715
2	0.078812
3	0.078855
4	0.079095
7	0.079177
8	0.079242
8	0.079592
9	0.079697
12	0.079997
13	0.080157
16	0.080205
17	0.080486
20	0.081506
30	0.081887
34	0.082061



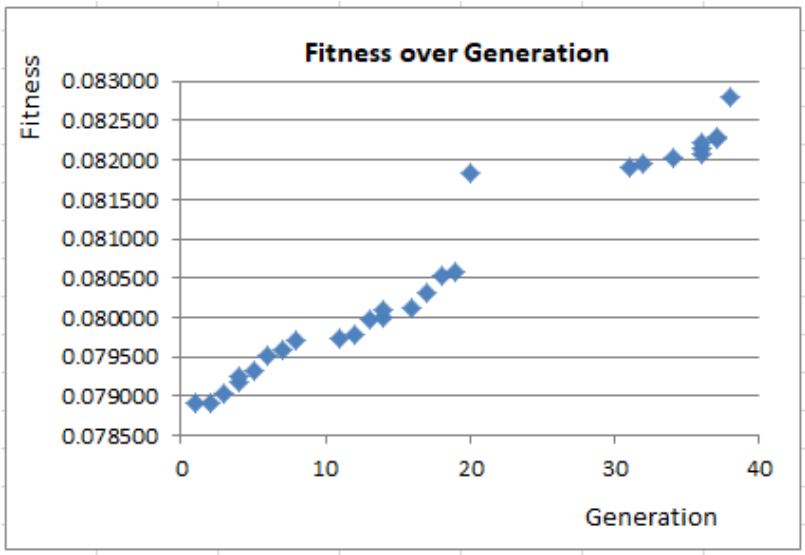
Detailed Results for Shipment 39

Gen No	Fitness
1	0.079195
3	0.079318
5	0.079412
6	0.079428
7	0.079523
7	0.079707
7	0.079850
14	0.079923
15	0.080013
16	0.080286
17	0.080824
22	0.081027
22	0.081121
23	0.081165
25	0.081218
26	0.081331
29	0.081334
31	0.081393
34	0.081484
38	0.081528
39	0.081609



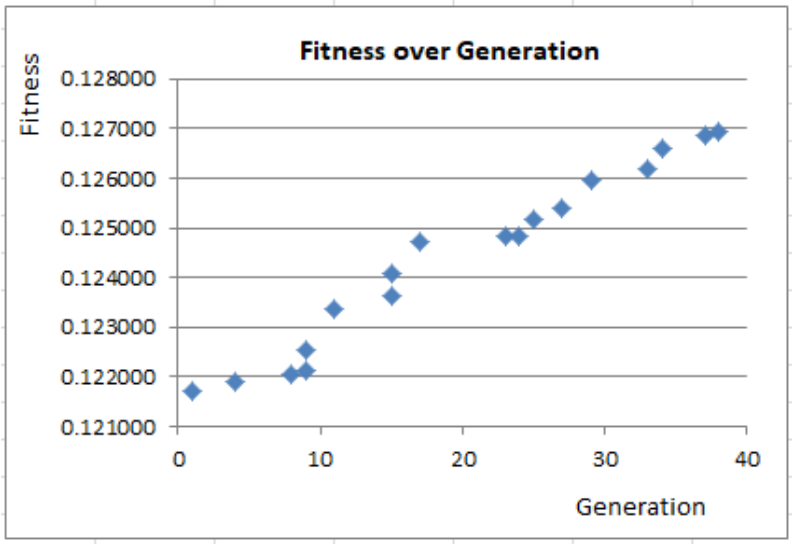
Detailed Results for Shipment 40

Gen No	Fitness
1	0.078902
2	0.078908
3	0.079042
4	0.079173
4	0.079246
5	0.079324
6	0.079523
7	0.079580
8	0.079700
11	0.079723
12	0.079793
13	0.079974
14	0.079987
14	0.080093
16	0.080122
17	0.080315
18	0.080525
19	0.080587
20	0.081833
31	0.081907
32	0.081947
34	0.082021
36	0.082088
36	0.082139
36	0.082230
37	0.082277
37	0.082305
38	0.082802



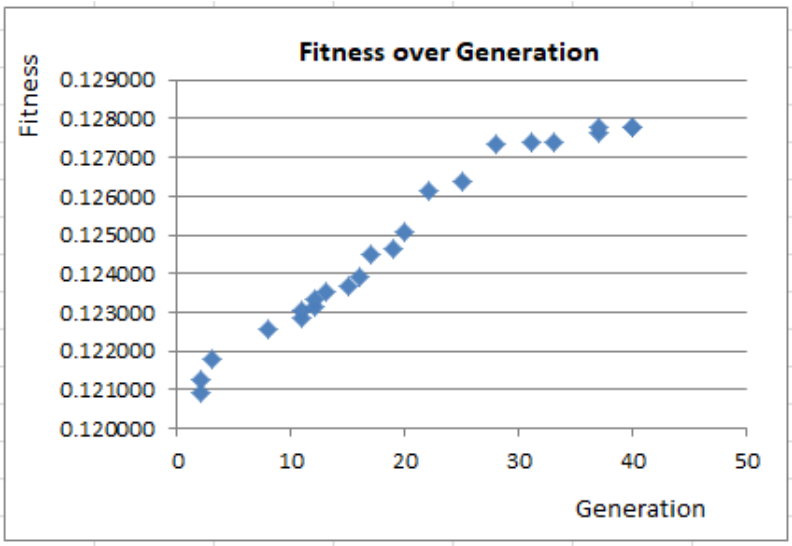
Detailed Results for Shipment 41

Gen No	Fitness
1	0.121729
4	0.121921
8	0.122041
9	0.122130
9	0.122557
11	0.123381
15	0.123625
15	0.124069
17	0.124727
23	0.124821
24	0.124836
25	0.125172
27	0.125384
29	0.125945
33	0.126175
34	0.126614
37	0.126847
38	0.126952



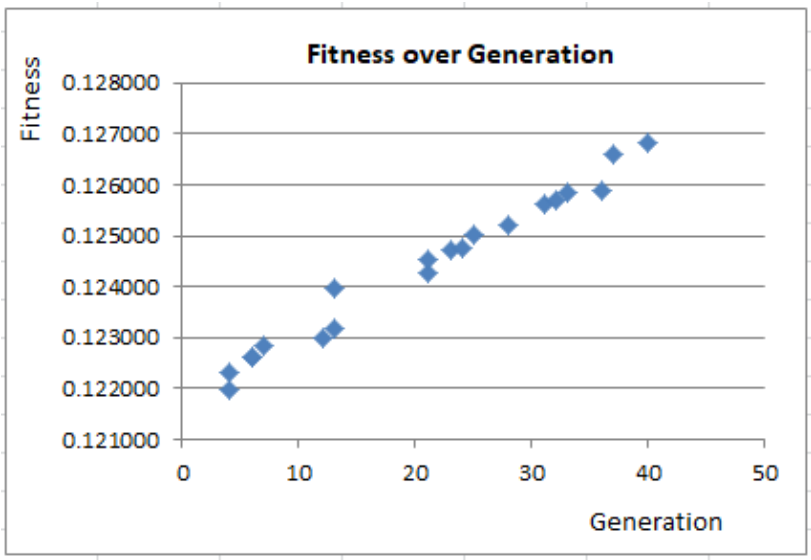
Detailed Results for Shipment 42

Gen No	Fitness
2	0.120941
2	0.121249
3	0.121810
8	0.122579
11	0.122865
11	0.123047
12	0.123145
12	0.123312
13	0.123518
15	0.123655
16	0.123916
17	0.124479
19	0.124649
20	0.125055
22	0.126135
25	0.126382
28	0.127340
31	0.127372
33	0.127380
37	0.127641
37	0.127763
40	0.127779
40	0.127796



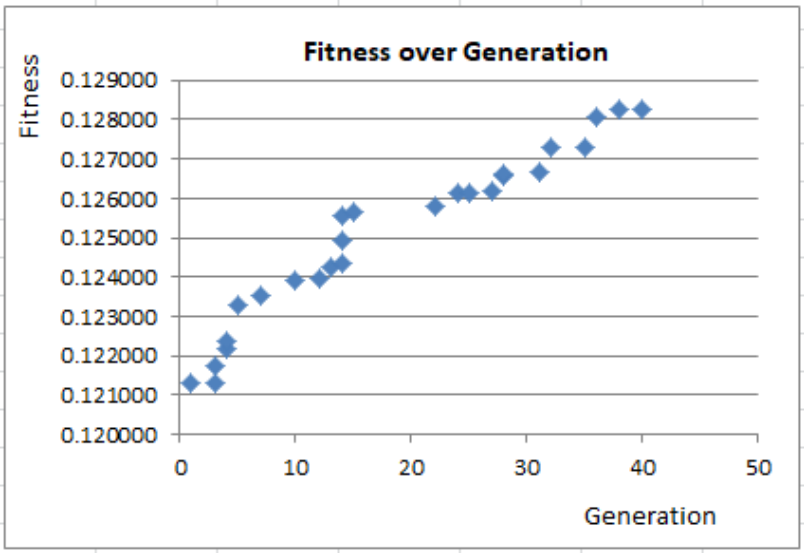
Detailed Results for Shipment 43

Gen No	Fitness
4	0.121996
4	0.122317
6	0.122617
6	0.122624
7	0.122827
12	0.122979
13	0.123198
13	0.123962
21	0.124262
21	0.124517
23	0.124719
24	0.124766
25	0.125031
28	0.125203
31	0.125612
32	0.125715
33	0.125849
36	0.125905
37	0.126590
40	0.126823



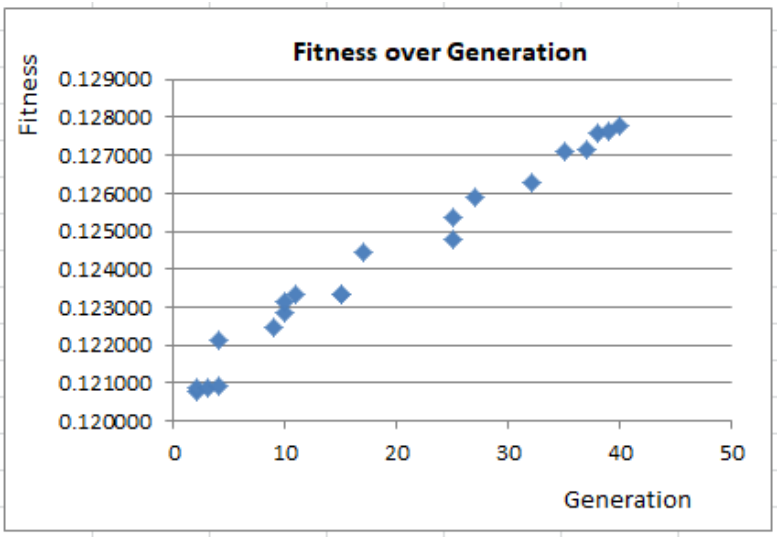
Detailed Results for Shipment 44

Gen No	Fitness
1	0.121315
3	0.121322
3	0.121721
4	0.122175
4	0.122347
5	0.123297
7	0.123518
10	0.123908
12	0.123954
13	0.124231
14	0.124370
14	0.124914
14	0.125557
15	0.125676
22	0.125778
24	0.126135
25	0.126159
27	0.126199
28	0.126550
28	0.126638
31	0.126678
32	0.127283
35	0.127307
36	0.128090
38	0.128279
40	0.128287



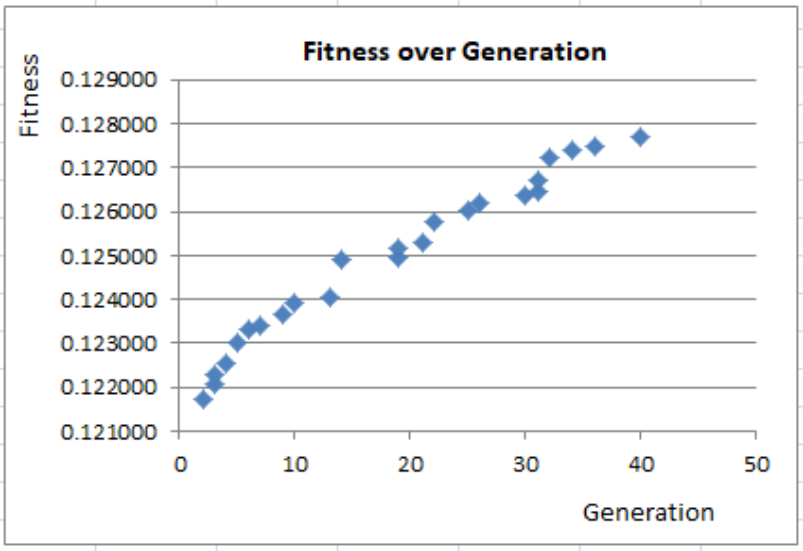
Detailed Results for Shipment 45

Gen No	Fitness
2	0.120780
2	0.120868
3	0.120882
4	0.120919
4	0.122108
9	0.122444
10	0.122843
10	0.123153
11	0.123320
15	0.123327
15	0.123358
17	0.124425
25	0.124789
25	0.125360
27	0.125913
32	0.126279
35	0.127121
37	0.127154
38	0.127575
39	0.127616
40	0.127787



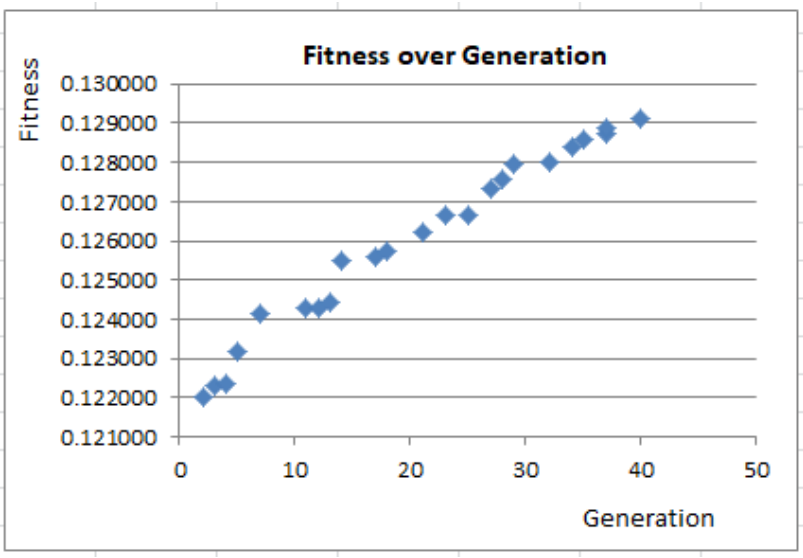
Detailed Results for Shipment 46

Gen No	Fitness
2	0.121729
3	0.122085
3	0.122302
4	0.122534
5	0.123001
6	0.123305
7	0.123419
9	0.123655
10	0.123923
13	0.124069
14	0.124891
19	0.124961
19	0.125188
21	0.125305
22	0.125755
25	0.126040
26	0.126215
30	0.126382
31	0.126454
31	0.126703
32	0.127218
34	0.127421
36	0.127478
40	0.127722



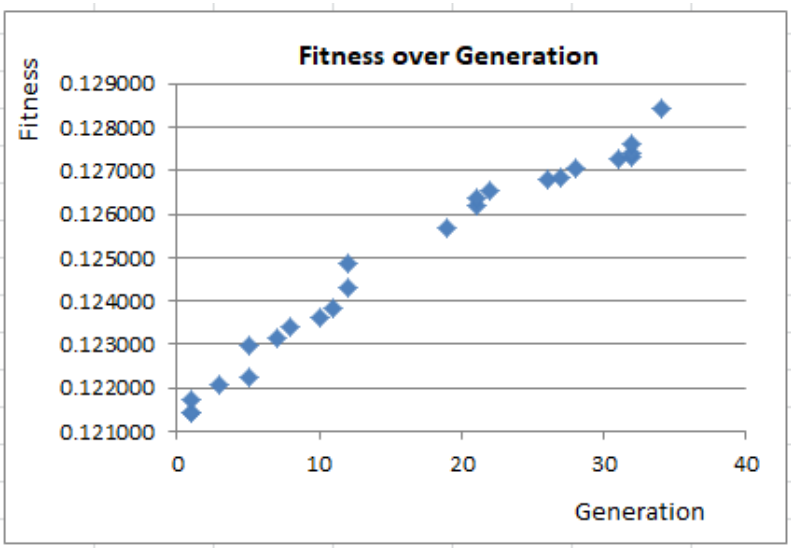
Detailed Results for Shipment 47

Gen No	Fitness
2	0.122018
3	0.122302
4	0.122339
5	0.123183
7	0.124131
11	0.124301
12	0.124309
13	0.124440
14	0.125510
17	0.125573
18	0.125715
21	0.126215
23	0.126638
25	0.126662
27	0.127348
28	0.127552
29	0.127957
32	0.127988
34	0.128397
35	0.128565
37	0.128714
37	0.128870
40	0.129123



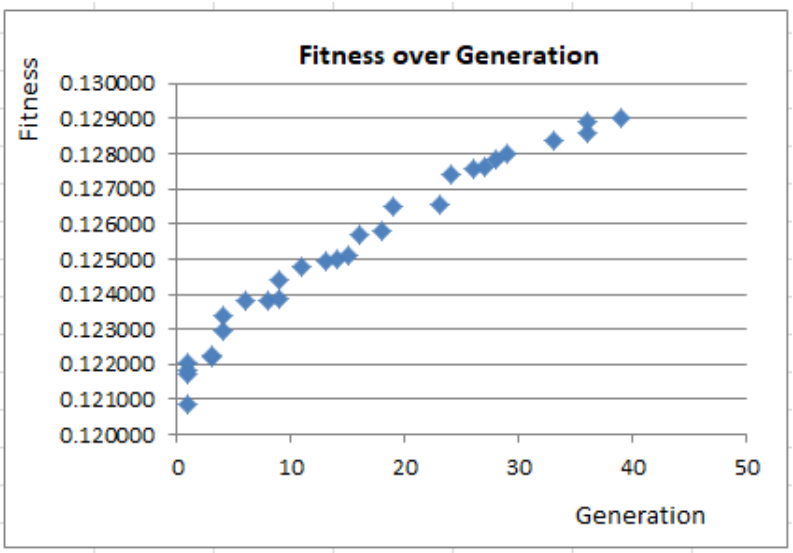
Detailed Results for Shipment 48

Gen No	Fitness
1	0.121411
1	0.121433
1	0.121736
3	0.122093
5	0.122249
5	0.122963
7	0.123153
8	0.123396
10	0.123609
11	0.123847
12	0.124293
12	0.124875
19	0.125683
21	0.126207
21	0.126374
22	0.126534
26	0.126791
27	0.126831
28	0.127057
31	0.127267
32	0.127307
32	0.127389
32	0.127616
34	0.128450



Detailed Results for Shipment 49

Gen No	Fitness
1	0.120839
1	0.121721
1	0.121847
1	0.122055
3	0.122212
3	0.122272
4	0.122979
4	0.123373
6	0.123816
8	0.123831
9	0.123877
9	0.124409
11	0.124797
13	0.124930
14	0.125008
15	0.125078
16	0.125707
18	0.125818
19	0.126486
23	0.126566
24	0.127405
26	0.127567
27	0.127610
28	0.127830
29	0.127985
33	0.128356
36	0.128614
36	0.128918
39	0.129011



Detailed Results for Shipment 50

Gen No	Fitness
1	0.121242
1	0.121529
1	0.121766
2	0.122302
2	0.122399
3	0.122858
5	0.122971
5	0.123145
8	0.123327
9	0.123602
10	0.123671
11	0.123923
12	0.124278
15	0.124285
15	0.125039
18	0.125266
20	0.125992
26	0.126088
28	0.126414
29	0.126759
30	0.126823
31	0.127230
33	0.127274
34	0.127600
36	0.127713
37	0.127747
40	0.127757